

Development of Intelligent Tutoring System Framework: Using Guided Discovery Learning

M. Tech. Project Dissertation

Submitted in partial fulfillment of the requirements
for the degree of

Master of Technology

by

Raja Shekhar

Roll No: 10305034

under the guidance of

Prof. Sridhar Iyer



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

Abstract

On-line learning and distance education are significantly gaining importance in the recent past, which demands the development of on-line tutoring systems. Merely presenting the content to the learner will not serve the tutoring purpose. The tutoring system should adapt itself to learner's subject knowledge. Tutoring becomes effective, if it includes hands on activities and multiple ways of teaching. We present the details of a tutoring system framework that adapts to learner's subject knowledge and supports 4 teaching styles namely guided discovery, socratic questioning(Q & A), scaffolding and game based learning. We are team of 4 members working on the development of this framework. In this report, we explain the ITS frame work using guided discovery teaching style.

Contents

1	Introduction	1
1.1	ITS	1
1.2	ITS framework & strategies	2
1.2.1	Motivation	2
1.2.2	Existing systems	2
1.2.3	Our approach	2
1.3	My work summary	3
2	Literature Survey	4
2.1	Intelligent tutoring system	4
2.1.1	General components of ITS	4
2.1.2	Specific case studies	4
2.2	Strategies	6
2.2.1	Guided discovery learning	6
2.2.2	Socratic questioning	7
2.2.3	Scaffolding	7
2.2.4	Game based learning	7
3	ITS Design Details for Guided Discovery Learning Strategy	8
3.1	Guided discovery using ITS framework	8
3.2	Steps to be followed	9
3.2.1	Steps to be followed by instructor	9
3.2.2	Steps to be followed by learner	9
3.2.3	Steps to be followed by ITS	9
4	ITS Framework & Work Flow	11
4.1	ITS framework architecture	11
4.2	Time sequence diagram	12
5	Implementation	14
5.1	Description of Modules	14
5.1.1	Login module	14
5.1.2	Course module	15
5.1.3	Topic module	15
5.1.4	Subtopic module	15
5.1.5	Quiz module	15
5.1.6	Administrator module	16
5.1.7	Controller module	16
5.2	Database	17
5.3	Challenges	18

6	Integration & Testing	19
6.1	Integration of the system	19
6.2	Tables	19
6.2.1	Common tables	19
6.2.2	Non-common tables	20
6.3	Testing	20
6.4	screen shots	22
7	Conclusion and Future Work	26
A	Database Schema	27
B	Source Code for Modified DFS Algorithm	31

List of Figures

2.1	SQLT-Web ITS architecture	6
2.2	Steps in Guided Discovery Learning Strategy	7
4.1	ITS framework architecture	11
4.2	ITS framework-Handling instructor interactions	12
4.3	ITS framework-Handling learner interactions	13
5.1	ITS directory structure	14
5.2	Database structure	17
6.1	ITS home page	22
6.2	Instructor home page	23
6.3	Creating a topic	23
6.4	Entering topic dependencies	24
6.5	Entering question into quiz	24
6.6	Student selecting quiz	25
6.7	Interactive window	25
A.1	course table	27
A.2	topic dependency table	27
A.3	login table	28
A.4	question table	28
A.5	upload file table	28
A.6	strategy sequence table	29
A.7	strategy table	29
A.8	student response table	29
A.9	subtopic dependency table	30
A.10	subtopic table	30
A.11	topic table	30

Chapter 1

Introduction

An intelligent tutoring system(ITS) is a software used for tutoring purpose. ITS adapts teaching according to the learner's subject knowledge and minimizes human instructor intervention in teaching. The instructor will input the required subject data(domain knowledge) to ITS. ITS uses this domain knowledge to teach learner. Learner gains subject knowledge by interacting with the ITS. Initially ITS assigns a knowledge level to each learner. Based on learner's performance, ITS provides feedback to learner, updates student knowledge level and uses this updated student level for further teaching. Thus it can replace a human tutor. ITSs are used to teach subjects which require less human instructor interaction like on-line courses and distance education.

Single method of teaching may not suit to teach all the topics. We need different styles of teaching for different concepts. The system which provides ITS functionality and supports multiple styles of teaching is ITS framework. Below we explain ITS, ITS framework and the strategies used for teaching.

1.1 ITS

An ITS is a software system that provides responsive and interactive teaching facilities for learners, tracks their progress , assesses their performance, sequences the curriculum and helps the learners to improve without human instructor intervention.

Responsive: ITS provides proper response and feedback to each learner action.

Interactive: Learner can easily interact with the ITS.

Tracking progress: ITS stores what topics learner covered to track learner process.

Assess performance: Evaluates and stores the results of learner's performance.

Help in improving: Feedback and hints provided by ITS help the learner to improve his pace and depth of learning.

The goal of ITSs development is to provide the benefits of one-on-one instruction automatically and cost effectively. ITSs are of 2 types[12].

- Standalone: The ITS software is installed on learner's machine and learner can access the ITS when he runs the software.
- Web-based: The ITS software is installed and runs continuously on a system, which functions as server. Learners can access the ITS through internet.

The web-based ITSs have following advantages over stand alone ITS.

- Learners are not constrained to use specific machines in their schools, and can access ITS from any location and at any time.

- Distributing software to learners and hardware/software compatibility problems are minimized.
- Updated versions of the ITSs will be available to learners.

Our frame work is web-based type ITS.

1.2 ITS framework & strategies

1.2.1 Motivation

In traditional class room teaching, instructor knows the knowledge of learners and adapts teaching to meet the learners knowledge. Instructor presents examples first and then generalizes the principles to teach some concepts or simply instructs learners to perform hands on activities and understand principles to teach some other concepts. Thus instructor uses different styles to teach different concepts. As the interaction between instructor and learner is less in distance education and on-line learning, we need an automated system which can tutor in multiple styles and adapt the teaching according to learner's knowledge to support on line learning. ITS framework is a technology solution to support the the above mentioned features.

1.2.2 Existing systems

ITSs are developed to teach various subjects like geography, circuits, medical diagnosis, computer programming, mathematics, physics, genetics, chemistry, etc.[1]. I read various ITSs namely autotutor-teaches physics, sqlt-web -teaches sql, activemath- teaches mathematics etc. Most of the available ITSs are developed with the aim of teaching one subject. Their implementation is dependent on the specific subject. Thus they can't support other kinds of subjects using the same framework. They follow single teaching style. In chapter 2, we explain sqlt-web ITS architecture as an example. The ITS which can tutor multiple subjects using multiple teaching-learning styles is not yet developed. We are team of 4 members working on the development of such ITS framework.

1.2.3 Our approach

Our ITS contains multiples courses. Each course can have multiple topics, each topic can have multiple subtopics and each subtopic has a quiz associated to it. The 4 strategies of our ITS framework follow the above mentioned course structure. In the 4 strategies we are implementing, instructor poses different types of questions to learner in different sequence to teach a concept. In guided discovery, instructor asks questions which require hands on activities to answer the questions. In question & answer strategy instructor asks a question to teach a concept. If learner answers wrong then instructor identifies the misconception the user has and asks another question to clarify the user's misconception. Thus all strategies use questions to teach the subject. So in our ITS framework, we store the subject data in the form of multiple-choice questions. The difference between the strategies is in the type of questions posed and the sequence of presenting the questions to the learner.

To teach a concept instructor creates the subtopic and adds questions to the quiz associated to that subtopic. Instructor chooses the strategy to be used at the time of entering questions into the quiz. When learner selects that subtopic ITS determines the strategy used and executes the corresponding strategy's logic to fetch the questions in proper order.

1.3 My work summary

We started with the survey of different ITSs and found that the method of evaluating the learner answers is the thing which restricts ITSs from supporting various types of subjects. We found that no attempts are made to build the ITS which supports multiple ways of teaching and web-based ITSs have advantages over stand-alone systems. So we planned to develop a web-based ITS and use multiple choice questions to make the evaluation scheme independent of subject. Then we studied different teaching-learning strategies to find the methods which fit into our framework. We chose the following 4 strategies.

- Guided discovery : This strategy is implemented by me.
- Socratic questioning : Implemented by Vikash.
- Scaffolding : Implemented by Chandrapal.
- Game based learning : Implemented by Praveen Dhanala.

Here we briefly explain the ITS for guided discovery learning strategy. Guided discovery uses hands-on activities for teaching. Hands-on approach is provided by creating a pop-up window through which the learner can enter commands and monitor the results. Initially we designed the database schema and the interface. An instructor can create number of courses, topics in each course and subtopics in each topic. As mentioned multiple choice questions are entered into each subtopic. The topics and subtopics in each topic should be taught in specific order. So we store 2 levels of dependencies, one at topic level(stores dependencies among topics) and the other at subtopic level(stores dependencies among subtopics) in order to ensure the proper sequence of learning. The criteria used for subtopic dependency is the threshold values set by instructor per each subtopic and criteria for topic level dependency is the completion of subtopics in each dependent topic. The questions in subtopics are of 2 types namely guiding questions, testing questions. To teach a concept we pose guiding questions which requires some hands-on methods to answer them. For guiding questions, we provide an interactive window through which the learner can enter commands and monitor the results and learn from them. Then we pose some test questions to check how much the user has learnt, these questions do not provide the support for hands-on activities. When learner tries to attempt a topic or subtopic ITS checks whether the dependencies are satisfied or not. ITS first displays guiding questions and then poses testing questions. Initially we designed each of our systems independently with common database, interface and later integrated into single system.

Report outline: In chapter 2 we explain an example ITS architecture and the 4 strategies implemented by us. How ITS supports guided discovery learning is explained in chapter 3. In chapter 4 we explain how the work flows in our ITS. Chapter 5 explains the implementation details of ITS including design decisions of tables, modules and functions of ITS. Integration of the system and screen shots of system is explained in chapter 6. Chapter 7 explains the scope for future work on ITS framework.

Chapter 2

Literature Survey

2.1 Intelligent tutoring system

2.1.1 General components of ITS

There are 4 basic components of ITSs namely domain model, student model, teaching model and user interface.

