October 2019

Topology Network Optimization of Facility Planning and Design Problems

Ravi Ratan Raj Monga
University of Massachusetts Amherst

Topology Network Optimization of Facility Planning and Design Problems

A Thesis Presented

By

RAVI RATAN RAJ MONGA

Submitted to the Graduate School of the

University of Massachusetts Amherst in partial fulfillment

of the requirements for the degree of

Master of Science in Mechanical Engineering

September 2019

Mechanical Engineering

Topology Network Optimization of Facility Planning and Design Problems

A Thesis Presented

By

RAVI RATAN RAJ MONGA

Approved as to style and content by:

_____

J. MacGregor Smith, Chair

_____

Hari Balasubramanian, Member

_____

Xian Du, Member

_____

Sundar Krishnamurthy, Department Head

Mechanical and Industrial Engineering

DEDICATION

To my patient and loving family.

ACKNOWLEDGMENTS

I would like to thank my advisor, J. MacGregor Smith, for his many months of thoughtful, patient guidance and support. I would also like to extend my gratitude to the members of my committee, Hari Balasubramanian and Xian Du, for their helpful comments and suggestions on all stages of this thesis. Together their guidance and selfless contribution to my professional development have been invaluable and will forever be appreciated. A token of gratitude to Professor David M. Warme for his precious time in assistance with the third phase of research.

I wish to express my appreciation to all the individuals who helped me in this thesis. A special thanks to Sri Deepika for her efforts in calibrating the C compiler for research. Thank you to Shifali Bansal for her tireless efforts in reviewing the many versions of this manuscript. Thanks, are also due to Vijeta Deshpande. A special thank you to all those whose support and friendship helped me to stay focused on this project and who have provided me with the encouragement to continue when the going got tough.

ABSTRACT

TOPOLOGY NETWORK OPTIMIZATION OF FACILITY PLANNING AND DESIGN PROBLEMS

SEPTEMBER 2019

RAVI RATAN RAJ MONGA, B.TECH, SRM UNIVERSITY

M.S.M.E, UNIVERSITY OF MASSACHUSETTS AMHERST

Directed By: Professor J. Macgregor Smith

The research attempts to provide a graphical theory-based approach to solve facility layout problem. Which has generally been approached using Quadratic Assignment Problem (QAP) in past, an algebraic method. It is a very complex problem and is part of the NP-Hard optimization class, because of the nonlinear quadratic objective function and (0,1) binary variables. The research is divided into three phases which together provide an optimal facility layout, block plan solution consisting of MHS (material handling solution) projected onto the block plan. In phase one, we solve for the position of departments in a facility based on flow and utility factor (weightage for location).The position of all the departments is identified on the vertices of MPG (maximal planar graph), which maximizes the possibility of flow. We use named MPG produced in literature, throughout the research. The grouping of department is achieved through GMAFLAD, a QSP (quadratic set packing) based optimizer. In Phase 2 ,the dual for the MPG's is solved consisting of department location as per phase 1, to generate Voronoi graphs. These graphs are then, expanded by an ingenious parameter optimization formulation to achieve area fitting for individual cases. Optimization modeling software, Lingo17.0 is used for solving the parameter optimization for generating coordinates of block plan. The plotting of coordinates for the block plan graphics is done via Autodesk inventor 2019. In phase 3, solution for MHS is achieved using an RSMT (Rectilinear Steiner minimal tree) graph approach. The Voronoi seed coordinates produced through phase 2 results are computed by GeoSteiner package to generated the RSMT graph for projection onto the block plan (Also, done by Inventor 2019). The graphical method employed in this research, itself has complex and NP- hard problem segmentes in it, which have been relaxed to polynomial time complexity by fragmenting into groups and solving them in sections. Solving for MPG & RSMT are a class of NP- Hard problem, which have been restrict to N=32 here. Finally, to validate the research and its methodology a real-life case study of a shipyard building for the data set of PDVSA, Venezuela is performed and verified.

TABLE OF CONTENTS

LIST OF TABLES

Table Page

LIST OF FIGURES

Figure Page

LIST OF SYMBOLS

n = The number of departments in a layout.

c_{ik} = Linear placement cost term for locating activity I in location k.

f_{ij} = The flow of material between activity i and activity j, where $i, j = 1, 2, \ldots \ldots . . n$.

d_{kh} = The distance between location k and location h.

x_{ik} = 1 if activity i is assigned to location K and is 0 otherwise.

d_{ij} = The distance between department i and j, where $i, j = 1, 2, \ldots \ldots . . n$.

x_{kt} denotes the t^{th} cluster of cells to which the kth activity can be assigned, i.e.

x_{kt} is 1 if the kth activity is assigned to the cluster of cells designated by t and x_{kt} is 0 if otherwise.

α_{ikt} is 1 if the ith cell is a member of the t^{th} cluster for activity kth, and $let\ a_{ikt} = 0$ otherwise.

A is a set of planar arcs indicating critical relationships between activity pairs x_k and x_j for each alternative (x_{km}, x_{jn}).

$d_{mn}$is the rectilinear distance between cluster m for activity k and cluster n for activity j.

u_{kt} is an expected utility-of-place coefficient for the t^{th} cluster of activity x_k.

u_{kj} is an expected utility of flows coefficient between activities x_k and x_j.

a_p is the area of the rectangle formed by x_i and y_j.

CHAPTER 1
INTRODUCTION

Commercial, manufacturing, industrial, utility & residential facilities are tangible spheres which are foundation of our modern world. They all have facility layout and design problems, which forms the key component of overall operations, both in terms of maximizing the effectiveness of the process and meeting the needs of humans. Facilities planning is majorly concerned with the design, layout, location, and accommodation of people, machines, and activities of a system or enterprise (both manufacturing, services, concept and even circuit boards) within a physical spatial environment. People, machines, vehicles, and processes are accommodated within the physical environment so that the objective of the system or enterprise (e.g. hospital, bank, manufacturing, telecommunications call center, malls, warehousing etc.) housed within the facility can be satisfactorily achieved [1].

The objective of all these layout plans leads to ensure smooth flow of work, material, and information through a system. Which quantifies to space where business's corresponding activities take place. The layout and design of that space greatly impact how the work is done—the flow of work, materials, and information through the system. The key to good facility layout and design is the integration of the needs of people (personnel and customers), materials (raw, finishes, and in the process), and machinery in such a way that they create a single, well-functioning system. [2] These facility & service systems themselves require an efficient network systems & flows, which must operate in a perfect symphony between all the sections within them. To achieve this intrepid goal, engineering principles are required to be created and implemented.

The facility design process is usually complex and requires significant amount of information before even starting. Taking the case of manufacturing, critical information including

the types of products to be manufactured, quantities to be produced, specific manufacturing processes, the sequence of manufacturing operations to be performed, markedly affects the product and process design. We have tried to achieve this in our current research by a systematic engineering approach that encompasses 3 main phases, whose independent but mutually exclusive evolution leads to a block plan diagrams for all types of facilities. Where the systematic procedure begins by constructing the maximal planar graph (MPG) based on facility factors such as size, number of sections, weightage factor etc. for the floor plan graph (FPG) layout/diagrams, which are finally incorporated with a material handling systems (MHS) solution.

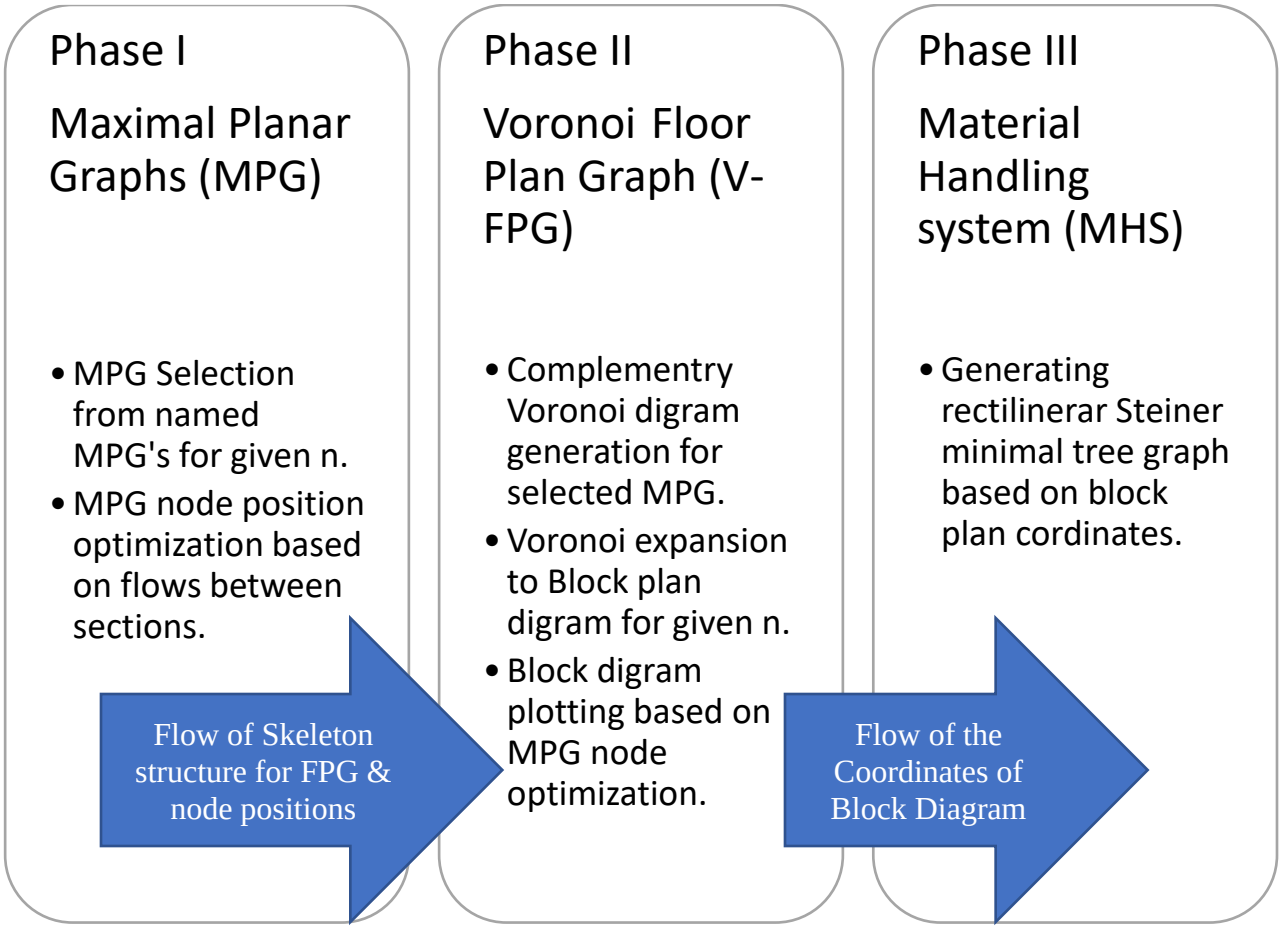


Figure 1 A succinct illustration of the method of the research for a facility layout solution.

In the first phase, we develop an optimal GMAFLAD (graphical multi-attribute facility layout and design) solution for given facility system comprising n nodes on an MPG. In the second phase, we solve for a dual graph layout solution, for the FPG using the Delaunay triangulation and generate & expand the complemery Voronoi graph. Finally, for the last phase we generate an MHS solution through a rectilinear Steiner minimal tree (RSMT) graph. The RSMT generates a minimum spanning tree pattern that is superimposed on the block diagram to represent optimum material handling path (Figure 1).The core of this work is to establish a paragon to solve for facility layout problem for creating FPG through graphical methods at large and implementing them as a whole.

This work is revealed through 6 chapters after the introduction, with chapter 2 being literature review, Chapter 3 discussing the phase 1 of the research, where we solve for grouping of departments on MPG vertices . Chapter 4 detailing the transformation of the graphs generated in phase 1, into block plan digram through solving for their dual (Voronoi graph) and expanding them through parameter optimization. Chapter 5 representing phase 3, where we tackle for MHS solution through projecting RSMT graph results. Chapter 6 incorporates a case study taken from an originally study performed for designing layout of a shipyard in Venezuela. Which has real data, utilized here, to validate and verify the method and capability of current research to produce FPG solution. Finally, in Chapter 7 we conclude our research and its outcomes, discussing the various assumptions and short comings. We also discuss the possibility expanding the work and creating an interactive application capable of solving and producing graphical results with numerical value for all 3 phases of research. [3]

CHAPTER 2
LITERATURE REVIEW

There are several approaches, which have been carried out in the past. A good survey of the general problem is reviewed in the Ph.D. thesis of Watson [4]. Heuristics for the MPG problem are discussed in the paper by Dyer et. al. [5]. One of the most recent approaches is to use a graph topology which is already planar, rather than generate the topology from first principles. More recently Pesch et. al. [6] presented an approach to generate the MPG using the deltahedron graph as a basis. Watson also discusses the problem of generating the floor plan dual graph (FPG) and the graph topology of the MHS. For which an effective method is to generate layout using the MPG that achieves required optimality. Generating an MPG for given number of department or "n" is an excellent & efficient way to determine layout configuration.

The FLP (facility layout planning) and location theory, in general, have been well studied since the 1960s. The FLP involves how to place n blocks of departments (e.g., machine tools, manufacturing cells, work centers, storage units, departments, etc.) with given area and shape requirements in a facility without overlap, to minimize the total material handling cost, which is often expressed as the material flow volumes weighted by the distances among departments. FLP play important roles in efficient operations of the modern manufacturing systems, and the layout of a facility is known to have a significant impact on manufacturing costs, work in process (WIP), lead times, productivity, etc. various types of FLPs had been extensively studied in the literature since Koopmans and Beckmann (1957) first modeled the problem as a quadratic assignment problem (QAP), which aims to find the optimal assignment of n departments to n predetermined locations to minimize the total material handling cost. The QAP is NP-complete (Sahni &

Gonzalez, 1976), prior to this exact branch and bound algorithm for QAP can be found in Lawler (1963) and Bazaraa and Elshafei (1979).

The quadratic assignment problem is generalization of the well-known linear-integer assignment problem, where integer variables make the problem nonlinear . In the model, each facility is assigned exactly to one location without considering any interaction between facilities. This interaction, in facilities location, is usually represented in terms of number of trips between facilities. The quadratic assignment problem allows the inclusion of the interaction, in addition to the consideration of distances between locations. The general form of QAP model given below differs from the standard form because it relates more closely to the software model of G-MAFLAD, which we are using. [1]

There are 2 terms in objective function: (1) a linear placement cost term and (2) a flow placement term, the flow term is nonlinear because of the integer variables. The linear placement cost term comes into picture only when fixed activities occur in the layout.

$$Maximize = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ik}x_{ik} + \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{h=1}^{n} f_{ij}d_{kh}x_{ik}x_{jh}$$

Subject to,

$$\sum_{i=1}^{n} x_{ik} = 1, \qquad k = 1,2,\ldots\ldots\ldots\ldots,n$$

$$\sum_{k=1}^{n} x_{ik} = 1, \qquad i = 1,2,\ldots\ldots\ldots\ldots,n$$

$$x_{ik} = 0,1 \qquad i,k = 1,\ldots\ldots,n$$

Where, c_{ik} = Linear placement cost term for locating activity i in location k

f_{ij} = The flow of material between activity i and activity j

d_{kh} = The distance between location k and location h

$x_{ik} = 1$ if activity i is assigned to location k and is 0 otherwise

The quadratic assignment problem (QAP) of assigning discrete entities to discrete locations is one of the most well-known and difficult combinatorial optimization problems in operations research. The unequal-area FLP represents a more complex optimization problem than the QAP. It is possible to adapt the QAP to the unequal-area FLP by breaking departments into small grids with equal areas and by not allowing the separation of grids of the same department by assigning large artificial flows between them. An unequal area FLP formulated as a QAP substantially increases the number of decision variables needed, so that even solving such a problem presents a significant challenge. To date, attempts to model the unequal-area FLP using exact optimization methods, such as mixed integer programming (MIP), have been limited due to the complexity of the problem. Montreuil (1990) [7] provides a MIP that has become somewhat of a baseline for extension, improvement, and comparison. The generalized quadratic assignment problem (GQAP) studies a class of problems that optimally assign M facilities to N locations subject to the resource limitation at each location. These problems arise naturally in yard management, where containers are to be in the storage areas with limited capacity, and in distributed computing where processing tasks are to be assigned to processors with limited computing resources. The GQAP is a generalization of the QAP, with the difference that multiple facilities can now be assigned to a single location subject to resource capacity at locations [8].

Branch and bound (BB, B&B, or BnB) is an algorithm design paradigm for discrete and combinatorial optimization problems, as well as mathematical optimization. A branch-and-bound algorithm consists of a systematic enumeration of candidate solutions by means of state space search: the set of candidate solutions is thought of as forming a rooted tree with the full set at the root. The algorithm explores branches of this tree, which represent subsets of the solution set.

Before enumerating the candidate solutions of a branch, the branch is checked against upper and lower estimated bounds on the optimal solution and is discarded if it cannot produce a better solution than the best one found so far by the algorithm. The algorithm depends on efficient estimation of the lower and upper bounds of regions/branches of the search space. If no bounds are available, the algorithm degenerates to an exhaustive search [9].

Levin in 1964 [10] first proposed using graph theory to model facility layout development with planar and dual graphs. Krejcirik in 1969 [10] developed one of the first computerized procedure of this kind, called RUGUR. It required a relationship diagram input. Since then, many other authors have developed procedure; Foulds discuss elaborately of alternative graph theoretic based approaches [11]. The deltahedron heuristic was developed by Foulds and Robinson in 1978 [12] and many subsequent variations. It began with four vertices connected to form a complete graph, then adds additional vertices, one at a time, within the graph into triangular faces to variation of the deltahedron heuristic.

MAFLAD (multi-attribute facility layout and design) has a long history. It is an optimality seeking branch-and-bound algorithm embedded in it for solving the QAP. The program interacts with the user, allowing the user to request a graphical display of the solution and to select one of the three heuristics in the branch and bound process. It can display the solution both graphically and numerically depending on the user's choice. The further development of this program from legacy language Fortran to C-sharp based programing with graphical user interface (GUI) was done by Patrick Simon as part of his MSc. degree at university of Massachusetts Amherst [1]. Smith and MacLeod in 1988 [13] showed that a Langrangian relaxation of the QSP results in relaxed assignment problem that can be solved efficiently with an exact branch-bound algorithm embedded in MAFLAD.

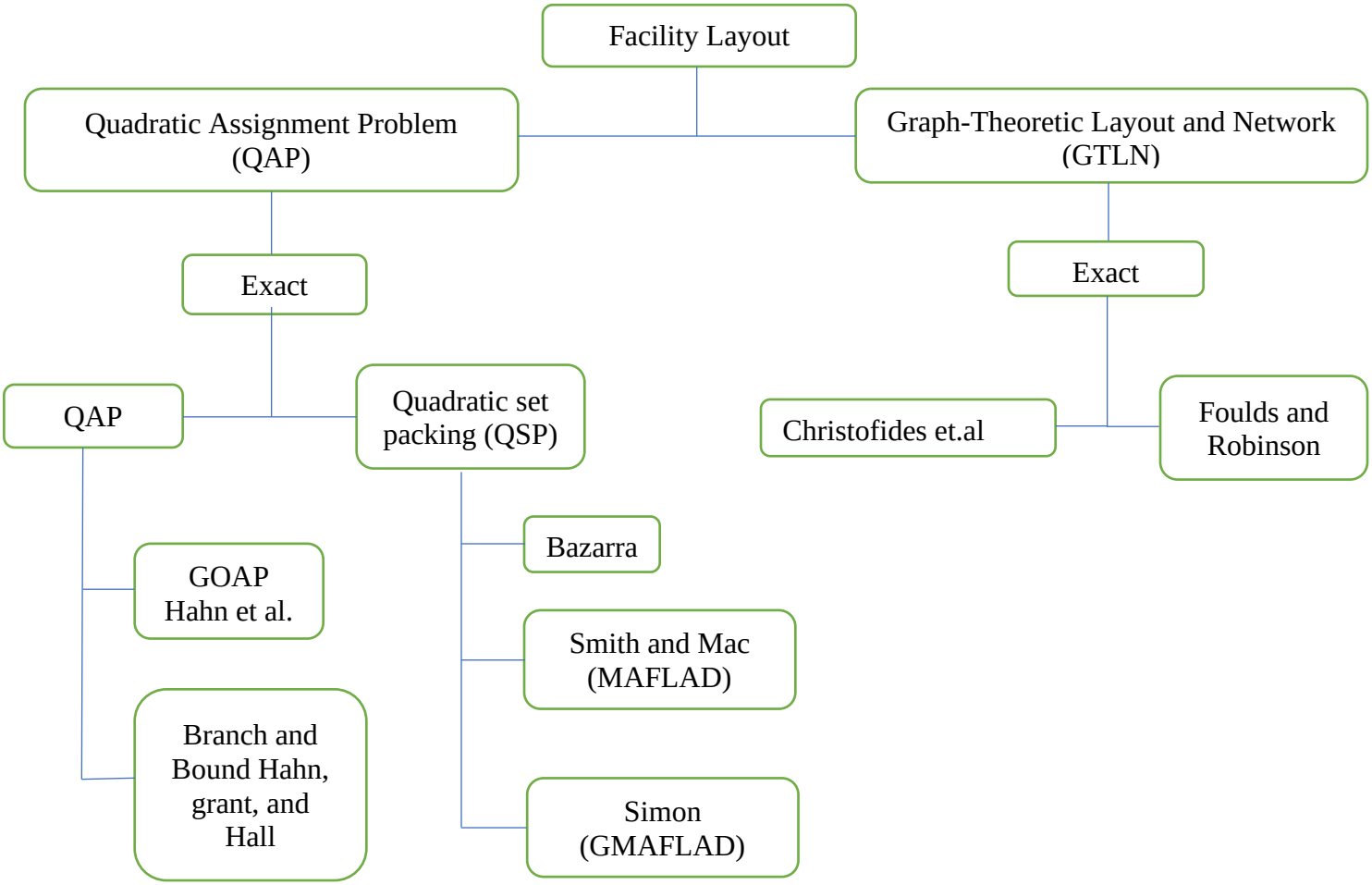


Figure 2 The classification of exact layout problem [8]

Armour and Buffa (1963) formulated the block layout problem in which a set of n departments with pre-specified dimensions (heights and widths) are arranged in a continuous rectangular region without overlapping. The block layout with unequal departmental areas (fixed or flexible side lengths) is called the unequal area facility layout problem (UA-FLP). Montreuil (1990) used a surrogate linear constraint to approximate the departmental area by the side lengths such that a rectangular department can have a flexible shape (shape constraint, e.g., the aspect-ratio or minimum side-length constraints) to meet the specified area requirement [7]. Many significant research publications involving normative approaches to solving location problems

from an operations research perspective are outlined in a bibliography by Francis and Goldstein (1974) This list includes much of the innovative and pioneering work that was done in the 1960s and 1970s for the FLP. Two more recent literature surveys on the FLP are done by Kusiak and Heragu (1987) [14] and Meller and Gau (1996) [15]. Meller and Gau discuss recent trends in facility layout research. They highlight a tendency towards concurrent design strategies, which incorporate design of a facility layout with design of a production system. In addition, a comparison is made between the state-of-the-art in facility layout software and the state-of-the-art in facility layout research [15]. The unequal-area FLP was originally formulated by Armour and Buffa (1963) [16]. The authors assume there to be a given fixed rectangular region, or facility, of dimensions H and W, where H is the height and W is the width. The number of departments, the area of each department, and the flow values associated with each pair of departments are assumed to be known. The authors provide a computer-aided heuristic algorithm that alleviated the need to consider all possible permutations of a layout and made the cumbersome methodology of using qualitative judgment to select several promising location patterns much less desirable. Their algorithm considered piecewise departmental exchanges in which adjacent departments of equal area could be relocated, or interchanged, if an improvement to the objective function was realized. Their methodology, formally known as CRAFT (computerized relative allocation of facilities technique), can be classified as an improvement algorithm. A variation of the quadratic assignment problem (QAP), that was modified to accommodate the FLP, was formulated and solved by Bazaraa (1975) [17]. The same rectangular layout from Armour and Buffa [16] is broken down into blocks of equal-area and uniform shape.

The authors report optimal solutions for FLPs with up to eight departments. Sherali et al. [18] provide a similar approach that significantly reduces errors in department areas by using a

polyhedral outer approximation of the area constraints and branching priorities. Using the polyhedral approximation and other innovative techniques, the authors report solutions for FLPs with up to nine departments. A new MIP formulation for the unequal-area FLP using the Flexbay structure is presented by Konak et al. [19]. Contrary to the work of Meller et al. [20] and Sherali et al. [18], the nonlinear department area constraints are modeled on a continuous plane without using any surrogate constraints and without utilizing linear relaxation techniques. This improvement permits the use of the Flexbay formulation in the form of a MIP. This is the first MIP in which area constraints are enforced 100% for the unequal-area FLP. The authors argue that although Flexbay formulations restrict possible layout combinations, it forms the basis of an aisle structure that facilitates the user transferring the block design into an actual facility design. Several extensions to this formulation are suggested, such as tightening the lower bound of the linear program, removing the mirror effect, modeling the number of bays as a decision variable, and modeling monuments or fixed regions within departments. A mirror effect occurs when decision variables assume different values but produce layouts that are identical in terms of interdepartmental distances and total cost. Restricting a department's centroid to a quarter of a layout eliminates the mirror effect and three fourths of all layout combinations. Modeling the number of bays as a decision variable was attempted, but the formulation was inefficient. Modeling fixed monuments within a department limits the shape and location changes that are possible for that department. A monument can be thought of as a predefined, fixed rectangle that acts to constrain a department's potential location. For example, a large broach could be a monument in a machining department. The machining department can be relocated, but its new location must still contain the broach. Although the MIP papers provide substantial improvements to the number

of departments that can be solved to optimality, each work acknowledges computational limitations of using MIP for the FLP.

As mentioned earlier, most facility layout solution strategies can generically be identified as either constructive heuristics or improvement heuristics. The same generalization can be made regarding the QAP. Due to the intractability of the QAP for any but small problems (n = 5 to 20 departments maximum), heuristic techniques have generally been preferable to the exact methods discussed in the previous section. The tradeoff between gains to CPU efficiency and degradations to solution quality using heuristic procedures for the QAP has been shown to be minimal (1981) [21]. The advantages and disadvantages of constructive and improvement procedures being utilized separately, sequentially, or cohesively, have been thoroughly discussed [21] and continue to be an active area of research [22]. Liggett [21] provides a computational analysis of utilizing constructive and improvement techniques on several well-known test problems. The differences between improvements to a random solution through pair-wise exchange versus constructed solutions are highlighted. Liggett concludes that constructive procedures are preferable since better solutions can be obtained at less cost. It should be noted that computers today are much faster and make it easier to use improvement algorithms. The author suggests that attempts to reduce the number of exchanges may alter the preference towards constructive procedures. Selecting the exchange that leads to the maximum cost improvement rather than selecting the first exchange that leads to improvement was not shown to be worth the addition in computational time. The results of Liggett [21] and Arapoglu et al. [22] indicate that improvements to solution quality and computational feasibility using heuristic procedures do not have to come at the expense of one or the other. In addition, the question of preference between constructive and improvement schemes is answered to some degree. More importantly, the potential for improvement using constructive

and improvement strategies in some manner of cohesion becomes a distinct possibility. The use of a cut tree methodology in the context of the construction and improvement of a facility layout was introduced by Montreuil and Ratliff (1989) [23]. The authors propose the use of a design skeleton because it has several desirable properties. If a designer wishes to place emphasis on the separation of certain departments, versus focusing strictly on departmental adjacency, a cut tree permits the partitioning of departments into two subsets and indicates the optimum partition. The links on a cut tree indicate average material movement. If the skeleton is used as the aisle structure, the designer is provided with valuable insight about how to increase or decrease aisle lengths. Furthermore, if an aisle structure is made up of segments of uniform length, then a cut tree will provide the aisle structure that minimizes total flow the authors point to an advantage of the cut tree methodology when used for multiple floor layout planning. They also mention that a layout designer's goal is ultimately to generate a satisfactory design rather than a theoretical optimum. In such a context, a cut tree provides an efficient and effective means of enhancing layout design. However, giving a layout designer the decision-making power to manually select cuts in a design skeleton could potentially lead to a satisfactory layout design that is considerably deviant from its optimal solution.

The issue of unreliable and frequently changing flow values between departments in a manufacturing environment, caused by product demand variability, is addressed by Kim and Klein [24]. A methodology for designing a layout based on the metric of the shortest path along aisles or corridors (SPAAC) was introduced by Benson and Foote [25]. A constructive heuristic, known as door-fast, is used to optimize the aisle structure and door placement once the department locations and a general aisle structure have been determined. By using the SPAAC metric, the solutions produced by door-fast are much more interpretable than results that use centroid to

centroid distance measures or other metrics that do not provide a clear indication of a material flow network. Chittratanawat and Noble [25] put forth an integrated facility layout methodology that incorporates department location, qualitative relationships, I/O location, and material handling equipment selection. The authors assume equal area departments of identical shape and that the flow of material within a department is adjustable to the selection of an I/O station. A construction algorithm generates an initial layout and a Tabu Search algorithm improves the initial solutions. The inclusion of material handling equipment selection is justified as a means of ensuring the best overall material handling costs for a manufacturing layout. If the selection of material handling equipment were not included, a best layout configuration might not correlate to the lowest material handling costs. The initial construction of a layout is determined to be a critical factor in the solution quality of the Tabu Search. Their approach performs markedly better with constructed initial solutions than with those that are generated randomly. Also, the methodology does not ensure diversity in the search space because Tabu moves are not recorded during the search process.

