GATutor: Greedy Algorithms Tutor using guided discovery learning approach.

Dissertation Report

Submitted in the partial fulfillment of the requirements

for the degree of

Master of Technology

by

Meenakshi Verma
Roll No. 123050014

under the guidance of

Prof. Sridhar Iyer

Department of Computer Science and Engineering
Indian Institute of Technology Bombay
Mumbai
2014

Acknowledgement

I would like to express my gratitude to my guide Prof. Sridhar Iyer, for his constant motivation, supervision and guidance, during the course of the seminar. His suggestions always helped me go forward in the right direction when the work was stuck.

Meenakshi Verma
Mtech2
Computer Science and Engineering
IIT Bombay

Abstract

Algorithms are a core of Computer Science. While there are established good material and visualizations to learn about Algorithms, it is important to devise ways of active learning of algorithms through visualization. Greedy algorithms are an important class of algorithms, and it is useful to develop an effective teaching strategy for them. We have created a framework for effective teaching of greedy algorithms based on guided discovery learning approach. We implemented the framework as an system named GATutor, incorporating principles of Education Technology, software design issues, and a user friendly interface. We have constructed learning material for four areas where greedy algorithms can be applied to find the optimal solution.

In this report, we have initially discussed about our framework, its design decisions, and how it will be helpful in effective learning. Later we have described the design of the GATutor system, implementation, and challenges faced. Finally we have tried to prove the efficacy of the system by conducting learning, attractiveness and usability evaluation of the system along with future work.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Need and Motivation

Algorithms being core of Computer science have always demanded attention in various ways. Efforts are required to improve their teaching strategy. Earlier they have been taught in schools in a passive way. Researchers and Instructors have always been interested in finding new ways through which teaching can become more interesting. To do so it is require to visualize algorithms and make students play with it. It is also necessary to guide them at each step so that they do not divert from right path. Greedy algorithms are the most common design techniques. Though they are simple but their learning objectives demand tough teaching strategies. Because they are teached in a passive way , most students learn it by rote. There are two broad areas of learning algorithms-one is analysing the existing algorithms and other is design new algorithms themselves. If we go through a textbook,it contains only the description of greedy algrothims and then a set of problems. But each problem actually consist of de-scription, optimal function selection, proof of optimality, coding, and complexity analysis. By making students learn through discovery, they will feel excited about learning algorithms as explained in [3]. To make students understand each of this we need to build a active learning system. Polya [9] remembers the time when he was a student himself: he was always perturbed by the question:Yes the solution seems to work, it appears to be correct; but how is it possible to invent such a solution? How could I invent or discover such things by myself? According to us with some prerequisite knowledge, stimulating questions, providing hints can help a student to bring an algorithm right from scratch.
We feel that with the availability of some prerequisite knowledge, timely hints, and stimulating questions posed by the instructor, one can always encourage students to redesign an algorithm right from scratch.

1.2 Overview of work done

1.2.1 Goal

We need to design a system that encourage the students to redesign an algorithm right from scratch with the availability of some prerequisite knowledge, timely hints, and stimulating questions posed by the instructor.

1.2.2 Theory Involved

We have designed a framework for teaching of greedy algorithms based on guided learning approach. Framework contains learning goals from shallow to deep. Proof of optimalily is also provided with the help of contradicting examples. It is difficult for a teacher to make students learn all these principles in a passive way and to make them practice with a variety of questions. Our system gives answers of various learning objectives at higher levels of blooms taxonomy also as explained in later sections.

1.2.3 Overview of System

This is a joint project work done along with Mukund Lahoti as presented in his thesis [12]. We have implemented the framework as a online visualization system to teach greedy algorithms to prove its usage. We have covered almost all areas in which greedy algorithms design techniques can be applied such as Graph algorithms which include Kruskals, prims, dijsktras, activity scheduling problem, and knapsack problem. We have used basic features of a greedy algorithm as our approach to proceed. Every greedy algorithm basically consist of a satisfying condition and selection function. Selection function can be defined as set of available candidates, returning at each step the most promising candidate with respect to some measure.

System has various modules:

- **Algorithms**
  We have covered various greedy algorithms:
  Minimum spanning tree algorithms such as Prim's, Kruskal's have been covered.Dijkstra's shortest path algorithm, Knapsack problem and Activity Scheduling algorithms are also included in our system.

  Graph algorithms which includes prims, dijkstras has been explained in later chapters of this report. Activity scheduling problem is also a part of this report. Kruskals algorithm and knapsack problem has been explained in Mukunds report [6].

- **Implementation**
  Implementation of web pages is done using java script, css and html.We

have made the pages interactive with the user, so that user will not get bore with the system. jQuery is used for developing quiz pages as it helps to control HTML events, animations, and other interactions on a web page by adding and removing style sheet handlers.

We have used mysql database for storing students information.It stores the log information of the students, his/her sequence of visiting the pages so that we can find out a pattern of user visiting the pages.

- **A real life example**
  A real life example is included with each algorithm learning in a form of game.

- **Modules for proving wrong choice via visualization**
  To remove any misunderstanding, we have also explained why the choice of a particular student is wrong.

- **modules for right choice via visualization**
  An interactive visualization screen has been developed for making students understand their choice.

- **Quiz module**
  We have provided a quiz module at the end of each algorithm for better practice and confidence of students.

- **Login module and student tracking system**
  We have also developed a login module having database which keeps a track on students pattern of accessing our system. With the use of which we can find out the most common misconceptions of the student, we can also find out whether student is actually playing with our system or just fluking.It has been covered in Mukund's report [6].

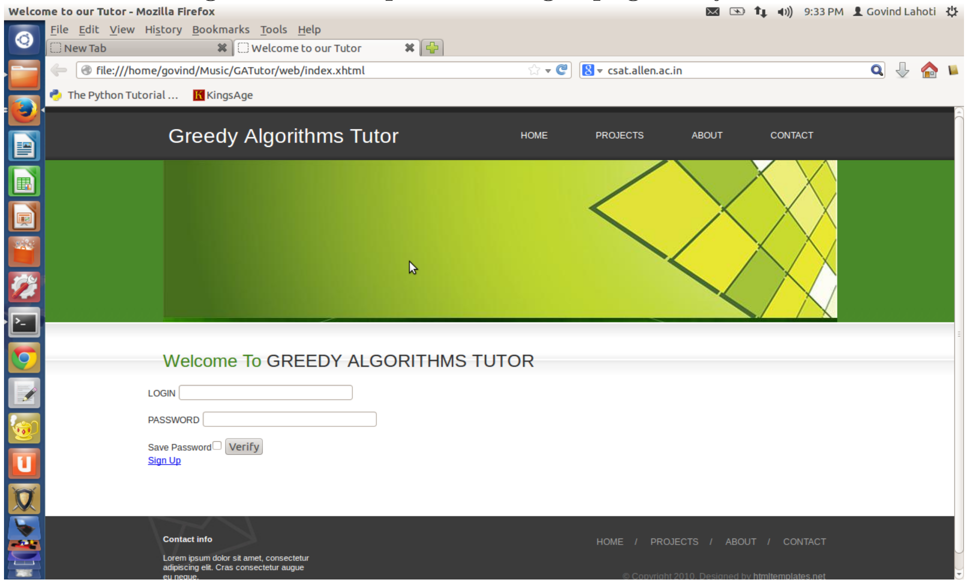
1.2.4 Results

We have conducted a series of experiments on our system.Initially in 1st stage if our system we conducted Pilot experiment on our system to know whether our system is effective or not. We modified our system as per user's feedback and then moved on to more additions of modules.

Then on final completion we conducted testing based on system's usability, learning and attractiveness. Results have been shown in later parts of this report.

1.2.5 Screen-Shots

Figure 1.1: Snapshot of Login page of system



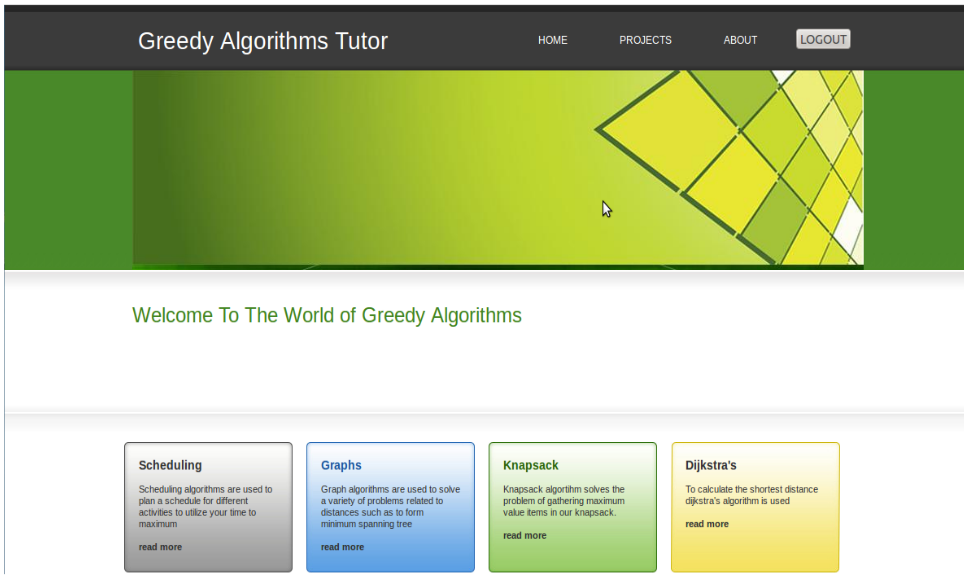
Welcome To GREEDY ALGORITHMS TUTOR

Contact info

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras consectetur augue eu neque.

© Copyright 2010. Designed by htmltemplates.net

Figure 1.2: Snapshot of Index page



Welcome To The World of Greedy Algorithms

Scheduling

Scheduling algorithms are used to plan a schedule for different activities to utilize your time to maximum

read more

Graphs

Graph algorithms are used to solve a variety of problems related to distances such as to form minimum spanning tree

read more

Knapsack

Knapsack algortihm solves the problem of gathering maximum value items in our knapsack.

read more

Dijkstra's

To calculate the shortest distance dijkstra's algorithm is used

read more

1.3 Organization of this report

Chapter 2 compares our system with various other systems already present on different parameters. **Chapter 3** gives the overview of our system. **Chapter 4** explains the detailed design of our System.**Chapter 5** consists of some experimental results. **Chapter 6** presents the future work.

Chapter 2

Related Work

2.1 Introduction

Algorithm tutor has been main focus of our research. Inspite of covering all areas, we have focused on greedy algorithms. There does not exist many systems which mainly focus on greedy algorithms. But there are some such as GREED-EX explained in [4]. Each and every system is master in some but lack other criteria. In the later section we have provided you with the characteristics of the various system which are related to pur area of interest.

2.2 Algo tutor systems

Following are the various algorithm tutor that exist in universe:

- AlgoTutor : AlgoTutor given in [11] is a Visual Algorithm Tracer and Program Pad embedded in it. It is a basic tutor which encourages learning programming through algorithmic design. It teaches students to trace the flow of basic algorithms.It has been tested over a group of student to teach basic programs including while loop, for loop etc.

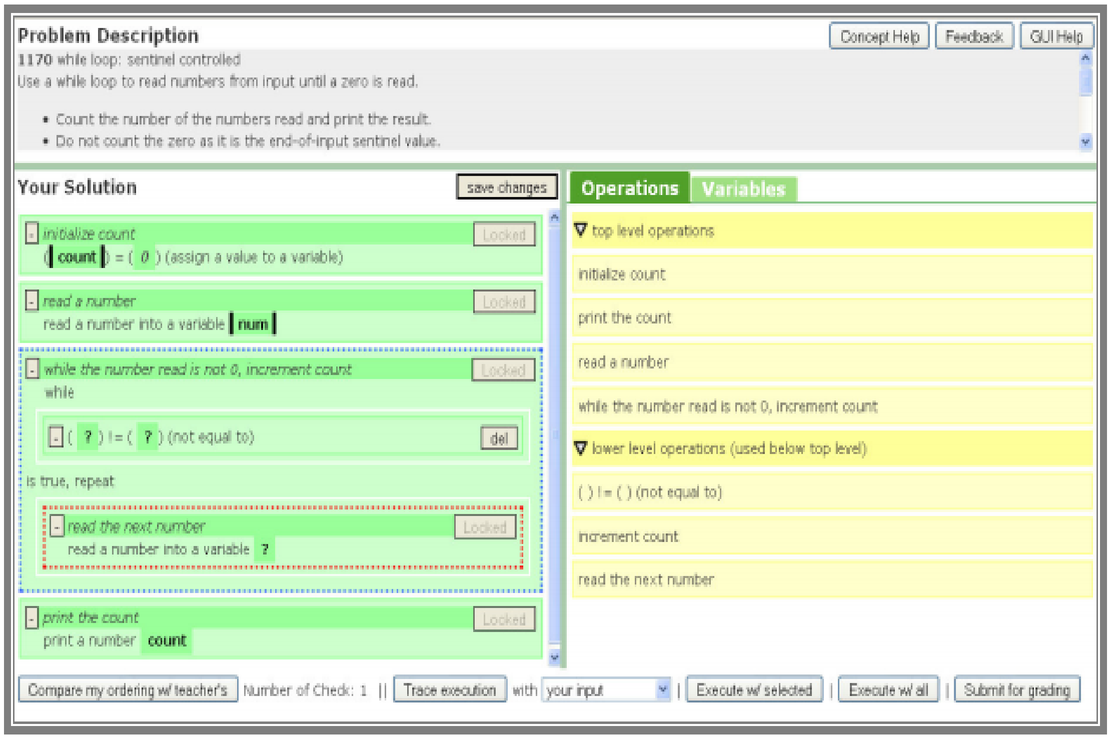
  The results were significant. It stated that teaching programming through algorithmic approach is beneficial rather than just making them learn for and while loops.

