A Modified Team-Based Learning Methodology for Effective Delivery of an Introductory Programming Course

Ashraf Elnagar
Department of Computer Science
University of Sharjah
Sharjah, UAE
ashraf@sharjah.ac.ae

Mahir S. Ali
Department of Computer Science
University of Sharjah
Sharjah, UAE
mahirali@sharjah.ac.ae

ABSTRACT
This paper sheds the light on an attempt to use a pedagogy integrating Team-Based Learning (TBL) for effective learning and hands-on experience in an introductory programming course. We have adopted a modified version of TBL to study its effect on students learning and to examine how teams intra- and inter-team active interactions influence students’ learning and grades. Therefore, the objectives of this work is to evaluate whether the proposed modified TBL methodology would or would not improve students’ experience of in-class engagement and their attitude about the value of using teams for learning, compared with experiences students would have had in the traditional lecture format. We refer to this method as LTBL, Lectures and Team-Based Learning, since we added a short but essential teaching component. To explore the effectiveness of this learning pedagogy, we evaluated students’ level of engagement and attitudes toward achieving course outcomes and the value of teamwork over four semesters. Our findings demonstrated that LTBL has a positive effect on the success rate and the achievement of learning outcomes.

Categories and Subject Descriptors
K.3.1 [Computers and Education]: Computer uses in Education - Collaborative learning.; K.3.2 [Computers and Education]: Computer and Information Science Education - Computer science Education.; D.1.4 [Programming languages]: structured programming

General Terms
Case study, interactive learning

Keywords
Introductory programming course, Team-Based Learning

1. INTRODUCTION
Experimenting with new and different methods in teaching the first programming course is not a new phenomenon [12, 8, 1, 7, 5]. For decades, computer science and other related IT departments have struggled to find a solution to the high drop rates or/and unacceptable level of failures and low grades of their students taking the first programming course[14, 4]. A paradigm shift in teaching CS courses, regardless of the language being taught, is needed to correct this situation and hopefully improves the students’ retention, improves their performance, and also attracts larger numbers to the IT discipline related majors. Adaptation of a variation of method called Team Based Learning, TBL, may be the answer. TBL is a teaching method proposed for and implemented by some medical schools[3, 13, 19, 6, 12, 15] and then migrated to a few IT departments. The implementation of TBL in programming courses has been reported in a number of publications [10, 11, 18]. Simply put, TBL is a method that replaces passive learning by active learning, where instead of the normal form of students listening to an instructor introducing a new concept, they become active participants in learning the new concept themselves. TBL has three phases: students reading the materials scheduled for each class prior to coming to the class (pre-class preparation), testing their understanding of the material, and finally applying the concept learned. The class time is used for group discussions and short tests and quizzes. The tests’ results are used as feedback to the students as well as the instructor on the level of understanding of the assigned topic. The feedback from the quizzes can then be used to justify either explaining the topic further or move on to the next topic. This method does not reduce the load and the effort of the instructor; on the contrary it requires him/her to prepare the lectures in a different way which involves greater effort than the straight forward passive lecture format, such as making the material of the next class available to the students prior to coming to class using any online software (such as the Blackboard or Moodle). This method also requires more time for marking. As an extension to TBL, Modified Team Based Learning (MTBL) is a method that has also been developed which, like TBL, requires the students to prepare the materials prior to coming to class, but unlike the classical TBL, MTBL requires the students to have individual self-assessment test for each assigned concept, discussion of the learned issues by the students and if necessary the instructor’s involvement for clarification of difficult or not understood topics, followed by a group discussion and finally, the group re-take the same quiz [16]. For our CS1 course, which has a hybrid of science and engineering majors, the classical TBL was customized to include some formal lectures. We named it Lecture and
Team Based Learning (LTBL). Just like TBL and MTBL, we still require the students to be prepared prior to attending classes but instead of allocating the whole class time for group discussion and testing, we allocate some classes for lectures only when we feel that the material requires further but formal explanation using the classical teaching method. However, most classes begin by giving a short lecture of the assigned topic(s) followed by group discussion and finally a short quiz. Based on the result of the quiz, the next upcoming class will be decided upon whether, it would all be a regular lecture, a discussion and a quiz lecture, or to move on to a new topic. This paper reports on our finding of using LTBL for 3 semesters and on the feedback we got from the students. We begin by a brief introduction to the methodology of TBL, MTBL, and LTBL. This paper is organized in the following sections. Section 2 describes the modified team-based learning methodology with details on course structure, team formation, and assessment criteria. The analysis and the findings of using LTBL are reported in Section 3 addressing both students’ performance and evaluation of LTBL effectiveness. The work is concluded in Section 4.