An integrated layout methodology specifically designed for semiconductor fabrication facilities is given by Peters and Yang [26]. The authors mention the proclivity of semiconductor manufacturers to adopt bay structured layouts that are not always the most efficient. A bay structure is accommodating to this industry since material movement is generally done through automated material handling systems that are designed as a either a spine configuration through some central location of a layout or as a perimeter configuration around a layout. Spine configurations restrict the number of departments in each bay to one to ensure interaction with a material handling system, whereas perimeter configurations mandate two departments within a bay to achieve the same functionality. These restrictions aid in optimizing both the block layout

and the material handling system. Using space filling curves to construct an initial material handling configuration, multiple layouts with different flow sequences can be generated for further consideration. Once an initial layout and flow sequence have been constructed, the location of I/O points and the determination of crossover points are made. Crossover points indicate positions on a material handling system at which flow can reverse direction by crossing over to the parallel path of the system. A flow network is constructed from the candidate I/O points and candidate crossover points. A steepest-descent-pairwise-interchange heuristic improves initial solutions. If a simple spine or perimeter configuration is not enough and the restrictions on bay structure are alleviated, space filling curves might aid in limiting the number of potential layouts for consideration. This would be especially useful if the material handling system under consideration was very expensive [25].

More precise methods of the linearization approximation of departmental areas for the UA-FLP can be found in Lacksonen (1997), Meller, Narayanan, and Vance (1999), Sherali, Fraticelli, and Meller (2003), and Castillo and Westerlund (2005). Most of the existing models of block layout design in a rectangular facility, the centroid to centroid (CTC) rectilinear distance is used in the calculation of the material handling cost, often as the surrogate distance of material movements between departments. However, there are many industrial practices in which hybrid/different distance metrics, such as Euclidean and Tchebychev, may be more appropriate than single rectilinear in terms of representing the actual travel distances of materials between manufacturing cells, such as conveyor belts, rail trolley, monorails, or overhead cranes. Other distance metrics such as the Tchebychev distance, which measures the maximum horizontal or vertical distance between departments, may also be more appropriate for the case of material handling through overhead cranes in heavy industries. O'Muirgheasa, Kadipasaoglu, and

Khumawala (2001) compared the rectilinear and Euclidean distance metrics to the actual distances in their experiments, and they showed that the objective function value of the Euclidean distance is closer to the actual distances than that of the rectilinear distance. Norman, Aapoglu, and Smith (2001) introduced the contour distance metrics into the detailed FLP with input/output (I/O) station designs, which measures the transport between I/O stations along the perimeters of departmental boundaries. Ozdemir, Smith, and Norman (2003) incorporated heterogeneous distance metrics, including euclidean, contour, rectilinear, and Tchebychev within the block layout design with flexible bay construct (which cuts the facility in one direction first), and they used a pre-defined binary matrix to allow departmental pairs being specified with different distance measures. However, due to the non-linear characteristics of the Euclidean metrics and the complexity in calculating the contour distance, their model is non-linear and difficult to be optimally solved. Hale, Huq, & Hipkin, 2012 used an expected distance function, defined as the probabilistic expectation of distance metric of interest (rectilinear, euclidean, etc.), instead of centroid-to-centroid distance to evaluate the order of placement of departments in the facility layout construction.

Niroomand & Vizvári(2013) summarized four types of material handling path used for detailed layout designs in literature, including: (1) the spine—departments are located on either side of a single direct line/path, (2) the circular (closed loop)—the path is a rectangle and all I/O stations are on the edges, (3) the ladder—several vertical and horizontal direct lines that serve as material handling paths, and (4) open field—the path is a shortest polyline between departments with Manhattan/rectilinear distances. Garcia-Hernandez, PalomoRomero, Salas-Morera, Arauzo-Azofra, & Pierreval, 2015 used only the Squared Euclidean distance as the metric in the unequal area facility layout problem (UA-FLP) and developed a hybrid evolutionary algorithm that

combines genetic algorithm, clustering method and niching techniques. Paes, Pessoa, and Vidal (2017) studied the UA-FLP problem in an unlimited floor space, and used one of the three distance metrics: Rectilinear, squared Euclidean, and Euclidean to measured CTC distance.

Machine learning has intimate ties to optimization, several problems are formulated as minimization of some loss function on a set of examples. Loss functions express the discrepancy between the predictions of the model being trained and the actual problem instances (for example, in classification, one wants to assign a label to instances, and models are trained to correctly predict the pre-assigned labels of a set of examples). The difference between the two fields arises from the goal of generalization, while optimization algorithms can minimize the loss on a training set, machine learning is concerned with minimizing the loss on unseen samples. [27]

In this paper, we formulate an intrinsic optimization method to transform the non-linear constraint of Euclidean distance into a set of variables solvable in polynomial time for N<32. This Euclidean distance approximation method guarantees that the deviation between the actual and approximate distances will be non-consequential. In addition, we develop a set of non-linear constraints to restrict the relative positions of department pairs for which interdepartmental distances are assumed arbitrary. Computational experiments are carried out to demonstrate the accuracy and computational efficiency of the Voronoi diagram. Further, MHS solution is incorporated using exact solution RSMT graphs for block plan, all represented in a case study.

CHAPTER 3
MAXIMAL PLANAR GRAPHS

According to graph theory, a planar graph can be drawn on the plane in such a way that its edges intersect only at their endpoints not along the axis. A simple graph is called maximal planar if it is planar but adding any edge (on the given vertex set) destroys that property. An MPG (figure 1) consists of n vertices, with n > 2, having precisely 3n − 6 edges and 2n − 4 faces. [2] There are various methods to generate them ranging from Apollonia, triangulation, strangulation etc.

The basic formulation is given by

n = the number of departments
f_{ij} = the total flow between departments i and j, where $i,j = 1,2,\ldots,n$
d_{ij} = the distance between departments i and j, where $i,j = 1,2,\ldots,n$

The objective function for minimizing the total product flow throughout a facility is given by,

$$Total\ Cost = \sum_{i=1}^{n} \sum_{\substack{j=1 \\ i \neq j}}^{n} f_{ij} d_{ij}$$

The goal is to partition the region into departmental sub-regions to minimize the total material movement and associated costs of the entire facility. The combinatorial nature of this formulation makes the consideration of all possible layouts virtually impossible.

Math world by Wolfram Alpha has catalogued all the named triangulated graphs, which is another way to designate MPG. It has graph listed from simple N=3 to N=32 and then some special purpose N=62 and 341(Moore's Graph) generated throughout various algorithm for other purposes

(It has the latest N=59 MPG, not listed). The different methods used for generating them, the reasons, specific outcome and various logic involved are beyond the scope of this research and have been omitted. The diagrams of the triangulated graph utilized have been depicted in Appendix-I [28].

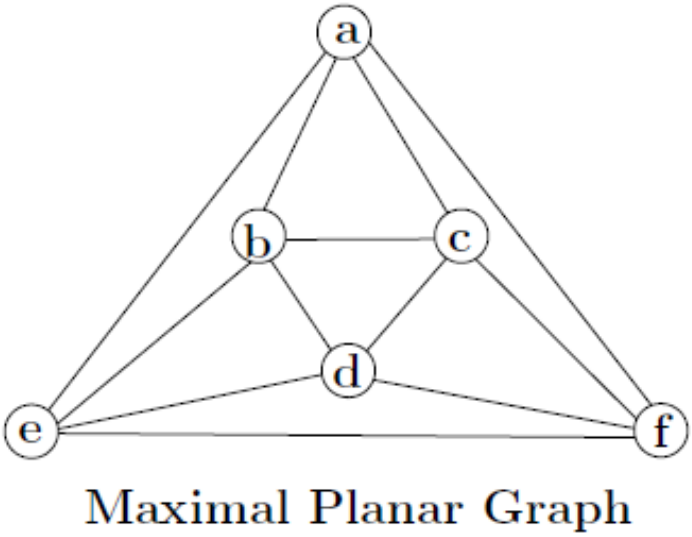


Figure 3 Example of Maximal planar Graph [8]

There is an issue with MPG possibilities, there are several for certain given value of n and there are none for other values of n. For the ones that don't have the MPG triangulation possible by any methodology, we will be observing the planar graph that approaches most closely to equation $3n - 6$ edges and $2n - 4$ faces and inserting an extra node, where feasible. For other n which has multiple possibility we take inspiration from Kelvin Watson's thesis, which shows templates up till n=8 in its appendix, generated by Bowen & Fisk technique. [5]He designates these templates using vertex degree numbering, where the base value gives number of interconnections for a vertex with other vertices and the superscript gives the number of vertices possessing similar order of interconnection. The formulation conjured is designated as index

formulation, it is of type $K^{U1}L^{U2}M^{U3}$ …n ,Where K, L, M…n denotes the interconnection for given vertices and U1, U2, U3…n denotes the number of vertices possessing similar order of interconnections. [4] Through this nomenclature it is observed that for all possible templates generated by Bowen & Fisk [5] for given number of nodes, intuitively multiplying all the bases with the raise to power of subscript gives distinct & ordered value, with highest order value having the maximum possible interconnection($K^{U1}*L^{U2}*M^{U3}$ …*n). This type of enumeration and sorting enables us to cherry pic the best possible MPG for given number of nodes.

After the MPG or Planar Graph for given value of n has been identified, we work on finding the location of each room or section based on the flow, weight factor and any other location attribute. For this purpose, we utilize the G-MAFLAD (graphical multi-attribute facility layout design) a computer program that was created by Smith and Macleod in 1988 and later developed with graphical interface by Patrick Simon, 2005 [13]. It solves problem of physically organizing several entities within given facility to optimality by means of underlying branch and bound algorithm. It is based on relaxed model of the QSP (Quadratic Set Packing) formulation [29].

which is,

$$Maximize\ \ Z = \sum_k \sum_t u_{kt}\, x_{kt} + \sum_k \sum_j u_{kj} \left(\sum_{mn \epsilon A_{kj}} \frac{1}{d_{mn}} x_{km} x_{jn}\right)$$

subject to,

$$\sum_k \sum_t \alpha_{ikt} x_{kt} \leq 1 \qquad \forall i = 1, \ldots, I \text{ cells}$$

$$\sum_t x_{kt} = 1 \qquad \forall k = 1, \ldots, K \text{ activities}$$

$$x_{kt} \epsilon \{0;1\} \qquad \forall k = 1, \ldots K \qquad \text{and} \qquad t = 1, \ldots, T$$

With,

x_{kt} denotes the t^{th} cluster of cells to which the k[th] activity can be assigned, i.e.:

x_{kt} is 1 if the k[th] activity is assigned to the cluster of cells designated by t and x_{kt} is 0 if otherwise.

α_{ikt} is 1 if the i[th] cell is a member of the t^{th} cluster for activity k[th], and $let\ a_{ikt} = 0$ otherwise.

A is a set of planar arcs indicating critical relationships between activity pairs x_k and x_j for each alternative (x_{km}, x_{jn}).

$d_{mn}$is the rectilinear distance between cluster m for activity k and cluster n for activity j.

u_{kt} is an expected utility-of-place coefficient for the t^{th} cluster of activity x_k.

u_{kj} is an expected utility of flows coefficient between activities x_k and x_j.

The index t in the binary decision variables of the linear term of the objective function, i.e. x_{kt} and u_{kt}, indicates the t^{th} alternate cluster for activity k and its associated utility value within the grid layout.

In the above formulation, activities are assigned to the layout as feasible combinations of cells within the grid (Smith and MacLeod [13]).

Based on the above algorithm through G-MAFLAD, we input the MPG's for given value of n and adding possibility of placing department at any vertices, though it can be done selectively, and weightage addition could also be done selectively for any section. We utilize greedy heuristic for all cases throughout the experiment, from 3 other heuristics available. The other 2 heuristics available are best future value heuristic and limited lookahead heuristic but have not been used due to its efficiency and output of greedy heuristic. To run the experiment, few of the flow matrix values have been taken from QAPLIB - a quadratic assignment problem library [30] but mostly have been generated through benchmark-Kate, which is another software compiled on Fortran and generates QAP flow matrices. The graphical GMFLAD result with numerical output for the

greatest value of objective function of all named MPG with their respective input flow matrices (generated using benchmark Kate or QAPLIB) have been listen in Appendix I

Phase I - example case

We will be discussing in detail an example for value of n =14. To select MPG for the input value of n= 14, we will use math-world by Wolfram-Alpha data base on tabulated graphs under MPG & planar graph. There are 2 named MPG present in the database for value of n = 14, namely small triakis octahedral graph (figure 4) and tetrakis hexahedral graph (figure 5). Out of these 2 graphs, we will first select the graph that has maximum number of interconnections possible. We will use index formulation to decide, developed in the chapter above.

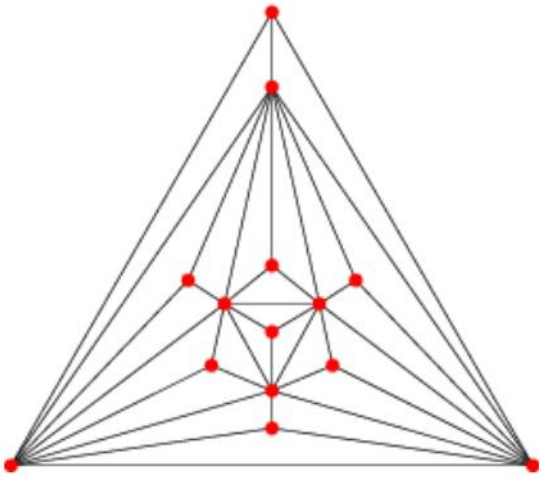


Figure 4 N= 14 MPG, Small Triakis Octahedral Graph

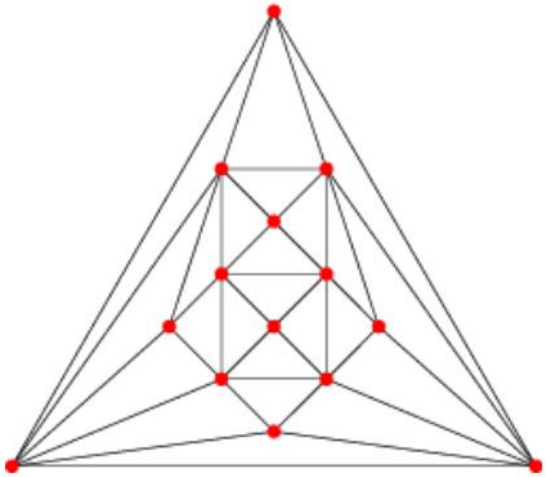


Figure 5 N= 14 MPG, Tetrakis Hexahedral Graph

To find the number of interconnection possible for each of the graph, we formulate based on the number of vertex and degree of freedom at each vertex, as depicted in Table 1. Subsequently, we make our selection based on maximum integer value obtained for given template in observed n. Based on calculations from table 1, tetrakis hexahedral graph has higher integer value. Also, if we observe the figures 4 and 5, it is graphically visible in Euclidean space that tetrakis hexahedral

graph is more regular to small triakis octahedral graph. Hence, we procced with tetrakis hexahedral graph as out MPG selection for n=14.

Table 1 Index formulation for N = 14 MPG cases and section.

Sr. No.	Graph Type	Template Multiplier	Enumeration
1	Small Triakis Octahedral Graph	$3^8 * 8^6$	1,719,926,784
2	Tetrakis Hexahedral Graph	$6^8 * 4^6$	6,879,707,136

Moving on to step 2 of phase I, we solve for vertices of MPG, in G-MAFLAD. We generate a 23 by 23 coordinate system by inserting rows and columns into G-MAFLAD and plotting the tetrakis hexahedral graph into it (Figure 6). G-MAFLAD allows us to put alternates and utility value for each activity or department inserted. To create the function in conceptual context with

Table 2 Flow Matrix generated by using Benchmark Kate for N=14 Problem

Activities	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	-	0	0	0	47	0	0	0	0	0	0	0	0	0
2	0	-	0	0	0	84	0	0	0	0	0	0	0	0
3	0	0	-	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	-	0	0	0	0	0	0	6	0	0	0
5	47	0	0	0	-	0	0	0	0	0	0	0	0	0
6	0	84	0	0	0	-	0	0	0	0	0	8	0	0
7	0	0	0	0	0	0	-	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	-	88	0	0	0	0	0
9	0	0	0	0	0	0	0	0	-		0	32	0	0
10	0	0	0	0	0	0	0	88	0	-	0	0	44	0
11	0	0	0	6	0	0	0	0	0	44	-	4	0	0
12	0	0	0	0	0	8	0	0	32	0	4	-	0	27
13	0	0	0	0	0	0	0	0	0	0	0	0	-	36
14	0	0	0	0	0	0	0	0	0	0	0	27	36	-

around the system integration, we allow all the activities to be able to be placed at every vertex on MPG and omit any utility value in this experimental example. For considering the flows or movement between departments we have used benchmark Kate QALP package, that randomly

generates the flow for input value of n, variables and range (table 2). We have inserted these flows through edit for flow matrix into G-MAFLAD and run the program for Greedy's heuristic with graphical and numerical output, figure 7 represents the numerical outcome with alternatives.

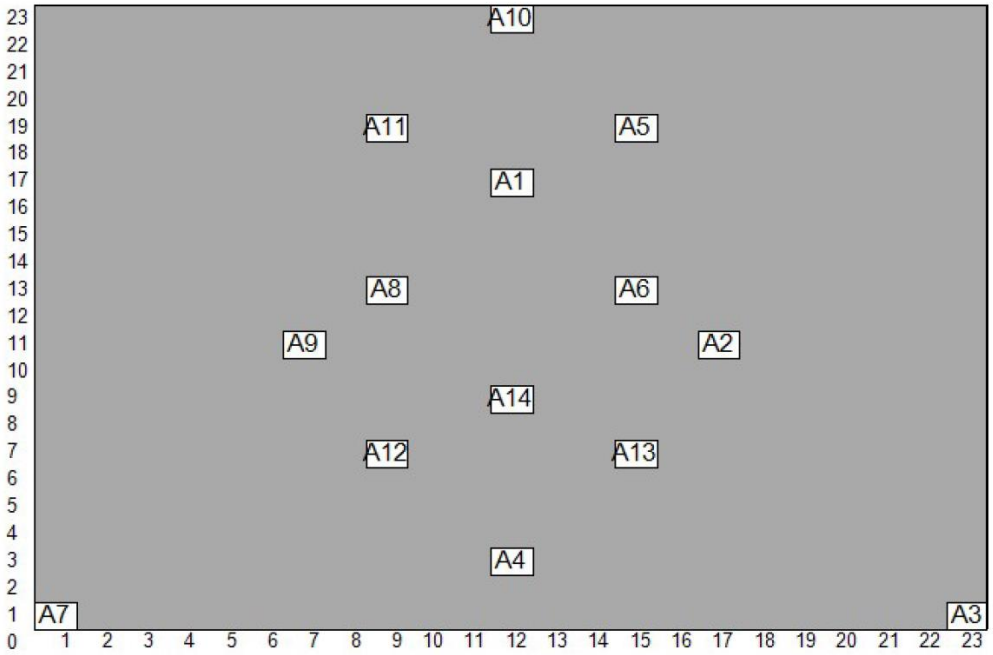


Figure 6 Tetrakis Hexahedral Graph (N=14) maximum value MPG layout for given flow matrices.

```
THE NAME OF THE DATA FILE IS rav11.dat
CPU TIME USED IN OPTIMIATION IS     0.0000 SEC.
THE STORAGE USED IN OPTIMIATION IS   13709
The value of the solution is    108.87
Activity      Alternate
   1              6
   2             14
   3              2
   4             12
   5              5
   6              8
   7              1
   8              7
   9             13
  10              3
  11              4
  12             10
  13             11
  14              9
```

Figure 7 The best value Numerical Output for the N=14 GMAFLAD solution.

CHAPTER 4
VORONOI DIAGRAM PLAN DEVELOPMENT

A Voronoi diagram is a partitioning of a plane into regions based on distance to points, in a specific subset of the plane. That set of points (called seeds, sites, or generators) is specified beforehand, and for each seed there is a corresponding region consisting of all points closer to that seed than to any other. These regions are called Voronoi cells. The Voronoi diagram of a set of points is a dual graph to its Delaunay triangulation.

Duality translates to concepts, theorems or mathematical structures into other concepts, theorems or structures, in a one-to-one fashion, often (but not always) by means of an involution operation: if the dual of A is B, then the dual of B is A. Such involutions sometimes have fixed points, such that the dual of A is A itself. [31]

In computational geometry, a Delaunay triangulation for a given set P of discrete points in a plane is a triangulation DT(P) such that no point in P is inside the circumcircle of any triangle in DT(P). Delaunay triangulations maximize the minimum angle of all the angles of the triangles in the triangulation; they tend to avoid sliver triangles (a triangle with one or two extremely acute angles, hence a long/thin shape, which has undesirable properties during some interpolation or rasterization processes) [32]. The triangulation is named after Boris Delaunay for his work on this topic from 1934. [33] The Delaunay triangulation of a discrete point set up in general position corresponds to the dual graph of the Voronoi diagram for P. Special cases include the existence of three points on a line and four points on circle.

Voronoi is named after Georgy Voronoi, and is also called a Voronoi tessellation, a Voronoi decomposition, a Voronoi partition, or a Dirichlet tessellation (after Peter Gustav Lejeune Dirichlet). They are also known as Thiessen polygons.

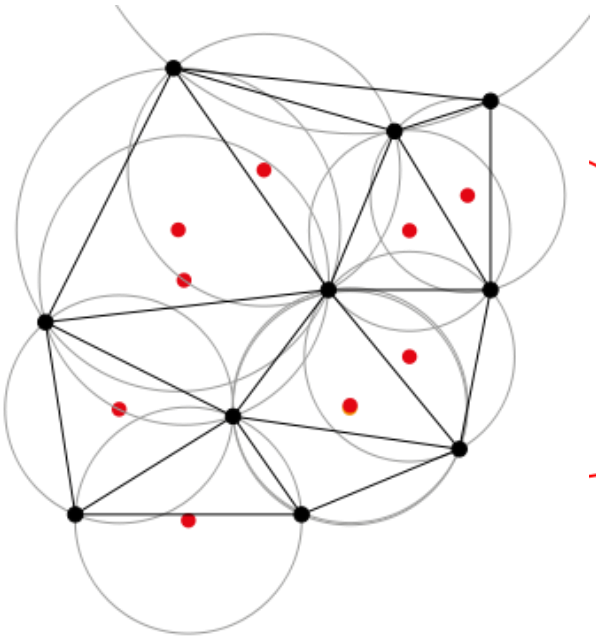


Figure 8 The Delaunay triangulation with all the circumcircles and their centers (in red) in the plane shown

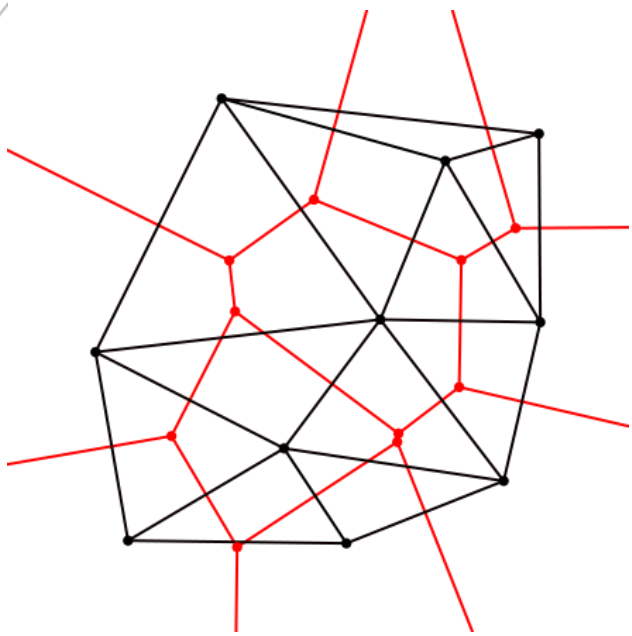


Figure 9 Connecting the centers of the circumcircles produces the Voronoi diagram

Let X be a metric space with distance function d. Let K be a set of indices and let $(P_k)_{k\epsilon K}$ be a tuple (ordered collection) of nonempty subsets (the sites) in the space X. The Voronoi cell, or Voronoi region, R_K, associated with the site P_k is the set of all points in X whose distance to P_k is not greater than their distance to the other sites P_j, where j is any index different from k. In other words, if d (x, A) = inf {d (x, a)|a ϵ A} denotes the distance between the point x and the subset A,

Then,

$$R_k = \{x \in X \mid d(x, P_k) \leq d(x, P_j) \text{ for all } j \neq k\}$$

The Voronoi diagram is simply the tuple of cells $(R_k)_{k\epsilon K}$. In principle, some of the sites can intersect and even coincide (an application is described below for sites representing shops),

but usually they are assumed to be disjoint. In addition, infinitely many sites are allowed in the definition (this setting has applications in geometry of numbers and crystallography), but again, in many cases only finitely many sites are considered. [34]

In the case where the space is a finite dimensional Euclidean space, each site is a point, there are finitely many points and all of them are different, then the Voronoi cells are convex polytopes and they can be represented in a combinatorial way using their vertices, sides, 2-dimensional faces, etc. Sometimes the induced combinatorial structure is referred to as the Voronoi diagram. However, in general the Voronoi cells may not be convex or even connected.

In the usual Euclidean space, we can rewrite the formal definition in usual terms. Each Voronoi polygon R_k is associated with a generator point P_k. Let X be the set of all points in the Euclidean space. Let P_1 be a point that generates its Voronoi region R_1, P_2 that generates R_2, and P_3 that generates R_3, and so on. Then, as expressed by Tran et al [35] "all locations in the Voronoi polygon are closer to the generator point of that polygon than any other generator point in the Voronoi diagram in Euclidean plane". [34]

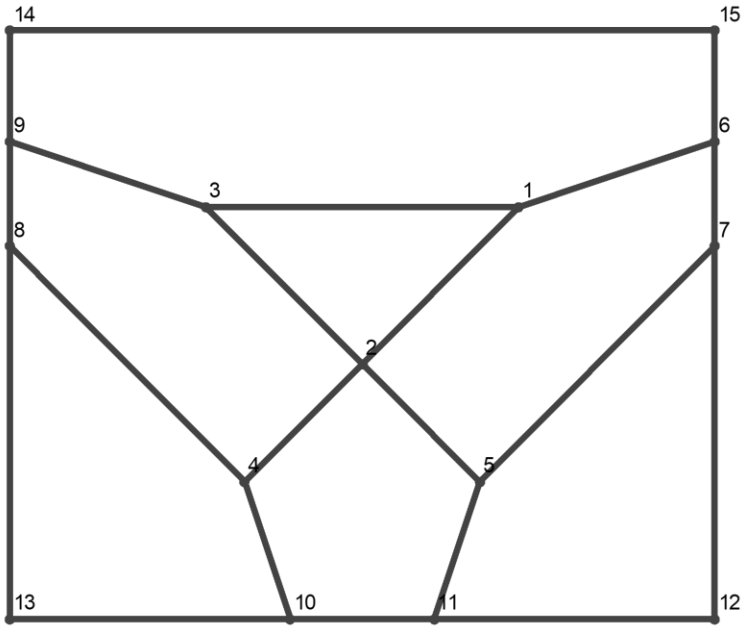


Figure 10 N=7 Voronoi diagram for Heptahedral 34 MPG.

We have used Mathematica 11.3 and it's in built computational geometry packages to generate Voronoi diagram for our named MPG and Planar graphs identified in previous section. The center of the circumcircle that produces Voronoi from converse of Delaunay triangulation have been translated based on the one produced by G-MAFLAD, for best possible layout for given N & flow matrix. These form the center of each N identified in previous section through G-MAFLAD solution. Which moves on to become the final location of each department identified by the interconnection selection and underlying branch and bound algorithm attributed to QSP modeling. Figure 10 gives the Voronoi triangulated diagram for the heptahedral MPG. All the Voronoi triangulations created for named MPG have been listed in chronological order in Appendix-II.

To generate block diagram from Voronoi diagram, we have utilized NLP (nonlinear programming) optimization tool LINGO17.0 by LINDO (linear, interactive, and discrete optimizer), a software package for nonlinear, integer programming and global optimization. The following formulation has been developed based on parameter optimization for given rectangles for n number of sections and compartments to generate plottable coordinates for our final block plan solution.

Here,

We have Voronoi polygons (p =1, ...P) and we wish to minimize the perimeter of the layout diagram subject to the area requirements for each cell activity:

$$\text{Minimize } Z = \sum_{i=1}^{n}\sum_{j=1}^{n} 2((x_i - x_{i-k}) + (y_j - y_{j-l})) \qquad \forall k \in i, l \in j$$

subject to,

$$(x_1) * (y_1) = a_1 \qquad \forall i = 1,2,3.....n$$

$$(x_2 - x_1) * (y_2 - y_1) = a_2 \qquad \forall j = 1,2,3.....n$$

$$(x_i - x_{i-k}) * (y_j - y_{j-l}) \geq a_p \qquad \forall p = 1,2,3.....n$$

$$x_i \geq 0\,; \qquad \forall i = 1,2,3\ldots..n$$

$$y_j \geq 0\,; \qquad \forall j = 1,2,3\ldots..n$$

Here,

x_i is the x coordinate of the department

y_j is the x coordinate of the department

a_p is the area of the rectangle of the polygon formed by x_i and y_j

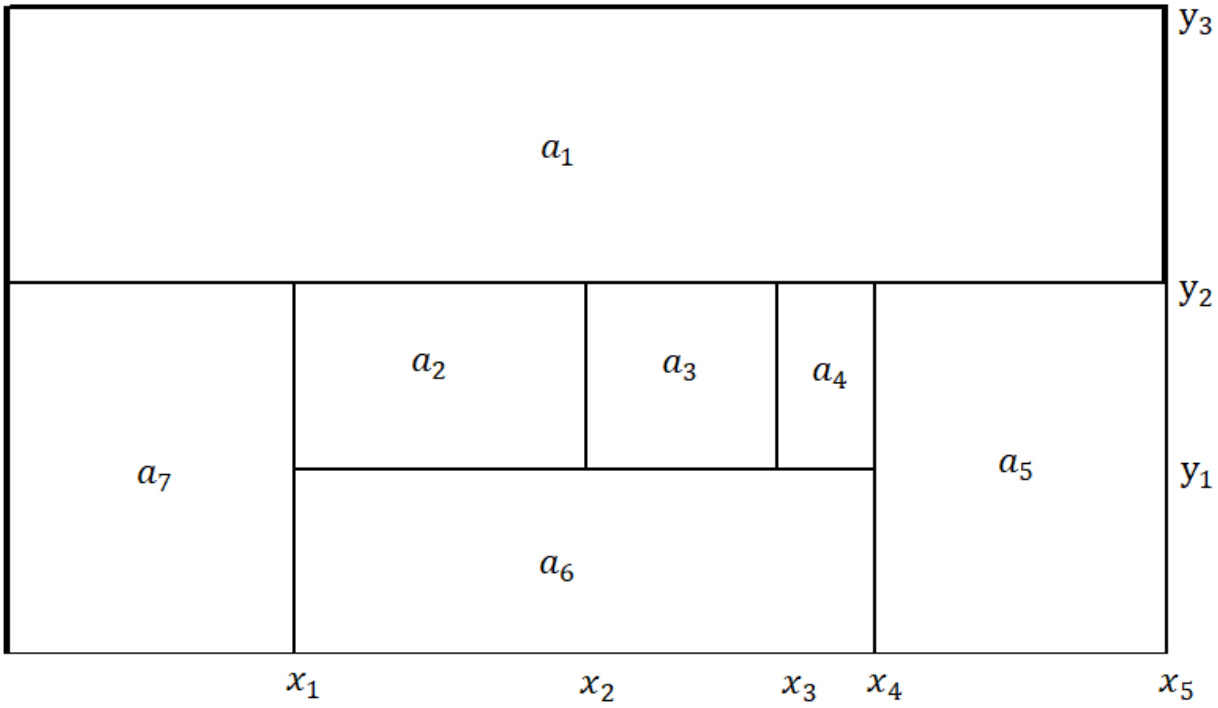


Figure 11 For N=7, Heptahedral 34 block diagram conversion of Voronoi triangulation given in figure 10

Figure 11 represents the Block plan plotted output for N=7 (heptahedral) MPG, based on following mathematical expansion of formulation discussed above.