  It gives a option of drag and drop through which student can construct a new algorithm using some predefined operation blocks. It also has a program pad which represents the code generated by the student.

  It basically consist of three components as

  1. building the algorithm 2.1
  2. Executing it step wise 2.2
  3. program pad 2.3

Figure 2.1: Algo Tutor [11]



Problem Description

1170 while loop: sentinel controlled

Use a while loop to read numbers from input until a zero is read.

- Count the number of the numbers read and print the result.
- Do not count the zero as it is the end-of-input sentinel value.

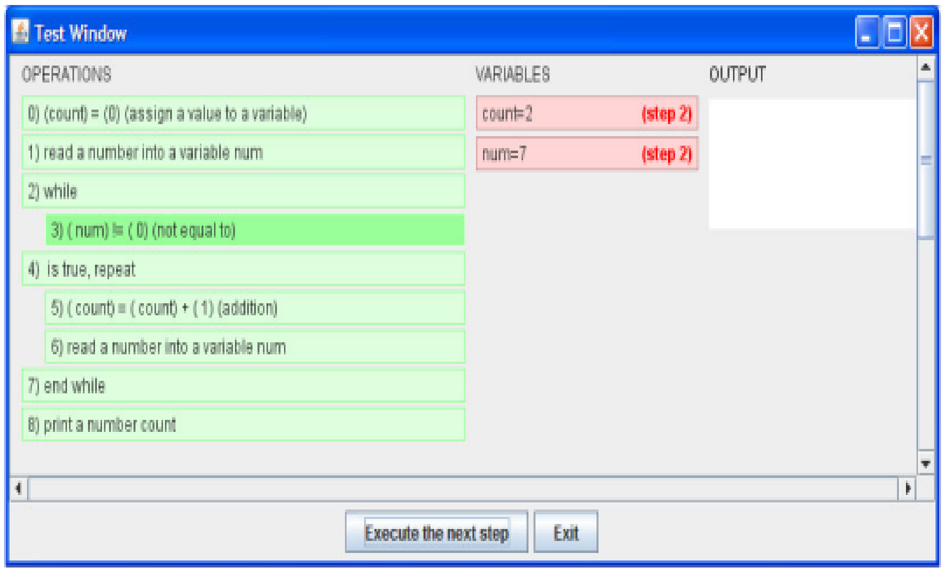
Your Solution

```
initialize count
    ( count ) = ( 0 ) (assign a value to a variable)
read a number
    read a number into a variable num
while the number read is not 0, increment count
    while
        ( ? ) != ( ? ) (not equal to)
    is true, repeat
        read the next number
            read a number into a variable ?
print the count
    print a number count
```

Operations

- top level operations
- initialize count
- print the count
- read a number
- while the number read is not 0, increment count
- lower level operations (used below top level)
- ( ) != ( ) (not equal to)
- increment count
- read the next number

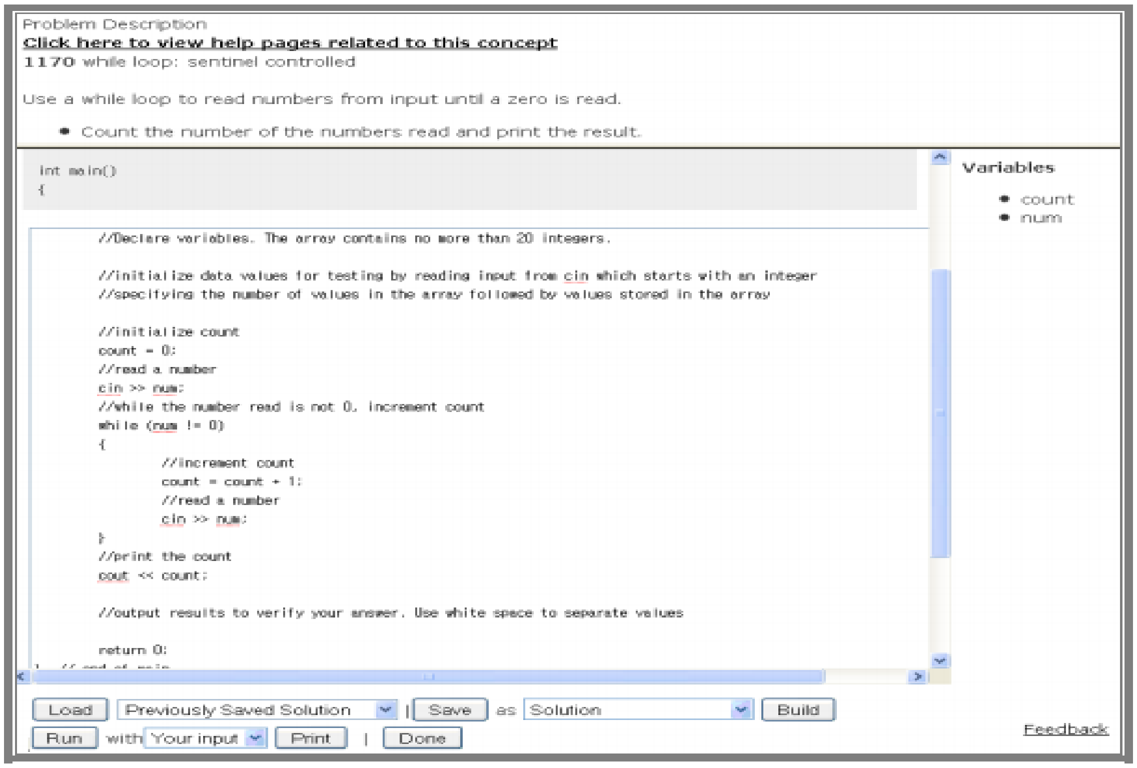
Figure 2.2: Algo Tutor [11]



Test Window

OPERATIONS	VARIABLES	OUTPUT
0) (count) = (0) (assign a value to a variable)	count=2 (step 2)	
1) read a number into a variable num	num=7 (step 2)	
2) while		
3) (num) != (0) (not equal to)		
4) is true, repeat		
5) (count) = (count) + (1) (addition)		
6) read a number into a variable num		
7) end while		
8) print a number count		

Figure 2.3: Algo Tutor [11]



Problem Description

Click here to view help pages related to this concept

1170 while loop: sentinel controlled

Use a while loop to read numbers from input until a zero is read.

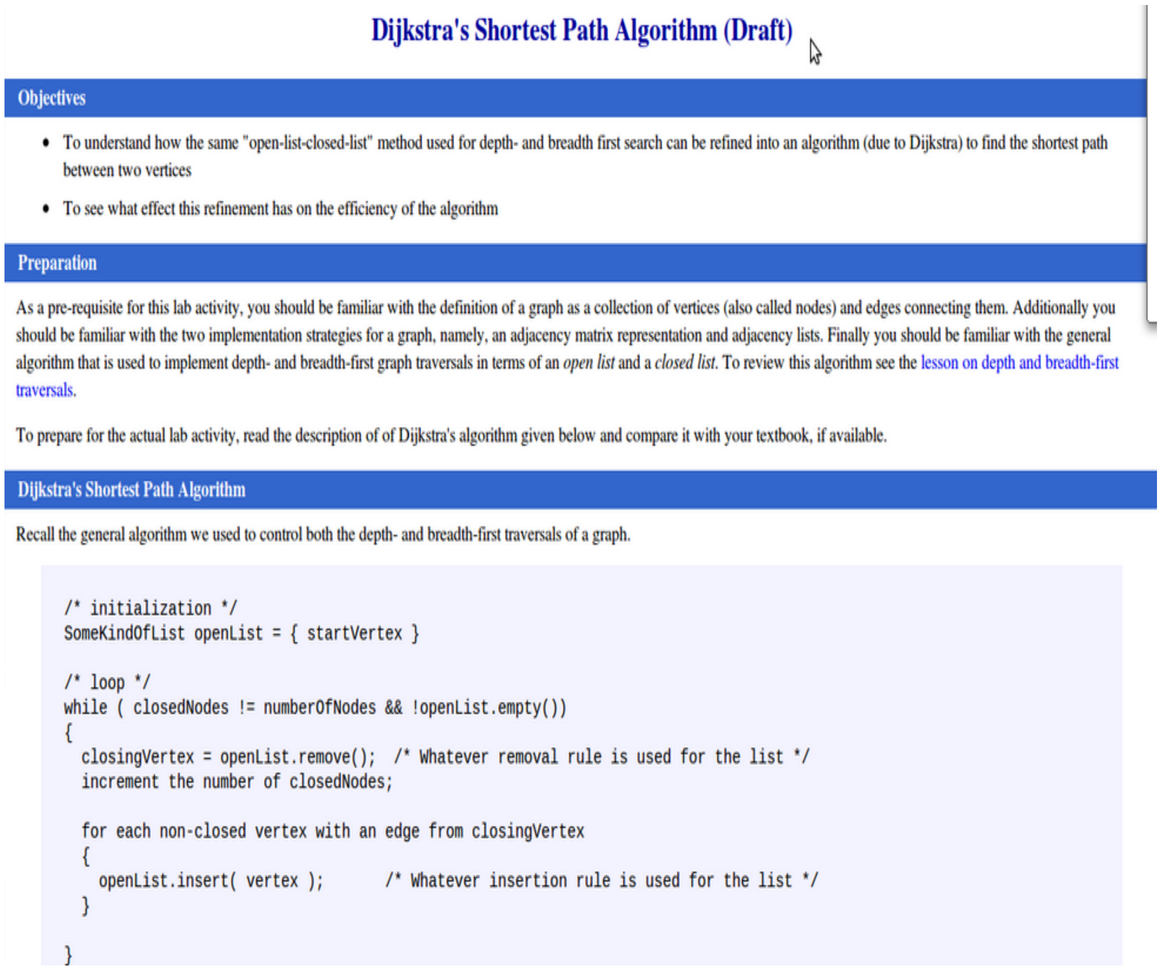
- Count the number of the numbers read and print the result.

```
int main()
{

        //Declare variables. The array contains no more than 20 integers.

        //initialize data values for testing by reading input from cin which starts with an integer
        //specifying the number of values in the array followed by values stored in the array

        //initialize count
        count = 0;
        //read a number
        cin >> num;
        //while the number read is not 0, increment count
        while (num != 0)
        {
                //increment count
                count = count + 1;
                //read a number
                cin >> num;
        }
        //print the count
        cout << count;

        //output results to verify your answer. Use white space to separate values

        return 0;
}   // end of main
```

Variables

- count
- num

Figure 2.4: JHAVE System [8]



Dijkstra's Shortest Path Algorithm (Draft)

Objectives

- To understand how the same "open-list-closed-list" method used for depth- and breadth first search can be refined into an algorithm (due to Dijkstra) to find the shortest path between two vertices
- To see what effect this refinement has on the efficiency of the algorithm

Preparation

As a pre-requisite for this lab activity, you should be familiar with the definition of a graph as a collection of vertices (also called nodes) and edges connecting them. Additionally you should be familiar with the two implementation strategies for a graph, namely, an adjacency matrix representation and adjacency lists. Finally you should be familiar with the general algorithm that is used to implement depth- and breadth-first graph traversals in terms of an *open list* and a *closed list*. To review this algorithm see the lesson on depth and breadth-first traversals.

To prepare for the actual lab activity, read the description of of Dijkstra's algorithm given below and compare it with your textbook, if available.

