Session 2247

Multimedia Courseware For Customized Learning Environment

Michael Khader
New Jersey Institute of Technology

Abstract

Presenting technical materials in distance learning format is a challenging task for educators. Many technical type courses are offered in the familiar asynchronous distance learning format, which has been available for years. Fewer courses, however, are offered in the synchronous distance learning format in which the traditional classroom settings are extended beyond the existing confines of the walls of a campus to distant sites. Assessment of synchronous and asynchronous formats is underway. Until now samples from these distinct modes of deliver are statistically limited and more experiments are needed to prove how viable these options of learning are. We believe the jury is still out on the effectiveness each individual mode of delivery. We present an alternative experiment that builds on the strength found in each of the synchronous and asynchronous modes.

Introduction

NJIT has a strong distance learning program that is being active for the past ten years. Within the program there are a number of developments under way including the virtual classroom, computer mediated learning, and the multimedia interactive lab-courseware. The focus of the project being presented is the integration of both the synchronous and asynchronous modes to form a new learning environment that can be customized o fit the need of each individual learner while maintaining the integrity of the learning experience.

The courses selected for this experience are of technical nature. Course selection criteria in this project include a hands-on component to explore the problems associated with offering lab-based courses in distance learning format [1]. One of the courses selected is a C/C++ application programming offered to junior level students in the Electrical Engineering Technology program. Although course includes an extensive set of project based hands-on work, it has a substantial theoretical component including algorithms and performance analysis. Currently, the course is offered in traditional setting in which the allotted time is divided into 2-hour lecture and 2-hour lab. It will be offered in a distance learning format in the spring of 1996 using both modes of delivery (synchronous and asynchronous)

The infrastructure over which this course is delivered includes both the gateway networks and the INTERNET. The gateway networks are used solely for the synchronous part of the experiment and the INTERNET is used in both the synchronous and asynchronous modality. The gateway network consists of a T1 interface and facility, and a collection of multimedia hardware presentation tools. The T1 facility interconnects two campuses of New Jersey Institute of Technology -- the Newark campus and the Technology and Engineering Center - a southern branch campus at Mount Laurel New Jersey. T1 is a telecommunications system that carries 24 channels each running at 64 kbps of information. Aggregate of these channels provides the bandwidth for delivering the two-way communications between the two campuses. The information may take the format of still images from the document reader, images of the classrooms including students and instructors, and computer mediated communications. These various forms of communications are bundled (simultaneously) into a 384 kbps (6 x 64 kbps) pipe and thus providing for a total interactive two way communications including video, audio and data.

The INTERNET serves as a second option for those learners who for whatever reasons prefer the asynchronous format. A multimedia-authoring tool is being used for courseware development including illustrative animated objects, interactive performance level measurement. Evaluation of video and audio interfaces over the INTERNET is underway. If it is proved that the benefits of interactive video and audio outweigh its performance drawbacks, it will be integrated into the courseware. The focus is on the contents being delivered and keeping distractions generated from the tools used at a minimum. The idea is to provide a learning alternatives while maintaining the required high standards.

Course Contents

The course encompasses two major components: the theoretical part including analysis of algorithms, data structures, and the fundamentals of the language -- syntactically and lexically, and hands-on components that includes practical performance of algorithms and data structure. Problem solving approaches are emphasized in both the practical and theoretical components. Students are given a practical project that requires most of the materials covered in class and materials outside the class that need some undergraduate level research. The laboratory assignments provide for building blocks that can be reused in the project. This structure of class materials forces the concept of modular design in applications development.

Presentation Tools

As previously stated, the course delivery includes both the synchronous and asynchronous models of distance learning that are integrated into a total environment allowing learner a choice of attending a classroom setting or a self past study. The materials for both the in-class and out of class are the same. The in-class delivery uses the T1 interface to connect the two distant sections of the course. The tools used are video conferencing, still image presentation and data collaborative tools that allows learners at both sites to share programs, make notes and annotations and exchange views with the instructor at the remote site in real time. Additionally, a combination of shared power point presentations and in-class analysis of algorithms and programs using a computer output that is projected to both sites simultaneously are employed. A document reader is used for free hands writing using plain papers when necessary.