Minimize $Z = 2 * [(x_1 + y_1) + (x_4 - x_1 + y_1) + (x_5 - x_4 + y_2) + (x_2 - x_1 + y_2 - y_1) + (x_3 - x_2 + y_2 - y_1) + (x_4 - x_3 + y_2 - y_1) + (x_5 + y_3 - y_2)$

Subject to,

$$(x_1) * (y_1) = a_1$$

$$(x_2 - x_1) * (y_2 - y_1) = a_2$$

$$(x_3 - x_2) * (y_2 - y_1) = a_3$$

$$(x_4 - x_3) * (y_2 - y_1) = a_4$$

$$(x_5 - x_4) * (y_2) = a_5$$

$$(x_4 - x_1) * (y_1) = a_6$$

$$(x_5) * (y_3 - y_2) = a_7$$

$$x_1, x_2, x_3, x_4, x_5 \geq 0 ;$$

$$y_1, y_2, y_3 \geq 0 ;$$

Where, we have inputted following areas-

$a_1 = 500$, $a_2 = 250$, $a_3 = 250$, $a_4 = 250$, $a_5 = 500$, $a_6 = 760$, $a_7 = 1000$

Similar to the above expansion of the formulation, we have expanded the equation for each Voronoi diagram created for given value of n until value of n=32, currently only named MPG Voronoi have been created and an NLP (non-linear program) optimization has been performed to obtain coordinates for block diagram, based on user defined input value of area. The above formulation is coded in LINGO17.0 with its constraints and user defined areas consisting of random selected values as input. As the facility size grows, the user must be meticulous in designating area values as the output solution could become unstable due to imaginary values, producing infeasible region with unbounded objective function value. The output thus obtained is may be utilized to plot a block plan diagram on a rectilinear as well as Euclidian system. The block plan diagram, thus generated for each value of n preserves the vertex location, generated during phase 1 with Greedy's heuristic. It also preserves, all the utility value (weightage), flows and other

conditional attributes. We have depicted the block plan output for all named MPG, based on the above sequence of steps with coordinate-based plotting of NLP optimization in Appendix-II

Sensitivity analysis

Sensitivity analysis is the study of how the uncertainty in the output of a mathematical model or system (numerical or otherwise) can be apportioned to different sources of uncertainty in its inputs. The process of recalculating outcomes under alternative assumptions to determine the impact of a variable under sensitivity analysis can be useful for variety of reasons in our work such as testing the robustness of the results of a model or system in the presence of uncertainty. It provides increased understanding of the relationships between input and output variables in a system or model. Uncertainty reduction, through the identification of model inputs that cause significant uncertainty in the output and should therefore be the focus of attention in order to increase robustness (perhaps by further research). Searching for errors in the model (by encountering unexpected relationships between inputs and outputs)

Table 3 The data sets for sensitivity analysis performed for Block Diagram N=14 & N=15

Figure 12															
Areas	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}	
Value	200	700	400	300	100	300	600	100	300	200	900	500	500	1000	
Figure 13															
Areas	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}	
Value	250	900	400	450	200	450	800	200	400	300	1000	600	600	1200	
Figure 14															
Areas	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}	
Value	180	680	330	280	95	290	580	110	280	220	890	480	480	880	
Figure 15															
Areas	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}
Value	800	800	500	800	800	250	250	250	250	250	250	400	200	400	1000
Figure 16															
Areas	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}
Value	600	600	600	600	600	150	150	150	150	150	150	200	100	200	1000
Figure 17															
Areas	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}
Value	1000	1000	800	1000	1000	350	350	350	350	350	150	450	200	450	2000

Model simplification – fixing model inputs that have no effect on the output or identifying and removing redundant parts of the model structure. Finding regions in the space of input factors for which the model output is either maximum or minimum or meets some optimum criterion, consider optimization and Monte Carlo filtering.

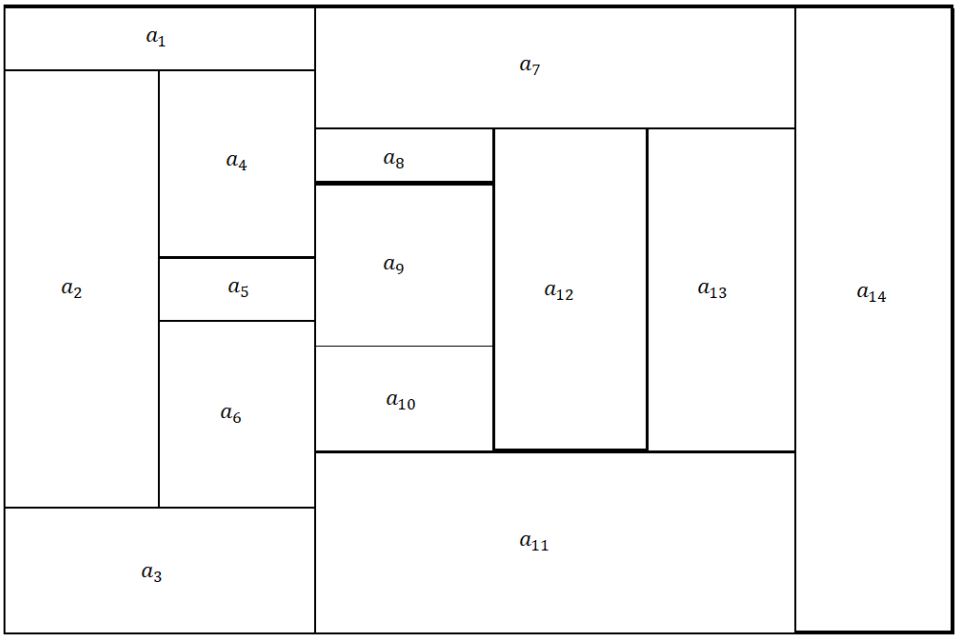


Figure 12 For N=14,1st Iteration, tetrakis hexahedral block diagram conversion of Voronoi triangulation

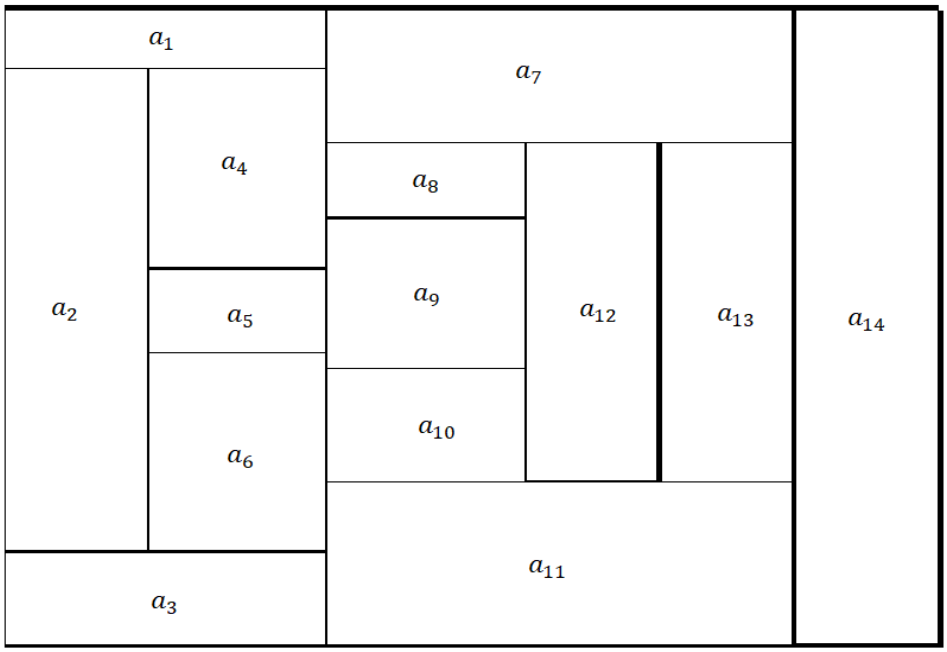


Figure 13 For N=14,2nd Iteration, tetrakis hexahedral block diagram conversion of Voronoi triangulation

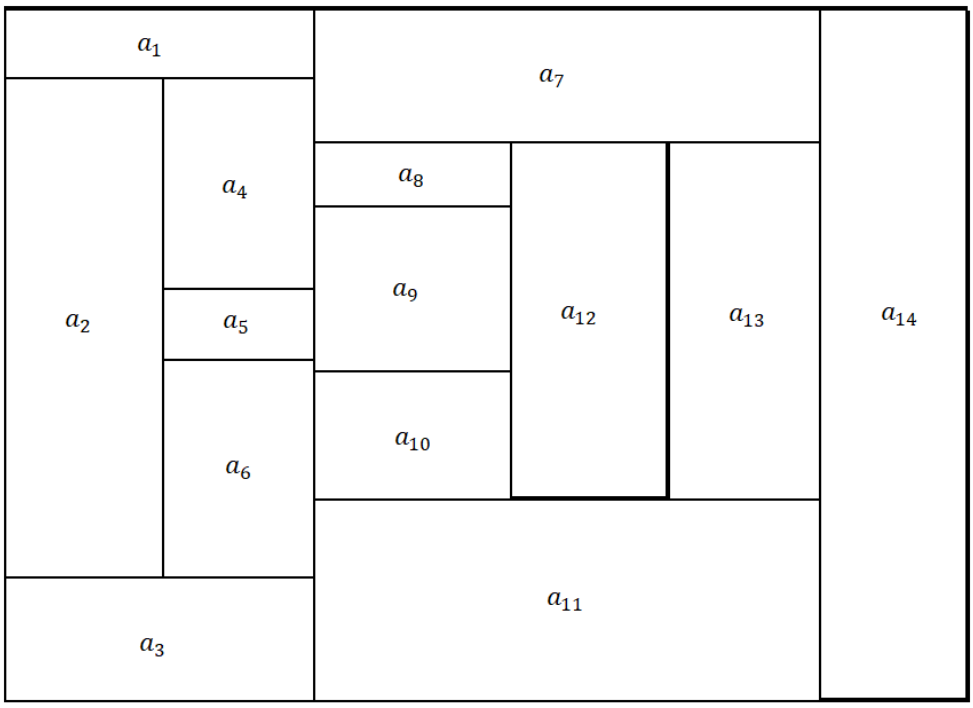


Figure 14 For N=14, 3rd Iteration, tetrakis hexahedral block diagram conversion of Voronoi triangulation

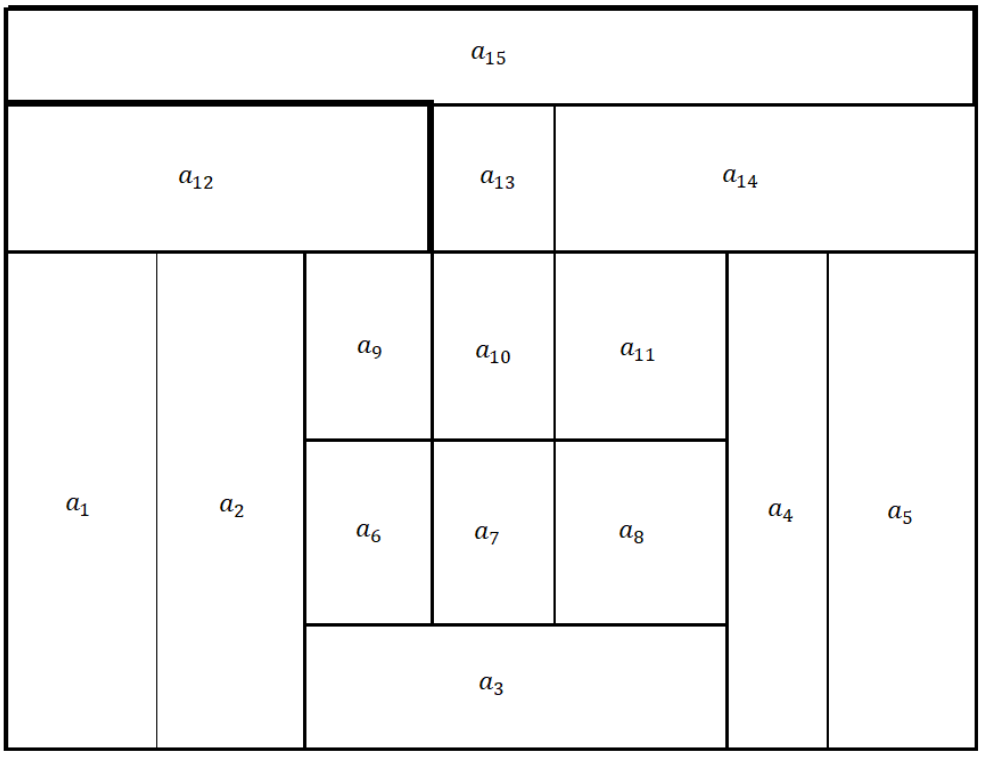


Figure 15 For N=15, 1st Iteration, Poussin graph block diagram conversion of Voronoi triangulation

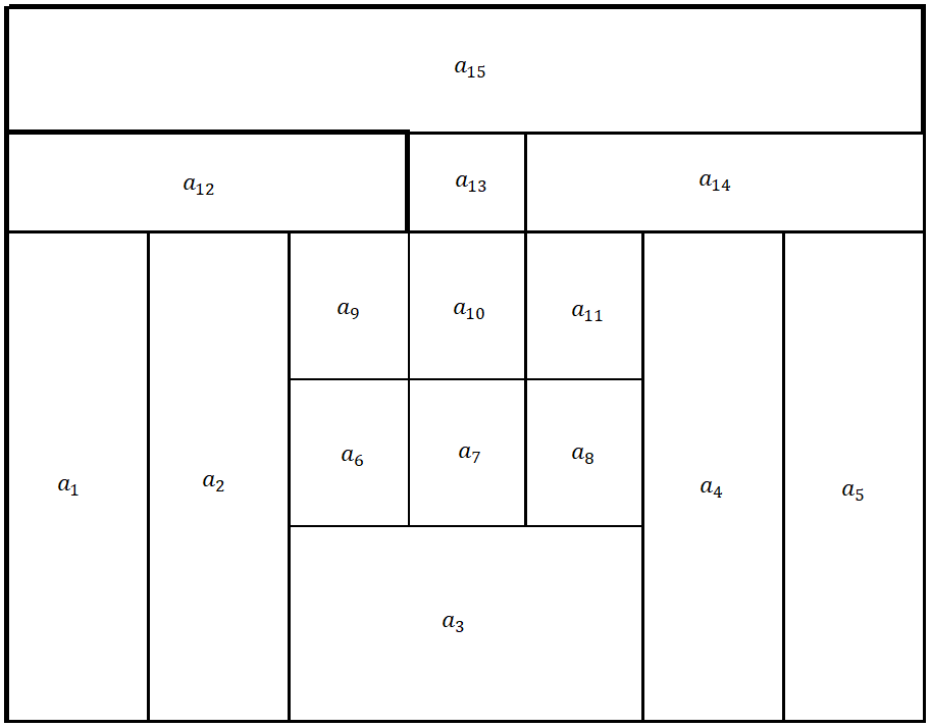


Figure 16 For N=15, 2nd Iteration, Poussin graph block diagram conversion of Voronoi triangulation

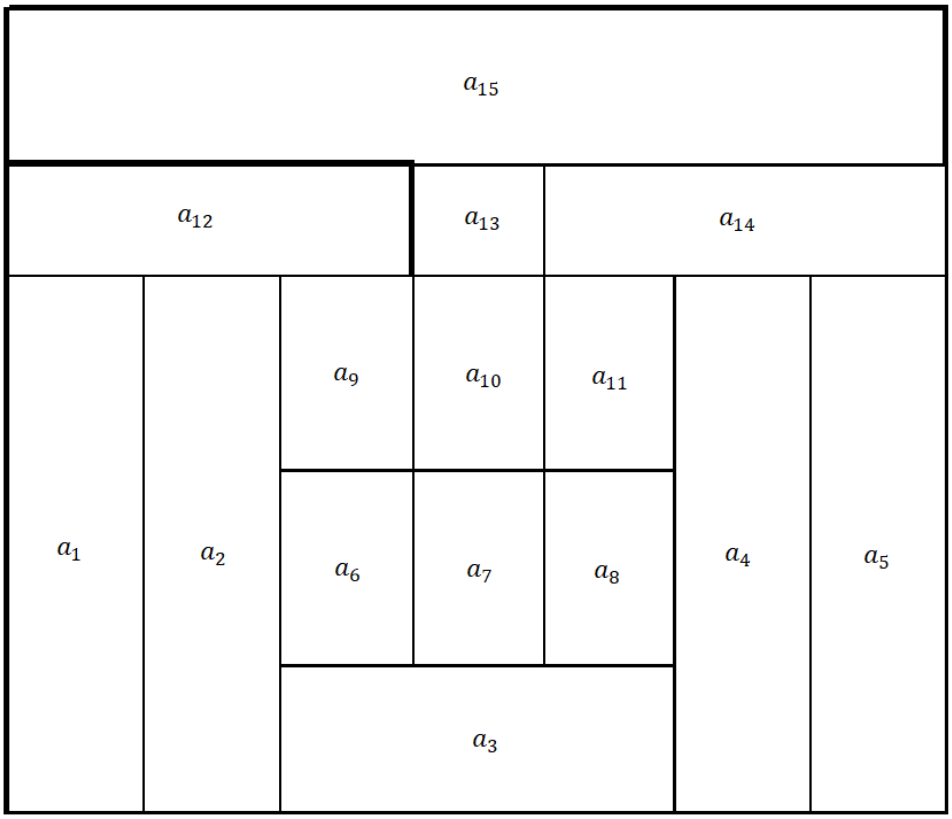


Figure 17 For N=15, 3rd Iteration, Poussin graph block diagram conversion of Voronoi triangulation

It can be easily seen through the iteration of block plan diagram, shown through the figures 12 to 17 above, our system of equation is very stable, even with the enormous variability of input area, as provided for cases above. We entered data with scaling of over 200 % for N=15 and scaling of up to 100 % in case of N=14, as represented in table 3, between the performed iteration and nonetheless the system gives a stable block plan coordinates output for each case. For the Case of N=14, considering, it includes a set of complex decision, right from MPG to the output Voronoi diagram, the solution system above proves with an outcome about robustness. These outcomes prove that the formulated system is very robust and can readily be deployed for all named MPG Voronoi graph expansion and beyond for area-based parameter optimization considering all practical purposes.

CHAPTER 5
MHS SOLUTION USING RSMT APPROACH

MHS stands for material handling system solution. From an engineering point of view, material handling is defined as the technique and art involved in the moving, packaging and storing of matter in any form. A material handling system is defined as series of equipment, element or devices designed in concert or in sequence to accomplish the movement, storage and control of material in an operating environment and with designated equipment and system design. There is a very strong correlation between the design of the facility and the design of the material handling system. Facilities are probably best configured once the facility layout design is well established. In a perfect world, the design of the MHS and the facility layout should be done simultaneously. In most manufacturing and service systems, coupling the design of the layout with the dynamic flow of the customers and products within the system in order to access the performance of alternative design layout is a most desirable property of a model [1]. We will discuss a symbiotic approach for MHS solution in this chapter for the block plans created in chapter 4.

The rectilinear Steiner minimum tree problem (RSMT) is a variant of the geometric Steiner tree problem in the plane, in which the Euclidean distance is replaced with the rectilinear distance. The problem may be formally stated as follows: given n points in the plane, it is required to interconnect them all by the shortest network which consists only of vertical and horizontal line segments. It can be shown that such a network is a tree whose vertices are the input points plus some extra points (Steiner points). Steiner tree problem, or minimum Steiner tree problem, named after Jakob Steiner, is an umbrella term for a class of problems in combinatorial optimization (finding an optimal object from a finite set of objects, Beasley, J. E. "Integer programming"). While Steiner tree problems may be formulated in several settings, they all require an optimal

interconnect for a given set of objects and a predefined objective function. Given an undirected graph with non-negative edge weights and a subset of vertices, usually referred to as terminals, the Steiner tree problem in graphs requires a tree of minimum weight that contains all terminals (but may include additional vertices).

The Steiner tree problem in graphs which has been used in our work of combinatorial optimization problems is the minimum spanning tree problem. If all vertices are terminals, the Steiner tree problem in the graph is equivalent to the minimum spanning tree. However, if it is non-negative, the minimum spanning tree problem is solvable in polynomial time, the decision variant of the Steiner tree problem in graphs is NP-complete (which implies that the optimization variant is NP-hard); in fact, the decision variant was among Karp's original 21 NP-complete problems. Despite the pessimistic worst-case complexity, our Steiner tree problem variants can be solved efficiently in practice for the value of n up till n=32, which encompasses a lot of real-world problems and it can even solve for more large-scale though it might not be efficient enough to do so [36] [37].

The general Steiner tree problem can be approximated by computing the minimum spanning tree of the subgraph of the metric closure of the graph induced by the terminal vertices. The metric closure of a graph G is the complete graph in which each edge is weighted by the shortest path distance between the nodes in G. This algorithm produces a tree whose weight is within a $2 - 2/t$ factor of the weight of the optimal Steiner tree; this can be proven by considering a traveling salesperson tour on the optimal Steiner tree. The approximate solution is computable in polynomial time by first solving the all-pairs shortest paths problem to compute the metric closure, then by solving the minimum spanning tree problem. [38]

For our research we have used the GeoSteiner version 5.1 — an optimization software package that solves the minimum spanning tree problem in hypergraphs, NP-hard problems. To the best knowledge of the authors, as of February 2018, GeoSteiner represents the computational state of the art for geometric Steiner tree problems in the plane. The program has been created by Warme, D.M. for spanning trees in hypergraphs with applications to Steiner trees for his Ph.D. thesis in computer science dept at the University of Virginia, 1998. This work was licensed under a creative common's attribution 4.0 international license until 2014 but since then they have released 5.0 and 5.1 version which are open source. The package currently solves the discussed NP-hard problems in the plane Rectilinear Steiner tree problem, GeoSteiner is written in ANSI C. The code makes heavy use of linear programming (LP); the public domain LP-solver lp solve is included (in a significantly modified form). However, the package also supports CPLEX, a proprietary product of IBM Inc., which is perhaps the fastest and most robust LP-solver available. The authors of GeoSteiner strongly recommend that we use CPLEX if possible. The core callable library requires no supplementary software or libraries (except the CPLEX library if GeoSteiner is configured to use CPLEX as its LP solver) [35].

The geometric Steiner tree problem is known to be NP-complete for the rectilinear metric, and NP-hard for the Euclidean metric. This is the fastest exact algorithms in practice for our problems and uses two phases: First, a small but enough set of full Steiner trees (FSTs) is generated and then a Steiner minimal tree is constructed from this set. These phases are called FST generation and FST concatenation, respectively. FST concatenation is almost always the most expensive phase and has traditionally been accomplished via simple backtrack search or dynamic programming.

```
// Running values based on vornoi centers for N=3 /
#include "geosteiner.h"
#include "stdlib.h"
int main (int argc, char** argv)
{       double terms [6] = {7.07,14.14,
        28.28, 14.14,7.07, 42.42};
        int i, nsps;
        double length, sps [3];
       /* Open GeoSteiner environment */
        if (gst_open_geosteiner () != 0) {
                printf ("Could not open GeoSteiner.\n");
                exit (1);}
        int nedges;
        int edges[6];
        int status;
        /* Compute Rectilinear Steiner tree */
        gst_rsmt (3, terms, &length, &nsps, sps, &nedges, edges, &status, NULL);
        // gst_rsmt (N, terms, &length, &nsps, sps, NULL, NULL, NULL, NULL);
        /* Display information about solution */
        printf ("Steiner tree has length %f\n", length);
        for (i = 0; i < nsps; i++) {
                printf ("Steiner point: (%f, %f)\n", sps[2*i], sps[2*i+1]);}
        printf("Number of edges: %d\n",nedges );
                for (i = 0; i < nedges; i++) {
        printf ("Steiner edge: (%d,%d)\n", edges[2*i],edges[2*i+1]);}
                printf("Status %d\n",status);
        /* Close GeoSteiner environment */
        gst_close_geosteiner ();
        exit (0);}
```

Figure 18 Image of forvn3.c a C script expository for compiling RSMT solution for inputted block plan layout as MHS solution.

All metrics currently handled by GeoSteiner are uniformly oriented metrics: Given a set of $\lambda \geq 2$ uniformly oriented directions in the plane, the distance between two points is defined to be the length of the shortest path in which all line segments have one of the given directions. The fact that the number of terminals in each FST in an SMT usually is small is what makes it possible to solve large problem instances to optimality. More specifically, the algorithms employed by GeoSteiner first generate a set of FSTs known to contain an SMT as a subset, and then the shortest possible union of FSTs interconnecting all terminals is selected; we say that the FSTs are concatenated. The concatenation problem is equivalent to finding a minimum spanning tree in a

hypergraph (problem MSTHG) [39]. An efficient solver for this subproblem forms a cornerstone of GeoSteiner.

```
int gst_rsmt (int            nterms,
              double*        terms,
              double*        length,
              int*           nsps,
              double*        sps,
              int*           nedges,
              int*           edges,
              int*           status,
              gst_param_ptr  param);
```

nterms	Number of points (or terminals).
terms	Input point coordinates $(x_1, y_1, x_2, y_2, \ldots)$.
length	Length of computed SMT.
nsps	Number of Steiner points.
sps	Steiner point coordinates.
edges	Edges of SMT (terminals have index 0 to nterms-1 while Steiner points have index nterms and up).
status	Solution status code (see page 114).
param	Parameter set (NULL=default parameters).

Returns value zero if an SMT was computed and non-zero otherwise.

Figure 19 The gst()rsmt argument of GeoSteiner program that has been used to create RSMT output solution script [22]

The script used in this research for generating RSMT solution for given block plan layout is depicted in figure 18. It computes a Rectilinear SMT for the Voronoi seed points or centers as terminal from the block plans of the previous chapter. We first call GeoSteiner environment using

```
gcc -Ilp_solve_2.3 -o forvn11 forvn11.c memory.o -L./ -lgeosteiner lp_solve_2.3/libLPS.a -lm
```

Figure 20 Depicts C language command for computing RSMT sol script in GeoSteiner for the Voronoi coordinates of N=11 case

.

the high-level function *gst rsmt()* to compute the RSMT (Figure 19). As arguments, we first pass the number of terminals which are block plans Voronoi seed coordinates and then a double array

terms to holds these terminal point or coordinates. Then it follows with variables length, nsps, and

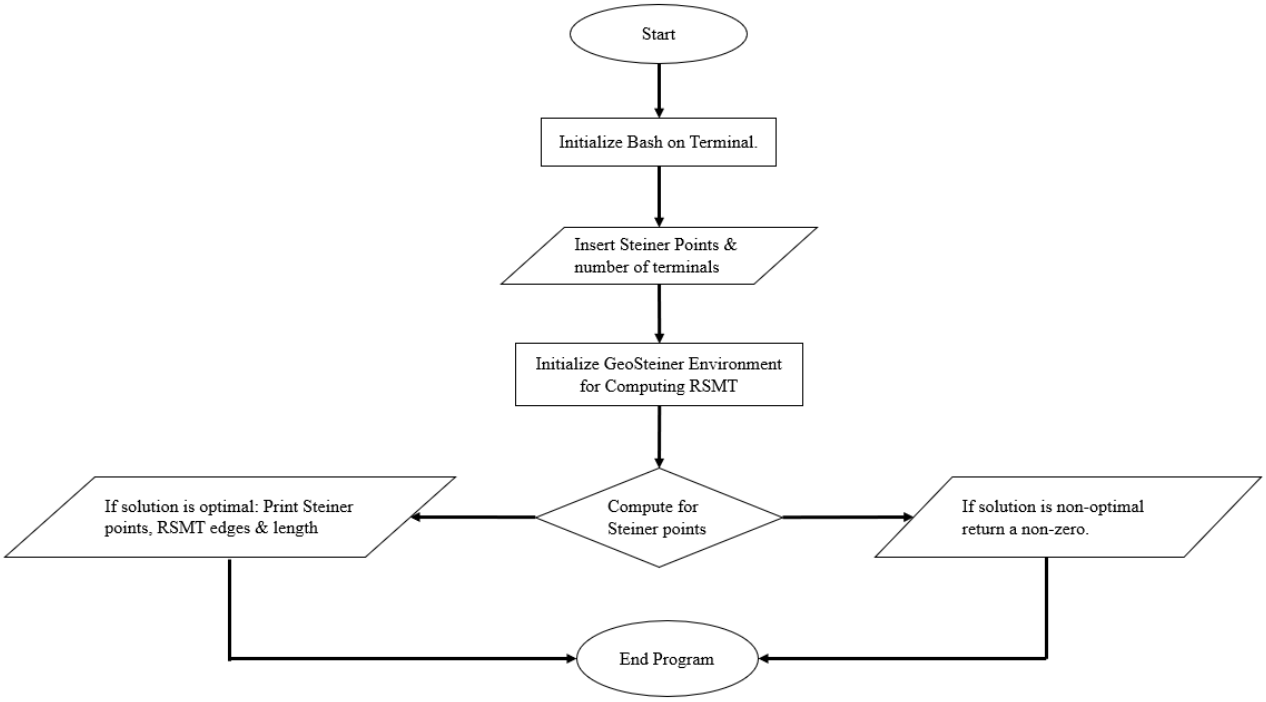


Figure 21 Flow Diagram for the compiling of RSMT solution based on which script for GeoSteiner is written.

sps, edges, status in which the length of the RSMT is computed, the number of Steiner points, the coordinates of the Steiner points are returned, the number edges forming the RSMT graph and the solution returns zero if the operation was successful and non-zero otherwise. The remaining arguments to gst rsmt() is given as NULL, causing corresponding inputs to assume default values, and corresponding outputs to be ignored; in default values are assumed for all GeoSteiner parameters. We print the length of the RSMT and the Steiner points.