Dijkstra's Shortest Path Algorithm

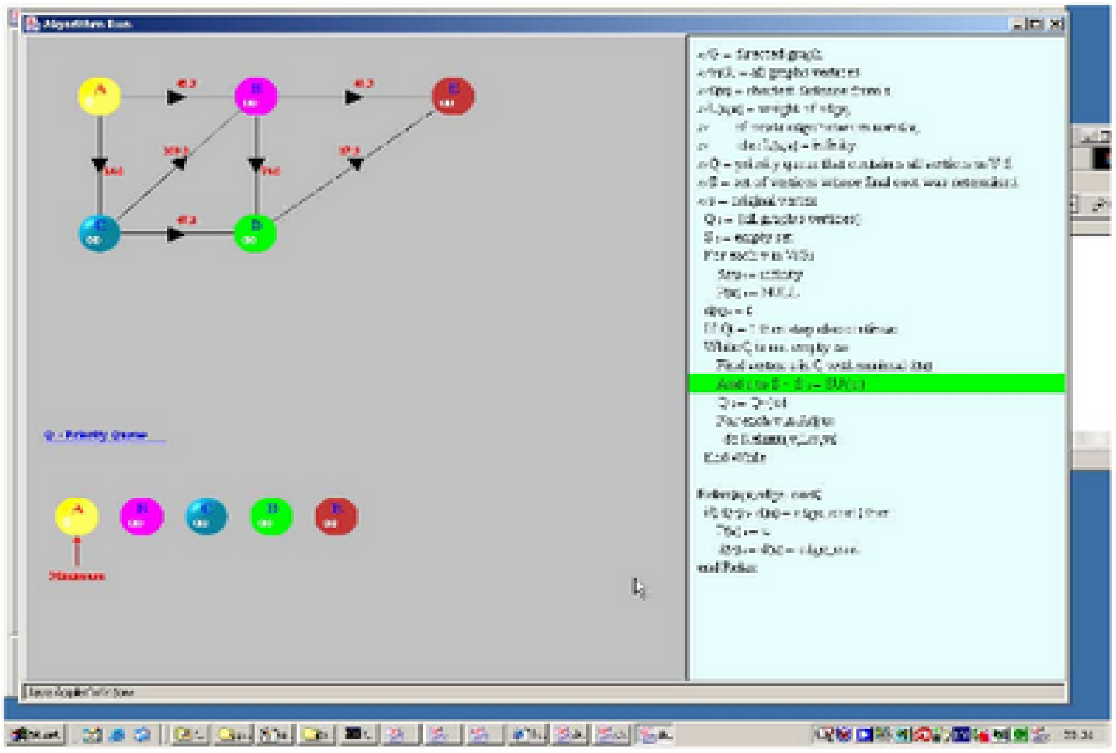
Recall the general algorithm we used to control both the depth- and breadth-first traversals of a graph.

```
/* initialization */
SomeKindOfList openList = { startVertex }

/* loop */
while ( closedNodes != numberOfNodes && !openList.empty())
{
  closingVertex = openList.remove();  /* Whatever removal rule is used for the list */
  increment the number of closedNodes;

  for each non-closed vertex with an edge from closingVertex
  {
    openList.insert( vertex );      /* Whatever insertion rule is used for the list */
  }

}
```

- JHAVE-An algorithm visualization system [8].

  It aims to animate algorithms which concludes that visualization of a algorithm increases its understanding rather than just reading the code of it. They are not interactive with student which makes it a little boring as student can get diverted while working on it. It supports many algorithms including graphs,Sorting,Hashing and miscellaneous.

Figure 2.5: ANIMAL System [10]

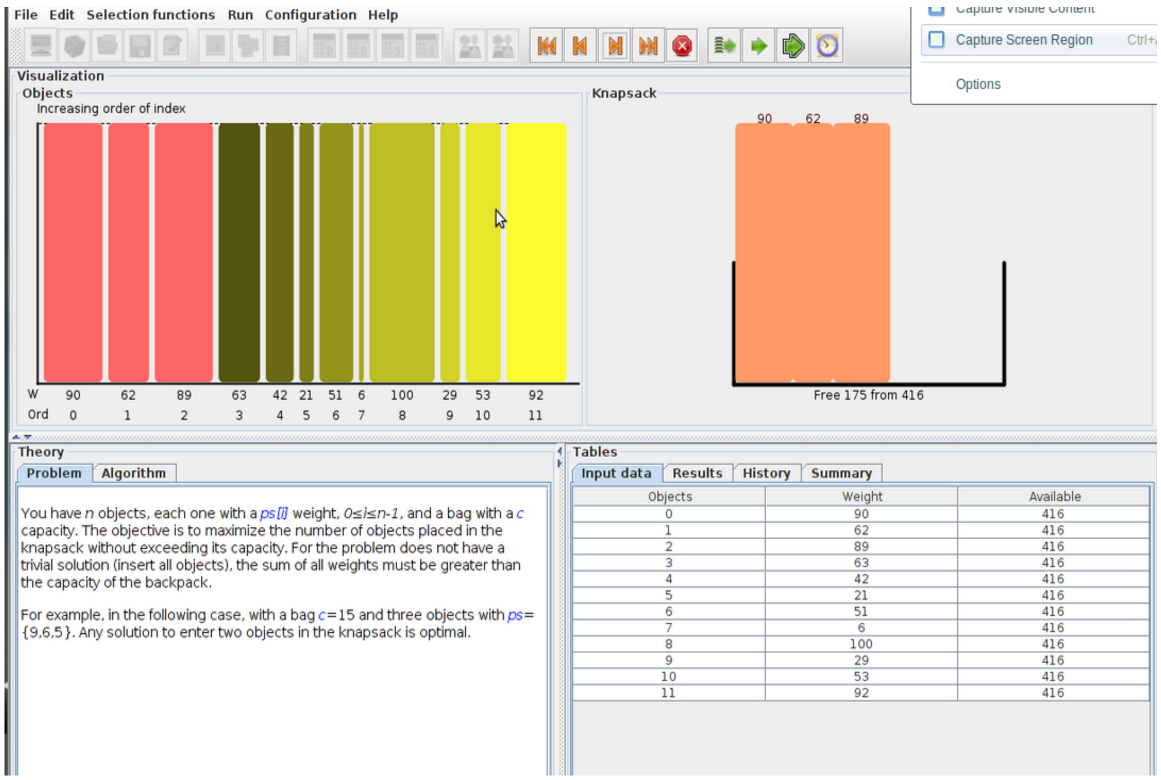


```
//G – directed graph
//V[G] – all graph's vertices
//d[u] – shortest distance from s
//w(u,v) – weight of edge
//      [illegible]
//      [illegible]
//Q – priority queue that contains all vertices in V[G]
//S – set of vertices whose final cost was determined
//s – original vertex
 Q := {all graph's vertices}
 S := empty set
 For each v in V[G]
   d[v] := infinity
   P[v] := NULL
 d[s] := 0
 If Q = {} then stop the algorithm
 While Q is not empty do
   Find vertex u in Q with minimal d[u]
   Add u to S : S := S U {u}
   Q := Q-{u}
   For each v adjacent to u
     RELAX(u,v,w)
 End While

Relax(u,v,edge_cost)
 if d[v] > d[u] + edge_cost then
   P[v] := u
   d[v] := d[u] + edge_cost
end Relax
```

- ANIMAL [10]

  It Uses animation to visualize algorithm with simultaneous code view provided.As the system is not interactive with the user, it make it somewhat bore.It supports Backtracking algorithms, Compression algorithms, Cryptography algorithms, Data structures, Graph algorithms, Graphics algorithms, Hardware-based algorithms, Hashing algorithms, Mathematics which is a good collection. But providing only animations of algorithms cannot seek the attention of a student for a longer and it is also difficult to enhance their learning process and to check whether they have learned it or not exactly.

Figure 2.6: GREED-EX System [12]


File Edit Selection functions Run Configuration Help
Capture Visible Content
Capture Screen Region Ctrl+
Options
Visualization
Objects
Increasing order of index
W 90 62 89 63 42 21 51 6 100 29 53 92
Ord 0 1 2 3 4 5 6 7 8 9 10 11
Knapsack
90 62 89
Free 175 from 416
Theory
Problem
Algorithm
Tables
Input data
Results
History
Summary

You have *n* objects, each one with a *ps[i]* weight, *0≤i≤n-1*, and a bag with a *c* capacity. The objective is to maximize the number of objects placed in the knapsack without exceeding its capacity. For the problem does not have a trivial solution (insert all objects), the sum of all weights must be greater than the capacity of the backpack.

For example, in the following case, with a bag *c*=15 and three objects with *ps*= {9,6,5}. Any solution to enter two objects in the knapsack is optimal.

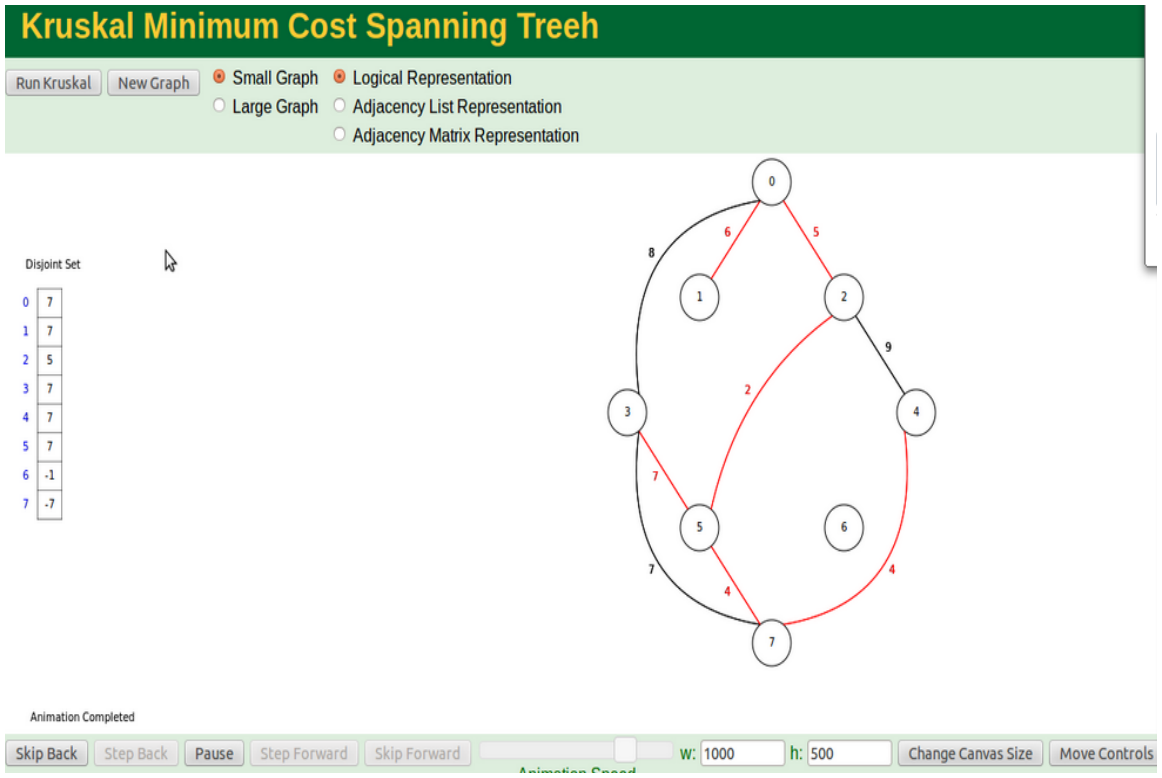
Objects	Weight	Available
0	90	416
1	62	416
2	89	416
3	63	416
4	42	416
5	21	416
6	51	416
7	6	416
8	100	416
9	29	416
10	53	416
11	92	416

- GREED-EX

  It is a algorithm tutor based on discovery learning approach. This system mainly focuses on greedy algorithms. It teaches two greedy algorithms-Acitivity selection problem and Knapsack.They uses a didactic method to teach greedy algorithms [12]. The didactic method asks the student to find out all the selection functions that could characterize an optimal greedy algorithm for a given optimization problem.

  There has been no guided approach followed by them. It might be possible that sometimes student might get mislead if he does get right direction. They have also not given any questions on which student can be judge. Main features of this system are students discover by himself, results table and history is provided to compare different selection functions. They have tested it on computer science majors enrolled at a university in a second-year mandatory course on design and analysis of algorithms, Results were better in experiment group in post test rather in pre-test.

