



University
of Glasgow | School of
Computing Science

Honours Individual Project Dissertation

VISUALIZATION OF CLASSICAL GRAPH THEORY PROBLEMS

Fatma Al-Sayegh
20 November 2022

Abstract

Whereas, formalism in mathematics gives the subject a structure and a level of abstraction which makes it applicable to the most general of the situations. Visualization of a concept on the other hand, restricts the theory to a particular example but it makes one to see a problem in a more concrete pattern. It may also allow the learner to see the same problem in a new light. The learner can then apply or extrapolate the learning to other instances of the problem in general. In this project we have tried to elucidate some classical problems in Graph Theory by the way of visualization on a Web application. The method of visualization are animations and user interaction with animations. Such methods, it is belived can help young students and self-learners to get the first brush with the subject of Graph theory.

Education Use Consent

I hereby grant my permission for this project to be stored, distributed and shown to other University of Glasgow students and staff for educational purposes. **Please note that you are under no obligation to sign this declaration, but doing so would help future students.**

Signature: Fatma Al-Sayegh Date: 20 November 2022

Contents

1	Introduction	1
2	Background	2
3	Analysis/Requirements	3
3.1	Scope of The Project	3
3.2	Criteria for Selection of Problems	3
3.3	Methods of Elucidation	3
3.3.1	Animation	4
3.3.2	User Interaction	4
3.4	Technical Scope of The Project	4
3.5	Requirements	4
4	Design	6
4.1	Wire-frame and Navigation	6
4.2	Animation Panel	6
4.2.1	Visual representation of graphs	6
4.3	Explanation Panel	7
4.4	User Stories for Elucidation of Topics	7
4.4.1	Graph Isomorphism	7
4.4.2	Max k Cut	8
4.4.3	Graph Coloring	9
4.4.4	Minimum Vertex Cover	9
5	Implementation	10
6	Evaluation	11
7	Conclusion	12
	Appendices	13
A	Appendices	13
	Bibliography	14

1 | Introduction

2 | Background

3 | Analysis/Requirements

In this chapter, the scope of the project, the criterion of selection of the problems in graph theory, the thinking behind choosing the methods of elucidation of the selected topics will be discussed. Finally to conclude the analysis the requirements of the project are stated.

3.1 Scope of The Project

Understanding a problem is a necessary first step in trying to solve it. It also enables a student to abstract out a mathematical problem from a real life scenario present in the fields of science and engineering.

This idea has guided this project to be restricted to one which helps a learner to understand the problem itself. The solution, on the other hand, or suggesting an algorithm to solve the problem, if required, is the second important step which has been deliberately not touched upon to keep the scope of this project clear, precise and limited.

3.2 Criteria for Selection of Problems

One of the most important criteria for selection of the problems for the project was based upon the importance of the topic in the field of graph theory. There are several text books in graph theory, which discuss various theorems and problems in the subject. There are a few problems which occur commonly and frequently in them. The order of their inclusion in the text books is based logically. Building the concepts from the basics to advanced. Therefore the problems included in the project should represent all hardness levels.

Since imagining a graph theory problem is largely a visual exercise, there was no dearth of problems which could offer themselves as a subject of an interesting visualization. The additional criteria therefore for filtering the candidate problems was based on whether they could be elucidated in the form of a simple and meaningful example, employed for animation and user interaction. The simplicity of the example doesn't in anyway imply triviality of the problem. Indeed here the assumption is that a simple example problem can make a student reach to the heart of the concept in it's generality fairly quickly. From there on she can extrapolate the learning to more complicated examples.

It is important to mention here that a survey among peers in the field of software engineering and computer science was conducted to gather suggestions on the shortlisted topics and methods. The data from the survey had a role in determining topics and the methods chosen.

3.3 Methods of Elucidation

As it has been discussed in the previous section that feasible methods of elucidation/exposition played a prominent role in the selection of the problems in the first place. These methods can be broadly classified as animations and user interactions or a combination of both.

