Exploring the Concept of Abstracting and Visualizing Formal Logic Proofs in Games

Development and Evaluation of Gentzen's Quest

Bachelor thesis

JOHANNES BACKLUND, LUDVIG EKMAN, MAGNUS HARRYSON,
JOHAN RONNÅS, ROBIN ÅSTEDT, JONATAN ÖBERG

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2018

Bachelor's thesis 2018

Exploring the Concept of Abstracting and Visualizing Formal Logic Proofs in Games

Development and Evaluation of Gentzen's Quest

Johannes Backlund, Ludvig Ekman, Magnus Harryson, Johan Ronnås, Robin Åstedt, Jonatan Öberg

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2018

Exploring the Concept of Abstracting and Visualizing Formal Logic Proofs in Games
Development and Evaluation of Gentzen's Quest
Department of Computer Science and Engineering
Johannes Backlund, Ludvig Ekman, Magnus Harryson, Johan Ronnås, Robin Åstedt, Jonatan Öberg

Supervisor: Andreas Abel, Department of Computer Science and Engineering

Bachelor's Thesis 2018
Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX with a template made by David Frisk
Gothenburg, Sweden 2018

Abstract

The subject addressed in this thesis is the evaluation of the concept behind the game Gentzen's Quest, but also to present Gentzen's Quest. The intent behind the concept is to explore a new abstraction level via graphics for doing formal logic proofs targeted towards the general populace. A pre-study was made which concluded this concept was original.

The priority in making Gentzen's Quest was to make it easily understandable when interacting (i.e very little prior knowledge should be needed to advance in the game) and fun as to encourage the player to continue.

A simple user-testing study was made to evaluate this, and the conclusion reached was that there is potential for further development and it was a working game concept. However a few parts of the game, namely those introducing rules and concepts in formal logic proofs, were difficult to abstract and consequently hard to understand by users as well as enjoy.

An essential part of the game concept is that it corresponds to proof construction in formal logic. The connection between the graphical abstraction and the formal logic is presented and argued for in the thesis.

Keywords: Logic proofs, natural deduction, proof calculus, propositional logic, smartphone game

Sammandrag

Denna tes ämnar att utvärdera konceptet bakom spelet Gentzen's quest och att presentera Gentzen's Quest. I grunden är det bakomliggande konceptet att utforska en ny grafisk abstraktionsnivå för att göra formella logiska bevis som gemene man ska kunna spela. En förstudie gjordes som kom fram till att konceptet som utvärderas är originellt.

Fokus vid utvecklingen av Gentzen's Quest var att det skulle vara lättförståeligt att interagera (d.v.s. endast lite förkunskap behövs för att man skulle kunna ta sig framåt i spelet) och roligt för att man ska vilja fortsätta ta sig framåt.

Ett enkelt användartest med frågor utfördes och slutsatsen som nåddes var att konceptet är fungerande med potential att utvecklas vidare, dock indikerades det att delar av spelet som introducerar regler och koncept i formell logik var svåra att abstrahera och därav svårförståliga och icke underhållande för användaren.

En kärnaspekt av spelkonceptet är att det motsvarar bevisföring i formell logik. Den grafiska abstraktionens koppling till den formella logiken är presenterad och argumenterad för i denna tes.

Nyckelord: Bevisföring, logiska bevis, mobilspel, naturlig deduktion, satslogik

Acknowledgements

First of all we would like to thank our supervisor Andreas Abel for the game concept and for providing guidance during the project. We also want to thank the testers for contributing valuable feedback to enhance Gentzen's Quest.

Finally we would like to thank Gerhard Gentzen for his contribution to the scientific community, most notably within the knowledge domain of formal logic proofs. The game developed is named in his honour.

Johannes Backlund, Ludvig Ekman, Magnus Harryson, Johan Ronnås, Robin Åstedt, Jonatan Öberg, 2018

Contents

List of Figures **xiii**

List of Tables **xv**

1 Introduction **1**
1.1 Purpose . . . 1
1.1.1 Evaluation Satisfaction Criteria . . . 1
1.1.2 Definition of the Concept . . . 1
1.1.3 Definition of Proof Equivalency . . . 2
1.2 Delimitations . . . 2
1.2.1 Platform . . . 2
1.2.2 Proof Calculus . . . 2
1.3 The Importance of Logic and Critical Thinking . . . 3
1.4 Gamification of Logic . . . 3
1.5 Structure of Thesis . . . 4

2 Theory **5**
2.1 The Logic Implemented in the Game . . . 5
2.1.1 Propositional Logic . . . 5
2.1.2 Natural Deduction . . . 7
2.2 Game Design . . . 9
2.2.1 Mobile Limitations . . . 9
2.2.2 Game Interface . . . 9
2.2.3 Level Design . . . 10
2.2.4 Tutorial Design . . . 10

3 Description of Gentzen's Quest **11**
3.1 Overview . . . 11
3.2 Proposition Design . . . 12
3.2.1 Atomic Propositions . . . 12
3.2.2 Absurdity . . . 13
3.2.3 Operators . . . 14
3.3 Main Views in the Game . . . 14
3.3.1 Gameboard . . . 14
3.3.2 Inventory . . . 16
3.3.3 Sandbox . . . 16

3.4 Rules . . . 16
3.5 Example of Level Gameplay . . . 18

4 Methods **21**
4.1 Implementing the Logic Engine . . . 21
4.1.1 Expressions . . . 21
4.1.2 Rules . . . 21
4.1.3 State Transitions . . . 22
4.2 Ensuring Correctness . . . 22
4.2.1 Evaluating Logical Equivalency . . . 22
4.2.2 Junit Tests . . . 22
4.3 Evaluating the Game by User-Testing . . . 24
4.3.1 Selection of Test Subjects . . . 24
4.3.2 Gameplay Testing . . . 24
4.3.3 Test Procedures . . . 25
4.3.4 Questionnaire . . . 25
4.3.5 Analysis of Test Results . . . 26
4.4 Design Decisions . . . 26
4.4.1 Main Interface . . . 26
4.4.2 Proposition Design . . . 26
4.4.3 Interacting with Propositions . . . 27
4.4.4 Rule Design . . . 27
4.4.5 Level Design . . . 27

5 Results of Game Testing **29**
5.1 Results of User Testing . . . 29
5.1.1 Results of Level Time and Completion . . . 29
5.1.2 Results of the User Questionnaire . . . 31
5.2 Result of Junit Tests . . . 32

