Faculty of Technology and Society
Department of Computer Science and
Media Technology

Bachelor Thesis
15 ECTS credits, Undergraduate

Teaching Concurrency in a Modern Manner
Flipped Classroom or Game-Based Learning

Utlärandet av flertrådat på ett modernt sätt
Flippat klassrum eller spelbaserat lärande

Mattias Hansen
Bobby Murphie

Examination: Bachelor Degree 180 credits
Subject: Computer Science
Program: System Developer
Date for Final Seminar: 2018-05-29

Supervisor: Farid Naisan
Examiner: Johan Holmberg

Abstract

Much research has been done to find ways to improve teaching concurrency, from visualization tools to game-based learning and flipped classroom. However, research on comparing these methods or models when teaching concurrency are lacking. This paper looks at the different results from students who are studying concurrent programming by comparing two different modern ways of teaching. It also looks at which method/model students prefer and keeps them more engaged. The authors of this paper compare a game-based learning approach to a flipped classroom approach. The game-based learning approach used in this paper is developed by Dr. Robert Marmorstein and uses the game OpenTTD [1]. The students learn about race condition, deadlock and starvation by using semaphores (railway signals) to prevent collisions. The flipped classroom approach in this paper is used in a concurrent programming course at Malmö University. After both of the approaches have been completed, the students take a test and answer a survey to see how much the students learn, how engaged they are and what they prefer. To gain an accurate result, each student that took part in the study only participated in one of the approaches. The results of the survey favor the OpenTTD lab approach as the students were more engaged during the exercise and preferred the exercise more. The students that participated in the OpenTTD lab also did better on the test when it came to explaining how to prevent/solve each synchronization problem, while in the flipped classroom students did better when it came to describing the problem.

Sammanfattning

Mycket forskning har gjorts för att hitta förbättrade sätt att lära ut concurrency. Allt från visualiseringsverktyg till spel-baserad inlärning och flippat klassrum. Dock så saknas forskning som jämför metoder och modeller som lär ut concurrency. Den här artikeln tar upp och tittar på resultat från studenter som studerar concurrent programmering genom att jämföra två olika moderna sätt att lära ut. Den tittar också på vilken metod/modell studenterna föredrar och håller dem mer engagerade. Författarna av denna artikel jämför ett spel-inlärnings tillvägagångssätt med ett flippat klassrum tillvägagångssätt. Spel-inlärnings tillvägagångssättet som används i denna artikel är utvecklad av Dr. Robert Marmorstein och använder sig av spelet OpenTTD [1]. Studenterna lär sig om race condition, deadlock och starvation genom att använda semaforer(järnvägssignaler) för att förhindra kollisioner. Det flippade klassrum tillvägagångssättet i denna artikel används i en flertrådad programmeringskurs på Malmö Universitet. Efter att båda tillvägagångssätten har genomförts tar studenterna ett test och svarar på ett frågeformulär för att se hur mycket studenterna har lärt sig, hur engagerade de är och vad de föredrar. För att få mer exakta resultat får bara studenterna delta vid ett av tillfällena där tillvägagångssätten genomförs. Resultaten från den här studien gynnar OpenTTD labbens tillvägagångssätt då studenterna verkar vara mer engagerade och föredra den lite mer. Studenterna som deltog i OpenTTD labben gjorde bättre ifrån sig på testet när det kommer till att förstå hur man förhindrar/löser varje synkroniseringsproblem, medans de studenterna som deltog i det flippade klassrummet gjorde lite bättre ifrån sig när det kom till att förklara/beskriva problemet.

Word-list

Thread - A sequence of instructions executed. Usually controller by the operating system and is a part of a **process**. Threads have their own memory stack, but not their own heap.

Process - An instance of a computer program. Each process have at least one **Thread**.

Concurrency - Concurrency refers to a program executing multiple instructions at the same time, having more than one thread that belongs to the **process**.

Race condition - Is a synchronization problem where at least two threads operate on the same piece of data at the same time. For example when thread A executes the following instruction "aNum = aNum + 1" while thread B is executing instructions using aNum. Thread A reads in aNum and sees that its value is 1. Thread B reads in 1 as well from aNum. Now thread A executes the instruction 1+1. Writes the value two to aNum. Thread be does not know aNum is set to two now. Executes the same instructions and writes two to aNum, that should have been three now.

Starvation - When multiple threads are using the same resource but one or more threads get little to no chance to use that resource.

Deadlock - When two threads are waiting on each other. For example, thread A has called a method while in a locked section. Thread B is in a locked section as well while waiting to enter the locked section thread A is in. This means that both threads are waiting on each other.

Semaphore - Is a variable or abstracted data type used for controlling access to resources that are not placed on the stack, usually instant variables. Semaphores are also a type of signals used for trains. When x trains are in between two signals the signals turn red, not allowing any other trains to enter. They have to wait in line until a train exits the area. A binary semaphore is when only one train can be in that area at a time.

Contents

1 Introduction

Concurrency has for a long time been an important part of programming and continues to be so. However, teaching students about concurrent programming has always been somewhat of a challenge, and still is. A big part of the challenge is going from sequentially executed programs to concurrently executed programs. In sequentially executed programs the statements are always executed in the same order, whilst in a concurrent executed program the statements can be executed at the same time or out of order. Over the years there have been many attempts to try to find new and fresh approaches to teaching students about concurrency and its concepts with varying results. This paper compares two of these approaches, a flipped classroom approach and a game-based learning approach using the game OpenTTD [2], to see how the results of these approaches measure up to each other.

In this paper, we will use the game OpenTTD which is an open-source business simulator based on the game Transport Tycoon Deluxe [2]. In this game, you build different means of transportation on land, in water, and in the air. To use this game for learning, we are going to use the exercises from the article Teaching Semaphores Using... Semaphores, where they use the inherent similarities between the simulation of a railroad and concurrent programming [1]. There are three different exercises and they each deal with different synchronization problems, namely, deadlock, starvation and race condition. In these exercises, a set of train tracks will represent the synchronization problem, and the students need to place or remove different items from the tracks to resolve the problems.

1.1 Background

1.1.1 Flipped Classroom

The basic idea of the flipped classroom model is to turn the traditional lecture on its head by having the students watch the lectures as videos in preparation at home, then doing the assignments together at school with teachers and other students [3]. The advantage of this model is that the students get the opportunity to talk with each other and with the teacher [4]. The model has become increasingly more popular and is being used for a wide variety of subjects at different levels of education. One major benefit of this model is the increased engagement of the students, which according to research-based evidence suggests that increased interactivity in the classroom improves learning [5]. As with all teaching methods/models, there are also some downsides to this model, such as that the session often requires a longer preparation time than a traditional lecture [4].

S. V. Moore and S. R. Dunlop [4] experiment with a flipped classroom approach as a way to teach concurrency and parallelism in programming to undergraduates in a course for two consecutive years. Although it did not solve all the inherent difficulties it succeeded in improving learning outcomes and even improved in the following year with some tweaks.

1.1.2 Game-Based Learning

The world today is filled with games, and unlike in the past, it is no longer only played by children or by a niche audience. Now there is something for everyone, from the newest Triple-A games to small time-wasting mobile games. Games are also a great tool for learning. Firstly, games engage people, which is an important part of learning, and it does so for a number of reasons. In the book *The Use of Computer and Video Games For Learning* [6] they list a number of these reasons, one of the reasons is that they are fun which brings the player enjoyment and pleasure, they have rules and goals which give structure and motivation, they give you outcomes and feedback which helps you to learn, and these are only a few reasons but there are many more. Secondly, they are a great way to encourage students, and to increase interest and confidence [6]. Nevertheless, as with all kinds of learning approaches, there can also be some downsides. One example is that if the game is too hard or if it has too many distracting elements, it could do more harm than good.

1.2 Related Work

There have been much research done for efficiently teaching concurrency. One approach that has a lot of research done is using tools for visually displaying how concurrent programs execute [7][8]. Two examples of these tools that have been used for research is COOPE and JThreadSpy. JThreadSpy is a tool for Java that allows the user to generate a sequential diagram [8]. The diagram shows thread executions and locked sections. It allows the user to detect deadlocks, race conditions and lets the user choose to see the visualization in different levels of abstraction. JThreadSpy has also been proven to help hasten the development of concurrent programs, and to produce solutions that pass more tests. COOPE is another tool that visually displays information to the user [7]. The information is displayed using three layouts, a table model, a sequence diagram and a state diagram. The authors of COOPE H. Leroux and C. Exton, do not cover how useful it is for students, but rather how to design such a tool. The tool is however designed to help students in their studies with object-oriented concurrency.

Research to improve how students learn concurrent programming has also been made using game-based learning [1][9][10][11]. This paper will cover a few examples of studies related to teaching or learning concurrency using gamification and game-based learning. Naoki Akimoto and Jingde Cheng tell us that teaching and learning concurrent programming is difficult [9]. They present a game they developed that can teach concurrency to people who are new to the concept. The game allows a user to experience concepts such as process, synchronization and communication. The game covers synchronization problems such as deadlock, livelock and race condition. Robert Marmorstein has used OpenTTD for an assignment to teach students about deadlock, starvation and race condition [1]. OpenTTD is an open source simulation game based on "Transport Tycoon Deluxe" which is developed by Chris Sawyer. OpenTTD is a game in which the user can build railway tracks and trains to transport materials and passengers. Marmorstein made three scenarios for the assignment, where each scenario covers one topic each, deadlock, race conditions and starvation. He tells us that all the students completed the assignment and received a lot of positive feedback about how the assignment helped them understand

synchronizations problems better. The exercise motivated students to discuss more in class. Rasha Moris and Edward Jackson also talk about using gaming as a tool and how it can be used to educate engineers [10]. The article tells us how students often spend a lot of time playing games and that they rarely spend that much time studying. When playing a game, a person utilizes critical thinking, problem-solving and parallel- and visual processing. The authors talk about how Materials Education Game (MEG) can be used for teaching Materials Engineering. A design is presented and implemented using 3dsMax, which is a program for making 3D models, animations, games and images. R. F. Maia and F. R. Gram uses a modified version of a game called the factory game, which was originally used to simulate business and production problem to teach students about concurrent and distributed programming [11]. Originally the game had a production line of dices that consisted of different types of machines. Every machine had a specific task and different characteristics. In the modified version each machine is represented as a thread with different characteristics. The demand for different types of products requires different threads on the production line. Through the local network data are sent and received from the production line in the market and suppliers. The result from the study consists of a survey where the students had to answer 16 questions connected to their learning. The survey is meant to measure if students learning inhibitors were reduced from the simulation. The result compares learning from, the game with learning from normal lectures. The study showed that most of the time students' learning inhibitors decreased after the simulation game.

