



# MATHEMATICS EXTENDED

## ESSAY

Optimal Poster Placement for Maximum Outreach  
using Graph Theory

Word Count: 3968

# Table of Contents

1. Introduction.....	1
1.1 Rationale.....	1
1.2 Research Question.....	3
2. Background and Concepts.....	4
2.1 Mathematical Optimisation .....	4
2.2 Graph Theory Concepts .....	5
2.3 Area of Study .....	7
3. Methodology .....	10
3.1 Mathematical Modelling.....	11
Vertices .....	11
Edges.....	17
3.2 Edge Weights .....	24
3.3 Multigraph Conversion .....	25
4. Analysis and Discussion .....	29
4.1 Determining the Paths taken by Students .....	29
Data Collection.....	29
Tabulating Data .....	31
Dijkstra's Algorithm .....	35
4.2 Determining Poster Necessity at Various Amenities .....	46
Allocating Weights to Vertices .....	47

A Greedy Algorithm.....	50
Application of Greedy Algorithm .....	51
5. Evaluation and Conclusion .....	58
5.1 Evaluating the Solution .....	58
Objective Functions .....	58
Objective Function 1 Calculation Example .....	60
Calculating Objective Function Values .....	61
5.2 Real Life Considerations.....	62
Calculating Objective Function Values .....	64
5.3 Comparisons .....	65
5.4 Assumptions and Limitations.....	67
5.6 Possible Improvements .....	68
5.5 Conclusion .....	69
10. Bibliography.....	71
Appendix A: Survey Questions .....	1
.....	1
Appendix B: Sample Survey Response.....	1
.....	1
Appendix C: Edge Weight Measurements.....	2
Appendix D: Dijkstra Python Code.....	9

## Table of Figures

Figure 1: Satellite view of St. Joseph Institution, Singapore .....	2
Figure 2: Methodology of Research Paper .....	<b>Error! Bookmark not defined.</b>
Figure 3: Sample Graph G.....	<b>Error! Bookmark not defined.</b>
Figure 4: Layout of St. Joseph Institution 1 <sup>st</sup> Floor .....	8
Figure 5: Layout of St. Joseph Institution 2 <sup>nd</sup> Floor .....	8
Figure 6: Layout of St. Joseph Institution 3 <sup>rd</sup> Floor .....	9
Figure 7: Layout of St. Joseph Institution 5 <sup>th</sup> and 6 <sup>th</sup> Floor .....	9
Figure 8: Layout of St. Joseph Institution 1st Floor with Vertices .....	14
Figure 9: Layout of St. Joseph Institution 2nd floor with Vertices.....	15
Figure 10: Layout of St. Joseph Institution 3rd floor with Vertices .....	15
Figure 11: Layouts of St. Joseph Institution 5th and 6th floor with Vertices.....	16
Figure 12: Layouts of St. Joseph Institution 1st and 2nd floor with Vertices and edge Q-H ....	17
Figure 13: Layout of St. Joseph Institution 1st floor with Vertices and Edges.....	18
Figure 14: Mathematical Model of St. Joseph Institution 1 <sup>st</sup> Floor .....	18
Figure 15: Layout of St. Joseph Institution 2nd floor with Vertices and Edges .....	19
Figure 16: Mathematical Model of St. Joseph Institution 2 <sup>nd</sup> Floor .....	19
Figure 17: Layout of St. Joseph Institution 3rd floor with Vertices and Edges .....	20
Figure 18: Mathematical Model of St. Joseph Institution 3 <sup>rd</sup> Floor .....	20
Figure 19: Layouts of St. Joseph Institution 5th and 6th floors with Vertices and Edges.....	21
Figure 20: Mathematical Model of St. Joseph Institution 5 <sup>th</sup> and 6 <sup>th</sup> Floor.....	21
Figure 21: 3D view of St. Joseph Institution Layouts with Vertices and Edges.....	22
Figure 22: 3D Mathematical Model of St. Joseph Institution .....	23

Figure 23: Measurement of School using iPhone measuring application .....	24
Figure 24: Multiple Edges on Graph representing St. Joseph Institution 1 <sup>st</sup> Floor .....	26
Figure 25: Multiple Edges on Graph representing St. Joseph Institution 2 <sup>nd</sup> Floor .....	26
Figure 26: Multiple Edges on Graph representing St. Joseph Institution 3 <sup>rd</sup> Floor.....	27
Figure 27: Simple Graph as a Mathematical Model Representing St. Joseph Institution.....	28
Figure 28: Areas Covered during Survey Dissemination.....	30
Figure 29: Survey Sample Response.....	31
Figure 30: Graph representing St. Joseph Institution 1st floor with numerical weights .....	38
Figure 31: Graph representing 1st, 2nd, 3rd, 5th and 6th floors of St. Joseph Institution with Numerical Vertex Weights .....	49
Figure 32: Real Life Poster Placement.....	63
Figure 33: Mathematical Model of St. Joseph Institution Compound with indicated Vertices for Suggested Poster Placement .....	69

## Table of Tables

### Table 1: Graph Theory

Table 1: Objectives of Research .....	4
Table 2: Graph Theory Terminology.....	5
Table 3: Table of Vertices representing Amenities.....	12
Table 4: Table of Vertices Representing Stairwell Entrances .....	13
Table 5: Legend of Vertex Representation .....	14
Table 6: Table of Areas covered by Persons A to E.....	30
Table 7: Table of Vertices representing Amenities.....	31

Table 8: Table of Sample Response Tabulation .....	32
Table 9: Table of Survey Responses .....	32
Table 10: Sample of Survey Response Tabulation .....	38
Table 11: Illustration of Dijkstra's Algorithmn .....	39
Table 12: Table of Sample Walk Taken Tabulation .....	42
Table 13: Results of Dijkstra's Algorithm .....	43
Table 14: Rationale for Poster Placement at Vertices .....	46
Table 15: Sample of Data Collected from Dijkstra's Algorithm.....	47
Table 16: Sample of Numerical weights assigned to Vertices .....	48
Table 17: Table of Numerical Weights assigned to Vertices .....	48
Table 18: Illustration of Greedy Algorithm Application .....	52
Table 19: Table of Suggested Amenities for Poster Placement.....	57
Table 20: Table of Objective Functions .....	59
Table 21: Binary Variable Definitions .....	59
Table 22: Sample Results.....	60
Table 23: Objective Function Calculations for Suggested Poster Arrangement.....	61
Table 24: Real-Life Amenities for Poster Placement and Vertices .....	62
Table 25: Objective Function Calculations for Real-Life Poster Arrangement.....	64
Table 26: Real Life and Suggested Vertices for Poster Placement .....	65
Table 27: Objective Function 1 Comparison.....	65
Table 28: Objective Function 2 Comparison.....	66
Terminology.....	5
Table 2: Table of Vertices representing Amenities.....	12
Table 3: Table of Vertices Representing Stairways.....	13

Table 4: Table of Areas covered by Persons A to E.....	30
Table 5: Table of Vertices representing Amenities.....	31
Table 6: Table of Sample Response Tabulation .....	32
Table 7: Table of Survey Responses .....	32
Table 8: Sample of Survey Response Tabulation .....	38
Table 9: Illustration of Dijkstra's Algorithmn .....	39
Table 10: Table of Sample Walk Taken Tabulation.....	42
Table 11: Results of Dijkstra's Algorithm .....	43
Table 12: Rationale for Poster Placement at Vertices .....	46
Table 13: Sample of Data Collected from Dijkstra's Algorithm.....	47
Table 14: Sample of Numerical weights assigned to Vertices .....	48
Table 15: Table of Numerical Weights assigned to Vertices .....	48
Table 16: Illustration of Greedy Algorithm Application .....	52
Table 17: Table of Suggested Amenities for Poster Placement.....	57
Table 18: Table of Objective Functions .....	59
Table 19: Binary Variable Definitions .....	59
Table 20: Sample Results.....	60
Table 21: Objective Function Calculations for Suggested Poster Arrangement.....	61
Table 22: Real-Life Amenities for Poster Placement and Vertices .....	62
Table 23: Objective Function Calculations for Real-Life Poster Arrangement.....	64

# 1. Introduction

## 1.1 Rationale

When campaigning for office, candidates need to reach their voters through a variety of mediums such as social media or campaign speeches. A more commonly overlooked method used during these electoral processes is the deployment of the humble poster. As a student council nominee, the publicity of my candidature is of great importance in gaining the upper hand against my competitors.

After some discreet scouting, I realised that my rivals had placed posters around the school, some indiscriminately and others well-placed in comparison to my own. This motivated me to consider – was there a more tactical approach to poster placement to gain an advantage over my rivals?

As a student of economics, I understood that to minimise the financial cost of printing posters I had to strategize on ensuring that I had mileage and return on investment for the cost that I am spending on this, especially considering the impact of these large individual poster prices on my student allowance. The optimal solution in my case would hence be one where posters were placed in amenities where a maximum number of students pass by while maintaining minimal use of resources.





*Figure 1: Satellite view of St. Joseph Institution, Singapore*

The area of study would be the school compound of St. Joseph Institution (SJI), Singapore, where I study, demarcated by the yellow border in figure 1.

In approaching the problem, I would have to consider a transport network of student travel between various areas of the school compound. Due to its history in modelling real-life transportation networks à la the The Königsberg Bridge problem (Carlson, 2010), I therefore decided to use graph theory to approach this problem.

## 1.2 Research Question

To better guide me in finding the optimal poster arrangement, I formulated my research question as such:

***“What is the optimal placement of council campaign posters in the St. Joseph Institution school compound to maximise outreach and minimise poster usage?”***

To unpack my research question, the problem posed can be said to require a solution that fulfils two specific requirements,

1. To maximise student outreach, posters must be arranged **to intercept the maximum quantity of students.**
2. To minimise financial costs, **the quantity of posters used must be minimised.**

## 2. Background and Concepts

### 2.1 Mathematical Optimisation

Mathematical optimisation commonly refers to the problem of **maximising or minimising** certain variables by finding the '**best**' values across a set of these variables. (DeepAI, 2019)

Recalling the criteria for optimal poster placement, two variables to be optimised are identified below.

*Table 1: Objectives of Research*

Requirement	Variable	Objective
1. Posters must be arranged to intercept the maximum quantity of students.	Quantity of students intercepted	Maximise
2. the quantity of posters used must be minimised	Quantity of posters placed	Minimise

As such, the problem posed in this essay can be characterised as an optimisation problem.

## 2.2 Graph Theory Concepts

Graph Theory is the use of mathematics to study the relationships between objects via ‘graphs’, mathematical structures used to model a connective network of such relationships.

(Galvin, 2009)

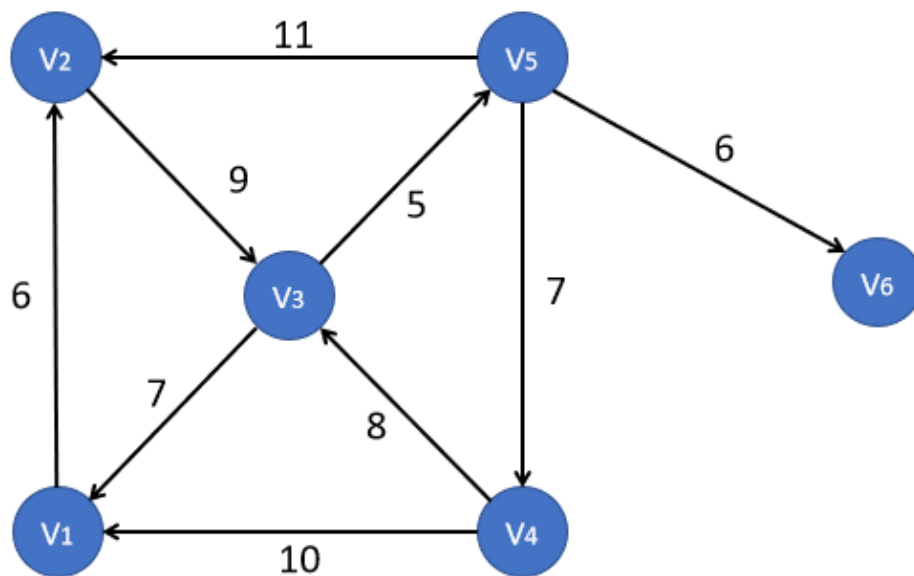


Figure 2: Sample Graph G

Various graph theory terminology are defined in table 2, contextualised to describe Graph G in figure 2.

Table 2: Graph Theory Terminology

Terminology	Explanation	Graph G Contextualisation
Graph, Vertex, Edge	<b>Graphs</b> are a finite set of dots and links, referred to as <b>vertices</b> and <b>edges</b> respectively where each vertex represents an object and each edge connects adjacent vertices to each other.	Graph G is a graph where $G = (V, E)$ $V = \{V_1, V_2, V_3, V_4, V_5, V_6\}$ and $E = \{V_1V_2, V_1V_3, V_1V_4, V_2V_3,$

	For a graph $G$ , graphs are formally denoted by a pair of sets $G = (V, E)$ where $V$ is a set of vertices and $E$ is a set of edges such that $V = \{V_1, V_2, V_3, \dots\}$ and $E = \{V_1V_2, V_1V_3, V_2V_3 \dots\}$ . (Galvin, 2009)	$V_3V_4, V_3V_5, V_4V_5, V_5V_6\}$
Walk	A walk is a sequence of vertices and edges of a graph, if you travel along a graph from vertex to vertex, you get a walk. (him0000, 2020)  For this essay, the convention $V_1, V_2, V_3 \dots$ will denote a walk from vertex $V_1$ to $V_2$ to $V_3$ .	In graph $G$ , a walk from $V_4$ to $V_3$ to $V_5$ to $V_6$ is denoted by $V_4, V_3, V_5, V_6$
Undirected and Directed Graphs	A <b>directed</b> graph is one where certain edges only allow travel in one direction, as indicated by an arrowhead at the end of an edge. Conversely, an <b>undirected</b> graph refers to graphs whose edges lack direction. (Guichard, n.d.)	Graph $G$ is directed, as indicated by the arrow heads on each edge.
Simple Graph, Multigraph	A <b>multigraph</b> is a graph with multiple edges between pairs of vertices while a <b>simple graph</b> is a graph where vertices are only connected by lone edges. (Weisstein, 2021)	Graph $G$ is a simple graph as each pair of vertices are connected by single edges.
Incomplete, Complete Graphs	A <b>complete graph</b> is one where every vertex is connected to every other unique vertex in the graph via an edge, otherwise, the graph is <b>incomplete</b> . (Weisstein, <i>Graph</i> 2021)	Graph $G$ is an incomplete graph as not all vertices are connected to every other unique vertex.

Size	The number of edges of a graph is referred to as its <b>size</b> , denoted by $ E $ . (Galvin, 2009)	The size of graph G, $ E  = 9$
Weight	A numerical value assigned to an edge in a graph is its <b>weight</b> . (Weisstein, <i>Graph</i> 2021)	For example, the weight of edge $V_4V_5$ is 7.
Order	The number of vertices in a graph is referred to as its <b>order</b> , denoted by $ V $ . (Galvin, 2009)	The order of graph G, $ V  = 6$
Degree	The <b>degree</b> of a vertex refers to the number of edges attached to it (Weisstein, <i>Vertex degree</i> 2021)	vertex $V_3$ has a degree of 4

## 2.3 Area of Study

The area of study to be modelled is the 1<sup>st</sup> to 6<sup>th</sup> floors of the SJI school building, excluding the 4<sup>th</sup> floor due to its inaccessibility to students. Of the school compound, only the areas outlined in the figures below will be considered due to the high student population density in these areas during student break periods.

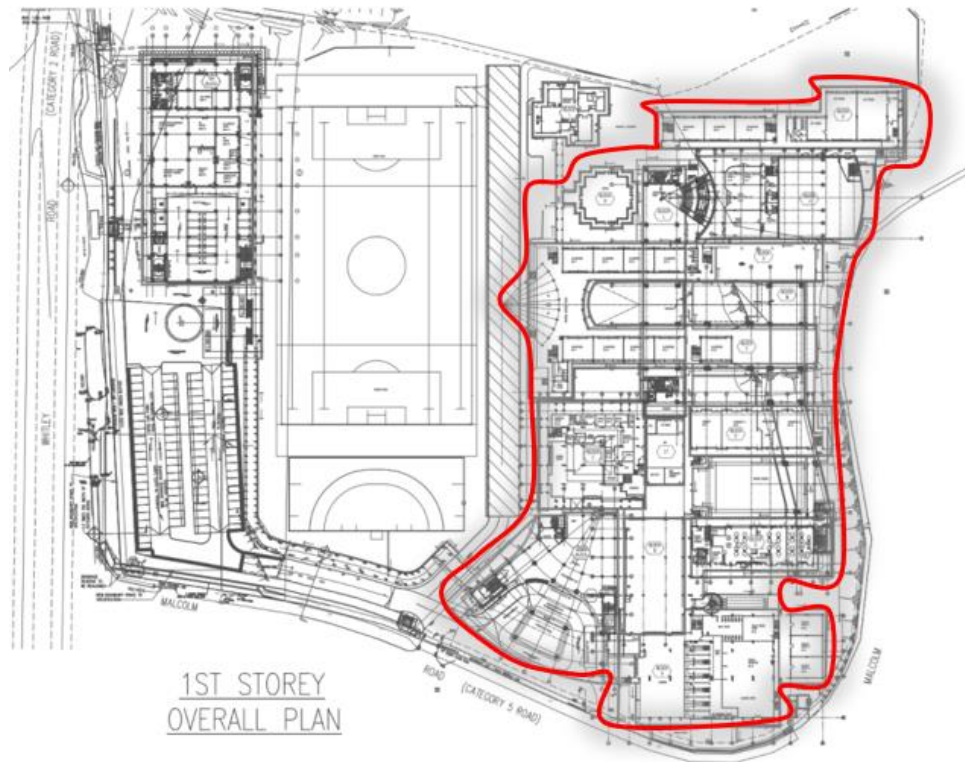


Figure 3: Layout of St. Joseph Institution 1<sup>st</sup> Floor (SJI, 1st Storey Overall Plan 2018)

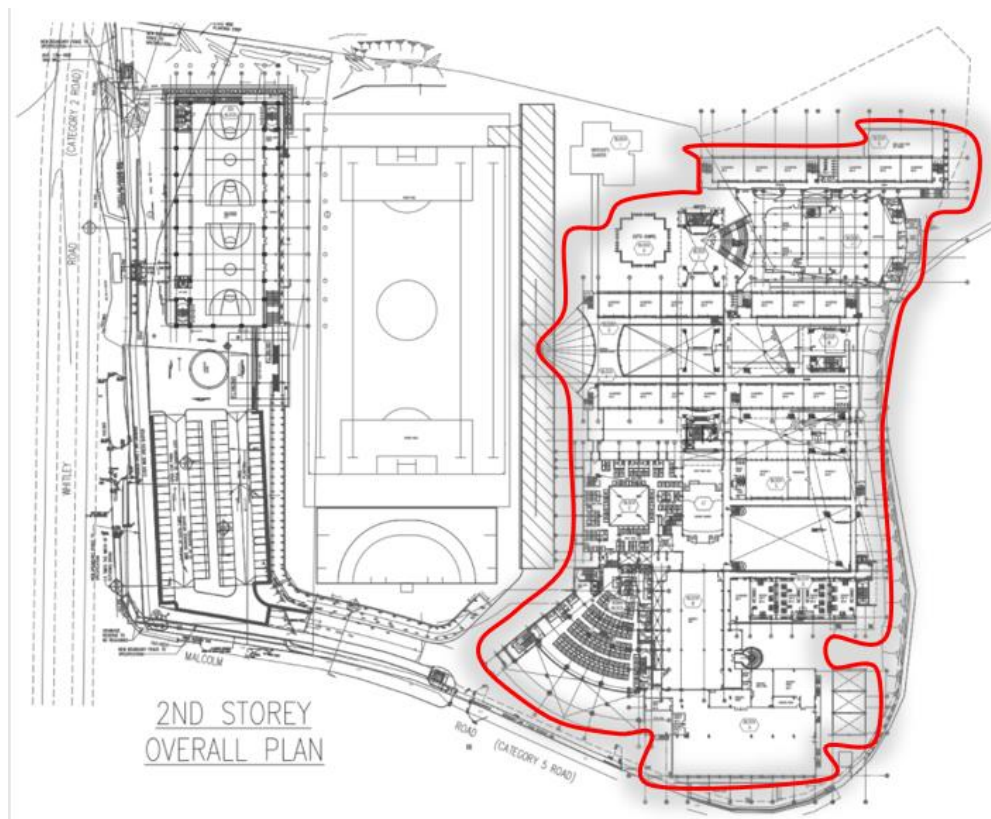


Figure 4: Layout of St. Joseph Institution 2<sup>nd</sup> Floor (SJI, 2nd Storey Overall Plan 2018)



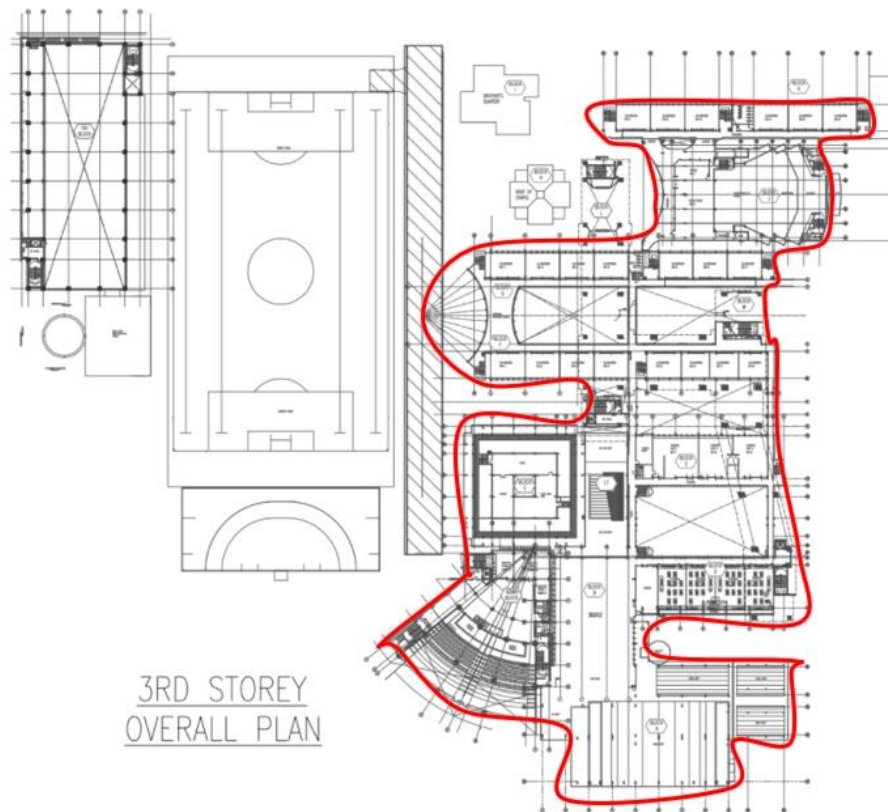


Figure 5: Layout of St. Joseph Institution 3<sup>rd</sup> Floor (SJI, 3rd Storey Overall Plan 2018)



Figure 6: Layout of St. Joseph Institution 5<sup>th</sup> and 6<sup>th</sup> Floor (SJI, 5th and 6th Storey Overall Plan 2018)



### 3. Methodology

To tackle my research question, I will undertake the methodology outlined in figure 7.