Figure 2.7: Galles Visualization System [2]

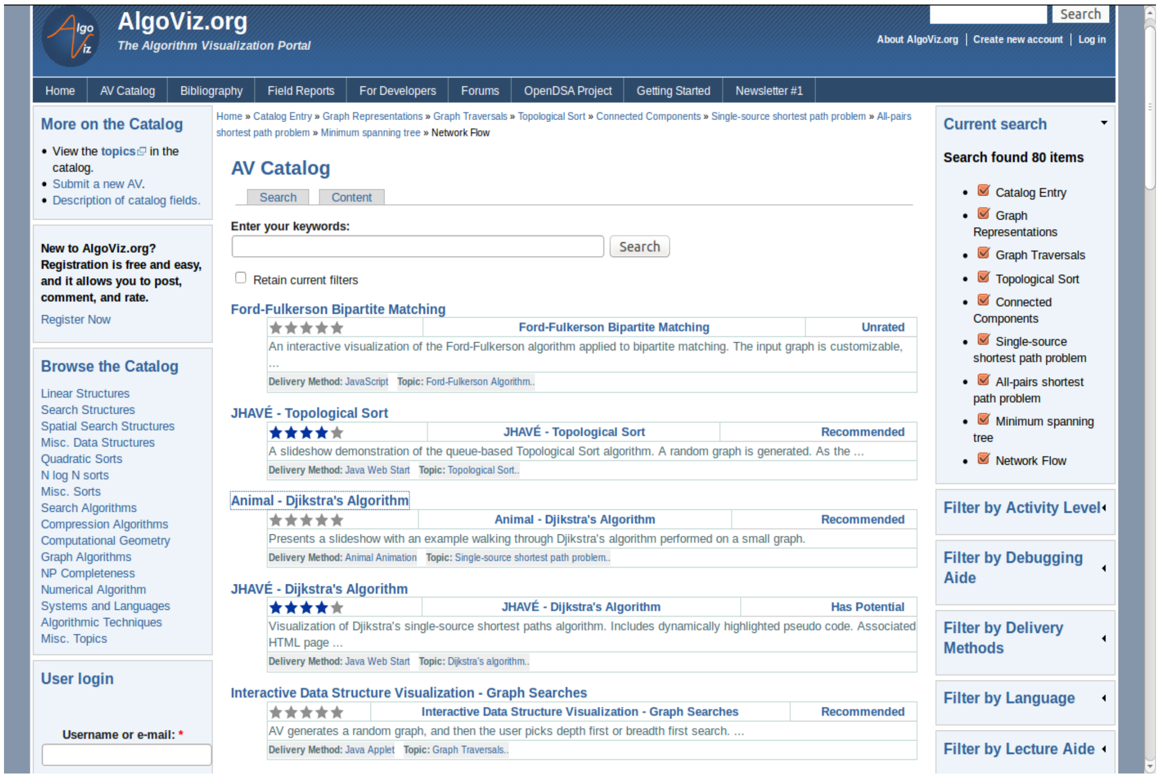


0	7
1	7
2	5
3	7
4	7
5	7
6	-1
7	-7

- http://www.cs.usfca.edu/ galles/visualization/

  It mainly focuses on animation of algorithm.No examples are included in the system and we cannot change the desired input.It includes Graphs,Sorting,Greedy,Compres ,Numerical algorithms.

Figure 2.8: Algoviz System [1]



AlgoViz.org

The Algorithm Visualization Portal

Search

About AlgoViz.org | Create new account | Log in

Home | AV Catalog | Bibliography | Field Reports | For Developers | Forums | OpenDSA Project | Getting Started | Newsletter #1

More on the Catalog

- View the topics in the catalog.
- Submit a new AV.
- Description of catalog fields.

New to AlgoViz.org? Registration is free and easy, and it allows you to post, comment, and rate.

Register Now

Browse the Catalog

Linear Structures
Search Structures
Spatial Search Structures
Misc. Data Structures
Quadratic Sorts
N log N sorts
Misc. Sorts
Search Algorithms
Compression Algorithms
Computational Geometry
Graph Algorithms
NP Completeness
Numerical Algorithm
Systems and Languages
Algorithmic Techniques
Misc. Topics

User login

Username or e-mail: *

Home » Catalog Entry » Graph Representations » Graph Traversals » Topological Sort » Connected Components » Single-source shortest path problem » All-pairs shortest path problem » Minimum spanning tree » Network Flow

AV Catalog

Search | Content

Enter your keywords:

Search

Retain current filters

Ford-Fulkerson Bipartite Matching

★★★★★	Ford-Fulkerson Bipartite Matching	Unrated

An interactive visualization of the Ford-Fulkerson algorithm applied to bipartite matching. The input graph is customizable, ...

Delivery Method: JavaScript Topic: Ford-Fulkerson Algorithm.

JHAVÉ - Topological Sort

★★★★★	JHAVÉ - Topological Sort	Recommended

A slideshow demonstration of the queue-based Topological Sort algorithm. A random graph is generated. As the ...

Delivery Method: Java Web Start Topic: Topological Sort.

Animal - Djikstra's Algorithm

★★★★★	Animal - Djikstra's Algorithm	Recommended

Presents a slideshow with an example walking through Djikstra's algorithm performed on a small graph.

Delivery Method: Animal Animation Topic: Single-source shortest path problem.

JHAVÉ - Dijkstra's Algorithm

★★★★★	JHAVÉ - Dijkstra's Algorithm	Has Potential

Visualization of Djikstra's single-source shortest paths algorithm. Includes dynamically highlighted pseudo code. Associated HTML page ...

Delivery Method: Java Web Start Topic: Dijkstra's algorithm.

Interactive Data Structure Visualization - Graph Searches

★★★★★	Interactive Data Structure Visualization - Graph Searches	Recommended

AV generates a random graph, and then the user picks depth first or breadth first search. ...

Delivery Method: Java Applet Topic: Graph Traversals.

Current search

Search found 80 items

- Catalog Entry
- Graph Representations
- Graph Traversals
- Topological Sort
- Connected Components
- Single-source shortest path problem
- All-pairs shortest path problem
- Minimum spanning tree
- Network Flow

Filter by Activity Level

Filter by Debugging Aide

Filter by Delivery Methods

Filter by Language

Filter by Lecture Aide

- www.algoviz.org-The Algorithm Visualization Portal [1]

  It contains animation of Algorithm and a portal where a collection of links to algorithm visualizations exists. It Integrates many algorithm visualization systems.Only animation is included. They basically have not build their own system.

2.3 Related Table

The following table shows comparisons of various algorithm tutor system.Same table has also been stated in Mukund's thesis [6] as table contains comparisons of systems studied by Mukund and me.

	System	Testing	Educational Theory	Strong Points	Weak Points	Supported Algorithms	Drill n Practice	Real-life example	Interactive
1	”AlgoTutor: AlgoTutor with Visual Algorithm Tracer and Program Pad embedded in it.”	Tutor has been tested on the university students using the concept of pre/post test and results has been shown in the paper.	Concept of ”operation blocks” has been used to design correct algorithm.	”1.It includes program pad so student can see how algo converts to program. 2. Its visualization tool helps student to debug code at each level. 3.Drag and drop interface is user friendly.”	”1.Only for novice learners and does not include high level programs. 2.It is not open source.”	basic level programs,not a particular algorithm support	not provided	not provided	No
2	JHAVE:Algorithm visualization system		Using animation to visualize algorithm	”1.Theory embedded. 2.quiz in between teaching.”	Not interactive.	Many algorithms are supported including graphs,Sorting,Hashing and Miscellaneous.	not provided	not provided	NO
3	ANIMAL		Using animation to visualize algorithm	1.Animation provided with simultaneous code view.	Not interactive	All Major Algorithms provided	not provided	not provided	NO
4	www.algoviz.org-The Algorithm Visualization Portal		Animation of Algorithm and a portal where a collection of links to algorithm visualizations exists	Integrated many algo visualization systems.	Only animation is included. Not own any system.	Graphs, Sorting, Greedy, Compression, Numerical	Not provided	Not provided	No

Table 2.1: Comparison of Algorithm Tutor Systems

	System	Testing	Educational Theory	Strong Points	Weak Points	Supported Algorithms	Drill n Practice	Real-life example	Interactive
5	GREED-EX	participants were computer science majors enrolled at our university in a second-year mandatory course on design and analysis of algorithms,Results were better in experiment group in post test rather in pre-test.	Discovery learning approach	Student discover by himself,results table and history is provided to compare different approaches	No drill and practice included,not checking whether student has understood or not,he may end up with wrong approach in mind.	Activity Selection, Knapsack.	Not provided	Not provided	NO
6	http://www.cs.usfca.edu/galles/visualization/	not available	Animation of algorithms	Visualization of Algorithm is provided	No examples are included,cannot change the desired input,	Graphs, Sorting, Greedy, Compression ,Numerical	No	No	NO
7	GAT-utor	A Pilot test has been conducted,results showed some improvement over the system along with some good comments,Evaluation based on usability,learning and attractiveness done	Guided Discovery Learning	Students can play with algorithm, They are guided if they are wrong, Proof of contradicting conditions are provided	Limited to greedy algorithms	Graphs, Scheduling,Knaapsack	Test has been provided	Uses real life examples	Involvment of student 100 percent of time

Table 2.2: Comparison of Algorithm Tutor Systems

Chapter 3

Design of GATutor

3.1 Introduction

It is an Intelligent Tutoring System which follows a rule based framework to teach a particular algorithm. It helps instructors to make their students teach algorithm in an interesting way through which they can apply algorithms in their daily life.

3.2 Teaching of Greedy Algorithms

3.2.1 Teaching Strategies of other systems

Algorithms Tutor so far existed mainly focuses on teaching greedy algorithms through animation that is visualization of algorithm. Few Systems using such a strategy are Animal, Jhave. Some systems teach algorithms by using building blocks, such as Algo tutor, it gives gives small building block to students, by rearranging them they have to construct a logical program.But this system exists for small programs only.There is a system names Greedex which used the concept of discovery learning which allows students to experiment with different possible logics that exist and then find the optimal and correct one.But it does not guide studetn at any step and also does not make sure that student is learning the right logic.

Basically tutors use theory, visualizations for learning of algorithms.Some uses the concept of discovery learning but then too in very abstract form.

3.2.2 Teaching of Greedy Algorithms-Our Strategy

We have developed a rule based framework for teaching greedy algorithms. All greedy algorithms will follow the same set of rules. Our system follow recall, understand, apply, analyse and evaluate level of Bloom's taxonomy as compared with [5].

1. Basic understanding of what are greedy algorithms(Understanding level)

2. Understanding of specific greedy algorithms(Understanding level)
3. Analysing which selection function is optimal(Analyse level)
4. Proving its optimality by showing counter examples for non-optimal questions(evaluate level)
5. Solving different such problems(Apply level)

We have used basic features of a greedy algorithm as our approach to proceed. Every greedy algorithm basically consist of a selection function and satisfying condition. Selection function can be defined as set of available candidates, returning at each step the most promising candidate with respect to some measure. Satisfying condition is a condition which allows us to .

There invlove some complications in teaching these learning goals at different levels of Bloom's Taxonomy. It is easy to teach one learning goal in a particular algorithm but difficult in other. Teaching a student to how to find out optimal selection function at each stage becomes complicated. Making him learn that the selection function he is thinking is actually optimal or not is other difficulty.

We are using guided discovery learning approach. There have been many variations of definition of discovery learning. Discovery learning as written in [7]occurs whenever the learner is not provided with the target information or conceptual understanding and must find it independently and with only the provided materials. Within discovery-learning methods, there is an opportunity to provide the learners with intensive or, conversely, minimal guidance, and both types can take many forms (e.g., manuals, simulations, feedback, example problems). Discovery learing is also a phenmenon which comes under constructivist approach. It is a approach which says that how knowledge is constructed in human beings when they are provided with existing chunk of knnowledge. The applications of this approach is basically discovery,experiments,group tasks,relating to real life problems. Sweller reported that a better alternative to Discovery Learning was Guided Instruction. Guided Instruction produced more immediate recall of facts than unguided approaches along with longer term transfer and problem-solving skills (Kirschner, Sweller, Clark, 2006).

Support for the regulation the learning process in discovery learning includes various measures:

1. Model progression, such as step-by-step model expansion (e. g. expanding the complexity of the model).
2. Planning support (e. g. using guiding questions, quests or even assignments).

3. Monitoring support (e. g. show what has already be done in the simulation)

4. Structuring the discovery process (e. g. providing students with a sequenced structure such as ”set-up, do, reflect”).

So we have folllowed guided doscovery learning so that student will not divert to wrong path. He can have of proof of his being wrong whenever he jumps to wrong choice.