1. **Domain model** : It contains the subject content like facts, rules and problem-solving strategies of the particular domain to be taught to the student. The expert knowledge module or domain expert or domain model serves 2 functions
 - (a) Serves as the source of knowledge to be presented to the student, which includes generating questions, explanations and responses.
 - (b) Provides a standard for evaluating the student's performance. It uses the rules and problem-solving strategies stored in it to generate the solutions to the problems submitted by user and identifies the gap between student response and ideal solution.
2. **Student Model** : It stores the knowledge of the user as the learning process advances. It serves as a source of information about the student and representation of the student's knowledge. It stores the data like which concepts the user has learnt and the weak areas of the learner. This data is used by teaching model for adaptation.
3. **Teaching Model** : Also known as teaching strategy or pedagogic module.
 - (a) It accepts information from the domain and student models and makes choices about tutoring strategies and actions.
 - (b) It is the logic containing the steps to be taken according to learner responses.
 - (c) It gets the student knowledge from student model and then compares it with the target goal and then performs actions like providing hints or feedback to overcome impasses in performance, explanations to tasks etc.
4. **User(student/instructor) interface model** : Provides interactive GUI for the user.

2.1.2 Specific case studies

Following are some ITSs which use different ways of interaction with user and developed using different technologies.

- Autotutor[2]: Stand alone ITS to teach Physics. It supports voice interaction between learner and ITS. ITS can ask questions and respond to the questions posed by learner.

- ActiveMath[11]: Web based ITS to teach Mathematics. It cant answer to the learners questions. Supports GUI interaction.
- SQLT-web tutor[12]: Web-enabled ITS to teach SQL. Supports GUI interaction.

Below we explain the SQLT-Web tutor modules and architecture.

An example ITS: SQLT-Web tutor

SQLT-Web tutor is a web-enabled ITS to teach SQL. It is developed in Common Lisp-HTTP(CL-HTTP). The interface displayed to learner contains a question and the basic prototype of a query i.e. the fields like from, where, groupby to write a query are displayed in proper order with empty text boxes beside them. Learner reads the question and fills in the text boxes with proper text and submits. ITS evaluates and reports errors if any. The user can pick the next question or ask the ITS for next question. ITS assigns different levels to users based on their knowledge. When user registers for ITS he can set his level to novice, intermediate or experienced. Later ITS updates user level based on his performance.

Logically the ITS will have 3 databases

1. Constraint Base: It stores set of constraints.
2. Domain Base: Contains questions at various levels and ideal solutions.
3. Student Base: Stores student knowledge like user name, level, problems solved and their responses etc.

The ITS uses constraint based modeling(CBM) to evaluate the learner’s solutions. A constraint will be in the form- “if some keyword appears in a statement then another keyword should follow it”. eg: If learner uses “join” keyword in a select statement then the select statement should also have “on” keyword, then only the query will be syntactically correct. Such constraints(around 600) are stored to check the syntax of responses. Some constraints will check for equivalent constructs. First the system checks for syntax validity and then checks whether it is a equivalent construct to the ideal solution or not.

Architecture:

The diagram 3.1 shows the architecture of SQLT-Web ITS.It contains 3 main components. They are

- Session manager
- Student model
- Pedagogical module

The functionalities of the components is explained below.

Session manager

It establishes and manages a session per user. When student enters credentials to login session manager contacts student modeler to check the existence of user and authenticates. Requests to create a new student model if student model does not exist. It records student actions and corresponding feedback in a log. Now onwards each user request is mapped to corresponding session and serves the requests by interacting with pedagogical module.

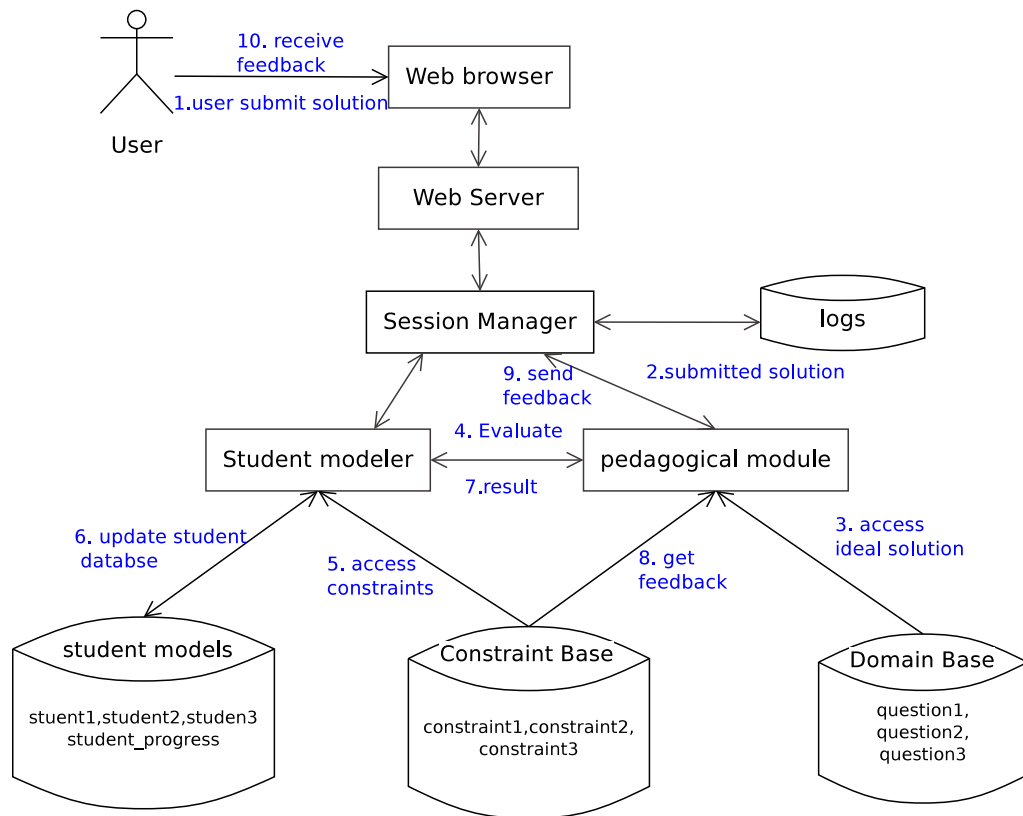


Figure 2.1: SQLT-Web ITS architecture

Student model

At the beginning of the first session with the system, the student selects the appropriate initial level novice, intermediate, or experienced. When ideal solution and user solution are provided to student model, it evaluates user solutions and updates their profiles(levels) in database accordingly by fetching constraints from constraint base. Level is incremented if student solves two or more problems consecutively within three attempts each.

Pedagogical module

It is responsible for sequencing the questions. It gets user response from session manager and sends it to student model along with ideal solution. Based on the result it displays feedback. When user requests for next question, it collects user profile from student model which contains the constraints the user answered wrongly for maximum times and brings the next question which has that constraint.

2.2 Strategies

2.2.1 Guided discovery learning

Motivation: In traditional teaching, concepts are directly explained by instructor which makes the learner passive and forces them to remember the principles. This instructor-centered teaching assumes that all learners have the same level of background knowledge in the subject matter so they can learn at same pace, which may not be the case always. So Constructivist theories, which consider the background knowledge of each learner in teaching evolved gradually and it is proved that learning through hands on activities performs better than direct transfer

of concepts[5][9]. Guided discovery is a constructivist teaching-learning strategy in which the learner involves in hands-on activities and learns through them.

Pedagogy: Diagram 2.2 shows the steps in guided discovery at higher level. It emphasizes on learning through experiments. The steps in guided discovery learning area

1. Instructor provides the instructions for doing the hands-on activities.
2. Learner performs hands-on activities, analyzes and submits results. Students gain knowledge through these activities and generalize the principles.
3. Instructor poses sequence of questions.
4. Learner answers the questions.
5. Instructor evaluates answers and provides feedback.

In our project, the student can type the commands in a pop-up window and see the results thus provides the hands-on approach.

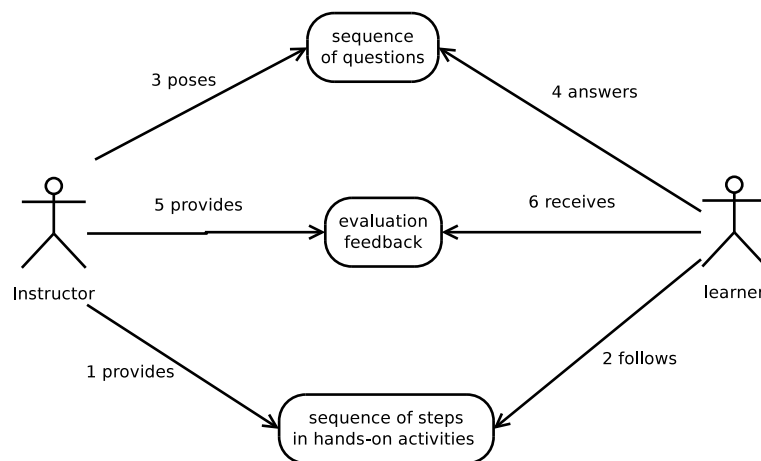


Figure 2.2: Steps in Guided Discovery Learning Strategy

2.2.2 Socartic questioning

This is explained in Vikash's[7] report.

2.2.3 Scaffolding

This is explained in Chandrapal's[16] report.

2.2.4 Game based learning

This is explained in Praveen's[6] report.

Chapter 3

ITS Design Details for Guided Discovery Learning Strategy

In this chapter we explain how ITS works for tutoring, how guided discovery learning is implemented using ITS and the sequence of steps to be followed by instructor, learner and ITS. We briefly explain the design details of our ITS framework below.

Our ITS can tutor multiple courses. Each course can have multiple topics and each topic can have multiple subtopics. Each subtopic contains questions entered by instructor. The questions in subtopics are used to teach a concept to learner. Instructor can edit the courses, create or edit topics in a course and create or edit subtopics in a topic. Instructor can create or edit questions in a subtopic. Instructor enters the dependencies between topics and subtopics such as “topic A is prerequisite for topic B”. Instructor enters pass criteria for each subtopic such as “learner should answer at least 4 testing questions in a subtopic to clear the subtopic”. This is referred as threshold value.