The flipped classroom model has been used and researched for teaching students about concurrency [4]. The paper "A Flipped Classroom Approach to Teaching Concurrency and Parallelism" compared using flipped classroom with ordinary lectures [4]. S. V. Moore and S. R. Dunlop introduce Flipped Classroom which is a pedagogical model where the normal lectures and homework elements are reversed. During class, students should spend their time discussing and doing exercises. The authors compared results from two consecutive years 2014 and 2015 of the same Parallel and Concurrent Programming course, and the flipped classroom was only used during the 2015 course. Both years 16 students attended the course, and the students from 2015 did generally better. The flipped classroom has also been tested for teaching high-performance computing [5]. During the course the teacher used journals which the students wrote in, this was a way to determine if the students actually watched the video lectures before attending the flipped classrooms. Most students watched the videos and the journals also helped to answer questions students had more quickly. The time the students had in class to discuss with their teacher was appreciated by the students. While the sample size makes it hard to evaluate how much the students benefited from this class format, it was clear that the students appreciated the format. During the course evaluation, the teachers had a hard time finding any negative feedback. The class format that was used in order to help the students engage more deeply with course content, talk to their teacher and increase their motivation was a success.

The following papers relate to evaluating how students work and think about concurrency and their studies that relate to concurrent programming [12][13][14]. J. Lönnberg, A. Berglund, and L. Malmi have written the paper "How Students Develop Concurrent Programs" that looks at how students approach developing and testing concurrent programs

[12]. The study gathers data by conducting interviews. The results ended up being more general and do not say much about concurrency specifically. One problem students end up having is that they misunderstand the assignments or what they are expected to accomplish. The study also shows that students tend to be weak in their testing, which could mean that the students do not understand concurrency well enough or that they do not understand the environment in which the programs function in. The paper suggests that using visualization tools to teach students may be helpful. However, the article "Evaluating a Visualisation of the Execution of a Concurrent Program" written by J. Lönnberg, L. Malmi, and M. Ben-Ari shows that students prefer to try to reason with programs in a more statically manner, even if the visualization tool proves to be helpful [14]. The paper tells us that students debug programs using mental tracing, hand tracing, and debuggers, using both forward and backward tracing. The students are instructed to use the visualization tool Atropos, but it is unclear to them how the tool fits into the debugging process, partly because it is a new tool for the students. J. Lönnberg and A. Berglund take a look at students understanding of concurrent programming [13]. Interviews were used to gather data. Lönnberg tells us that students understand tuple space as "a specification, i.e. as a set of operations and how their inputs and outputs relate." The paper looks at what the students think it means to write and debug a concurrent program, that writing a concurrent program can be experienced as solving technical problems.

1.3 Definition of Problem

A lot of research has been conducted in this field. Tools that visualizes multi-threaded execution has been tested when programming to see what the students learned [7][8]. The flipped classroom has also been tested specifically for teaching concurrent programming [4]. Studies have also been conducted to determine what difficulties students have when learning about concurrency [12][13][14]. There are also studies testing how well game-based learning does when teaching concurrency [1][9][10][11].

The problem today is that these new methods/approaches have not been compared with each other. This means that in the current state there is no knowledge of which methods/approaches have certain advantages or disadvantages compared to each other.

1.4 Purpose

The purpose of this paper is to measure and compare the effectiveness of two different teaching approaches when teaching concepts of concurrent programming to undergraduate students. In doing this we shed some light on what advantages and disadvantages the two approaches have compared to each other. This study is also meant to give background to an area in this field that is lacking.

1.5 Research Questions

In this section of the paper, the research questions are stated. The following questions are answered throughout this paper:

- Q1 - How do student results compare between game-based learning and flipped classroom?
- Q2 - How does students engagement compare between game-based learning and flipped classroom?
- Q3 - Do students prefer game-based learning or flipped classroom?

1.6 Limitations

This paper will only cover a select part of concurrent programming, namely, deadlock, starvation and race condition. The reason for this is that we only have a limited time to do our data collection and analysis. Another limitation is the choice of approaches that are being compared, although these two approaches are suitable choices for this paper's purpose, the fact is that they were also chosen because of their availability and flexibility.

It is also important to keep in mind that the flipped classroom has been used before and improved over time while the OpenTTD lab is relatively new. The lab has been tested previously [1]. It is also important to understand that the flipped classroom is held by an experienced lecturer while the OpenTTD lab is held by the authors of this paper, who are pedagogically uneducated and inexperienced.

Another limitation is the number of participants that will attend the OpenTTD Lab, or take the test/survey that follows both the lab and the flipped classroom. Since we need students with no prior experience with concurrency, but enough experience in the field, we have chosen to only take participants from the course "Concurrent Programming" at Malmö University. The flipped classroom method is a part of the "Concurrent Programming" course and the students will get points for their attendance that will help them in their final exam. The OpenTTD lab and the test/survey, however, is not a part of the course and will not give extra rewards to the students, which may limit the number of willing participants.

2 Method

2.1 Method Description

To determine which teaching approach is most effective for learning about synchronization problems a method has been developed to compare flipped classroom and game-based learning which uses OpenTTD. The method involves a flipped classroom approach used in the "Concurrent Programming" course at Malmö University and an OpenTTD lab used at Longwood University in the course "Operating System" [15] by Dr. Robert Marmorstein. After the lab and the flipped classroom, the students will be given a link to a Google Form. The Google Form contains both the test and the survey. The test will be used as data for determining which approach the students learn the most from. The survey is used to gather data that can answer which approach is preferred and engages the students the most.

As the flipped classroom is a part of the course "Concurrent Programming" at Malmö University the OpenTTD lab will be held first. This lab is voluntary as it is not a part of the course. Anyone who attends this lab will not be given a test and survey after the flipped classroom. The test that the students will take after each approach will have the same questions. This is to prevent uneven results that that can come from different questions or from having two opportunities to learn rather than the approaches used to teach the students.

The authors of this paper will observe the students during the time the students are attending the lab and the flipped classroom. This is to gain an understanding of how the students think and what difficulties they face. The observation is not a means of gathering data, but rather a way of detecting possible weaknesses in the lab, test, and survey. Making it possible to discuss these weaknesses and how the method could be improved.

2.1.1 OpenTTD Lab

In the OpenTTD Lab, the students are to solve a set of tasks in the game OpenTTD. The lab was originally created by Dr. Robert Marmorstein at Longwood University [16]. During the lab, students will load up three different save files that all have pre-built train tracks with problems representing different synchronization problems found in concurrent programming, namely, race condition, deadlock, and starvation. For each task, there is a short description of what is happening in the given scenario and how it relates to a synchronization problem in concurrent programming. The students will have to find a solution to these problems by using what they know of the problems and with the tools in the game. The students can choose to put up or take down signals such as semaphores, add or remove tracks and more. After the students have solved the last task they are asked if their solution, although working can be made more efficient.

2.1.2 Flipped Classroom

For the flipped classroom, the students are to watch a set of videos about dining philosophers and the sleeping barber before the exercise. During class, the students are divided into six different groups with four students each to discuss problems and solutions to either dining philosophers or the sleeping barber. Each group is given quiz questions about race conditions, starvation, and deadlocks that they are to answer. The quiz is to be submitted through the learning platform within the following day. During the second half of the flipped classroom, everyone will discuss the answers to the quiz questions together with the teacher.

It may be worth noting that the flipped classroom is not mandatory. However, the students who attend the flipped classroom and complete the quiz will be given an extra point for the exam.

2.1.3 The Test

The test is designed to see how students who attend the two teaching approaches compare to each other. In both the flipped classroom and the OpenTTD lab, students will learn about deadlock, starvation and race condition. Because of this, the test will cover these three problems. When writing the questions for the test it is important to know how much the students know about these concepts. One of the authors of this paper has attended all the lectures before the OpenTTD lab and flipped classroom to gain a grasp of where the students are. This is to ensure that an accurate test can be written.

When determining how many points a question is worth the students' previous knowledge comes to mind. The maximum points on a question are set upon how much the students have learned both from the teaching methods and from previous lectures. The lectures before the OpenTTD lab and the flipped classroom covers shared-memory, critical sections, non-deterministic behavior, atomic operations, locks and thread interference. They also cover the synchronization problems deadlock, starvation and race condition. Deadlock is explained as two or more threads that are waiting for each other. The paper needs the pen, but the pen is busy waiting on the paper. Race conditions are explained as an uncontrolled thread that can be greedy and use a resource before another has a chance to use that resource. For example, one thread writes to a variable many times before another thread has read the variable. This means that the thread that reads did not read all the values that the other thread wrote to the variable. Starvation is introduced as when a thread is not given the chance to use a resource and uses a bridge with one lane as a metaphor to describe starvation. In the example only one lane gets to use the bridge, leaving the other lane stuck. This can be a result of the race condition. Fairness is also introduced, where every thread or process should be given a fair amount of time each. Fairness is usually an attribute of a scheduler.

Using what the students learned from previous lectures and what they will learn from the OpenTTD lab or flipped classroom the test can be written. How many points each question can give the students is based on how much of a detailed answer the students can give.

Worth noting is that reproducing this method to use in another study means that the test questions cannot be copied. This method relies on the students' background to create the test. By using the same test on different groups the research question Q1 could have different results. It can be seen as both approaches give the same results if the test questions are too easy or too hard for the students.

2.1.4 The Survey

The purpose of this survey is to gauge the students' overall experience of the OpenTTD lab compared to the flipped classroom, but also to a traditional lecture. The questions in this survey are made to see what the students prefer, help them the most and get them the most engaged. Every question is on a linear scale from 1 to 5, where 1 is the most negative and 5 is the most positive, and 3 is neutral. The following section covers what questions the survey contains and motivates the purpose of each question.