6 Discussion and Conclusion **35**
6.1 Evaluating Gentzen's Quest . . . 35
6.1.1 Conclusions From User Testing . . . 35
6.1.2 Validity of Testing Results . . . 36
6.1.3 Logical Correctness . . . 36
6.2 Evaluating the Concept of Gamification of Formal Logic . . . 36
6.3 Further Development of Gentzen's Quest . . . 37
6.3.1 Tutorial Design . . . 37
6.3.2 Level Design . . . 37
6.4 Ethical Implications . . . 38
6.5 Conclusion . . . 38

Bibliography **39**

A Pre-study: A Search for Games with Similar Concept **I**
A.1 Introduction . . . I
A.1.1 Aim . . . I

A.1.2 Limitations I
A.2 Method I
A.3 Result II
A.3.1 Simple Multiple-Choice Questions About Logic II
A.3.2 Calculators for Formal Logic III
A.3.3 Building and Proving Formal Propositions III
A.3.4 Solving Proofs with Sequent Calculus III
A.3.5 Logical Combinations of Transistors IV
A.4 Conclusion IV
A.5 Discussion about the Scope of the Pre-Study IV

B Raw Response Data from User Testing **V**
B.1 Comments from User During Testing V
B.2 Did you understand the game? VI
B.3 Did you enjoy the game? VI
B.4 Other Comments from after the User Completed the Game VI

List of Figures

3.1 The start screen. 11
3.2 A menu with all the levels in the category "Basic: Conjunction". Once a level is completed it turns green in the menu. 12
3.3 The next category - *Basic: Implication* is first locked, but is unlocked when the first category is finished. 13
3.4 The symbols of the game representing atomic propositions. 13
3.5 Graphical representation of absurdity 14
3.6 Visual representation of operators in the game 14
3.7 Let P and Q be represented by the blue and green symbols respectively. Then the graphical abstractions within the game are equivalent to the following expressions: (a): $P \rightarrow Q$, (b): $P \quad (P \rightarrow Q)$, (c): $P \rightarrow Q) \quad ((P \quad (P \rightarrow Q))$. 15
3.8 The icons shown on the gameboard 15
3.9 The different views available once a level has been started. 16
3.10 Visual representation of conjunction rules. 17
3.11 Visual representation of disjunction rules. 18
3.12 Visual representation of implication rules. 18
3.13 Visual representation of absurdity elimination. 19
3.14 Visual representation of law of excluded middle. 19
3.15 Level walkthrough: Implication introduction. 20

5.1 The amount of time on average it took for the testers to complete the levels, categorised by experience. 30
5.2 Percentage of the testers that finished each level 31
5.3 The testers' grade of enjoyment of the game 32
5.4 The testers' grade of comprehension of the game 32
5.5 Results of Junit tests. Green "ok" symbol means that the test passed. 33

List of Tables

2.1 Truth table of conjunction 6
2.2 Truth table of disjunction 6
2.3 Truth table of implication 6
2.4 Truth table of negation 7

5.1 Level number and corresponding level name 30

A.1 Search words used when looking for a game with the concept of mimicking formal proofs. The asterisk (*) is a joker sign, meaning that the word can end in different ways. II
A.2 The names of the applications found that uses multiple-choice questions about logic. They can all be found at the Google Store. III
A.3 The names of the applications where the user can create and try out propositions. They can all be found at the Google Store. III
A.4 The names of the found applications that can be used to prove a proof in formal logic. They can all be found at the Google Store. III
A.5 The names of the applications found that uses logic gates and transistors. They can all be found at the Google Store. IV

1

Introduction

Doing proofs in formal logic and mathematics is often seen as a difficult and tedious process. In contrast, there exists a large number of puzzle games that people enjoy for recreation. The process of constructing proofs in formal logic can be compared to a puzzle game. There exists a clear goal and well defined rules available to the player. The goal of this thesis is to explore this idea through developing a game called Gentzen's Quest.

As a pre-study has shown (see appendix A), there is a lack of exploratory smartphone games in which one can solve formal logic proofs using graphical abstractions.

This suggests that there is an opportunity to improve upon and explore ways for the general populace to engage in formal logic through games. The importance of logic, and the benefits and issues of gamification as tools are explored in chapter 1.3.

1.1 Purpose

The purpose of this thesis is to describe the development of Gentzen's Quest and evaluate it as a proof of concept for a smartphone game in which the mechanics are equivalent to a given proof calculus.

1.1.1 Evaluation Satisfaction Criteria

The concept will be evaluated on the following concepts:

1. The game should be enjoyable.
2. The game should be easily understandable.
3. The game play should be logically equivalent to formal methods of proofs. See definition in section 1.1.3.

1.1.2 Definition of the Concept

The criteria for the explored and evaluated concept are the following requirements:

1. Abstract graphical representation of propositions, rules of inference and logical operators.
2. Pre-defined goals for each proof which can be reached in an exploratory way.

1.1.3 Definition of Proof Equivalency

The game mechanics are considered equivalent to a given proof calculus if the following criteria are upheld:

1. For every possible logical proposition in the given proof calculus there exists a unique representation of it in the game, and every such representation in the game corresponds to a unique logical proposition.
2. For every set of propositions H, the set of derivable propositions D using inference rules defined by the proof calculus is exactly equal to the set of derivable propositions in the game given the start state of only the propositions H, using the supplied in-game functionality.

1.2 Delimitations

In order to achieve what was proposed in 1.1 and given the time limits that the project faced, the scope of the thesis needed to be narrowed and focused.

1.2.1 Platform

The focus is mainly on evaluating the concept of a game such as described in the previous section, and as such it is of low priority for the scope of this project to provide support for more than one computing platform. Due to several factors, such as the personal choice of computing platforms used by the authors, and pre-existing knowledge of programming languages among the authors, Android was chosen as the target computing platform.

1.2.2 Proof Calculus

The proof calculus chosen to be emulated within the game will have a drastic impact on gameplay and the rest of development, and it was important that the proof calculus could be represented as an exploratory puzzle game, as this seemed like a good basis for a game to be fun and engaging.

For this purpose a type of forward reasoning called natural deduction was chosen, described in more detail in section 2. This includes both constructive as well as classical logic, also explained in section 2, as both open up the possibility for challenges of varying difficulty and complexity.

1.3 The Importance of Logic and Critical Thinking

Logic is in essence as defined by Merriam-Webster's dictionary:

> *"A science that deals with the principles and criteria of validity of inference and demonstration: the science of the formal principles of reasoning."* [1]

In other words, logic is a study of how one can trust conclusions made by way of logical reasoning. In this way, logic is not concerned with the substance of reasoning, but instead the method of inference. This is done by making abstractions of the content and breaking down the steps of reasoning to a finite set of rules.