Learner can choose the courses, topics and subtopics. The subtopic contains questions of a specific concept. When learner selects a topic, ITS checks whether the prerequisite topics are completed prior to this or not by using the dependencies entered by instructor. At the time of selecting the subtopic also these dependencies are checked. Adaptation logic for the guided discovery strategy is executed for fetching the questions.

Adaptation logic: Adaptation logic is used to determine the next question to be displayed to user. ITS first presents all guiding questions and then displays testing questions. If user correctly answers the threshold number of testing questions then an option for reattempting the quiz or attempting the remaining testing questions is provided otherwise all testing questions in that quiz are presented to learner. Adaptation logic is implemented by sequence manager module which is present inside quiz module.

3.1 Guided discovery using ITS framework

In guided discovery learning the instructor poses the questions in proper sequence and evaluates user response. In our ITS framework, to teach a concept we create a subtopic with the concept name and store the questions to be posed and their answers to the subtopic quiz. ITS will fetch the questions in sequence, receives user responses and sends them over network for execution, if required. Below we explained the support provided by ITS to teach a concept in detail.

- The sequence of questions to be posed are stored as multiple choice questions in ITS database. Based on the purpose of the questions, we classify them as

- Guiding questions- guides student toward concept including hands on activities. To answer these questions, user can type the commands in an interactive window and observe the results.
- Assessment questions/testing questions- To assess how well a user learnt. If user answers the number of questions specified by instructor then he learnt the concept well.

3.2 Steps to be followed

It is the responsibility of the instructor to provide the sufficient input data to domain knowledge base. ITS is responsible for further user actions. Below we explain the steps to be followed by instructor, learner and ITS.

3.2.1 Steps to be followed by instructor

- Instructor logs in.
- Create/edits or selects a course.
- Creates/edits or selects a topic in chosen course.
- Enters prerequisite dependencies for created topic.
- Creates/edits or selects a subtopic in chosen topic.
- Chooses the teaching-learning strategy to teach the course.
- Enters data to create a subtopic.
- Enters prerequisite dependencies for created subtopic.
- Enters questions to create the quiz in a subtopic.
- Enters threshold value for the subtopic.
- Can view report of students.

3.2.2 Steps to be followed by learner

- Learner logs in.
- Can select a course.
- Can view his report at topic level or select a topic.
- Can view his report at sub-topic level or select a subtopic.
- Can type commands in pop-up window.
- Submits answers to questions displayed by ITS.

3.2.3 Steps to be followed by ITS

ITS provides different facilities to different to roles of users i.e. it provides distinct features to instructor and learner.

When instructor logs in

- Provides facility to create or edit the course structure(course, topics and subtopics).
- Validates the submitted data.
- Stores data to knowledge base.
- Displays requested reports.
- Checks for cycle in topic-dependency and subtopic-dependency.

When learner logs in

1. Display course.
2. Display selected report or checks for topic-dependency for selected topic and displays the result.
3. Checks for subtopic-dependency for selected subtopic and displays the result.
4. Use adaptation logic and determines whether to present guiding question or test question.
5. Display questions from the quiz.
6. Send data over network if required.
7. Evaluate learner response.
8. Update learner knowledge.
9. Display report and proceed to next subtopic.
10. Provides option for reattempt or attempt remaining questions.

Chapter 4

ITS Framework & Work Flow

ITS framework is the system developed by integrating the 4 independent systems developed by our team. The course structure would be similar in all 4 strategies. The difference between the strategies is in the sequence of presenting the questions to the learner and the method of teaching. In socratic questioning strategy, While creating the quiz instructor will enter the next question to be presented to learner per each choice chosen by user. In Scaffolding instructor enters hints and in guided discovery we have 2 types of questions. Thus implementation of quiz module is different for each strategy. Below we explain the framework and work flow in our frame work.

4.1 ITS framework architecture

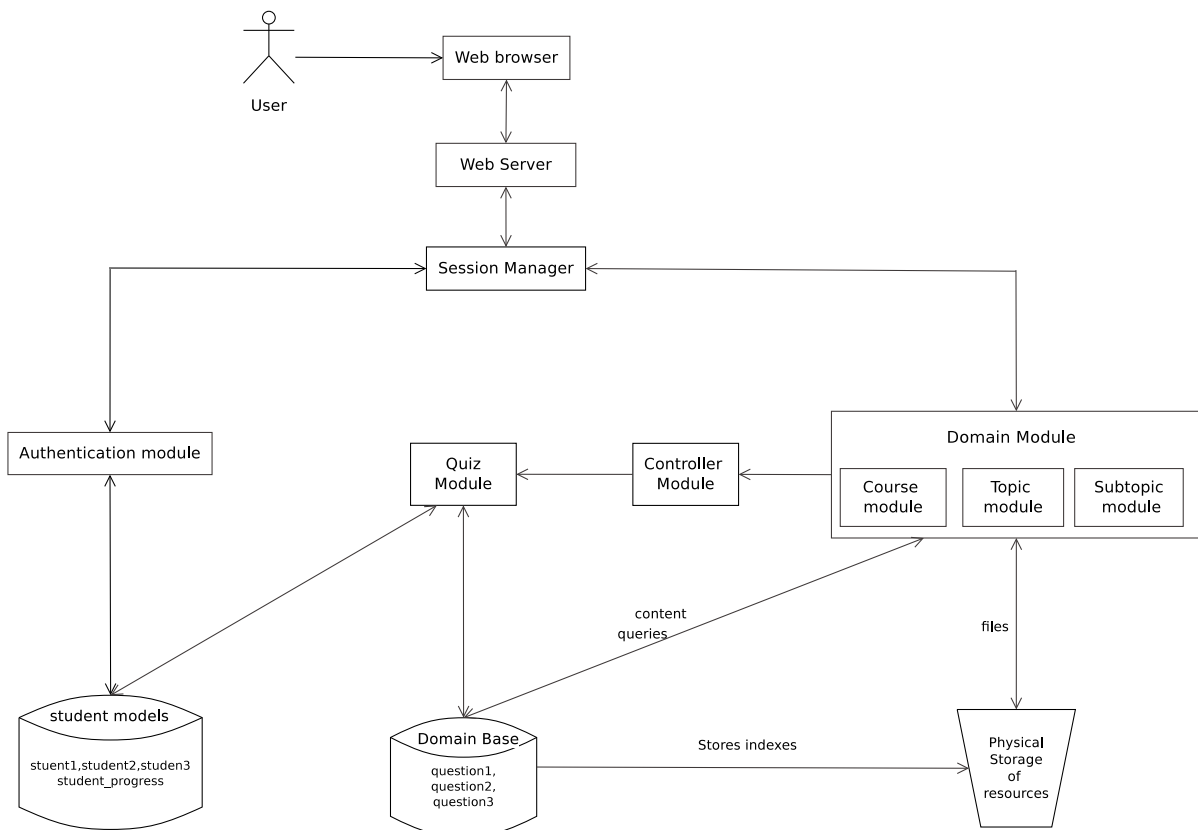


Figure 4.1: ITS framework architecture

Figure 4.1 shows the architecture of our ITS framework. Logically the database can be divided into 2 parts.

1. **Domain database:** Stores the data related to the subjects to be taught. It contains information about the courses, topics, subtopics, strategies used, questions stored in quizzes and indexes to the files uploaded by instructor. Instructor inputs the domain knowledge into this.
2. **Student database:** Stores the details of ITS users(students/instructor). It contains information like user credentials and learner progress parameters such as questions attempted by learner and their responses etc.

Portion of the system memory is allotted for storing the resources uploaded by instructor. The components of the framework work are briefly explained below. In next chapter we discuss the functionalities of each module in great detail.

- **Session manager:** It creates and maintains a session per each user upon login. On user login it performs authentication then sets the session parameters and uses these parameters to identify the session.
- **Domain module:** It provides the functionalities for entering and presenting the contents of the course to be taught. It stores the subject knowledge provided by instructor to the domain database and fetches it upon further requests.
- **Controller module:** This module finds the strategy chosen/suitable for the user and redirects to the corresponding quiz module.
- **Quiz module:** It provides the functionalities for entering questions into a quiz and presenting them to learner.

4.2 Time sequence diagram

Time Sequence Diagram For Instructor

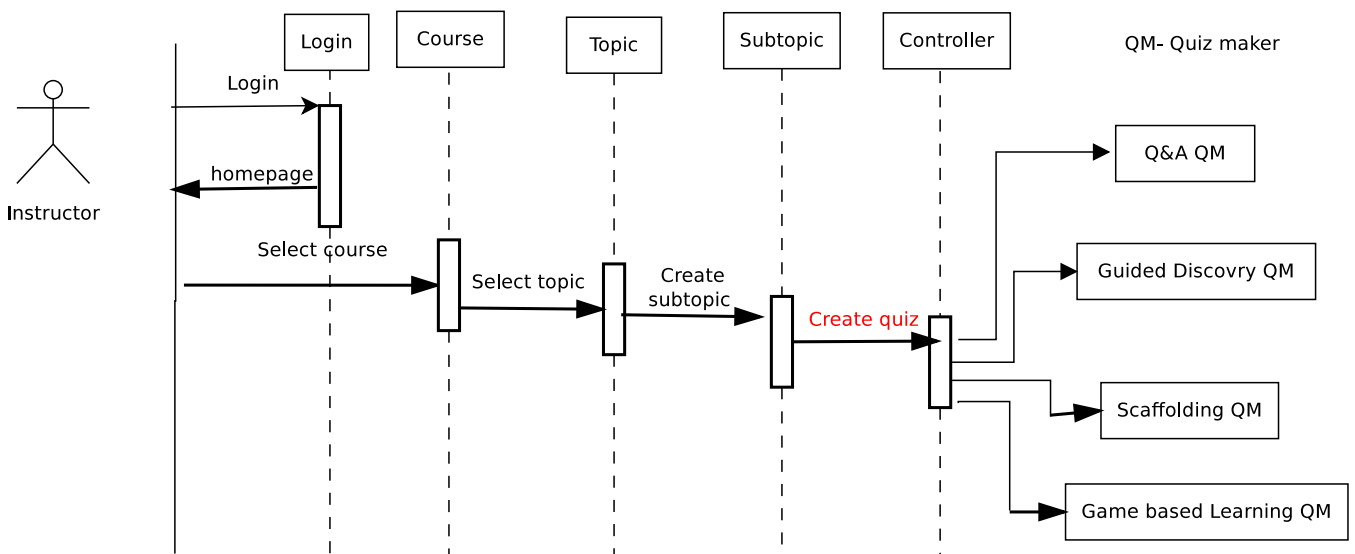


Figure 4.2: ITS framework-Handling instructor interactions

Figure 4.2 shows the time sequence of the steps for creating a quiz by instructor. The interface will be same till subtopic level but it will be different for for each strategy for quiz module. Now the instructor chooses the strategy. The controller module redirects the instructor to the corresponding quiz maker module, which renders the interface for that strategy.