What did you think of this exercise?
This question of the survey is a general question to see if the students liked the exercise or not. The results from this question can later be used to see which approach the students preferred.

How did you prefer this exercise to a traditional lecture?
This question is meant to see if the students prefer the exercise they took part in traditional lectures.

Did this exercise make it easier for you to learn about synchronization problems?
Learning about synchronization problems can be difficult, and the purpose of this paper is to find a more optimal way of learning. This question asks the students if they think this approach helps them to better understand. Besides, this question can also be used to gauge student engagement, as engagement is an important part of how we learn.

How engaged were you during the exercise?
To be engaged during a session is an important part of learning, and knowing how engaged the students are during the OpenTTD lab and the flipped classroom is important when gauging their interest, and how efficient the approach is.

How much do you think you learned?
By making the students take the test we can see how much the students know after the session, it is also important to know how much of it they learned during this session. By knowing how much the students think they learned can also be used to gauge how engaged the students were during the exercise.

Would you like to see more of these exercises in the future?
For most of the students who are taking this survey, it is the first time experiencing any of these approaches. This question is a good way to measure the students' interest and their preference.

Other Comments
In this section the students can say anything and all about the OpenTTD lab, the flipped classroom, the test and the survey. It is likely that we have missed something that can be great to discuss.

2.2 Alternative Methods

As a means to generate data, other methods could have been used. To get an understanding of what interests and motivates students interviews could be used instead of a survey. This would allow the authors of this paper to gain more detailed data and a clearer understanding of what the students think. However, the time limit both for the authors and the students makes this method more challenging and limits the number of students that can participate.

Interviews could also be used for gathering data to validate whether the flipped classroom or game-based learning using OpenTTD would yield better results. The problem with using interviews is that it would be difficult to measure the difference. It would only allow for soft values while using a test gives hard values. As mentioned earlier, there is also the problem with time, meaning the generated data from interviews would include fewer students.

Similar problems would arise from observation to gather any real tangible data as well. Observing all the participating students would prove difficult, but by using cameras or recording devices this could be solved; however, it would be too time-consuming. Measuring values by observing would also be a problem.

As there exists previous research comparing different tools and approaches with normal lectures these results could be compared with each other. The problem with this is that this would mean comparing the student with very different backgrounds and data gathered from different tests. This would make the data much less reliable as one test group would have more previous knowledge and experience than the other test group. The upside is that enough data can be gathered through a literature review and there would be less work required to gather the result. This method would also not be able to answer research question Q2 and Q3 while still related to the same approaches used to answer Q1.

3 Result

The results cover how the students did on the test and what they think of the session they took part in.

3.1 Test

The test has a maximum of 13 points. Each question asks what the following synchronization problem is and how to prevent it. Question one is about race condition with a maximum of five points. Question two is about starvation and the last question is about deadlock, both with a maximum of four points each. Every A question asks the students to describe the given synchronization problem. Meaning question 1: A ask the students what a race condition is. As the students have learned more about race condition during the course more is expected from the students when answering this question. This is why question 1: A offers three points while the rest of the sub-questions have a maximum of two points. Every B question asks the student to explain how the given synchronization problem can be solved or prevented. For example, question 3: B asks the students how deadlocks can be prevented or solved. The questions are from a more theoretical perspective. The students are not expected to write any code.

3.1.1 OpenTTD Lab

This section contains tables of the result from the test students took right after the OpenTTD labs.

Student/Question	**1: A**	**1: B**	**2: A**	**2: B**	**3: A**	**3: B**	**Total**
OpenTTD Lab Student 1	2	2	0	0	2	0	6/13
OpenTTD Lab Student 2	0	0	0	0	0	0	0/13
OpenTTD Lab Student 3	2	1	2	1	1	1	8/13
OpenTTD Lab Student 4	2	2	2	1	1	0	8/13

Table 1: OpenTTD Lab 1, Test Result

In table 1, it shows that student 2 did not get any points. This is because the student did not answer any questions in the test. From what the authors of this paper can tell, the student chose not to answer rather than that not knowing the answer. Since this cannot be proven, his result is included.

Student/Question	**1: A**	**1: B**	**2: A**	**2: B**	**3: A**	**3: B**	**Total**
OpenTTD Lab Student 5	2	2	2	2	2	2	12/13

Table 2: OpenTTD Lab 2, Test Result

The result shown in Table 2 comes from the second OpenTTD lab. The lab was held because too few students attended the first lab. In this lab, students that did not attend

the flipped classroom were offered a chance to get a bonus point on the exam in the concurrent programming course.

3.1.2 Flipped Classroom

This section contains a table of the result from the test students took after the flipped classroom. Worth noting here is that the students took the test and answered the survey from home. This is because it was difficult to make them do it right after the flipped classroom.

Student/Question	1: A	1: B	2: A	2: B	3: A	3: B	Total
FC Student 1	1	1	1	0	2	0	5/13
FC Student 2	3	2	1	0	0	0	6/13
FC Student 3	2	1	1	0	2	1	7/13
FC Student 4	2	1	2	0	2	0	7/13
FC Student 5	3	1	2	2	2	0	10/13
FC Student 6	2	1	2	1	2	0	8/13
FC Student 7	2	1	2	1	1	0	7/13

Table 3: Flipped Classroom, Test Result

3.2 Survey

The result from the survey is displayed through graphs. Each graph covers all the questions from one approach. One section covers the survey results from the OpenTTD lab and another section from the flipped classroom. The results are shown on a linear scale ranging from one to five.

3.2.1 OpenTTD Lab

This section contains graphs of the result from the survey students took after the flipped classroom.

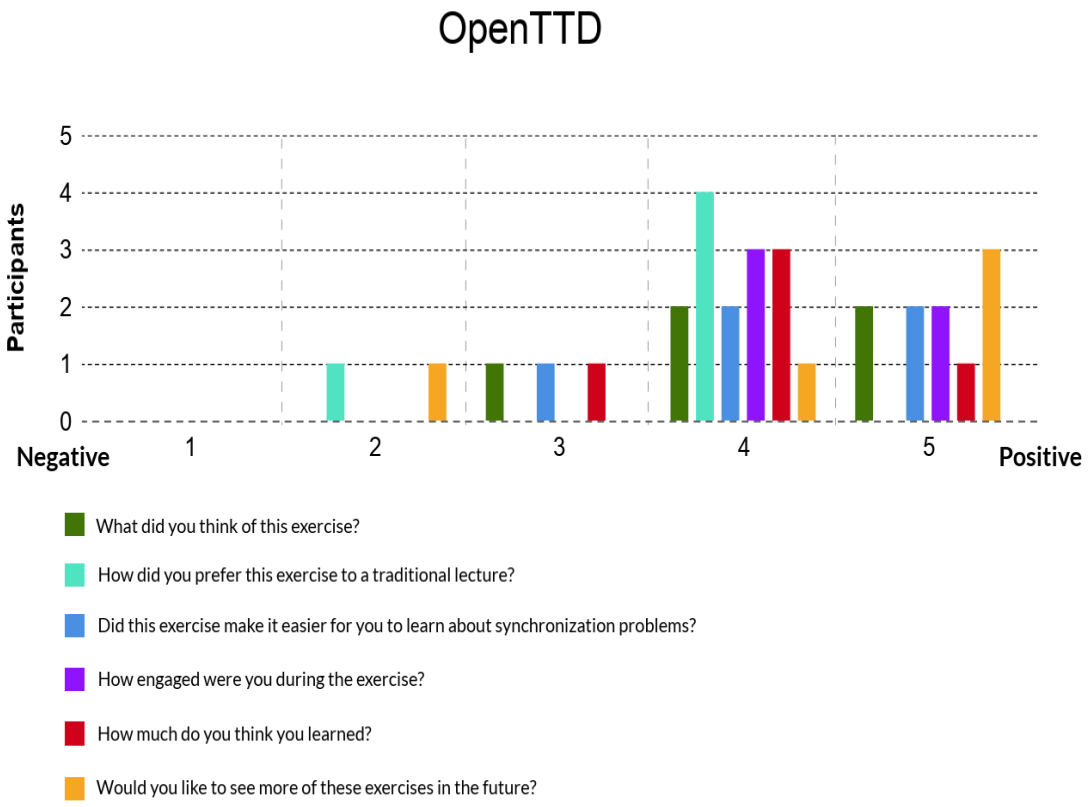

Figure 1: OpenTTD Survey - Result from the questionnaires

The students' comments about the lab (optional)

It was good fun but a bit too much overhead before you reached the actual problem. Too much fiddling with the menus and such. I would have preferred if you got to the actual problem faster.

Interesting way to look at programming, would have been interesting for beginners (like myself) in programming to have a workshop with this.

3.2.2 Flipped Classroom

This section contains graphs of the result from the survey students took after the flipped classroom.