For instance, the logical reasoning from "it is raining and rain makes the ground wet, therefore the ground is wet" can be reformulated in a formal way by using an abstract representation. This representation can consist of letters, for example P and Q, representing the two statements, it is raining and the ground is wet, respectively. The logical reasoning is then represented by: "P" and "P implies Q", therefore "Q".

This example highlights how the abstraction enables the study of the actual steps of reasoning and inference without considering the context. A more detailed description of the structure of formal logic and variations within the science is given in section 2.1.

Developing an understanding of formal logic and logical reasoning is valuable, especially in technical fields, because it has a positive effect on critical thinking [2] [3]. For instance, in a study examining the relationship between performance in propositional logic and computer science, the hypothesis that there is a strong relationship was confirmed [4]. In several technical applications one needs ability for problem solving closely related to logical reasoning. More importantly, the study of formal logic enables one to validate the correctness of ones logical deduction, which could be of value in many different aspects of life.

1.4 Gamification of Logic

Gamification is in essence as defined by Merriam-Webster's dictionary:

> *"Definition of gamification: the process of adding games or game-like elements to something (such as a task) so as to encourage participation"* [5]

A meta analysis about the effectiveness of computer games as tools in education reported that 34 out of the 69 studies showed significant positive effects of computer based games compared to conventional instruction [6]. It also reported that only

one study concluded that conventional instruction was more effective than computer games.

In addition a meta analysis of how mobile devices affect students in teaching and learning conclude that:

> *"Instructional strategies are important for e ective learning with information technology. [..] Researchers must find the 'key' to integrating mobile devices with instructional strategies and ingeniously match the unique features of mobile devices to the resolution of specific pedagogic challenges. Doing so will maximise the impact of those features on learning outcomes."* [7]

Arguably, simply making information available is not a successful way of imparting understanding or knowledge. There is a need for well-thought-out strategies to make sure learning is effective when using information technology.

With digital games being an effective tool in learning, there is an opportunity to present logical reasoning as a game. The structure of proving formulas in formal logic can be compared to that of solving a puzzle. One has a clear goal that should be shown, sometimes from a number of starting points, with a finite set of actions that can be applied. These actions come in the form of *inference rules*, further explained in section 2.1, which are ways to manipulate logical formulas.

1.5 Structure of Thesis

The thesis is structured by first presenting the purpose and the motivation for the project. The applied theory relevant to the thesis is then presented followed by a thorough description of Gentzen's Quest. This description is presented at this point as to provide the reader an understanding of the game which is referenced in the later parts. After the description of Gentzen's Quest the method that was used to implement the game and evaluate it follows. The results of the evaluation is then presented followed by a discussion of the conclusions made from the results as well as the conclusion of this thesis.

2

Theory

This chapter provides much of the necessary theory needed for understanding this thesis, with a focus on logic and game design.

2.1 The Logic Implemented in the Game

Formal logic is divided into multiple systems or branches. The branch used in the game is a form of propositional logic with a *classical* as well as *constructive* natural deduction system. Each of these terms will be further explained and formally defined in this section.

2.1.1 Propositional Logic

Propositional logic is concerned with the study of propositions. A *proposition* or a *declarative sentence* is a statement which could be argued to be either true or false [8]. Examples of propositions or declarative sentences are: "The sun is shining" or "The sum of three and five is eight", as these statements could be argued to be either true or false. However, in the study of propositional logic the interest is not in the content of the statement but instead the relation between different statements. As such the statement can be replaced with for example the distinct symbol P. These types of propositions that consist of a single statement are called *atomic propositions* [8] because they can not be deconstructed into smaller propositions.

Another type of proposition is *absurdity*. Absurdity's formal notation is and is an arbitrary representation that a logical contradiction exists under the current premises.

Propositions can be expanded with the use of so called logical *operators* (or *connectives*) to create more complex propositions. These operators act upon one or two propositions and create a combined proposition. The operators used in propositional logic are: conjunction, disjunction, implication and negation. The definitions of these four operators are as follows:

Conjunction, in layman's terms '*and*', or in formal logical notation ' ', is a binary operator that takes two propositions and makes a new single proposition. For ex-

ample, take the atomic propositions “the sun is shining” and “there are no clouds”, which together with the logical operator conjunction could make the new proposition “the sun is shining *and* there are no clouds”. Which in formal notation could be written as P Q. This new proposition's truth value solely depends on its atomic propositions' truth values. P Q is only true if both P and Q are true independently. The truth values of the logical operator conjunction are shown in table 2.1.

Table 2.1: Truth table of conjunction

P	Q	$P \quad Q$
T	T	T
T	F	F
F	T	F
F	F	F

Disjunction, in layman's terms '*or*', or in formal logical notation ' ', is also a binary operator. Disjunction is used to state that at least one of two propositions are true. A disjunction is false if and only if both propositions in the disjunction are false. The truth values of the logical operator disjunction are shown in 2.2.

Table 2.2: Truth table of disjunction

P	Q	$P \quad Q$
T	T	T
T	F	T
F	T	T
F	F	F

Implication, or in formal logical notation '$\Rightarrow$', is yet another binary operator. It is used to state that one proposition implies another. The meaning of $P \Rightarrow Q$ could be interpreted as “if P is true then Q is as well”. The truth values of the logical operator implication are shown in table 2.3.

Table 2.3: Truth table of implication

P	Q	$P \Rightarrow Q$
T	T	T
T	F	F
F	T	T
F	F	T

Negation, or in formal logical notation '$\neg$', is a unary operator and it is used to state the negation of a proposition. The truth values of the logical operator negation are shown in the table 2.4.

Table 2.4: Truth table of negation

P	$\neg P$
T	F
F	T

2.1.2 Natural Deduction

Natural deduction is a kind of calculus used within propositional logic for reasoning about propositions. In natural deduction there is a collection of inference rules which are used to prove formulas. These inference rules are also the foundation for the game mechanics in Gentzen's Quest. With the help of these inference rules it is possible to, from a given set of propositions (often called *premise* or *hypothesis*), prove other propositions and then finally reach the proposition we wanted to prove, known as the *conclusion*. Below all inference rules required to build up a natural deduction system within *classical* and *constructive* propositional logic will be defined and briefly explained. These definitions are taken from the book *Logic in Computer Science* [8].

The inference rules for conjunction introduction and conjunction elimination are self-explanatory and stated below.