2. MODIFIED TEAM-BASED LEARNING

2.1 Course structure

The course (programming 1) is a 4-credit course, which is offered over a 16-week period and consists of two parts: lectures and practical sessions. Each week students are expected to attend three hours in a classroom and a two-hour practical session in a laboratory. Attendance is mandatory in both parts. The course is a multi-section and several instructors are involved in teaching the lectures as well as the lab sessions, all under the supervision of the course coordinator. The course is a required course for CS majors as well as engineering and some science majors. In the traditional model, the course was taught using lectures, homework assignments, and practical exercises (during the lab sessions). Students were encouraged to revise the material and contact teaching assistants and instructors for help whenever required. Online material was made available in addition to several programming exercises. Quizzes and one or two exams were conducted throughout the semester to assess students’ comprehension of the course content. The course also has a final exam, which consists of two parts, the first part is a practical exam conducted in a laboratory where students are given programming problems and required to implement them and submit their solutions during a two-hour exam period. The second part, which is conducted a week after, is a traditional written exam. The grade is determined by the lab exercises, homework assignments, quizzes, and the two final exams. It was reported over the past years (in the course assessment report) that the performance of the students in general is under-satisfactory and therefore several of the course outcomes were not satisfied. Course average was used to be in the "D+" range. In Spring 2011, we decided to depart from the traditional teaching and learning model to the team-based model. However, requiring and expecting the students to read and prepare for the classes on their own would pose another obstacle. This is because of two main reasons; First, our students are not prepared for this move especially immediately after leaving high school (first year students) where the instructor is the center of attention and students are passive recipients. Second, we believe team-based learning may not be the appropriate tool for teaching some of the heavy-technical courses in engineering and sciences. Heavy reliance on self-learning in such courses would lead to frustration, discouragement, and likely to fire back. As a result, we proposed a methodology to teach CS1 using a modified team-based learning to overcome both problems above and make a smooth transition from the traditional model to the new one. The course still has two parts: lectures and laboratories. The first change we made was to the lab format. The lab instructor now spends the first 10-15 minutes describing the algorithms required (not the code) to solve the assigned problems (usually two problems) and then the students are giving the rest of the time to write, compile, and run the programs (currently we teach C++). At completion time, the grades are immediately provided to the students by the instructor. A similar method to the one used in ACM National Programming Competition is used to evaluate each student’s work. The instructor and the lab assistance have a data set that they use to test each submitted program to ensure the correctness of the submitted algorithms. By the end of the semester the students will have completed between 12 and 13 lab sessions which carry 15-20% of the semester grade. Further, there is a final lab exam at the end of the semester which carries between 10-15% of the final grade. The remaining semester grade is divided between the course work and the theoretical final exam. We have eliminated homework from the course since almost all the problems we assign have their solutions available on the Internet. In the past, lab time was unstructured; the students were giving written description of the lab’s assignment and then the instructor and lab assistance would answer questions and inquiries from individual students. Their task was simply to make sure that the students don’t copy from each other and to submit their assignments prior to leaving the laboratory. The second and the major change we made to the course was during the formal lecture time, which we describe in the next section.