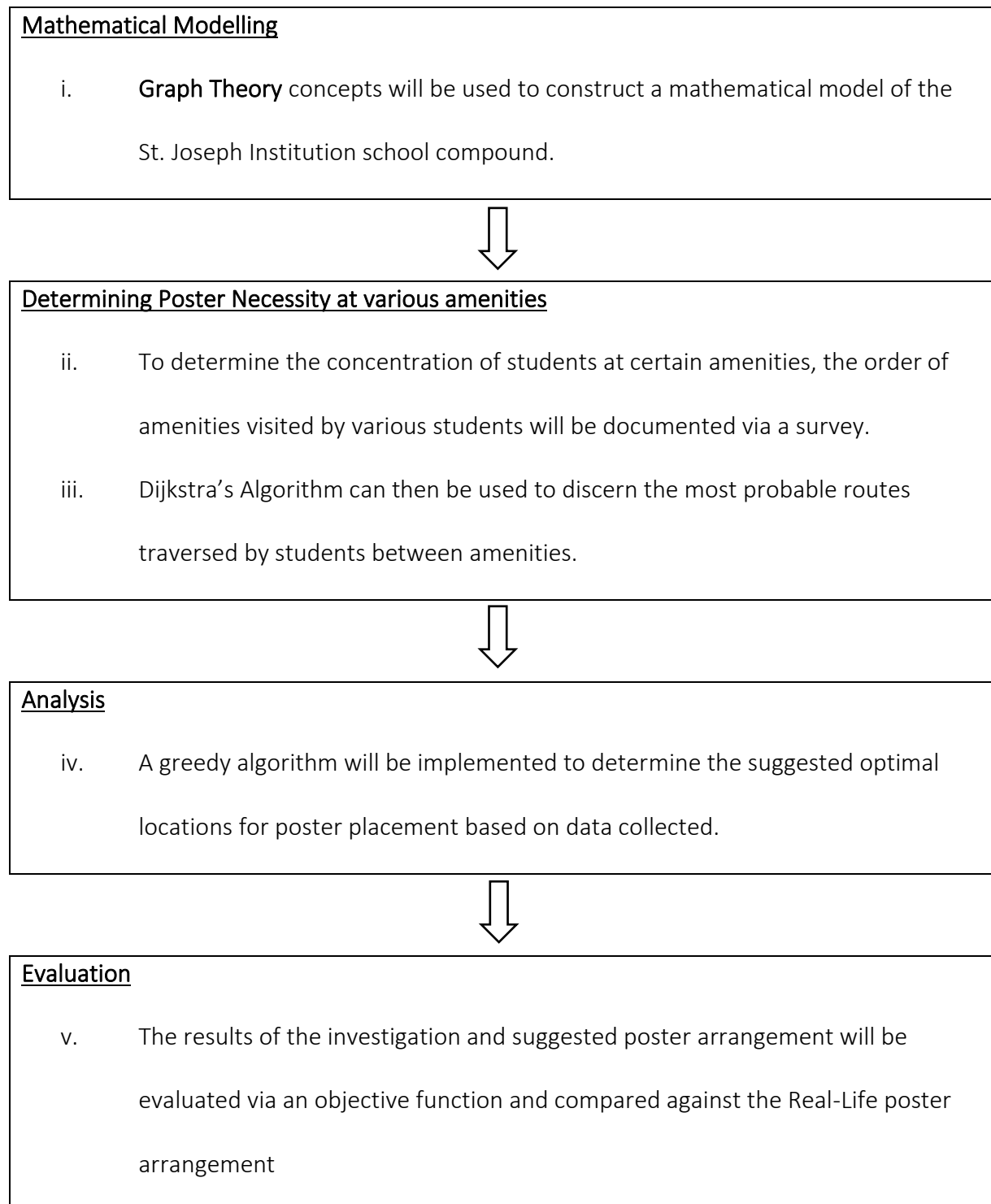


Figure 7: Methodology

## 3.1 Mathematical Modelling

### Vertices

Each school amenity frequented by students will be represented by a vertex.

**Here, it is assumed that students exclusively travel from one amenity to another. Hence, each of these frequently visited amenities were assigned a vertex.**

To determine the amenities to be studied, a survey among the students of St. Joseph's Institution was conducted via google form. Since voters in the electoral process were all students currently studying at the IB level, these particular students formed the target group of respondents, obtaining a total of 30 responses each from both the year 5 and year 6 levels.

A sample of the survey questions and sample responses can be found in Appendix A and B respectively.

From the survey, 27 amenities in total were located. Each amenity was then assigned a corresponding vertex labelled by a capital or lowercase letter. Classrooms were not taken into account due to the school-wide restriction on placing posters in classrooms. The table of letters and corresponding amenities can be found below in table 3.

Table 3: Table of Vertices representing Amenities

Vertex	Amenity	Vertex	Amenity
A	Level 1 Lab 1	N	Level 2 Lab 3
B	Level 1 Lab 2	O	Level 2 Lab 4
C	Level 1 Lab 3	P	Library
D	Level 1 Lab 4	Q	Annex
E	Canteen	R	Level 2 Benches
F	Foyer	S	Level 3 Lab 1
G	Prefect Benches	T	Level 3 Lab 2
H	Oval	U	Level 3 Lab 3
I	James Miller Room	V	Level 3 Lab 4
J	Secondary School Benches	W	Block B Level 5 Open Area
K	Art Room	X	Block A Level 5 Open Area
L	Level 2 Lab 1	Y	Block B Level 6 Open Area
M	Level 2 Lab 2	Z	Block A Level 6 Open Area
z	Annex 2		



Since the mathematical model will be representative of multiple floors of the school compound, each stairwell serves as the bridge between different floors. To traverse multiple floors, each student must have access to a stairwell, hence, stairwell entrances would also be represented by vertices.

*Table 4: Table of Vertices Representing Stairwell Entrances*

Vertex	Stairwell Entrance	Vertex	Stairwell
1A	Stairway 1A	2J	Stairway 2J
1B	Stairway 1B: Spiral Stairwell 2B	2K	Stairway 2K
1C	Stairway 1C	3A	Stairway 3A
1D	Stairway 1D	3D	Stairway 3D
1E	Stairway 1E	3C	Stairway 3C
1F	Stairway 1F	3E	Stairway 3E
1G	Stairway 1G	3F	Stairway 3F
1H	Stairway 1H	3G	Stairway 3G
1J	Stairway 1J	3L	Stairway 3L
1M	Stairway 1M	3H	Stairway 3H
1K	Stairway 1K	3J	Stairway 3J
2A	Stairway 2A	3K	Stairway 3K
2B	Stairway 2B: Spiral Stairwell 2B	5M	Stairway 5M
2C	Stairway 2C	5E	Stairway 5E
2D	Stairway 2D	5L	Stairway 5L
2E	Stairway 2E	5A	Stairway 5A
2F	Stairway 2F	6M	Stairway 6M
2G	Stairway 2G	6E	Stairway 6E
2L	Stairway 2L	6L	Stairway 6L
2H	Stairway 2H	6A	Stairway 6A

To model the compound, layouts provided by the school administration were utilised as reference materials for the sketching of the graph. The vertices of the graph have been marked out on each floor layout below.

Table 5: Legend of Vertex Representation

Legend:	
	denotes a vertex representing a stairway entrance
	denotes a vertex representing an amenity

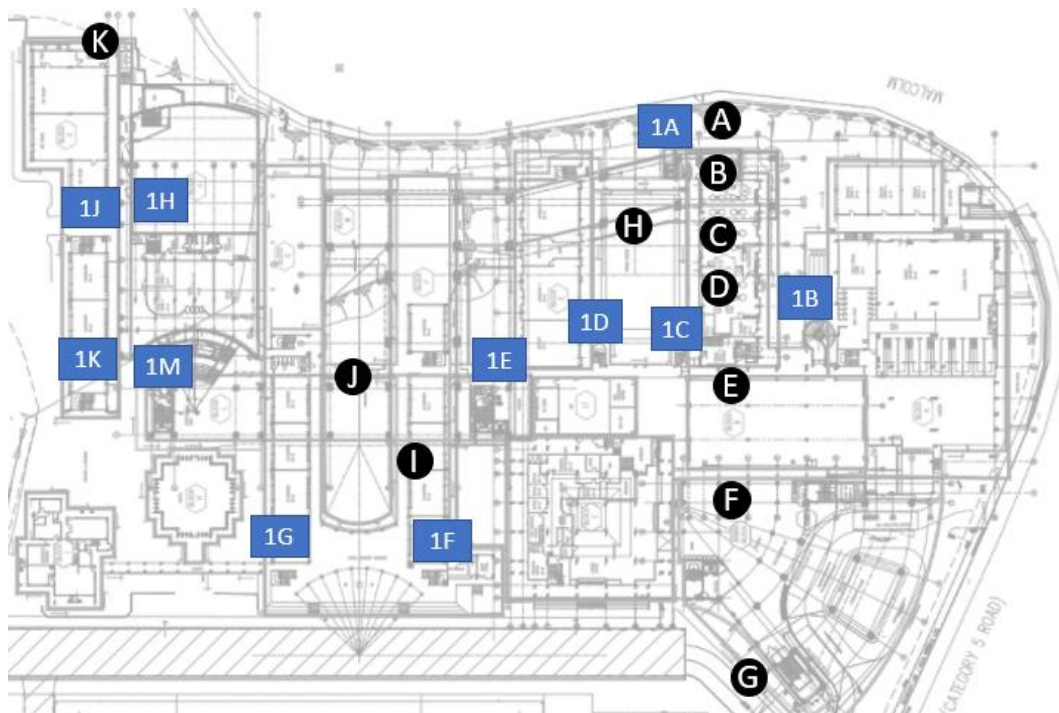


Figure 8: Layout of St. Joseph Institution 1st Floor with Vertices (SJI, 1st Storey Overall Plan 2018)

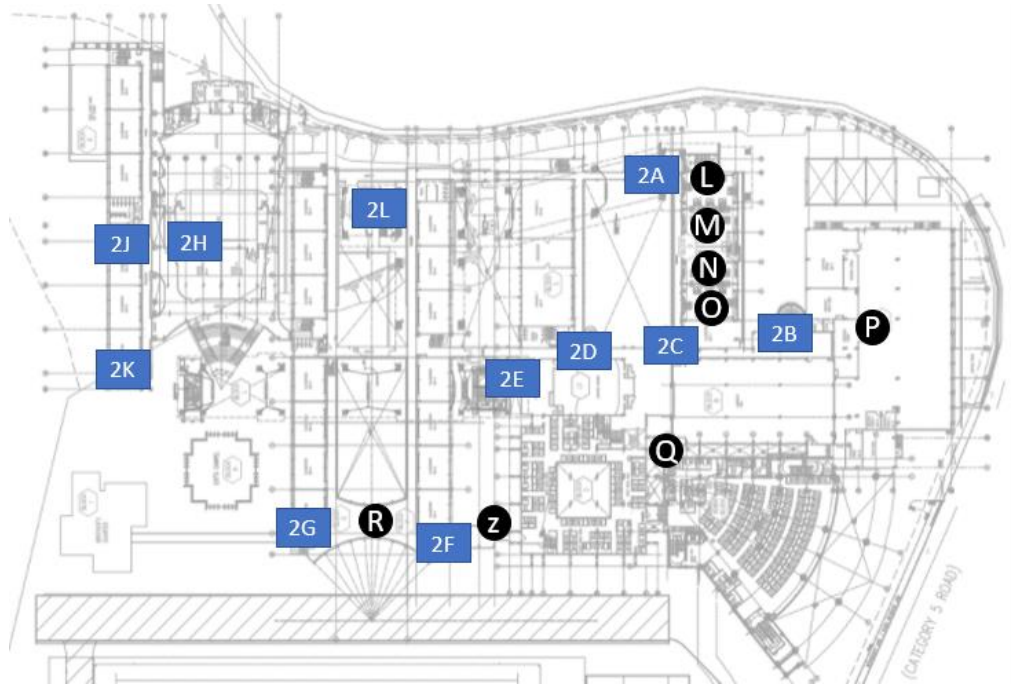


Figure 9: Layout of St. Joseph Institution 2nd floor with Vertices (SJI, 2nd Storey Overall Plan 2018)

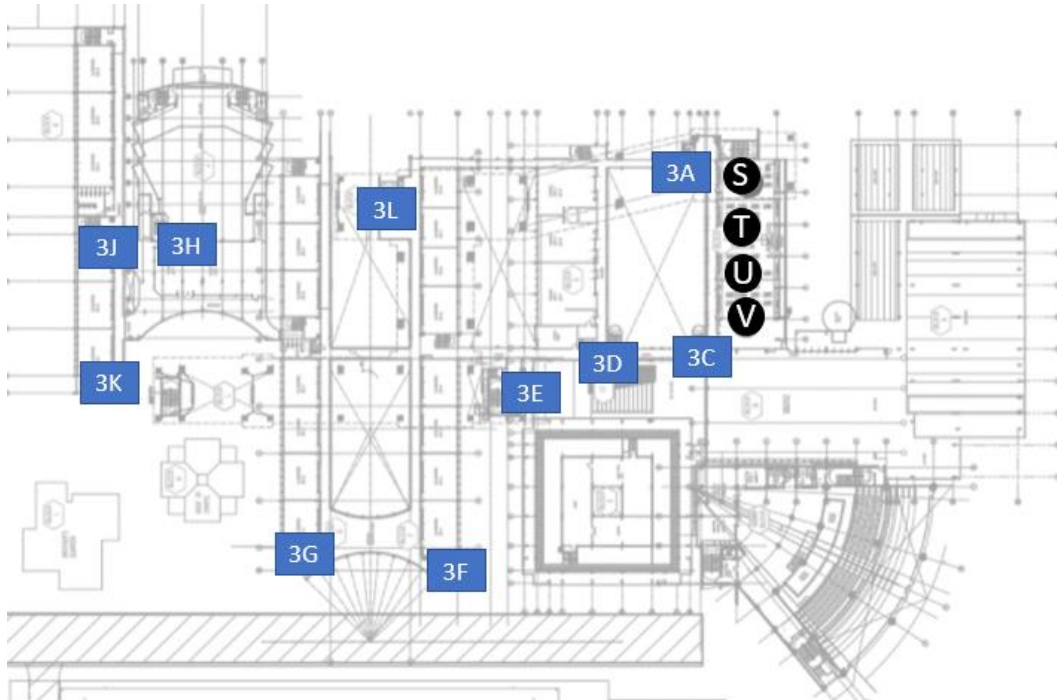


Figure 10: Layout of St. Joseph Institution 3rd floor with Vertices (SJI, 3rd Storey Overall Plan 2018)

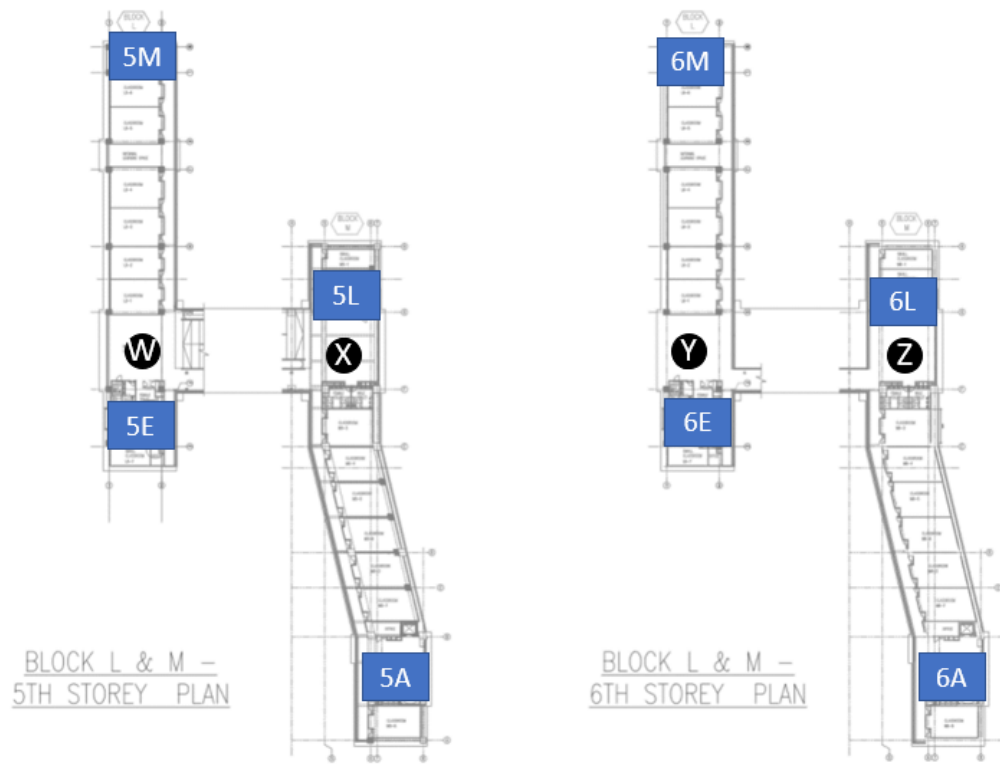


Figure 11: Layouts of St. Joseph Institution 5th and 6th floor with Vertices (SJI, 5th and 6th Storey Overall Plan 2018)

## Edges

Each edge will represent the walkway paths taken by students from one location to another.

Here an assumption is made that Students will keep to these pathways and abstain from walking across grass patches, decorative elements etc.

These edges can be said to be **undirected** as students can traverse them in both forward and backward direction, providing for an **undirected graph**.

Additionally, the graph can be characterised as an **incomplete** graph as not all 2 vertices are connected to one another by an edge. For example, a straight-lined edge Q-H (represented by the red line) between vertices Q and H cannot exist as the two vertices are separated on different floors with no designated pathway directly connecting the two.