Out-of-class learners use the INTERNET to follow the in-class pace or study on their own pace. MULTIMEDIA TOOLS BOOK II is used to develop the courseware in HTML formats. The materials are prepared in the form of a book with chapters and table of contents for each chapter. Hot links are provided to navigate back and fourth with the material. Each section(s) related to a topic is followed by interactive questions and answers session. If the learner enters the wrong answer, a notification is provided with a prompt to try again or use the hot links until the correct answer is given. The hot links provide an easy navigation tool that adds efficiency during a learning session in that the time spent to go back and fourth between topics is minimized and the focus is on the course material. Appendix B shows examples of “BOOK on C” home page on the INTERNET accessible by all learners enrolled in the class.. Additionally all power points presentations made in class are made available on the INTRENET. All MIDTERMS and FINAL EXAMS will be proctored and must be attended by both in-class and out-class students.

WEB Authoring

Initially we opted to use the MULTIMEDIA TOOLBOOK II for web authoring. It is a powerful package for generating and administering computer-based education courseware. Video, audio, and animated objects can be easily incorporated with less time invested in the mechanics of the development of the courseware and with more time devoted to the contents. We deferred incorporation of the video and audio object for the next phase of the project. We used a hybrid courseware that uses both the latest MS Power Point and the Multimedia Tool Book II. By incorporating an HTML filter to convert the course material to format that is suitable for the web. Shown below is an example of material prepared in frame-format as part of the numerical analysis home page.

EXAMPLE

EET 344 Lecture 3 Outline

- Program Control
- Relation Operators
- The Open Branch
- The Closed Branch
- The IF, IF-ELSE, and IF-ELSE-IF Construct
- The Switch Statement
- Examples
- Debugging Your Program : if we have time!

Lecture 3- Relational Operators

- Relational operators are symbols that indicate a relationship between two quantities.
- The two quantities can be any of
    - variables, constants, functions, and valid C expressions
- Relational Operators Symbols
    - > : greater than, 10 > 7, 3> 6
    - >= : greater than or equal to, A = B = 20, A >= B, A >= 25
    - < : less than, 7 < 10, A = 2, B =7, B < A
    - <= : less than or equal to, 3 <= 15
    - = = : equal to, 3+5 = = 2*4, 3+5 = = 3*4
    - != : not equal to, 3-1 != 2*4, 3-1 ! = 2

Lecture 3 - Example on evaluating relational expressions

- #include <stdio.h>
    - main()
    - {
    - int logic_value;
    - printf("This program prints the logic values of the following relations"):
    - logic_value = (4 > 5);
    - printf("\n(4 > 5) is %d\n", logic_value);
    - printf("\n5 > 3) is %d\n", logic_value = (5 > 3));
    - printf("\n(15 >= 3*5) is %d\n", logic_value = (15 >= 3*5));
    - }
    - Note relational and logical expressions evaluate to either TRUE or FALS (1 or 0)

If - Statement

- Construction: if (expression) statement : expression is a valid C expression and statement is a single C statement or a group of statements in braces
- example_1:

```
int exam_mark;
printf("enter mark:");
scanf("%d", &exam_mark);
if (exam_mark >= 90)
printf("you got a A");
```

If Statement -- Continue

- example_2

```
char c;
if ( c = getchar() == 'q')
{
printf("Are you sure you want to quit, Y or N?"); ");
c = getchar()
if ( c == 'Y');
{
cleanup();
exit(0);
}
}
```

The IF-Else Statement

- Construction:

```
if ( expression)
statement 1
else
statement 2
```

Again the expression is any valid C expression and statement1 statement 2 can be single or group in braces

#include <stdio.h>

- #include <stdio.h>

```
#define PI 3.14159
main()
{
int selection ;
float radius;
printf("This program computes the are of a circle, square\n\n");
printf("(1) Compute area of circle\n")
printf("(2) Compute area of square\n\n\n");
printf("Selection:")
scanf("%d", &selection);
if(selection == 1)
{
printf("give me the radius of the circle:");
```