Finally, we close the GeoSteiner environment and the program ends. We encourage to run the forvn.11c program (file enclosed with this research) using the compiler commands that have been represented in figure 20, which calls LP solver from the callable library into the GeoSteiner environment and initializes the program input for the bb.c program script (FST solver) to produce

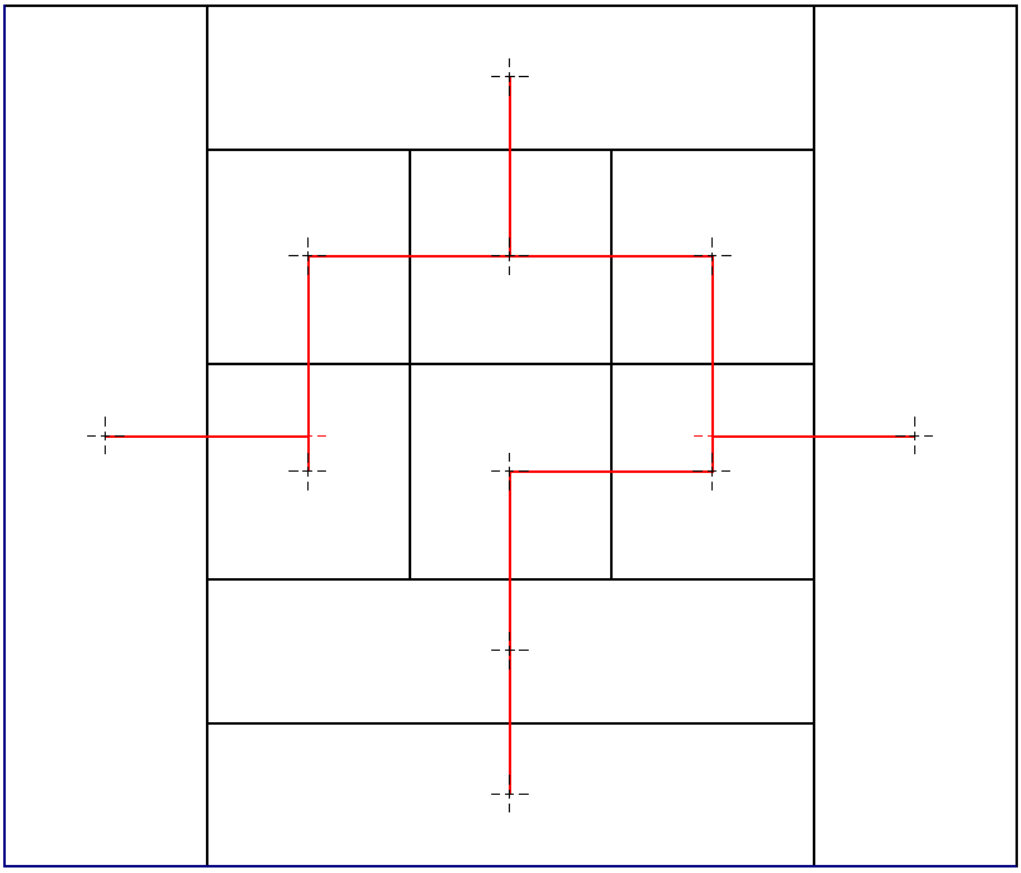


Figure 22 Block plan layout for N=11 with RSMT graph sol embedded as the red lines with Steiner points and Voronoi Centers.

a run file, forvn11. The result obtained from running the output file, forvn11 is depicted in figure 22, it is drafted in Autodesk Inventor 2019 student version package using standard.ipt file format under ANSI standard using the output Steiner points and edges from the print solution.

Phase III-Example Case

We will discuss the methodology of N=26 case in this section to give an understanding of how block plan layout embedded with RSMT graph solution for MHS are created in Appendix III. Firstly, we took the coordinates of the Voronoi center (Table 4) or seed sites generated in Phase II of our research, also depicted in the previous chapter and Appendix II. The coordinates obtained for these Voronoi center or seeds are fed into the GEO Steiner program through a C programming

language script as depicted in figure 18 with parameters initialized for N=26 (forvn26.c file enclosed with this document for reference.). The file was compiled in similar fashion (figure 20) using the legacy gcc C compiler and calling the LP solver from the callable library and running the files through the bb.c program.

Table 4 Voronoi seed coordinates for N=26 block plan layout solution generated in chapter 4.

Figure 22												
X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	X11	X12	X13
4.05	12.97	29.11	73.81	114.76	126.32	133.62	29.11	49.58	66.29	83.01	99.73	114.76
X14	X15	X16	X17	X18	X19	X20	X21	X22	X23	X24	X25	X26
29.11	49.58	66.29	83.01	99.73	114.76	20.19	49.58	66.29	83.01	99.73	122.1	68.43
Y1	Y2	Y3	Y4	Y5	Y6	Y7	Y8	Y9	Y10	Y11	Y12	Y13
30.82	30.82	13.3	13.3	13.3	30.82	30.82	37.47	37.47	37.47	37.47	37.47	37.47
Y14	Y15	Y16	Y17	Y18	Y19	Y20	Y21	Y22	Y23	Y24	Y25	Y26
54.99	54.99	54.99	54.99	54.99	54.99	68.3	68.3	68.3	68.3	68.3	68.3	76.42

For the purpose of compiling above C program, we have used bash on Ubuntu on windows shell in the windows 10 interface with non- legacy command module instruction and IBM C-plex solver with additional of independent gcc compiler.The solution file thus obtained reflects the RSMT edges and Steiner points for the inputted points in the sequence of the graph forming connections or tree, it also tells us the length of the RSMT. The solution gives a zero value if optimality is obtained, the entire arrangement of how the solution is printed can be understood through the argument represented in figure 19 and explained previously in this chapter.

The output values obtained in the solution file fornvn26 (generated from compiling forvn26.c script) is drafted using Autodesk Inventor professional 2019 student version software package. The draft is created using software's professional grade solution for CAD visualization, and documentation in industry standard DWG™ drawings. All the Voronoi seed points with Steiner points generated are drafted on the block plan layout solution plotted based on phase II results. The edges are drafted with red color lines based on the solution sequence of edges. The

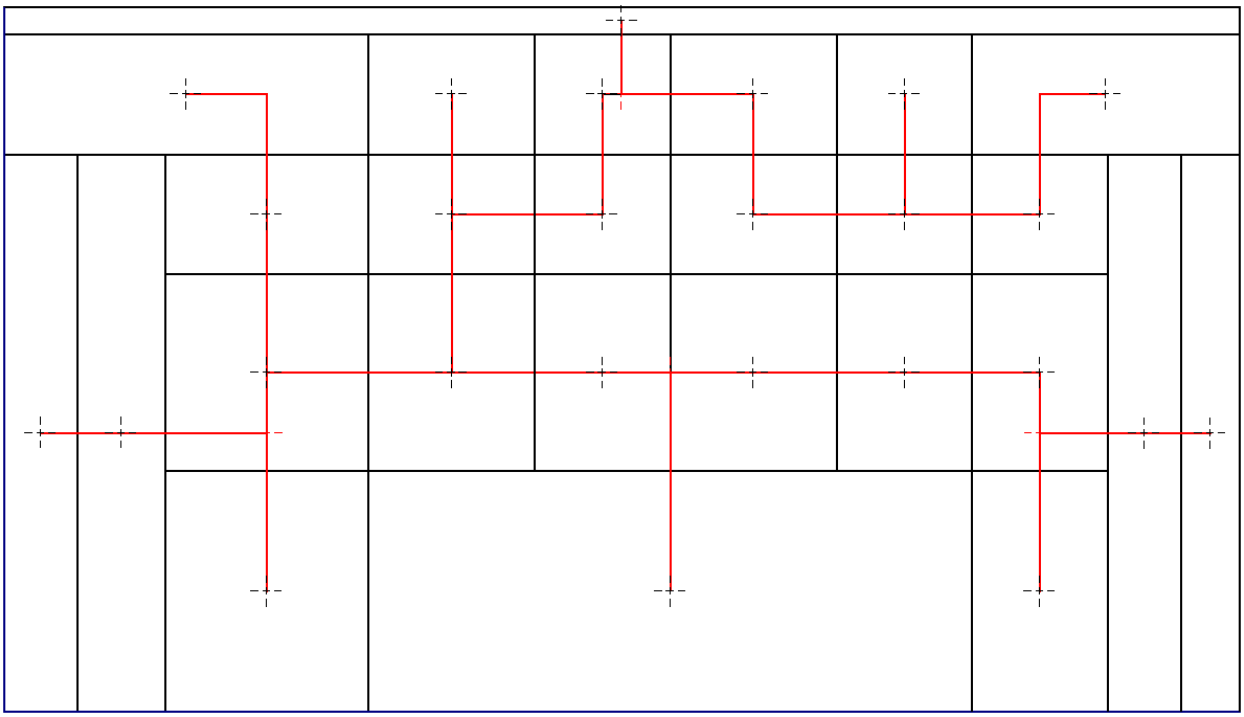


Figure 23 Block plan layout for N=26 with RSMT graph sol embedded as the red lines with Steiner points and Voronoi Centers.

final draft for N=26 is depicted in figure 23, where the red color lines mark the RSMT graph solution for the given N and the cross dots represent all the Voronoi seed sites and Steiner points generated through GEO Steiner.

CHAPTER 6
FINAL SYNTHESIS- A CASE STUDY

In this age of competitive market, companies must produce cost effective products & services, which can be realized by minimized production cost and higher effectiveness. The effective facility planning can significantly reduce the operational costs of companies. An adequate facility layout can result in the improvement of the performance of any facility. This is a demanding but ultimately most rewarding task. In this chapter, we will amalgamate all the 3 phases of our research and in order to achieve an effective facility layout. We will walk through a real-world case study by running our methodology, explained in detail. It will help us in rationalizing on how to apply our research for real world facility designing problems. Also, the outcome of the work has a commercial value that can be used as a sales material.

Description of case study

For the purpose of analysis and validation of our research methodology, we have selected Int. J. Naval Archit. Ocean Eng. (2013) 5:132~146. It is a case study that dives into the modern age shipbuilding paradigm, where for several decades, Asian nations such as Korea, Japan and China have been leading the shipbuilding industry since the decline in Europe and America. However, several developing countries such as India, Brazil, etc. are making an entrance into the shipbuilding industry. These developing countries are on lookout for Joint Ventures, technical partners and information providers because of lack of experiences and technologies. This has led for shipbuilding engineering companies of developed countries to get a chance of engineering business against those developing countries.

The subject in the case study is Petróleos de Venezuela, S.A. (PDVSA) Shipyard company of Venezuela. It is originally national owned oil company, which is getting into the shipbuilding industry in order to make a ship for their own oil company. Shipbuilding business starts with a shipyard construction, which involves a large-scale investment initially. The starting point of shipyard construction is to design a shipyard layout. For this purpose, four kinds of engineering parts are required. These are civil, building, utility and production layout engineering. Among these, production layout engineering is most important because its result becomes foundation of the other engineering parts and determine the shipyard capacity in the shipyard lifecycle.

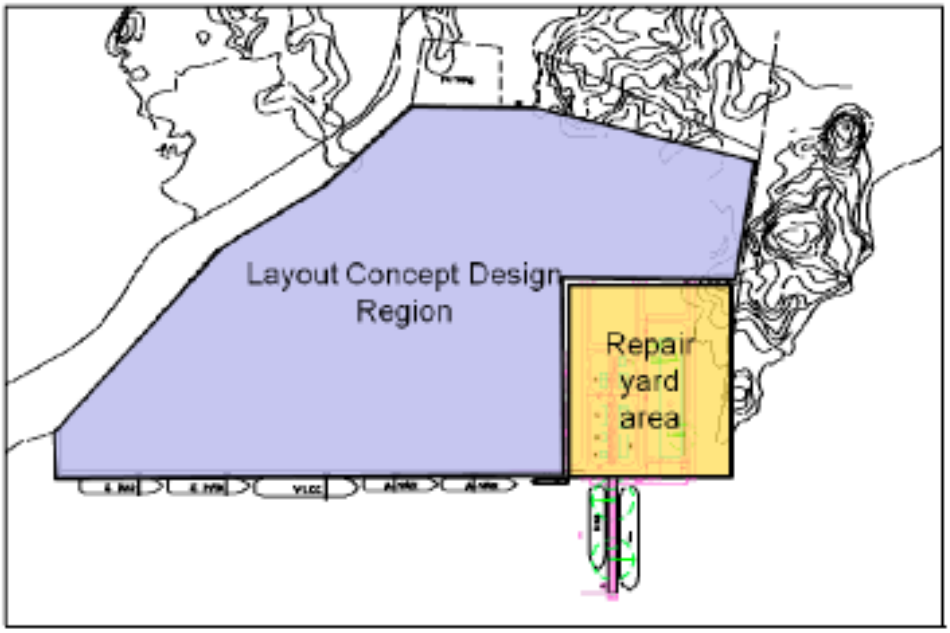


Figure 24 Physical Geometry of shipyard target site for PDVSA in Venezuela of South America [40].

The case study being used consists of integrated shipyard layout design methodology developed on the actual product data of planned ship and the actual shipbuilding operation data for the preliminary shipyard layout design phase. The case study is conducted for the newly

planned project in Venezuela of South America (Figure 24). The data in this case study is differentiated as being unique as the actual product data from the target ship and the actual shipbuilding operation data is available for utilization. The starting point of the case study shipyard is a green-field project (Greenfield project is one which is not constrained by prior work, it is constructing on unused) for the construction of new shipyard. It leaps ahead of other examples about layout design, because most research cases are in the tower of ivory, which means that there is little consideration of real operation. Here, a shipyard layout design for preliminary phase is conducted for the target of newly planned shipyard at Venezuela of South America with an integrated method that can deal with actual master data from the shipyard [40]. Moving on to understanding the case study in the sphere of our research, we will demonstrate the methodology developed by us in the three phases of our research in following three section. These sections will be in a binding form filling all the gaps and giving a holistic picture of our work. It will help the reader in forming a complete perspective towards the research phases detailed in there individually chapters.

Phase I

As discussed in Chapter 3, Planar graph are among the most ubiquitous graphs that arise in applications and can be drawn in the plane without edges crossing. The class of maximal planar graph (MPG) are special group of planar graphs that have the greatest number of edges for given vertices without any edge intersecting. These graphs form the fundamental for layout designing in our research by providing position skeleton for grouping based on number of vertices. To sort the grouping, we utilize GMAFLAD (A GUI based QSP optimizer) software application, discussed earlier. It helps us identify best fit location of department on vertices based on flow matrix (flow

of people and material between departments) and any weightage (higher utility value to a department for sorting, based on location, example dispatching or receiving area next to boundary wall of the facility get higher utility value compared to other vertices) if applicable.

We began in the similar fashion as employed for the example in chapter 3. After opening GMAFLAD software application, we open a new file, add number of required columns, rows and click on resize, then we plot icosahedral graph (N=12) as depicted in figure 9 in appendix 1, by clicking add for the activities and then marking the tiles generated by clicking resize. We selected icosahedral graph here, because our case study has 12 departments for which the facility must be

Table 5 Flow or activity among each work area or department for the PDVSA shipyard in Venezuela at brownfield state [40].

	1	2	3	4	5	6	7	8	9	10	11	12
1		600	0	0	0	0	0	0	40	0	0	0
2	0		600	0	0	0	0	0	0	0	0	0
3	0	0		200	0	0	300	100	0	0	0	0
4	0	0	0		150	25	25	0	0	0	0	0
5	0	0	0	0		100	50	0	0	0	0	0
6	0	0	0	0	0		100	25	0	0	0	0
7	0	0	0	0	0	0		475	0	0	0	0
8	0	0	0	0	0	0	0		600	0	0	0
9	0	0	0	0	0	0	0	0		640	0	0
10	0	0	0	0	0	0	0	0	0		640	0
11	0	0	0	0	0	0	0	0	0	0		640
12	0	0	0	0	0	0	0	0	0	0	0	

designed and optimized. While marking the tiles we add equal number of alternate by clicking add for alternate. From the 12 departments for shipbuilding 2 are docks, which need to be on the boundary wall of the factory corresponding to the ocean for which we only add alternate to corners of the icosahedral graph plotted. We also added a weightage factor with utility value 2 for both the Docks, Main Dock (Dock-1) & Supplement Dock (Dock-2). The Flow matrix for this phase is processed from the case study for the given 12 departments or section as shown in table 5 and used as input into by selecting edit flow matrix button.

```
THE NAME OF THE DATA FILE IS cas12.dat
CPU TIME USED IN OPTIMIATION IS     0.0000 SEC.
THE STORAGE USED IN OPTIMIATION IS   15553
The value of the solution is   1639.84
Activity      Alternate
   1              4
   2              5
   3              6
   4              3
   5              1
   6              2
   7              8
   8             12
   9             11
  10              9
  11              7
  12             10
```

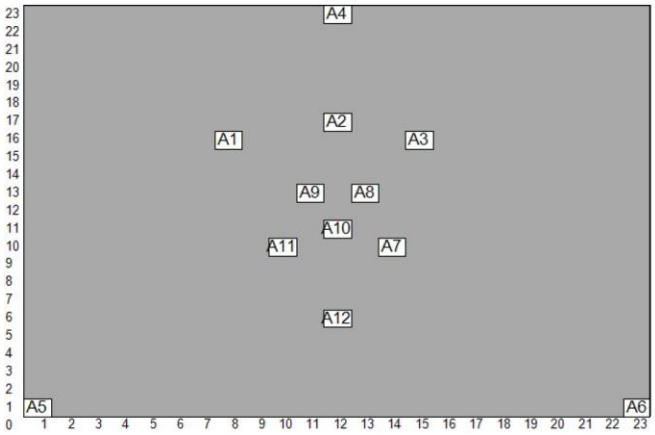


Figure 25 GMAFLAD output for shipyard model for given flow, N and alternates available optimized for most interconnections.

After completing above process, we saved the file with .dat extension (naming restricted to 5 character for file by the software application) and exit the edit input file. We select the Greedy heuristic radio button and click on run to execute our model. After computing GMAFLAD generates result file, which pops up automatically, the GMAFLAD output for our case study is shown in figure 25. It depicts the optimum site location on the MPG for all the departments based on input factors. It forms the basis of phase two of our research by identifying grouping or position of departments on MPG vertices. We solve for the dual of the MPG and expand the graph generated into block diagram in next section.

Phase II

Delaunay triangulations are leveraged heavily in many applications, especially computer graphics, as they are ways to break up regions into triangles. 3D graphics cards are optimized to render triangles very efficiently. If you create a dual graph of a Voronoi diagram (connect each node to every other node that shares an edge), you end up with a graph that is a Delaunay Triangulation, a construct just as interesting as Voronoi tessellations. In our research we do the converse and generate dual (the Voronoi complement) with our MPG (a Delaunay Triangulation)

as primal. We frequently need to find the nearest location of variety of enterprises near us. A map divided into cells, each cell covering the region closest to a center, can assist us in our quest. Such a map is called a Voronoi diagram, named after Georgy Voronoi, a mathematician born in Ukraine in 1868. Voronoi diagrams, these constructions are pretty and simple to understand and are easily constructed with computer software, they can be depicted as colorful charts, indicating the region associated with each service point or site. [41] We have explained them in detail in chapter 4, with the formulation and procedure to solve the dual of MPG to generate the Voronoi graphs, all these graphs are illustrated in appendix 2 .

Table 6 Area calculation summary from case study for block plan generation through optimization.

Sr No.	Shop	L	B	Area(m2)
1	Dock 2	570	90	51,300
2	Painting	100	489	48,947
3	Unit assembly	200	489	97,893
4	Dock 1	660	105	69,300
5	PE	100	507	50,691
6	Grand assembly	220	445	97,893
7	Sub assembly	220	445	97,893
8	Fabrication	110	222	24,473
9	Plate stock	80	154	12,298
10	Cutting shop	130	88	11,400
11	Treatment shop	128	50	6,400
12	Outfitting	215	228	48,947

In phase two, our goal is to generate the block plan for facility layout with the results generated by GMAFLAD for position of departments on vertices of MPG in phase one. For our case study we use the pre-solved Voronoi graph for icosahedral graph (N=12) as depicted in figure 7 of appendix 2. It is generated through graphical package of Mathematica 11.0 containing solver to generate dual for input MPG (Delaunay tribulation). The Voronoi graphs generated for listed MPG in our research can be directly utilized for any case, based on the given value of N. These Voronoi structure now need to be expanded into the block diagram for which we have created a

formulation based on parameter optimization, that requires user input of area as discussed in chapter 4. We use Lingo 17.0 optimization software package to solve the NLP (nonlinear programing) constraint and objective function of the formulation. The model needs input of area apart from the standard equation expansion. We have created it in this manner to allow for the user

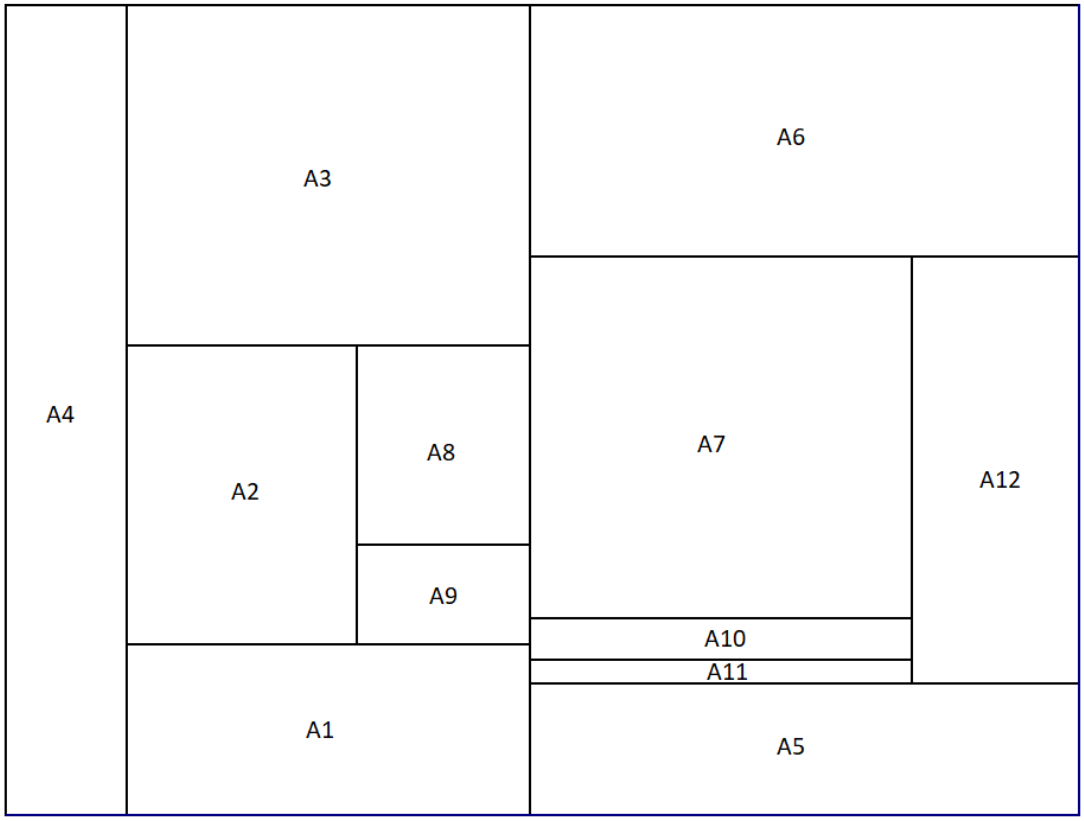


Figure 26 The Block plan output for the case study based on input area with GMAFLAD MPG output embedded in for grouping.

to optimize different facility for given value of N, based on available area data set. For our case study we have put together the area of all the departments and tabulated it in table 6. The script containing the model for the case study created to generate block plan coordinates through expanding the Voronoi graph by discussed formulation and area input is named Lingo Casestudy12.lg4, It is enclosed with this research. We can run it directly by using the Lingo17.0 version and compiling the script.

Finally, we plot the block plan based on the coordinates output from the Lingo17.0 software application. We use Autodesk inventor 2019 professional student package to draft the block plan as depicted in Figure 26. It portrays the optimized location of all the departments through GMAFLAD solution being embedded in it. We can easily identify in the figure 25 that A4 & A1, which form the Main Dock (Dock-1) & Supplement Dock (Dock-2) end up being on the edge facing the port during area fitting, while being optimized for flow with location restriction.

Phase III

Material handling systems are more complex than ever, and integrators must manage everything from challenging space constraints to a limited labor pool and an intense pace of change to make a variety of moving parts work together in an efficient symphony. A sizeable proportion of manufacturing expenses can be attributed to facility layout and material handling. Facility layout decisions involve designing the arrangement of elements in manufacturing systems. Among the most critical material handling decisions in this area are the arrangement and design of material flow patterns [42]. We have used RSMT (rectilinear steiner minmal tree) graph to project the optimum MHS solution on to the block plan for our research and we will do that for the case study in this section.

Rectilinear steiner's minimal tree problem is variant of Steiner's minimal tree problem, where to find the shortest possible network interconnecting a set of points in the plane is done using rectilinear distance. If the points are linked directly to each other by straight line segments, we obtain the minimal spanning tree. But Steiner's problem allows for additional points – now called Steiner points – to be added to the network, yielding Steiner's minimal tree [43]. This

generally results in a reduction of the overall length of the network causing overall reduction in both displacement and distance for material & people flow, Thus making the system optimised.

Table 7 Voronoi center coordinates of the Block Plan layout for GeoSteiner RSMT result Input. to produce the MHS solution.

Figure 26											
X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	X11	X12
55.51	291.1	577.11	120.99	148.69	318.8	71.88	185.86	311.74	538.29	268.64	341.52
Y1	Y2	Y3	Y4	Y5	Y6	Y7	Y8	Y9	Y10	Y11	Y12
231.00	70.37	231.00	301.37	301.37	301.37	632.23	535.02	535.02	632.23	705.26	853.19

To achieve optimal MHS solution in our research we have used GEOSteiner 5.1, The GeoSteiner software package has for about 20 years been the fastest (publicly available) program for computing exact solutions to NP hard, Steiner tree problems in the plane asking for a shortest possible interconnection of a set of points under some given metric. It was accomplished by Warme, Winter and Zachariasen during as computational study, published in 2000, which documented the performance of the GeoSteiner approach—allowing the exact solution of Steiner tree problems. It was achieved by using a two phase method consisting of FST generation followed by FST concatenation. To produce the RSMT solution for case study, GeoSteiner needs input coordinates to connect them optimally. We have used the vornoi centre coordinates, which are listed in table 7 to produce RSMT solution for our case study. These coordinates have been calculated based on the block plan coordinates produced during area fitting of vornoi digram by parameter optimization in phase 2, using simple alegabric calculations.