- Conjunction introduction: From P and Q infer $P \quad Q$.
- Conjunction elimination: From $P \quad Q$ infer P or infer Q.

Disjunction introduction shown below should be interpreted as if you know one proposition to be true the disjunction of that proposition and any proposition (true or false) will be true.

- Disjunction introduction: From P infer $P \quad Q$ for any proposition Q.

The reasoning behind the disjunction elimination rule shown below is easier than it might seem. The disjunction tells us that at least one of the propositions (in the example P and Q) are true, the two implications tell us that both P and Q imply R and therefore it does not matter which proposition in the disjunction is true.

- Disjunction elimination: From $P \quad Q$, $P \Rightarrow R$, and $Q \Rightarrow R$ infer R.

The implication introduction rule shown below works by making an sub proof. A hypothesis is made as an assumption (P in the example). If we can from this assumed true proposition P infer any other proposition Q we may infer that P implies Q. The process of making these so called sub proofs are called opening *scopes*. In these scopes (or sub proofs) new scopes can be opened. A scope can be closed at any given time during the proof.

- Implication introduction: If it is possible to infer Q from P infer $P \Rightarrow Q$.

Implication elimination as shown below is easily understood when put in words like it was in the introduction: it is raining (P), rain makes the ground wet ($P \Rightarrow Q$),

therefore the ground is wet (Q).

- Implication elimination: From P and $P \Rightarrow Q$ infer Q.

Given that a proposition leads to both another proposition and that proposition's negation it can be inferred that the initial proposition must be false therefore the negation of the initial proposition can be inferred.

- Negation introduction: From $P \Rightarrow Q$ and $P \Rightarrow \neg Q$ infer $\neg P$.

The negation elimination rule shown below clearly shows how absurdity is the embodiment of a logical contradiction.

- Negation elimination: From P and $\neg P$ infer .

With this knowledge we can change how negation is viewed. Instead of noting the negation of P as $\neg P$ we can instead see it as $P \Rightarrow$.

With absurdity we can introduce it's only rule of inference: Absurdity elimination which is defined below.

- Absurdity elimination: From infer any proposition Q.

With this inference rule and the new way of noting negation we can remove the negation rules. Negation introduction which in the new notation would be " From $P \Rightarrow Q$ and $P \Rightarrow (Q \Rightarrow$) infer $P \Rightarrow$," can be reached by implication introduction by assuming P then using implication elimination twice to get Q and $Q \Rightarrow$, once again using implication elimination to get , and then lastly using implication introduction to get $P \Rightarrow$. Negation elimination follows trivially from its rewritten form: From P and $P \Rightarrow$ infer .

The set of rules described thus far is the set that is used in what is called *constructive logic*. In order to reach *classical logic*, which is the logic which will be implemented in the game, we need one more rule namely *the law of excluded middle* [9]. Below the law of the excluded middle, abbreviated LEM, is defined (this is also written with negation as implication of absurdity).

- Law of the excluded middle: infer Q $(Q \Rightarrow$) for any Q.

The main difference between classical and constructive logic is exactly the concept that LEM describes, namely that any proposition can be assumed to be either true or false.Tthis assumption is not made in constructive logic which makes it "weaker" than classical, meaning there are theorems which can be proven in classical logic but not in constructive [10].

It should be mentioned that there are several inference rules that can be introduced instead of LEM in order to have a system of classical logic. These rules are all derivable from each other and arguably its not necessary to introduce them all [9]. LEM was chosen in the definition of the implemented logic system as it embodies the assumption that any proposition is either true or false.

2.2 Game Design

In order to develop and evaluate an implementation of formal logic as a mobile game, the ideas and concepts in this section were identified as important parts in the game design. Based on these ideas, Gentzen's Quest can be evaluated to determine the value and prospect of gamification of formal logic.

Several of the references in the following sections are to articles from the website Gamasutra, written by individual writers. This means that many of the ideas presented here are based on the writers' opinions and are not necessarily scientifically supported. However, these writers are professionals in their field or have some kind of credentials in game development, indicating that their opinions are of value.

2.2.1 Mobile Limitations

When designing for a mobile device, one of the most important factors is the interface that mobile devices provide. As Scolastici and Nolte writes in their book *Mobile Game Design Essentials*, games on mobile devices are limited in screen size as well as in the average time of a play session [11]. This means that the interface should be designed as to accommodate for the limited screen size and the gameplay should suit shorter play sessions.

2.2.2 Game Interface

Formal logic as a game can give rise to considerable challenges in providing a fun and intuitive experience due to its complexity. In Dalmau's article on game design he identifies a number of key concepts in making a complex game mechanic intuitive to understand [12]. Included among these concepts are

- Representing information through visual symbols.
- Feedback from user actions.
- Consistency throughout design.

The first point, *representing information through visual symbols* refers to the use of symbolic pictures instead of words. Dalmau argues for this by stating that it is language independent and often easier to interpret.

Feedback from user taken actions refer to the game providing acknowledgement and information based on user interactions.

Finally, *consistency throughout design* refers to keeping game elements consistent as to improve the user experience. Dalmau identifies the main points to focus on: colour scheme, typography, dialog design, controls.

Dalmau suggests that by following these strategies and guidelines, one can shorten the learning cycle and make complex game ideas more accessible to the player.

2.2.3 Level Design

There exists a challenge in designing the contents of game levels to be difficult enough to be engaging, but easy enough to avoid being frustrating. Daul describes this challenge in her book [13] by referencing Csikszentmihalyi's *Flow Theory*. Flow Theory states that there is an optimum level of difficulty for enjoyment, and Daul suggests that this theory should be applied when designing levels.

Baron further outlines how Flow Theory can be implemented in game design by discussing four methods that promote the state of flow: [14]

- *Have concrete goals with manageable rules.*
- *Demand actions to achieve goals that fit within the person's capabilities.*
- *Have clear and timely feedback on performance and goal accomplishment.*
- *Diminish extraneous distraction, thus facilitating concentration.*

Baron then concludes that

> *If game developers are able to include design considerations that take these characteristics into account they will drastically improve player engagement*.

2.2.4 Tutorial Design

There is also a problem in how to introduce new game mechanics in an effective way. Ray identifies different characteristics among players in his article on game tutorials, and describes how to design in-game explanations around these characteristics [15].

One of the major differing characteristics that Ray describes is that of "explorative acquisition players", who learn by testing mechanics for themselves, and "modeling acquisition players", who learn by having mechanics demonstrated before using it themselves. Ray then suggests that an effective tutorial should try to cater to both these types of players.