2.2 Methodology

Each formal class begins by the instructor describing today’s topic and assignment. Based on his/her feeling of the students’ comprehension of the assigned topic, he/she may either spend 10-15 minutes describing the topic or let the group discussion begin. Similar to the laboratory format, the instructor monitors the various groups and intervene when necessary either to explain a certain point the students are having a problem understanding or to probe the groups with a question. After about 15-20 minutes, the instructor stops the discussion and begins a quiz. Therefore, LTBL requires the instructor to (i) divide the students into teams (takes place at the beginning of the semester), (ii) briefly explain the concept of today’s class, (iii) allocate time for the groups to discuss and explain the new concept to each other, and finally (iv) examine the students by giving a short quiz on today’s topic only, and (v) give them immediate feedback. Hence, the class time is divided into activities which are mainly carried out by the students. Beside the recommended textbook, which the students are required to purchase, the university Blackboard is used to provide the students with the materials in the form of powerpoint files. These files, along with other useful materials, are made available on weekly bases to the students. Further, the online syllabus describes in details the topics to
be covered each week, the grading scheme, and the dates for the various exams. We don’t make the laboratory assignments available ahead of time because of our concerns that the students may copy the solutions from the Internet. However, they may access the Internet during the laboratory time but because of the presence of the instructor and the lab assistance, their continuous monitoring of the students’ progress in each session, and their familiarity with each student’s ability, which they become familiar with after couple of weeks, there is very little chance of copying the whole C++ code.

2.3 Team formation
Teaching with small groups is not an uncommon practice. However, it is different from TBL groups. Teams are a special type of groups that are more effective and productive. Groups are formed at an early stage of the semester (first week) with a core objective of turning them into teams in a few weeks. This is achieved when group members start to gain confidence in each other and build up loyalty to the welfare of the group. Over time, teams become cohesive. Teammates will soon realize that a team can accomplish far more than an individual team members or even a newly-formed group. Therefore, it is important to appropriately form the teams while guaranteeing that they will have the required resources to complete assignments and participate effectively in active learning. In accordance with team formation policies, we use three basic principles to guide the team formation process, which are create teams in the first week of the term, use transparent selection criteria, and diversify teams. We developed a survey to help forming the teams taking into account two important parameters which lead to successful teams formation. One important criterion is the human resources. We distribute students across the teams based on their knowledge of the subject, GPA, and performance in similar subjects. Such factors are likely to predict students’ performance in this course. The second main criterion is to avoid self-selected teams. We collect information on students’ majors and level in cohort (freshmen, sophomore, junior, and senior). Research has indicated that self-assigned teams are less effective in the course performance when compared to instructor-assigned teams. Diversity in teams is a positive sign. When students are asked to form teams, it is expected to find homogeneous teams that lack diversity. On the other hand, when instructors assign teams taking into account students’ skills, assets, and liabilities, we will get diversified teams which will lead to more effective and productive results [2]. Research shows that showed that instructor-selected teams outperform student-selected teams. In addition, the selection process should be a transparent one in order to assure fair distribution of resources. Otherwise, this could hinder the whole process. As for the size of the teams, we keep it between 5 and 6 students. This is in line with recommendations reported in the literature, [13].

2.4 Student assessment
In the traditional curriculum, the assessment criteria included two tests (30%), lab exercises (10%), homework assignments (10%), final practice lab test (10%), and final exam which accounted for the remaining 40%. In the modified TBL curriculum, the assessment scheme was changed to include one midterm test (20%), lab exercises, including...

3. ANALYSIS AND RESULTS
In order to properly evaluate the possible effect of LTBL on student performance, we compared the final results between two groups of Programming 1 multi-sections over four semesters (from Fall 2010 to Spring 2012). While we subscribe to the LTBL paradigm, other instructors still employ the traditional lecture model. Our study covers 603 students over two years with enrolment between 131 and 198 students. The course is offered in multi-sections and taught by several instructors. It has a course coordinator to set up the general policies and observe the implementation of common exams. Therefore, all instructors follow the same marking scheme with the exception of replacing LTBL assessment tools with other tools such as quizzes and homework assignments. All students took the same common exams and lab exercises. The comparison of the performance of the two sets of students is based on their final grades as well as their achievement of the course outcomes. Data were collected in Microsoft Excel and transformed into two data sets (LTBL vs. traditional) for analysis. The course has six learning outcomes (numbered A-F; see Appendix A).