The formulation, method, argument and related attributes for writting the code to produce the model for creating RSMT solution in our research has been explained in detail in chapter 5. The code structure, argument and compiling commands for the case study is in simlar order, the file containg the C launguage code for the case study is named forcsn12.c and has been enclosed with this research. On compiling the script we get the (It genrates 4 steiner points to connect the

case study voronoi cordinates optimally) cordinates for steiner points (red points

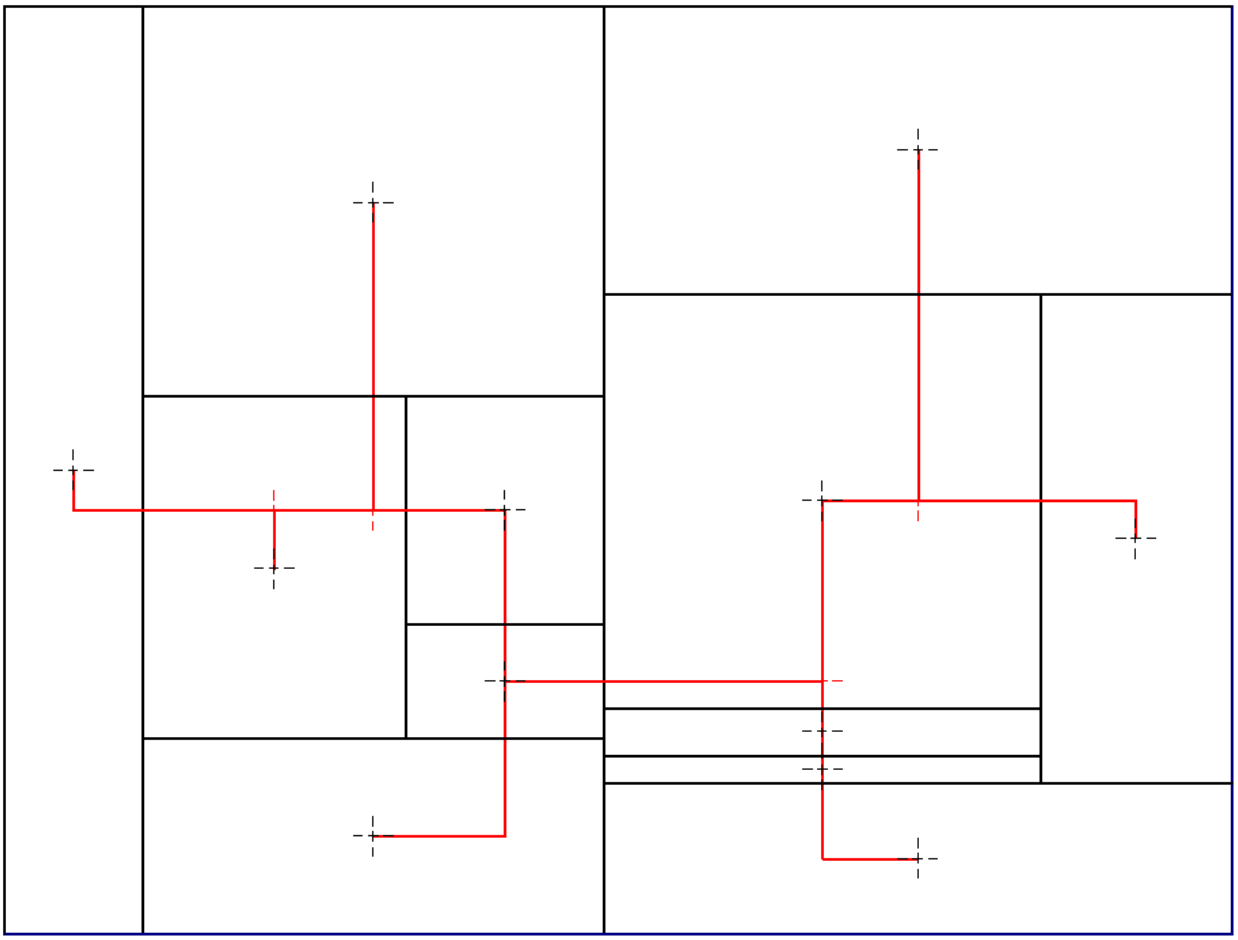


Figure 27 Case study Block Plan solution projected with RSMT graph consisting of Steiner points for MHS solution.

marked on figure 27) , the length of steiner tree with edges forming the RSMT and solution status (The solution is optimal for input cordinates). The graphical output of which has be represented in figure 27, where we have plotted the RSMT graph to dipict the MHS solution on top of block plan genrated in phase two. The digram has been created by using the Autodesk Inventor 2019 professional student version. All the dimensions are actual and have been shrunk to a ratio of 1:1000 to depict onto the sheet.

CHAPTER 7
DISCUSSION

This work gives a detailed outline of what has been done in the field of facility layout, how it has been split into two broad Sections namely, QAP (Quadratic Assignment Problem and GTLN (Graph Theoretic Layout & Network). This research explores the realm of graphical theory for designing facility layout with advent of developments in computational power, efficiency and tools which were not present before and limited the study to QAP, where it has been focused on quantitative equation-based method previously and where most work was done. It still lacks a definite solution and catapults us to our current realm of GTLN for exploration.

In section 1,We have formed a very promising yet robust method to select MPG graphs with department placement through GMAFLAD for any kind of facility based on the number of sections up till N=32. But nonetheless the templates are limited to named MPG and must be expanded through assumption for remaining sections, meanwhile the technology to develop geometries for them develops. There is some difficulty with MPG plotting, as GMAFLAD is a first-generation GUI based application, it has limited flexibility with cell assignment (number of cell , cell alignment with respect to matrix resolution and cell marking technique).

Likewise, in second phase we have solved for the dual of MPG (generating Voronoi graphs) and subsequently plotted the block plan diagram by expanding the graphs, with coordinates populated through optimization for parameter with user defined area input. The expansion on geometry and fitting of area based on data of cases, takes some liberties with grouping arrangement to make the graph rectilinear from Euclidian. It is a concern because it does so by moving the Voronoi cells cluster into standard rectangular layout, (used for warehouses & factories) thereby dissociating the underlying MPG combination to some extent.

In the 3rd phase for the MHS solution, we have used the RSMT graph with an exact algorithm solver, software package GeoSteiner. It works and complies very complex (rectilinear up to 1000 points) problems very rapidly. It has been written in legacy ANSI C programming language, preventing and lag due to conversion of commands from higher order to lower order. GeoSteiner software package—first released in 1999—has been the fastest (publicly available) program for computing exact solutions to Steiner tree problems in the plane. These graphs generated via two phase method are project into the block plan to represent the optimum MHS solution.

Finally , we have explored all the methods employed in each section on research in details with a case study for shipyard layout designing for national oil company of Venezuela. These real data set from case study are very valuable and helped in verifying the functionality of all the three phases of research and validating these methodologies. It sets a president and confirms the commercial and scientific significance of the work with a plausible real time application.

Moving further into the future, we would be exploring ideas of building a software package to completely automate the process from MPG selection to Voronoi generation and finally block plan generation as a single step process based on user input of value of n, areas, flow matrix and any weightage or other attributes for the input parameters. We would also like to incorporate the ability of MHS system into the software.

CHAPTER 8
CONCLUSION

We believe the idea of tackling facility layout designing through the underdeveloped branch of GTLN gives us a promising opportunity to not only explore but develop methodology and SOPs. The idea to develop a systematic procedure for constructing the floor plan graph (FPG) layout along with the MPG, and the MHS explored has a harmonious yet fresh out of the box engineering approach to a classical and a difficult problem. The methodology used in this work is a unique approach, where the area utilization , overall processes and flow rhythm are improvised with overall displacement minimization, thus causing a domino effect of reducing cost and time, throughout the length and breadth of sampled facility. It also leads to individual process optimization in a department by optimizing for their input flow pattern. The approach itself is very cost efficient , lucid and rapidly solvable as the method contains mostly graphical based assertion with minimum data requirement for compilation. It gives a baseline mathematical model with sample data or rather "training data" in machine learning terminology. Which shall be deployed during writing for software package to accomplish all the phases in one step data input for results. It would be capable of augmenting the process further with addition of few self-predictions for decisions without being explicitly programmed to perform the all possible task.

APPENDIX A

MPG GMAFLAD SOLUTION

We have depicting all the named MPG GMAFLAD solution in this section for objective function maximization and related graphical output with input graph and flow matrices. For generating these solutions, we have plotted the named solutions in GMAFLD interface, observing the rows and column space generated as Euclidean space for plotting our triangulated graphs. We have inserted all possible location for all possible n departments and have not add any weightage to any n. For the flow matrix inserted in each problem we have generated instances through benchmark Kate QAP problem generator.

1) For N=3

```
THE NAME OF THE DATA FILE IS ravi2.dat
CPU TIME USED IN OPTIMIATION IS     0.0000 SEC.
THE STORAGE USED IN OPTIMIATION IS     211
The value of the solution is     13.08
Activity      Alternate
   1              3
   2              2
   3              1
```

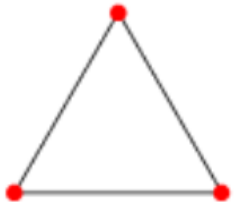
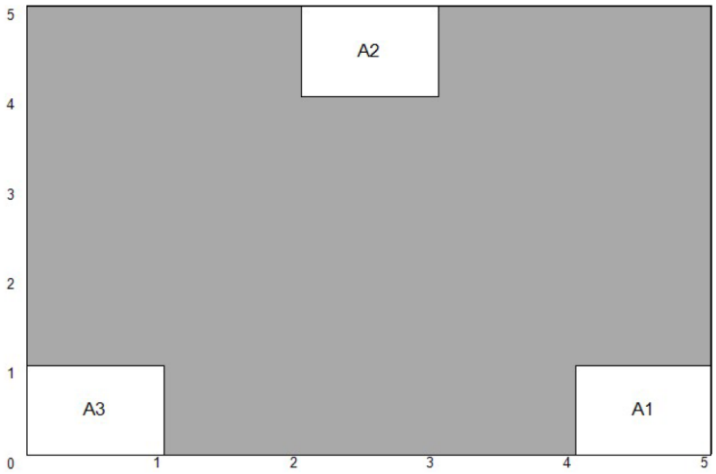


Figure N=3 triangle MPG



	M	F	O
M	-	5	30
F		-	20
O			-

"Load Matrix"

expected # of times people move between departments

2) For N=4,

```
THE NAME OF THE DATA FILE IS ravi1.dat
CPU TIME USED IN OPTIMIATION IS     0.0000 SEC.
THE STORAGE USED IN OPTIMIATION IS     609
The value of the solution is      7.34
Activity      Alternate
   1              1
   2              2
   3              4
   4              3
```

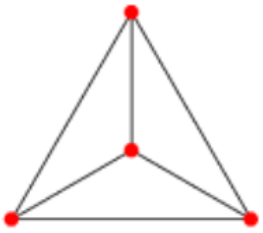
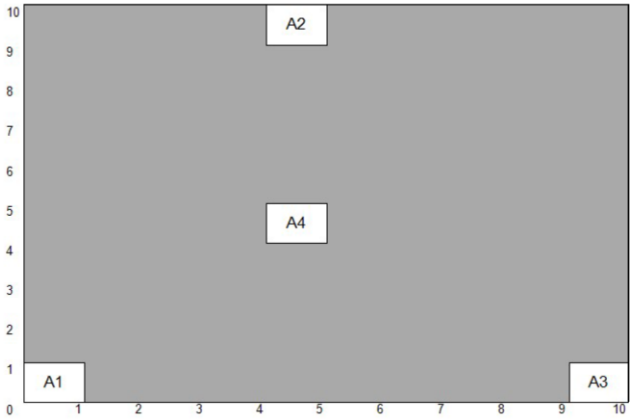


Figure N=4 tetrahedral MPG



	A	B	C	D
A		2	4	4
B	1		1	3
C	2	1		2
D	4	1	0	

3) For N=5

```
THE NAME OF THE DATA FILE IS ravi3.dat
CPU TIME USED IN OPTIMIATION IS     0.0000 SEC.
THE STORAGE USED IN OPTIMIATION IS    1223
The value of the solution is      3.30
Activity      Alternate
   1              5
   2              3
   3              1
   4              4
   5              2
```

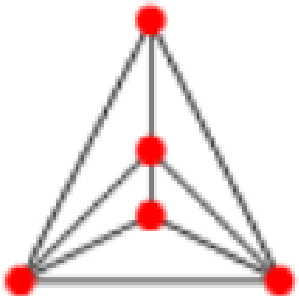
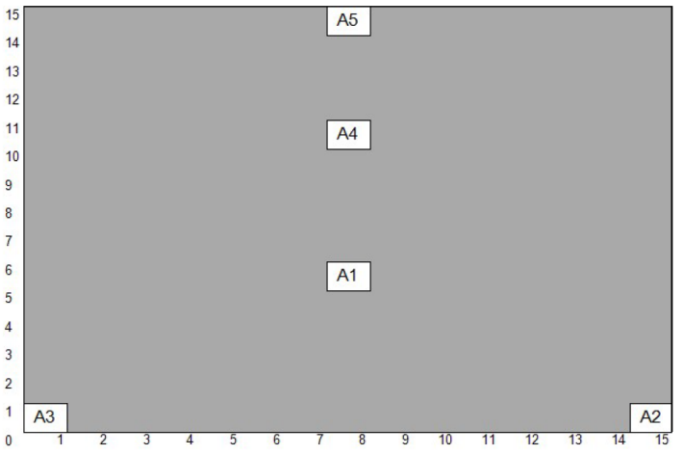


Figure Hexahedra MPG

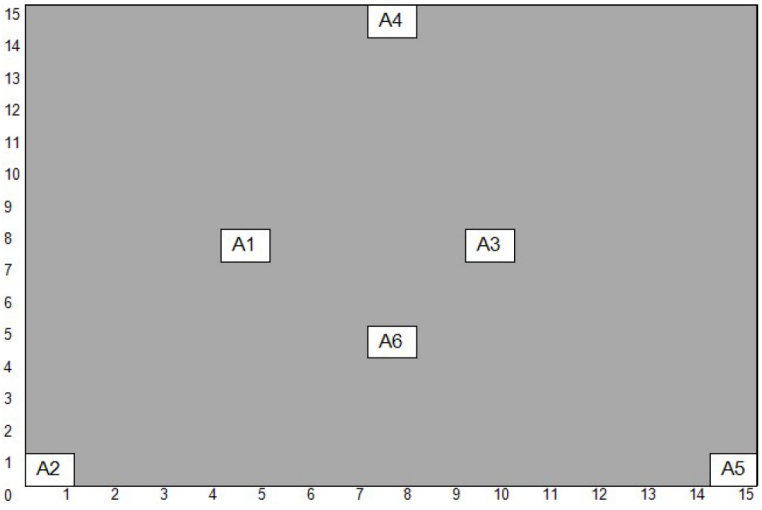


	A	B	C	D	E
A	-	5	2	4	1
B		-	3	0	2
C			-	0	0
D				-	5
E					-

4) For N=6

```
THE NAME OF THE DATA FILE IS ravi4.dat
CPU TIME USED IN OPTIMIATION IS     0.0000 SEC.
THE STORAGE USED IN OPTIMIATION IS    1797
The value of the solution is     61.66
Activity      Alternate
   1              4
   2              1
   3              5
   4              3
   5              2
   6              6
```

Figure N=6 octahedral MPG



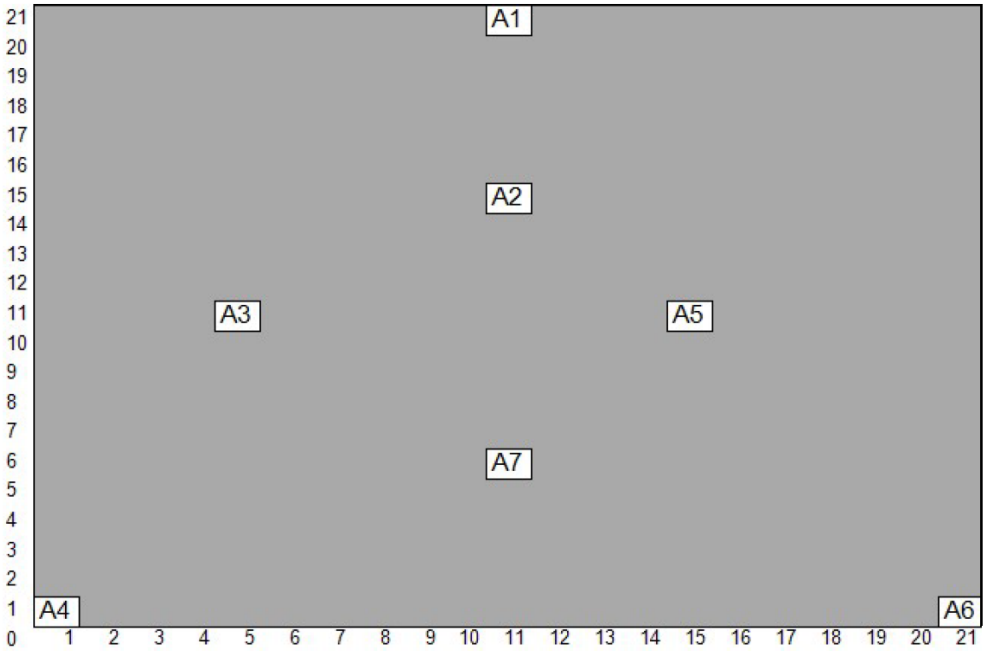
Trips between departments

1	2	3	4	5	6
-	20		20		80
	-	10		75	
		-	15		90
			-	70	
				-	
					-

5) For N=7

```
THE NAME OF THE DATA FILE IS ravi5.dat
CPU TIME USED IN OPTIMIATION IS     0.0000 SEC.
THE STORAGE USED IN OPTIMIATION IS    2739
The value of the solution is     42.70
Activity      Alternate
   1              3
   2              4
   3              6
   4              1
   5              7
   6              2
   7              5
```

Figure For N=7, Heptahedral MPG



Activities	1	2	3	4	5	6	7
1	-	45	0	12	0	17	0
2	45	-	64	0	94	0	0
3	0	64	-	13	0	0	23
4	12	0	11	-	0	11	0
5	0	94	0	0	-	0	24
6	17	0	0	13	0	-	0
7	0	0	23	0	24	0	-

6) For N=8,

```
THE NAME OF THE DATA FILE IS ravi7.dat
CPU TIME USED IN OPTIMIATION IS     0.0000 SEC.
THE STORAGE USED IN OPTIMIATION IS    3961
The value of the solution is     84.39
Activity      Alternate
   1              8
   2              3
   3              4
   4              1
   5              7
   6              2
   7              5
   8              6
```

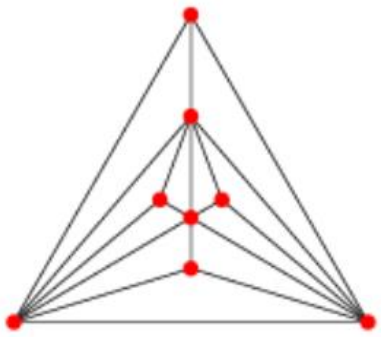
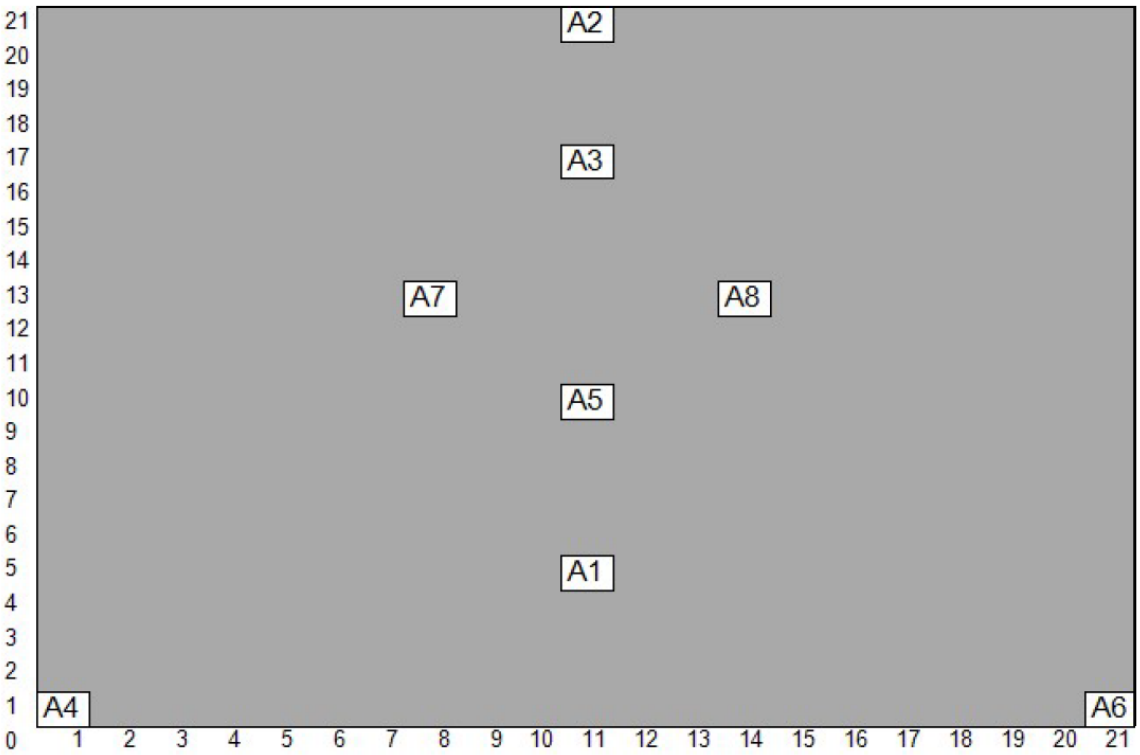


Figure N=8 triakis tetrahedral MPG



Activities	1	2	3	4	5	6	7	8
1	-	12	0	0	38	0	7	32
2	12	-	64	0	0	11	0	0
3	0	64	-	0	0	0	44	0
4	0	0	0	-	0	33	0	0
5	38	0	0	0	-	0	68	121
6	0	11	0	33	0	-	0	0
7	7	0	44	0	68	0	-	5
8	32	0	0	0	121	0	5	-

7) For N=9,

```
THE NAME OF THE DATA FILE IS ravi8.dat
CPU TIME USED IN OPTIMIATION IS     0.0000 SEC.
THE STORAGE USED IN OPTIMIATION IS    7243
The value of the solution is     77.82
Activity      Alternate
   1              8
   2              6
   3              4
   4              3
   5              9
   6              2
   7              1
   8              5
   9              7
```

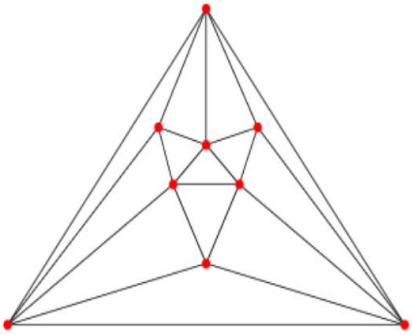
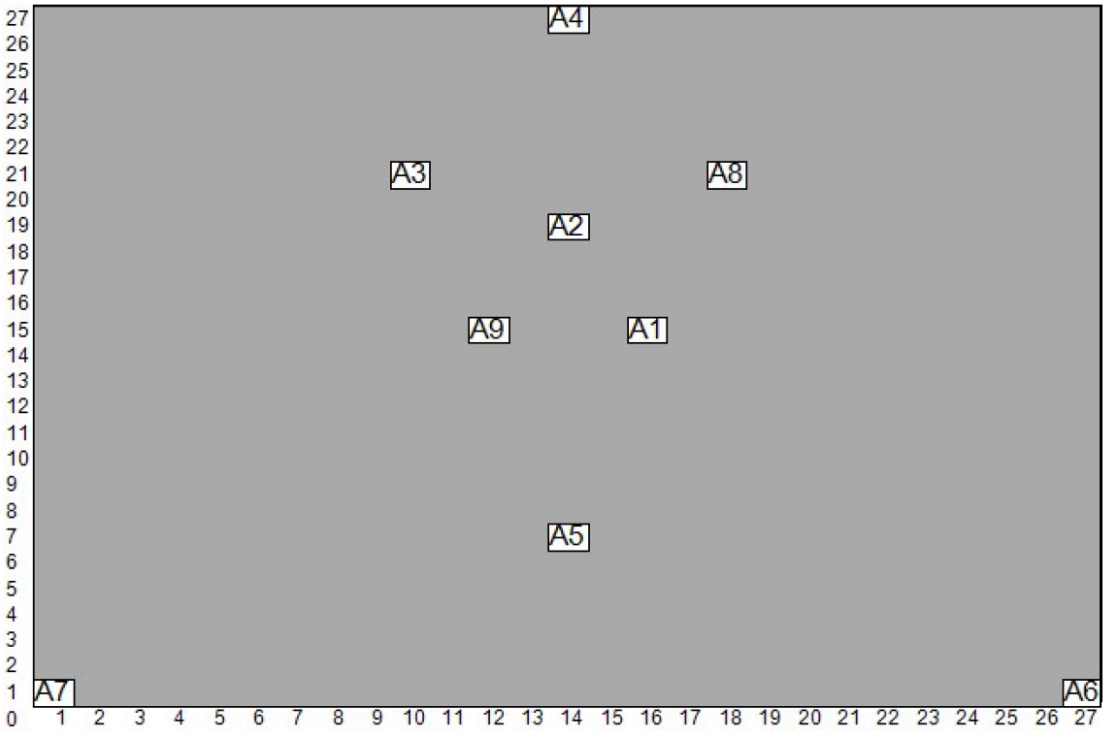


Figure N=9 Fritsch MPG



Activities	1	2	3	4	5	6	7	8	9
1	-	68	0	0	0	17	0	0	37
2	68	-	84	0	34	0	0	55	0
3	0	84	-	46	0	0	16	0	28
4	0	0	46	-	0	11	0	0	0
5	0	34	0	0	-	0	12	0	26
6	17	0	0	11	0	-	0	18	0
7	0	0	16	0	12	0	-	0	14
8	0	55	0	0	0	18	0	-	9
9	37	0	28	0	26	0	14	9	-

8) N=11

```
THE NAME OF THE DATA FILE IS ravi9.dat
CPU TIME USED IN OPTIMIATION IS     0.0000 SEC.
THE STORAGE USED IN OPTIMIATION IS   12987
The value of the solution is    104.60
Activity      Alternate
   1              6
   2             10
   3             11
   4              3
   5              8
   6              5
   7              9
   8              1
   9              4
  10              2
  11              7
```

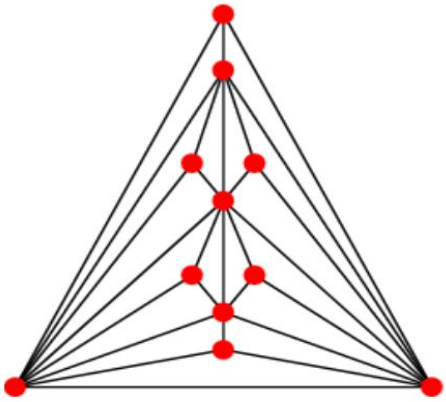
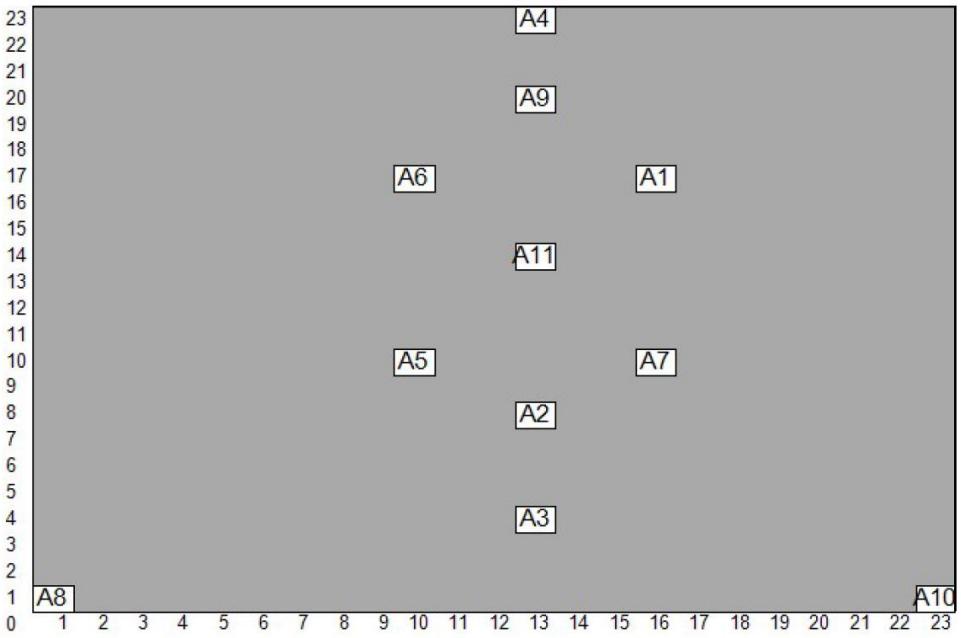


Figure N=11 Goldner-Harary MPG



Activities	1	2	3	4	5	6	7	8	9	10	11
1	-	36	0	0	0	0	0	67	0	0	54
2	36	-	14	0	54	0	32		0	0	0
3	0	14	-	7	0	0	0	13	0	4	0
4	0	0	0	-	0	27	0	0	62	0	0
5	0	54	0	0	-	0	18	0	0	0	19
6	0	0	0	27	0	-	0	0	24	0	0
7	0	32	0	0	18	0	-	0	0	21	0
8	67	0	13	0	0	0	0	-	0	0	17
9	0	0	0	62	0	24	0	0	-	0	18
10	0	0	4	0	0	0	21	0	0	-	4
11	54	0	0	0	19	0	0	17	18	4	-

9) For N=12

```
THE NAME OF THE DATA FILE IS rav10.dat
CPU TIME USED IN OPTIMIATION IS     0.0000 SEC.
THE STORAGE USED IN OPTIMIATION IS    9981
The value of the solution is    195.08
Activity      Alternate
   1              9
   2             12
   3              8
   4              2
   5              7
   6             11
   7             10
   8              6
   9              4
  10              5
  11              1
  12              3
```

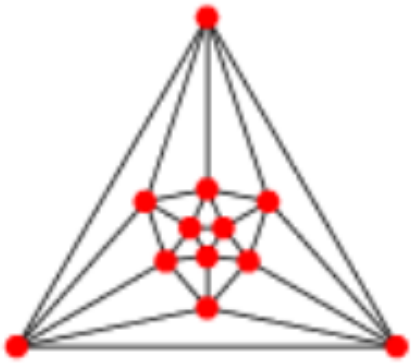
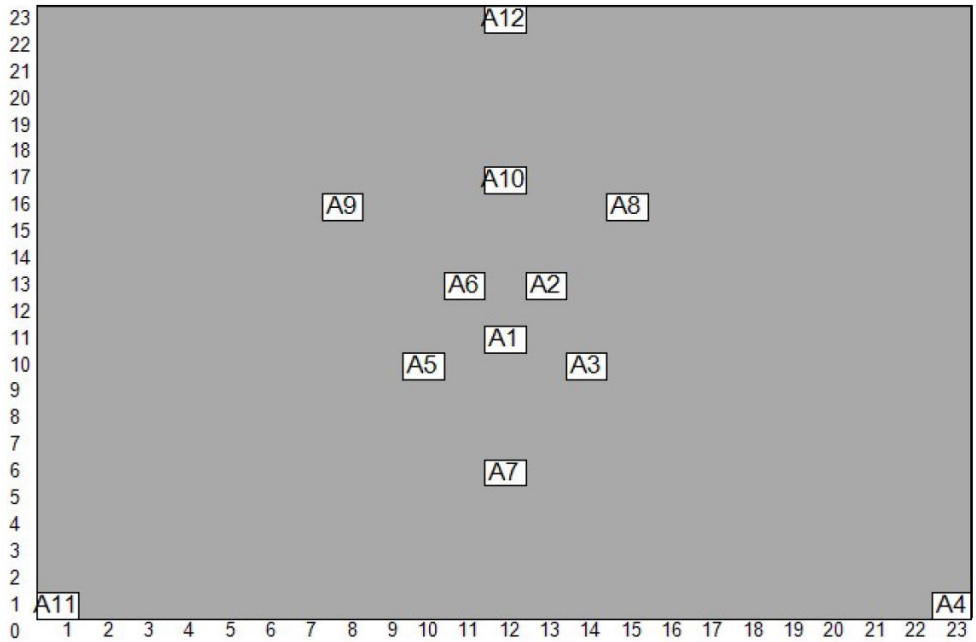