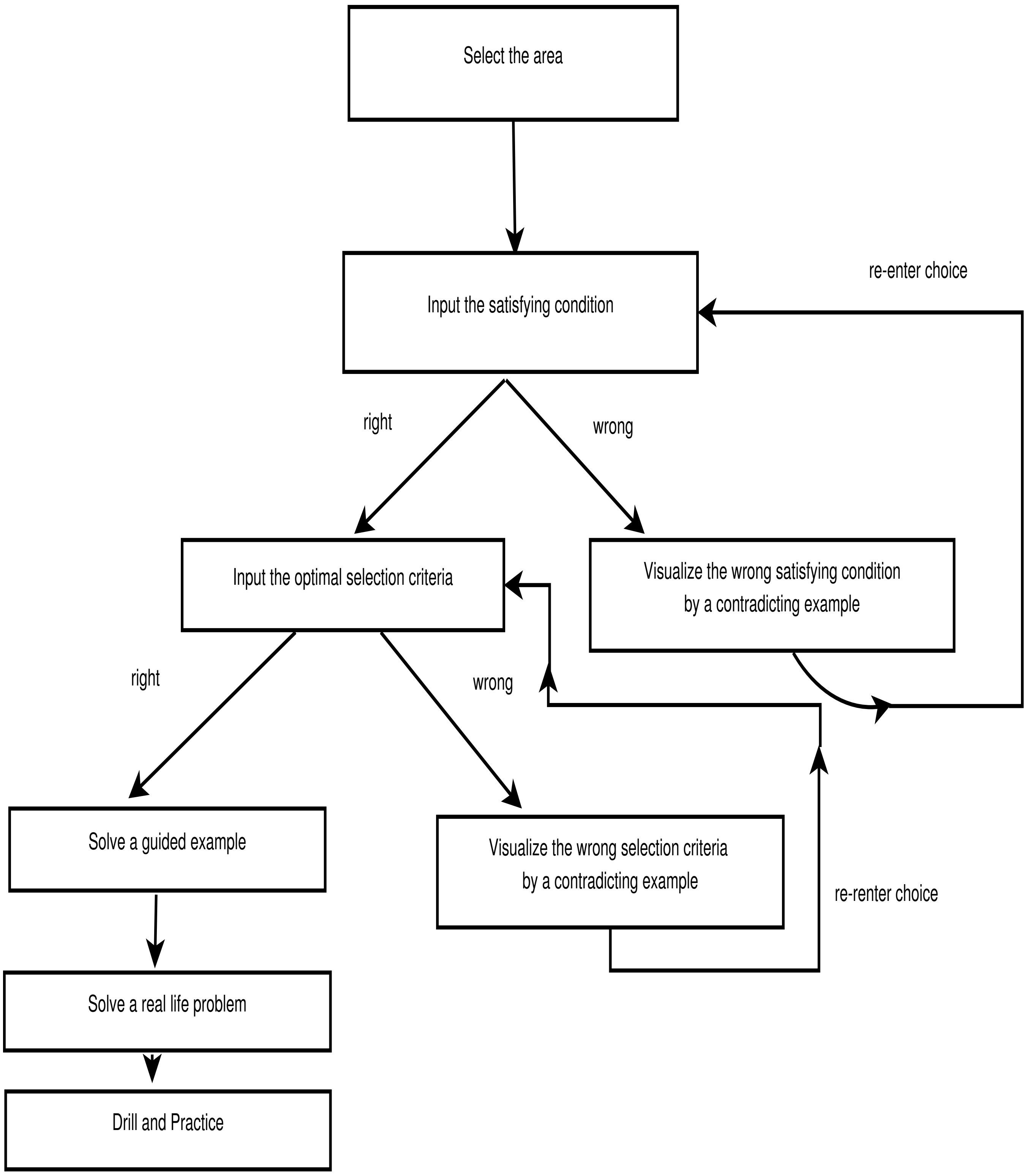
Our framework is as follows:

The learning objective of our framework is to increase the interest and understanding of the student in learning greedy algorithms so that he can apply it whenever needed and will help him to learn other greed algorithm easily.

Greedy algorithms by definition consists of a satisfying condition and a optimal selection function. We have used this as a key measure to develop our system and to make student discover each step of the algorithm himself. We have covered all the areas where greedy algorithm is applied. Whenever the sudent gives wrong answer we give him the proof by a contadicting example. We have related every algorithm to real life to gain consistent interest from student. At the end a drill and practice session have been provided to students to increase their confidence level in the particular algorithm.

3.3 Framework

Figure 3.1: Framework for teaching Greedy Algorithms


Select the area
Input the satisfying condition
re-enter choice
right
wrong
Input the optimal selection criteria
Visualize the wrong satisfying condition by a contradicting example
right
wrong
Solve a guided example
Visualize the wrong selection criteria by a contradicting example
re-renter choice
Solve a real life problem
Drill and Practice

3.4 System Specification

3.4.1 User Characteristics

User must be familiar with basic computer science knowledge to use this system. This system is basically developed for those who are in B.tech 2nd year as greedy algorithms are taught in 2nd year of engineering but it can be used by anyone who is interested to seek knowledge.

3.4.2 Operating Environment

As System is built using Java servlet pages,MySql,Javascript,JQuery it could be run as a web based framework on any operating system which support these technologies.

3.4.3 Users of the system

Student:Student can use this system to learn various Greedy Algorithms. Students should have basic computer science knowledge of what algorithms are and should know how to use a computer based system. Students will have to interact with the system giving input whenever required by the system and then attempting the quiz module which will help them to practice particular algorithm.

Teacher: Teacher are the users of the system who will analyse the data stored by the system that how many students attempted a particular algorithm, their test score in that algorithm, the option they selected each time while going to next step. This will help teachers in knowing the common misconceptions among students.

3.5 Functional Requirements

3.5.1 User as a Student

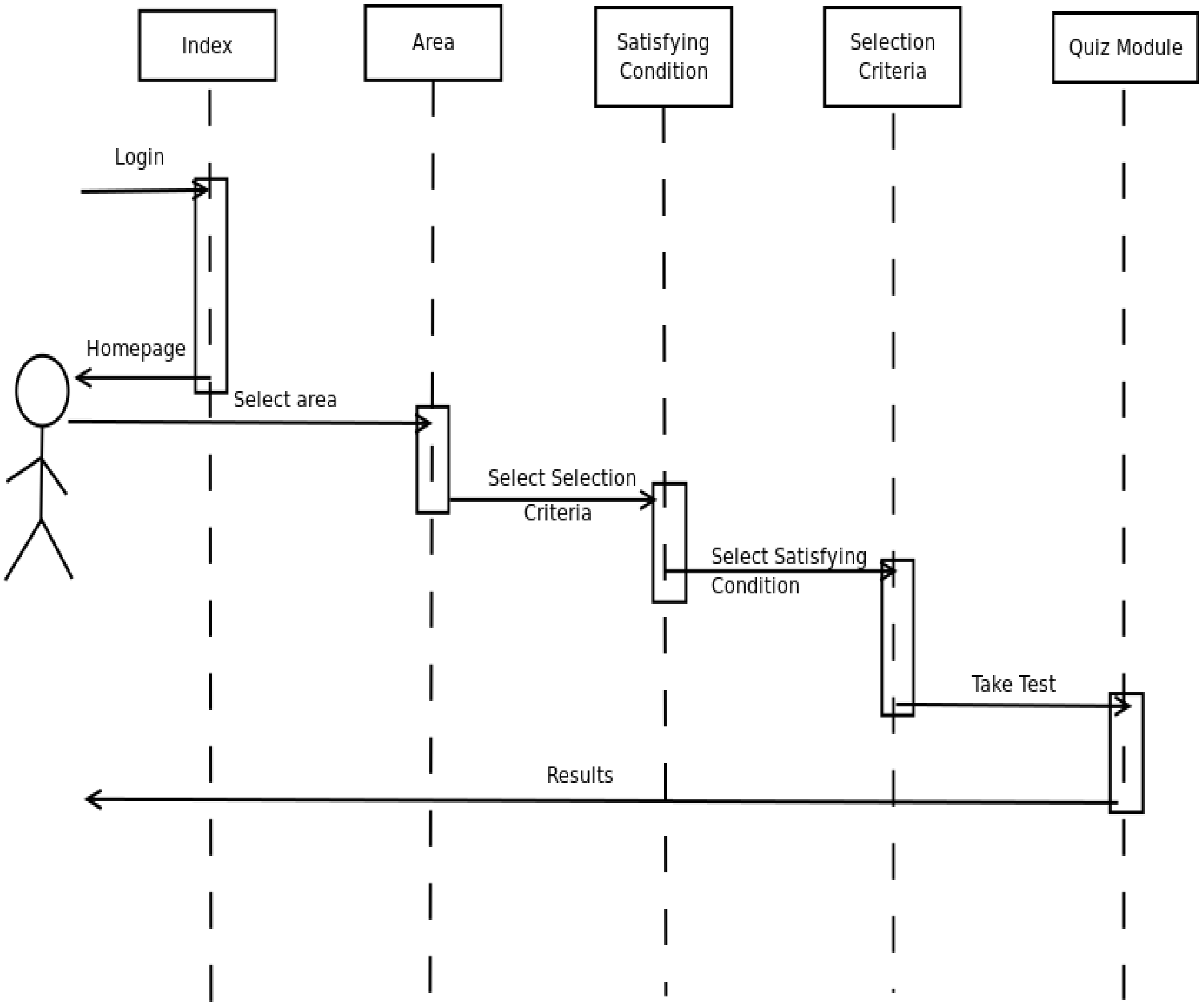
- Student will have to register to access the system, or if already register will just have to login.
- He will need to first select a particular area where he wants to work
- Once selected a particular area, system will drive him through the algorithm.

3.5.2 User as a Teacher

- Teacher will login as admin

- He will analyze the web pages containing the tables related to the student test score,how many students attempted the test, how many completed all four areas of algorithms.
- He can also see the web page which contains a table which how many students selected a particular option of a particular algorithm.

Figure 3.2: Sequence diagram of User Activities



Chapter 4

Design and Implementation of GATutor

4.1 Introduction to system

The system is designed using following technologies:javascript, jQuery, svg images, java server pages and html. There are various design related questions that we should answer. Backend uses mysql database and structured query language to feed students log data.

My part consist of Prim's, Dijkstra's, and Activity Scheduling algorithms. Kruskal's, Knapsack problem is explained in Mukund's thesis [6]. Each algorithm learning is based on the framework shown in 3.1 Login and statistics module and its usage is explained in Mukund's thesis [6] Initially we start by giving a real life puzzle to students , then step step by tells him how to solve it by using a particular algorithm as explained in the flow diagram. Our system consists of following modules:

- Content
- Implementation of the system
- Information storage for analysis

4.1.1 Content

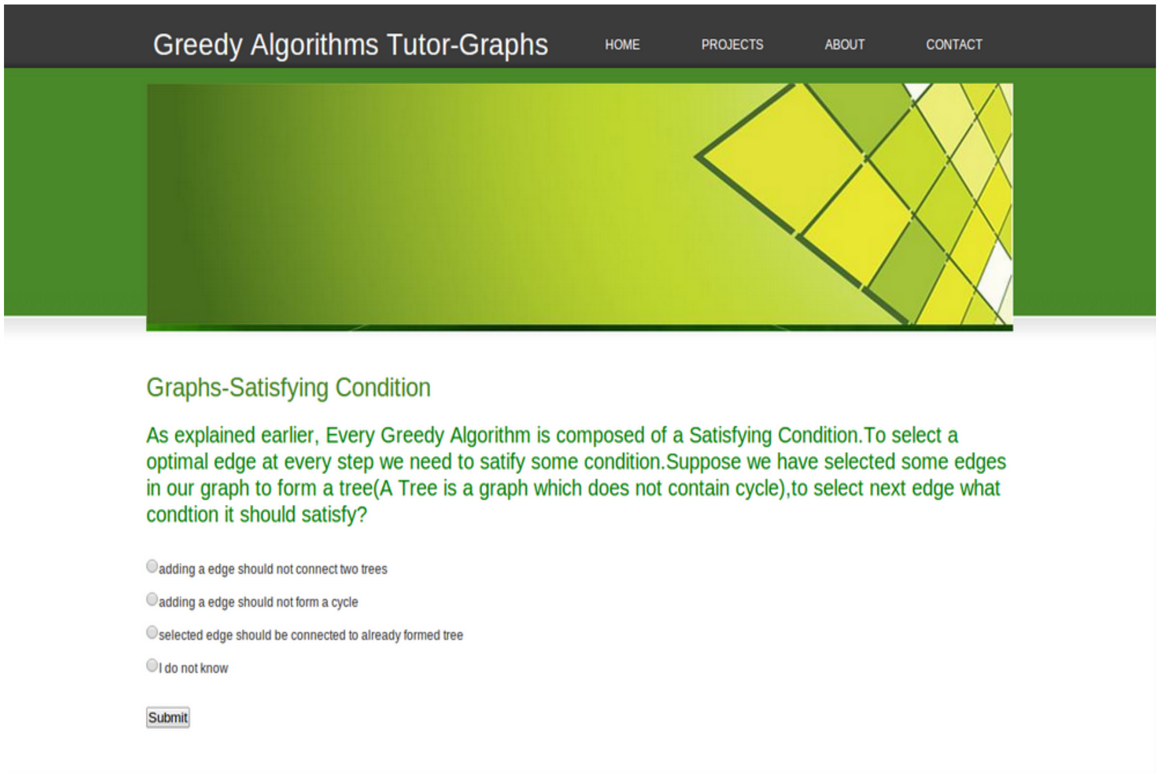
Introduction to Greedy Algorithms

Initially for better understanding of what greedy algorithms are, the system initializes with a introduction to greedy algorithms. The main characteristics of greedy algorithms have been stated and an example have been visualized.