PPT Slide

```
scanf ("%f", &radius);
printf("The area of the circle is %f", PI*radius*raduis);
}
else
{
printf("give me the length of one side of the square");
scanf("%f", &length);
printf("The area of the square is %f", length*length);
}
}
```

The IF-ELSE IF Statement

- Construction

```
if (expression 1)
statement 1
else if (expression 2)
statement2
else if (expression 3)
statement 3
......
else if (expression n)
```

n is as large as your system allows it

If-Else-If statement, a program to map exam marks to letter grades

```
#include <stdio.h>
main()
{
int exam_mark;
printf("enter mark:");
scanf("%d",&exam_mark);
if(exam_mark >= 90)
printf("you got an A");
else if (exam_mark >= 80)
printf("you got a B");
else if (exam_mark >= 70)
printf("you got C");
else if(exam_mark >= 60)
printf("you bone head got a D");
else if (exam_mark <= 59 )
printf("you should be expelled from school");
}
```

Similar to if-else-if : the C switch statement

- Construction

```
switch(expression)

{
case constant_expression_1: statement
case constant_expression_2: statement
case constant_expression_3: statement
….
case constant_expression_n: statement
default: statement
}
```

Practical Performance of Algorithms and Data Structures - A Class Project

The study of data structures, algorithms, and their performance has been an essential part of the computer science and engineering curriculum for a long time. Analysis of algorithms and abstract data types, however, is typically limited to their theoretical behavior. Students are introduced to the upper bounds, lower bounds, right bounds and the rules used to estimate the running times of language constructs. Students are then asked to estimate worst-case, best-case and the average-case running time of a well known algorithms and small pieces of codes written in a Pascal or a C-like language.

The effect of using different data structures, languages, compilers, memory hierarchies, cache sizes, page sizes, and CPU’s on the performance of an implementation are completely ignored. From an implement’s point of view, all these factors play key roles in determining the performance of a piece of software. Another point that is often missed while estimating the performance of an algorithm is its interaction with the memory management subsystem. Performance of an algorithm that frequently interacts with the memory management subsystem cannot be estimated correctly because the performance of the memory management subsystem is dependent on memory loading of the computer executing the piece of software at hand. Therefor the best way to measure the performance of an algorithm is to implement it and run it on different platforms under varying memory loading by using different compilers, collect actual running times, and find polynomials that best fit the collected times under different conditions. The practical aspects and experimentation, however, are overlooked in traditional algorithms and programming courses. Presented in this paper is a laboratory project that is being used in the C/C++ numerical analysis using C/C++ course. In this application performance measurement project students are asked to do the following:

1) Implement the simple matrix multiply algorithm on the following platforms:
    a) A i80486DX66-based PC
    b) A SUN SPARC workstation available in the SUN lab
2) Execute the implementation for two N x N matrices of integers, for N = 0,25, 50,....,350, and collect the execution times for all runs for non-optimized and optimized codes generated by the compilers at hand. Repeat each run at least three times, every times with new matrix, and compute the average of the three.
3) The running time, T(N), for the algorithm is O(), i.e., T(N) K for two constants K and , when N . Compute the average of K for both platforms and for both optimized and non-optimized codes.

Analysis:

A careful analysis of the requirements will identify a number of software resources that are needed to effectively perform the exercises identified in the project.
1) The implementation will use three two-dimensional arrays, two for storing input/ matrices and one to store the final result. The C code will look like

```
for(i=0; i<MAX;i++)
        for(j=0; j<MAX;j++)
                for(k=0; k<MAX;k++)
                {
                        Result[i][j]+=Matrix1[j][k]
                        *Matrix2[k][j].
```

2) The project requires the ability to randomly generate integer matrices. This can be accomplished by using a random number generator available on the host operating system. In UNIX environment one may use the library routine rand().
3) The project requires determining the average execution time. Using a start-stop clock that can be started, stopped and displayed will suffice for this task on the 486 PC platform. Using the shell command that displays the time taken to execute the process will work for the UNIX platform. Regardless of the method, it is important to make sure that the time displayed is for execution of the given piece of code only.
4) The data generation time should not be charged to the algorithm and thus a mechanism to separate the two times must be devised. Implementing a driver to generate the random matrices as a separate piece of code, appending the output of this piece of code into a header file to populate the data arrays used for the matrices, then including the header file

with the source code for the multiplication algorithm provides a clean way to separate the two times, generation time and multiplication time.