3

Description of Gentzen's Quest

This chapter will describe the features of the developed game Gentzen's Quest, how it looks and how a user would interact with it. If the reader has an interest of further examining the game, Gentzen's Quest is available in the Google Play Store and the source code can also be downloaded, compiled and deployed on Android devices by fetching it from Github[16].

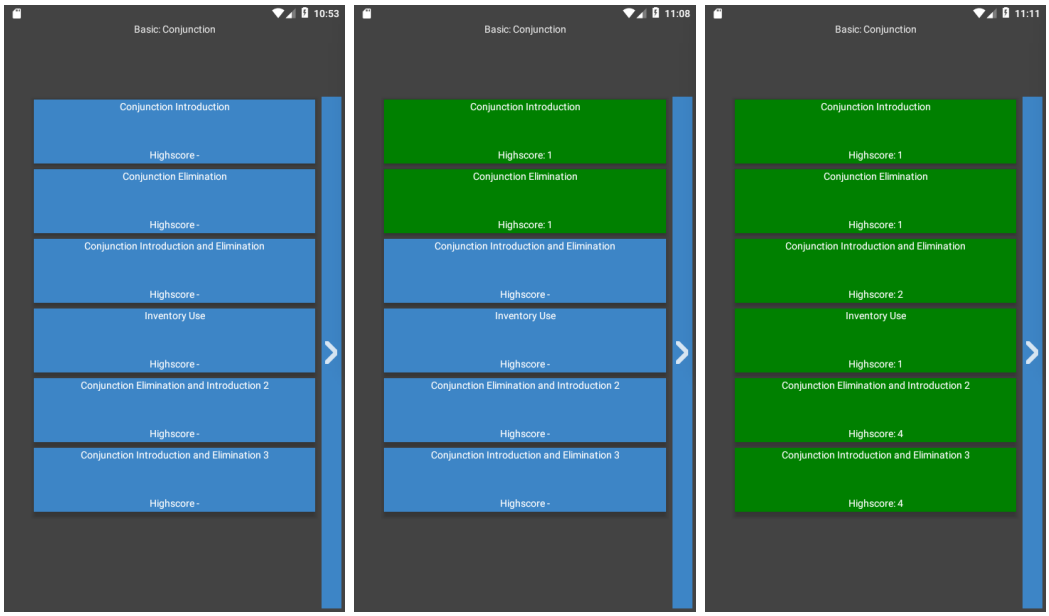
3.1 Overview

This section describes how the player navigates the menus of the game.

When opening the application, the player is met by a start screen as seen in Figure 3.1, where the player can choose either *Start* or *Quit*, to start or quit the game.

Figure 3.1: The start screen.

When selecting *Start* the player enters a menu for level selection, see Figure 3.2. Different colours represents different states of levels as shown in figure 3.3. Grey is for locked levels, blue is for unlocked levels and green is for finished levels.



(a) No levels are completed. **(b)** Two levels are completed. **(c)** All levels are completed.

Figure 3.2: A menu with all the levels in the category "Basic: Conjunction". Once a level is completed it turns green in the menu.

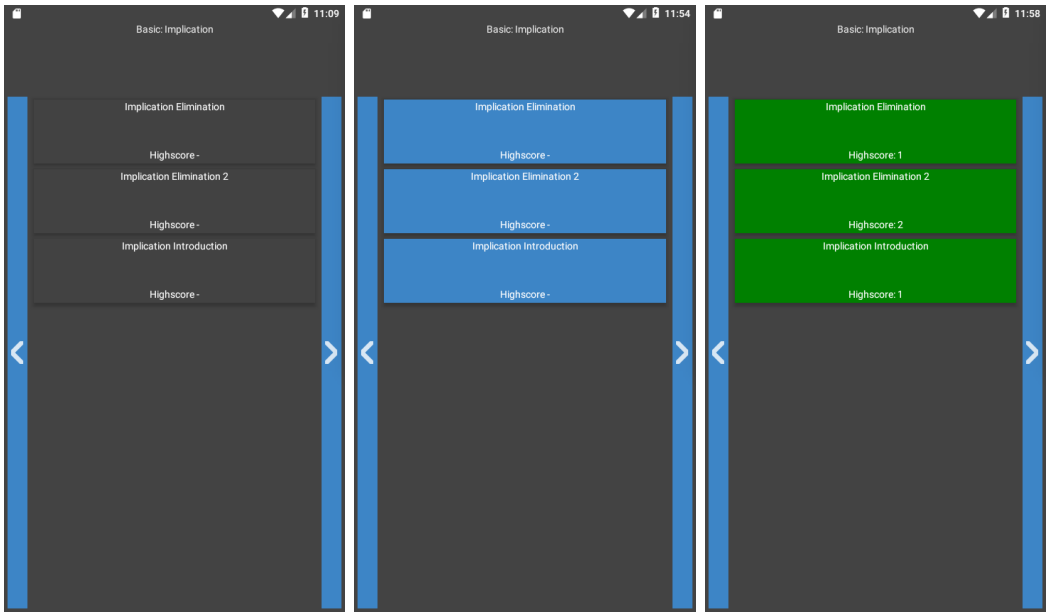
When a certain amount of the levels in a category are finished the next category is unlocked, and the player can swipe or tap the border to the right to get the next category of levels. Since the first categories are introducing the game, the player has to clear all previous levels. For later categories at least 70% has to be finished.

3.2 Proposition Design

In this section the representation of propositions within the game is presented and explained. An explanation of what a proposition is can be found in the Theory chapter, together with other theory about logic required to fully understand this chapter.

3.2.1 Atomic Propositions

Atomic propositions are represented in the game by the symbols in Figure 3.4. In formal logic these are normally described as letters, often P or Q. In this paper we call the graphic representation of atomic propositions *symbols*, in order to distinguish between formal logic and the game.



(a) The second category is locked, as the previous category is not finished.

(b) The previous category is finished, and the category is unlocked.

(c) All levels are completed.

Figure 3.3: The next category - *Basic: Implication* is first locked, but is unlocked when the first category is finished.