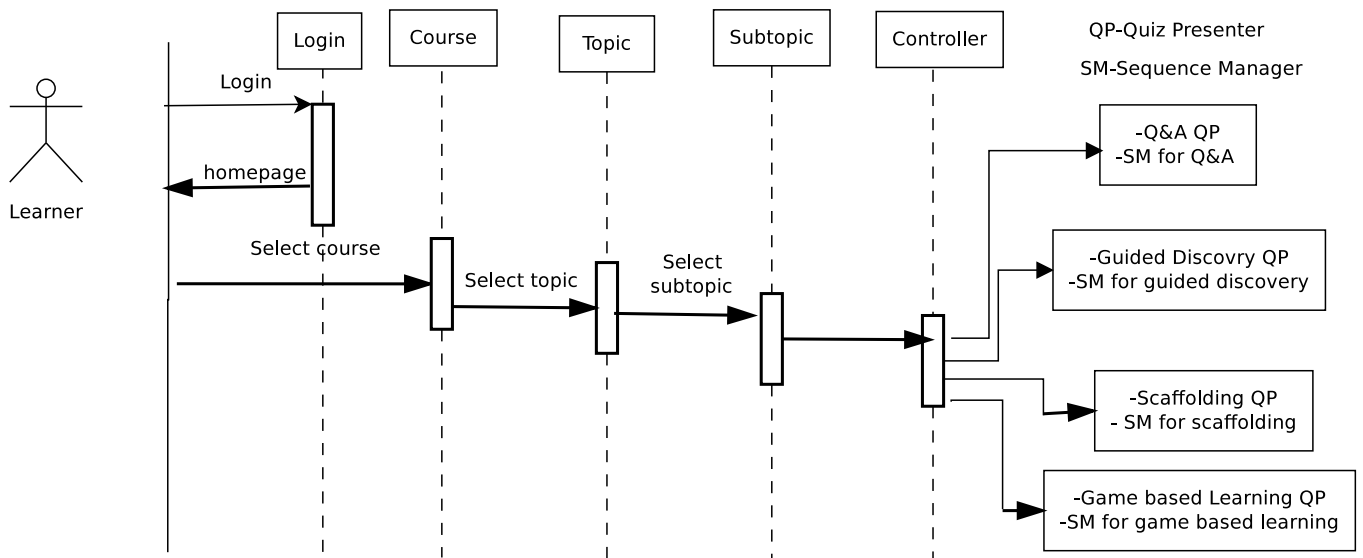


Figure 4.3: ITS framework-Handling learner interactions

Time Sequence Diagram For Learner Figure 4.3 shows the time sequence of the steps for attempting a quiz by learner. Learner selects the course, topic and subtopic during which the topic dependency and subtopic dependencies are checked. The controller module determines the proper strategy for learner and redirects to corresponding quiz module which displays questions.

Chapter 5

Implementation

We used php, html, javascript, jquery and ajax in our development. We used mysql database manager. We implemented the system through which the instructor can create hierarchal course structure. Learner can attempt quizzes and use the pop-up window for executing code snippets. ITS takes care of the ordering of the topics and subtopics. The directory structure is shown in figure 5.1. Quiz directory implementation will be different for each strategy.

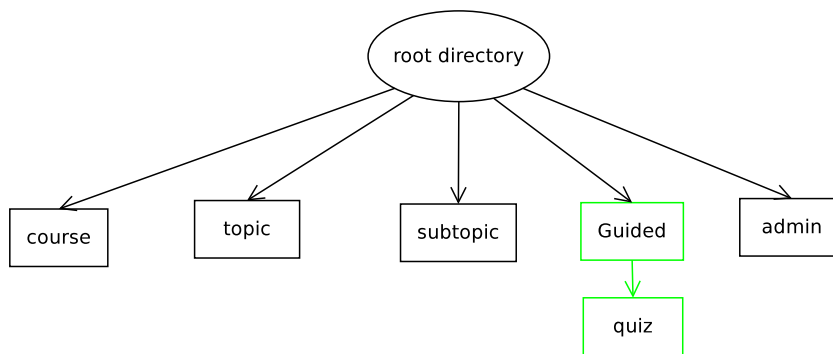


Figure 5.1: ITS directory structure

Below we explain the modules of our system, database schema details.

5.1 Description of Modules

This section explains the different modules of our ITS system, their functionality and how these modules interact with each other.

5.1.1 Login module

- Provides login functionality to users- Performs authentication by comparing user credentials with the data stored in database, identifies the type of user(administrator, instructor and learner) and redirects to corresponding home page
- Sets the session variables required for maintaining sessions. Creates session per each user.
- Access control: Session variables set at login time are used for access control.

5.1.2 Course module

- Provides all functionalities required to create, edit and delete a course.
- Interacts with input validation module to validate the user submitted input while creating or editing a course.
- Stores the valid data to database using database handling module.

5.1.3 Topic module

For instructor it provides the functionalities required to create, edit and delete a topic and stores valid data into database.

Topic Dependency Graph-Loop Detection:

The order of teaching the topics need to be maintained. Here we refer prerequisite relation as dependency i.e. if topic-A is prerequisite to topic-B then we say topic-B depends on topic-A. To maintain the proper order of teaching we store the dependencies among the topics in a graph structure and ensure that the student will attempt the topics in the same order. Whenever instructor changes the dependencies there is a possibility of getting into a cycle while storing the dependencies among the topics. For this purpose we used modified depth first graph traversal algorithm for detecting the cycle. Each time the instructor enters the dependencies this algorithm is run and checks for cycle.

Modified Depth First Graph Traversal Algorithm for Loop Detection Initially all the nodes are colored white. When a node is visited it is turned into red. We move on to descendants using depth first algorithm. When a node is visited completely it is turned into green. During the traversal, if we encounter another red node, then there is a cycle in the graph. Source code for this algorithm is shown in **appendix B**.

It implements the function `isindependent(topicid)` which returns whether the topic with `topicid` is independent or not. We defined independency as, Topic-A is independent of Topic-B iff there is no edge from Topic-A to Topic-B in topic-dependency graph or if there an edge from A to B and all the subtopics in Topic-B are cleared by the student. Topic-A is independent if it is independent of all topics in the course.

When learner selects a topic then `isindependent` function is called to perform above mentioned checks.

5.1.4 Subtopic module

- For instructor: Provides functionalities for creating, editing, deleting a subtopic.
- Receives subtopic dependencies and runs the modified DFS algorithm for cycle detection
- For Learner: `isindependentsubtopic(subtopicid)` function is implemented. It checks whether the subtopic with its id value `subtopicid` is independent of other subtopics or not. A subtopic-A is independent of subtopic-B iff there is no edge from A to B in graph or topic-B is cleared by learner prior to this. Subtopic-A is independent if it is independent of all subtopics in the chosen topic.

5.1.5 Quiz module

- For instructor: Provides the functionalities for creating, editing, deleting questions from a subtopic.

- For Learner it fetches the questions from chosen subtopic from the point where learner suspended.
- Evaluates learner responses and stores learner progress to database.
- Contains adaptation logic which fetches next question based on student progress
- As mentioned we have 2 types of questions. Quiz module provides interactive window for supporting the hands-on activities while attempting guiding questions.

Interactive Window: We implemented an interactive window using which the student can submit the C code commands and see the results and learn from it. It is possible that the students system does not have a compiler. So, the user commands are sent over the network to a remote server which echoes the response of user commands. Later we supply a series of testing questions to know how much the student learnt. We track the student responses and use this data for future teaching purpose.

```
## Pop-up interactive window generation code
<script language="javascript" type="text/javascript">
<!--
function popitup(url) {
    newwindow=window.open(url,'name','height=500,width=500');
    if (window.focus) {newwindow.focus();}
    return false;
}
// --></script>
</head><body>
<a href="index.html" onclick="return popitup('form.html')">

## execution on remote system code
exec('echo \' \'$_GET['code'].'\>temp.c'); //write user input into the file
exec('gcc temp.c'); // compile the file
echo exec('./a.out'); //suply the output
exec('rm ./a.out'); // remove the ./a.out
```

5.1.6 Administrator module

- **Student enrollment:** It handles the enrollment requests of students
- **Strategy sequencing:** Strategies are assigned priorities and questions are displayed from priority topics for the learner. It handles the configuration of priorities of strategies
- **New Strategy:** It provides facility for adding new strategies.
- **Uploading resources:** It handles uploading the resources.
- **Generate Reports:** Generates various types of student progress reports like the number of questions attempted by learner in each subtopic etc.

5.1.7 Controller module

- It gets user input and finds the suitable strategy.
- Based on the strategy redirects to corresponding strategy's suitable module.

5.2 Database

The diagram 5.2 shows the relations between tables of part of the database. Keys are shown in bold letters and the arrows indicate the foreign key relation. Below we explained the purpose the tables. Complete database schema is shown in **appendix A**.