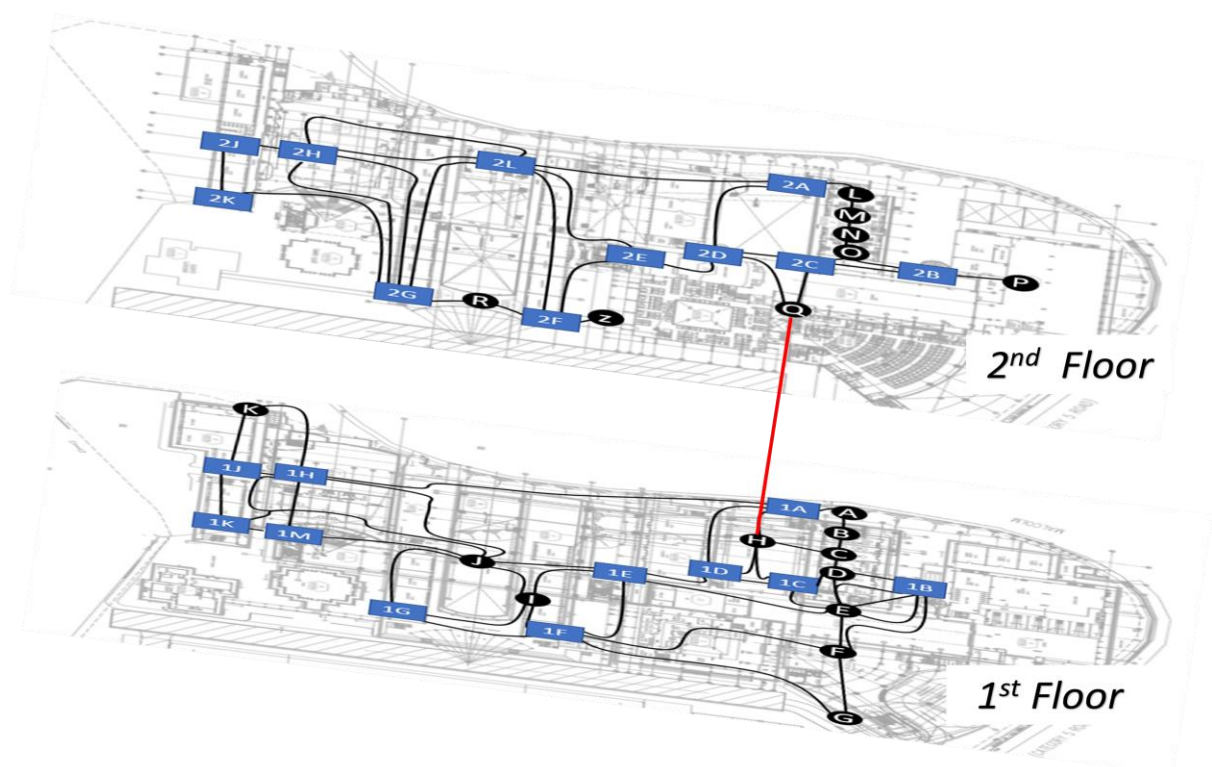


Figure 12: Layouts of St. Joseph Institution 1st and 2nd floor with Vertices and edge Q-H



Upon drawing in edges for each floor, the figures below are obtained,

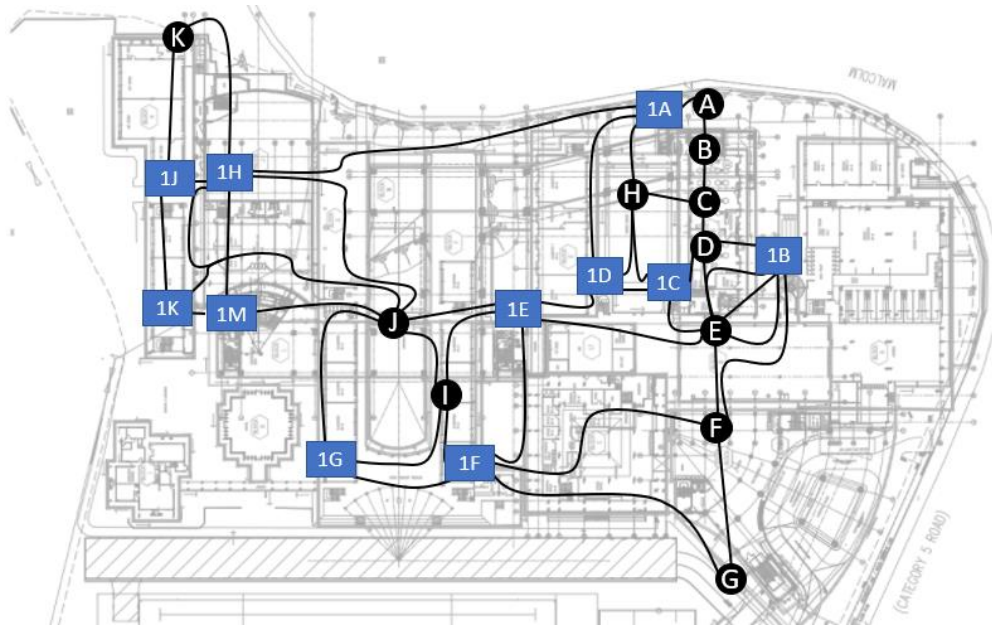


Figure 13: Layout of St. Joseph Institution 1st floor with Vertices and Edges

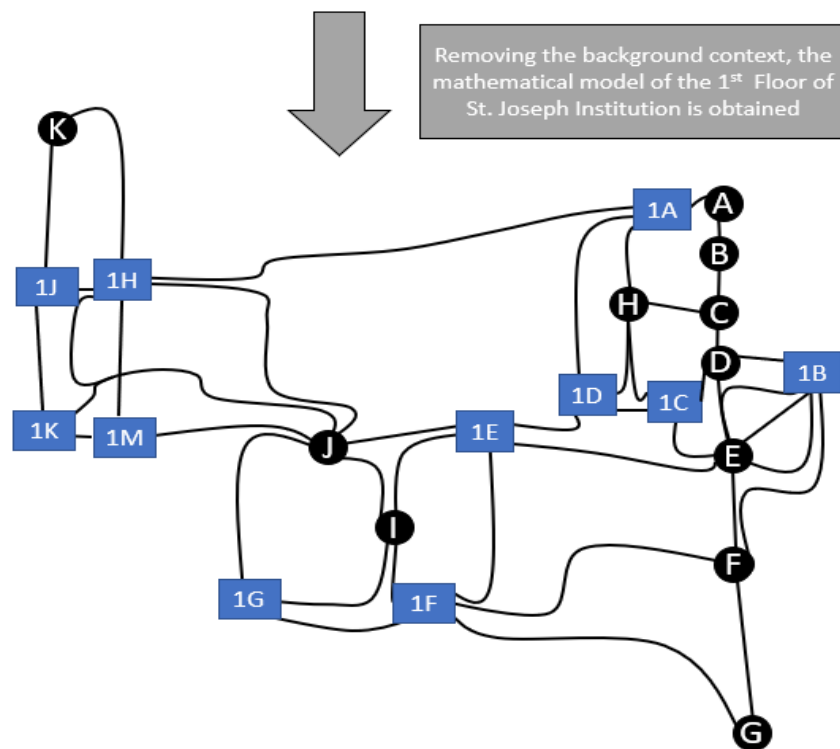


Figure 14: Mathematical Model of St. Joseph Institution 1<sup>st</sup> Floor

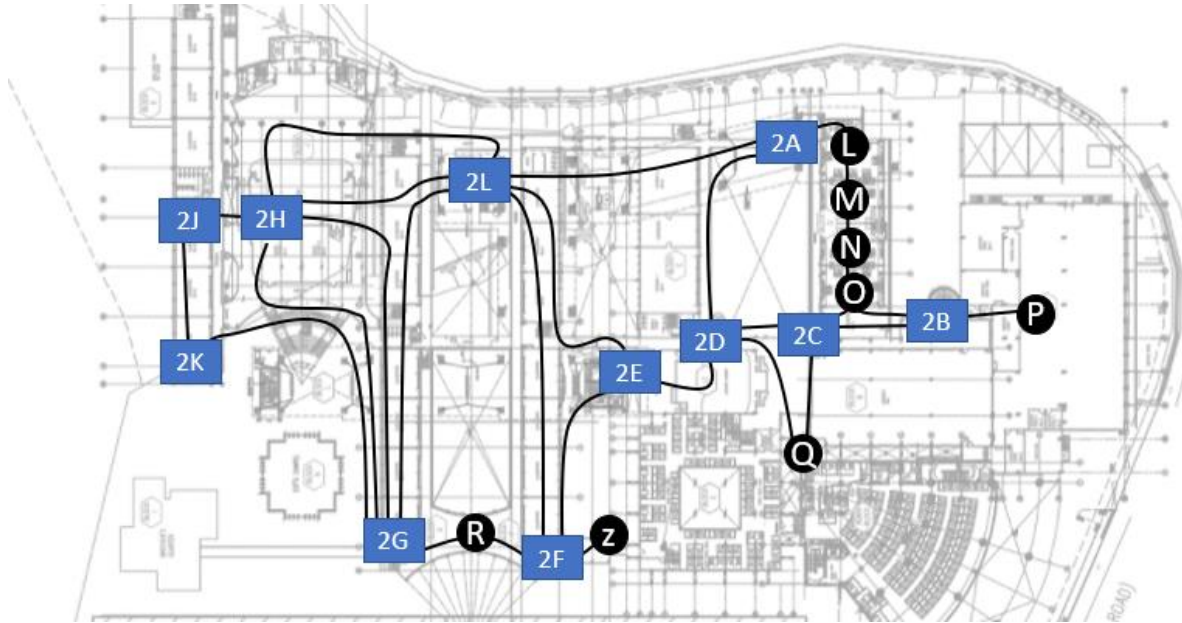


Figure 15: Layout of St. Joseph Institution 2nd floor with Vertices and Edges

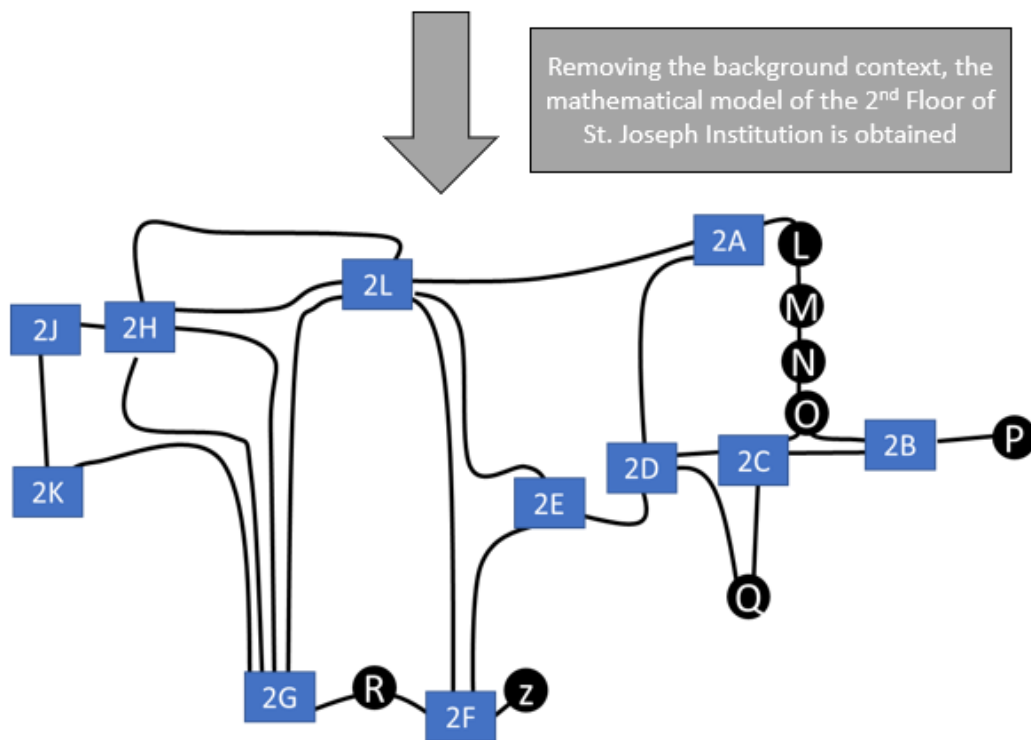


Figure 16: Mathematical Model of St. Joseph Institution 2<sup>nd</sup> Floor

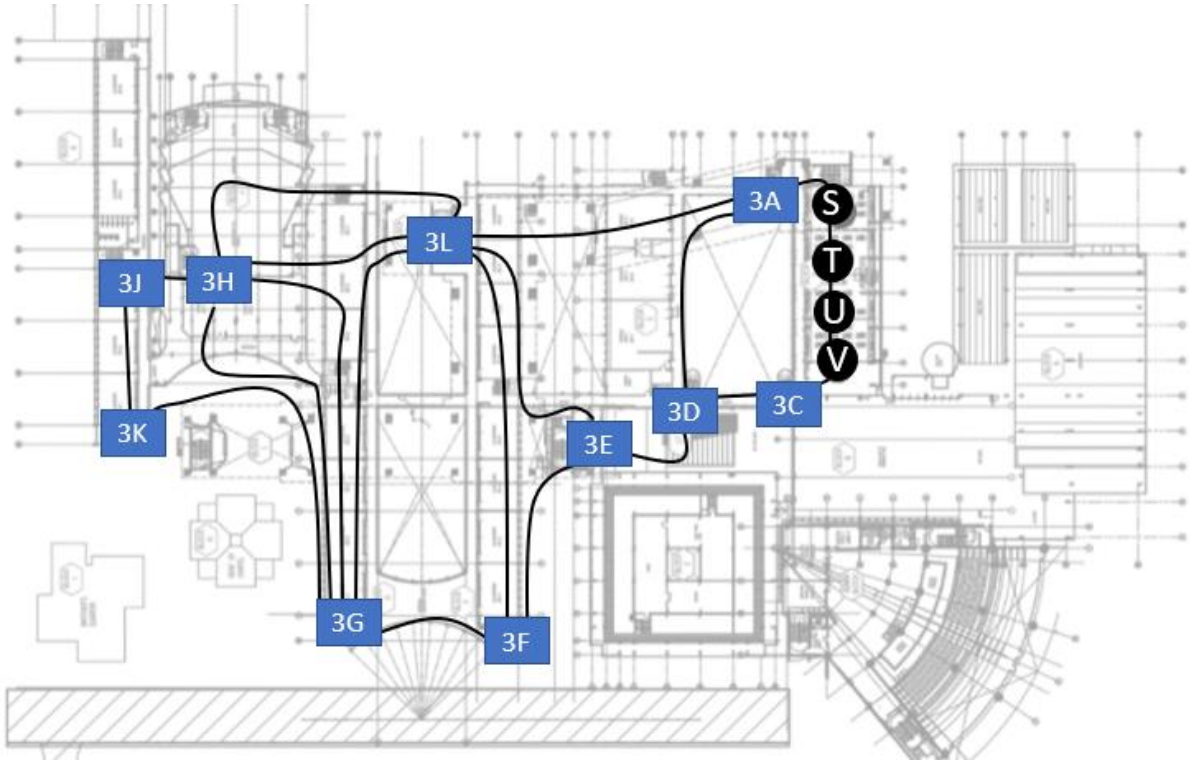


Figure 17: Layout of St. Joseph Institution 3rd floor with Vertices and Edges

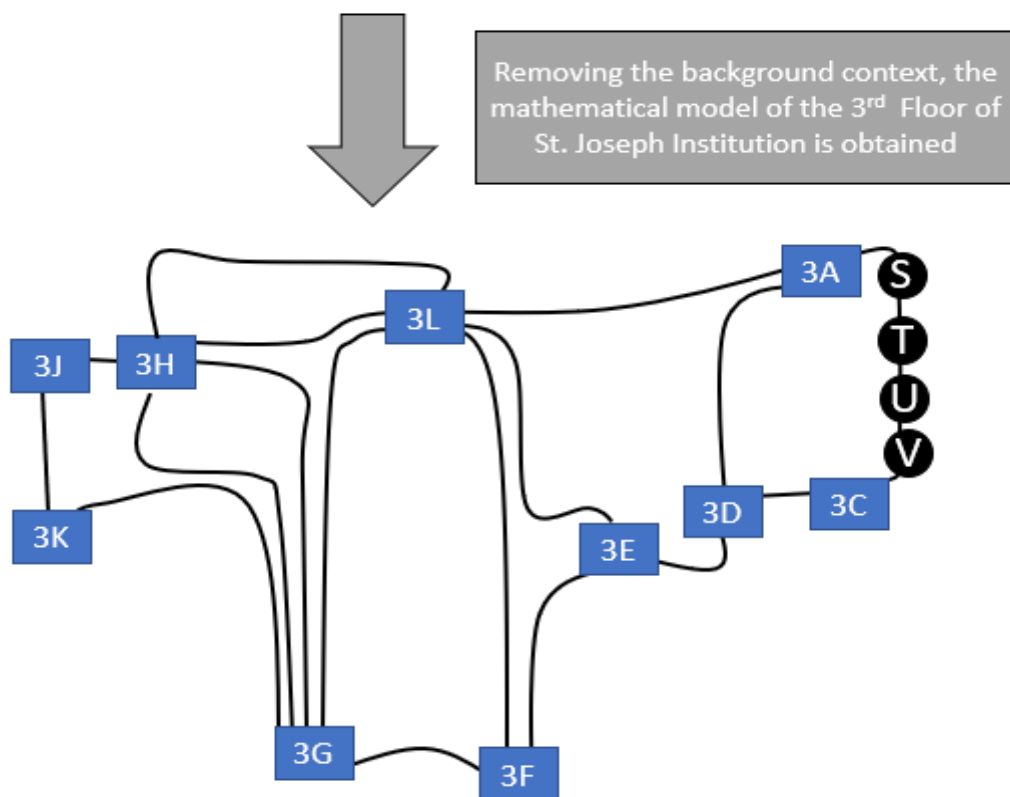


Figure 18: Mathematical Model of St. Joseph Institution 3<sup>rd</sup> Floor

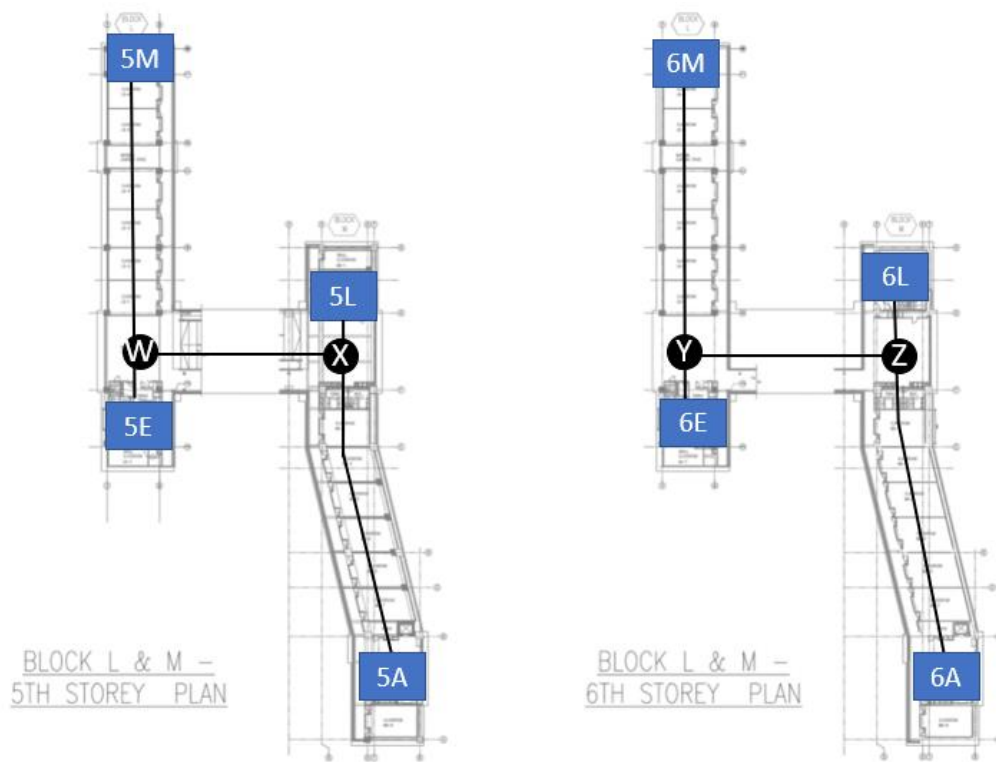


Figure 19: Layouts of St. Joseph Institution 5th and 6th floors with Vertices and Edges

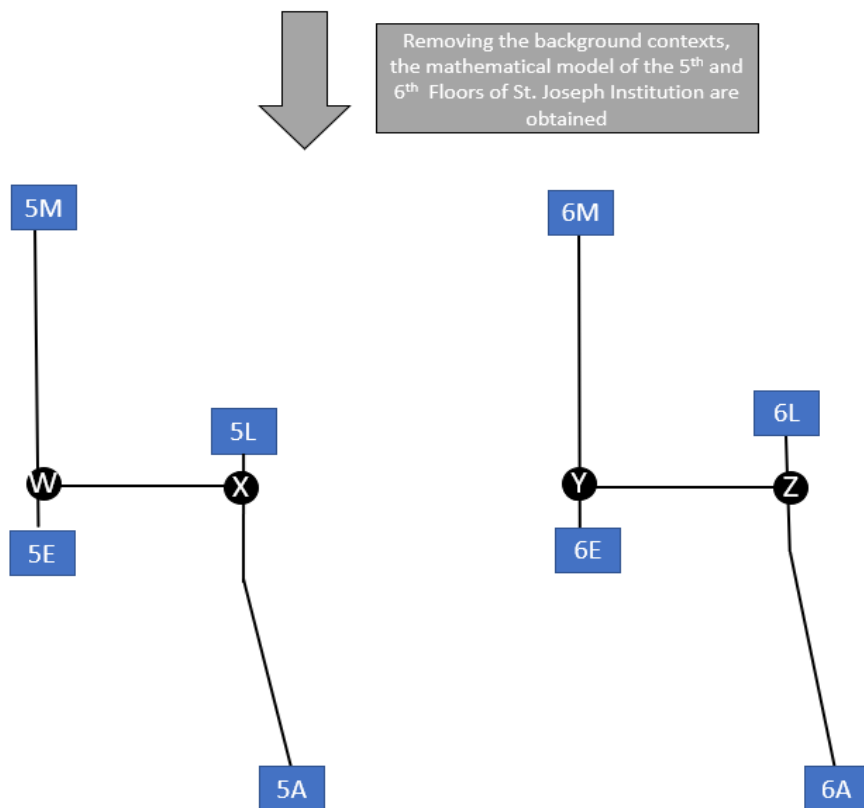


Figure 20: Mathematical Model of St. Joseph Institution 5<sup>th</sup> and 6<sup>th</sup> Floor

Layering the 2D layouts on top of one another, a graph representing a 3D view of each floor of the school is displayed in figure 21.

