Engaging Students by Intertwining Puzzle-Based and Problem-Based Learning

Jalal Kawash
University of Calgary
2500 University Dr. NW
Calgary, Alberta, Canada
+1 (403) 220 6619
jalal.kawash@ucalgary.ca

ABSTRACT
In this paper, we describe our experience in a first-year computer science course on problem solving that is aimed at non-majors. The majority of the target audience of this course is from management and social science faculties. The course is unique in the sense that it covers proper problem solving skills that are typically only covered in a course directed at computer science or engineering students. We describe how puzzles are used as bait that lures the target audience to subjects such as logic, algorithms, and graph theory. That is, we discuss how we intertwine puzzle-based and problem-based learning in order to engage, an often unmotivated, crowd. Our initial assessment of the approach shows that the use of puzzles is perceived by students as helpful to learn the course material. Furthermore, it is contributing to their interest in computing.

Categories and Subject Descriptors
K.3.2 [Computers and Education]: Computer and Information Science Education – Computer science education, Curriculum, literacy.

General Terms
Design, Human Factors, Theory

Keywords
Computer Science education, problem-solving, first-year curriculum, computer literacy, puzzles

1. INTRODUCTION
“There are three buckets. One is labeled with oranges, the second with apples, and the third with oranges & apples. The labels are guaranteed to be wrong. You can draw one fruit at a time from a single bucket. What is the minimum number of draws you can make in order to determine which bucket contains what?” This is one of many similar interview questions tabled to potential software developers at Microsoft. Other IT companies, such as Google, present similar questions to their interviewees. Developers’ Web sites are full of such puzzles compiled by interviewees (for example, see Glassdoor’).

The choice of such IT giants to use puzzles as interview questions is not surprising. Poundstone addressed this trend in interview questions [1]. It is believed that there is correlation between the ability to solve such puzzles and the ability to solve problems in general. Falkner et al. [2] clearly frame it: “There is a strong connection between the ability to solve puzzles and the ability to solve industry and business problems.”

Therefore, it is natural to employ and exploit puzzles in teaching problem-solving and critical thinking. This has been the case for a long time in elementary and secondary education. Puzzle-based learning has recently found its way to post-secondary education. Universities like the University of California, Santa Barbra [3], Carnegie Mellon University [2], and Adelaide University [2] introduced puzzle-based courses, typically at the freshman level, aimed at engineering and computer science students.

Puzzles are attractive because they illustrate problem-solving rules in a simple setting. At the same time, many conclusions are applicable to real-world problems. Due to their entertaining and provoking natures they engage students.

However, the full power and the benefits of this approach in post-secondary education is not yet completely understood. It is still a work in progress. Recently, Falkner et al. [2] described their experience at Carnegie Mellon and Adelaide universities with puzzle-based courses. The motivations for Falkner et al. is to address two issues: (1) teaching thinking skill rather than simply content and (2) the decline in math skills among students.

Falkner et al. describe learning as a continuum (their illustration is reproduced in Figure 1) with three areas: puzzle-based, problem-based, and project-based. Puzzle-based learning is domain independent, but the other two areas are domain-specific. Puzzle-based learning fosters critical thinking, logical and abstract reasoning. In problem-based learning, students acquire domain-specific knowledge and use this knowledge for reasoning when solving specific problems. Finally, in project-based learning, students...

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

SIGITE’12, October 11–13, 2012, Calgary, Alberta, Canada.
Copyright 2012 ACM 1-58113-000-0/00/0010…$10.00.

1http://www.glassdoor.com/Interview/Microsoft-Interview-Questions-E1651.htm
learning, the learners work in teams to identify the questions, and these questions are answered while dealing with uncertainty and changing conditions.

![Image](Figure 1. A reproduction of Falkner et al.'s learning continuum [2])

There are other papers that deal with the use of puzzles in the classroom with varying objectives. Parahami’s [3] concern is retention of engineering and computer science students. Levitin and Papalaskari [4] and Shilov and Yi [5] use puzzles in the context of teaching the design and analysis of algorithms and model checking respectively. Michalewicz and Michalewicz use puzzle-based learning for engineering students [6]. A major objective of their work is to break the ice barriers that students typically have with mathematics. Rao studies the use of puzzles and storytelling in a software engineering course [7]. Rao finds that this approach improves the quality of end-term projects, slightly improves final marks, and reduces team friction. Xu and Mayer [8] present their experience with using puzzles to teach critical thinking in an on-line course. Cha et al. [9] design an experiment that the shows the use of physical puzzles (with tangible objects) improves students’ strategies for problem-solving and enhances their grasp of abstraction and generalization.