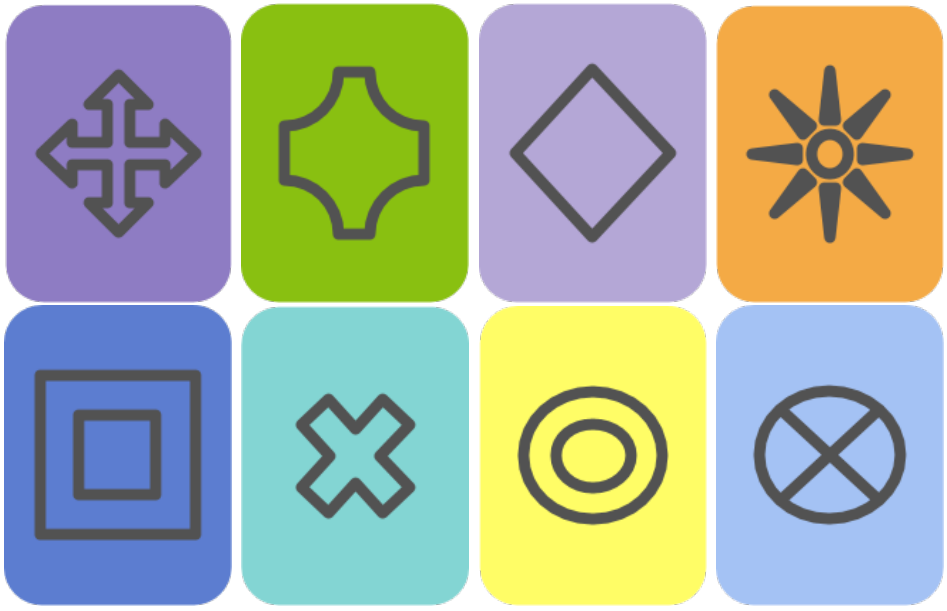


Figure 3.4: The symbols of the game representing atomic propositions.

3.2.2 Absurdity

Absurdity is represented by a solid red block as shown in figure 3.5, and without any other details to make it easily distinguishable from atomic propositions.

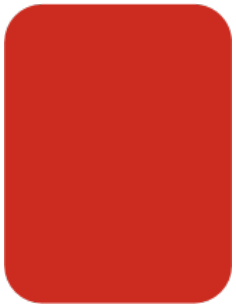


Figure 3.5: Graphical representation of absurdity

3.2.3 Operators

The graphical representation of the three operators in logic calculus described in section 2 are shown in figure 3.6. The operators are placed in between two propositions which are arranged either vertically or horizontally. The first proposition is always to the left or top, and the second is to the bottom or right. Examples of more complex propositions which show this arrangement is shown in figure 3.7.

(a) Implication: →

(b) Disjunction:

(c) Conjunction:

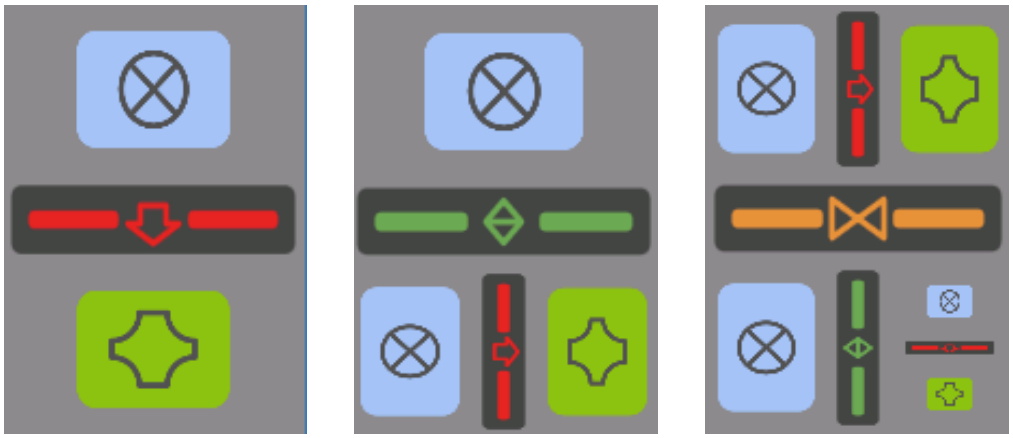
Figure 3.6: Visual representation of operators in the game

3.3 Main Views in the Game

When playing a level the player mainly interacts with three different views, the gameboard, the inventory and the sandbox. This section will describe the functionality of each of these views.

3.3.1 Gameboard

The gameboard is the main view the player interacts with. This view shows the propositions currently being operated on by the player and displays the applicable rules when propositions are selected. Furthermore, the goal of the level is always shown on the gameboard, as well as a few icons representing functionality accessible to the player. These icons are shown in figure 3.8.



(a) A single implication

(b) An implication nested in a conjunction

(c) A series of nested implications, a conjunction and a disjunction

Figure 3.7: Let P and Q be represented by the blue and green symbols respectively. Then the graphical abstractions within the game are equivalent to the following expressions: (a): $P \to Q$, (b): $P \quad (P \to Q)$, (c): $P \to Q) \quad ((P \quad (P \to Q))$.

(a) Trashcan icon

(b) Assumption icon

(c) Inventory icon

(d) Question mark icon

Figure 3.8: The icons shown on the gameboard

One of these icons is the trashcan, which can be used to remove unwanted propositions on the gameboard and reduce clutter. However, all propositions that have been created are saved in the inventory which means that the progress made is not lost.

Another icon is the thought bubble, which enables the player to create a new assumption by opening the sandbox view.

The final icons are the briefcase which opens the inventory, and the question mark which displays the level description.

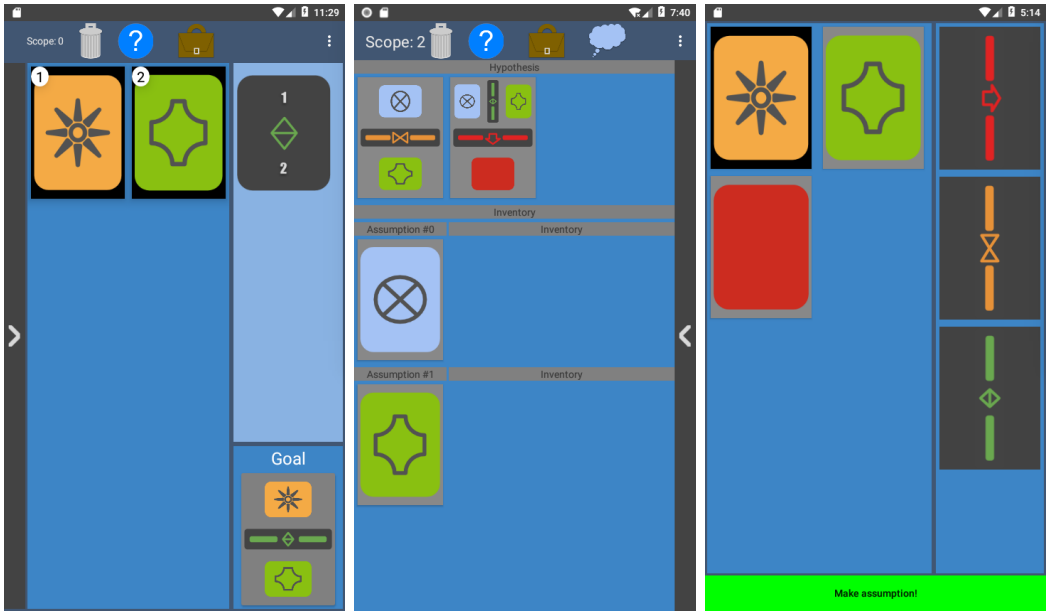
An example of the gameboard when the game is in session is shown in figure 3.9a.