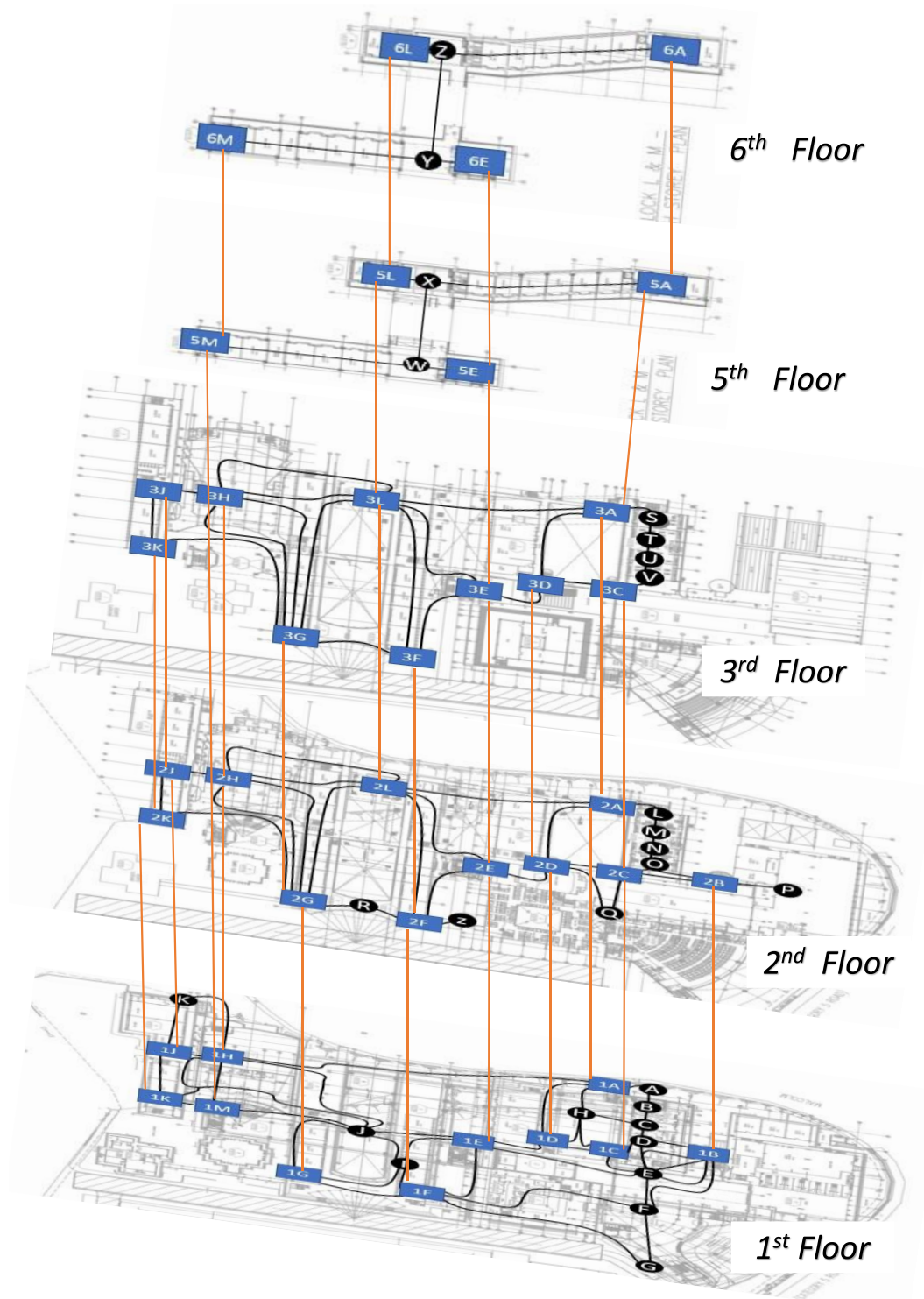


Figure 21: 3D view of St. Joseph Institution Layouts with Vertices and Edges

Removing the background context, the mathematical model representing St. Joseph Institution can be obtained

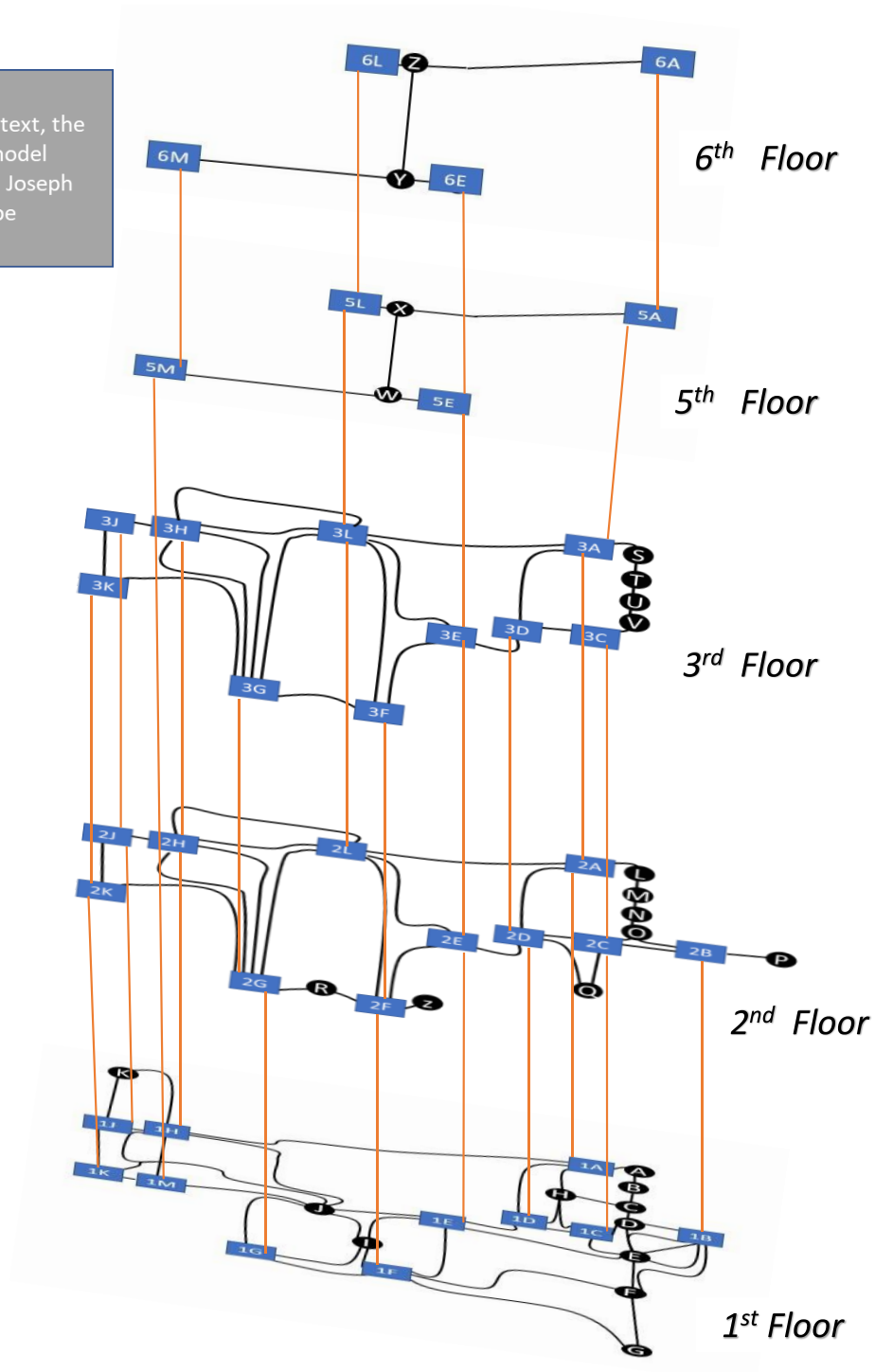


Figure 22: 3D Mathematical Model of St. Joseph Institution

Upon outlining the edges, the undirected, incomplete graph modelling the SJI compound is obtained with an order of  $|V| = 67$  and size  $|E| = 121$



## 3.2 Edge Weights

Since Dijkstra's Algorithm can only be applied to weighted graphs, to complete the mathematical model of the school, a numerical weight was assigned to each edge between vertices. Each edge was assigned a numerical weight directly proportional to the distance of the path it represents, taking on this value of distance.

The iPhone camera measuring application was used to measure the distances of each edge on school grounds, as depicted in figure 23. This tool was selected as it has been proven to provide precise measurements to the centimetre, ensuring accurate representation of the school by the mathematical model.



Figure 23: Measurement of School using iPhone measuring application

For example, the walkway between vertices G and F was measured to be a distance of 48.2 metres (rounded off to 1 d.p.). Hence, a weight of 48 is allocated to this edge (G,F) or (F,G), writing this in the table below.

Table 6: Sample Measurements

Edge Number	Vertices		Numerical Value
1	G	F	48.2

A table of these measurements can be found in Appendix C.

### 3.3 Multigraph Conversion

To determine the optimal locations for poster placement, the exact walks taken by students between vertices is to be ascertained since the optimal locations, are places where the maximum number of students are intercepted. To accomplish this, Dijkstra's Algorithm will be used to find the shortest path between vertices. However, Dijkstra's Algorithm cannot be applied to multigraphs (Biswas, Alam, & Doja, 2013).

This poses a problem as, due to the various multiple edges between vertices as depicted in figures 24 to 26, the graph model of the school is a **multigraph** and Dijkstra's Algorithm cannot be applied. As such, the largest valued multiple edges in the graph will be manually removed to acquire a **simple graph**.

With the aim being to determine the **shortest** walk between two vertices that a student travels, this practice is suitable as the longer edge would most likely be disregarded as it would merely lengthen the walk distance.



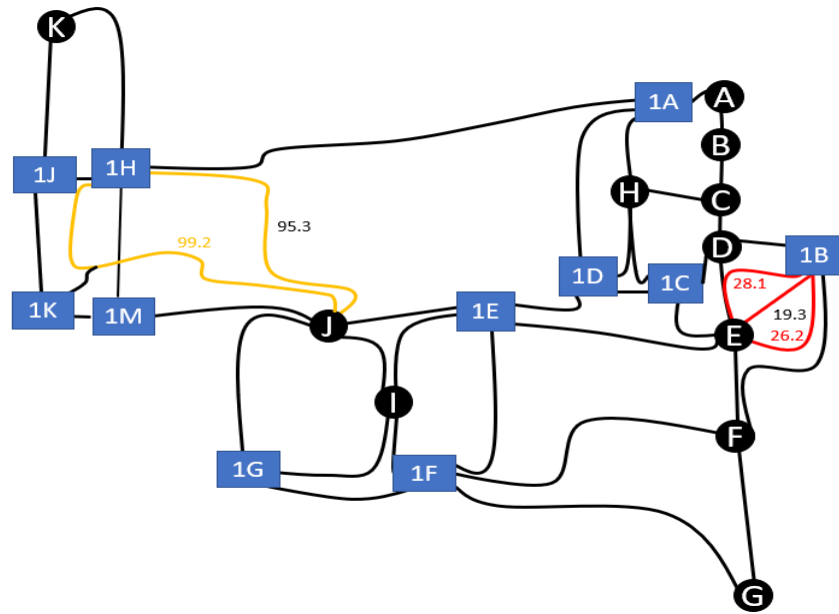


Figure 24: Multiple Edges on Graph representing St. Joseph Institution 1<sup>st</sup> Floor

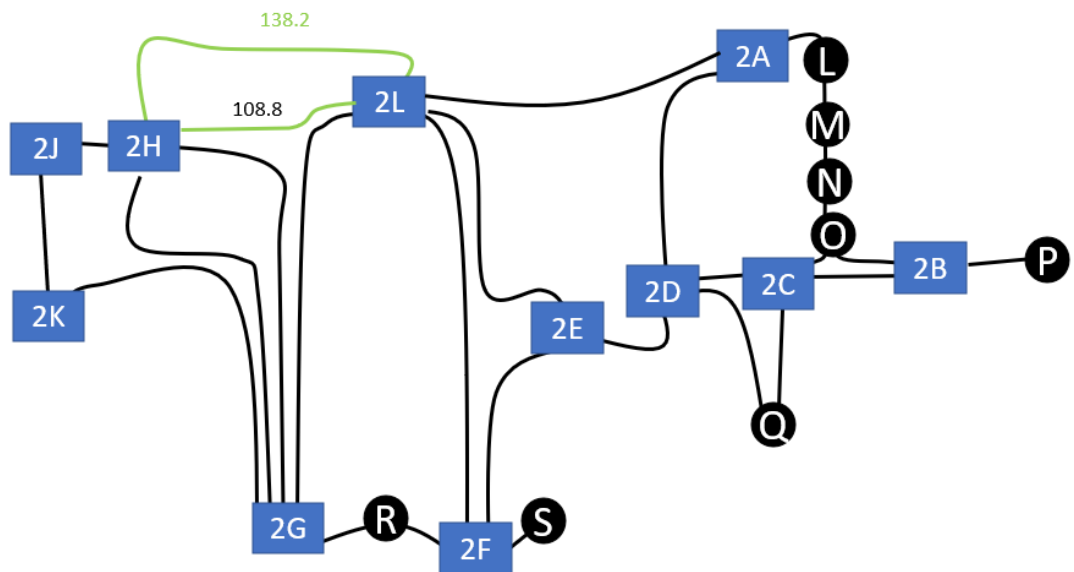


Figure 25: Multiple Edges on Graph representing St. Joseph Institution 2<sup>nd</sup> Floor

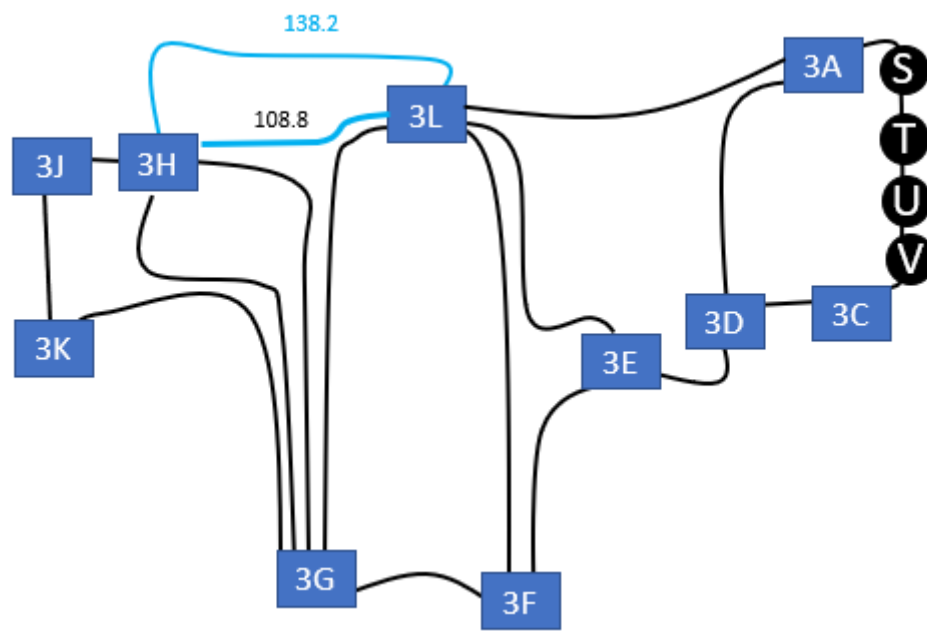


Figure 26: Multiple Edges on Graph representing St. Joseph Institution 3<sup>rd</sup> Floor

Upon removing the larger multiple edges (edges with coloured weights in figures 24 to 26), the simple graph in figure 27 is obtained.

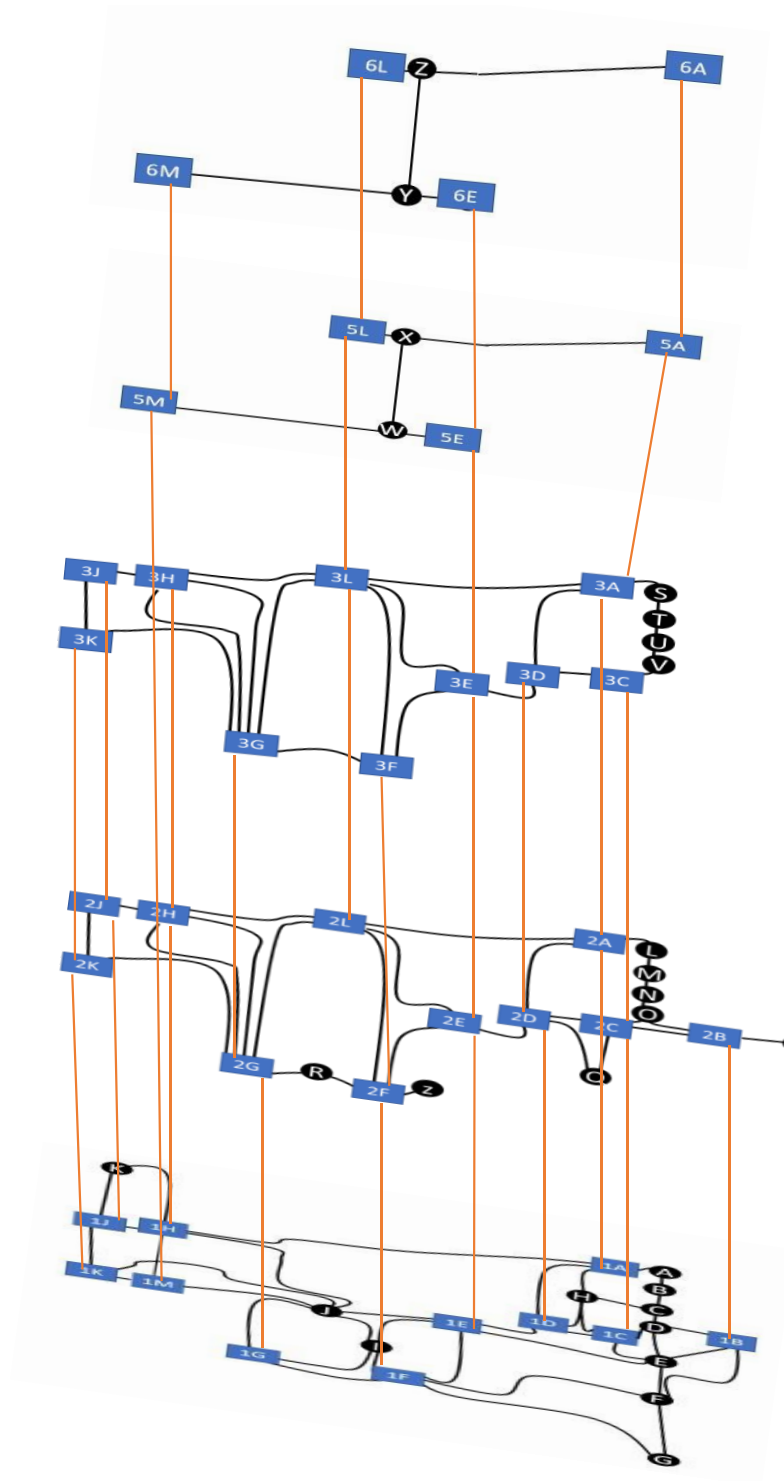


Figure 27: Simple Graph as a Mathematical Model Representing St. Joseph Institution

## 4. Analysis and Discussion

### 4.1 Determining the Paths taken by Students

Recalling that the 'optimal' placement of posters is at locations where the quantity of students intercepted is at a maximum, a comprehension of student motion throughout the school is required to determine the amenities the maximum quantity of students visit.

#### Data Collection

To determine the paths taken by individual students, a second survey was conducted to catalogue the **order of visitation** of various amenities that SJI students frequent during their free periods.

Since the people whose votes are considered during the council electoral process are students from the year 5 and year 6 cohorts, these two groups of students formed the target group of this survey. I enlisted the help of 4 of my close friends in disseminating the survey, each person stationed on a different floor to ensure a variety of unbiased results. The survey was conducted over a 5 day period during student break periods, obtaining a total of 242 responses.

Person	Area Covered by Person
A	1 <sup>st</sup> Floor
B	2 <sup>nd</sup> Floor
C	3 <sup>rd</sup> Floor
D	4 <sup>th</sup> Floor
E	5 <sup>th</sup> Floor



## Tabulating Data

Responses cannot be edited

### Survey on locations frequented in SJI compound during student free periods

Hello fellow Josephians! I am conducting some research for my Mathematics Extended Essay. I would like to obtain some relevant Data from SJI students year 5 and above, so please help me by answering the following questions. Data collected from this survey will only be used for my extended essay and will be kept confidential and anonymous. Thank you so much for taking the time to complete my survey!

**\*Required**

Which areas/amenities do you usually visit during your free/lunch period(s)? Please list in chronological order of visitation. \*

Level 1 Lab 1, Oval

Figure 29: Survey Sample Response

Table 8: Table of Vertices representing Amenities.