Why Greedy Algorithms

Initially for better understanding of what greedy algorithms are, the system initializes with a introduction to greedy algorithms. The main characteristics

Figure 4.1: Snapshot of satisfying condition page



Graphs-Satisfying Condition

As explained earlier, Every Greedy Algorithm is composed of a Satisfying Condition.To select a optimal edge at every step we need to satify some condition.Suppose we have selected some edges in our graph to form a tree(A Tree is a graph which does not contain cycle),to select next edge what condtion it should satisfy?

- adding a edge should not connect two trees
- adding a edge should not form a cycle
- selected edge should be connected to already formed tree
- I do not know

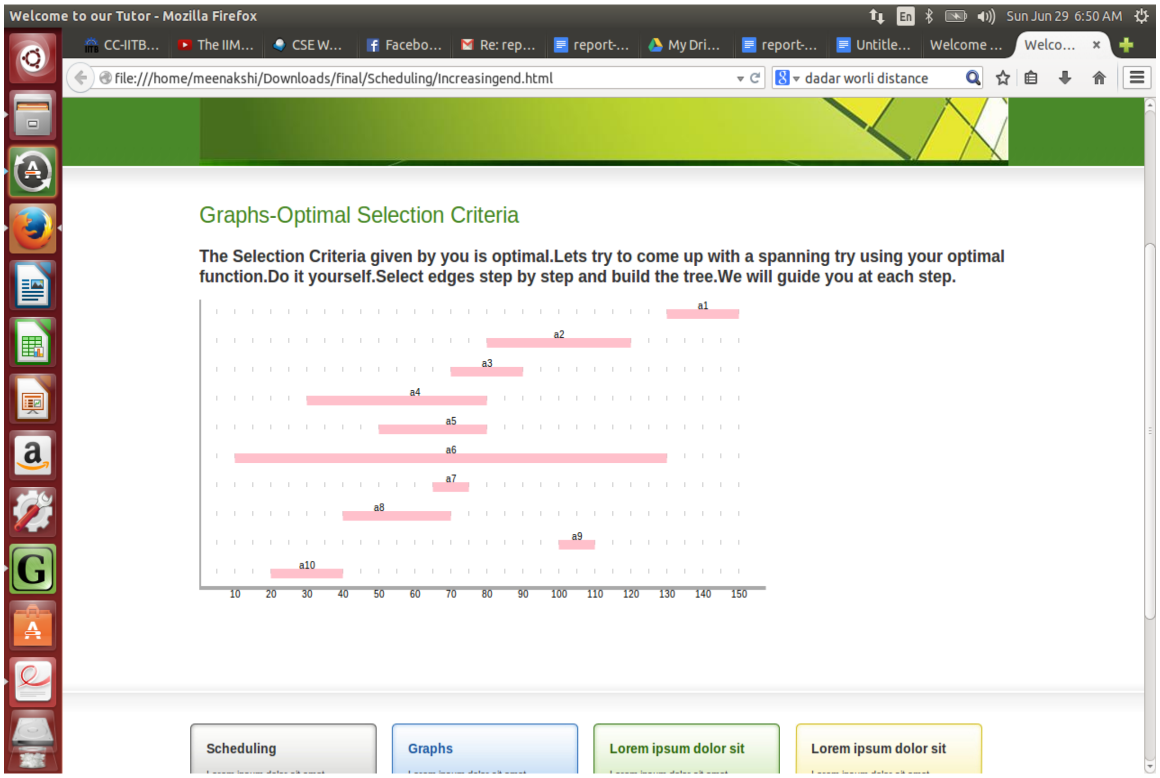
Submit

of greedy algorithms have been stated and an example have been visualized. Greedy algorithms are very common algorithms of computer science, they are used in various areas. They are simple but require a interactive learning to let student design new greedy algorithm in a different situation.

Finding satisfying condition

A satisfying condition allows us to select the next candidate based on the selection function. It results in the valid output. We can exactly explain what is satisfying condition by providing you an example. Activity scheduling problem states that we need to schedule maximum number of activities from the given set of activities in given time frame. Satisfying condition in this problem would be to add next activity to our set the new activity should not overlap with the already selected activities in our present set.

Figure 4.2: Snapshot of Optimal selection criteria page



Graphs-Optimal Selection Criteria

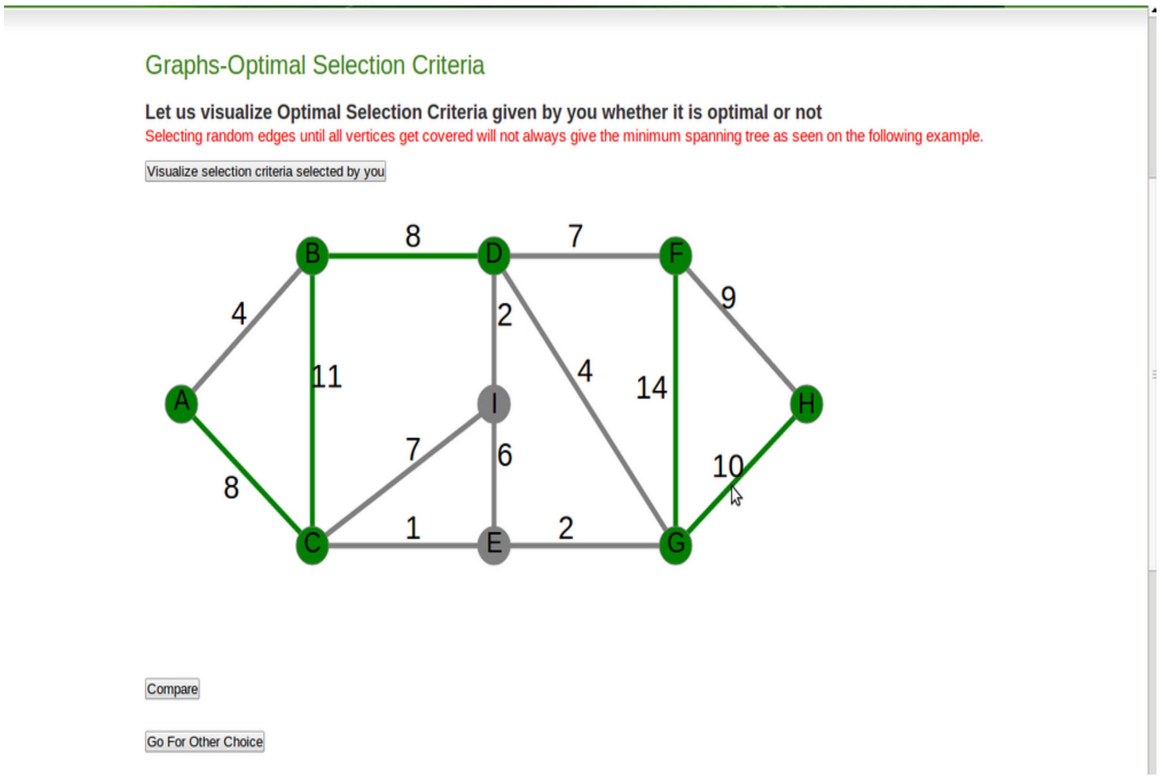
The Selection Criteria given by you is optimal.Lets try to come up with a spanning try using your optimal function.Do it yourself.Select edges step by step and build the tree.We will guide you at each step.

Finding optimal Selection Criteria

A selection function is defined on the set of available candidates, returning at each step the most promising candidate with respect to some measure as taken from ref. We want student to find the optimal selection criteria by himself. This will not make learning boring as student himself will have to discover the optimal function apart from the fact that he will be guided if he goes to the wrong path. To make the design process explicit, we list some selection functions initially and ask learner to discover the optimal one. Optimal selection criteria can be explained by taking example of kruskal's algorithm which results in the formation minimum spanning tree. For this optimal selection criteria would be to select edges in increasing order.

Similarly for activity scheduling problem it would be to select activites in increasing order of their end time.

Figure 4.3: Snapshot of page with contradicting example


Graphs-Optimal Selection Criteria

Let us visualize Optimal Selection Criteria given by you whether it is optimal or not

Selecting random edges until all vertices get covered will not always give the minimum spanning tree as seen on the following example.

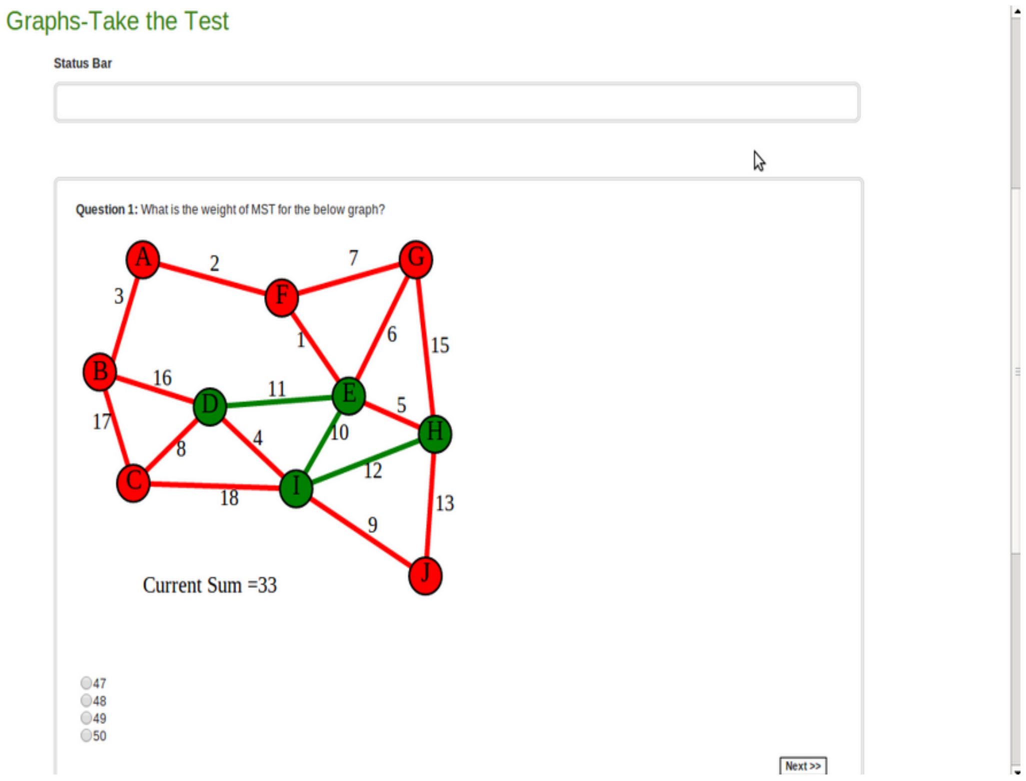
Visualization of Contradicting examples

A controller has been provided which will visulaize the wrong functions based on the user input. It helps in finding the proof of wrong functions and will help in removing common misconceptions from users mind. We have then used a Re-enter choice button so that student can learn from his/her mistake and can change his choice. We have used this until we take the student to right path.

Functionality of this module is to make user realize that what he is thinking coreect is actually wrong.Simply saying that you have selected the wrong option we make hin visualize it by showing the visualization of the wrong option selected.

Visualization is shown by using java script code.

Figure 4.4: Snapshot of quiz page



Question 1: What is the weight of MST for the below graph?