3.3.2 Inventory

The inventory saves all propositions currently available to the player, in other words, the propositions which have been created in the current or previous scopes. The inventory stores no duplicates and sorts the propositions by scopes. The first scope shows the hypotheses while all deeper scopes show the assumption of that scope and all propositions on that scope's gameboard. Propositions can be introduced from the inventory to the current gameboard by a simple tap. An example of the inventory when the game is in session is shown in figure 3.9b.

3.3.3 Sandbox

When making an assumption or when applying some of the rules explained in section 3.4 there is a need for the player to manually create arbitrary propositions. This is done in what is called the sandbox environment. A player constructs a proposition by selecting atomic propositions available and, if necessary, combining these with operators. When the player has created the desired proposition, it is brought to the gameboard by selecting it and pressing the button at the bottom of the screen. In figure 3.9c a screenshot of the sandbox is shown.



(a) Gameboard **(b)** Inventory **(c)** Sandbox

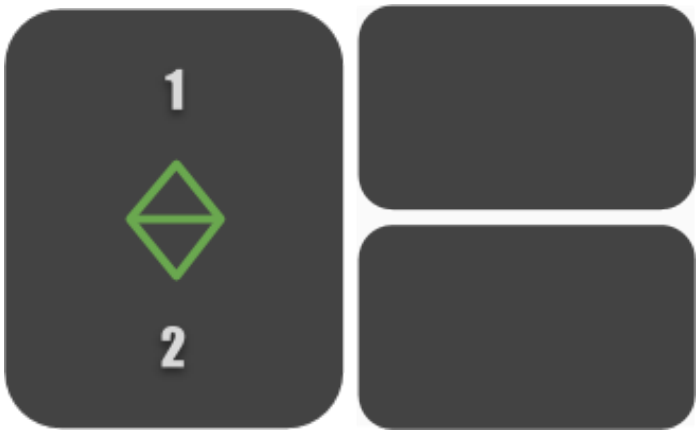
Figure 3.9: The different views available once a level has been started.

3.4 Rules

Rules are what is used to create propositions and from the starting propositions reach the goal of the level. These are designed to function exactly like their natural

deduction counterparts. Rules are shown on the right hand side of the gameboard, see figure 3.9a. All rules except the law of excluded middle are applied to propositions on the gameboard.

Figure 3.10 shows the visual representation of the rules conjunction introduction and elimination respectively. Conjunction introduction can be applied by selecting any two propositions on the gameboard. As the symbol suggests, the order in which the propositions are selected determines the arrangement of the new proposition. Conjunction elimination can be applied by selecting a proposition with a conjunction operator as the root operator of the nested structure.



(a) Conjunction introduction. **(b)** Conjunction elimination.

Figure 3.10: Visual representation of conjunction rules.

Figure 3.11 shows the visual representations of the rules disjunction introduction and elimination. Disjunction introduction can be applied by selecting any proposition on the gameboard. Disjunction introduction has two representations of the rule that can be applied and as the symbols suggest the arrangement of the new propositions differs between the two representations. Applying the rule requires the player to create a proposition in the sandbox, once done this proposition in disjunction with the previously selected proposition is created and brought to the gameboard. Disjunction elimination can be applied by selecting three propositions where one is a disjunction and the other two are implications that match the disjunction in the way described in the theory section, see chapter 2.

Figure 3.12 shows the visual representation of implication introduction and elimination respectively. Implication introduction can only be applied when an assumption has been made. The proposition that then will be created will consist of the assumed proposition and the selected proposition, as indicated by the visual representation. Implication elimination can be applied when an implication and its matching proposition according to the theory in chapter 2 is selected.

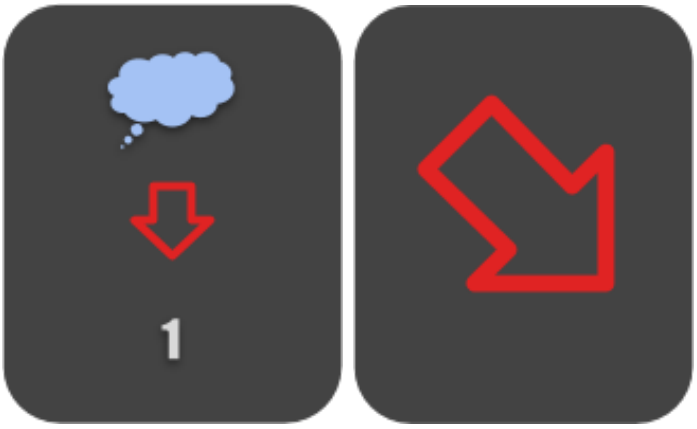
Figure 3.13 shows the visual representation for the rule absurdity elimination. Absurdity elimination can be applied by selecting an absurdity on the gameboard. Applying the rule requires the player to create a proposition in the sandbox, once

(a) Disjunction introduction.

(b) Disjunction introduction but the other way around, since order matters.

(c) Disjunction elimination.

Figure 3.11: Visual representation of disjunction rules.



(a) Implication introduction.

(b) Implication elimination.

Figure 3.12: Visual representation of implication rules.

done, the proposition is created and brought on to the gameboard.

Figure 3.14 shows the visual representation for the rule law of excluded middle. This rule can always be applied since it requires no selection. Applying the rule requires the player to create a proposition in the sandbox, once done, the proposition in disjunction with its negation is created and brought on to the gameboard.

3.5 Example of Level Gameplay

This section will describe an example of how a level can be played and completed.

Figure 3.13: Visual representation of absurdity elimination.