Vertex	Amenity	Vertex	Amenity
A	Level 1 Lab 1	N	Level 2 Lab 3
B	Level 1 Lab 2	O	Level 2 Lab 4
C	Level 1 Lab 3	P	Library
D	Level 1 Lab 4	Q	Annex
E	Canteen	R	Level 2 Benches
F	Foyer	S	Level 3 Lab 1
G	Prefect Benches	T	Level 3 Lab 2
H	Oval	U	Level 3 Lab 3
I	James Miller Room	V	Level 3 Lab 4
J	Secondary School Benches	W	Block B Level 5 Open Area
K	Art Room	X	Block A Level 5 Open Area
L	Level 2 Lab 1	Y	Block B Level 6 Open Area
M	Level 2 Lab 2	Z	Block A Level 6 Open Area
z	Annex 2		

In the sample response in figure 29, the student indicates the amenities they usually visit, initially visiting **Level 1 Lab 1 (vertex A)**, before traveling to the **Oval (vertex H)**. Since 7 other students had also indicated that they took this route, we can catalogue this as shown in table 9.

Table 9: Table of Sample Response Tabulation

Order of Vertex Visitation	Number of Students taking this route
A, H	8

A table of the compiled survey results can be found in table 10.

Table 10: Table of Survey Responses

Order of Vertex Visitation	Number of students taking this route
M, E, P, K	4
E, P	6
F, P, X	4
W, Y, Z	3
H, P, E	4
E, H	12
W, P, X	3
B, E, X	3
O, P, E	8
F, P, U	3
W, E, P	4
W, Y, E, P, K	1
X, P, E	2
W, Z, B	5
W, X, Z, Y	1

C, E, J	4
X, E, P, F, J, K	1
W, P	5
N, P	3
X, E	3
E	5
A, E	9
S, H	4
W, E, P, H	1
H, O, z,	5
G, J	10
X, R, K	4
B, E, P, H	3
X,R	4
A, H	8
K, I	2
B, H, F	1
W, E	5
W, J	2
X, E, Q	2
C, E, P	1
C, E, z	6
Y, z, E, S	2
U, Q, P, L	1
Z, P	5
Y, E	3
A, F	6



B, z	2
D, l	3
T, l	5
S, P	2
L, W, X	1
z, E, P	2
Z, z, J	1
Y, R, l	3
F, U, K	3
G, F, E	2
C, E	5
F, P	7
I, Q	4
I, K, P	2
T, G	2
L, P	6
F, l	8
J, G	3
z, A, R	4
J, G, E	1
P, S	6
S, Q, H	2

## Dijkstra's Algorithm

### What is Dijkstra's Algorithm?

Dijkstra's algorithm is a single source shortest path algorithm that simultaneously finds the sequences of vertices travelled in a path with the smallest weight from a single "source" vertex to all other vertices in a graph. (Techie Delight, 2021) In this essay, a variant of the algorithm that finds the sequence of vertices travelled from one vertex to another distinct vertex will be employed, terminating upon the visitation of the target vertex.

By implementing Dijkstra's Algorithm, **it is assumed that students always take the shortest walk between amenities.** This assumption is highly reflective of real life as most students would take the route of lowest travel time and distance to preserve their free time.

Among a variety of shortest path algorithms such as the Bellman-Ford or Floyd-Warshall algorithm, Dijkstra's Algorithm was selected due to the following factors,

1. **Suitability:** To find the shortest path between two distinct vertices, as compared to Floyd-Warshall algorithm (which exclusively solves for the shortest path between all vertex pairs) Dijkstra's algorithm, is more suitable as it can be adapted to terminate as soon as it has determined the shortest path from the source to target vertex.
2. **Permissibility:** Since the model does not contain any negative weights, Dijkstra's Algorithm, which can only be applied to graphs with non-negative weights, can be applied, always finding the shortest path between vertices.
3. **Ease of Implementation:** Since it can handle graphs with negative weights, the Bellman-Ford algorithm seems to be a more versatile and powerful option. However, Dijkstra's

algorithm is easier to implement and has a lower computational time, making it the more efficient choice.

4. **Simulatability:** Since Dijkstra's algorithm only considers the vertices directly adjacent to a certain vertex at any given time, Dijkstra's algorithm has an advantage in being able to simulate a student's choice of travel at a given time as students would similarly, likely take the immediately perceivable shortest path in their local vicinity.

## Application of Dijkstra's Algorithm

### **Step 1:**

Select the starting 'source' vertex as the current vertex. Assign a tentative distance of  $\infty$  to every vertex in the graph from the 'source' vertex.



### **Step 2:**

Marking the current vertex with a box to indicate its visitation. A visited vertex never be considered again - that is, the distance recorded is now minimal.



### **Step 3**

Assign candidate distances to all vertices adjacent to the current vertex such that,  
*Candidate distance = distance to current vertex + length of edge between current and adjacent vertex*

If this candidate distance < current tentative distance, overwrite it as the new tentative distance.



### **Step 4:**

If the target vertex is still unvisited, repeat steps 2 - 3, selecting the unvisited vertex with the lowest tentative distance as the next current vertex. Otherwise, the algorithm is terminated.

Figure 30: Application of Dijkstra's Algorithm (Joshi, 2017)

## Illustrating Dijkstra's Algorithm

To illustrate the algorithm, I attempt to establish the shortest path (A, H) as an example:

Table 11: Sample of Survey Response Tabulation

Order of Vertex Visitation	Number of students
A, H	5

The floor we will be focusing on is that of the 1<sup>st</sup> floor as depicted below in figure 31.

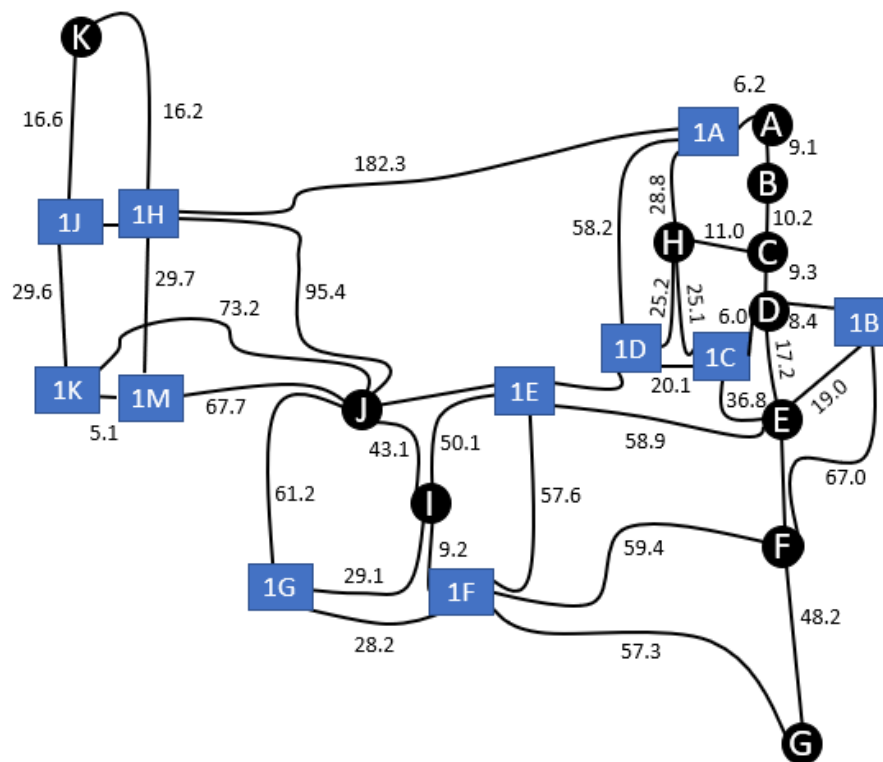
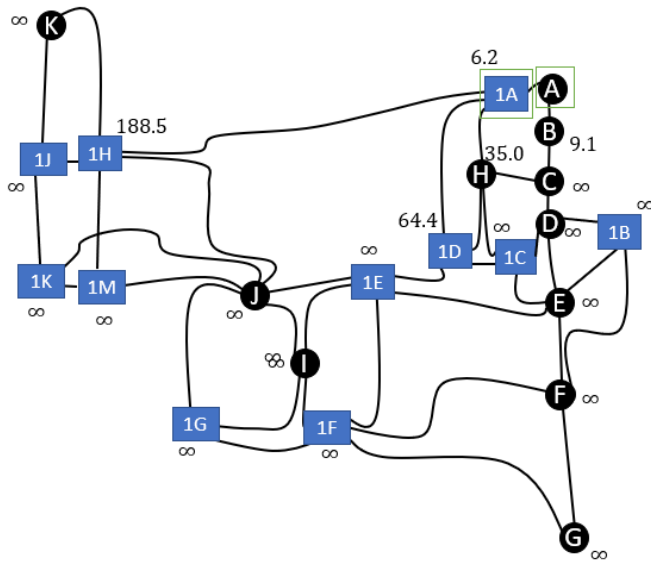


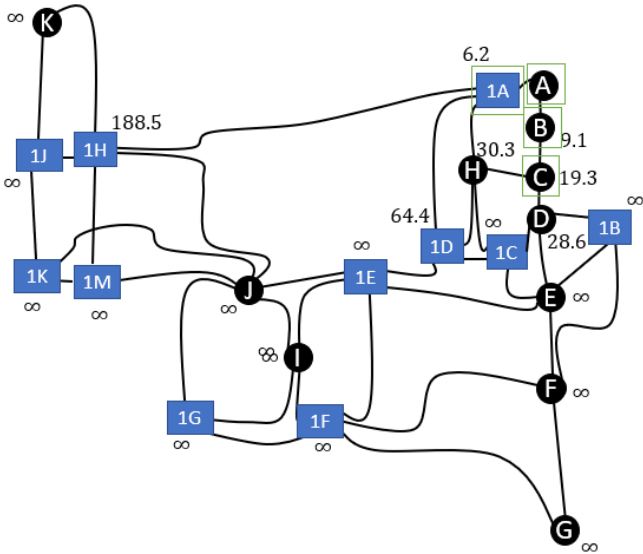
Figure 31: Graph representing St. Joseph Institution 1st floor with numerical weights

Table 12: Illustration of Dijkstra's Algorithm

Diagrammatic Representation	Explanation									
	<p>1. Current Vertex A</p> <p>Starting from vertex A, begin by assigning a tentative distance of infinity to each vertex in the system. Mark vertex A with a box to indicate its visitation.</p>									
	<p>2. Current Vertex: A</p> <p>We can subsequently assign a finite tentative distance to each adjacent vertex as determined by the weight of the edges connecting them to vertex A.</p> <table><tr><td></td><td colspan="2">Vertices adjacent to A</td></tr><tr><td></td><td>B</td><td>1A</td></tr><tr><td>Distance from A</td><td>9.1</td><td>6.2</td></tr></table>		Vertices adjacent to A			B	1A	Distance from A	9.1	6.2
	Vertices adjacent to A									
	B	1A								
Distance from A	9.1	6.2								



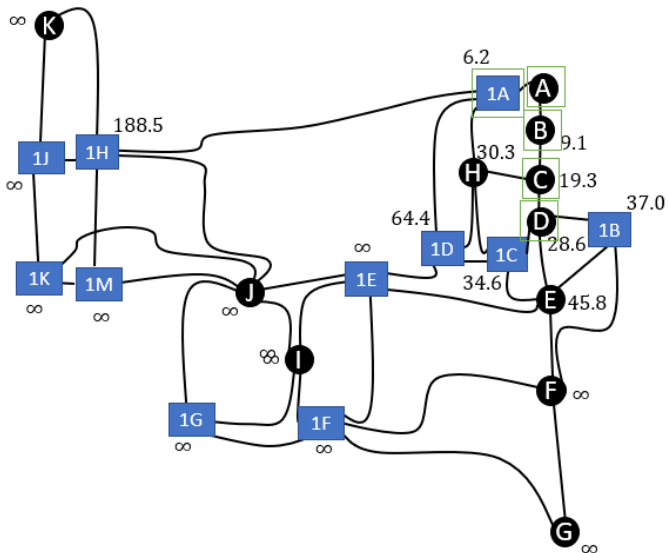
3. Current Vertex: 1A



#### 5. Current Vertex: C

	Vertices adjacent to C	
	D	H
Candidate	28.6	30.3
Distance from A		

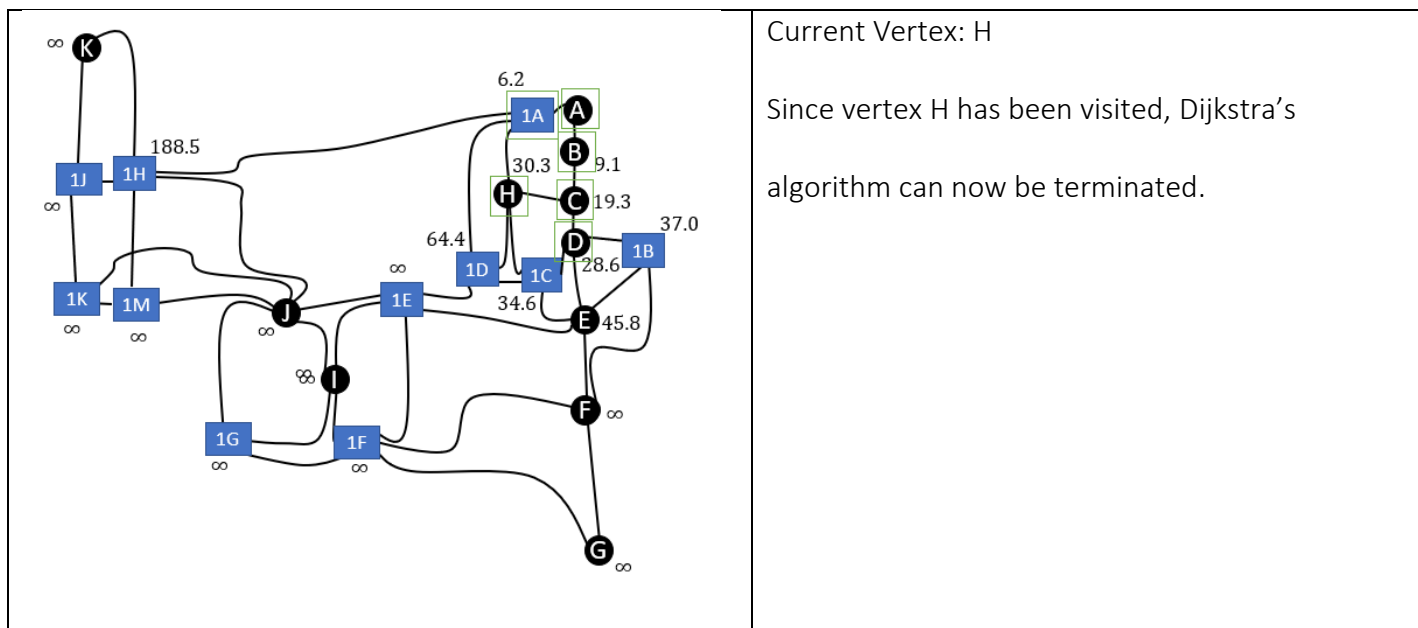
Since the candidate distance for vertex H obtained from this visitation (30.3) is smaller than that obtained from the visitation of vertex 1A (35.0), the previous candidate distance is replaced, from 35.0 to 30.3.



#### 6. Current Vertex: D

	Vertices adjacent to D		
	1C	E	1B
Candidate	34.6	45.8	37.0
Distance from A			





From this, we have ascertained that the final shortest distance is 30.3 m and the shortest walk taken is (A, B, C, H), which can be written in a table as such,

Table 13: Table of Sample Walk Taken Tabulation

Order of Vertex visitation	Number of students	Exact walk taken by student(s)	Exact walk taken by student(s) where only amenity vertices are considered
A, H	8	A, B, C, H	A, B, C, H

Manual application of the algorithm would be extremely tedious and time consuming, hence, a python code was used to apply the algorithm. This python code can be found in Appendix D.

A table documenting these walks taken by students can be found below in table 14.

Table 14: Results of Dijkstra's Algorithm

Order of Vertex visitation	Number of students	Exact walk taken by student(s)	Exact walk taken by student(s) where only amenity vertices are considered
M, E, P, K	4	M, N, O, 2B, 1B, E, 1B, 2B, P, 2B, O, 2C, 2D, 3D, 3E, 3H, 2H, 1H, K	M, N, O, E, P, O, K
E, P	6	E, 1B, 2B, P	E, P
F, P, X	4	F, E, 1B, 2B, P, 2B, O, 2C, 2D, 3D, 3E, 5E, W, X	F, E, P, O, W, X
W, Y, Z	3	W, 5E, 6E, Y, Z	W, Y, Z
H, P, E	4	H, C, D, 1B, 2B, P, 2B, 1B, E	H, C, D, P, E
E, H	12	E, D, C, H	E, D, C, H
W, P, X	3	W, 5E, 3E, 3D, 3C, 2C, O, 2B, P, 2B, O, 2C, 2D, 3D, 3E, 5E, W, X	W, O, P, O, W, X
B, E, X	3	B, C, D, E, 1E, 2E, 3E, 5E, W, X	B, C, D, E, W, X
O, P, E	8	O, 2B, P, 2B, 1B, E	O, P, E
F, P, U	3	F, E, 1B, 2B, P, 2B, O, 2C, 3C, V, U	F, E, P, O, V, U
W, E, P	4	W, 5E, 3E, 2E, 1E, E, 1B, 2B, P	W, E, P
W, Y, E, P, K	1	W, 5E, 6E, Y, 6E, 5E, 3E, 2E, 1E, E, 1B, 2B, P, 2B, O, 2C, 2D, 3D, 3E, 3H, 2H, 1H, K	W, Y, E, P, O, K
X, P, E	2	X, W, 5E, 3E, 2E, 2D, 2C, O, 2B, P, 2B, 1B, E	X, W, O, P, E
W, X, Z	5	W, X, 5L, 6L, Z	W, X, Z
W, X, Z, Y	1	W, X, 5L, 6L, Z, Y	W, X, Z, Y
C, E, J	4	C, D, E, 1E, J	C, D, E, J
X, E, P, F, J, K	1	X, W, 5E, 3E, 2E, 1E, E, 1B, 2B, P, 2B, 1B, E, F, E, 1E, 2E, 3E, 3H, 2H, 1H, K	X, W, E, P, E, F, E, K
W, P	5	W, 5E, 3E, 2E, 2D, 2C, O, 2B, P	W, O, P

N, P	3	N, O, 2B, P	N, O, P
X, E	3	X, W, 5E, 3E, 2E, 1E, E	W, E
E	5	E	E
A, E	9	A, B, C, D, E	A, B, C, D, E
S, H	4	S, 3A, 2A, 1A, H	S, H
W, E, P, H	1	W, 5E, 3E, 2E, 1E, E, 1B, 2B, P, 2B, 1B, D, C, H	W, E, P, D, C, H
H, O, z,	5	H, 1C, 2C, O, 2B, 1B, E, F, 1F, 2F, z	H, O, E, F, z
G, J	10	G, 1F, I, J	G, I, J
X, R, K	4	X, W, 5E, 3E, 2E, 2F, z, 2F, R, 2G, 2H, 1H, K	X, W, z, R, K
B, E, P, H	3	B, C, D, E, 1B, 2B, P, 2B, 1B, D, C, H	B, C, D, E, P, D, H
X,R	4	X, W, 5E, 3E, 3H, 2H, 2G, R	X, W, R
A, H	8	A, B, C, H	A, B, C, H
K, I	2	K, 1H, 2H, 2G, R, 2F, 1F, I	K, R, I
B, H, F	1	B, C, H, C, D, E, F	H, D, E, F
W, E	5	W, 5E, 3E, 2E, 1E, E	W, E
W, J	2	W, 5E, 3E, 2E, 1E, J	W, J
X, E, Q	2	X, 6A, 5A, 3A, 2A, 1A, A, B, C, D, E, D, 1C, 2C, Q	X, A, B, C, D, E, D, Q
C, E, P	1	C, D, E, 1B, 2B, P	C, D, E, P
C, E, z	6	C, D, E, F, 1F, 2F, z	C, D, E, F, z
Y, z, E, S	2	Y, 6E, 5E, 3E, 2E, 2F, z, 2F, 1F, F, E, D, C, B, A, 1A, 2A, 3A, S	Y, F, E, D, C, B, A, S
U, Q, P, L	1	U, V, 3C, 2C, Q, 2C, O, 2B, P, 2B, O, N, M, L,	U, V, Q, O, P, O, N, M, L
Z, P	5	Z, 6L, 5L, 3L, 3E, 2E, 2D, 2C, O, 2B, P	Z, O, P
Y, E	3	Y, 6E, 5E, 3E, 2E, 1E, E	Y, E

A, F	6	A, B, C, D, E, F	A, B, C, D, E, F
B, z	2	B, C, D, E, F, 1F, 2F, z	B, C, D, E, F, z
D, I	3	D, E, F, 1F, I	D, E, F, I
T, I	5	T, U, V, 3C, 2C, 1C, D, E, F, 1F, I	T, U, V, D, E, F, I
S, P	2	S, 3A, 2A, L, M, N, O, 2B, P	S, L, M, N, O, P
L, W, X	1	L, 2A, 3A, 5A, X, W, X	L, I, X, W, X
z, E, P	2	z, 2F, 1F, F, E, 1B, 2B, P	z, F, E, P
Z, z, J	1	Z, 6L, 5L, 3L, 3E, 2E, 2F, z, 2F, 1F, I, J	Z, z, I, J
Y, R, I	3	Y, 6E, 5E, 3E, 3H, 2H, 2G, R, 2F, 1F, I	Y, R, I
F, U, K	3	F, E, D, 1C, 2C, 3C, V, U, V, 3C, 3D, 3E, 2H, 1H, K	F, E, D, V, U, V, K
G, F, E	2	G, F, E	G, F, E
C, E	5	C, D, E	C, D, E
F, P	7	F, E, 1B, 2B, P	F, E, P
I, Q	4	I, 1F, F, E, D, 1C, 2C, Q	I, F, E, D, Q
I, K, P	2	I, 1F, 2F, R, 2G, 2H, 1H, K, 1H, 2H, 3H, 3E, 3D, 2D, 2C, O, 2B, P	I, R, K, O, P
T, G	2	T, U, V, 3C, 2C, 1C, D, E, F, G	T, V, D, E, F, G
L, P	6	L, M, N, O, 2B, P	L, M, N, O, P
F, I	8	F, 1F, I	G, I
J, G	3	J, I, 1F, G	J, I, G
z, A, R	4	z, 2F, 1F, F, E, D, C, B, A, 1A, 2A, 2L, 2G, R	z, F, E, D, C, B A, R
J, G, E	1	J, I, 1F, G, F, E	J, I, G, F, E
P, S	6	P, 2B, O, N, M, L, 2A, 3A, S	P, O
S, Q, H	2	S, T, U, V, 3C, 2C, Q, 2C, 1C, H	V, Q, H

## 4.2 Determining Poster Necessity at Various Amenities

Using the results from table 13, a numerical value proportional to the number of students passing through each non-stairwell amenity will be allocated to each amenity vertex, indicating the number of students passing by the vertex and hence, the level of necessity at the corresponding amenity.