3.3.1 Animation

The visual medium has dimensions such as of color, position, shape and motion. Generating them with a computer program, scenes can be created attracting the users attention towards a particular aspect of it with the help of animations. The scene in this project consists of a graph, which undergoes transformations of color, shape and position to make a certain aspect of it emphasized for the purpose of explaining a concept. Animations are used in this project to make problems like Graph Isomorphism, Max Cut, and Tree Width. For instance, the example problem of Graph Isomorphism, was explained by morphing a graph to change its shape to acquire another radically different shape. It will be explained later how it can be considered as a visual proof that the two graphs in the scene are thus isomorphic.

3.3.2 User Interaction

Although animations go a long way in terms of having a user's mind involved in the learning process they only offer a linear narrative. On the other hand, user interaction with an animation not only makes the experience more immersive, it also leads to a natural multiplicity of stories from a single program. This happens as a human input results in a novel path from one state to another in the program just like a video game.

3.4 Technical Scope of The Project

To achieve the above mentioned features The program has data structures to hold graphs and algorithms to visually and geometrically manipulate them.

It's important to note here the program does not contain algorithms to solve the listed problems. As the scope of the program was limited to the purposes of visualization and not coding the algorithms which can solve instances of the mentioned problems. Therefore in this project, although the data type of Graphs (Set of Vertices and Set of Edges) and the associated functions, are quite general and can support operations of various kinds, care is taken that we provide the solution to the visualization program before hand to give enough information to the various animations and user interactions.

3.5 Requirements

Based on synthesis of the above sections it is concluded that there should be a web application written for the desktop browser with user interactive animations which elucidate the following graph theory problems -

1. Graph Isomorphism
2. Max Cut
3. Graph Coloring
4. Minimum Vertex Cover
5. Tree Width

And to achieve all this by programming the following items -

1. Data structures which represent vertices, edges and graphs.
2. Display the above entities as Scalar Vector Graphics on screen.
3. Translation and shape transformation of the graphs for animation.
4. Generation and handling of events triggered by user interaction with the elements in the animation for user interaction.

5. Display appropriate text in synchronization with the animations and user interaction.

Doing so in a manner which fulfills the following subjective qualities –

1. Substantive learning impact
2. Ease of Use
3. Coherent Story telling
4. Pleasing Aesthetics

And finally, evaluating the application with the help of the peers on parameters which can be broadly classified into the following categories –

1. Quality of Elucidation
2. User Experience
3. Learning Impact

4 | Design

This chapter gives the details of how the requirements of the project, are converted to specific design specifications.

4.1 Wire-frame and Navigation

The web application is a **Single Page Application**. Where the navigation from one topic to another occurs according to the user inputs. When the state moves from one graph theory to another topic, the data on the screen, that is the graphics and the text, change on the same page without changing the url.

The page is vertical divided into two parts, the left part of the page contains an instance of an animation, the right contains explanation of the topic and advice on how to interpret the animation along with navigation and control buttons.

The text in the explanation part of the page is dynamic in nature, if the animation has facility of user interaction with it's elements, the corresponding text on the right responds with advises on the state of affairs and what the user should do next.

There is a navigation bar at the bottom of the page, with left and right arrows, along with the names of the previous and the next topics, to hop from one topic to another.

4.2 Animation Panel

As mentioned in the preceding section, the left half of the page is for graphics, which contains an animation, a user-interaction session or a combination of both. The animation contains one or more than one graphs. These graphs undergo, according to needs of the topic in consideration transformations of appearance and annotation.

4.2.1 Visual representation of graphs

The graphs are represented as vertices and edges joining those vertices. Although the geometrical placement of the vertices is of no consequence in the subject of graph theory, for the purpose of visualization, vertices are assigned a 2 dimensional position. The edges, don't contain attributes such as length or positions of endpoints of a line segment, they are rather defined as a relation between a pair of vertices.

Appearance of Vertices The vertices of graphs in the animations are color filled circles of the same size save for certain exceptions. The color of the vertices have been allotted by varying mostly the hue and keeping saturation and lightness relatively same in the **HSL** (Hue Saturation Lightness) color space. The vertices contain a name inside the circle as an integer. The names of vertices were chosen as integers as it was assumed that such a representation would help the developer and also the user to keep a track of them as order of integers is understood better by humans.