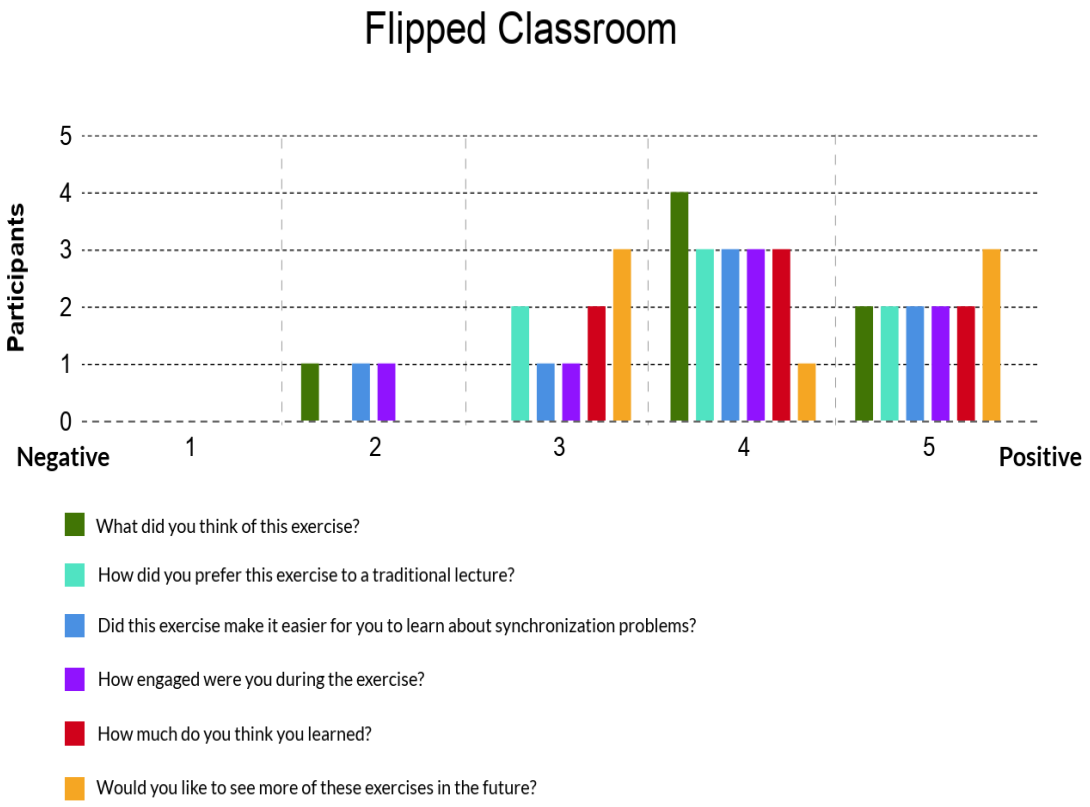

Figure 2: Flipped Classroom - Result from the questionnaires

The students' comments about the flipped classroom (optional)

Really fun and educating idea to do like this, get a much wider view of how thread and other scenario works.

4 Analysis

4.1 Test

This section looks at the differences and similarities between the students' test result from the flipped classroom and the OpenTTD lab. This is done by comparing the result from each question and the total result. The comparison is done by looking at the average and the median. As the "A"-part of each question is about describing the given synchronization problem and the "B"-part of each question is about preventing or solving the problem, this section will include how each approach did with this.

4.1.1 Average

Approach/Question	**1: A**	**1: B**	**2: A**	**2: B**	**3: A**	**3: B**	**Total**
OpenTTD	1,6/3	1,4/2	1,2/2	0,8/2	1,2/2	0,6/2	6,8/13
OpenTTD*	2/3	1,75/2	1,5/2	1/2	1,5/2	0,75/2	8,5/13
Flipped Classroom	2,14/3	1,14/2	1,57/2	0,57/2	1,57/2	0,14/2	7,14/13

Table 4: Average Results - From each test question and total result

Table 4 shows the average result from a more general viewpoint, covering the result from each question and the total result. There is not a big difference overall between the two approaches. The flipped classroom does a little better than OpenTTD, but the difference is too small to be reliable. OpenTTD does a little better when excluding the student that did not answer any questions, but again the difference is too small to be reliable.

Approach/Question	**1: A**	**2: A**	**3: A**	**Total**
OpenTTD	1,6/3	1,2/2	1,2/2	4/7
OpenTTD*	2/3	1,5/2	1,5/2	5/7
Flipped Classroom	2,14/3	1,57/2	1,57/2	5,29/7

Table 5: Average Results - Understanding the synchronization problems

Table 5 shows that the students from the flipped classroom have better average results when it comes to describing synchronization problems. However, when excluding the student that did not answer the difference is very small.

Approach/Question	**1: B**	**2: B**	**3: B**	**Total**
OpenTTD	1,4/2	0,8/2	0,6/2	2,8/6
OpenTTD*	1,75/2	1/2	0,75/2	3,5/6
Flipped Classroom	1,14/2	0,57/2	0,14/2	1,86/6

Table 6: Average Results - Solving the synchronization problems

Table 6 shows that students that did the OpenTTD lab hold a better average when it comes to solving/preventing synchronization problems, especially when it comes to starvation(2: B) and deadlocks(3: B).

*Excluding the student that did not answer any questions.

4.1.2 Median

Approach/Question	**1: A**	**1: B**	**2: A**	**2: B**	**3: A**	**3: B**	**Total**
OpenTTD	2/3	2/2	2/2	1/2	1/2	0/2	8/13
OpenTTD*	2/3	2/2	2/2	1/2	1,5/2	0,5/2	8/13
Flipped Classroom	2/3	1/2	2/2	0/2	2/2	0/2	7/13

Table 7: Median Results - From each test question and total result

Table 7 shows that there is not much difference between each group when looking at the median. Seeing as few students attended, it is hard to use the median to compare the differences. What we can get a hint of here is that the B questions tend to have a higher score by students from the OpenTTD labs. Table 9 gives a better overview of this.

Approach/Question	**1: A**	**2: A**	**3: A**	**Total**
OpenTTD	2/3	2/2	1/2	5/7
OpenTTD*	2/3	2/2	1,5/2	5/7
Flipped Classroom	2/3	2/2	2/2	5/7

Table 8: Median Results - Understanding the synchronization problems

Table 8 shows little difference, the flipped classroom approach has a slight advantage on 3: A but it is too small to make the assumption that the flipped classroom approach helps students understand the different synchronization problems better.

Approach/Question	**1: B**	**2: B**	**3: B**	**Total**
OpenTTD	2/2	1/2	0/2	3/6
OpenTTD*	2/2	1/2	0,5/2	3/6
Flipped Classroom	1/2	0/2	0/2	2/6

Table 9: Median Results - Solving the synchronization problems

Table 9 points towards that the students that attended the OpenTTD lab know how synchronization problems can be solved or prevented better than the students from the flipped classroom, especially when it comes to starvation (2: B).

4.1.3 Exceptional Student

The result from the student that attended the second OpenTTD lab sticks out with an exceptional result, 12/13. If this study had many more students available to gather data than this would not be worth noting. Since this is not the case it is worth noting that the result of this study can have effects on the average result. This is why looking at the median is important, even if the median is not very reliable either since too few students attended. Still looking at the limitations of the students that can be included makes it important to look at the average and the median.

Comparing the two best test results again indicates that the flipped classroom was not as effective at teaching the students how to solve or prevent synchronization problems. The best result (12/13) missed one point on the first question (1: A), while the second best result came from a student that attended the flipped classroom and got 10/13 points. That student missed one point on question 1: B (solve/prevent race conditions) and got zero points on the last question (3: B - solve/prevent deadlocks).

4.2 Survey

In this section we look at the students survey results from the OpenTTD lab and the flipped classroom to compare the two and see the overall experience from the two approaches.

4.2.1 Comparison

The students did not attend both OpenTTD lab and the flipped classroom, which makes this more of a comparison of results rather than a direct comparison between the two. The graphs in this section, unlike those in the result section, are displayed in percent, to make it a more fair comparison as the number of students are not the same for the two different approaches.

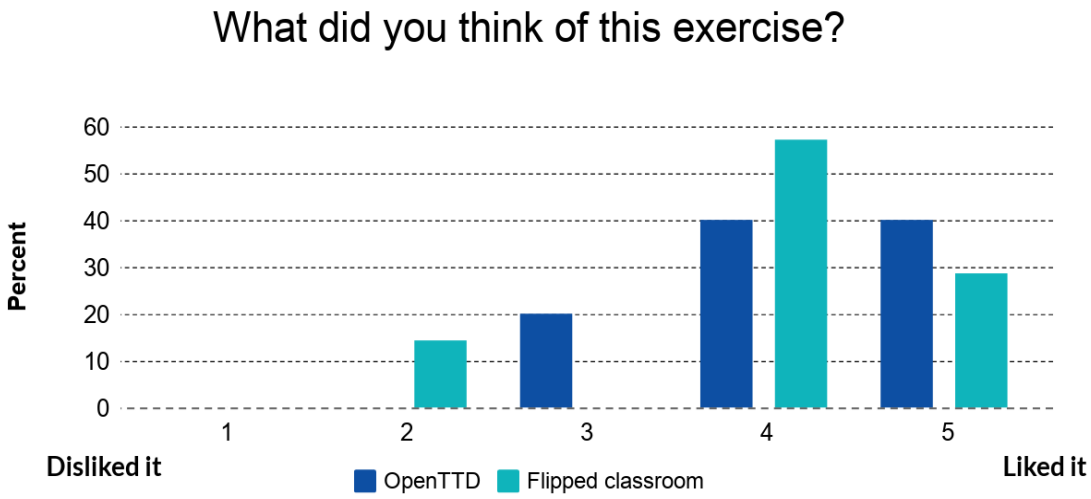

Figure 3: Comparison - What did you think of this exercise?

OpenTTD lab Average: 4.2
Flipped Classroom Average: 4

Figure 3 shows that students who attended the OpenTTD lab liked it more than students attending the flipped classroom.

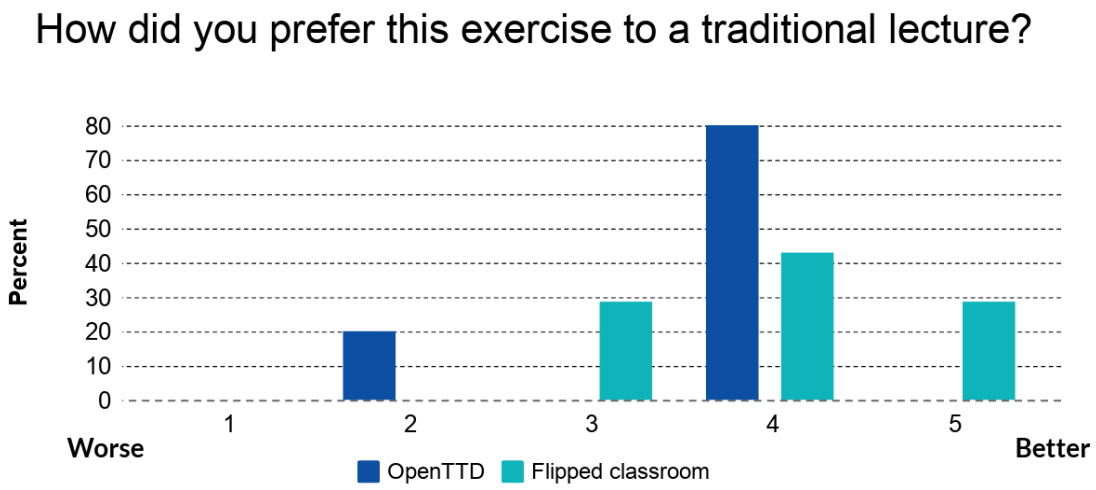

Figure 4: Comparison - How did you prefer this exercise to a traditional lecture?