Here, it is assumed that the quantity of students passing through these amenities is an appropriate indicator of poster necessity at the amenity.

It is also worth noting that only vertices will be taken into consideration for poster placement rather than edges.

*Table 15: Rationale for Poster Placement at Vertices*

Rationale:
Consider pasting a poster along an edge $(x,y)$ . Any student traveling along $(x,y)$ would pass by vertices $x$ and $y$ and intercept the poster. If you set the poster at either vertex $x$ or $y$ it is logical to assume that the same quantity of students will be intercepted by the same poster, However, if there are students who pass by either vertex $x$ or $y$ without traversing edge $(x,y)$ , the poster along edge $(x,y)$ is not intercepted. Hence, the quantity of students intercepted at a vertex $x$ or $y$ will definitely be $\geq$ the quantity of students intercepted along an edge $(x,y)$ in the graph.

Additionally, only vertices that represent amenities are taken into consideration as students tend to spend more time at amenities, increasing general awareness of the poster as opposed

to vertices representing stairwells where a student's interaction with a poster is fleeting and limited.

When only amenities are considered as vertices in the graph, the order of the graph is reduced to  $|V| = 27$ .

### Allocating Weights to Vertices

Consider the following sample of the results determined from the survey in table 16,

*Table 16: Sample of Data Collected from Dijkstra's Algorithm*

Order of Vertex visitation	Number of students	Exact walk taken by student(s)	Exact walk taken by student(s) where only amenity vertices are considered
A, H	8	A, B, C, H	A, B, C, H
U, Q, H	2	U, V, 3C, 2C, Q, 2C, 1C, H	U, V, Q, H

Via Dijkstra's algorithm, it was determined that the shortest walk taken by the 8 students was  $A \rightarrow B \rightarrow C \rightarrow H$ , as it was the shortest walk from A to H. Similarly the exact walk taken by 2 students from U to Q to H is  $U \rightarrow V \rightarrow Q \rightarrow H$ . A value of 8 can then be assigned to each of the vertices along walk  $\{A, B, C, H\}$  while a value of 2 is assigned to each vertex along walk  $\{U, V, Q, H\}$ , summing the values assigned on a single vertex to obtain a cumulative value (e.g.  $8 + 2 = 10$  for vertex H). From this, we obtain the values in table 17.

Table 17: Sample of Numerical weights assigned to Vertices

Vertex	Numerical Value	Vertex	Numerical Value
A	8	U	2
B	21	V	2
C	34	Q	2
H	10		

Repeating this process for every walk, we obtain the numerical values displayed in table 15 and figure 31

Table 18: Table of Numerical Weights assigned to Vertices

Vertex	Numerical Value	Vertex	Numerical Value
A	17	O	60
B	26	P	84
C	50	Q	9
D	84	R	19
E	109	S	8
F	65	T	7
G	26	U	12
H	43	V	16
I	47	W	59
J	21	X	30
K	14	Y	11
L	12	Z	15
M	9	z	29
N	11		

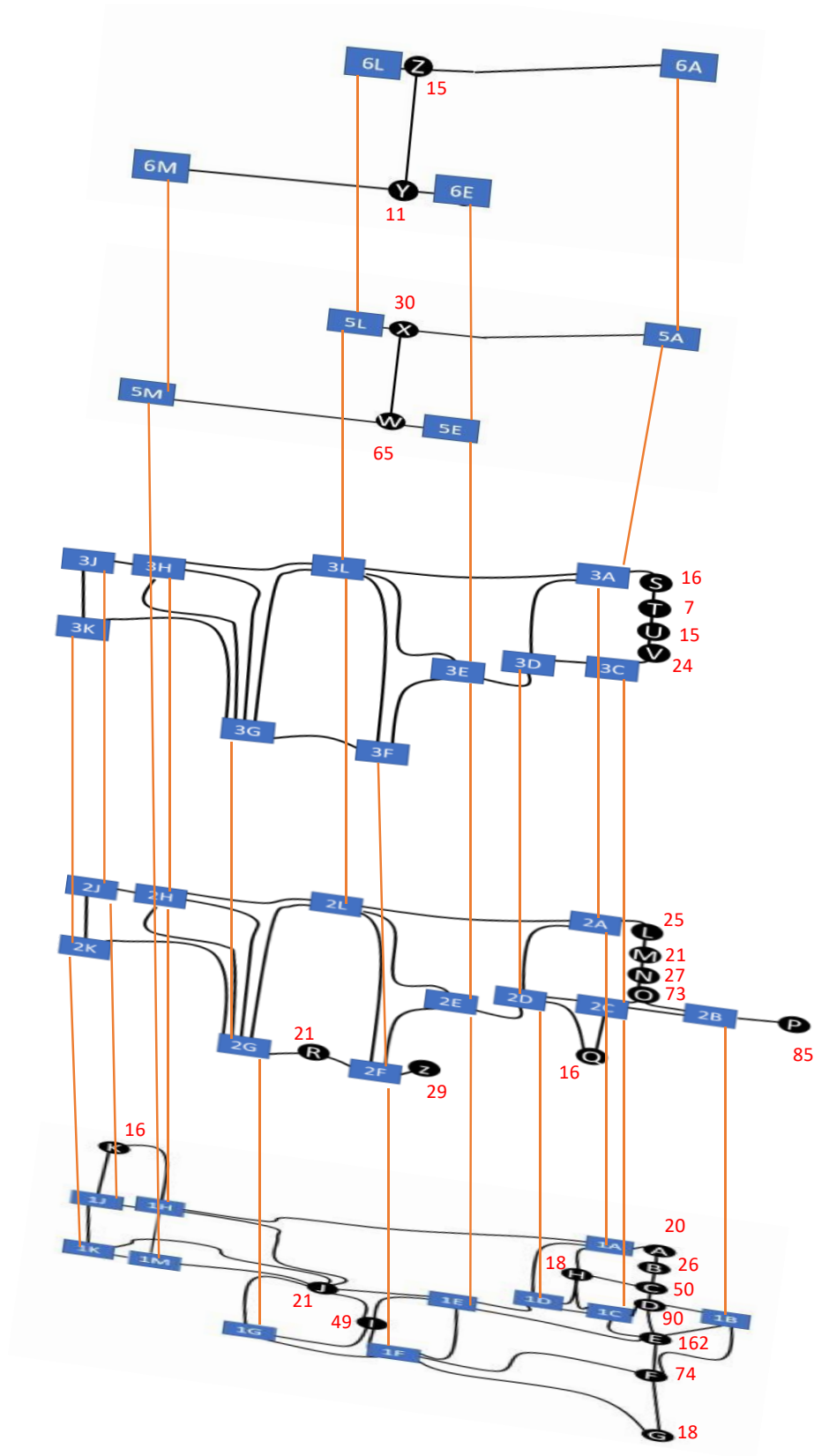


Figure 32: Graph representing 1st, 2nd, 3rd, 5th and 6th floors of St. Joseph Institution with Numerical Vertex Weights



## A Greedy Algorithm

**Greedy algorithms** are commonly used in approaching various optimisation problems by considering the locally optimal solution at any point in time. (ScienceDirect, 2019) This means that at each point of decision making, the algorithm chooses what, at present, appears to be the best outcome without considering any future or past implications. This nature of greedy algorithms only taking the locally optimal solution poses a flaw as the globally optimal solution is not guaranteed.

However, an advantage of greedy algorithms is their ease of implementation and computational efficiency, a result of its short-sighted scope of application. Additionally, considering the relatively small order of the graph ( $|V| = 27$ ), a globally optimal solution is likely to be obtained.

A greedy algorithm will hence be applied to the graph representing the St. Joseph Institution compound to determine the optimal poster placements.

## Application of Greedy Algorithm

1. The vertex with the greatest numerical weight is selected and a poster is placed, indicated by a red cross on the vertex



2. All students with the selected amenity as part of their walk is eliminated from the model and the numerical weight they contribute to each vertex is deducted from the vertex. (E.g. If 6 students take a walk along vertices A, B and C, if a poster is placed at B , a value of 6 would be deducted from the values assigned to vertices A and C .)



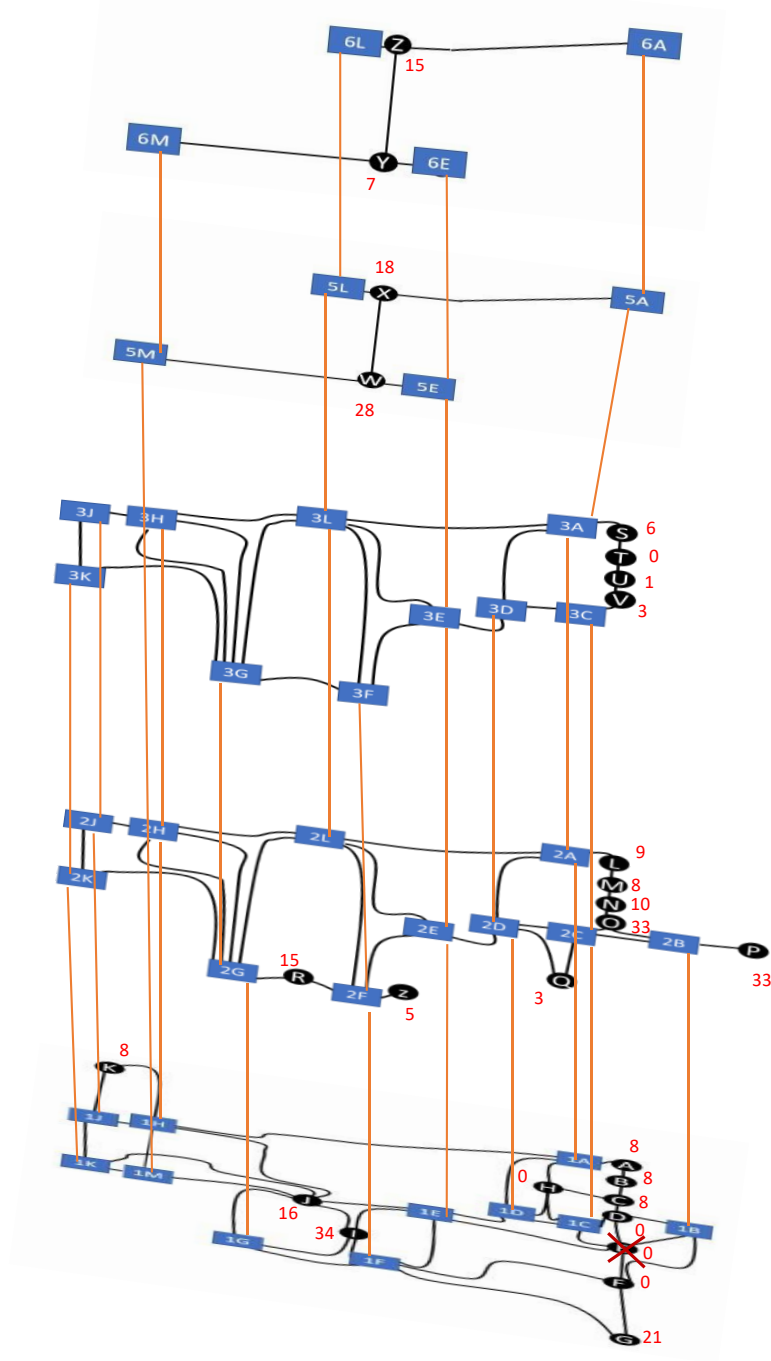
3. Steps 1-2 are repeated until the weight of all vertices is 0

*Figure 33: Application of Greedy Algorithm*

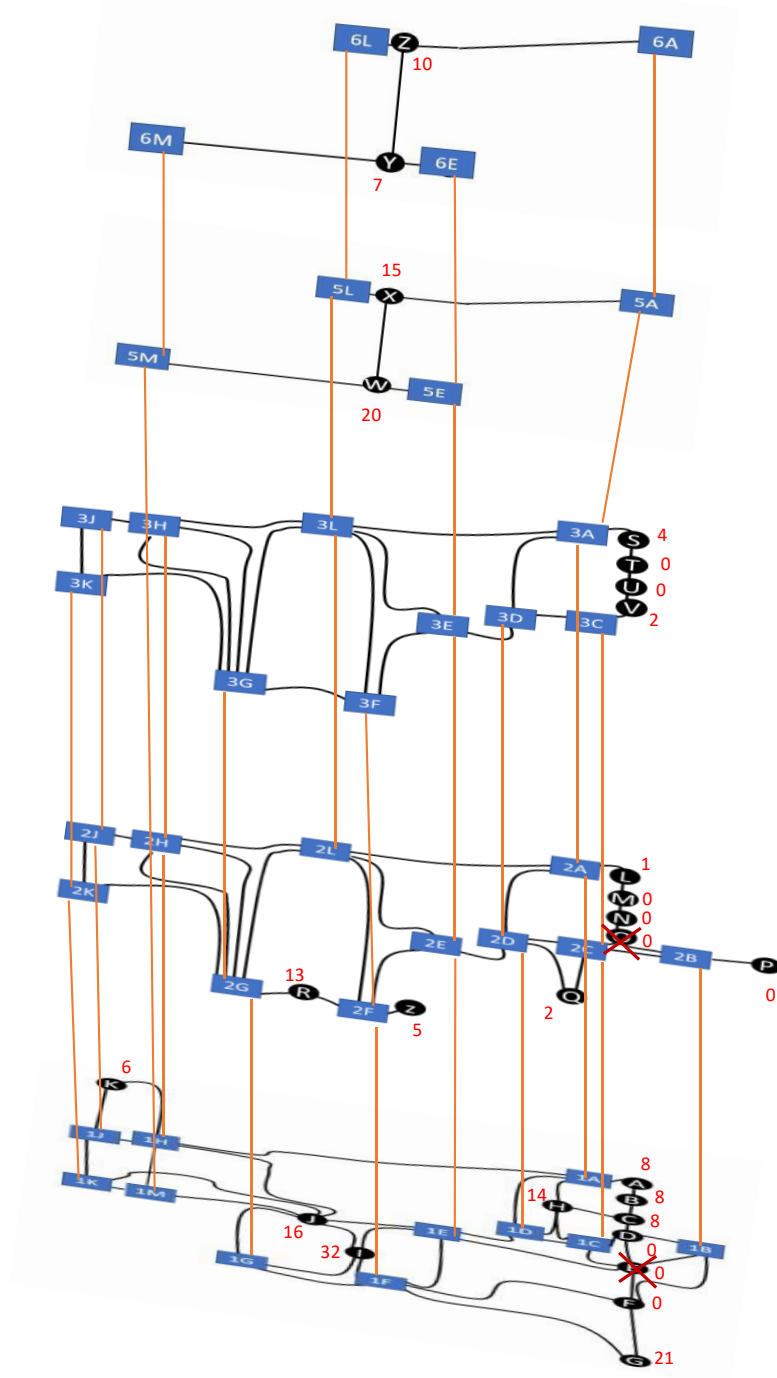
This greedy algorithm can now be applied to the graph representing the St. Joseph Institution compound, as detailed in table 19.

Table 19: Illustration of Greedy Algorithm Application

## After First Placement: E

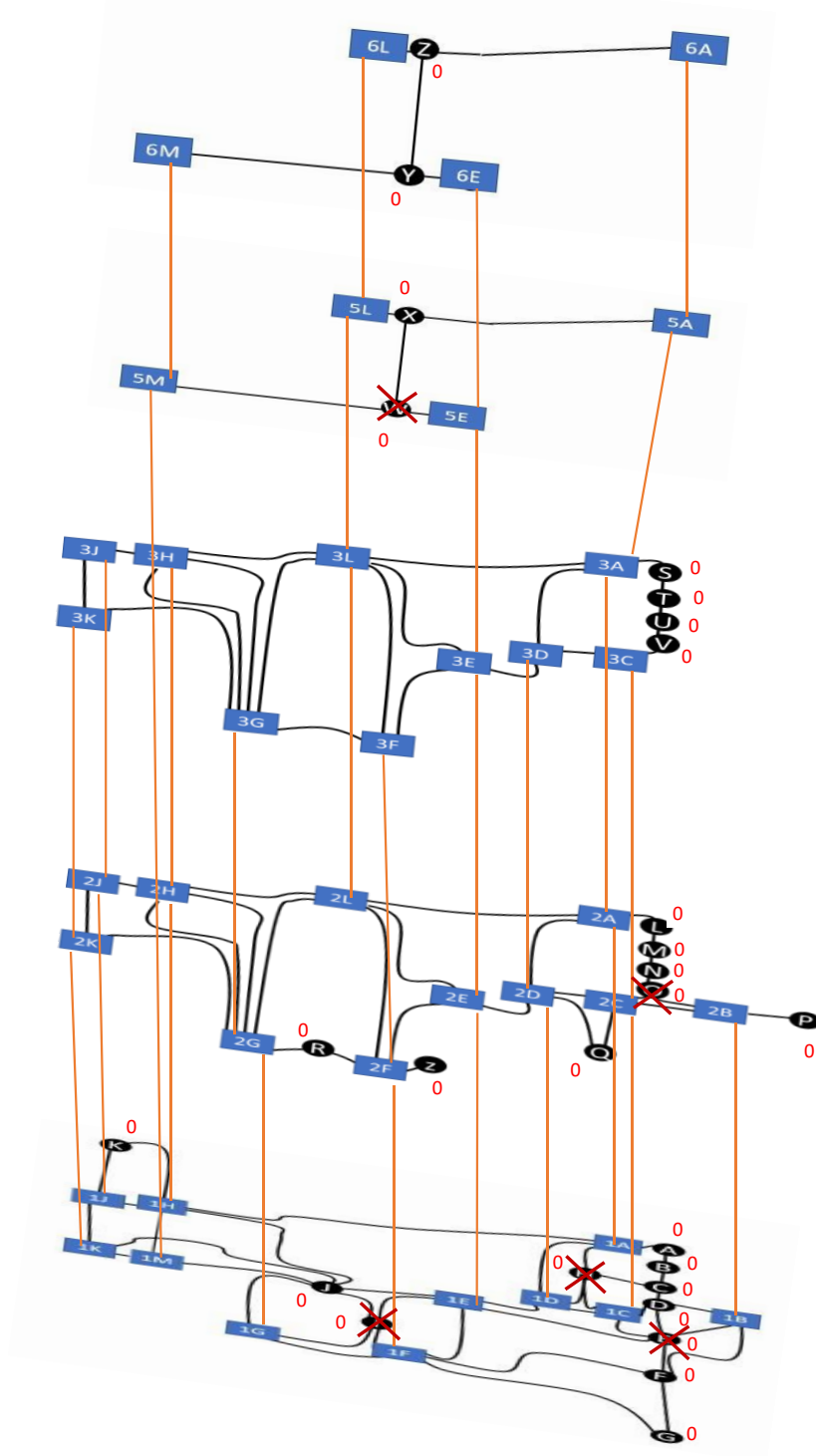


As the largest weight of 162 was allocated to vertex E, a poster was situated at E, marking it with a red cross. All students who had walked past vertex E are then eliminated. The algorithm is repeated for all subsequent vertices.



Currently, Vertices P and O both possess the largest weight of 33. Comparing the degrees (number of edges attached) of vertices P and O, 1 and 3 respectively, O was selected as the vertex for a poster to be placed due to its higher degree and therefore implied increased connectivity. All students passing through vertex O are now eliminated.