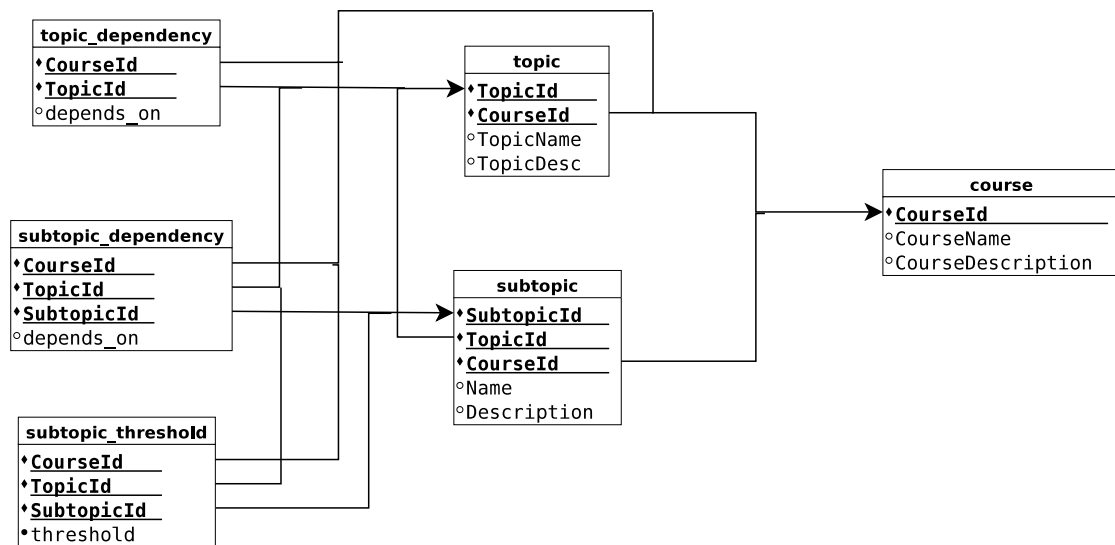


Figure 5.2: Database structure

- **login_table**: Stores the user credentials.
- **course_table** : Stores the data about courses like course name, course id and course description.
- **topic_table**:Stores the data about topics like topic name, topic id and topic description.
- **topic_dependency table**: Stores dependencies among topics. Has 2 fields- topic id and depends_on field on which the topic depends. Multiple topics in depends_on field are separated by ;.
- **subtopic_table**:Stores data about subtopic like subtopic name, subtopic id and subtopic description.
- **subtopic_dependency table**: Stores dependencies among subtopics. Has 2 fields- subtopic id and depends_on field on which the subtopic depends. Multiple subtopics in depends_on field are separated by ;.
- **subtopic_threshold table**: It stores the minimum number of questions to be answered correctly by a learner in order to progress to next subtopic.
- **question_table**: Stores the data about questions like question description, its 4 options, correct option and type of question.
- **strategy_table**: Stores information about the various strategies used in our ITS framework.
- **strategy_sequencing_table**: Stores the priorities set by instructor for the strategies.
- **student_response table**: User responses for each question are stored in this table. Stores question id, type and result.

5.3 Challenges

We are working on the development of ITS framework. ITS is multi-disciplinary area. It requires the knowledge of computer science and teaching-learning strategies. The challenges we faced during ITS development are

- Worked on the literature of ITSs to understand how they work and find the non-existing features in current ITSs.
- It is noticed that the ITSs are specific to single subject and are using single teaching strategy. The evaluation of user responses scheme restricts current ITSs to work on single subject.
- We came with multiple choice questions to make our framework subject independent.
- Studied different teaching-learning strategies and analyzed which of these can be taught using simple multiple choice questions by mapping them to software system. Finally we came up with 4 strategies namely guided discovery, socratic questioning, scaffolding and game based learning which fit into our framework.
- Thought of various alternative to map the steps like providing hands on activities in teaching strategy to software system. We decided to provide the interactive window as a solution.
- Designing the common database which suits for all our strategies is made easily. But then we had numerous changes to it, when started working on developing independent systems. So database integration has become a problem.
- Developed a common prototype and then each of developed individual systems on this prototype. We faced some problems in integration due to some conflicting interfaces defined.

Chapter 6

Integration & Testing

6.1 Integration of the system

This chapter includes combined work of all four of us. First we come up with a database design which full fills the requirements of our 4 strategies and chose proper themes to suit them. As mentioned, most of the modules are common to all our strategies. Initially we developed a basic prototype model, which allows 2 roles instructor and student. The basic functionalities like creating courses, topics, subtopics and generating reports are developed commonly. Now each of us worked on this prototype and the designed database to meet their strategy's requirements. Each strategy has its own quiz generator module and sequence manager and some other different operational features like providing feedback or interactive window etc. We developed a controller module which chooses the proper strategy and displays the corresponding page.

Controller Module: All the 4 strategies share the same interface for instructor/learner at the time of login. They will be working on same interface till subtopic level. Quiz module is different to all our approaches. It requires different interface for each strategy for entering questions. So when user forwards to quiz level then we need a program which takes the strategy chosen by user and redirects to corresponding strategy's quiz page. Controller module performs this functionality. In implementation, for instructor the file subtopiccheck.php redirects him to proper quiz page for creating quiz. For learner the page will be controller.php. So these 2 files take user input and diverse the control flows appropriately.

Adding new strategy: . It is easy integrate new strategies into our framework. All it needs is a separate quiz module to handle that strategy. Then it requires an addition of extra condition checking in subtopiccheck.php and controller.php in which we give the url of new strategy quiz module.

6.2 Tables

6.2.1 Common tables

- Login table
- Course table
- Topic table
- Subtopic table
- Strategy table

- Question table : We combined all fields required by each strategy in one table.
- Upload_file table
- Student_info table
- Student_progress table
- Student_response table

6.2.2 Non-common tables

1. Guided Discovery Strategy
 - subtopic_threshold table: It stores the threshold value for each subtopic.
 - topic_dependency :Stores the dependencies among topics
 - subtopic_dependency: Stores the dependencies among subtopics
2. Scaffolding Strategy
 - Student level
3. Socratic Questioning Strategy
 - Sequencing table
4. Tables used for integration
 - Strategy table
 - Strategy priority table

6.3 Testing

I used the following data for testing.

- Course: C Programming, Topic: Arrays, Subtopic: 1-Dimensional array
- Target audience: CS101 students

Guiding Questions: All questions will be finding the output of given code segment

1.

```
int a[4]={2,3,7},b[]={1,5};
printf("%d %d %d",a[0],a[1],a[2]);
printf("%d\t%d\t%d\n",a[1],1[a],b[1]);
```

 - (a) 0 1 2 1 1 1
 - (b) 2 3 7 3 3 5
 - (c) 2 3 7 3 garbage 5
 - (d) 2 3 7 3 0 5
2.

```
float f[5]={3,4.5,6};
char c[6]={'r','a','j'};
printf("%f %f %f",f[0],1[f],f[2]);
printf("%c %c",c[0],2[c]);
```

- (a) 3.000000 4.500000 6.000000 r j
- (b) 3 4.5 6 r j
- (c) 3.000000 garbage 6.000000 r garbage
- (d) 3 garbage 6 r garbage

3. `char str="string";
printf("%c %c %c s",str[0],str[1],str[2],str+1);`

- (a) s t r tring
- (b) s t r string
- (c) compilation error
- (d) s t r str

4. `int a[4]={1,2,3},b[8];
float f[3]={3,4.5,6},g[2]={1.1,2.2};
char c[6]={97,98},s[]="name";
printf("%d %d %d %d",sizeof(int),sizeof(a[3]),sizeof(a),sizeof(b));
printf("%d %d %d %d",sizeof(float),sizeof(f[1]),sizeof(f),sizeof(g));
printf("%d %d %d %d",sizeof(int),sizeof(c[0]),sizeof(c),sizeof(s));`

- (a) 4 4 12 32 8 8 24 16 1 1 2 4
- (b) 4 4 4 4 8 8 8 8 1 1 1 1
- (c) 4 4 12 0 8 8 24 16 1 1 2 4
- (d) 4 4 12 32 8 8 24 16 1 1 2 garbage

5. Assume all address start at 100
`int a[5]={1,2,3},b[8];
float f[6]={3,4.5,6},g[]={1.1,2.2};
char c[6]={97,98},s[]="name";
printf("%d %u %u %u %d",sizeof(int),&a[0],&a[1],&a[2],&a[1]-&a[0]);
printf("%d %u %u %u %d",sizeof(float),&f[0],&f[1],&f[2],&f[1]-&f[0]);
printf("%d %u %u %u %d",sizeof(int),&c[0],&c[1],&c[2],&c[4]-&c[0]);`

- (a) 4 100 104 108 1 8 100 108 116 1 1 100 101 102 4
- (b) 4 100 104 108 4 8 100 108 116 8 1 100 101 102 4
- (c) 4 100 100 100 1 8 100 100 100 1 1 100 100 100 4
- (d) none of the above

Testing Questions

1. An array is
 - (a) Collection of similar data types grouped under same name
 - (b) Collection of similar data types grouped under same name
 - (c) Collection of similar data types grouped under same name
 - (d) Collection of similar data types grouped under same name

2. What is the output of the following program

```
int a[4]={1,2,3,4};
for(int i=0;i<4;i++){
scanf("%d",&a[i]);
printf("%d",a[i]);
}
```

- (a) 1 2 3 4
- (b) 0 1 2 3
- (c) 0 1 3 4
- (d) none

3. 'triple' is a new inbuilt data type of size 5 byte. triple a[5] is a variable and
`printf("%d %d %u %u %d",sizeof(a[0]),sizeof(a),&a[0],a,&a[1],a[4]-a[1]);`

- (a) 5 25 100 100 105 3
- (b) 5 25 100 5 105 3
- (c) 5 25 100 100 105 15
- (d) 5 25 100 garbage 105 3