OpenTTD lab Average: 3.6
Flipped Classroom Average: 4

Figure 4 shows that students from both the OpenTTD lab and the flipped classroom prefer the exercise they attended more than a traditional lecture when it comes to learning about synchronization problem, and the flipped classroom slightly more so.

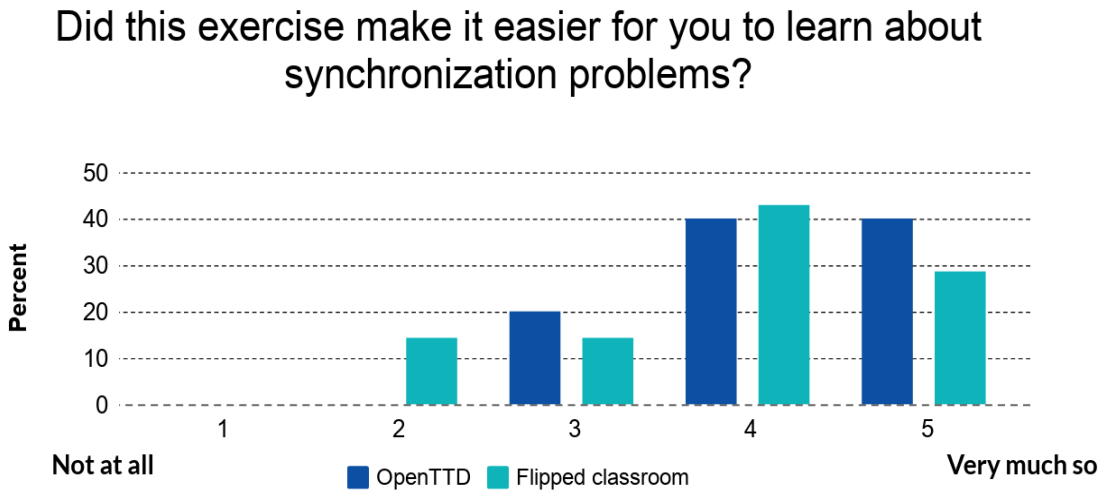

Figure 5: Comparison - Did this exercise make it easier for you to learn about synchronization problems?

OpenTTD lab Average: 4.2
Flipped Classroom Average: 3.86

Figure 5 shows that students who attended the OpenTTD lab felt like they had it easier when learning about synchronization problems, but the results from the test show that the students from the flipped classroom have a better average result when it comes to explaining the problems.

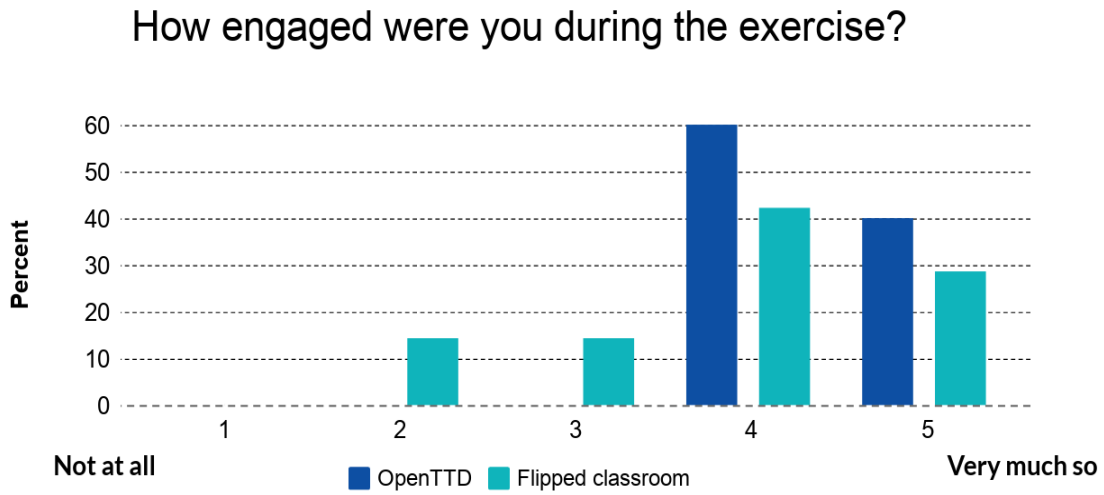

Figure 6: Comparison - How engaged were you during the exercise?

OpenTTD lab Average: 4.4
Flipped Classroom Average: 3.86

Figure 6 shows the biggest difference when it comes to results from the survey. The students from the OpenTTD lab felt that they were more engaged during the exercise.

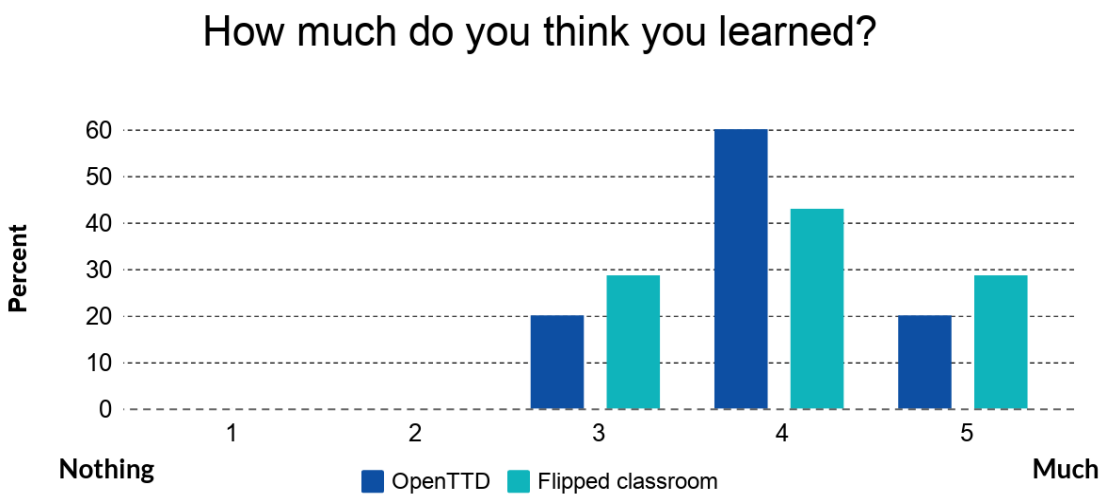

Figure 7: Comparison - How much do you think you learned?

OpenTTD lab Average: 4
Flipped Classroom Average: 4

Figure 7 shows that the students from the OpenTTD lab and the flipped classroom thought they learned roughly the same, although the flipped classroom had more students who chose 5s and 3s. Looking at the result of the test shows that the students from the flipped classroom are better at explaining the synchronization problems, while the students from the OpenTTD lab are better at solving them.

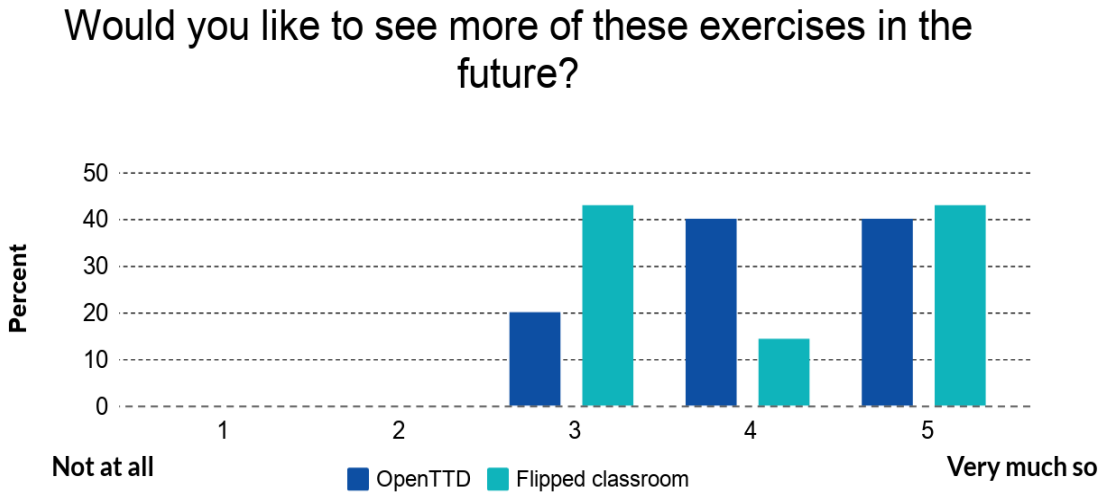

Figure 8: Flipped Classroom - Would you like to see more of these exercises in the future?

OpenTTD lab Average: 4.2
Flipped Classroom Average: 4

Figure 8 shows that the students from the OpenTTD labs and the flipped classroom are interested in seeing more of these exercises in the future and slightly more so from the OpenTTD labs.