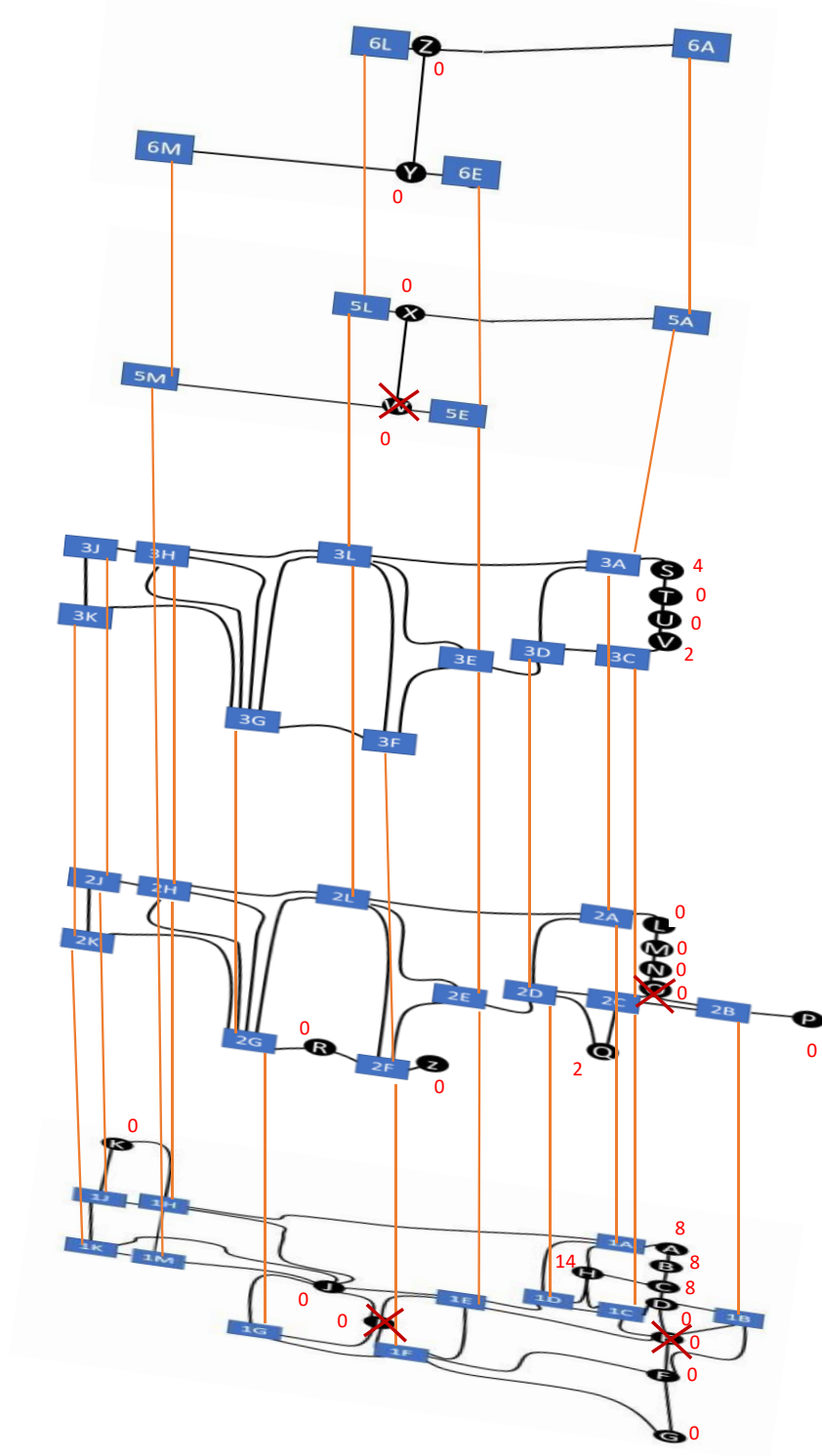
## Third Placement: I



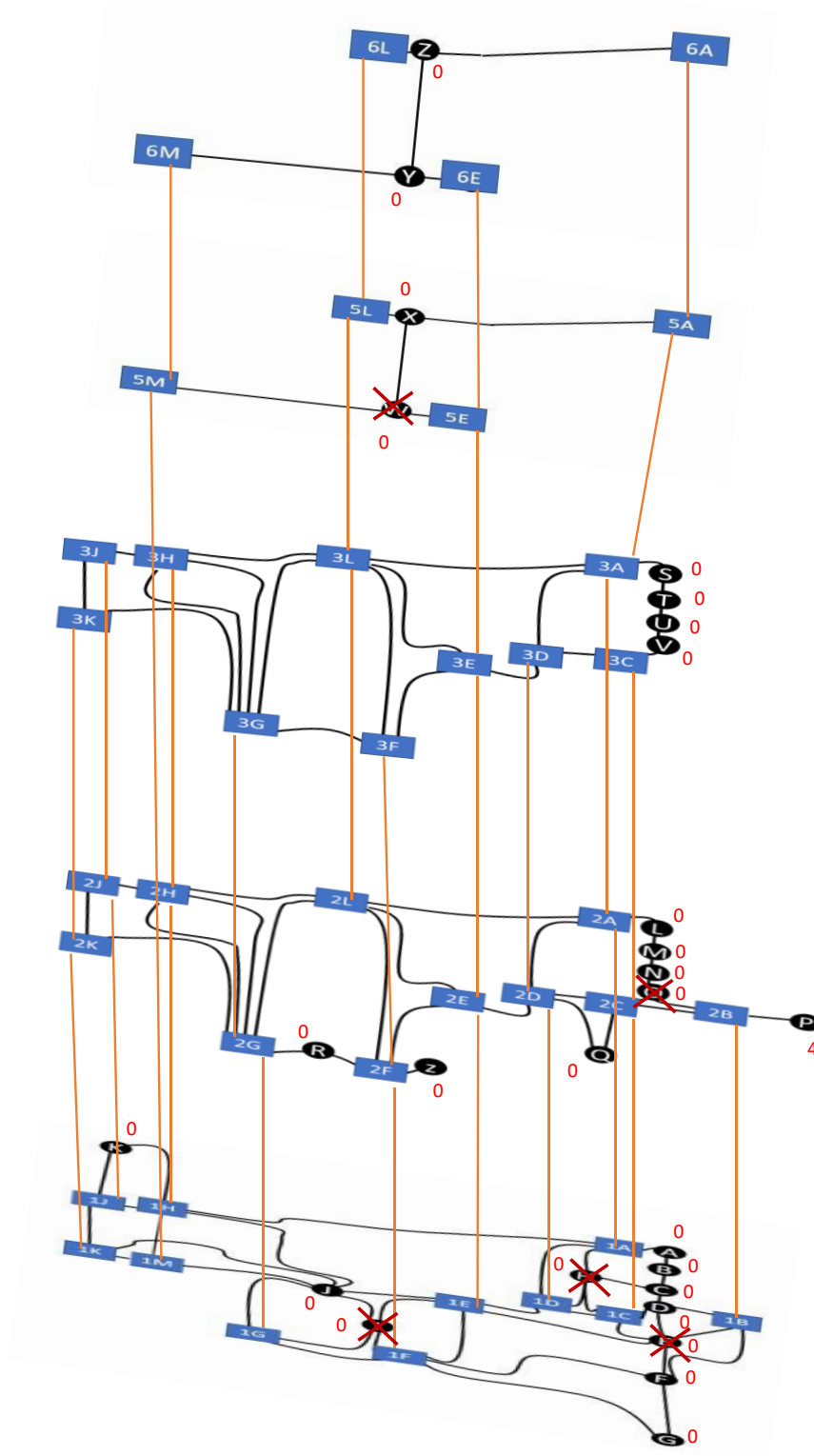
Vertex I now possesses the largest value (32), and is hence selected as the third vertex for poster placement.

All students passing through I are then eliminated.

## Fourth Placement: W



Vertex W now possesses the largest numerical weight of 15. It is hence selected as the third vertex for poster placement. All student walks passing through vertex W are eliminated.



With vertex H now carrying the largest weight, it is selected as the final vertex for poster placement. Since the weight of all vertices = 0 the algorithm can now be terminated

The suggested solution indicates that **5 posters** should be placed at vertices **E, O, I, W and H**.

The vertices selected in this suggested arrangement are translated to their corresponding amenities in table 20.

*Table 20: Table of Suggested Amenities for Poster Placement*

Vertex Representing Amenity	Amenity of Suggested Placement
Canteen	E
Level 2 Lab 4	O
Counsellor's Room	I
Block B Level 5 Open Area	W
Oval	H



## 5. Evaluation and Conclusion

### 5.1 Evaluating the Solution

#### Objective Functions

The goal of an **optimisation process** is to obtain parameter values that **maximises or minimises** the value of an **objective function** based on a pre-defined set of conditions. (AnyLogic, n.d.)

**Objective functions** are mathematical expressions that describes the results of an operation which uses optimisation parameters as inputs. (AnyLogic, n.d.) The suggested solution to the optimisation problem of poster placement will hence be evaluated via objective functions. In table 20, the two conditions for optimal poster placement are translated into objective functions.

Table 21: Derivation of Objective Functions

Condition		Objective Function	Objective
1	The quantity of students intercepted are maximised	$O_1 = \sum_{w \in W} s_w X_w$	Maximise
2	The quantity of posters used is minimised	$O_2 = \sum_{v \in V} Y_v$	Minimise
<p><math>w</math> denotes a certain walk .</p> <p><math>W</math> denotes a set of all walks.</p> <p><math>v</math> denotes a certain vertex made to represent a school amenity.</p> <p><math>V</math> denotes the set of vertices made to represent various school amenities</p> <p><math>s_w</math> denotes the number of students along a certain path <math>w</math></p> <p><math>X_w</math> and <math>Y_v</math> denote binary variables that determine the existence of a poster situated at a vertex <math>V</math> or along a walk <math>W</math>, defined in table 21.</p>			

Table 22: Binary Variable Definitions

Binary Variable Definitions	
$X_w$	$\begin{cases} 0 & \text{if no poster is situated along path } w \\ 1 & \text{otherwise} \end{cases}$
$Y_v$	$\begin{cases} 0 & \text{if no poster is situated at vertex } v \\ 1 & \text{otherwise} \end{cases}$

## Objective Function 1 Calculation Example

Consider the set of results in table 23.

Table 23: Sample Results

Order of Vertex Visitation	Number of students	Exact Walk taken by student(s)	Exact Path taken by student(s) where only vertices that represent a facility is considered
E, H	12	E, D, C, H	E, D, C, H

For objective function 1,

$$O_1 = \sum_{w \in W} s_w X_w$$

For this example,  $w$  denotes the walk E, D, C, H. Since a poster has been placed at both vertices E and H,  $X_w = 1$ , while  $s_w$  which represents the number of students along this walk is 12.

$$\therefore s_w X_w = (1)(12) = 12$$

Summing the values of  $s_w X_w$  for all walks  $w$ , the value of  $O_1$  can be found.

## Calculating Objective Function Values

Table 24: Objective Function Calculations for Suggested Poster Arrangement

Objective Function	Objective Function Calculations
1	$  \begin{aligned}  O_1 = & 3(1) + 6(1) + 4(1) + 3(1) + 4(1) + 12(1) + 3(1) + 3(1) \\  & + 6(1) + 3(1) + 4(1) + 1(1) + 2(1) + 5(1) + 1(1) + 4(1) + 1(1) \\  & + 1(1) + 5(1) + 3(1) + 3(1) + 5(1) + 9(1) + 4(1) + 1(1) + 5(1) \\  & + 10(1) + 4(1) + 3(1) + 4(1) + 8(1) + 2(1) + 1(1) + 5(1) + 2(1) \\  & + 2(1) + 1(1) + 6(1) + 2(1) + 1(1) + 5(1) + 3(1) + 6(1) + 2(1) \\  & + 3(1) + 5(1) + 2(1) + 1(1) + 2(1) + 2(1) + 1(1) + 3(1) + 1(1) \\  & + 3(1) + 2(1) + 5(1) + 7(1) + 4(1) + 2(1) + 2(1) + 2(1) + 6(1) \\  & + 8(1) + 3(1) + 4(1) + 1(1) + 6(1) + 2(1) = 242  \end{aligned}  $
	$\% \text{ of students intercepted} = \frac{242}{242} \times 100\% = 100\%$
2	$  \begin{aligned}  O_2 = & 0 + 0 + 0 + 0 + 1 + 0 + 0 + 1 + 1 + 0 + 0 + 0 + 0 + 0 + 1 + 0 \\  & + 0 + 0 + 0 + 0 + 0 + 0 + 1 + 0 + 0 + 0 + 0 = 5  \end{aligned}  $
	$\% \text{ of amenity vertices that have a poster placed} = \frac{5}{27} \times 100\% = 18.5\%$

The suggested solution achieves optimality for  $O_1$ , where  $O_1 = 242$  yielding the maximum value as indicated by the 100% rate of students intercepted.

However, the achievement of optimality for  $O_2$ , where  $O_2 = 5$  remains up to the debate as while the solution yields a relatively small percentage, it is unknown whether a smaller percentage could be obtained.

## 5.2 Real Life Considerations

To further evaluate the effectiveness of the suggested solution in achieving the objectives of investigation, the initial real-life arrangement of posters among the school compound can be taken as a basis for comparison.

In real life, the posters were situated at the locations as catalogued in table 25 and figure 34.

*Table 25: Real-Life Amenities for Poster Placement and Vertices*

Location	Vertex
Canteen	E
Foyer	F
Secondary School Benches	J
Level 5 Block A Open Area	X
Level 6 Block B Open Area	Y
Level 6 Block A Open Area	Z

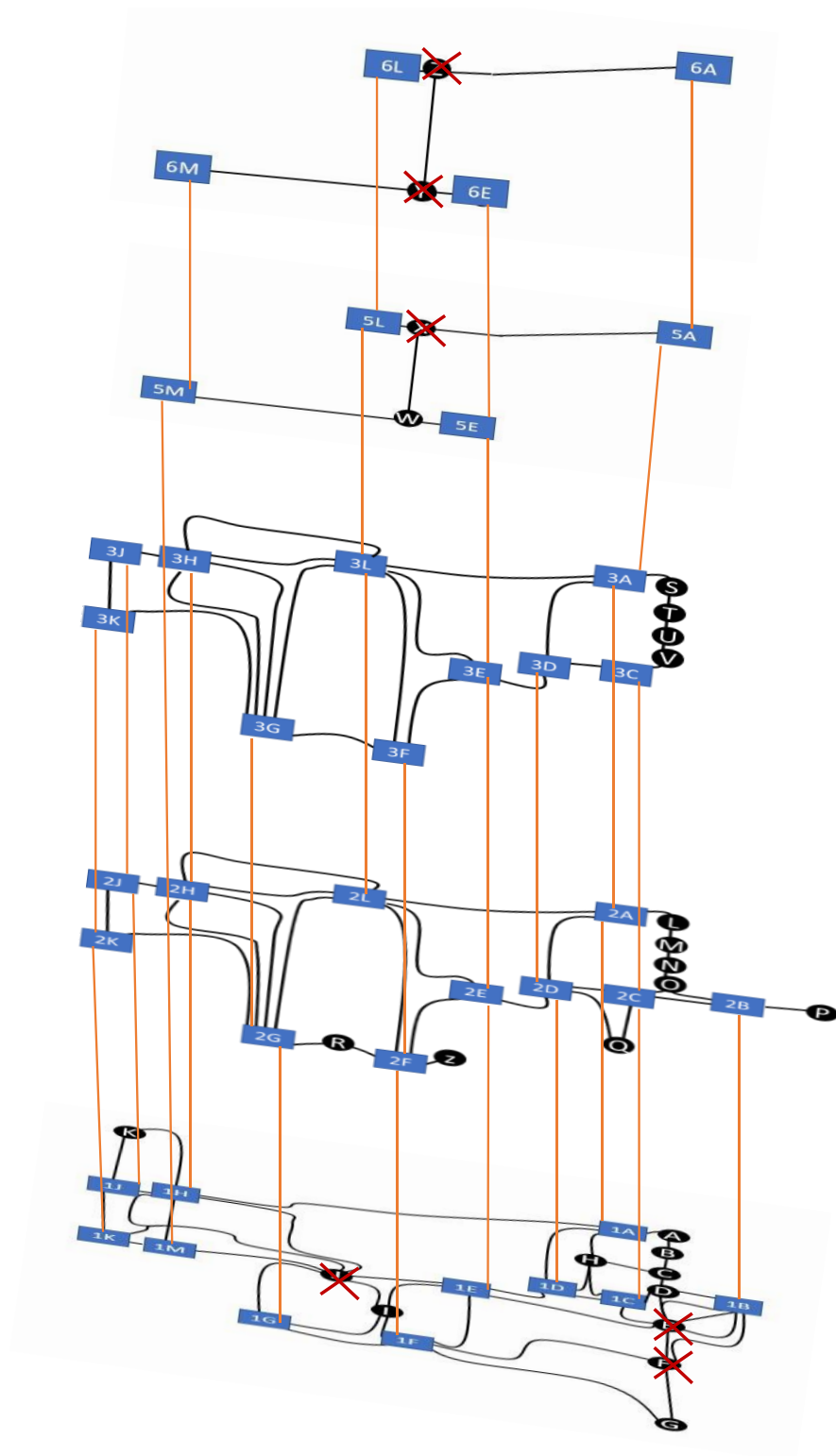


Figure 34: Real Life Poster Placement

## Calculating Objective Function Values

Objective functions  $O_1$  and  $O_2$  can once again be employed to evaluate the optimality of the Real-Life Poster Placement against the suggested solution.

Table 26: Objective Function Calculations for Real-Life Poster Arrangement

Objective Function	Objective Function Calculations
1	$  \begin{aligned}  M_1 = & 3(1) + 6(1) + 4(1) + 3(1) + 4(1) + 12(1) + 3(1) \\  & + 3(1) + 6(1) + 3(1) + 4(1) + 1(1) + 2(1) + 5(1) + 1(1) \\  & + 4(1) + 1(1) + 1(1) + 5(0) + 3(0) + 3(1) + 5(1) + 9(1) \\  & + 4(0) + 1(1) + 5(1) + 10(1) + 4(1) + 3(1) + 4(1) + 8(0) \\  & + 2(0) + 1(1) + 5(1) + 2(1) + 2(1) + 1(1) + 6(1) + 2(1) \\  & + 1(0) + 5(1) + 3(1) + 6(1) + 2(1) + 3(1) + 5(1) + 2(0) \\  & + 1(1) + 2(1) + 2(1) + 1(1) + 3(1) + 1(1) + 3(1) + 2(1) \\  & + 5(1) + 7(1) + 4(1) + 2(0) + 2(1) + 2(1) + 6(0) + 8(0) \\  & + 3(0) + 4(1) + 1(1) + 6(0) + 2(0) = 193  \end{aligned}  $
	$\% \text{ of students intercepted} = \frac{193}{243} \times 100\% = 79.4\%$
2	$  \begin{aligned}  M_2 = & 0 + 0 + 1 + 1 + 0 + 0 + 0 + 1 + 0 + 0 + 0 + 0 + 0 + 0 \\  & + 0 + 0 + 0 + 1 + 1 + 1 = 6  \end{aligned}  $

## 5.3 Comparisons

Table 27: Real Life and Suggested Vertices for Poster Placement

	Real Life Placement	Suggested Placement
Number of Posters	6	5
Vertices with Posters Placed	E, F, J, X, Y, Z	E, O, I, W, H

Table 28: Objective Function 1 Comparison

Comparison of objective function 1 fulfilment		
	Real Life Placement	Suggested Placement
Quantity of Students considered	243	
Quantity of Students Intercepted	193	242
% of Students Intercepted	79.4%	100%

From the calculations of objective function 1, it is observed that the suggested placement was able to intercept a larger quantity and percentage of students of as compared to the real-life situation, posing a significant improvement compared to the real-life placement.



Table 29: Objective Function 2 Comparison

Comparison of objective function 2 fulfilment		
	Real Life Placement	Suggested Placement
Total quantity of vertices	21	
Quantity of Vertices with Posters Placed	6	5

In calculating objective function 2, it can be seen that the suggested placement yielded a smaller number of posters to be used, providing for a more optimal solution compared to the real-life placement in terms of minimising.

## 5.4 Assumptions and Limitations

Firstly, this investigation assumes that situating a poster at a certain amenity will guarantee its visibility. It fails to consider that not all locations are equal due to factors such as the geometry of the amenity floor plans.

Additionally, the investigation also assumes that students will consistently traverse the edges as dictated by school walkways between two amenities. However, students straying from these paths is a very plausible circumstance for most due to a multitude of factors. For example, students could stray from a certain walkway due to weather conditions such as rain, or due to the walkway being overcrowded.

Finally, in section 5.1 when calculating the objective functions, double-counting was not taken into account; it was assumed that once a student had walked past a poster, any subsequent interception of a poster by the same student was to be disregarded. This indicates the failure of the investigation to address that when students interact with a poster multiple times, the memorability factor of the poster and hence student outreach is increased.

## 5.6 Possible Improvements

Possible improvements to the investigation could consider the geometrical layout of each individual location in conjunction to this investigation to improve the visibility factor of the posters. This extension could be evaluated as a variant of the well-known 'Art Gallery Problem' which considers the minimum number of people required to maximise visibility of a polygon shaped room. Additionally, an investigation of arrangement alterations to be made as a response to changing factors such as weather or construction resulting in student detours can be considered.

## 5.5 Conclusion

This extended essay has been able to illustrate the application of graph theory in determining the optimal arrangement of council campaign posters in the St. Joseph Institution school compound to maximise student outreach and minimise poster usage.

The alternative solution suggests that poster should be situated at the amenities that correspond to vertices E, O, I, W and H, as depicted in figure 35.