In this paper, we describe our experience with using puzzles in a course on problem-solving. What is unique about this experience is that our course is not aimed at computer science or engineering students, unlike all the papers cited earlier. The course is a service to students from different disciplines, but they are mainly from management and social sciences. Hence, its design is not concerned with preparing future software developers, computer scientists, or engineers. A common goal with other research, though, is to foster critical thinking and problem-solving methods and techniques in real-life scenarios.

The course is unique in its design. To the best of our knowledge, no other course has its breadth and depth in computer science topics and is, at the same time, targeted to students outside the computer science or engineering disciplines, namely students in management, social sciences, and the arts [10].

Unlike Falkner et al.’s courses, we use puzzle-based learning to support problem-based learning. That is, in the lectures, we move through their learning continuum from puzzle-based learning to problem-based learning. Hence, the use of puzzles serves as bait. A typical lecture starts with puzzle-based learning, transitions to the domain-specific problem-based learning, comes back and forth as needed. We also touch on project-based learning in the course, but this is conducted outside the lectures and is outside the scope of this paper. Unlike Xu and Mayer, we use puzzles to engage students in a classroom rather than on-line.

The rest of this paper is organized as follows. Section 2 gives a brief overview of the course in order to contextualize the following sections. Section 3 provides our approach to the use of puzzles and a detailed example. A preliminary assessment of the students’ perception of this design is presented in Section 4. Finally Section 5 concludes the paper.

2. THE COURSE

CPSC 203 is titled “Introduction to Problem Solving Using Application Software”. It is a first year computer literacy course for non-majors. The big ideas [11] of the course are in Table 1. These big ideas are of a long-lasting nature in a constantly changing field.

The bulk of our students are from the management and social sciences schools. Yet, students from natural sciences, communications, and other disciplines also register in the course. Hence, there is a wide variety of students who take the course. It is a required course for students taking 3rd year information systems management courses, and is an elective course for other disciplines.

<table>
<thead>
<tr>
<th>Table 1. Big Ideas and Understandings of the course</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Big Ideas</strong></td>
</tr>
<tr>
<td>1 Abstraction</td>
</tr>
<tr>
<td>2 Generalization</td>
</tr>
<tr>
<td>3 Problem Modelling</td>
</tr>
<tr>
<td>4 Problem Solution Development</td>
</tr>
</tbody>
</table>

2.1 Course Format

The course is a 3-credit hour course delivered in a combination of lectures and tutorials: 150 minute lectures and 100 minute tutorials every week in a 13-week semester. The course’s intake...
of students is at least 600 per semester, with lectures ranging from 100 to 175 students in a regular 13-week term (and about 60 students in a condensed 6-week term), with a cap of 25 students for each tutorial. Staffing requirements are from 2-4 instructors and at least 14 teaching assistants (TAs) per semester.

2.2 Course Content
The course’s objectives are achieved via a set of concepts and a set of skills. While these are not completely independent, the concepts are covered in the lectures and the skills are covered in the tutorials. The scheduling takes into consideration the relationships between the concepts and skills so that they are run in parallel between lectures and tutorials. The concept topics include:

FUNDAMENTALS
- **Logic, naïve set theory, and algorithms:** propositional logic, truth tables, quantifiers, sets, relations, and functions, algorithms, correctness and efficiency of algorithms.
- **Graphs and trees:** graphs (undirected, directed, and labeled), Euler paths, example graph algorithms such as graph coloring, applications to scheduling, trees, binary trees, coding, Huffman’s coding algorithm, finite state machines.

ARCHITECTURE
- **Computer Organization:** modern computer components, memory hierarchy, multi-core machines, magnetic and optical disk operation.

APPLIED PROBLEM SOLVING
- **Databases:** Entity-Relationship (ER) modeling and ER diagrams, database schema, translating an ER diagram to database schema, first, second, and third normal forms, Query By Example (QBE), SQL, set operations in SQL, natural joins in SQL, aggregate functions in SQL.
- **Programming:** Alice programming, object-oriented concepts, variables, functions, conditionals, lists, loops, event-based programming, searching and sorting, top-down and bottom-up designs.