3.1 Students’ Performance
Figure 1 depicts the comparison results between the traditional and LTBL methodologies which was reported recently in Spring 2012. It is clear that LTBL has outperformed the traditional method on each of the outcomes. The overall performance of LTBL sections is 79.3 whereas it is 69.2 for the remaining sections. This is quite a noteworthy improvement. The summary of how both groups as a whole performed in terms of final grades is shown in Figure 2. The figure illustrates the percentage of students who scored all grades...
ranging from "A" (≥ 90) to "F" (< 60). Notice that the percentages of students who scored "A" to "C" grades in the LTBL sections are higher than its counterpart in the other sections. On the other hand, the percentages of students who scored "D" and "F" grades in the traditional methodology based sections are higher than its counterpart in the LTBL sections. The results of (Spring 2012) are in line with the results of the three preceding terms as shown in Table 1, which summarizes the final average score for each of the course outcomes between the two sets of sections. The numbers in the table fluctuate from one semester to another because of a number of variable parameters such as type of students, instructors, and most important the assessment tools. For example, over the past several years, we have noticed that the quality of the student body that takes the course in the Spring semesters is usually higher than those who take it in Fall semesters. Numbers that are shown in bold font indicate boarder-line performance.

We present Figure 3 to show the improvement trend, for each of the course outcomes, over the last four semesters between LTBL and the traditional methodologies based on Table 1.

Figure 2: Comparison of students overall performance in terms of final grades in Spring 2012.

Table 1: Achievement of Course Outcomes

<table>
<thead>
<tr>
<th>Term</th>
<th>Mode</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fall 2010</td>
<td>LTBL</td>
<td>75</td>
<td>73</td>
<td>73</td>
<td>71</td>
<td>75</td>
<td>66</td>
</tr>
<tr>
<td>Fall 2010</td>
<td>Traditional</td>
<td>72</td>
<td>70</td>
<td>71</td>
<td>61</td>
<td>66</td>
<td>57</td>
</tr>
<tr>
<td>Spring 2011</td>
<td>LTBL</td>
<td>90</td>
<td>83</td>
<td>73</td>
<td>74</td>
<td>71</td>
<td>61</td>
</tr>
<tr>
<td>Spring 2011</td>
<td>Traditional</td>
<td>84</td>
<td>80</td>
<td>74</td>
<td>60</td>
<td>65</td>
<td>61</td>
</tr>
<tr>
<td>Fall 2011</td>
<td>LTBL</td>
<td>87</td>
<td>85</td>
<td>74</td>
<td>71</td>
<td>79</td>
<td>72</td>
</tr>
<tr>
<td>Fall 2011</td>
<td>Traditional</td>
<td>84</td>
<td>83</td>
<td>67</td>
<td>61</td>
<td>74</td>
<td>68</td>
</tr>
<tr>
<td>Spring 2012</td>
<td>LTBL</td>
<td>92</td>
<td>88</td>
<td>75</td>
<td>72</td>
<td>77</td>
<td>71</td>
</tr>
<tr>
<td>Spring 2012</td>
<td>Traditional</td>
<td>83</td>
<td>83</td>
<td>63</td>
<td>60</td>
<td>65</td>
<td>60</td>
</tr>
</tbody>
</table>

3.2 Evaluation of LTBL

Student’s narrative comments elicited both positive and negative feedback on LTBL. Positive attributes included the ability to cover a vast amount of material in a short time; retention of concepts was aided and stimulated by team discussion, and course coordinators during team discussions helped to immediately clarify misconceptions. Negative attributes of our initial LTBL offering included the notion that some team assignments were too long to complete; more practice clinical application questions would have been helpful, and more time for team discussion was needed to ensure correction of mistakes and proper understanding of the tested concepts. Faculty from our department who attended the LTBL sessions were impressed by the students’ cognitive engagement during the team discussions and showed interest in adopting the LTBL concept for more information technology courses.