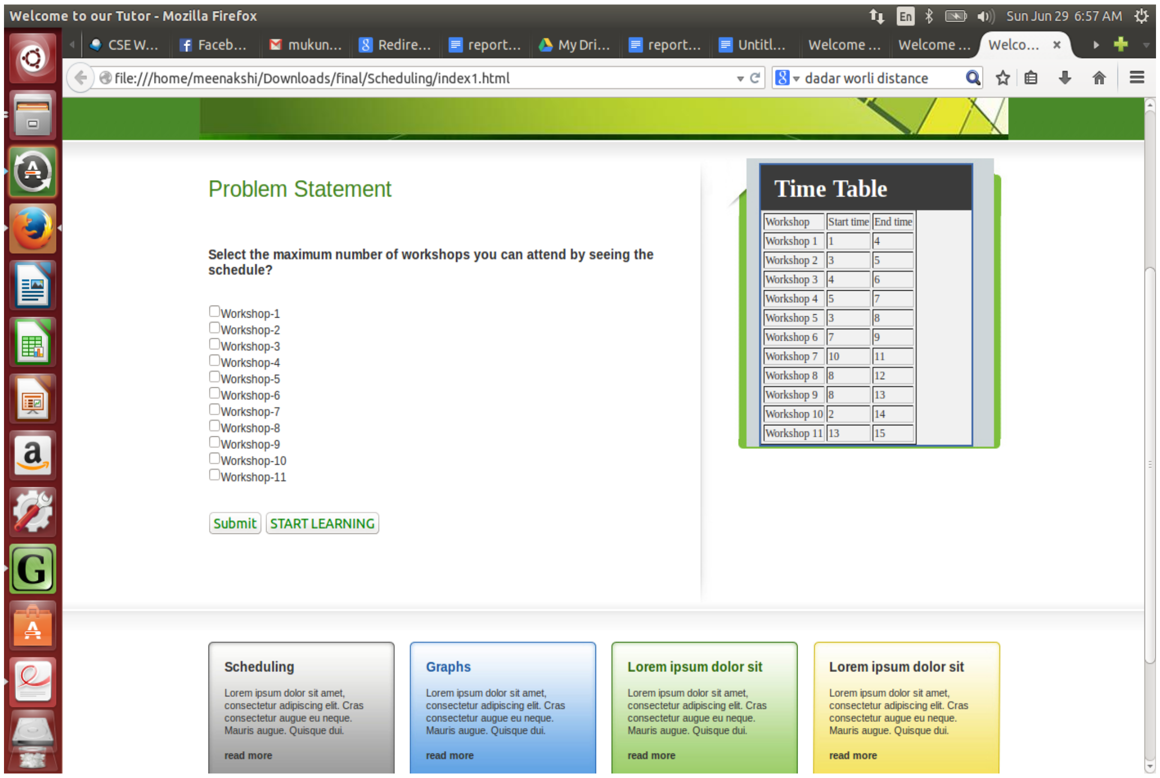
Current Sum =33

47
48
49
50

4.1.2 Evaluation of student for each algorithm to increase his/her confidence and understanding

A quiz module has been provided which acts as a drill and practice session and increases learners confidence. It also has a option of reattempt with weighted marks provided. how quiz module is implemented-using jquery-why we have used jquery

Figure 4.5: Snapshot of real life example page


Welcome to our Tutor - Mozilla Firefox
Sun Jun 29 6:57 AM
file:///home/meenakshi/Downloads/final/Scheduling/index1.html
dadar worli distance

Problem Statement

Select the maximum number of workshops you can attend by seeing the schedule?

- [ ] Workshop-1
- [ ] Workshop-2
- [ ] Workshop-3
- [ ] Workshop-4
- [ ] Workshop-5
- [ ] Workshop-6
- [ ] Workshop-7
- [ ] Workshop-8
- [ ] Workshop-9
- [ ] Workshop-10
- [ ] Workshop-11

Submit START LEARNING

Time Table

Workshop	Start time	End time
Workshop 1	1	4
Workshop 2	3	5
Workshop 3	4	6
Workshop 4	5	7
Workshop 5	3	8
Workshop 6	7	9
Workshop 7	10	11
Workshop 8	8	12
Workshop 9	8	13
Workshop 10	2	14
Workshop 11	13	15

Scheduling
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras consectetur augue eu neque. Mauris augue. Quisque dui.
read more

Graphs
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras consectetur augue eu neque. Mauris augue. Quisque dui.
read more

Lorem ipsum dolor sit
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras consectetur augue eu neque. Mauris augue. Quisque dui.
read more

Lorem ipsum dolor sit
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras consectetur augue eu neque. Mauris augue. Quisque dui.
read more

Relating to real life problems

We have related every algorithm to real life problem so that student finds interest in solving it. We have first introduced a real life problem, asks student to solve it, and then tell him how he can solve it easily using a specific algorithm.

4.1.3 Implementation and Design of System

Java servlet pages embedded with java script

We have used JSP pages for displaying the theoritical content of the system. For coding a pariticular algorithm, we have used java script. It receives the input by the used and accordingly manages the content of the jsp pages, then output the respective jsp page.

Below is the code of Prim's Algorithm. It is written in java script. We have used svg code for displaying the images, and accordingly used SVGObjectElement and SVGPathElement to detect the input from user on the image. After recieving the input we have changed the value in the images accordingly.

We have made 87 Java servlet pages consisting of java script code implementing variations of different algorithms to help student find out correct algorithm him selves.

```
var arr_id = new Array("CE","EG","DI","DG","AB","DF","AC","FH");
var i=0;
function modify(e){
    var t = e.target;
    if(t=="[object SVGPathElement]")
        if(t.id==arr_id[i])
        {t.setAttribute("stroke","green");i=i+1;document.getElementById
        (t.id[0]).setAttribute("fill","green");document.getElementById
        (t.id[1]).setAttribute("fill","green");union( find(t.id[0]) ,
        find(t.id[1]) );}
        else if(t.id=="DI" && i==1 || t.id=="AB" && i==3)
        {t.setAttribute("stroke","green");var temp=arr_id[i];arr_id
        [i]=arr_id[i+1];arr_id[i+1]=temp;i=i+1;document.getElementById
        (t.id[0]).setAttribute("fill","green");document.getElementById(
        t.id[1]).setAttribute("fill","green");union( find(t.id[0]) ,
        find(t.id[1]) );}
        else if(t.id=="BD" && arr_id[i]=="AC")
        {t.setAttribute("stroke","green");arr_id[i]="BD";i=i
        +1;document.getElementById(t.id[0]).setAttribute
        ("fill","green");document.getElementById(t.id[1]).setAttribute
        ("fill","green");union( find(t.id[0]) , find(t.id[1]) );}
        else alert("Sorry your choice is not consistent with the
        chosen selection criterion");
}
  document.documentElement.addEventListener('click',function(e){if
  (check(e)==1)alert("Tree should not have a cycle!!");else modify(e);
  if(i==8)
  {alert("Congratulations!!");
  document.getElementById("val").innerHTML="You have formed the
  minimum spanning tree.Now help the government to design a topology
  for electricity distribution";
  }},false);
  ]]></script>
```

SVG images with clickable interface

SVG stands for Scalable vector Graphics. It is basically used to define vector based graphics for the web. As our system is web based, it was the best option for us because they are scalable, and every element and every attribute in SVG files can be animated.

We have used this property to make clickable images which gives a feel like that of playing a game. As per our input these images changes their values of different elements and objects thus making the system more interactive and interesting to the user.

JQuery used for implementing quiz module

We have used JQuery for building the quiz module. JQuery is a library of javascript that makes it easier for users to control, manage different events, animations at an html page. To show the moving bar in the quiz, to make it sure that students attempt all question and attempting a question in later attempt gives reduced marks to student proportional to the number of attempts has been taken care of.

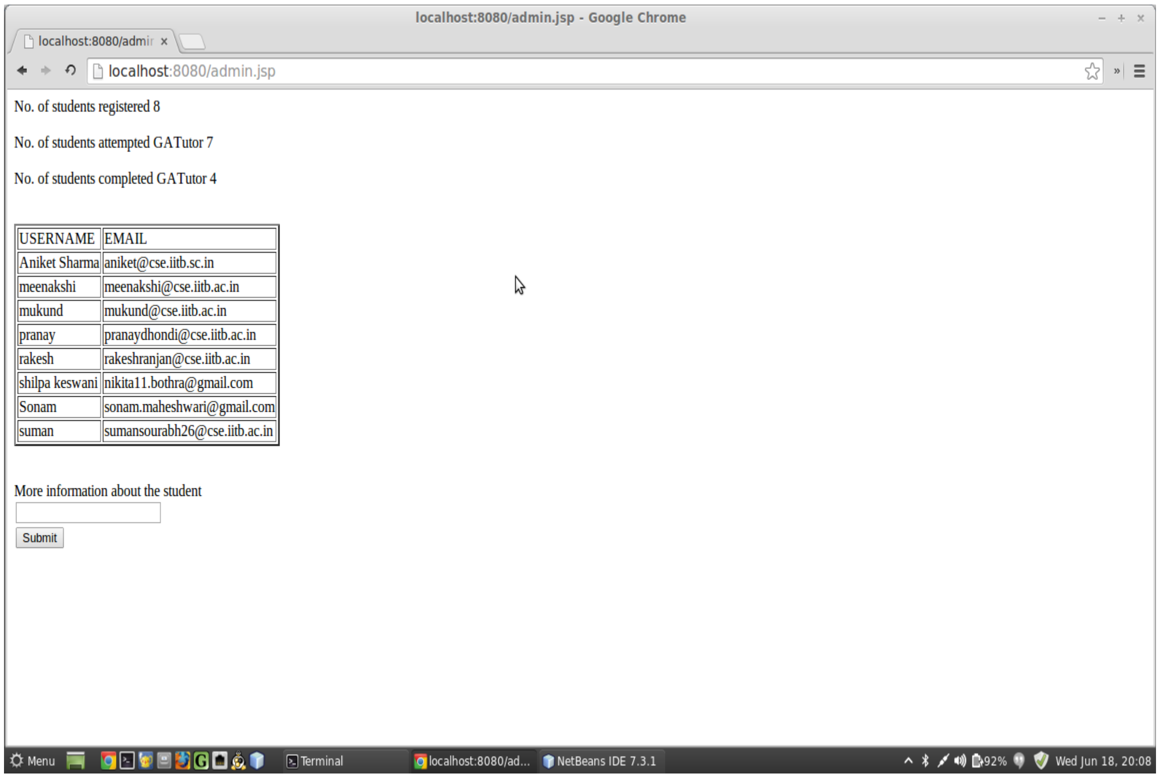
Other Benefits of JQuery are it is lightweight and does not delay the opening of web page as other technologies do such as Flash. 4 lines written in JQuery is equivalent to 16 line of javascript code and we can easily create flash like effects using simple plugins of JQuery library.

4.1.4 Information Storage for analysis

We need to store students information, about how he traversed the system, how many algorithms he attempted, his test score in all attempts to know about his learning process. So storing student information has been prove useful for the instructors to know the statistics of the class. It can also be used to find out the common misconceptions of the class as there is a table which displays number of times an option have been selected.

The detailed design of database is shown in Mukund's Thesis [6]. Below are the snapshots of the information which is displayed to instructor in 4.6.

Figure 4.6: Snapshot of Information displayed to Admin



No. of students registered 8

No. of students attempted GATutor 7

No. of students completed GATutor 4

USERNAME	EMAIL
Aniket Sharma	aniket@cse.iitb.sc.in
meenakshi	meenakshi@cse.iitb.ac.in
mukund	mukund@cse.iitb.ac.in
pranay	pranaydhondi@cse.iitb.ac.in
rakesh	rakeshranjan@cse.iitb.ac.in
shilpa keswani	nikita11.bothra@gmail.com
Sonam	sonam.maheshwari@gmail.com
suman	sumansourabh26@cse.iitb.ac.in

More information about the student

Submit

Chapter 5

Evaluation of the System

5.1 Software Testing

A regressive testing has been conducted on the system to ensure that all link are working properly and the output provided at each stage is correct.

5.2 Learning and Attractiveness

Evaluation of system based on learning and attractiveness has been explained in detailed in Mukund's Thesis [6].

5.3 Usability

5.3.1 Implementation

We have used System Usability Scale(SUS) to measure the usability of the system. We took 20 Btech 2nd year students for testing our system.

5.3.2 Sample

We took 20 students, from B.tech 2nd year from IIT Bombay who were not knowing about this algorithm before. We asked them to fill a survey form to evaluate our system on different parameters.

5.3.3 Data Collection

-

We asked the students to fill System Usability Scale(SUS) questionare based on 4 point likert scale that is Strongly agree,Agree,Disagree and Strongly Disagree.