The fundamentals provide a basic framework of concepts in which problem solving skills can be developed through clear logical reasoning and the construction of simple graph-based models. The fundamentals are often illustrated with “classic” examples from computer science. These fundamentals are then augmented with an understanding of modern systems. Finally, we focus on solving real-world problems taken from various domains (business, sciences, and arts) first through developing database applications, and finally via developing simple programs. In the last section, we repeatedly emphasize the use of fundamentals to design an appropriate solution.

The resulting lectures cover a smaller set of concepts than typical of many survey courses – but cover them in greater depth than most survey courses. Students are introduced to fundamental concepts in computer science with sufficient grounding that they could pursue higher level (2nd or 3rd year) courses in the same subjects. Judicious choice of problems and examples is used to appeal to the diverse backgrounds of the students.

The tutorials cover lower-level, hands-on skills, such as using spreadsheets and database management systems. We will not discuss tutorials further since what we discuss in this paper is based on the lecture design.

3. THE USE OF PUZZLES
The puzzles and problems used in the classroom are of three types, categorized by their objectives. The first category constitutes domain-independent puzzles and is aimed at igniting students’ curiosity and interest. Such puzzles are used at the beginning of a lecture before a certain subject is introduced. However, these puzzles are chosen carefully to serve as a bridge to the lecture material. The second category’s objective is reinforcement. It belongs to problem-based learning. These are used in order to allow students to directly or indirectly apply the concepts learnt in a lecture. The third category is used as a reflection mechanism, which permits students to entertain the limitations, extensions, and approximations of learnt concepts. This can be either in the puzzle-based or problem-based arena.

3.1 Detailed Example
We present an example lecture in the context of the fundamentals module, discussing Euler paths in graph theory. An objective of this lecture is to see how graphs can be used to abstractly represent real-life problems and how it can lead to generalized conclusions and universal applications.

Pre-lecture puzzle: The class is challenged with tracing shapes, such as the ones depicted in Figure 2. The shapes are chosen such that one shape is not traceable, the second is traceable as long tracing starts and ends at different points, and the third is traceable as long as tracing starts and ends in the same point. The students are given a few minutes to try to trace these shapes, working in teams. Then, they are asked to try to reflect with their team mates on why shapes are or are not traceable, and if they are, they should try to see if all shapes are traceable in the same way. More specifically, if a shape is traceable, can it be traced starting at any point in the shape or not? If not, where does tracing start and where does it end? No answers are given at this stage to these fundamental graph theory questions. Yet, the students are told that the answers will be given later in the lecture.

![Figure 2. Traceable and non-traceable shapes](image-url)
**Lecture:** The lecture is started by the Konigsberg story leading to a graph representation of the town’s layout. The question of taking the famous town tour, crossing each bridge exactly once should lead the students to make the connection to tracing shapes. Is the Konigsberg graph traceable, and if not, why? The lecture proceeds to exploring Euler’s theorems regarding an Euler cycle and an Euler non-cycle path. Once these fundamentals have been thoroughly explained, the Konigsberg graph is revisited as well as the shapes used in the pre-lecture puzzle. Students now get the chance to see how these shapes can be represented as graphs and how Euler’s theorems hold the answers to what is traceable and what is not. The lecture continues with Euler’s algorithm for finding a tour (alternatively tracing a shape).

**Post-lecture reflection:** The pre-lecture puzzle and the lecture in the form that was explained earlier do indeed capture students’ attention. At this stage, we used to expect students to pose the question “so what?” or “how does this benefit me in real-life?” However, students are quite submerged, engaged, and having fun with tracing shapes to the extent that we have not yet faced any question of this form! Hence, we are the ones who pose this question.

So, one remaining piece of the puzzle is how this relates to real-life problems. Here, one example is to use a scheduling problem from real-life, such as scheduling equipment when paving streets. The students are given a map of a small section of downtown Calgary (Streets in certain areas of downtown Calgary are straight, forming a nice grid of lines to work with). The objective of the problem is to schedule paving equipment so that all the streets in the area are paved, and it is not desirable to have the equipment driven on newly paved streets. It will take students a few minutes of discussion in teams to realize that this can be represented as graph problem and that an optimal solution would be an Euler path for this graph. We have also used other take-home reflection problems, where students need to design a courier delivery schedule.