Figure For N=9, Icosahedral MPG



Activities												
1	-	90	10	23	43	0	0	0	0	0	0	0
2	90	-	0	0	0	88	0	0	0	0	0	0
3	10	0	-	0	0	0	26	16	0	0	0	0
4	23	0	0	-	0	0	0	0	0	0	0	0
5	43	0	0	0	-	0	0	0	0	0	0	0
6	0	88	0	0	0	-	0	0	1	0	0	0
7	0	0	26	0	0	0	-	0	0	0	0	0
8	0	0	16	0	0	0	0	-	0	96	0	0
9	0	0	0	0	0	1	0	0	-		29	37
10	0	0	0	0	0	0	0	96	0	-	0	0
11	0	0	0	0	0	0	0	0	29	0	-	0
12	0	0	0	0	0	0	0	0	0	37	0	-

10) For N=15

```
THE NAME OF THE DATA FILE IS ra012.dat
CPU TIME USED IN OPTIMIATION IS     0.0000 SEC.
THE STORAGE USED IN OPTIMIATION IS   18335
The value of the solution is    251.58
Activity      Alternate
   1              7
   2             15
   3              9
   4              8
   5             12
   6              2
   7              3
   8             14
   9              4
  10              1
  11             10
  12              5
  13             13
  14             11
  15             15
```

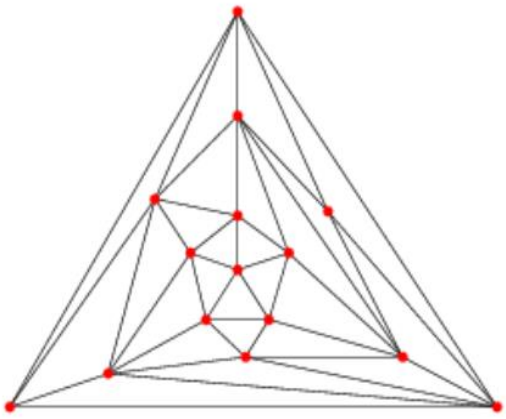
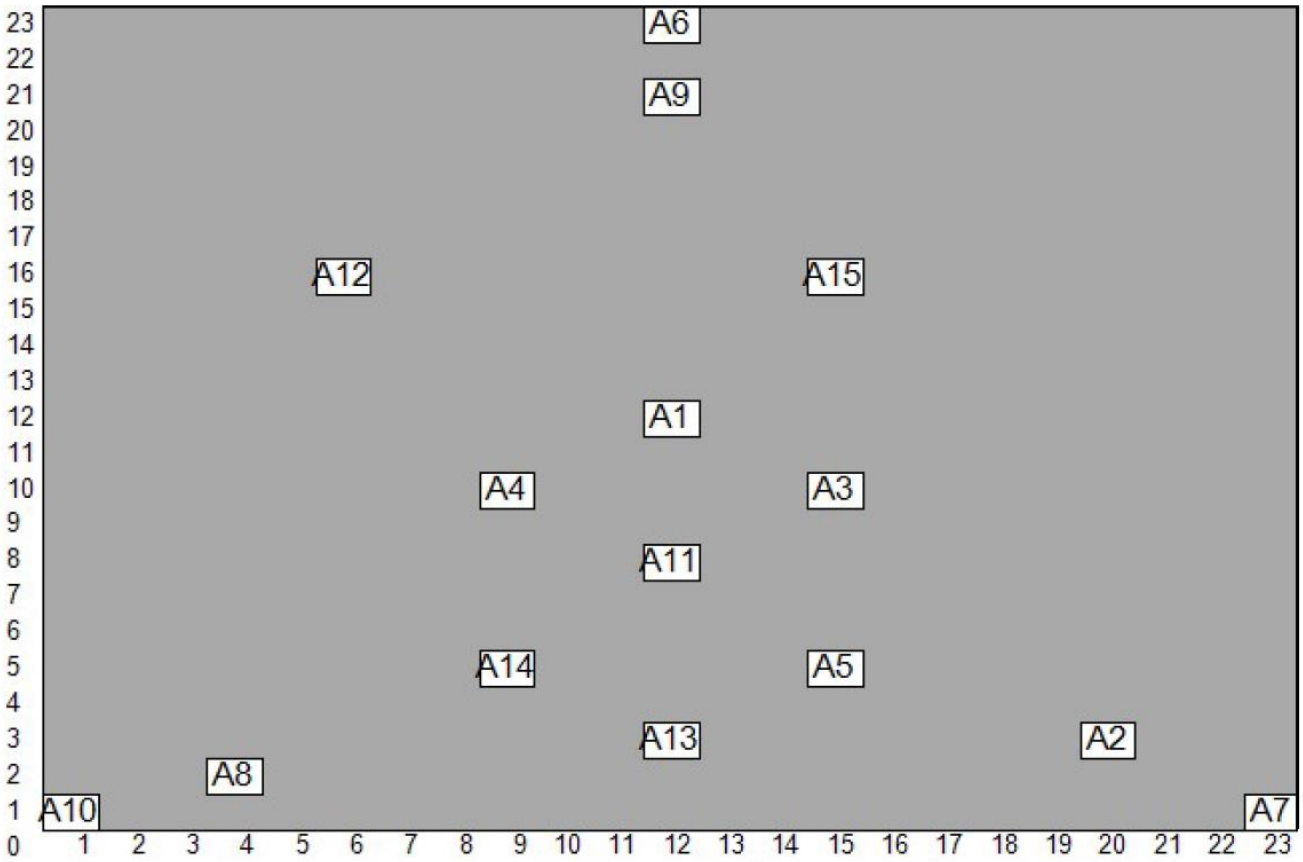


Figure N=15 Poussin MPG



11) For n = 17

```
THE NAME OF THE DATA FILE IS rav13.dat
CPU TIME USED IN OPTIMIATION IS     0.0000 SEC.
THE STORAGE USED IN OPTIMIATION IS   22047
The value of the solution is     40.64
Activity      Alternate
   1              2
   2              4
   3             12
   4              6
   5             13
   6             17
   7              9
   8              5
   9              3
  10             15
  11              7
  12              1
  13              8
  14             14
  15             10
  16             11
  17             16
```

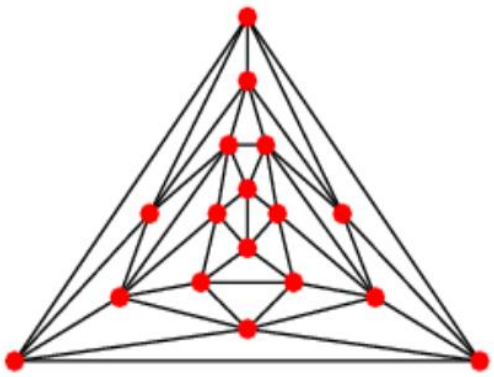
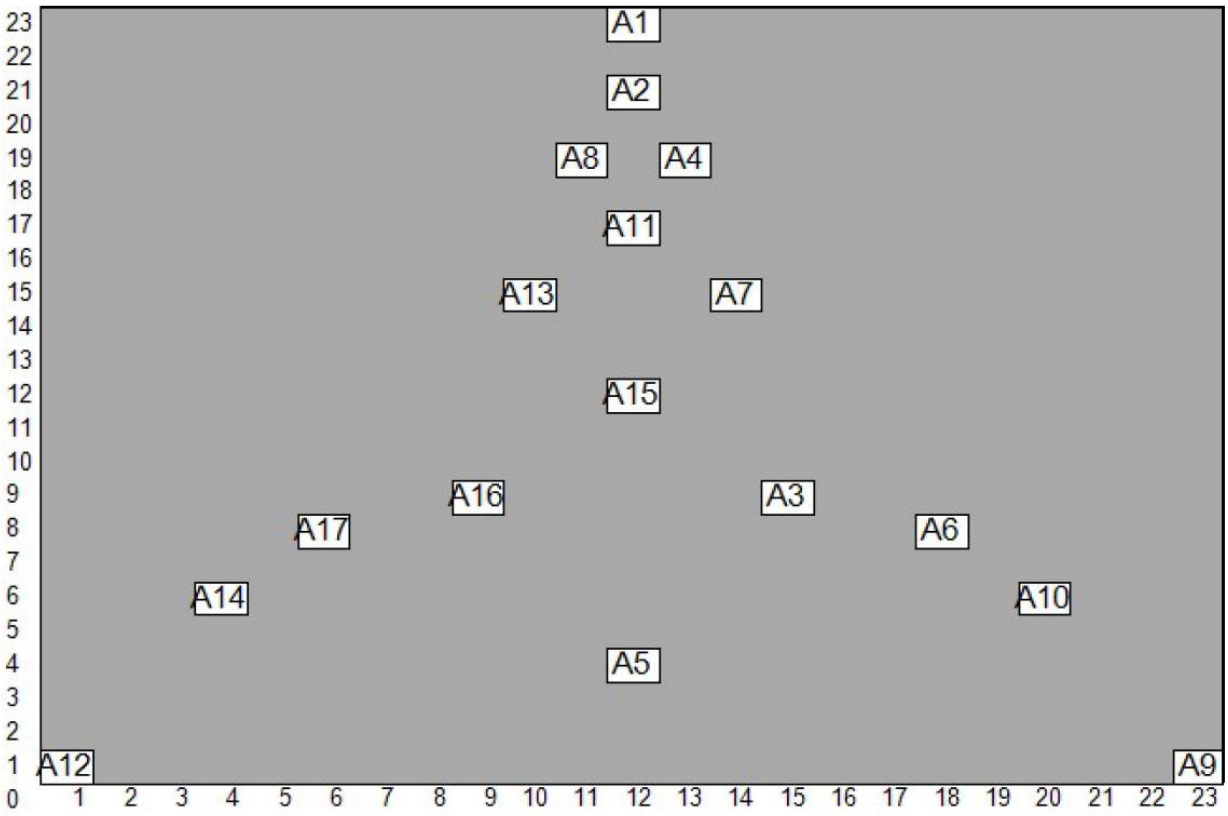


Figure N=17 Errera MPG



Activities	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
1	-	12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	12	-	0	4	0	0	0	6	0	0	0	0	0	0	0	0	0
3	0	0	-	0	0	6	0	0	0	0	0	0	0	0	0	0	0
4	0	4	0	-	0	0	0	26	0	0	0	0	0	0	0	0	0
5	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	6	0	0	-	0	0	0	14	0	0	0	0	0	0	0
7	0	0	0	0	0	0	-	0	0		0	0	2	0	0	0	0
8	0	6	0	26	0	0	0	-	0	0	0	14	0	0	0	0	0
9	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0
10	0	0	0	0	0	14	0	0	0	-	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	14	0	0	-	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0
13	0	0	0	0	0	0	2	0	0	0	0	0	-	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	-	0	0	14
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	3	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	-	2
17	0	0	0	0	0	0	0	0	0	0	0	0	0	14	0	2	-

12) For N=23

```
THE NAME OF THE DATA FILE IS rav14.dat
CPU TIME USED IN OPTIMIATION IS     0.0000 SEC.
THE STORAGE USED IN OPTIMIATION IS   40851
The value of the solution is     42.53
Activity      Alternate
   1             23
   2             14
   3             22
   4             21
   5             15
   6             20
   7             18
   8             10
   9             17
  10              9
  11             16
  12             12
  13              6
  14              3
  15              5
  16             11
  17              4
  18             19
  19              2
  20             13
  21              1
  22              8
  23              7
```

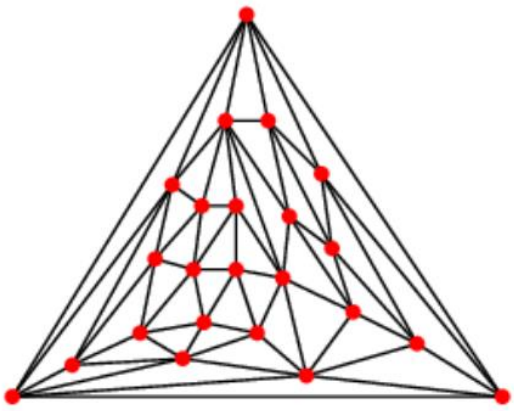
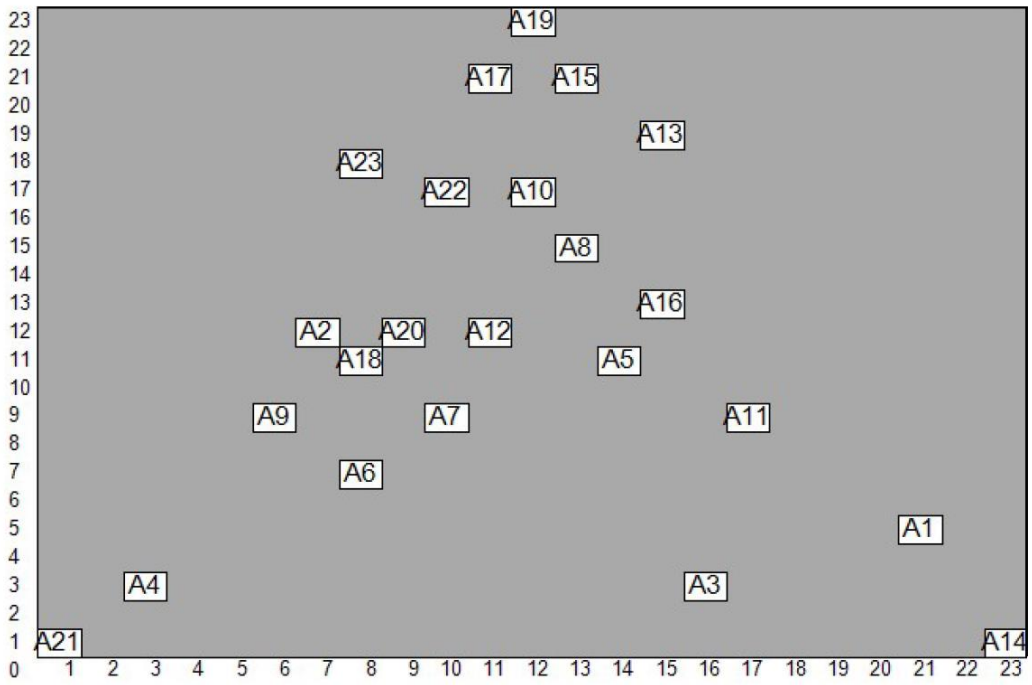


Figure N=23 Kittell MPG



Activities	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
1	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	11	0	0	0	0	0
3	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0	14	0	0	0	0	0	0	0
6	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	-	0	14	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	14	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	0	15	0	0	0	0	0	0
16	0	0	0	0	14	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	15	0	-	0	0	0	0	0	0
18	0	11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	0	12	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	12	0	-	0	0	0
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	14
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	14	-

13) For N=25

```
THE NAME OF THE DATA FILE IS rav15.dat
CPU TIME USED IN OPTIMIATION IS     0.0000 SEC
THE STORAGE USED IN OPTIMIATION IS   59211
The value of the solution is    183.97
Activity      Alternate
   1              25
   2              21
   3               9
   4              23
   5              24
   6              18
   7              22
   8              15
   9               7
  10              17
  11              13
  12              19
  13              14
  14              16
  15               5
  16              12
  17              10
  18              20
  19               4
  20               3
  21              11
  22               2
  23               6
  24               8
  25               1
```

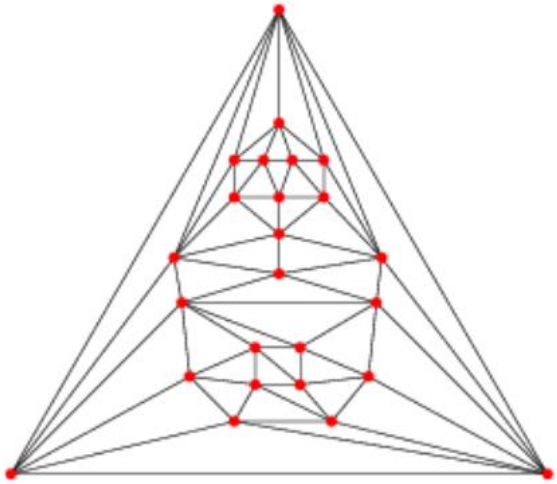
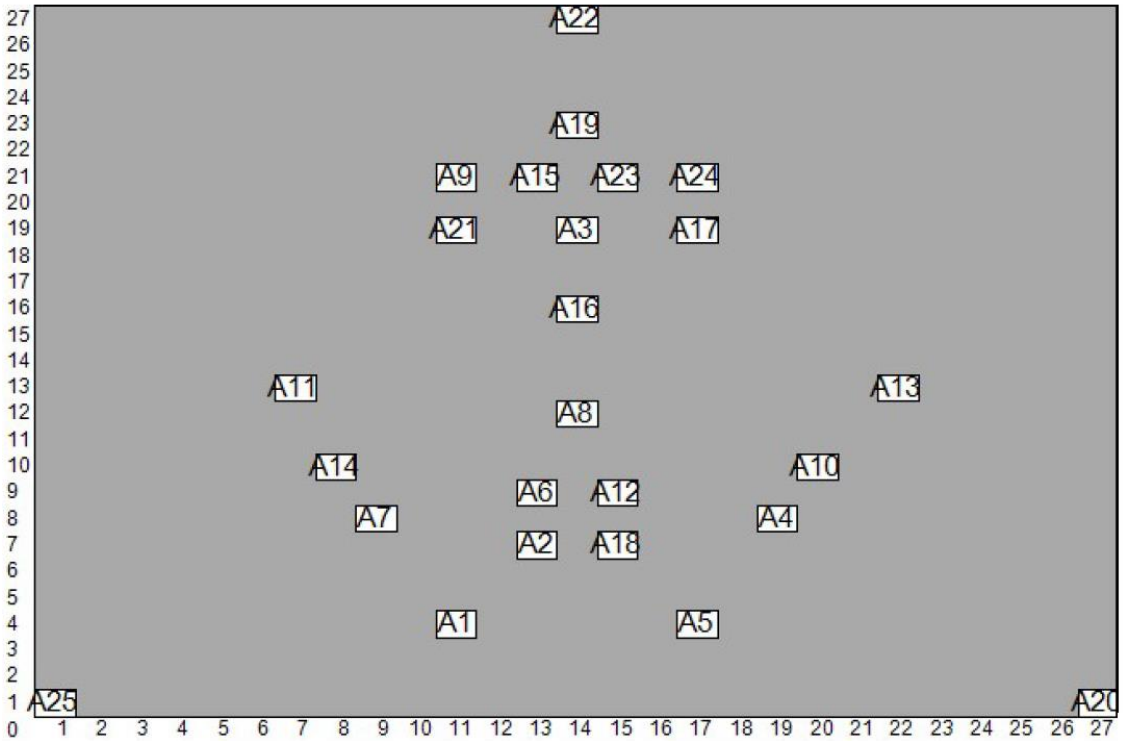


Figure For N= 25Heawood Four-Color MPG



Activities	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
1	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	54	0	0	0	0	0	0	0
3	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	-	0	0	0	0	0	34	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	-	0	0	0	0	0	0	27	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	-	0	0	0	0	0	19	0	0	0	0	0	44	0	0	0	0
10	0	0	0	34	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	-	0	0	22	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	91	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	27	0	0	0	0	22	0	0	-	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	19	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0
18	0	54	0	0	0	0	0	0	0	0	0	91	0	0	0	0	0	-	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0
21	0	0	0	0	0	0	0	0	44	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	62	0
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	62	-	0
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-

14) For N=26

```
THE NAME OF THE DATA FILE IS rav16.dat
CPU TIME USED IN OPTIMIATION IS     0.0000 SEC.
THE STORAGE USED IN OPTIMIATION IS   71933
The value of the solution is    418.50
Activity      Alternate
   1             26
   2             23
   3             21
   4             22
   5             24
   6              8
   7             15
   8              7
   9             16
  10             19
  11             12
  12              6
  13              5
  14             11
  15             20
  16              9
  17              4
  18              3
  19             17
  20              2
  21             25
  22             10
  23              1
  24             18
  25             13
  26             14
```

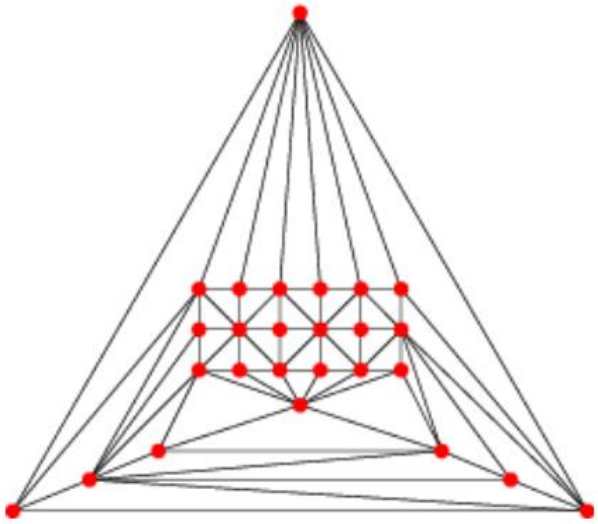
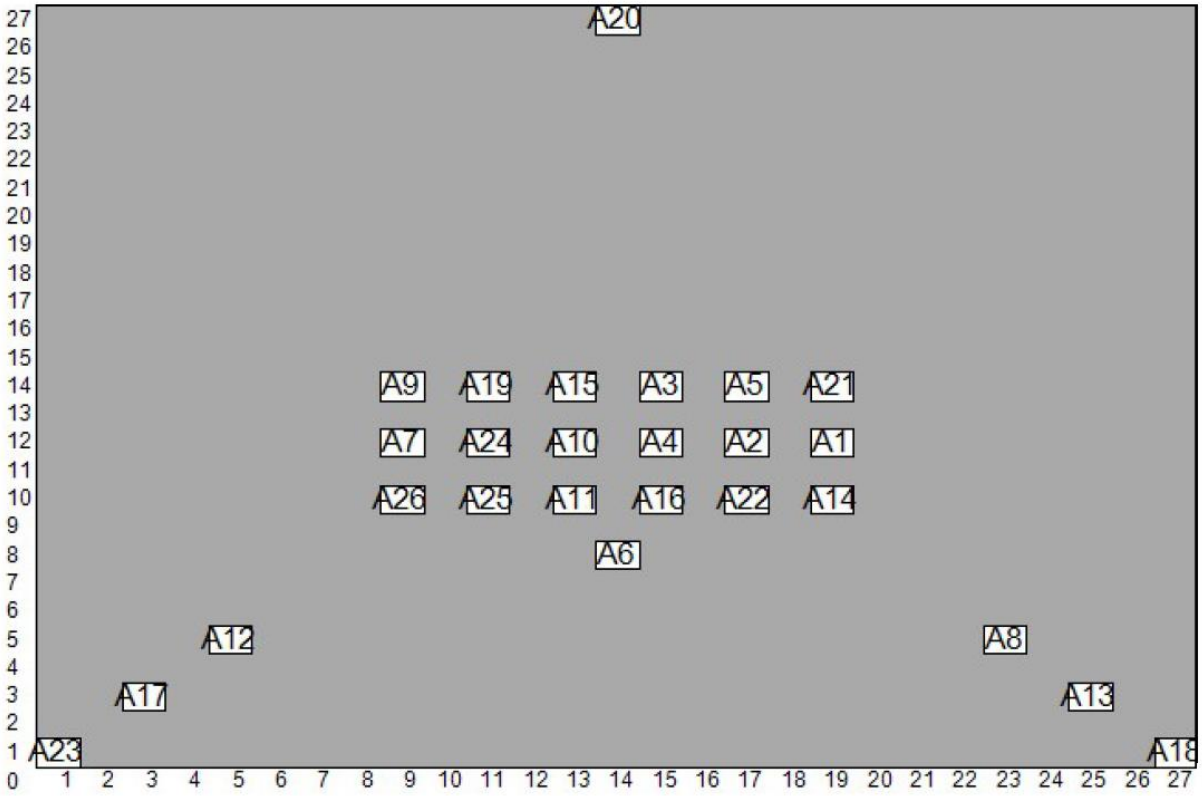


Figure For N= 26, Disdyakis Dodecahedral Graph



Activities	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
1	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	43	0	0	0	0	0
2	0	-	0	33	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	-	0	0	0	0	0	0	0	0	0	0	0	65	0	0	0	0	0	0	0	0	0	0	0
4	0	33	0	-	0	0	0	0	0	69	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	91	0	0	0	0	0
6	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	37	0	0
8	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	72	0	0	0	0	0	0	0
10	0	0	0	65	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	73	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0	78	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	49	0	0	0	0
15	0	0	65	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0	56	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	73	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	72	0	0	0	0	0	0	0	0	0	-	0	0	0	0	87	0	0
20	0	0	0	0	91	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0
21	43	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0	0	49	0	0	0	0	0	0	0	-	0	0	0	0
23	0	0	0	0	0	0	0	0	0	0	0	78	0	0	0	0	0	0	0	0	0	0	-	0	0	0
24	0	0	0	0	0	0	37	0	0	0	0	0	0	0	0	0	0	0	87	0	0	0	0	-	73	0
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	73	-	89
26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	89	-

15) For N= 32

```
THE NAME OF THE DATA FILE IS rav17.dat
CPU TIME USED IN OPTIMIATION IS     0.0000 SEC.
THE STORAGE USED IN OPTIMIATION IS  115593
The value of the solution is    250.94
Activity      Alternate
   1             16
   2             10
   3             31
   4             20
   5             14
   6             13
   7              9
   8             19
   9             15
  10             32
  11             30
  12             12
  13             29
  14             28
  15             27
  16             26
  17             25
  18             24
  19             23
  20             22
  21             21
  22             17
  23              7
  24             11
  25              6
  26              8
  27              5
  28              4
  29              3
  30              2
  31             18
  32              1
```

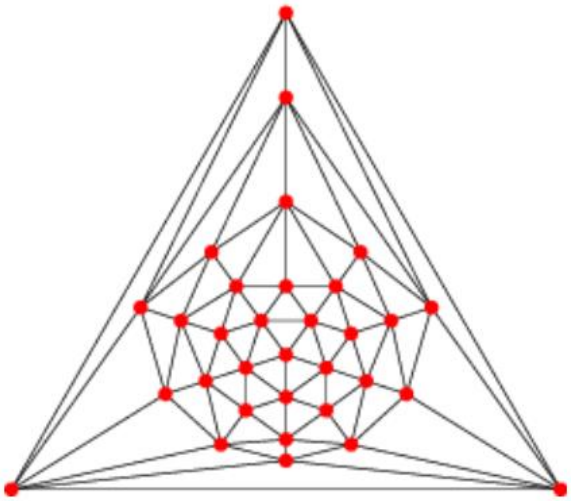
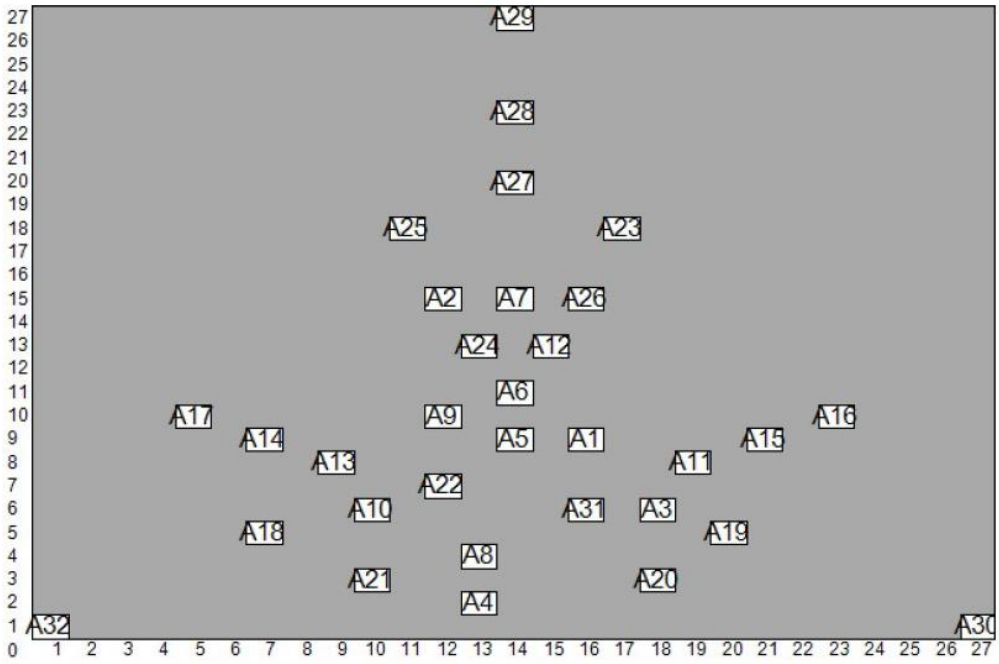


Figure For N=32, Pentakis Dodecahedral Graph



Activities	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
1	-	0	0	0	68	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	-	0	0	0	0	44	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	91	0
4	0	0	0	-	0	0	0	69	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	68	0	0	0	-	49	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	87	-	0	0	39	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	44	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	87	0	0	0	0	0	0
8	0	0	0	69	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	39	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	59	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0	0	0	59	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0
26	0	0	0	0	0	0	87	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0
27	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0
28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0
29	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0
30	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	0	0
31	0	0	91	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	0
32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-

APPENDIX B

VORONOI AND BLOCK DIAGRAMS GENERATED

Voronoi Diagram Generation

The Following section represents Voronoi diagram constructed through Delaunay triangulation for named MPG graph using their Euclidian coordinate. The Graphs are made using Mathematica 11.3 inbuilt graphical module function for the input coordinates of each MPG.