In order to elicit students thoughts and suggestions for improving the delivery of the course, we conducted a comprehensive survey (developed based on [17]) about their experiences with LTBL. The survey includes 26 statements covering three major categories: LTBL effectiveness, teamwork skills, and potential drawbacks of LTBL. The statements of the first category probed student’s preparation for team discussions, usefulness of LTBL for acquiring knowledge, importance of peers discussion for deeper understanding, and preparation for exams. The second category
examines the attitudes about working with peers, collaboration skills, and mutual respect during team discussions. The last part of the survey sheds the light on potential drawbacks such as getting a way without doing much, frustration from team work, and socializing instead of working. The response of the students indicates the extent to which there is agreement with each statement in the survey. The questionnaire was administered at the end of the course before the final examination, and was computer analyzed from students responses, which remained confidential. Out of 75 respondents, 12 were not used due to missing data or incomplete survey. As a result, the final sample size included (84%) respondents. The analysis lead to three main findings summarizing the students’ opinions on LTBL effectiveness, teamwork skills, and potential risks. Each of these three categories is comprised of a number of statements (specifically, 13, 9, and 4 statements, respectively). Each statement is scored as 1, 2, 3, 4, or 5; a higher score represents a favorable view. We have included one statement related to students’ perceptions of LTBL methodology, which says "Both lecture presentations and TBL complement each other". Figure 4 depicts the response of all respondents which summarizes the overall positive ratings (95.5% agreement) of their perception of the modified TBL when compared to traditional or even TBL paradigms. Only 2.2% indicated disagreement with this statement. Figure 5 summarizes the overall constructive scores of the students view on the degree of LTBL effectiveness with a total agreement of 92.6%. Similarly the results were encouraging on the positive contribution of LTBL to teamwork skills. Figure 6 shows an agreement of 88.3% and a disagreement of 10.6%. There are some potential drawbacks as a result of using LTBL such as getting a free-ride, socializing instead of doing the work, and frustration while working as a team. Students’ perception is in line with instructors’ perceptions too. The results show a disagreement of 45.5% vs. an agreement of 41.1%. This means there is a conflict on whether LTBL has a negative side. This is not surprising as the above shortcomings are controversial issues, [9].

4. CONCLUSION

We presented in this paper the results of using a modified version of the pedagogy TBL, which is a method of teaching that departs from the classical passive learning method, where the instructor is the center of attention in the class, to making the student the center of attention. LTBL, Lecture and Team-Based Learning is a modification to the classical TBL methodology which is better suited for programming courses and possibly other IT courses. The results from our three semester experiments show that students who are serious about learning have benefited well from this shift in the teaching methodology, which also reflected well on the course outcomes. Our intention is to continue teaching programming 1 using LTBL and will continue to refine the method so that improvements continue in the students’ success rate and achieving higher percentage of course outcomes. The results are very encouraging and will be recommended to our CS department to consider LTBL as a possible pedagogy for teaching the majority, if not all, the courses in CS department.

5. ACKNOWLEDGMENTS
The authors would like to thank instructors of the Department of Computer Science who evaluated the LTBL methodology. Special thanks are due to Mohammed Lataifeh who helped in producing the figures. We would like to extend our thanks to Programming 1 students who provided excellent feedback on improving some of the processes over the past 2 years.

6. REFERENCES


APPENDIX

A. COURSE LEARNING OUTCOMES

The following are the course learning outcomes:

A. Identify the different phases of problem solving.
B. Analyze simple problems, design algorithms and write them in pseudo-code or flowcharts.
C. Develop, test, and debug computer programs.
D. Apply the concepts of variables, data types, input, output, expressions, assignment, the processes of decision-making and repetition.
E. Apply the concepts of modular programming using predefined and user-defined functions.
F. Use the concept of arrays and apply them to practical and real problems.