The second remaining piece of the puzzle is what to do when an Euler path is impossible due to the characteristics of the given graph. This serves as a bridge to the topic of approximating a solution. That is, if an Euler path cannot be found, then it is required that the number of edges to be re-traversed should be minimized. Students are then asked to brain-storm identifying real-life problems with Euler tours can be used as part of their solution.

**Take-home problems:** Students leave with some graph examples where Euler paths are impossible. They are asked to find an approximate path that minimizes edge re-traversal. Often, they are also asked to modify (or at least think about modifying) the presented algorithms in the lecture so that it minimizes edge re-traversal, when an Euler path is impossible.

### 3.2 Other Examples

In this section, we give a few other puzzle examples that we have used and the context in which they are used. We do no intend this to be a comprehensive treatment of all the puzzles we use.

**Set theory:** The course includes a week-worth of material on naive set theory. One puzzle that we have used is the following counting problem (from [12]):

An examination in three subjects, Algebra, Biology, and Chemistry, was taken by 41 students. The following table shows how many students failed in each single subject and in their various combinations.

<table>
<thead>
<tr>
<th>Subjects:</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>AB</th>
<th>AC</th>
<th>BC</th>
<th>ABC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Failed:</td>
<td>12</td>
<td>5</td>
<td>8</td>
<td>2</td>
<td>6</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>

How many students passed all three subjects?

In the context of this course, students are expected to solve this counting problem using Venn diagrams.

**Graph coloring:** Graph coloring often spans a 75 minute lecture. An example puzzle used is:

A framer has a canoe that can fit a single object in addition to him. He wants to transport three objects using his canoe from one bank of the river to the other. These objects are a sheep, wolf, and a bundle of hay. The farmer cannot leave some of these objects unattended; he cannot leave the wolf with sheep unattended and he cannot leave the sheep with the hay bundle, otherwise, one will eat the other. How should he safely transport these objects?

This puzzle serves as a good bridge to constrained scheduling. While the puzzle is simple, we see benefits in modeling it as a graph of objects, where adjacent vertices are the ones in conflict (cannot be left unattended). The use of graph coloring on this graph can give valuable hints on where to start and how to formulate the schedule.

We have often constructed more complex versions of this puzzle, by adding more objects and increasing the size of the canoe to justify the use of graphs and graph coloring.

A reflection problem on the subject of graph coloring follows. This example is an application of the graph coloring algorithm, yet it illustrates additional constraints that must be met in order to get an optimal solution. Any legitimate coloring of the resulting graph is not necessarily an acceptable solution. This creates some rich reflective discussion. The problem is:

You are managing a team of 15 workers: Alice, Bob, Chuck, Dianne, Elise, Frank, George, Hasan, Isabel, Jacky, Kip, Lora, Mike, Norman, and Olga (A to O). You’re to come up with a schedule for your staff for 3 time slots, but you have the following restrictions:

- Couples should not put them in the same slot; couples are:
  - Alice and Bob
  - Kip and Jackie
  - Norman and Olga
  - George and Dianne
  - Isabel and Hasan
• People with the same specialty should not work in the same slot, if possible. The specialty groups are:
  - Elise, Bob, and Chuck
  - Alice, Dianne, and Hasan
  - Lora, Norman, and Mike
• People who had friction working together in the past, should not work in the same lot if possible. These are:
  - George and Kip
  - Lora and Chuck
  - Olga and Hasan
  - Alice and Chuck
  - Mike and Jackie
  - Elise and Isabel
  - Frank and Isabel
  - Frank and Bob
  - Frank and Elise

Algorithmic thinking and efficiency: One puzzle that promotes algorithmic thinking is (this also seems to be an interview question; see My Tech Interviews):

You've got someone working for you for seven days and a gold bar to pay them. The gold bar is segmented into seven connected pieces. You must give them a piece of gold at the end of every day. If you are only allowed to make two breaks in the gold bar, how do you pay your worker?

Another puzzle that touches on efficiency of algorithms is:

A merchant learnt that one of his 4096 golden coins is fake. All coins look exactly the same (including the fake one). The only difference between a fake and a real coin is in the weight; a fake coin is lighter. Find an efficient algorithm to locate the fake coin.