When a particular vertex is needed to be shown differently than the rest then its size and color are displayed differently. For example, a user-selected vertex in some of the animations are shown bigger and its color changed to golden. The gold color it was observed makes the vertex in consideration stand out differently from other colors in the animation.

Appearance of Edges Edges, although are defined very algebraically as a relationship between a pair of vertices it gets drawn by referring to the positions of the related vertices as a straight line segment white in color.

When an edge is supposed to shown differently than the rest of the edges, its width is increased and color changed to the same value of golden as a selected vertex. Again it was found that this color and thickness, made the edge standout from the rest of the edges and helped in showing it distinguished without being unpleasantly distracting.

4.3 Explanation Panel

Whereas, the animation panel on the left page contains all the graphical components of the topic elucidation. The right half of the page is occupied by the Explanation Panel. It contains the title of the problem being explained, its definition and instruction on how to go ahead with starting the animation or user-interactive tasks. It also contains control buttons for starting, re-starting and pausing animations. For user-interactive tasks it has buttons to reset the task.

As the animation or a user-interactive task progresses the explanation panel generates explanatory and instructive text.

4.4 User Stories for Elucidation of Topics

This section describes, what the user is intended to experience while interacting with the individual topics in the application with a special consideration towards learning impact and understandability.

When the user opens the web application on a browser of her choice, the first topic she sees is that of Graph Isomorphism. She may stay there to interact with the example of the topic or navigate to other topics using navigation buttons at the bottom to have a bird's eye view of other topics.

Each graph theory problem has its own character and require a different approach for elucidation. The following sections will explain these approaches with their intended experience on the user with learning outcomes which may be achieved.

4.4.1 Graph Isomorphism

The user is presented with a graph on the left and textual explanation on the right of the page. The textual explanation portion of the page also has some media buttons such as play/pause and reset to interact with the explanation. The text explanation briefly defines graph isomorphism and advises the user to press the play button.

Animation: When the user presses the play button, a new graph emerges out of the old one while the keeping the edges between any two vertices conserved. While keeping the connectivity between the vertices intact, the graph transforms into a completely new shape, almost giving a visual proof that the two graphs on the screen are isomorphic to each other.

User Interaction: After the two isomorphic graphs have separated from each other, the user is advised in a text panel to choose a vertex by either hovering over a vertex or pressing the corresponding number on the keyboard. Doing so, will change the visual appearance of the selected vertex, the edges incident on the selected vertex and the adjacent vertices to the selected

vertex in both graphs. The selected vertex will be enlarged to a new radius and change its color from its original color to golden color making it stand apart from the rest of the vertices. The edges incident on the vertex will change their colors to the same gold color. The adjacent vertices to the selected vertex will form a golden halo around them. This color transformation will distinguish a kind of a subset in the two displayed graphs. At this point of time, in the text panel the user is pointed out that the selected vertex has the same number of edges connecting to the same adjacent vertices in both the graphs. The user is also advised to inspect other vertices of the graphs and convince herself that each vertex of the graph has the same adjacent vertices in both the isomorphic counterparts.

Learning Impact: The transformation of a graph into a radically different looking graph but being essentially the same as far as the connectivity between the vertices go acts as a visual proof that the graphs are isomorphic. While individually inspecting each vertex will re-confirm this idea to the user. After having experienced the concept of graph isomorphism in this way it is assumed that the concept and definition of the term would be clear to her.

4.4.2 Max k Cut

Max Cut has two animations one after the other. The first animation is about Max 2 Cut and the second is about Max 3 Cut. It is assumed that the user will extrapolate the concept of the general Max k Cut after understanding the first two cases ($k = 2$, $k = 3$) and extrapolating it over greater values of k . The examples shown in the animations are a nearly bipartite and a tripartite graph for $k = 2$ and $k = 3$ respectively. A nearly bipartite and a tripartite graph is used to elucidate the topic as it is easier for the user to visualize how the two graphs can be segregated to sets of vertices such that the maximum number of edges pass between such sets.