6.4 screen shots

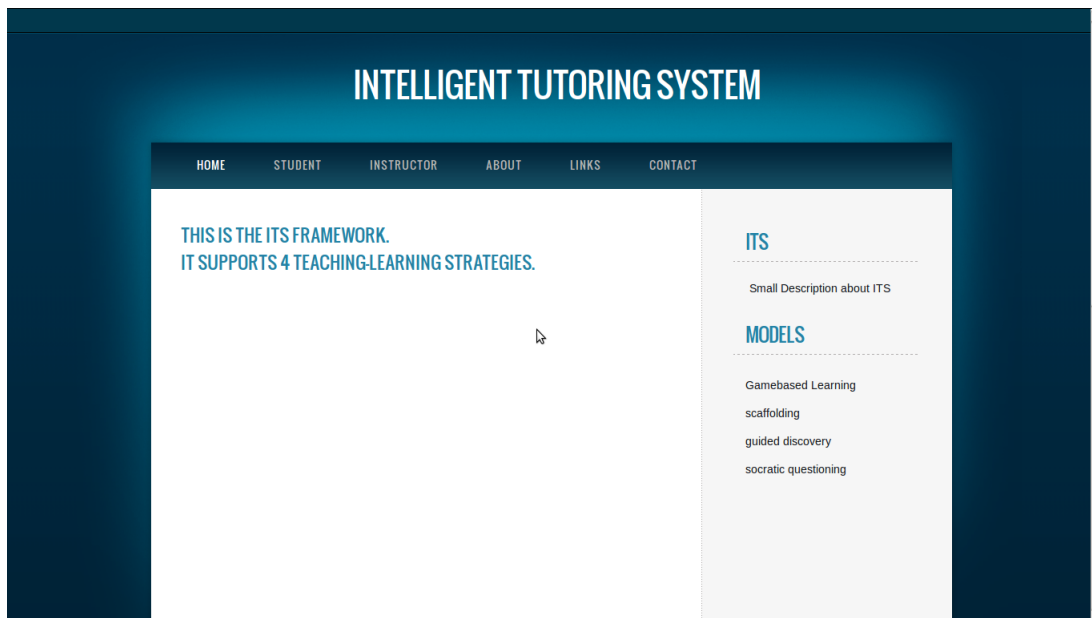


Figure 6.1: ITS home page

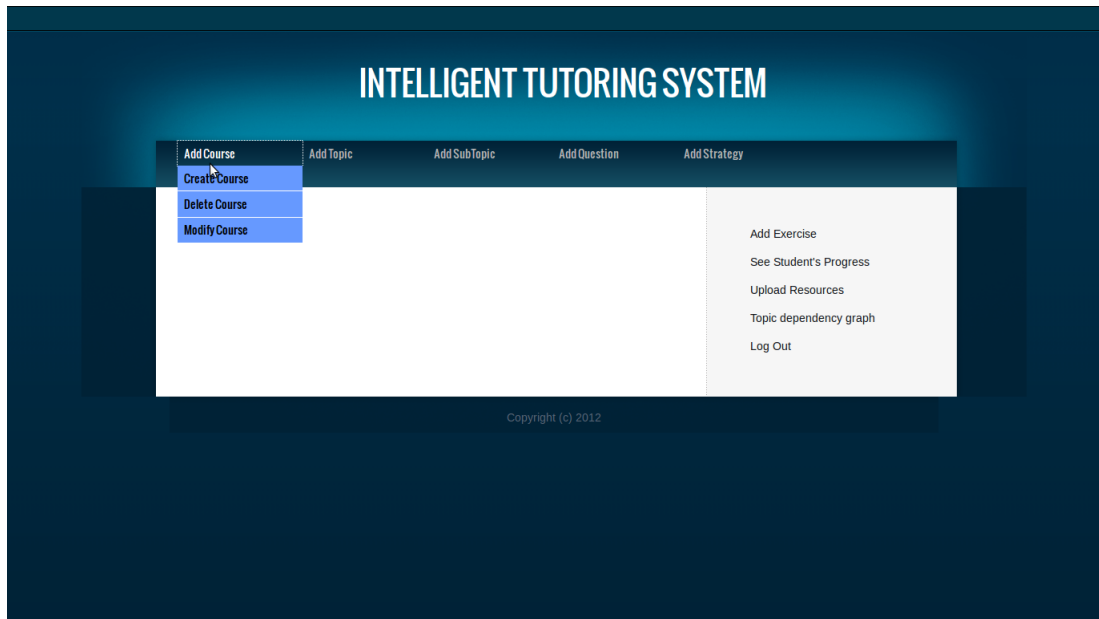


Figure 6.2: Instructor home page

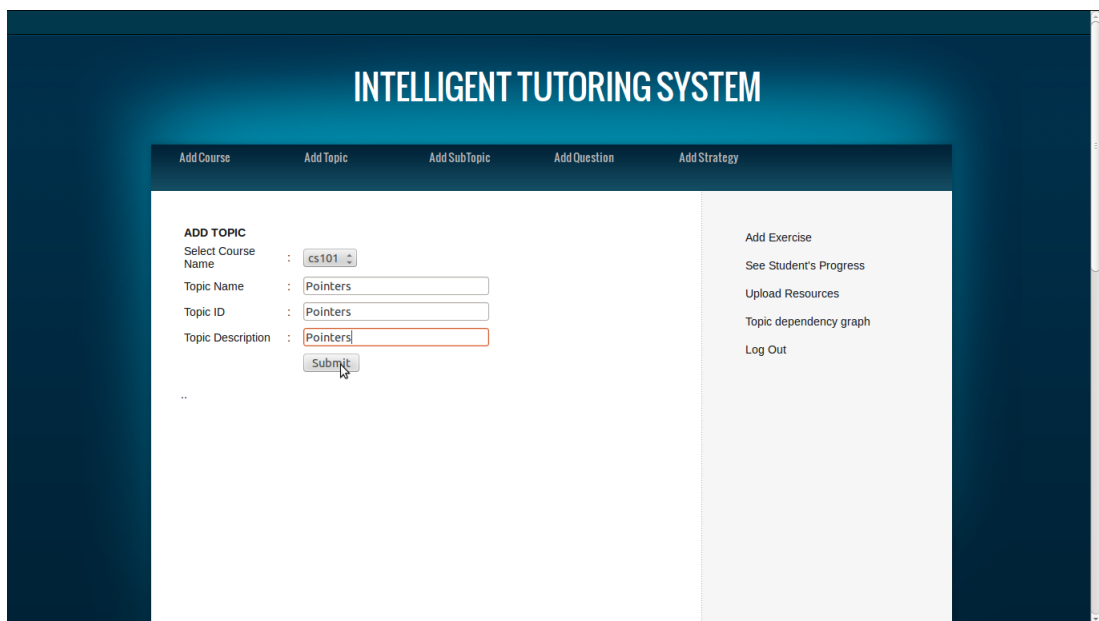


Figure 6.3: Creating a topic

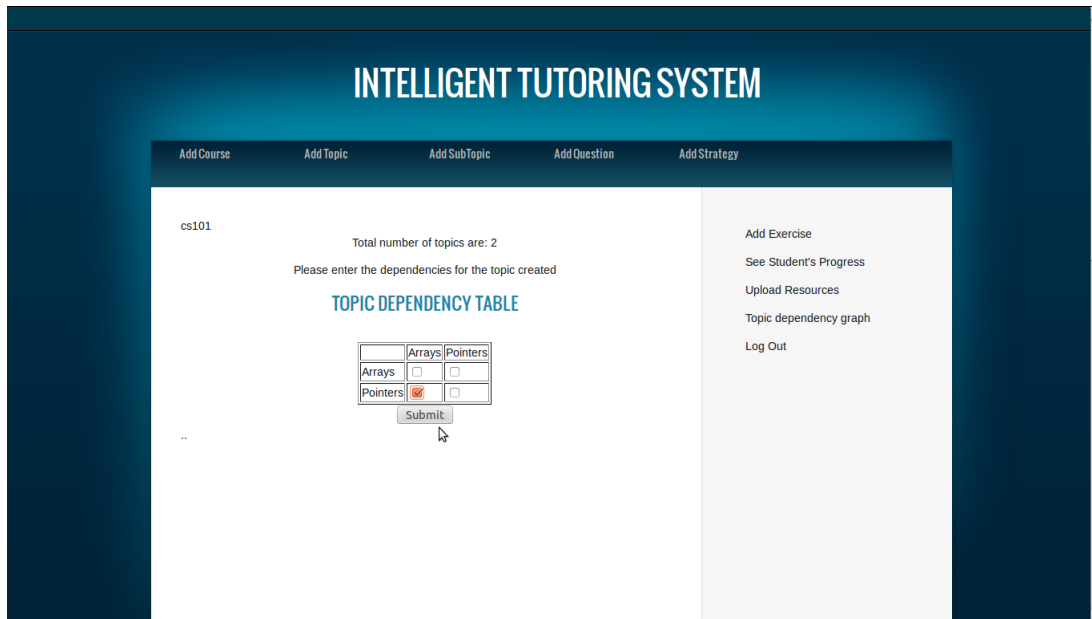


Figure 6.4: Entering topic dependencies

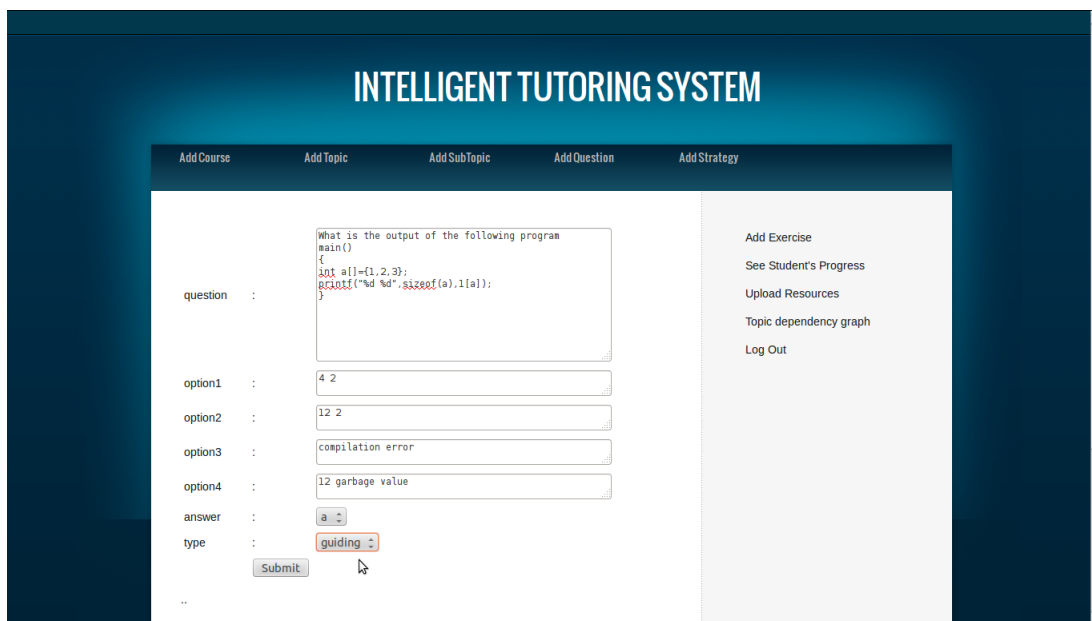


Figure 6.5: Entering question into quiz

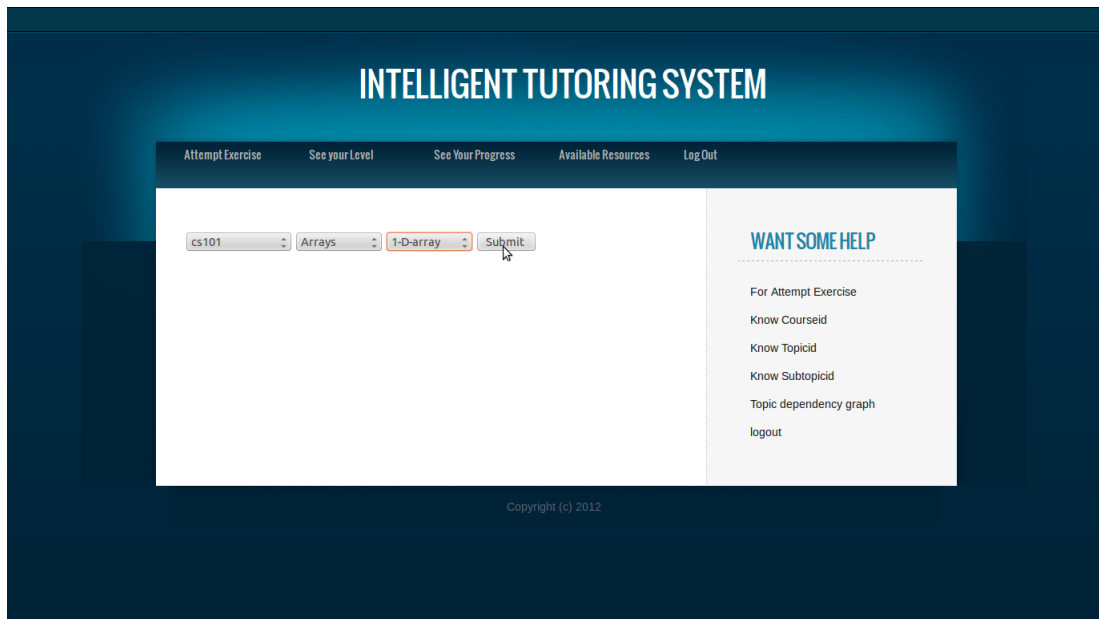


Figure 6.6: Student selecting quiz

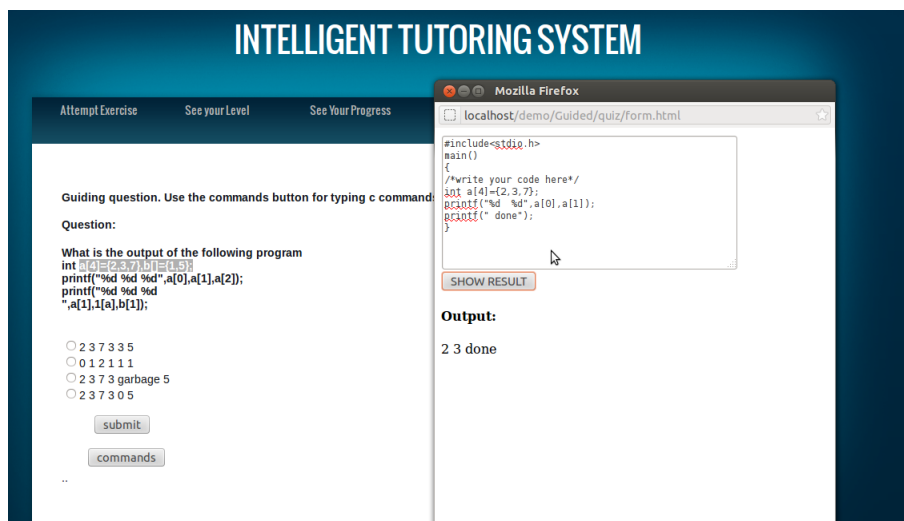


Figure 6.7: Interactive window

Chapter 7

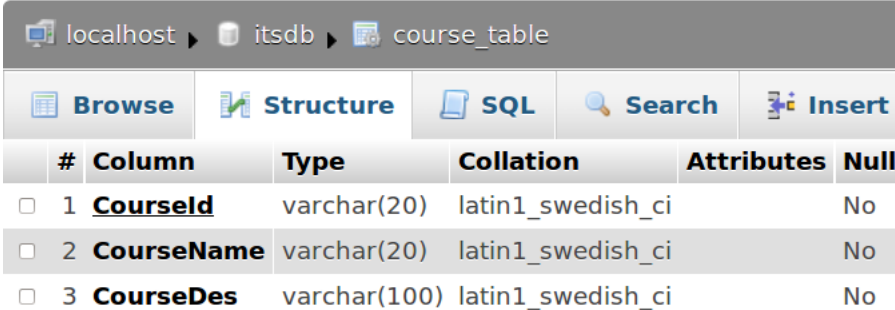
Conclusion and Future Work

We developed basic ITS framework which uses multiple choice questions and different styles of teaching. Our framework gives a provision for addition of new teaching strategies. It can be improved in the following areas.

- **Adding teaching strategies** : Current framework is working for 4 strategies. As mentioned earlier more teaching strategies can be added easily to the system.
- **Introducing subjective questions**: Current system works for MCQ questions. We can make it to work for subjective questions also. It requires some artificial intelligence or machine learning algorithms for evaluation schemes.
- **Introducing artificial intelligence**: Currently we are using a simple scheme for assigning a strategy to be used by learner. We can use complex machine learning or AI algorithms which can track the user behavior more precisely and choose the strategy based on observed patterns.