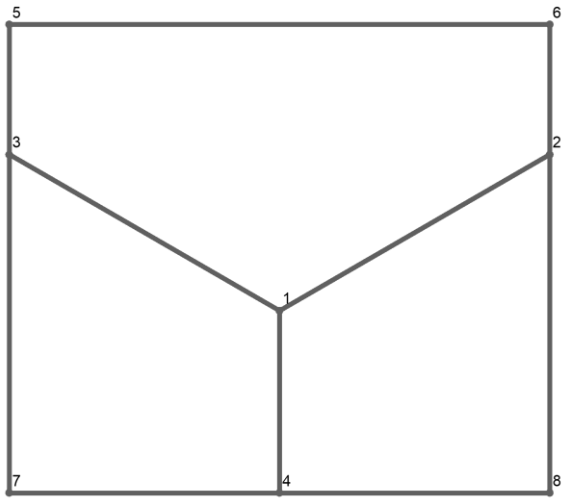


Figure For N=3 Voronoi diagram for Triangle MPG.

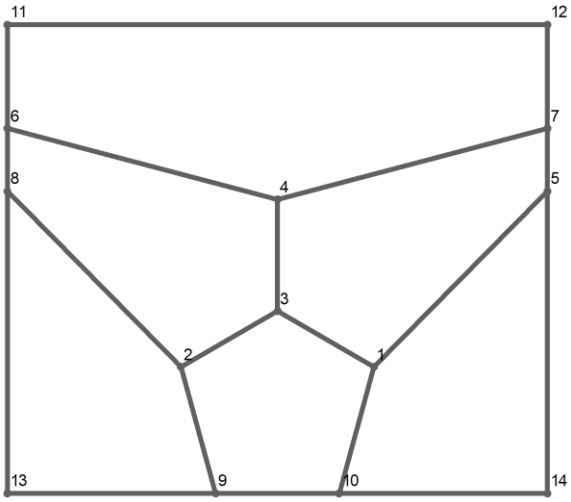


Figure For N=6 Voronoi diagram for Octahedral MPG.

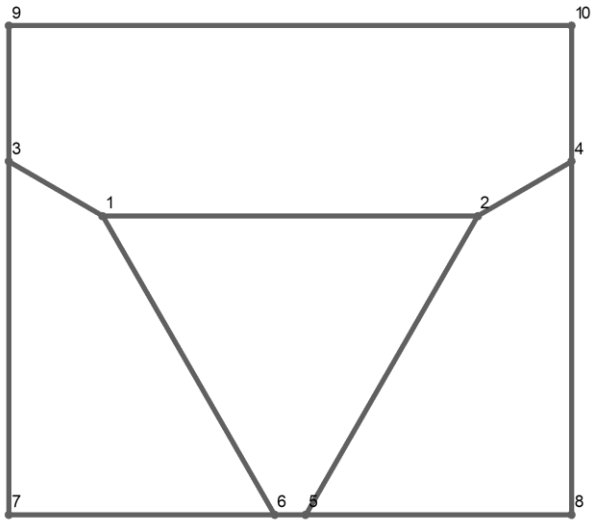


Figure For N=4 Voronoi diagram for Tetrahedral MPG

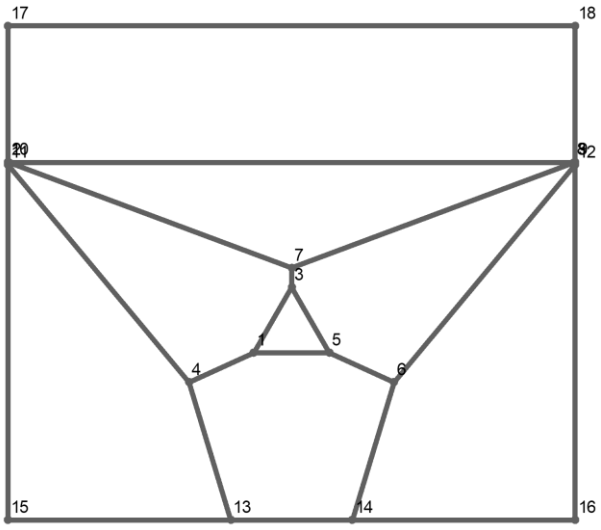


Figure For N=8 Voronoi diagram for Triaskis Tetrahedral MPG.

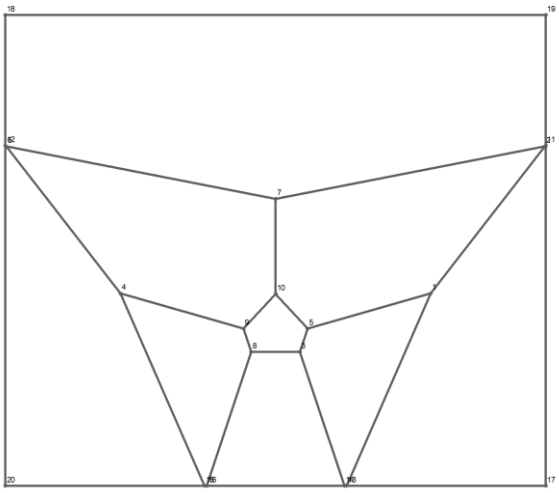


Figure For N=9 Voronoi diagram for Fritsch Named MPG.

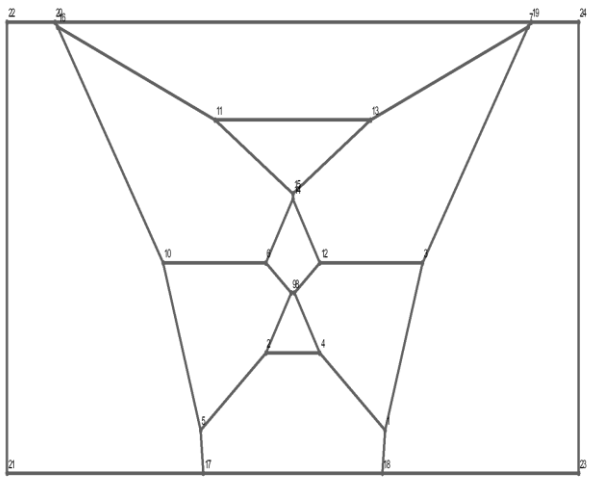


Figure For N=11 Voronoi diagram Goldner-Harary MPG.

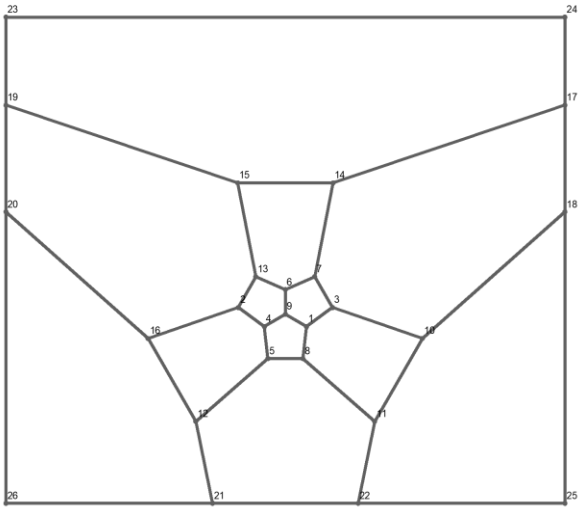


Figure For N=12 Voronoi diagram for icosahedral MPG

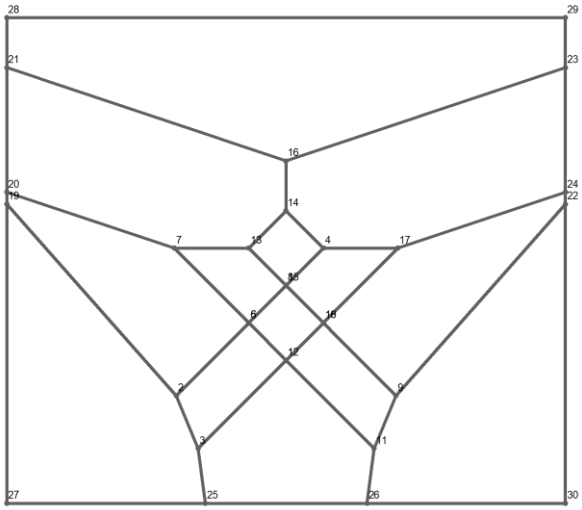


Figure For N= 14 Voronoi diagram for tetrakis hexahedral MPG

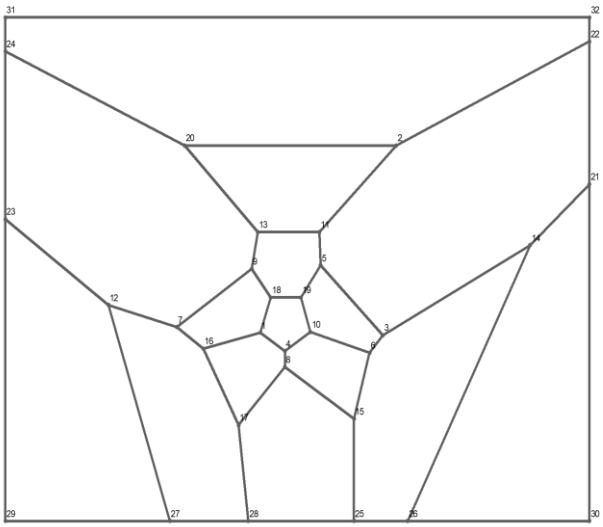


Figure For N= 15 Voronoi diagram for Poussin MPG

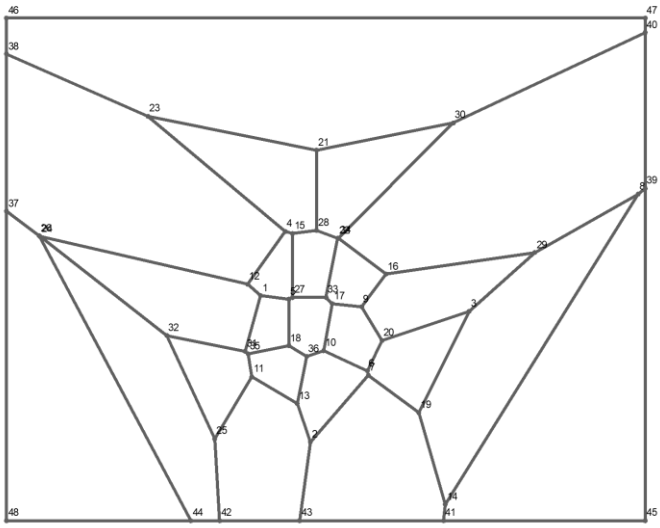


Figure For N= 23 Voronoi diagram for Kittell MPG

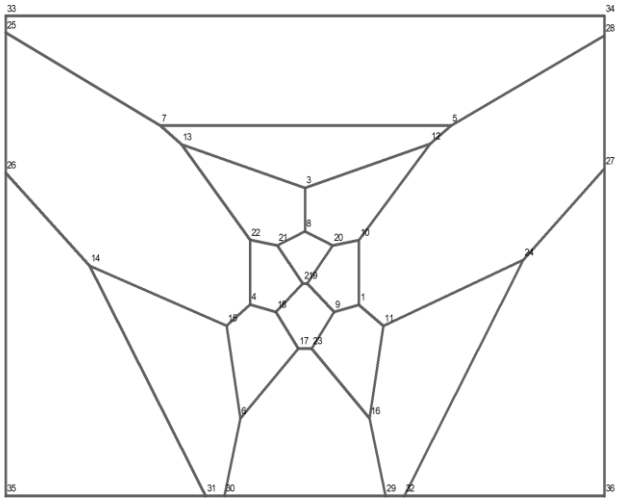


Figure For N= 17 Voronoi diagram for Errera MPG

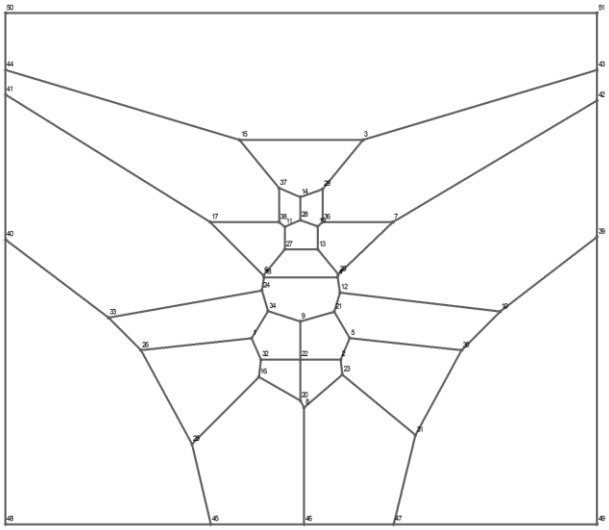


Figure For N= 25 Voronoi Diagram for Heawood four-color MPG

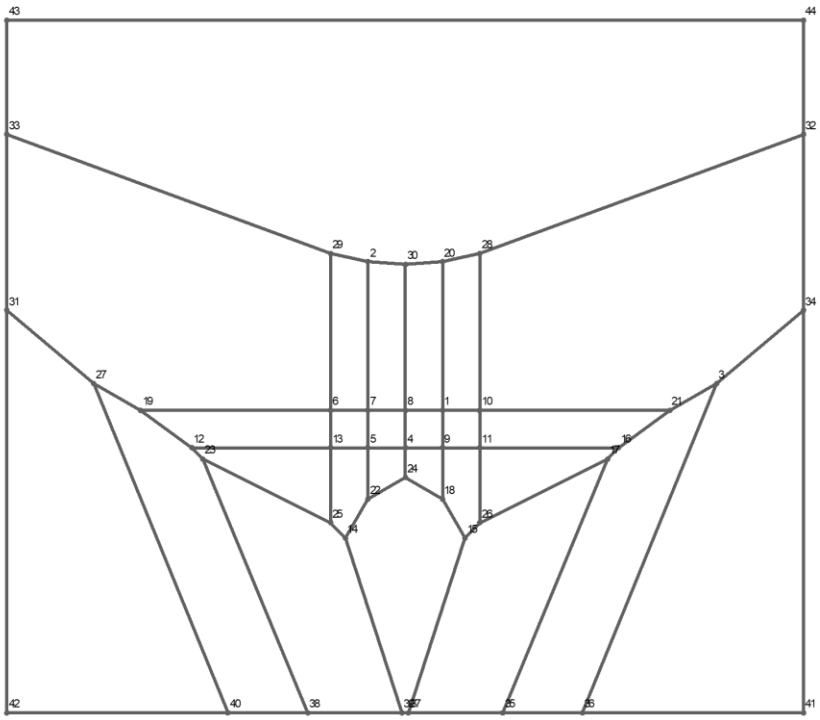


Figure For N=26 Voronoi diagram for Disdyakis dodecahedral MP

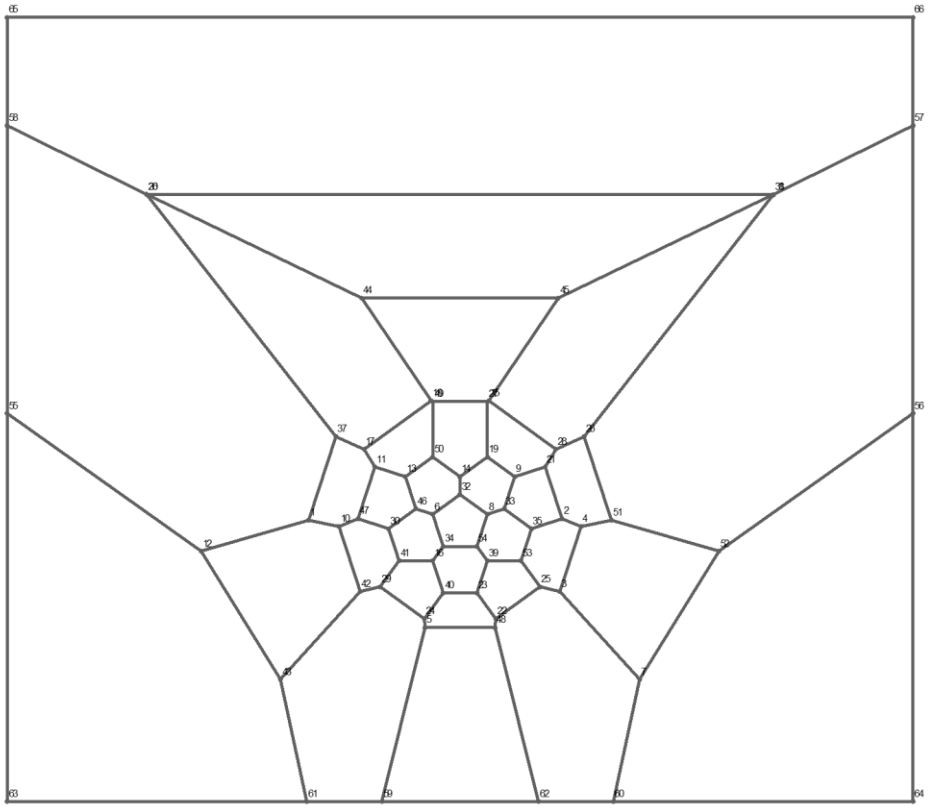


Figure For N= 32 Voronoi diagram for Pentakis dodecahedra MPG

Block Digram Genration

Following Section represents the final graphical block plan layout, which is plotted using Mathematica 11.3 inbuilt graphical module. The coordinates for the block plan from related Voronoi diagram are created through formulation represented in chapter 4. The formulation is used to generate coordinates for plotting the Block plan. Lingo optimization tool has been used to generate coordinates for given value of area for these block plan.

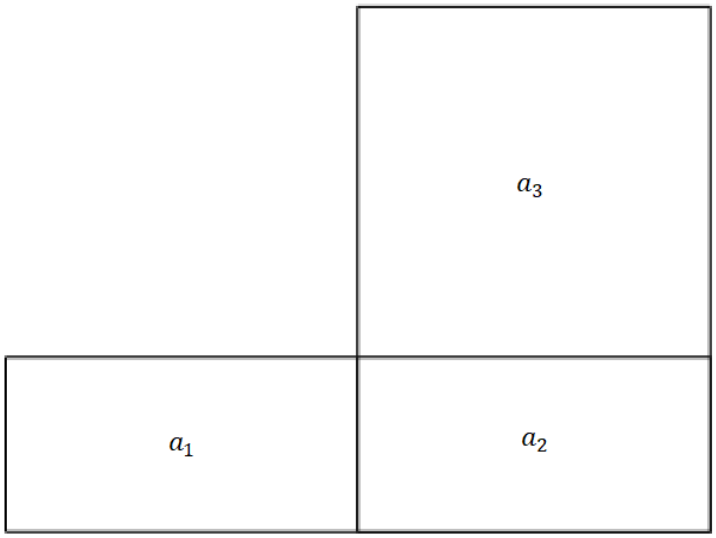


Figure Diagram For N=3, Triangle block diagram conversion of Voronoi triangulation

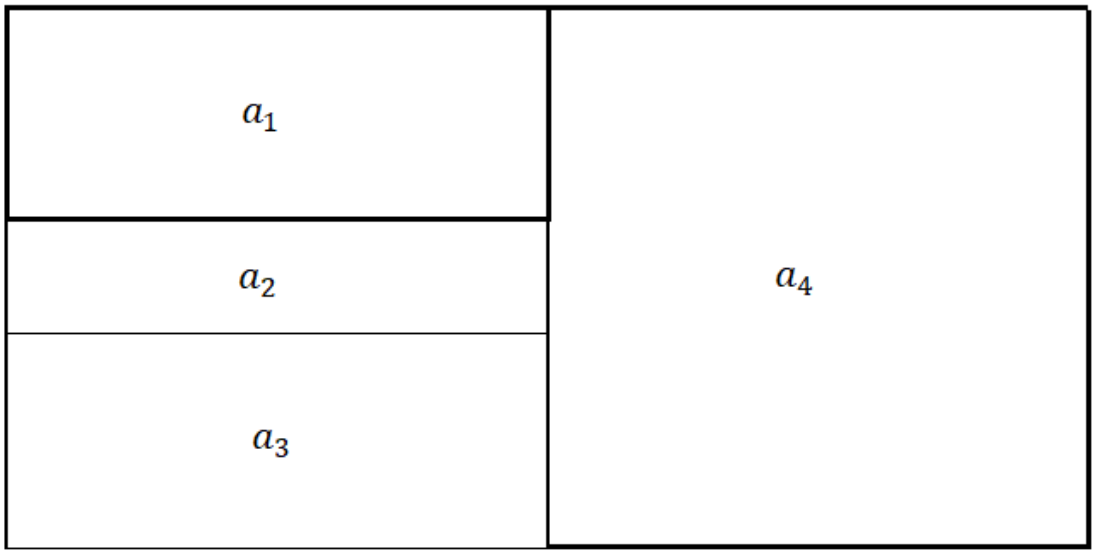


Figure Diagram For N=4, Tetrahedral block diagram conversion of Voronoi triangulation

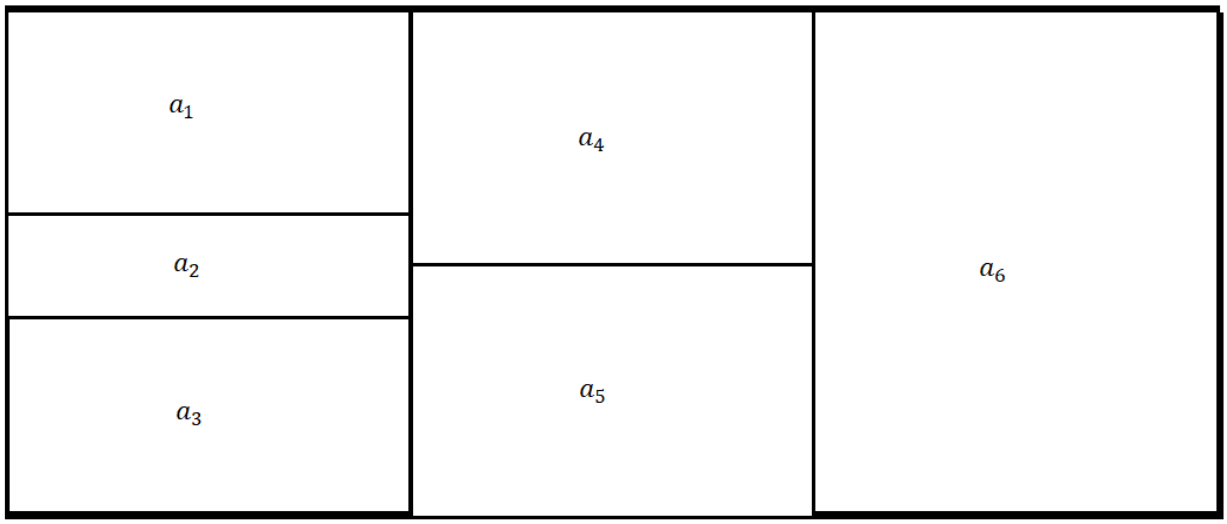


Figure Diagram For N=6, Octahedral block diagram conversion of Voronoi triangulation

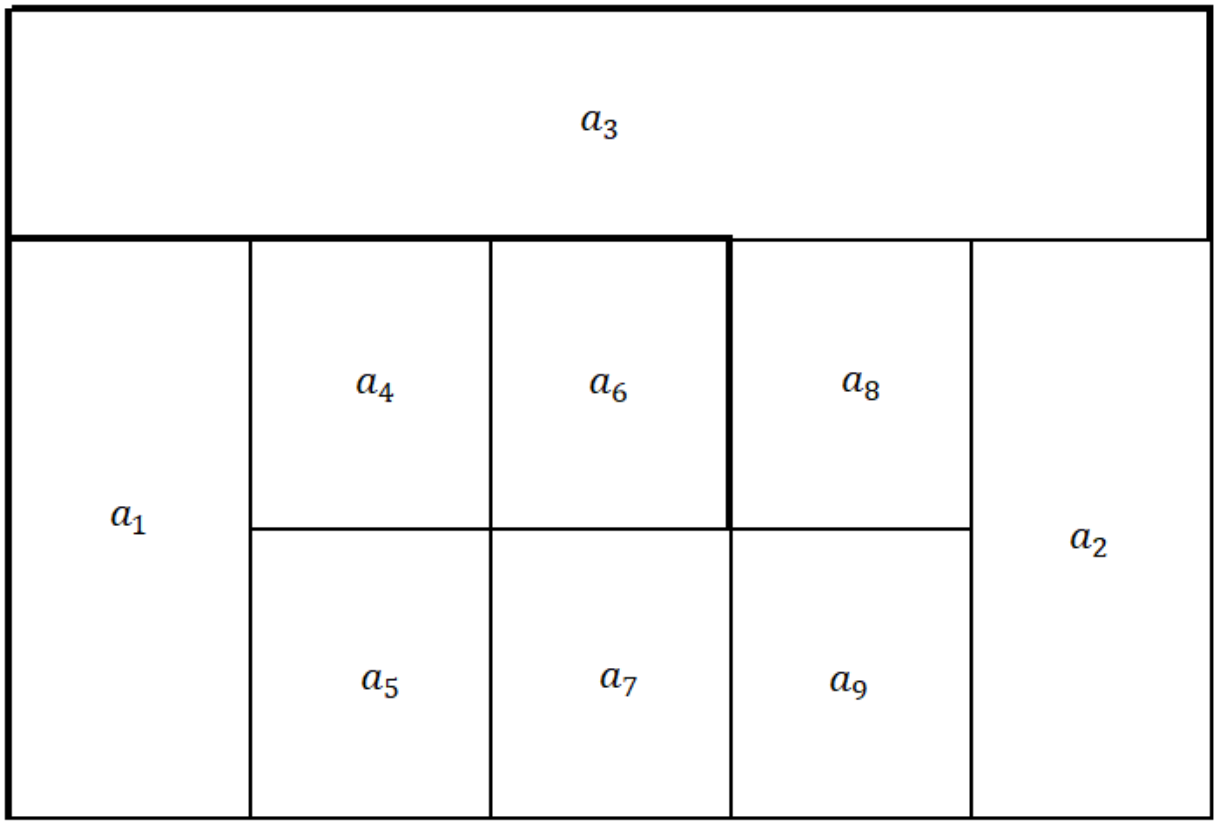


Figure Diagram For N=9, Fritsch block diagram conversion of Voronoi triangulation

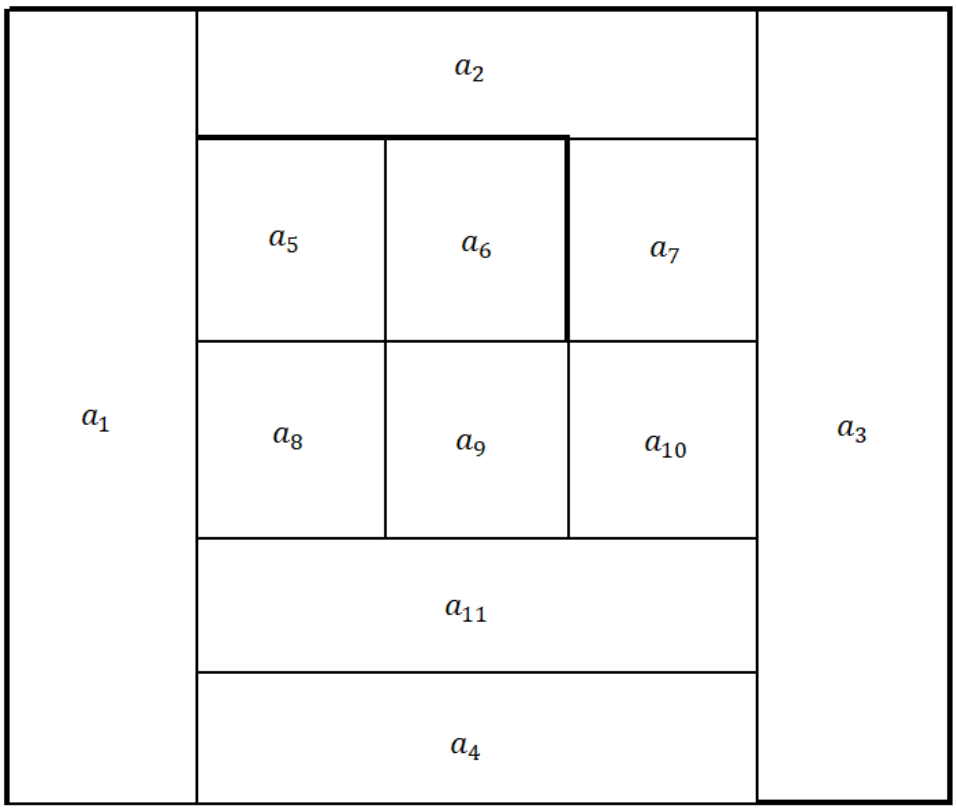


Figure Diagram For N=11, Goldner-Harary block diagram conversion of Voronoi triangulation

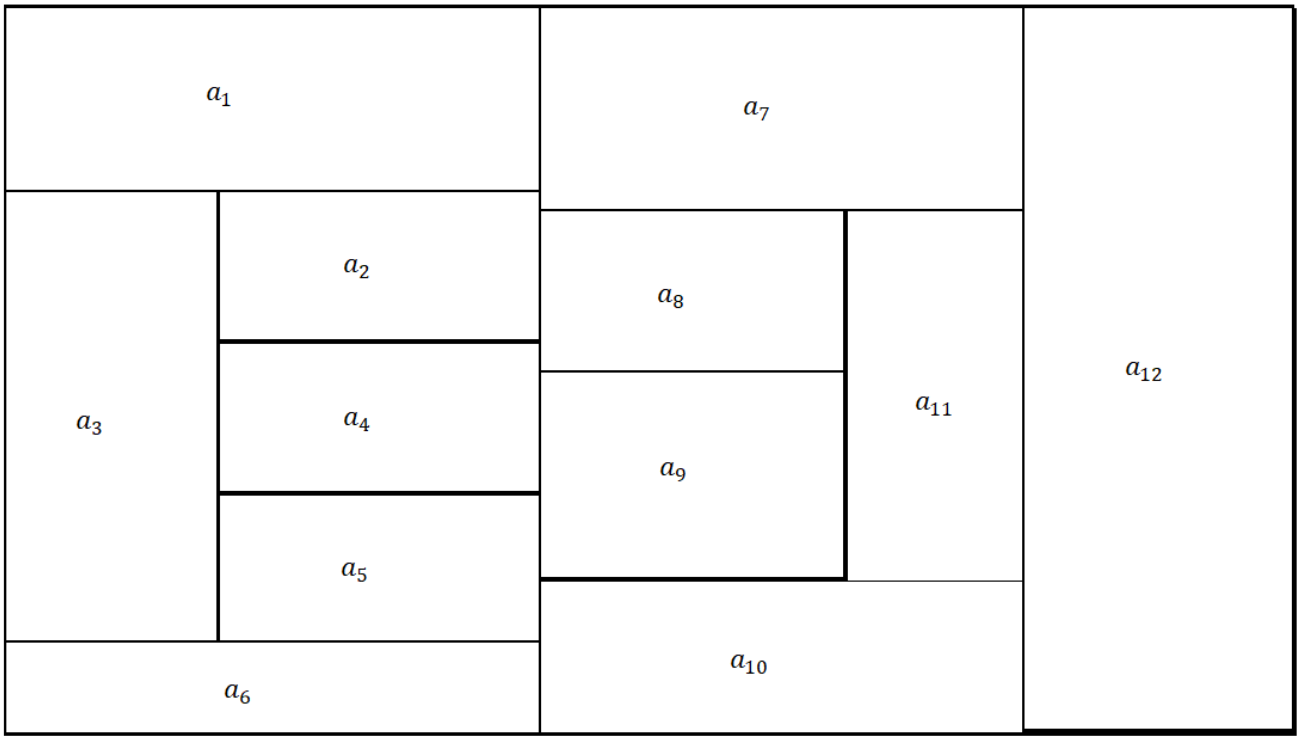


Figure Diagram For N=12, icosahedral block diagram conversion of Voronoi triangulation

Appendix C

RSMT GRAPHS

Following Section represents a visual exhibition of the graphical block plan layout with RSMT (rectilinear Steiner minimal tree) solution embedded in it. The RSMT solution represented here is created using GeoSteiner 5.1 package. The layout is plotted using Autodesk Inventor 2019's inbuilt 2-D plotting solution. The block plan represented here are alike to the block plan created in second phase, through expanding the Voronoi diagram for given value of n with user defined area input.