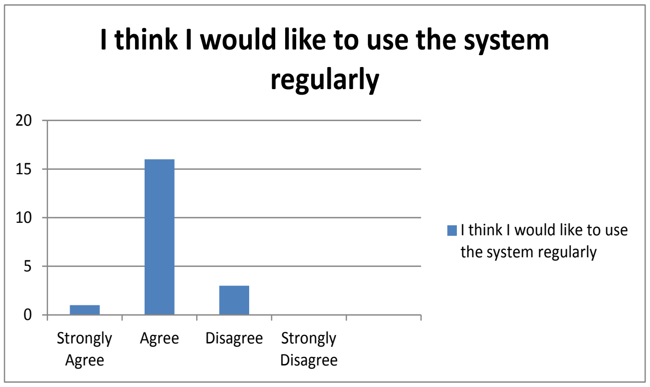
Following were the questions:

1. I think that I would like to use this system frequently.
2. I found the system unnecessarily complex.
3. I thought the system was easy to use.
4. I think that I would need the support of a technical person to be able to use this system.
5. I found the various functions in this system were well integrated.
6. I thought there was too much inconsistency in this system.
7. I would imagine that most people would learn to use this system very quickly.
8. I found the system very cumbersome to use.
9. I felt very confident using the system.
10. I needed to learn a lot of things before I could get going with this system.

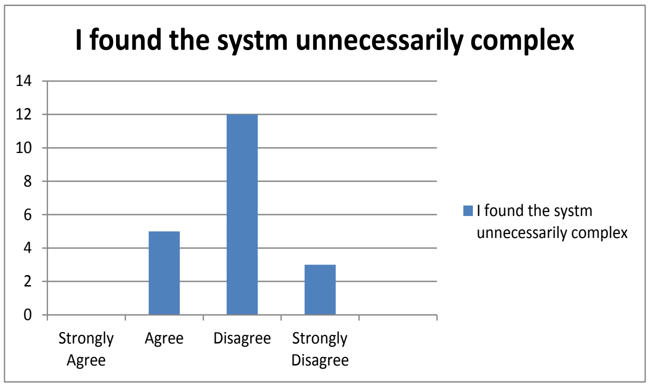
5.3.4 Results

Following are the results shown in graphical form.

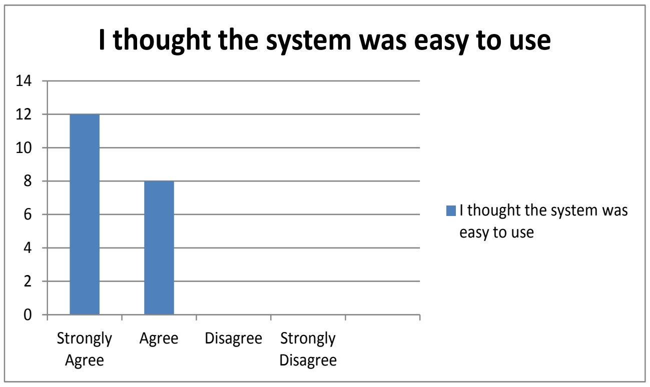
- 1: Total 16 students agree that they would like to use the system regularly for others topics also. We can infer from this as maximum students want to use the system regularly, they would have definitely found it interesting.



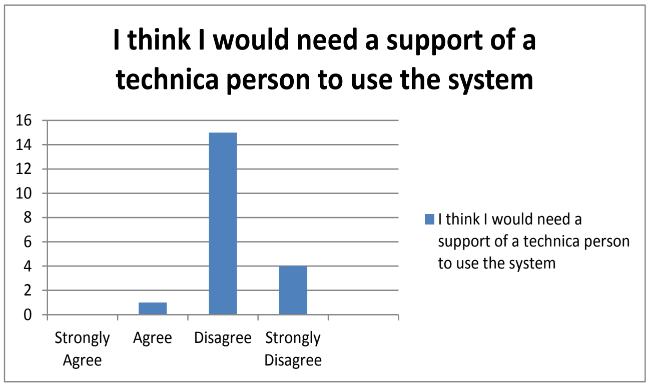
- 2: Only 5 students agree with the fact that system was unneccessarily complex,we also got some feedback about our changes for our user interface, implementing those can make our system more easy to use.



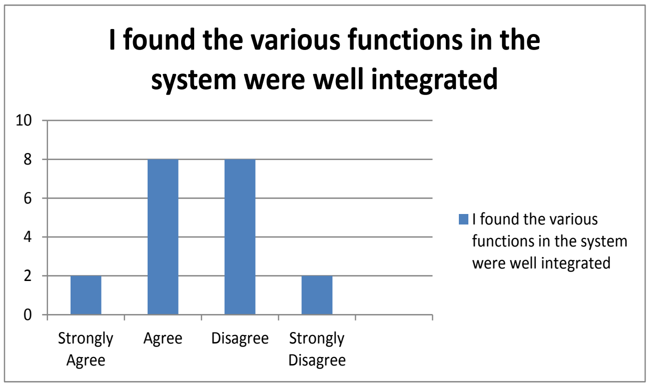
- 3: We can easily conclude that system was easy to use for students as not a single student was against it.



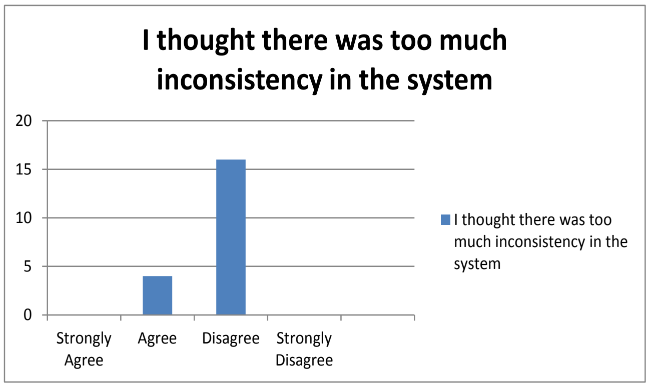
- 4: Students would not require need of a technical person to access the system. It is beneficial for a environment where student independently wants to learn a topic and does not have any one for support.



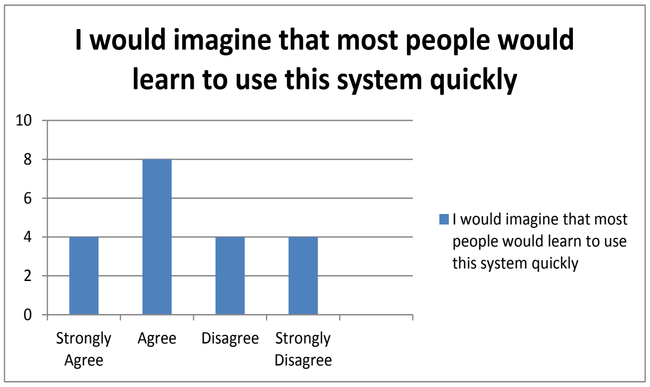
- 5: It was a ambiougous scenario for us as we got the same votes in the favour as against. We need to integrate the modules for the ease of usage of them. :



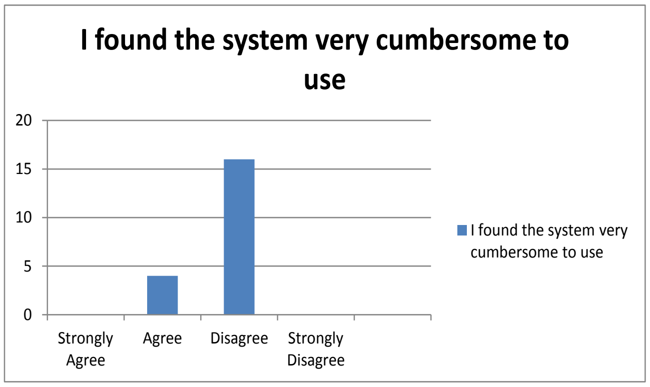
- 6: System was not inconsistent except a few places where students were finding it difficult to click on the images.



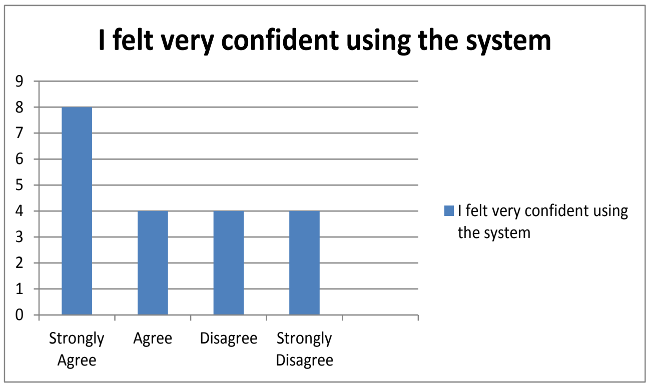
- 7: The system can be used in schools and colleges where there are large number of students with different intellectual level, as system does not require any special skill and can be used by a normal student to learn algorithm.



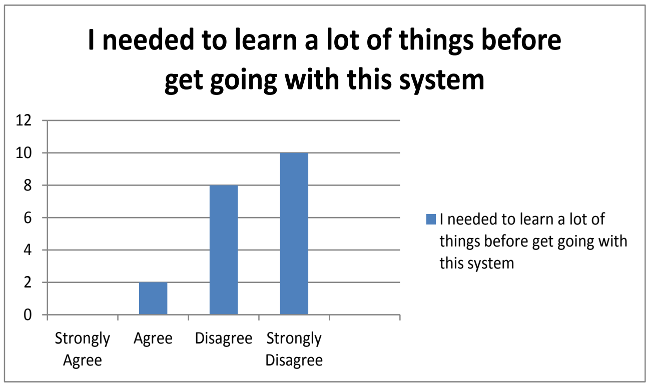
- 8:It can be easily concluded from the pole that system was not cumbersome to use.



- 9: It increased confidence of the student which can infer that student was able to learn the algorithm effectively.



- 10: System just require you to have a basic computer science knowledge nothing more than that.



We can draw following inferences from the results:

- From the graphs we can figure out that students would like to use the system for other topics also and would prefer to use this system for studying algorithms. System was overall easy to use and user friendly which can be depicted from questions 2 and 3. The system can be used independently and it was homogeneous too. It increased students understanding of the topic.

Chapter 6

Conclusion and Future Work

Our system to teach Greedy Algorithms to students through visualization is ready. It contains most of the popular greedy algorithms. We have used the approach of guided discovery learning along with an interactive interface which makes this system unique in the area of Computer Science. Through Pilot experiments conducted in first phase of this project, we were able to detect the minor faults which existed in the system and it helped us to make the system more effective. We tested the system on a bunch of students to know its effectiveness and we were proved right. The system is effective overall in teaching greedy algorithms in an active way.

In near future, we would like this system to contain almost all popular algorithms of computer science with this general framework. It would be really helpful for students to study from this system as they will gain confidence while working on this system as it will make them feel that they themselves have explored the algorithm. We would also like to extend this system to built a scratch pad which will convert the algorithmic steps developed by the students into a pseudo code and then into running code where students can run their algorithm and see the code.

Bibliography

[1] AlgoViz.org. The algorithm visualization portal, http:// algoviz.org/, 2012.

[2] David Galles. http://www.cs.usfca.edu/ galles/visualization, harney science center, department of computer science, university of san francisco.

[3] M. Ashraf Iqbal and Sara Tahir. Should we teach algorithms?,iranian journal of electrical and computer engineering, vol.2, no.2, summer-fall 2003.

[4] IEEE Computer Society J. Angel Velazquez-Iturbide, Member. Greedex: A visualization tool for experimentation and discovery learning of greedy algorithms.

[5] Madrid Spain J. ngel Velzquez-Iturbide Departamento de Lenguajes y Sistemas Informticos, I Escuela Tcnica Superior de Ingeniera Informtica Universidad Rey Juan Carlos C/ Tulipn s/n 28933 Mstoles. The design and coding of greedy algorithms revisited.

[6] Mukund Lahoti. Gatutor: Intelligent tutoring system for greedy algorithms, m.tech thesis, iit bombay, 2014.

[7] Naomi J. Aldrich Louis Alfieri, Patricia J. Brooks and Kingston University City University of New York Harriet R. Tenenbaum. Does discovery-based instruction enhance learning?

[8] T.L. Naps. Jhave: Supporting algorithm visualization, ieee computer graphics and applications, vol. 25, no. 5, pp. 49-55, sept. 2005.

[9] Polya. How to solve it; a new aspect of mathematical method ,princeton university press, 1988.

[10] G. Roling and B. Freisleben. Animal: A system for supporting multiple roles in algorithm animation, visual languages and computing, vol. 13, no. 2, pp. 341-542, 2002.

[11] Jungsoon Yoo Chrisila Pettey Suk Seo and USA Sung Yoo, Middle Tennessee State University. Teaching programming concepts using algorithm tutor.

[12] J.A. Velazquez-Iturbide and A. Perez-Carrasco. Active learning of greedy algorithms by means of interactive experimentation,proc. 14th ann. conf. innovation and technology in computer science education (iticse 09), pp. 119-123, 2009.