- **Response Time** : We did not considered the response times of learner. Some existing ITSs use this response time patterns in various styles to present the content to learner. One can implement a module for time response analysis.
- **Collaborative learning** : A module for collaborative learning can be developed. Addition of collaborative learning makes a way for adding some more teaching-learning strategies.

Appendix A

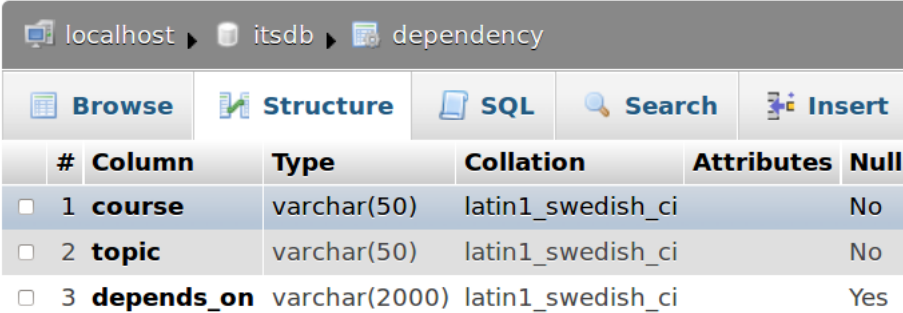
Database Schema



The screenshot shows a database management interface with the following table structure:

#	Column	Type	Collation	Attributes	Null
<input type="checkbox"/>	1 CourseId	varchar(20)	latin1_swedish_ci		No
<input type="checkbox"/>	2 CourseName	varchar(20)	latin1_swedish_ci		No
<input type="checkbox"/>	3 CourseDes	varchar(100)	latin1_swedish_ci		No

Figure A.1: course table



The screenshot shows a database management interface with the following table structure:

#	Column	Type	Collation	Attributes	Null
<input type="checkbox"/>	1 course	varchar(50)	latin1_swedish_ci		No
<input type="checkbox"/>	2 topic	varchar(50)	latin1_swedish_ci		No
<input type="checkbox"/>	3 depends_on	varchar(2000)	latin1_swedish_ci		Yes

Figure A.2: topic dependency table

#	Column	Type	Collation	Attributes	Null
<input type="checkbox"/> 1	name	varchar(50)	latin1_swedish_ci		No
<input type="checkbox"/> 2	password	varchar(20)	latin1_swedish_ci		No
<input type="checkbox"/> 3	login_id	varchar(50)	latin1_swedish_ci		No
<input type="checkbox"/> 4	type	varchar(20)	latin1_swedish_ci		No