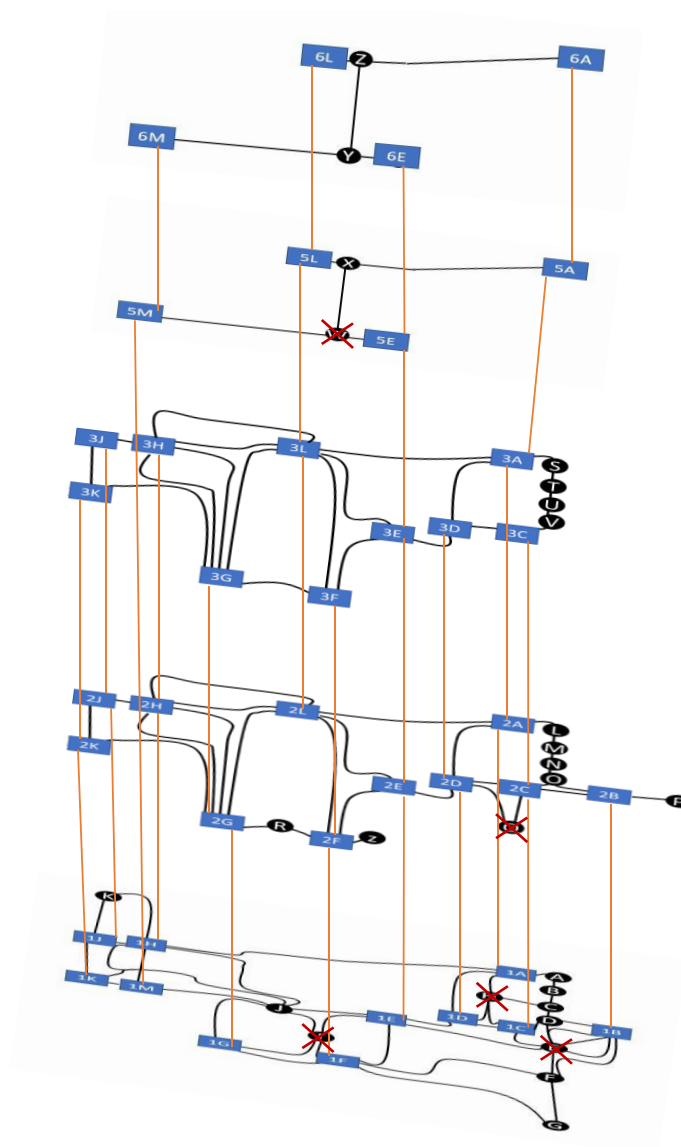


Figure 35: Mathematical Model of St. Joseph Institution Compound with indicated Vertices for Suggested Poster Placement

However, whether this solution can be considered 'optimal' remains up to debate. The use of objective functions, derived from a predetermined set of conditions, indicates that although the suggested arrangement yielded the maximum value for the first objective function, it is impossible to ascertain whether a smaller value can be obtained for the second objective function which evaluates the optimisation criteria of minimising.

Despite this, the investigation continued to hold value via its comparison to the real-life poster arrangement. The objective function calculations determined that the alternative solution provided a more optimal arrangement than the real-life poster placement, yielding a higher value for objective function 1 as well as a lower value for objective function 2 as compared to the Real-Life solution. However, it should be addressed that the result disparities between the Real-Life and suggested solutions could be a result of other factors. For example, the Real-Life arrangement could have considered a different objective such as maximising the number of interceptions as opposed to the case in this essay which considered maximising the number of students intercepted.

To conclude, this extended essay has been largely able to achieve its objectives, obtaining a solution that maximised the quantity of students intercepted by posters and minimising the quantity of posters used.

## 10. Bibliography

- Biswas, S. S., Alam, B., & Doja, M. (2013). Generalization of dijkstra's algorithm for extraction of shortest paths in directed multigraphs. *Journal of Computer Science*, 9(3), 377-382. doi:10.3844/jcssp.2013.377.382
- Borner, G., Burkert, A., Burton, W., Coustenis, A., Dopita, M., Eckart, A., . . . Trimble, V. (n.d.). Brief Review of Mathematical Optimization. Retrieved April 07, 2021, from <https://link.springer.com/content/pdf/bbm%3A978-1-4419-0699-1%2F1.pdf>
- DeepAI. (2019, May 17). Mathematical optimization. Retrieved April 12, 2021, from <https://deepai.org/machine-learning-glossary-and-terms/mathematical-optimization>
- Galvin, D. (2009). Discrete Mathematics, Spring 2009 Graph theory notation. Retrieved April 11, 2021, from [https://www3.nd.edu/~dgalvin1/60610/60610\\_S09/60610graphnotation.pdf](https://www3.nd.edu/~dgalvin1/60610/60610_S09/60610graphnotation.pdf)
- Guichard, D. (n.d.). 5.11 directed graphs. Retrieved April 11, 2021, from [https://www.whitman.edu/mathematics/cgt\\_online/book/section05.11.html](https://www.whitman.edu/mathematics/cgt_online/book/section05.11.html)
- Him0000. (2020, May 21). Mathematics: Walks, TRAILS, PATHS, cycles and circuits in graph. Retrieved April 11, 2021, from <https://www.geeksforgeeks.org/mathematics-walks-trails-paths-cycles-and-circuits-in-graph/>
- Joshi, V. (2017). Finding The Shortest Path, With A Little Help From Dijkstra. Retrieved April 12, 2021, from <https://medium.com/basecs/finding-the-shortest-path-with-a-little-help-from-dijkstra-613149fbdc8e>
- ScienceDirect. (2019). Greedy algorithm. Retrieved April 12, 2021, from <https://www.sciencedirect.com/topics/engineering/greedy-algorithm>
- AnyLogic Simulation Software. (n.d.). Retrieved April 11, 2021, from [https://help.anylogic.com/index.jsp?topic=%2Fcom.xj.anylogic.help%2Fhtml%2Foptimization%2FDefining the Objective.html](https://help.anylogic.com/index.jsp?topic=%2Fcom.xj.anylogic.help%2Fhtml%2Foptimization%2FDefining%20the%20Objective.html)
- SJI. (2018). 1st Storey Overall Plan [Digital image]. Retrieved April 12, 2021.
- SJI. (2018). 2nd Storey Overall Plan [Digital image]. Retrieved April 12, 2021.
- SJI. (2018). 3rd Storey Overall Plan [Digital image]. Retrieved April 12, 2021.
- SJI. (2018). 5th and 6th Storey Overall Plan [Digital image]. Retrieved April 12, 2021.

Techie Delight. (2021, March 16). Single-source shortest paths – dijkstra's algorithm.  
Retrieved April 12, 2021, from <https://www.techiedelight.com/single-source-shortest-paths-dijkstras-algorithm/>

Weisstein, E. (2021). Vertex degree. Retrieved April 11, 2021, from  
<https://mathworld.wolfram.com/VertexDegree.html>

Weisstein, E. (2021, April 7). Graph. Retrieved April 11, 2021, from  
<https://mathworld.wolfram.com/Graph.html>

# Appendices

## Appendix A: Survey Questions

### Survey on locations frequented in SJI compound during student free periods

Hello fellow Josephians! I am conducting some research for my Mathematics Extended Essay. I would like to obtain some relevant Data from SJI students year 5 and above, so please help me by answering the following questions. Data collected from this survey will only be used for my extended essay and will be kept confidential and anonymous. Thank you so much for taking the time to complete my survey!

\*Required

Which areas/amenities do you usually visit during your free/lunch period(s)? \*

Your answer

## Appendix B: Sample Survey Response

Responses cannot be edited

### Survey on locations frequented in SJI compound during student free periods

Hello fellow Josephians! I am conducting some research for my Mathematics Extended Essay. I would like to obtain some relevant Data from SJI students year 5 and above, so please help me by answering the following questions. Data collected from this survey will only be used for my extended essay and will be kept confidential and anonymous. Thank you so much for taking the time to complete my survey!

\*Required

Which areas/amenities do you usually visit during your free/lunch period(s)? \*

Canteen, Library, James Miller Room



## Appendix C: Edge Weight Measurements

Vertices		Numerical Value
G	F	48.2
G	1F	57.3
F	E	16.6
F	1B	67.0
D	1B	8.4
E	1B(1)	28.2
E	1B(2)	19.0
E	1B(3)	26.4
E	1E	58.9
1E	1F	57.6
E	D	17.2
D	C	9.3
C	B	10.2
B	A	9.1
A	1A	6.2
1C	D	6.0
H	1A	28.8
H	C	11.0
H	1C	25.1
H	1D	25.2
1D	1C	20.1

1D	1E	32.1
1A	1D	58.2
E	1C	36.8
F	1F	59.4
1E	J	31.2
1E	I	50.1
1F	I	9.2
I	J	43.1
1F	1G	28.2
1G	J	61.2
1H	1J	5.1
1H	J(1)	99.4
1H	J(2)	95.4
1J	1K	29.6
1K	1M	5.1
1M	J	67.7
1M	1H	29.7
1H	K	16.2
1J	K	16.6
1A	1H	182.3
P	2B	15.4
2B	O	8.3
2B	2C	24.2

O	2C	6.4
O	N	9.4
N	M	9.5
M	L	9.4
L	2A	6.3
2A	2D	57.7
2A	2L	73.4
2D	2L	72.4
2C	Q	19.6
2D	2C	19.9
2D	Q	32.3
2D	2E	31.6
2E	2F	59.4
2F	z	3.8
2F	R	14.5
R	2G	9.3
2E	2L	24.9
2E	2G	83.8
2G	2K	122.7
2E	2K	73.7
2E	2H	98.4
2H	2J	5.7
2K	2J	25.3

2H	2L	108.6
2H	2K	33.3
2H	2G	45.3
2L	2F	102.6
2G	2L	41.0
3A	S	6.4
S	T	9.4
T	U	9.5
U	V	9.4
v	3C	6.4
3A	3L	73.4
3C	3D	19.9
3A	3D	67.6
3D	3L	72.4
3D	3E	31.5
3E	3F	59.4
3L	3F	102.6
3E	3L	24.9
3F	3G	24.8
3G	3L	41.0
3G	3K	122.9
3K	3H	33.3
3G	3H	45.3

3E	3H	98.6
3H	3J	5.3
3H	3L	108.6
3J	3K	25.7
5M	a	8.2
a	b	8.5
b	c	8.0
c	d	8.0
d	e	8.0
e	f	8.1
f	W	10.1
W	5E	8.4
W	X	24.0
X	5L	7.4
X	g	13.7
g	h	10.3
h	i	7.0
i	j	7.0
j	k	7.1
k	l	7.1
l	5A	12.8
6M	m	8.2
m	n	8.5

n	o	8.0
o	p	8.0
p	q	8.0
q	r	8.1
r	Y	10.1
Y	6E	7.4
Y	Z	24.1
Z	6L	7.1
Z	s	13.7
s	t	10.3
t	u	7.9
u	v	7.0
v	w	7.1
w	x	7.1
x	6A	12.8
1A	2A	4.9
2A	3A	5.1
3A	5A	14.6
5A	6A	7.3
1B	2B	4.9
1C	2C	4.9
2C	3C	5.1
1D	2D	4.9

2D	3D	5.1
1E	2E	4.9
2E	3E	5.1
3E	5E	14.6
5E	6E	7.3
1F	2F	4.9
2F	3F	5.1
1G	2G	4.9
2G	3G	5.1
1M	5M	24.7
5M	6M	7.3
1H	2H	4.9
2H	3H	5.1
1J	2J	4.9
2J	3J	5.1
1K	2K	4.9
2K	3K	5.1
2L	3L	5.1
3L	5L	14.6
5L	6L	7.3

## Appendix D: Dijkstra Python Code

```
from collections import defaultdict, deque

class Graph(object):
    def __init__(self):
        self.nodes = set()
        self.edges = defaultdict(list)
        self.distances = {}

    def add_node(self, value):
        self.nodes.add(value)

    def add_edge(self, from_node, to_node, distance):
        self.edges[from_node].append(to_node)
        self.edges[to_node].append(from_node)
        self.distances[(from_node, to_node)] = distance
        self.distances[(to_node, from_node)] = distance

def dijkstra(graph, initial):
    visited = {initial: 0}
    path = {}

    nodes = set(graph.nodes)

    while nodes:
        min_node = None
        for node in nodes:
            if node in visited:
                if min_node is None:
                    min_node = node
                elif visited[node] < visited[min_node]:
                    min_node = node
        if min_node is None:
            break

        nodes.remove(min_node)
        current_weight = visited[min_node]

        for edge in graph.edges[min_node]:
            try:
                weight = current_weight + graph.distances[(min_node, edge)]
            except:
                continue
            if edge not in visited or weight < visited[edge]:
                visited[edge] = weight
                path[edge] = min_node

    return visited, path

def shortest_path(graph, origin, destination):
    visited, paths = dijkstra(graph, origin)
    full_path = deque()
    _destination = paths[destination]

    while _destination != origin:
```



```

        full_path.appendleft(_destination)
        _destination = paths[_destination]

    full_path.appendleft(origin)
    full_path.append(destination)

    return visited[destination], list(full_path)

if __name__ == '__main__':
    graph = Graph()

    for node in ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H',
'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P', 'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z', 'z',
'1A', '1B', '1C', '1D', '1E', '1F', '1G', '1H', '1J', '1M', '1K', '2A', '2B', '2C', '2D',
'2E', '2F', '2G', '2L', '2H', '2J', '2K', '3A', '3D', '3C', '3E', '3F', '3G', '3L', '3H',
'3J', '3K', '5M', '5E', '5L', '5A', '6M', '6E', '6L', '6A', 'a', 'b', 'c', 'd',
'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's',
't', 'u', 'v', 'w', 'x', 'z']:
        graph.add_node(node)

    graph.add_edge('G', 'F', 48.13)
    graph.add_edge('G', '1F', 56.94)
    graph.add_edge('F', 'E', 15.95)
    graph.add_edge('E', '1D', 25.54)
    graph.add_edge('F', '1B', 66.53)
    graph.add_edge('D', '1B', 8.45)
    graph.add_edge('E', '1B', 19.12)
    graph.add_edge('E', '1E', 46.74)
    graph.add_edge('1E', '1F', 57.97)
    graph.add_edge('E', 'D', 17.04)
    graph.add_edge('D', 'C', 9.45)
    graph.add_edge('C', 'B', 9.50)
    graph.add_edge('B', 'A', 9.46)
    graph.add_edge('A', '1A', 6.34)
    graph.add_edge('1C', 'D', 6.46)
    graph.add_edge('H', '1A', 29.05)
    graph.add_edge('H', 'C', 11.31)
    graph.add_edge('H', '1C', 25.05)
    graph.add_edge('H', '1D', 25.39)
    graph.add_edge('1D', '1C', 19.99)
    graph.add_edge('1D', '1E', 31.65)
    graph.add_edge('1A', '1D', 57.67)
    graph.add_edge('E', '1C', 36.96)
    graph.add_edge('F', '1F', 59.04)
    graph.add_edge('1E', 'J', 30.57)
    graph.add_edge('1E', '1D', 50.03)
    graph.add_edge('1F', 'I', 9.41)
    graph.add_edge('I', 'J', 42.89)
    graph.add_edge('1F', '1G', 27.80)
    graph.add_edge('1G', 'J', 61.45)
    graph.add_edge('J', '1K', 73.27)
    graph.add_edge('J', '1J', 82.77)
    graph.add_edge('1H', '1J', 5.37)
    graph.add_edge('1H', 'J', 94.57)
    graph.add_edge('1H', '1K', 33.34)
    graph.add_edge('1J', '1K', 30.24)
    graph.add_edge('1K', '1M', 5.38)
    graph.add_edge('1M', 'J', 67.90)
    graph.add_edge('1M', '1H', 30.24)
    graph.add_edge('1H', 'K', 16.33)
    graph.add_edge('1J', 'K', 27.05)

```

```

graph.add_edge('1A', '1H', 182.12)
graph.add_edge('P', '2B', 15.41)
graph.add_edge('2B', 'O', 8.36)
graph.add_edge('2B', '2C', 24.27)
graph.add_edge('O', '2C', 6.46)
graph.add_edge('O', 'N', 9.45)
graph.add_edge('N', 'M', 9.50)
graph.add_edge('M', 'L', 9.46)
graph.add_edge('L', '2A', 6.34)
graph.add_edge('2A', '2D', 57.67)
graph.add_edge('2A', '2L', 73.34)
graph.add_edge('2D', '2L', 72.34)
graph.add_edge('2C', 'Q', 19.60)
graph.add_edge('2D', '2C', 19.99)
graph.add_edge('2D', 'Q', 32.38)
graph.add_edge('2D', '2E', 31.65)
graph.add_edge('2E', '2F', 59.44)
graph.add_edge('2F', 'z', 3.81)
graph.add_edge('2F', 'R', 14.65)
graph.add_edge('R', '2G', 9.63)
graph.add_edge('2E', '2L', 24.89)
graph.add_edge('2E', '2G', 83.88)
graph.add_edge('2G', '2K', 122.97)
graph.add_edge('2E', '2K', 73.27)
graph.add_edge('2E', '2H', 98.64)
graph.add_edge('2H', '2J', 5.37)
graph.add_edge('2K', '2J', 25.37)
graph.add_edge('2H', '2L', 108.76)
graph.add_edge('2H', '2K', 33.34)
graph.add_edge('2H', '2G', 45.34)
graph.add_edge('2L', '2F', 102.26)
graph.add_edge('2G', '2L', 41.90)
graph.add_edge('3A', 'S', 6.34)
graph.add_edge('S', 'T', 9.45)
graph.add_edge('T', 'U', 9.50)
graph.add_edge('U', 'V', 9.46)
graph.add_edge('V', '3C', 6.46)
graph.add_edge('3A', '3L', 73.34)
graph.add_edge('3C', '3D', 19.99)
graph.add_edge('3A', '3D', 67.66)
graph.add_edge('3D', '3L', 72.74)
graph.add_edge('3D', '3E', 31.65)
graph.add_edge('3E', '3F', 59.44)
graph.add_edge('3L', '3F', 102.26)
graph.add_edge('3E', '3L', 24.28)
graph.add_edge('3F', '3G', 41.90)
graph.add_edge('3G', '3L', 122.97)
graph.add_edge('3G', '3K', 33.34)
graph.add_edge('3K', '3H', 45.34)
graph.add_edge('3G', '3H', 98.64)
graph.add_edge('3E', '3H', 5.37)
graph.add_edge('3H', '3J', 108.76)
graph.add_edge('3H', '3L', 25.37)
graph.add_edge('3J', '3K', 8.02)
graph.add_edge('5M', 'a', 8.05)
graph.add_edge('a', 'b', 8.05)
graph.add_edge('b', 'c', 8.05)
graph.add_edge('c', 'd', 8.05)
graph.add_edge('d', 'e', 8.06)
graph.add_edge('e', 'f', 10.31)
graph.add_edge('f', 'W', 8.04)

```

```

graph.add_edge('W', '5E', 24.10)
graph.add_edge('W', 'X', 7.14)
graph.add_edge('X', '5L', 13.71)
graph.add_edge('X', 'g', 10.31)
graph.add_edge('g', 'h', 7.09)
graph.add_edge('h', 'i', 7.09)
graph.add_edge('i', 'j', 7.10)
graph.add_edge('j', 'k', 7.10)
graph.add_edge('k', 'l', 7.10)
graph.add_edge('l', '5A', 12.58)
graph.add_edge('6M', 'm', 8.02)
graph.add_edge('m', 'n', 8.05)
graph.add_edge('n', 'o', 8.05)
graph.add_edge('o', 'p', 8.06)
graph.add_edge('p', 'q', 8.04)
graph.add_edge('q', 'r', 8.10)
graph.add_edge('r', 'Y', 10.31)
graph.add_edge('Y', '6E', 7.43)
graph.add_edge('Y', 'Z', 24.10)
graph.add_edge('Z', '6L', 7.14)
graph.add_edge('Z', 's', 13.71)
graph.add_edge('s', 't', 10.31)
graph.add_edge('t', 'u', 7.09)
graph.add_edge('u', 'v', 7.08)
graph.add_edge('v', 'w', 7.10)
graph.add_edge('w', 'x', 7.10)
graph.add_edge('x', '6A', 12.58)
graph.add_edge('1A', '2A', 4.98)
graph.add_edge('2A', '3A', 5.12)
graph.add_edge('3A', '5A', 14.66)
graph.add_edge('5A', '6A', 7.35)
graph.add_edge('1B', '2B', 4.98)
graph.add_edge('1C', '2C', 4.98)
graph.add_edge('2C', '3C', 5.12)
graph.add_edge('1D', '2D', 4.98)
graph.add_edge('2D', '3D', 5.12)
graph.add_edge('1E', '2E', 4.98)
graph.add_edge('2E', '3E', 5.12)
graph.add_edge('3E', '5E', 14.66)
graph.add_edge('5E', '6E', 7.35)
graph.add_edge('1F', '2F', 4.98)
graph.add_edge('2F', '3F', 5.12)
graph.add_edge('1G', '2G', 4.98)
graph.add_edge('2G', '3G', 5.12)
graph.add_edge('1M', '5M', 24.76)
graph.add_edge('5M', '6M', 7.35)
graph.add_edge('1H', '2H', 4.98)
graph.add_edge('2H', '3H', 5.12)
graph.add_edge('1J', '2J', 4.98)
graph.add_edge('2J', '3J', 5.12)
graph.add_edge('1K', '2K', 4.98)
graph.add_edge('2K', '3K', 5.12)
graph.add_edge('2L', '3L', 35.12)
graph.add_edge('3L', '5L', 14.66)
graph.add_edge('5L', '6L', 7.35)
print(shortest_path(graph, 'G', 'F'))

```