An implementation of the multiplication algorithm was executed and analyzed on the 486-PC and the SPARC station platforms. The algorithm was also run three times on each platform with optimized and non-optimized codes. Every run used newly generated input matrices of random integers. Table I show a sample of running times for both platforms for optimized and non-optimized code. Clearly the data presented in the table shows that the i486DX66 outperform the Sun SPARC. One explanation for the SPARC performance is that its hardware is optimized for floating point arithmetic and performs integer operations poorly.

TABLE I
Sample Execution Times (seconds) of I486 and SPARC platforms

Matrix Size (N)	Non-optimized i486DX66	Non-optimized Sun SPARC	Optimized i486DX66	Optimized Sun SPARC
0	0.00	0.00	0.00	0.00
50	0.20	0.60	0.08	0.30
100	1.55	4.60	0.60	2.20
150	5.16	14.51	2.01	7.11
200	12.17	34.75	4.76	15.19
250	23.75	65.64	9.23	31.44
300	41.00	112.73	15.93	50.19
350	65.21	187.49	25.30	80.37

This project has been used in the traditional offering of the C/C++ numerical analysis programming course. We are using it in the distance learning environment as well. The project can be implemented individually. Students are not required to use a computing environment with a particular speed as long they document accurately the speed of the machines used in the projects. Students may opt to compare two PC platforms. If they wish to use the SPARC station, the can do so by easily access the SUN lab remotely. The additional requirement for this project is for students to plot the performance curves of each platform used for the experimentation, and put the results on the INTERNET to make them available to other students in the same section of the class or in other sections as well. In this way discussions are stimulated and different implementations can be further compared. The goal is by the end of the course, a tabulated performance information on a variety of platforms with different speed and computing environment architecture be available as a kind of simple benchmarking for students in the class to use.

Future development: This project will be further developed to illustrate the arithmetic operation by incorporating animation objects showing the detailed steps of code execution and the relative speed of executing the algorithm on different platforms.

SUMMARY

This experiments attempts to bridge the gaps found between the synchronous and asynchronous distance learning format. We believe through a well organized delivery method of the material in an environment that integrated the beneficial components of both modality while minimizing the distractions found in distance learning, a new environment will emerge. This environment, we believe, will encourage enrollment in distance education type offerings. We have no misgivings regarding the difficulties in developing this environment. Skepticism comes from both sides of the isls learners and educators. However if a flexible environment is developed that allows in-class and out-class delivery and thus allowing for a custom tailored environment for each learner, this modality will be widely accepted. The effort on the part of faculty delivering courses in distance learning is great and far exceeds that of traditional class offerings. But in the long run it is beneficial to reach out beyond the walls of the campus to learners who may otherwise not participate.

References

1- Khader & Barnes, Computer Supported Multimedia Interactive Distance Learning for Engineering and Technology, ASEE96

2- Khader & Barnes, Laboratory Courses in a Distance Learning Setting, FIE 96

3- Mansoor Sarwar, “Laboratory Exercises for Practical Algorithm Evaluation, IEEE transaction on Education, November issue of 1996.

4- Telecommunication Protocols and Design , John Spragins, Adison Wesley 1992

Professor Michael Khader earned his BS degree in Electrical Engineering in 1982 from Polytechnic Institute of New York. He earned his Master degree in Computer Science from Stevens Institute of Technology in 1991. Professor Khader is currently enrolled in a Doctorate program at stevens Institute of Technology. His main area of research is video conferencing over the public network environment. He is a member of IEEE and many subsequent sections. He is also a member of ASEE.