In both the cases of $k = 2$ and $k = 3$, the weight of all the edges is taken to be equal to 1. This decision has been taken as with $w = 1$, the answers to both the max cut problems are more visual than unequal weights.

Max 2 Cut Animation It starts with an Original graph on the left and definition of Max K Cut and Max 2 Cut on the right of the web page. The right part of the page also contains media buttons to pause and play the animations. It also has a button for switching from Max 2 Cut example, to Max 3 Cut example. The Max 2 Cut animation starts with a graph, which starts when the user presses the play button. As the animation progresses a new graph emerges out of the original one and translates towards right changing its shape to segregate its vertices into two sets forming a Maximum 2 Cut. The two sets move vertically up and down and increase the distance between themselves, revealing the number of edges passing from one set to another which the user can intuitively tell is greater than the number of edges between any other two sets which may have been formed from the vertices of the graph.

The user is advised to put up a pre-defined horizontal line by pressing a button in the explanation panel. The line is drawn between the two sets of vertices. The intersection points between the edges and the max cut line is shown by blue dots. These user is advised to observe the number of intersection points which tell the user the number of edges passing from one set to another.

Max 3 Cut Animation Just like the animation for Max 2 Cut, this animation is started by the user by pressing the play button. The animation on the right starts with a tripartite graph arranged in a circular form. As the animation progresses the graph gets Divided into three sets of vertices in which the three sets translate in directions which are set 120° apart from each other. The graph transforms from a circularly arranged one to a triangular form. The user can draw the Max 3 Cut lines at any point in the progress of the animation. These are three lines separating each set from the rest of the graph, with points of intersection shown in blue showing the number of edges passing from one set to the rest of the sets.

Learning Impact: Although the examples in the Max 2 Cut and Max 3 Cut can be seen as simple ones as the first one was nearly a bipartite graph and the second one was a tripartite graph, they do a good job at defining the problem well. Not just that such visualization explains the problem well it also may act as artwork especially in the case of Max 3 Cut, which may inspire an imaginative student to want to investigate the subject further.

4.4.3 Graph Coloring

Graph Coloring is explained to the user with the help of a user-interactive task. The user is presented with a graph which has all the vertices in color white. On the explanation panel to the right he is given the definition of the problem and advice on how to complete the task.

The task is to choose colors from a color palette of three colors namely red, green and blue, and color vertices in the graph such that no two adjacent vertices have the same color. Whenever the user colors two or more adjacent vertices the same color the text panel warns them to make amends. There is a reset button in the explanation panel to un-color all the vertices to start all over again if the user wants to start from the beginning.

The user is challenged to first challenged to color the graph successfully in just two colors but as it would be soon clear to the user it can only be done in three.

Learning Impact: The aim of the user task is to make the user understand the problem, and retain this in their memory in for a long period of time as user interactive task can create an impression of a problem for a very long period of time than just watching an animation as a spectator.

4.4.4 Minimum Vertex Cover

Minimum Vertex Cover, just by the nature of the problem is chosen to be elucidated by the help of a user-interactive task. The user is given a graph along with explanation of the Minimum Vertex Cover problem. He is also explained how to complete the task. The task is to select vertices successively either by clicking them or pressing a number corresponding to the vertex of choice on the keyboard. When he selects the vertex, the selected vertex and all the edges incident on it are displayed differently. He has to thus highlight all the edges by selecting the minimum number of vertices in the graph. If he has done this task effectively then he would not choose more vertices than required to cover all the edges in the graph. When the user covers all the edges by only selecting four vertices, he is given a congratulatory message for having done the task right. If he covers the graph by selecting more than four vertices then he is advised to do the same in just four.

Learning Impact: Just like Graph coloring, in the case of Minimum Vertex Cover too, it is assumed that a user-interactive task is effective not just in explanation of a topic but also, retention of the concept for a long period of time.

5 | Implementation

6 | Evaluation

7 | Conclusion

A | Appendices

7 | Bibliography