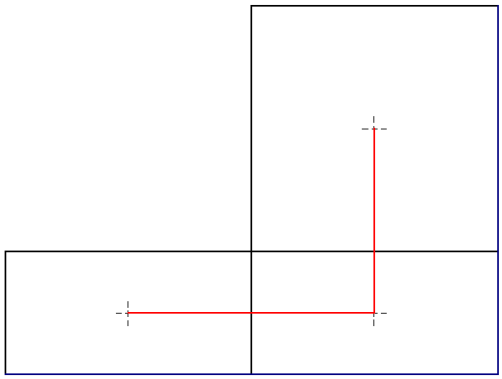


Figure N=3 Picture of Block Plan Diagram with RSMT solution for MHS embedded.

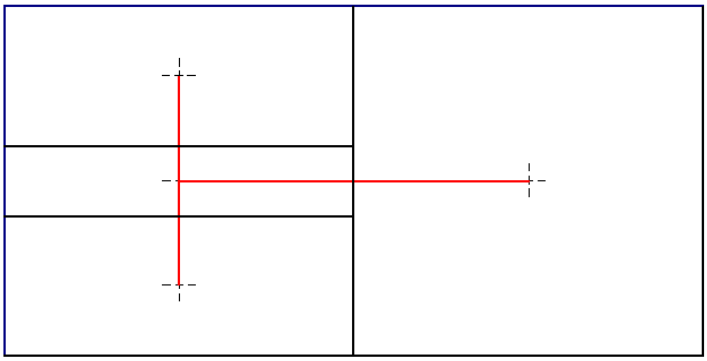


Figure N=4 Picture of Block Plan Diagram with RSMT solution for MHS embedded.

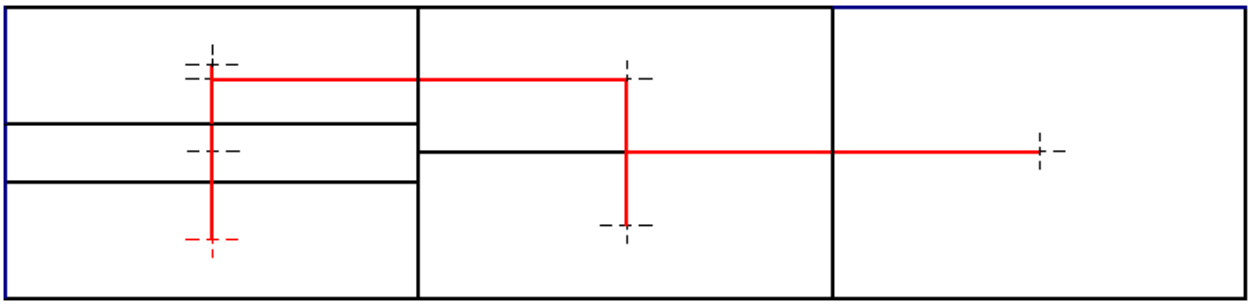


Figure N=6 Picture of Block Plan Diagram with RSMT solution for MHS embedded.

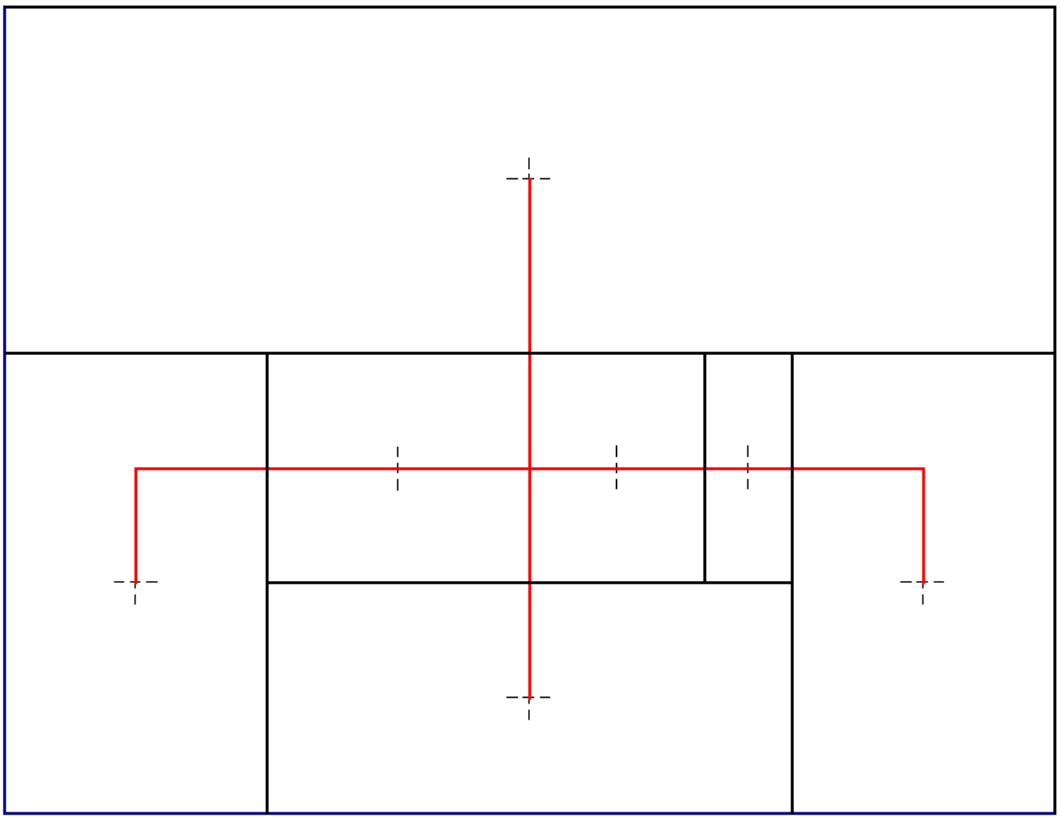


Figure N=7 Picture of Block Plan Diagram with RSMT solution for MHS embedded.

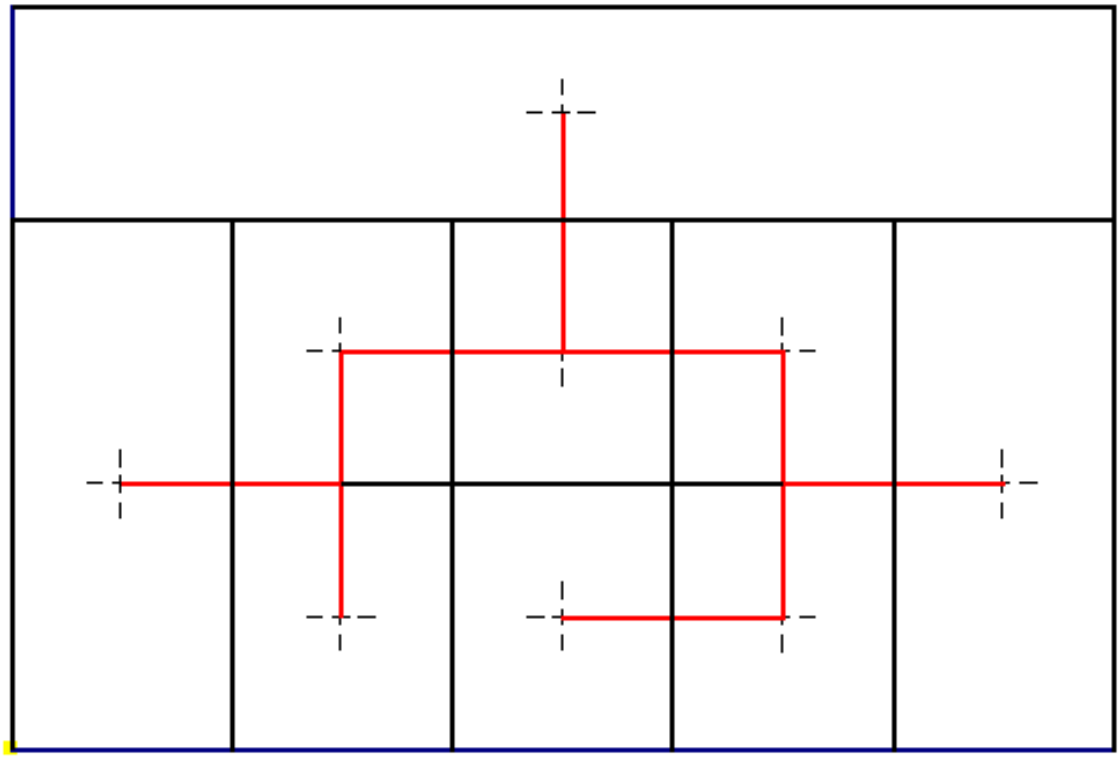


Figure N=9 Picture of Block Plan Diagram with RSMT solution for MHS embedded.

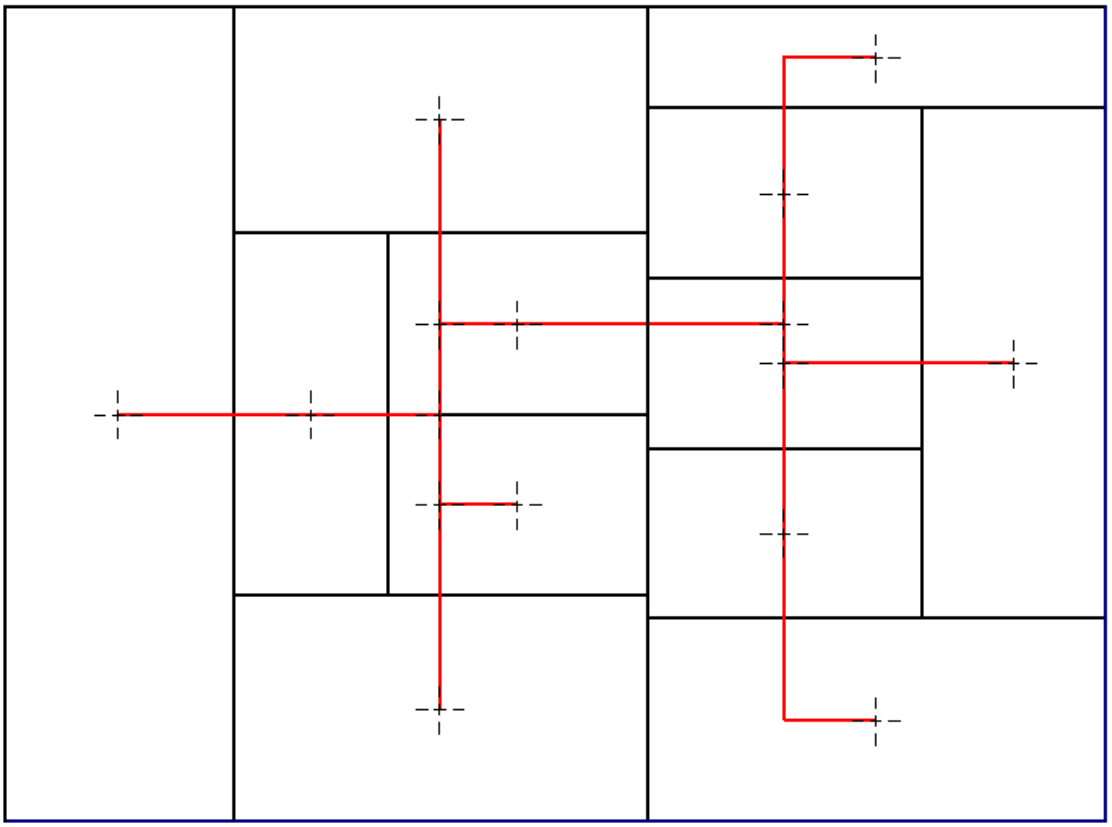


Figure N=12 Picture of Block Plan Diagram with RSMT solution for MHS embedded.

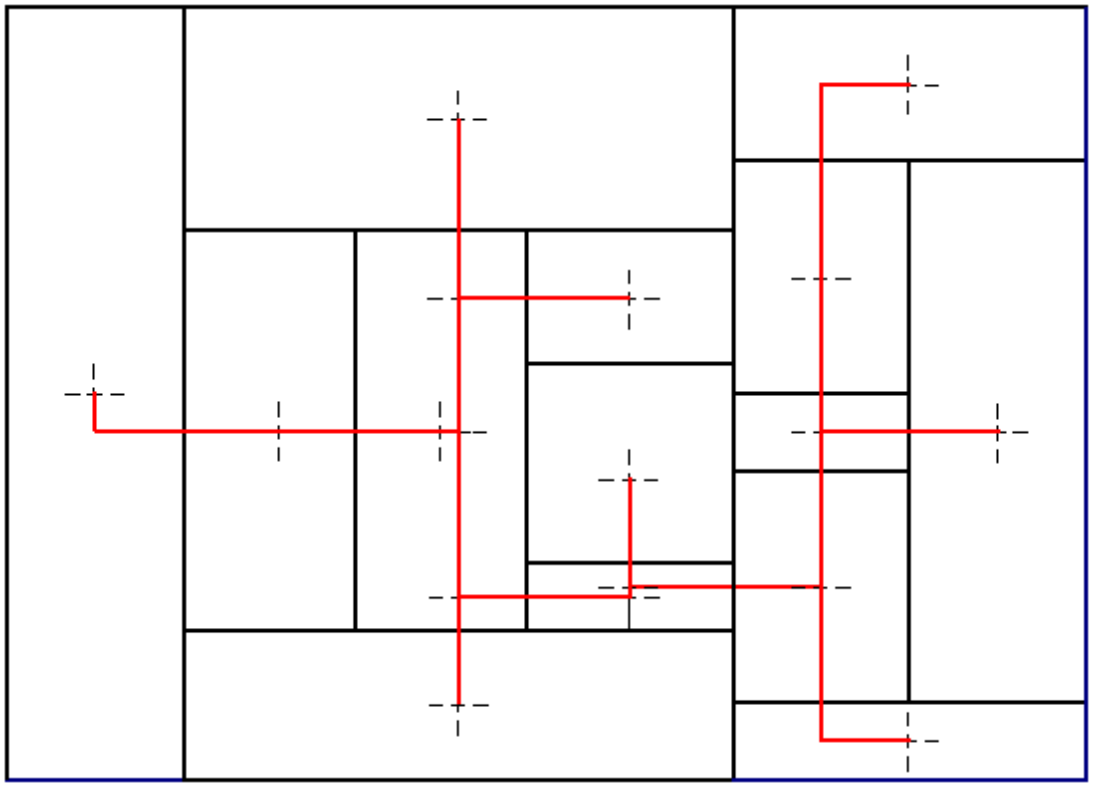


Figure N=14 Picture of Block Plan Diagram with RSMT solution for MHS embedded.

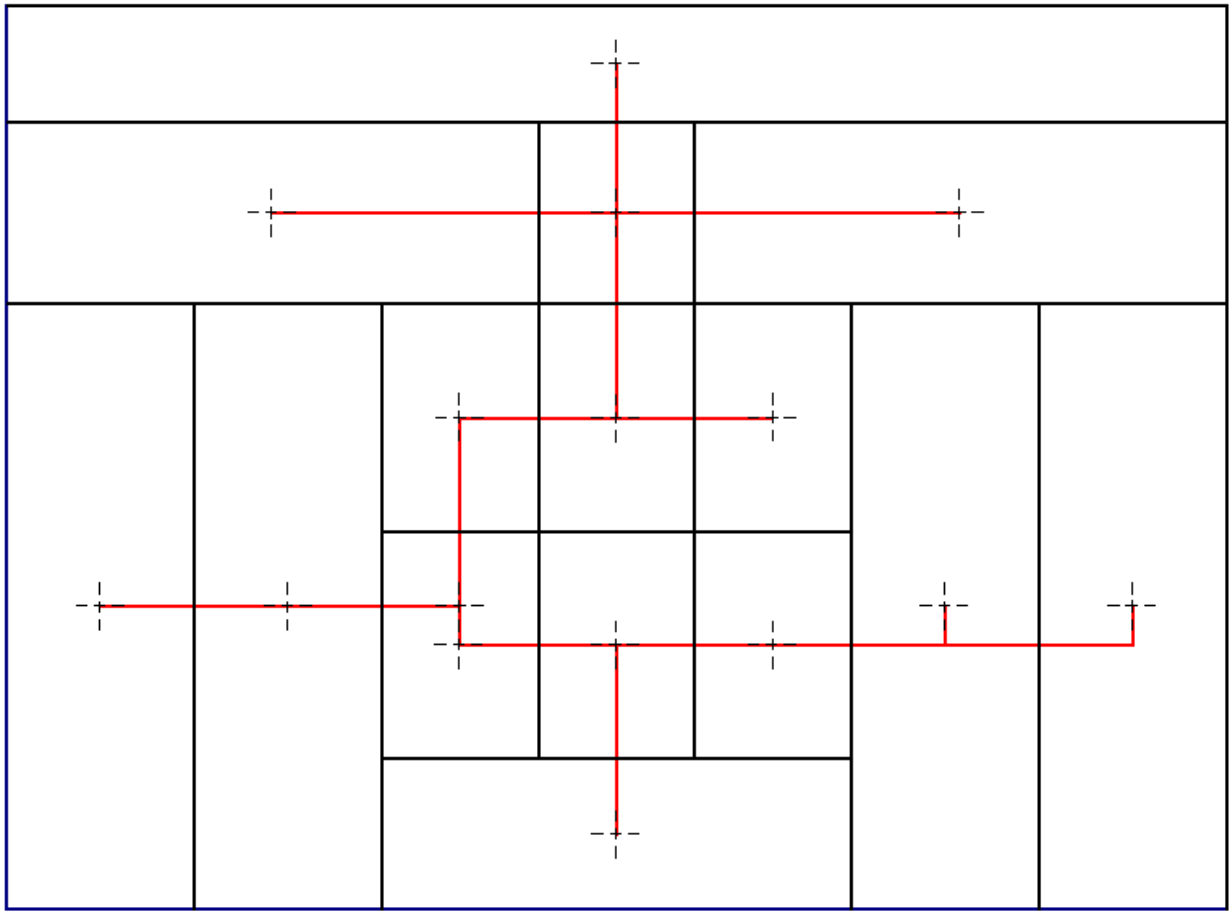


Figure N=15 Picture of Block Plan Diagram with RSMT solution for MHS embedded.

BIBLIOGRAPHY

[1] A. Garcia-Diaz and J. M. Smith, Facilities planning and design, Pearson Prentice Hall, 2008.

[2] "Facility Layout and Design," 30 November 1899. [Online]. Available: https://www.inc.com/encyclopedia/facility-layout-and-design.html. [Accessed 25 June 2018].

[3] Y. Xie, S. Zhou, Y. Xiao, S. Kulturel-Konak and A. Konak, "A β-accurate linearization method of Euclidean distance for the facility layout problem with heterogeneous distance metrics," *European Journal of Operational Research*, vol. 265, no. 1, pp. 26-38, 2018.

[4] K. Watson, *Graph theoretic facility layout design and evaluation: theoretical and practical considerations: a thesis presented in partial fulfilment of the requirements for the degree of Ph. D. in Operations Research at Massey University (Doctoral,* 1996.

[5] M. E. Dyer, L. R. Foulds and A. M. Frieze, "Analysis of heuristics for finding a maximum weight planar subgraph.," *European Journal of Operational Research,* vol. 20, no. 1, pp. 102-114, 1985.

[6] E. Pesch, "Efficient facility layout planning in a maximally planar graph model," *International Journal of Production Research,* vol. 37, no. 2, pp. 263-283, 1999.

[7] B. Montreuil, " A modelling framework for integrating layout design and flow network design," in *Material handling'90* , Springer, Berlin, Heidelberg, 1991, pp. 95-115.

[8] A. A. Pessoa, P. M. Hahn, M. Guignard and Y. R. Zhu, "An improved algorithm for the generalized quadratic assignment problem," *Electrical and Systems Engineering,* 2008.

[9] Wikipedia, "Branch and bound," [Online]. Available: https://en.wikipedia.org/w/index.php?title=Branch_and_bound&oldid=863106124.

[10] P. Levin, "Use of graphs to decide the optimum layout of buildings," *The Architects' Journal,* vol. 7, pp. 809-815, 1964.

[11] M. Krejcirik, "Computer-aided plant layout," *Computer-Aided Design,* vol. 2, no. 1, pp. 7-19., 1969.

[12] L. R. Foulds, "Techniques for facilities layout: deciding which pairs of activities should be adjacent," *Management Science, 29(12),* , vol. 29, no. 12, pp. 1414-1426, 1983.

[13] M. Smith and R. Macleod, " A Relaxed Assignment Algorithm For The Quadratic Assignement Problem," *INFOR: Information Systems and Operational Research, 26(3),*, vol. 26, no. 3, pp. 170-190, 1988.

[14] A. Kusiak and S. S. Heragu, "The facility layout problem," *European Journal of operational research,* vol. 29, no. 3, pp. 229-251, 1987.

[15] R. D. Meller and K. Y. Gau, "The facility layout problem: recent and emerging trends and perspectives," *Journal of manufacturing systems,* vol. 15, no. 5, pp. 351-366, 1996.

[16] G. C. Armour and E. S. Buffa, "A heuristic algorithm and simulation approach to relative location of facilities," *Management Science,* vol. 9, no. 2, pp. 294-309, 1963.

[17] M. S. Bazaraa, "Computerized layout design: a branch and bound approach," *AIIE transactions,* vol. 7, no. 4, pp. 432-438, 1975.

[18] H. Sherali, B. Fraticelli and R. Meller, " Enhanced Model Formulations for Optimal Facility Layout," *Operations Research,* vol. 51, no. 4, p. 629–644, 2003.

[19] A. Konak, S. Kulturel-Konak, B. A. Norman and A. E. Smith, " A new mixed integer programming formulation for facility layout design using flexible bays," *Operations Research Letters,* vol. 344, no. 6, pp. 660-672, 2006.

[20] Wikipedia, "Sensitivity analysis," [Online]. Available: https://en.wikipedia.org/w/index.php?title=Sensitivity_analysis&oldid=846760482.

[21] R. S. Liggett, "The quadratic assignment problem: An experimental evaluation of solution strategies," *Management Science,* vol. 27, no. 4, pp. 442-458, 1981.

[22] R. Arapoglu, B. Norman and A. Smith, "Locating Input and Output Points in Facilities Design – A Comparison of Constructive, Evolutionary, and Exact Methods," *IIE Transactions on Evolutionary Computation,* vol. 5, no. 3, pp. 192-203, 2001.

[23] B. Montreuil and H. D. Ratliff, "Utilizing cut trees as design skeletons for facility layout.," *IIE transactions,* vol. 21, no. 2, pp. 136-143, 1989.

[24] J. Kim and C. Klein, "Location of Departmental Pickup and Delivery Points for an AGV System," *International Journal of Production Research,* vol. 34, no. 2, pp. 407-420. , 1996.

[25] B. Benson and B. Foote, " DoorFAST: A Constructive Procedure to Optimally Layout a Facility Including Aisles and Door Locations Based on an Aisle Flow Distance Metric," *International Journal of Production Research,* vol. 35, no. 7, pp. 1825-1842, 1997.

[26] B. Peters and T. Yang, "Integrated Facility Layout and Material Handling System Design in Semiconductor Fabrication Facilities," *IEEE Transactions on Semiconductor Manufacturing,* vol. 10, no. 3, pp. 360-369., 1997.

[27] Wikipedia, "Machine learning," [Online]. Available: https://en.wikipedia.org/wiki/Machine_learning. [Accessed 30 June 2019].

[28] E. W. Weisstein, " Triangulated Graph," MathWorld--A Wolfram Web Resource, [Online]. Available: http://mathworld.wolfram.com/TriangulatedGraph.html.

[29] Pelosi, J. Smith and R.P, "Conversational optimization and layout planning Environment and Planning,,," 1984, pp. B11:63-86.

[30] R. BURKARD, E. ÇELA, S. KARISCH and F. RENDL, "QAPLIB Problem Instances and Solutions," [Online]. Available: http://anjos.mgi.polymtl.ca/qaplib/inst.html#BO.

[31] Wikipedia, "Duality (mathematics)," [Online]. Available: https://en.wikipedia.org/w/index.php?title=Duality_(mathematics)&oldid=851661023.

[32] Wikipedia, "Glossary of computer graphics," [Online]. Available: https://en.wikipedia.org/w/index.php?title=Glossary_of_computer_graphics&oldid=852437283.

[33] B. Delaunay, " Sur la sphere vide," *Izv. Akad. Nauk SSSR, Otdelenie Matematicheskii i Estestvennyka Nauk,* vol. 7, no. 793-800, pp. 1-2, 1934.

[34] A. Hameurlain, J. Küng and R. (. Wagner, Transactions on Large-Scale Data-and Knowledge-Centered Systems I (Vol. 5740), Springer, 2009.

[35] D. Warme, P. Winter and M. Zachariasen, "GeoSteiner-Software for Computing Steiner Trees," [Online]. Available: http://www.geosteiner.com/.

[36] F. Dehne, J. R. Sack and C. D. (. Tóth, Algorithms and Data Structures: 11th International Symposium,, Banff, Canada: Springer Science & Business Media., August 21-23, 2009.

[37] P. Berman, M. Karpinski and A. Zelikovsky, "1.25-approximation algorithm for Steiner tree problem with distances 1 and 2.In Workshop on Algorithms and Data Structures," Springer,, Berlin, Heidelberg, 2009.

[38] M. W. Bern and R. L. Graham, " The shortest-network problem," *Scientific American,* vol. 260, no. 1, pp. 84-89, 1989.

[39] D. M. Warme and J. S. Salowe, Spanning trees in hypergraphs with applications to Steiner trees, University of Virginia, 1998.

[40] Y. J. Song and J. H. Woo, "New shipyard layout design for the preliminary phase & case study for the green field project," *International Journal of Naval Architecture and Ocean Engineering,* vol. 5, no. 1, pp. 132-146, 2013.

[41] P. Lynch, "How Voronoi diagrams help us understand our world," The Irish Times , Dublin, 2017.

[42] Asef-Vaziri and G. Laporte, "Loop based facility planning and material handling. European Journal of Operational Research," *European Journal of Operational Research,* vol. 164, no. 1, pp. 1-11, 2005.

[43] T. Gowers, "The Steiner Minimal Tree," ThatsMaths, Wordpress, 2015.

[44] L. R. Foulds and D. Robinson, "Graph Theoretic Heuristics for the Plant Layout Problem," *Internat. J. Productions Res.,* vol. 16, pp. 27-37, 1978.

[45] Wikipedia, "Steiner tree problem," [Online]. Available: https://en.wikipedia.org/w/index.php?title=Steiner_tree_problem&oldid=847641135.

[46] Wikipedia, "Voronoi diagram," [Online]. Available: https://en.wikipedia.org/w/index.php?title=Voronoi_diagram&oldid=850751749.

[47] E. W. Weisstein, "Triangulated Graph," [Online]. Available: http://mathworld.wolfram.com/TriangulatedGraph.html. [Accessed 5 April 2018].

[48] Wikipedia, "Planar graph," [Online]. Available: https://en.wikipedia.org/wiki/Planar_graph#Maximal_planar_graphs. [Accessed 5 April 2018].

[49] C. R. Shebanie, "AN INTEGRATED, EVOLUTIONARY APPROACH TO FACILITY LAYOUT AND DETAILED DESIGN. University of Pittsburgh," 2004. [Online]. Available: https://pdfs.semanticscholar.org/d462/1dfa19a31e42ff59e1a2e4d3fe717b3b7f3d.pdf.. [Accessed 10 April 2018].

[50] R. Bowen and S. Fisk, "Generation of Triangulations of the Sphere," *Mathematics of Computation,* vol. 21, no. 98, p. 250, 1967.

[51] E. Pesch, F. Glover, T. Bartsch, F. Salewski and I. Osman., "Efficient Facility Layout Planning in a Maximally Planar Graph.," 1999.