Figure A.3: login table

#	Column	Type	Collation	Attributes	Null
<input type="checkbox"/> 1	QuestionId	int(50)			No
<input type="checkbox"/> 2	StrategyId	varchar(20)	latin1_swedish_ci		No
<input type="checkbox"/> 3	CourseId	varchar(20)	latin1_swedish_ci		No
<input type="checkbox"/> 4	TopicId	varchar(20)	latin1_swedish_ci		No
<input type="checkbox"/> 5	SubtopicId	varchar(20)	latin1_swedish_ci		No
<input type="checkbox"/> 6	QuestionDes	varchar(500)	latin1_swedish_ci		No
<input type="checkbox"/> 7	Option1	varchar(400)	latin1_swedish_ci		No
<input type="checkbox"/> 8	Option2	varchar(400)	latin1_swedish_ci		No
<input type="checkbox"/> 9	Option3	varchar(400)	latin1_swedish_ci		No
<input type="checkbox"/> 10	Option4	varchar(400)	latin1_swedish_ci		No
<input type="checkbox"/> 11	CorrectAns	varchar(20)	latin1_swedish_ci		No
<input type="checkbox"/> 12	hint1	varchar(50)	latin1_swedish_ci		No
<input type="checkbox"/> 13	hint2	varchar(50)	latin1_swedish_ci		No
<input type="checkbox"/> 14	hint3	varchar(50)	latin1_swedish_ci		No
<input type="checkbox"/> 15	type	varchar(25)	latin1_swedish_ci		No

Figure A.4: question table

#	Column	Type	Collation	Attributes	Null
<input type="checkbox"/> 1	course_id	varchar(50)	latin1_swedish_ci		No
<input type="checkbox"/> 2	topic_id	varchar(50)	latin1_swedish_ci		No
<input type="checkbox"/> 3	file_name	varchar(50)	latin1_swedish_ci		No
<input type="checkbox"/> 4	file_url	varchar(50)	latin1_swedish_ci		No

Figure A.5: upload file table

localhost ▶ itsdb ▶ strategy_sequencing_table

Browse Structure SQL Search Insert

#	Column	Type	Collation	Attributes	Null
<input type="checkbox"/>	1 CourseId	varchar(10)	latin1_swedish_ci		No
<input type="checkbox"/>	2 TopicId	varchar(10)	latin1_swedish_ci		No
<input type="checkbox"/>	3 SubTopicId	varchar(30)	latin1_swedish_ci		No
<input type="checkbox"/>	4 s1	int(10)			No
<input type="checkbox"/>	5 s2	int(10)			No
<input type="checkbox"/>	6 s3	int(10)			No
<input type="checkbox"/>	7 s4	int(10)			No

Figure A.6: strategy sequence table

localhost ▶ itsdb ▶ strategy_table

Browse Structure SQL Search Insert

#	Column	Type	Collation	Attributes	Null
<input type="checkbox"/>	1 StrategyName	varchar(50)	latin1_swedish_ci		No
<input type="checkbox"/>	2 StrategyId	varchar(20)	latin1_swedish_ci		No
<input type="checkbox"/>	3 StrategyDesc	varchar(25)	latin1_swedish_ci		Yes

Figure A.7: strategy table

localhost ▶ itsdb ▶ student_response_table

Browse Structure SQL Search Insert

#	Column	Type	Collation	Attributes	Null
<input type="checkbox"/>	1 StudentId	varchar(15)	latin1_swedish_ci		No
<input type="checkbox"/>	2 StrategyId	varchar(10)	latin1_swedish_ci		No
<input type="checkbox"/>	3 CourseId	varchar(10)	latin1_swedish_ci		No
<input type="checkbox"/>	4 TopicId	varchar(10)	latin1_swedish_ci		No
<input type="checkbox"/>	5 SubtopicId	varchar(10)	latin1_swedish_ci		No
<input type="checkbox"/>	6 QuestionId	int(25)			No
<input type="checkbox"/>	7 Hint_used	int(10)			No
<input type="checkbox"/>	8 Marks_obtained	int(11)			No

Figure A.8: student response table

localhost ▶ itsdb ▶ subtopic_dependency

Browse Structure SQL Search Insert

#	Column	Type	Collation	Attributes	Null
<input type="checkbox"/> 1	course	varchar(50)	latin1_swedish_ci		No
<input type="checkbox"/> 2	topic	varchar(50)	latin1_swedish_ci		No
<input type="checkbox"/> 3	subtopic	varchar(50)	latin1_swedish_ci		No
<input type="checkbox"/> 4	depends_on	varchar(2000)	latin1_swedish_ci		Yes

Figure A.9: subtopic dependency table

localhost ▶ itsdb ▶ subtopic_table

Browse Structure SQL Search Insert

#	Column	Type	Collation	Attributes	Null
<input type="checkbox"/> 1	<u>CourseId</u>	varchar(20)	latin1_swedish_ci		No
<input type="checkbox"/> 2	<u>TopicId</u>	varchar(20)	latin1_swedish_ci		No
<input type="checkbox"/> 3	<u>SubtopicId</u>	varchar(20)	latin1_swedish_ci		No
<input type="checkbox"/> 4	SubtopicName	varchar(50)	latin1_swedish_ci		No
<input type="checkbox"/> 5	SubtopicDes	varchar(100)	latin1_swedish_ci		Yes

Figure A.10: subtopic table

localhost ▶ itsdb ▶ topic_table

Browse Structure SQL Search Insert

#	Column	Type	Collation	Attributes	Null
<input type="checkbox"/> 1	<u>CourseId</u>	varchar(20)	latin1_swedish_ci		No
<input type="checkbox"/> 2	<u>TopicId</u>	varchar(20)	latin1_swedish_ci		No
<input type="checkbox"/> 3	TopicName	varchar(20)	latin1_swedish_ci		No
<input type="checkbox"/> 4	TopicDes	varchar(100)	latin1_swedish_ci		Yes

Figure A.11: topic table

Appendix B

Source Code for Modified DFS Algorithm

```
$id= mysql_result($result,0,"id"); #if cycle=1 then cycle is present in given graph
$arr=array($id),$full=array(),$full_count=0,$i=1,$cycle=0;
while ($full_count<$num && $cycle==0) {
    if(array_diff($arr,$full)){
        $query2='SELECT depends_on FROM dependency where topic=\'\'.'$id.'\'\'';
        $stored_data=mysql_query($query2) or die('select.. query failed');
        $stored=mysql_result($stored_data,0,"depends_on");
        $var=explode(" ", $stored);
        if($var[0]=="") {
            $full[$full_count++]=$id;
            for($m = (count($arr)-1) ; $m>=0 ;$m--) {
                if(!in_array($arr[$m], $full)) {
                    $id=$arr[$m];break;
                }
            }
        }
    }
    else {
        for($k=0;$k<count($var);$k++) {
            if(!in_array($var[$k],$full)) {
                if(in_array($var[$k],$arr)) {
                    $cycle=1; #cycle detected
                    break;
                }
                else {
                    $arr[$i++]=$var[$k];
                    $id=$var[$k]; break;
                }
            }
        }
    }
    if($k==count($var)) {
        $full[$full_count++]=$id; ##now select the next node to be visit
        for($m= (count($arr)-1) ; $m>=0 ;$m--) {
            if(!in_array($arr[$m], $full)) {
                $id=$arr[$m]; break;
            }
        }
    }
} } } } }
```

Bibliography

- [1] Wikipedia. http://en.wikipedia.org/wiki/Intelligent_tutoring_system.
- [2] Patrick Chipman, Andrew Olney, and Arthur C. Graesser. In *The Autotutor3 Architecture*, University of Memphis, USA.
- [3] Albert T. Corbett, Kenneth R. Koedinger, and John R. Anderson. *Handbook of Human-Computer Interaction*, chapter 37. USA, 1997.
- [4] Farhad Soleimanian Gharehchopogh and Zeynab Abbasi Khalifelu. Using intelligent tutoring systems in instruction and education. In *2nd International Conference on Education and Management Technology*, volume 13, Singapore, 2011.
- [5] Jong Suk Kim. *The Effects of a Constructivist Teaching Approach on Student Academic Achievement, Self-concept, and Learning Strategie*. Asia Pacific Education Review by Education Research Institute, Chungnam National University, Korea.
- [6] Praveen Kumar. Development of an intelligent tutoring system framework for game based learning. M.Tech thesis, IIT Bombay, 2012.
- [7] Vikash Kumar. Development of an intelligent tutoring system framework for socratic questioning. M.Tech thesis, IIT Bombay, 2012.
- [8] Brown Ann L. and Campione Joseph C. *Classroom lessons: Integrating cognitive theory and classroom practice*, chapter 9, pages 229–241. Cambridge, MIT Press, 1994.
- [9] Tom Lord, Holly Travis, Brandi Magill, and Lori King. *Comparing Student-Centered and Teacher-Centered Instruction in College Biology Labs*. Indiana University of Pennsylvania, Indiana, 2005.
- [10] Richard E. Mayer. *Should There Be a Three-Strikes Rule Against Pure Discovery Learning?* University of California, Santa Barbara.
- [11] Erica Melis and J rg Siekmann. In *ActiveMath: An Intelligent Tutoring System for Mathematics*, German Research Institute for Artificial Intelligence, Germany.
- [12] Antonija Mitrovic. An intelligent sql tutor on the web. University of Canterbury, New Zealand, 2003.
- [13] Muska Mosston and Sara Ashworth. *Teaching Physical Education*. 2008.
- [14] Tom Murray. Authoring intelligent tutoring systems: An analysis of the state of the art. In *International Journal of Artificial Intelligence in Education*, Computer Science Dept., University of Massachusetts, 1999.
- [15] National Extension Water Outreach Education. *Explanation of Teaching Continuum*.

- [16] Chandra pal Singh. Development of an intelligent tutoring system framework for scaffolding. M.Tech thesis, IIT Bombay, 2012.
- [17] L Jean Piaget. *To Understand Is To Invent. The Future of Education*. Grossman publishers., NEW YORK, 1976.
- [18] Michael J. Prince and Richard M. Felder. *Inductive teaching and learning methods:Definitions, comparisons and research bases*.
- [19] Catherine J. Rezak. In *Improving Corporate Training Results with Discovery Learning Methodology*.
- [20] Valerie J. Shute and Joseph Psotka. Intelligent tutoring systems present, past and future. 1994.

Acknowledgment

I take this opportunity to express my sincere gratitude towards my guide ***Prof. Sridhar Iyer*** for his constant support, motivation and guidance. Without his deep insight into this domain and his valuable time for this project, it would not have been possible for me to move ahead properly. He has been remarkable in his attempt to keep me motivated in this project and has always tried to improve me with proper feedback.