4.2.2 Overall Preference

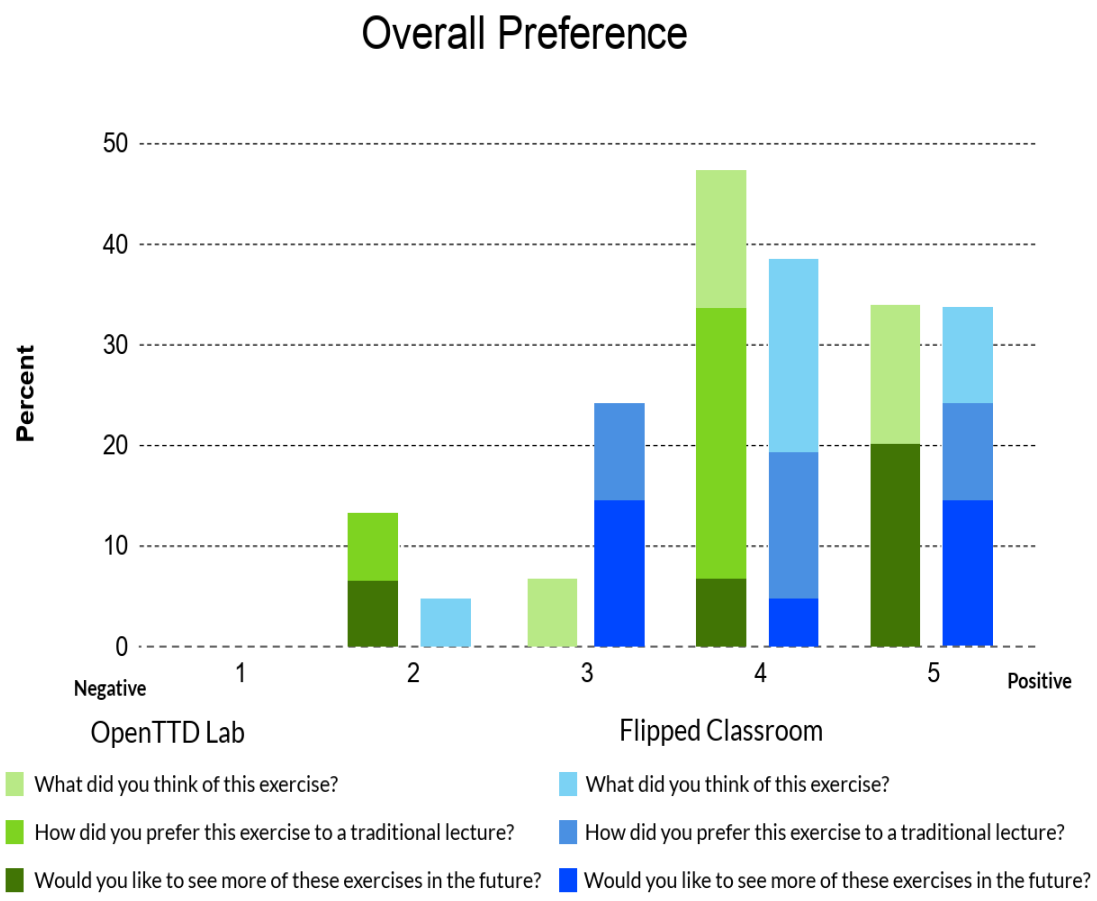

Figure 9: Overall Preference - Compares results from different questions to see what the students preferred

OpenTTD lab Average: 4.13
Flipped Classroom Average: 4

When combining all the results from the questions that deal with the students' preference, we see that the overall preference is roughly the same. The OpenTTD have a higher percentage of 2s which drags it down a bit, but with a lower percentage of 3s and a higher percentage of 4s brings the methods to overall similar results.

4.2.3 Overall Engagement

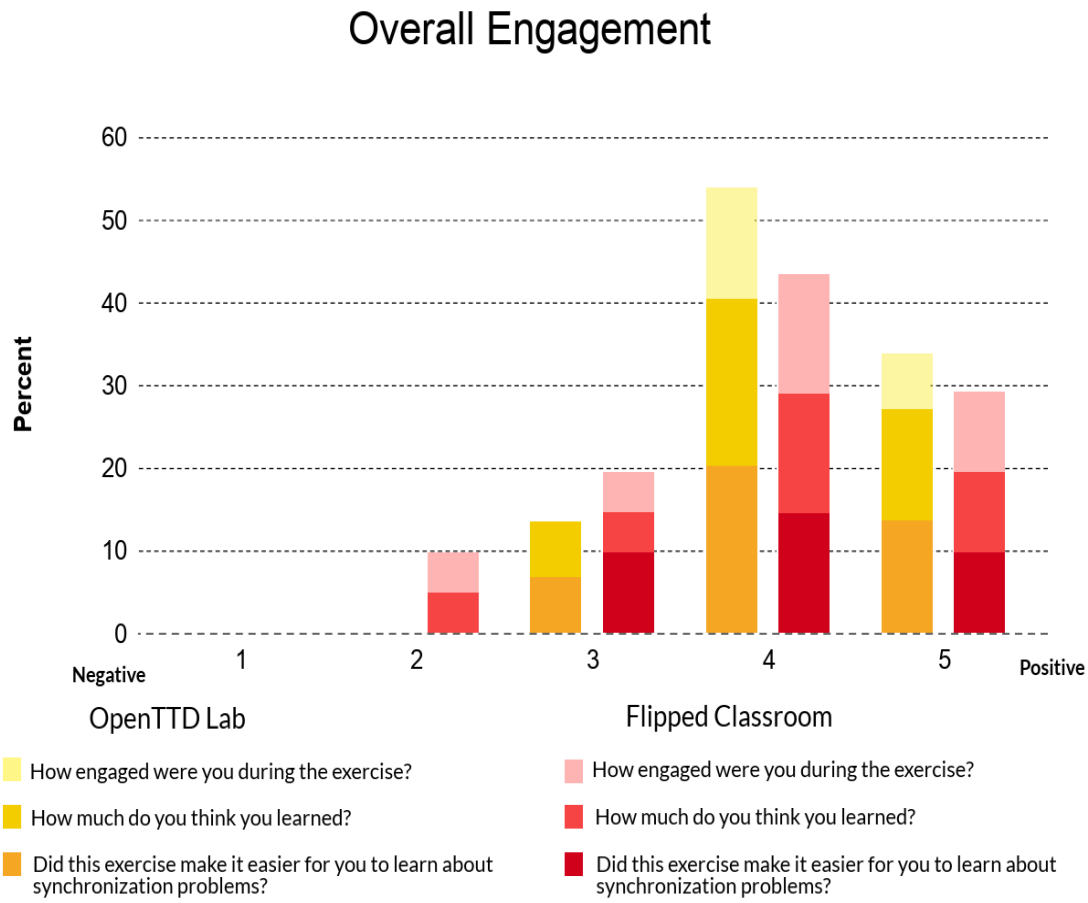

Figure 10: Overall Engagement - Compares results to see overall engagement from relevant questions

OpenTTD lab Average: 4.2
Flipped Classroom Average: 3.9

Figure 10 shows a combination of the survey results from questions where in some way the students' engagement can be gauged, and by looking at the graph and the average we can see that the engagement from the students during the OpenTTD lab is higher.

5 Discussion

In this section, the authors of this paper will discuss the results and how they compare with each other, how the test and survey could have been improved and what difficulties there are. What the student thought and which questions they asked. What difficulties that arose from the different teaching approaches and how they can be improved.

A clear downside to keep in mind is that there are too few students that volunteered. This makes the answers to the research questions weak or inconclusive.

5.1 Test

5.1.1 Comparing the Results

From looking at the analysis, it can be seen that OpenTTD have better results when it comes to solving or preventing synchronization problems. While there is not enough data to be strong enough proof, it does point in that direction. With one point difference in median and 1,64 difference in average, where the maximum score is 6. This is not strange as the OpenTTD lab focuses more on having the student solve the problems rather than on what the problems are. Still the difference between how well students understand the problems is quite small (average of 5.29 compared to 5, both with a median of 5). This is only true if not counting the student that did not answer any questions on the test. As it is unlikely that the student did not know anything, it is only most reasonable to assume that he decided not to answer. If it is not the case then it is still reasonable not to put much weight on that student's result, as all students should have a certain understanding of these problems as the lectures from the course had already talked about the synchronization problems. Meaning that all students should know something, more than that is the fact that according to the survey question "How much do you think you learned?" the lowest score for the OpenTTD lab is a 3, which means that all the students attending the lab learned at least something.

Looking at the best results it is still supported that the OpenTTD lab have better results when it came to preventing/solving synchronization problems, as mentioned previously in the analysis. The problem here is that the best result comes from the second OpenTTD lab, where he might have tried harder as he is given an extra point for attending the lab. This is different from the first lab where they do not receive an extra point for attending. While the student's test result does not determine whether the student gets an extra point or not, it was unfortunately overlooked at the time and as such the authors of this paper do not know if this is the case.

With all of this in mind, the research from this study still points towards that using the OpenTTD lab gives better student results when it comes to preventing/solving synchronization problems. The study may also point towards the flipped classroom, giving better results about understanding/describing the problems, but this is much more inconclusive.

5.1.2 Difficulties and Possible Improvements

The limitation of the authors' pedagogical education and experience has caused difficulties in creating the test and correcting the students' answers. As mentioned in the method sections, guidelines were created for correcting the test, however, without previous experience, it is hard to determine if a point should be awarded when the answer is unexpected or weak/unclear. A common mistake the students made on the test was that they used the word process even though they meant thread. To make the results more even only one of the authors corrected the results, he had to decide whether or not to give the students a point when this kind of situations occurred.

One way to improve the test is to make each question mandatory on the Google Forms, this way it is safe to say whether or not the students know the answer. This prevents students from skipping the test. How the students should answer each question can also be made more clear. The only thing to indicate the length of the question is the number of points the question is worth but never describes how well to answer the questions.

5.2 Survey

5.2.1 Comparing the Results

From looking at the graphs and the averages in the analysis we can see that the result from the OpenTTD lab tends to be slightly higher than the results from the flipped classroom. The only question where flipped classroom gets a higher average is on the question "How did you prefer this exercise to a traditional lecture?". This question is a bit different as it asks the students to compare the approach to something they have already experienced. One likely reason for the flipped classroom to have a higher average in this question is that it is much more similar to a traditional lecture than the OpenTTD lab.

The two questions where the average differs the most in OpenTTDs favor are "Did this exercise make it easier for you to learn about synchronization problems?" and "How engaged were you during the exercise?". The reason for these differences here is most likely caused by the fact that the OpenTTD lab is much more of a practical exercise where you are placed in scenarios where you have to learn and solve synchronization problems to finish the exercise.

From looking at the overall preference in the analysis we can see that the OpenTTDs result when it comes to student preference in general, is slightly higher than that of the flipped classroom. The reason for this could be that the OpenTTD lab is an approach for game-based learning and games for most students are more enticing.