This puzzle allows the students to see the benefits of logarithmic time algorithms over linear-time algorithms without having the background to analyze the complexity of algorithms. The students see the benefits of finding the fake coin in 12 steps (weighing operations) rather than 2048. This is further reinforced by asking the students to double the total number of coins and re-do the calculations. Doubling the number of coins adds one more step to the logarithmic approach, while it adds 2048 to the linear approach.

One reflection problem asks the students to formulate an algorithm that works for any number of coins, not necessarily a power of two.

Another asks the students to improve on the logarithmic time. In the context of a first year course, we have formulated this as a fixed puzzle:

The merchant has 8 coins, one of which is a fake and is lighter. The logarithmic approach discussed earlier yields the fake coin in three steps. Find another algorithm that can achieve this in 2 steps only. Then, we can move to more complex instances in order to generalize the algorithm.

Efficiency of algorithms in the context of our course is a topic that we do not dare to wonder to had not it been discussed in the context of puzzles.

Logical inference:

We finally give a puzzle that promotes logical inference:

A basket contains 5 hats (3 black and 2 white). A person randomly chooses three hats from the basket, and puts them on 3 blindfolded people. Let’s call these 3 people A, B, and C. Then, these people are lined up so that A can see the hats of B and C, B can only see that hat of C, and C can see no hats. When the blindfolds are removed, A says “I do not know the color of my hat”; B says “Neither do I”; C quickly says “My hat is black”. C is right; explain her logic.

4. ASSESSMENT

Our assessment is still work in progress. In a recent on-line survey of a 6-week condensed term, we asked students if puzzle-based learning helps them better learn the material. The survey also asked them if puzzle-based learning got them interested in the course material. There were 56 students registered in the course, and roughly 30 students constantly attended the lectures. Attendance is not mandatory. We received 16 valid instruments (53% of the attending population and 28.5% of the whole class population). The student’s perceptions are summarized in Figure 3. This is the second and most recent offering of the course under this format.

![Figure 3. Summary of students' feedback in a condensed 6-week semester.](image)

Our first offering of this course was in a regular 13-week semester, with 150 enrolled students. Only 27 valid (on-line) instruments were received. A different survey was used, but the assessment of the effect of the use of puzzles on “learning” was common between the surveys used in both semesters. Figure 4 shows a summary of the feedback for both offerings (for comparison).

The feedback shows a vast majority (Especially in the second offering) rating the approach as helpful in learning the material and getting interested in the subject. Written positive feedback includes comments like: “puzzles keep me awake”, “puzzles are provoking”, and “it is fun to talk to and work with people on these brain teasers”. Negative comments include: “I would be more happy learning the required material” and “I do not feel

---

2. [http://www.mytechinterviews.com/gold-for-7-days-of-work](http://www.mytechinterviews.com/gold-for-7-days-of-work)
comfortable talking to others in the classroom; working alone is frustrating”.

Figure 4. Summary of students’ feedback on “how helpful are puzzles in learning the course material.

5. CONCLUSIONS

Puzzles are fun and engaging. They can teach some universal problem-solving skills in a simple context. Technology giants, such as Microsoft and Google, have shifted their interviewing paradigms to look for and hire problem-solvers. Their interview questions typically consist of puzzles. Some universities have designed “all-puzzle” freshman courses in order to retain students and counter the problem of declining math skills in students. We have benefited from such previous work and implemented a course design that uses puzzles as bait to catch students’ interest in problem solving. One challenging aspect in this course is that students are not from engineering or computer science. This is a service course and our students are largely from management and social sciences.

Our students used to find that computer science is “dry” and often too abstract to relate with. Motivating and engaging students have been an ongoing challenge in this course. The introduction of puzzles has started to break the ice. The feedback we are getting from students indicate that they are more engaged and more interested in learning the subject. Yet, more data needs to be collected in order to have more thorough analysis, especially to address the negative feedback.

6. ACKNOWLEDGMENTS

My thanks to project Engage and its staff at the University of Calgary, who gave me the necessary theoretical and logistical help for this redesign and its execution.

I am also thankful to Mishtu Banarjee for his feedback on an earlier draft of this work.

7. REFERENCES