From looking at the overall engagement in the analysis we can see that the average result for the OpenTTD is higher here as well, and also most likely has something to do with that OpenTTD is a game which usually requires the students to be more engaged during the exercise to get through it.

5.2.2 Difficulties and Possible Improvements

When using a quantitative method such as a survey that consists of questionnaires you need to get a fair number of participants in order for it to yield more reliable results, which in this case is lacking. This has led to overall weaker results. To improve this it would have to be ensured that a sufficient number of participants would participate beforehand, and if not, turn to a qualitative method such as interviews to get more detailed answers from a fewer number of participants.

5.3 OpenTTD Lab

One problem that occurred during the lab was that an older student had problems relating to the game, not being from the virtual generation. This is not strange as game-based learning is aimed towards the generation that have interest in games. Apart from this, at least one student thought that it took too much time to get to the problem, spending too much time on game mechanics. A way to improve the lab to prevent this is to give a better introduction to the game. In the lab that was held the students were thrown into the game without an introduction.

A few problems were observed during the OpenTTD labs. For example, difficulties in finding how to reload a saved game (scenario). There were graphical problems that made the save /load icon disappear when they use window sized screen.. The students also had difficulties understanding how the signals work. One of the students used the wrong signal. Many problems were solved fast as they got help. All of the students that attended the OpenTTD labs thought that it was interesting and were clearly engaged during the labs. As the labs showed some difficulties, not everyone preferred this approach compared to a normal lecture. This may change if the planning and the execution of the lab are improved.

To improve the lab the authors of this paper could have made contact with the creator (Dr. Robert Marmorstein) of the OpenTTD lab to see if he could give tips or other instructions on how to hold the lab. As he has experience holding the lab and experience as an educator, he is likely to have useful feedback to come with.

Apart from all the problems the students thought the lab was fun and interesting. One student commented on the survey that he/she thought it is an interesting way of looking at programming and it would be interesting to have a workshop using OpenTTD.

5.4 Flipped Classroom

The flipped classroom was held by an experienced educator, which makes it difficult for the authors of this paper to comment much on how the flipped classroom was held or how it could be improved. What can be said is that the students sat in groups like they are meant to. During the flipped classroom, most of the discussions were about synchronization problems, but there was also some sidetracking.

One downside of the flipped classroom that took place was that the teacher was busy and could not join the students in their discussion as much as it is normally meant to when

using the flipped classroom model [5]. As the advantage of the flipped classroom is to let the students spend more time discussing both with each other and with the teacher. This could however not be helped.

6 Conclusion

It can be concluded that the students' results points in favor when using the OpenTTD lab approach (average of 3.5 compared to 1.86, the median of 3 compared to 2) to teach the theoretical aspects of how synchronization problems can be solved or prevented. When it comes to describing each synchronization problem the flipped classroom may be better (average of 5.29 compared to 5, both with a median of 5), but dealing with such small quantitative data makes it difficult to draw a conclusion. The data support that OpenTTD is a better approach for preventing/solving problems is stronger than the flipped classrooms data that supports that the approach is better for describing each problem. Keeping in mind that the maximum points for solving/preventing synchronization problems are 6. So the difference of one more point in the median and 1.64 as average is sufficient evidence that the OpenTTD lab gives better student results when learning about how to solve or prevent synchronization problems. While the difference of 0.29 for describing each problem is too weak to draw any conclusions at all.

When looking at the result, analysis and discussion, we can draw the conclusion that the OpenTTD lab is the more preferred approach of the two, but not by much. The OpenTTD lab has an overall preference average of 4.13 while the flipped classroom has an average of 4. This is a very small difference and is not reliable due to the small amount of quantitative data. The flipped classroom, however, is more preferred over a traditional lecture compared to the OpenTTD lab.

The conclusion we can draw about which of the two approaches is more engaging for the students is more reliable as the difference is more substantial, as the OpenTTD lab has an average of 4.2 while the flipped classroom has an average of 3.9, which is more than double the difference than that of difference for the overall preference.

The small data size, however, means that the students' test and survey results mostly points in favor of the OpenTTD lab approach, but the data are not sufficient enough to give reliable answers to the research questions of this paper. It can only point towards what is more likely. To draw a reliable conclusion more students would have to participate in the study.

6.1 Future Work

As the authors of this paper did not find any previous research from the literature review comparing different modern methods for teaching concurrency with each other it is safe to say that there is much to be done. Given the small data size that could be used in this study, a similar study may be needed to verify the findings from this study. As mentioned earlier in this paper the only reliable way to see what approach gives better students' results it to create a test that takes the students' background into account.

To find out what method that works better for teaching students about concurrency it is also important to do many different studies that compare many different approaches. As each model or method does not include any strict guidelines or rules. Meaning that

different approaches can come from the same method. To cover more of this area, it is also relevant to see more research that has been done in this area under different fields, not just from multi-threading. The authors of this paper did not have the time budget to search for research from other fields, but it could lead to valuable data or better insight.

As also mentioned earlier in this paper interviews could also be used to determine what students prefer and make them more engaged. This is a method that could be used in another study to both verify Q2 and Q3 and to gain a better insight into this matter.

References

[1] R. Marmorstein, "Teaching semaphores using... semaphores," *J. Comput. Sci. Coll.*, vol. 30, pp. 117–125, Jan. 2015.

[2] O. team, "About openttd." Available at https://www.openttd.org/en/about, Accessed: 2018-03-15.

[3] J. Bergmann and A. Sams, *Flip YOUR Classroom*, ch. 2. ASCD, 2012.

[4] S. V. Moore and S. R. Dunlop, "A flipped classroom approach to teaching concurrency and parallelism," in *2016 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, pp. 987–995, May 2016.

[5] J. Zarestky and W. Bangerth, "Teaching high performance computing: Lessons from a flipped classroom, project-based course on finite element methods," in *Proceedings of the Workshop on Education for High-Performance Computing*, EduHPC '14, (Piscataway, NJ, USA), pp. 34–41, IEEE Press, 2014.

[6] M. Alice and S.-S. Carol, *The use of computer and video games for learning*. Learning and Skills Development Agency, 2007.

[7] H. Leroux and C. Exton, "Coope: a tool for representing concurrent object-oriented program execution through visualisation," in *Proceedings Ninth Euromicro Workshop on Parallel and Distributed Processing*, pp. 71–76, 2001.

[8] G. Malnati, C. M. Cuva, and C. Barberis, "Jthreadspy: Teaching multithreading programming by analyzing execution traces," in *Proceedings of the 2007 ACM Workshop on Parallel and Distributed Systems: Testing and Debugging*, PADTAD '07, (New York, NY, USA), pp. 3–13, ACM, 2007.

[9] J. Akimoto, N. Cheng, "An educational game for teaching and learning concurrency," in *Proceedings of the 1st international conference on knowledge economy and development of science and technology,*, (Saitama University, 255, Shimo-Okubo, Sakura-Ku, Saitama, Japan), pp. 34–39, 2003.

[10] R. Morsi and E. Jackson, "Playing and learning? educational gaming for engineering education," in *2007 37th Annual Frontiers In Education Conference - Global Engineering: Knowledge Without Borders, Opportunities Without Passports*, pp. F2H–1–F2H–6, Oct 2007.

[11] R. F. Maia and F. R. Graeml, "Playing and learning with gamification: An in-class concurrent and distributed programming activity," in *2015 IEEE Frontiers in Education Conference (FIE)*, pp. 1–6, Oct 2015.

[12] J. Lönnberg, A. Berglund, and L. Malmi, "How students develop concurrent programs," in *Proceedings of the Eleventh Australasian Conference on Computing Education - Volume 95*, ACE '09, (Darlinghurst, Australia, Australia), pp. 129–138, Australian Computer Society, Inc., 2009.

[13] J. Lönnberg and A. Berglund, "Students' understandings of concurrent programming," in *Proceedings of the Seventh Baltic Sea Conference on Computing Education Research - Volume 88*, Koli Calling '07, (Darlinghurst, Australia, Australia), pp. 77–86, Australian Computer Society, Inc., 2007.

[14] J. Lönnberg, L. Malmi, and M. Ben-Ari, "Evaluating a visualisation of the execution of a concurrent program," in *Proceedings of the 11th Koli Calling International Conference on Computing Education Research*, Koli Calling '11, (New York, NY, USA), pp. 39–48, ACM, 2011.

[15] R. Marmorstein, "Cmsc 442: Operating systems." Available at https://marmorstein.org/~robert/Fall2017/cs442.html, Accessed: 2018-05-11.

[16] R. Marmorstein, "Lab 1: Semaphores." Available at http://marmorstein.org/~robert/Fall2017/cs442/Lab1-Trains.pdf, Accessed: 2018-05-11.

Appendices

Appendix A: OpenTTD Lab Instructions

The OpenTTD lab instructions starts at the next page as it is an exact copy of the pdf file used during the lab.

Game-based learning using OpenTTD

Originally created by Dr. Robert Marmorstein at Longwood University

---

The dictionary.com definition of “semaphore” is “any of various devices for signaling by changing the position of a light, flag, etc”(http://dictionary.reference.com/browse/semaphore?s=t as of Sept. 2015). Semaphores were used by ships and trains as signals to avoid collisions. In computer science, we use the word “semaphore” to mean something different – a special data structure that prevents “collisions” in concurrent code.

Semaphores in computer science don't use lights, flags, and so forth (they typically usually use an integer variable and a “wait list” instead), but in many ways they are very similar to the semaphores used by trains. In this lab, we are going to use a train simulation game called OpenTTD (an open source close of transport tycoon deluxe) to build some intuition about semaphores and how they work.

Getting Started

To get started download the OpenTTD Lab zip file from here https://drive.google.com/open?id=1XweDiPkqk5BkvFePqZ4H1NsRfZ1mNVie, and extract it to a place where it can be easily accessed.

To start the game from the openttd.exe, when started click on “Load Game”. Then select race_condition.sav from the exercise folder, which will be needed for the first exercise.

If the NewGRF file is missing download it by clicking on “Find missing content”. Choose the first one and press download.

When the game loads, you should see a map which contains two train tracks. One is a circular train track. The other is a linear track which crosses the circle. You should see an oil train operating on the linear track.

What you need to know

You can zoom in and out using the plus and minus keys on your keyboard or the scroll on the mouse. You can click and drag with the mouse or use arrow keys to scroll around and look at the various elements of the game. Sometimes you may find it difficult to see the track or the signals behind the trees and other decorations. You can make the trees invisible by holding down the mouse button on the “gear” icon at the top of the screen and selecting “transparency options”. Click on the trees to hide them. Click again to show them. You can

also make different kinds of buildings, roads, tracks, and signs transparent. If you want to make them invisible and not just transparent, you can click the little rectangle below the icon as well.

The game is currently paused, but when it resumes, the train will pick up oil at the oil well at one end of the track and delivers it to the refinery. Every successful transportation of cargo earns you money that you can use to build more trains, tracks, and so forth.

Near the middle of the track is a small brick building. This is the train depot. When trains break down, they go to the depot for repairs. You can also purchase new trains or change over to a new type of cargo at the depot.

The circular track has four train stations and a depot of its own. One station is near the coal mine. Another is at the sawmill. A third is at the power plant and a fourth is near the forest. There are two trains on the circular track. One picks up wood from the forest and delivers it to the sawmill. The other picks up coal from the coal mine and delivers it to the power plant.

Exercise 1 - Race Condition

To resume the game, you can click the “pause” button at the top-left of the screen. The oil and wood trains should narrowly miss each other.

If you watch the trains for a minute, you'll notice that when the oil train reaches the station next to the refinery, it stops, unloads, and then a little message with the word “Income” and a dollar amount floats up from the train. This indicates that the train has successfully delivered a cargo.

If you let the game run long enough, however, the wood and coal trains will collide. Obviously, this isn't a good thing. The problem is that the trains are using a shared resource – the segment of track in front of the depot. When they both try to use it, they crash.

This is very similar to something that can happen when two threads, which are running concurrently, try to write to a shared resource (such as a variable or data structure). When one process starts to write, the other process overwrites part of the data structure, which causes the program to misbehave or even crash. We call the part of the program which accesses the shared resource the “critical section”. Since the problem only happens if the two processes access the critical section at the same time, we say that the program has a “race condition”. If one of the programs “beats” the other one to the part of the code that accesses the resource, it will finish in time to avoid the problem, but if it “loses the race” they will both be in the critical section at the same time.

In software, we use a construct called a “semaphore” to prevent these problems. In OpenTTD (and with real trains), we use “signal semaphores” instead.

The simplest kind of signal semaphore is a simple light (or flag) that tells a train if it can proceed. When the signal is red, the train must stop. When it is green, the train can proceed.

When two trains need to share a track, the track can be divided into “signal blocks” by placing semaphores along small sections of track. Whenever a train enters the signal block (the space between two semaphores), the semaphores at both ends turn red, preventing any other trains from entering that same section of track. When the train leaves the signal block, the semaphores turn green, which allows other trains through. Signal blocks are very like “critical sections” in a program and the way OpenTTD behaves is exactly like “mutual exclusion” code that protects a critical section in a program.

Reload the race_condition.sav file (by clicking on the “disk icon” and holding the mouse button down, then selecting “Load game” from the menu that comes up). Place semaphores on the track that will prevent the trains from crashing.

To place a semaphore, click on the “train track” icon in the toolbar. Then click on the icon that looks a little like a stoplight. A little window will come up that contains a set of signal lights and flags. Select the tiny one in the bottom left corner.

Note: When you place a signal, it can face either forward, back, or “both” directions. To change orientations, you can click on the signal until you get the orientation you want. Trains cannot pass signals from behind so, for now, make sure that all of your signals are either all point the same way or are set to the “both” orientation.

Try to use as few signals as possible – but be sure the trains cannot collide. If you've done everything correctly, your trains should start to make money and allow you to pay off your “debt” (you start the game with a loan which you can pay off by clicking on the “coins” icon in the toolbar” and clicking “Repay $20,000”). When you get it working, click on the “disk icon” and save your game as “solution_1.sav”.

Hint: You can read more about signal blocks on the OpenTTD wiki: https://wiki.openttd.org/Signals#Block_signals

Exercise 2 - Deadlock

Now load the game named “deadlock.sav” (click on the “floppy disk” icon and hold down the mouse button, then go to “Load Game”).

In this scenario, I have created three tracks that cross each other in a sort of “triangle” shape. To prevent the trains from colliding, I have added semaphores that prevent any two trains from being in the triangle at the same time.

Go ahead and unpause the game. At first it may seem like nothing happens, but if you look closely at the bottom of the screen, you'll see that time is passing (the day and month in the status bar are incrementing slowly).

What's happened is that the trains are all stuck. Train 1 is blocking train 2, while train 2 is blocking train 3, which is in the way of train 1.

In programming, this is called a deadlock. Deadlocks occurs when two or more processes hold resources needed by the other processes. For example, one process may need the printer and the disk to finish. Another process may need the disk and the network to finish. A third may need the network and the printer to finish. If the first process has the printer, the second process has the disk, and the third process has the network, none of them can finish and give back their resources. So they get stuck in a deadlock.

In the case of your trains, the “resources” are the tracks that make up the legs of the triangle. Essentially what happens is this: one of the trains gets partway across the intersection, but then cannot enter the critical section on the far side of the intersection. Let's assume that it's the coal train. The reason the coal train cannot enter the critical section on the far side of the intersection is that the wood train is currently in that critical section. But the wood train is also stuck because it needs to get into the critical section on the far side of the sawmill – and it can't, because the coal train is in that critical section (in fact, the coal train is in TWO critical sections. It's a long train!)

You can temporarily solve the problem by making one of the trains give up the track. Click on train 1 (the oil train on the top right-hand side of the triangle). Then click the curved arrow. This tells the train to reverse directions. It will leave the triangle, allowing the other trains to escape.

This works for awhile – but eventually they will all get stuck again. The problem is that we have too MANY semaphores! Instead of keeping the trains out of the entire triangle, they only protect small pieces of track. This allows a train to get partway through the intersection, blocking other trains.

To fix this, you need to move some of the signals. To do this you will have to add new signals and delete the old ones. If you click on the railroad track icon, and then click on the “traffic light” icon, you can enable the “bulldozer tool”. Clicking on the bulldozer and then on some signals allows you to remove those signals.

See if you can figure out how to get the trains moving again without allowing them to collide. Don't forget you can always reload to start over from the beginning.

When you get it working, click on the “disk icon” and save your game as “solution_2.sav”.

Exercise 3 - Starvation

Even when a system is not deadlocked, it is possible for a thread to become blocked for long periods of time (or even forever) while other threads take turns monopolizing a resource it needs. This is called “starvation”.

Load “starvation.sav”. This scenario looks a lot like the first activity, but when you unpause the game, you will only see two trains: the oil train and the coal train. Where is the wood train? It's in the depot. If you click on the little red brick depot, you'll see that the wood train is

patiently waiting there. If you click on it, you'll see it's running and wants to get to “Frindinghead Woods”, but it can't.

The reason for this is that the circular track is one critical section – and the coal train and oil train keep “taking it”. This means that the wood train can never get in. This looks a little like a deadlock, but if you wait long enough, the coal train will eventually go to the depot, which frees the wood train. Then it's the coal train's turn to starve!

Starvation often happens when a critical section is too large or when there are many processes running. When either of these happens, slower threads tend not to get the critical section very often and can “starve”.

Fix the problem by adding semaphores to split the critical section up into smaller pieces. When you get it working, click on the “disk icon” and save your game as “solution_3.sav”.

Efficiency(Optional)

The train system you have now should work without collisions or deadlocks – but it's pretty inefficient. If you pay really close attention to the wood train you'll see that it spends a lot of time waiting. See if you can improve the efficiency of the system. You can add (or remove) semaphores, add additional track, or even add additional trains. See if you can find a system that keeps everything moving except when it's actually loading or unloading cargo. Save your result as “Solution4.sav”

After you’re done with the exercises please take this short test and survey. The survey can be found here: https://goo.gl/forms/uiy9uf0z6ijZw2Dh1

Appendix B: Flipped Classroom

Info

The following information is translated from Swedish to English.

We will have 4 flipped sessions during the semester: this is how they work:

- You are to read all material and watch all video lectures (link to external resources) at home before coming to the scheduled session.
- The first hour you will be divided into groups of 4 people in each group. You will be given a few quiz-questions that you will discuss in the group. I will be there to answer questions and give consultation.
- The next lecture hour we discuss together the answers to the quiz questions.
- You will be given one day to send in the group's answers on Its L.

Attendance and submission of each quiz is NOT mandatory but will give bonus points as an addition on the exam when you reach ¿= 30 on the exam or re-exam:

- 0.5 p for attendance (only for flipped classroom lectures).
- 0.5 p for submitted answers.
- So 1 p bonus for each FC, on the exam.

Attendance is highly recommended!

FC 1 Material

These videos must be watched at home before coming to the lecture.

- Dining Philosophers (YouTube video): Problem and Solution, Watch this one as well,
- The Sleeping Barber: Problem description, This one as well, With solution

Code tips (not mandatory):

- Dining Philosophers (The link does not work but will be fixed).
- The Sleeping Barber

FC 1 Quiz

Synchronization - Classical problems
Write your answers in a docx, ppt, pdf or normal text file and upload it on Its L, one submission per group and do not forget to include all group members when submitting the on Its L.

Each group will work with one of the following cases.

A. DINING PHILOSOPHERS
B. SLEEPING BARBER

The groups and cases will be decided during the lecture.

QUESTIONS

Consider the following synchronization concepts.

- Deadlock
- Livelock
- Race condition
- Starvation

1. Which are the shared resources?
2. Which of the synchronization problems (a) to (d) according to above are represented in the case (A or B)?
3. Can the scenario (A or B) look like a Writer/Reader-problem? Motivate both for a yes- or no answer.
4. Give suggestion of a brief solution and mutex for the scenario. Describe the suggestion with words or using pseudocode (or both) and indicate which classes (with their responsibility), numbers or semaphores and mutex and are required for in the solution. You do not need to program but you can if you want to.

Good luck!

Farid Naisan,