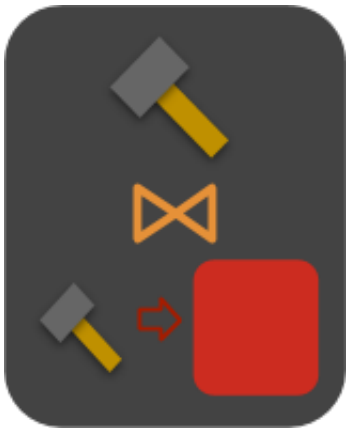
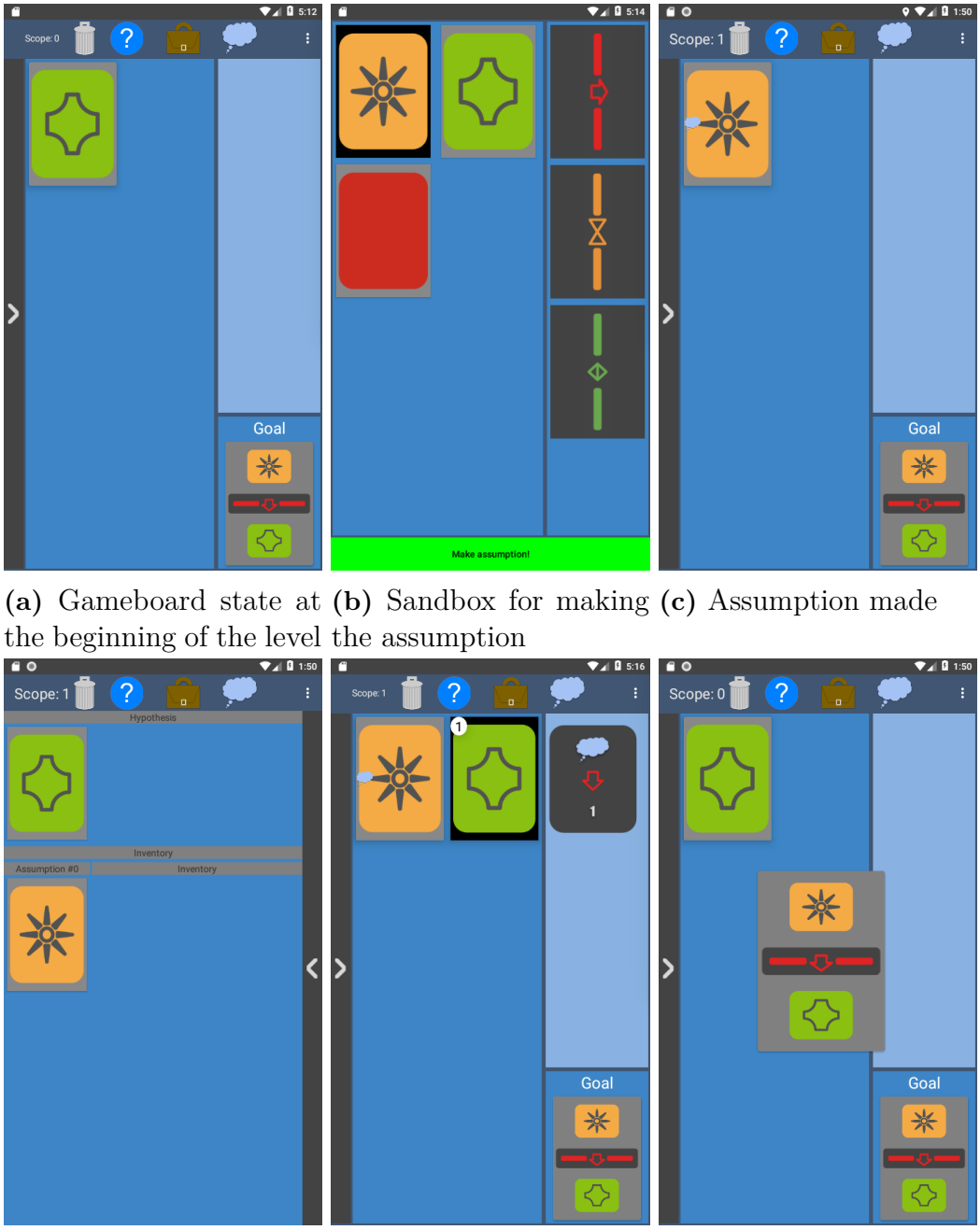


Figure 3.14: Visual representation of law of excluded middle.

This example showcases the level *Implication introduction* which is the tutorial level for learning how to create implications in the game. In figure 3.15a one can see the starting state of the gameboard. From there the player is supposed to make an assumption by pressing the thought bubble in the toolbar at the top of the screen. This brings the player to the sandbox screen shown in figure 3.15b. From here the player is supposed to select the orange symbol and make the assumption, bringing the player to the gameboard state shown in figure 3.15c.

Now the player needs to open the inventory bringing them to the screen shown in figure 3.15d. From here the green atomic proposition needs to be tapped to be brought to the gameboard, this new state is shown in figure 3.15e. From here the green atomic proposition should be selected and the rule of implication introduction should be applied bringing the player to the screen shown in figure 3.15f. This concludes the example of the tutorial level *Implication introduction*.



(a) Gameboard state at the beginning of the level

(b) Sandbox for making the assumption

(c) Assumption made

(d) Inventory

(e) Make implication introduction

(f) Level complete!

Figure 3.15: Level walkthrough: Implication introduction.

4

Methods

This chapter describes the methods used to develop and evaluate Gentzen's Quest.

4.1 Implementing the Logic Engine

This section describes how the logic described in section 2.1 was implemented in the game.

4.1.1 Expressions

An expression object is a mutually recursive data structure to represent any given logical proposition. To represent expressions we use a super class to represent any type of expression, and various sub classes to represent the various type of root expressions. An expression can for example either be an atomic proposition, absurdity, or a binary operator, which also is implemented by the the three binary connectives seen in classical logic namely conjunction, disjunction, and implication. The various operators then stores their operands as member variables. Negation is omitted and is implemented as a proposition implicating absurdity, like it is described in section 2.1.

Expressions can be created arbitrarily by combining existing expressions with operators, or by using the atomic propositions or absurdity. Another way of creating new expressions is by applying rules on a set of given expressions. These are explained in the following section.

4.1.2 Rules

A rule object in the logic engine represents an inference rule of natural deduction in classical logic. This object contains a list of expressions being operated on, as well as the inference rule to be applied.

These rules are automatically generated from a given set of expressions, as each type of inference rule can only be applied on a certain set of expressions previously proved in a natural deduction proof.

4.1.3 State Transitions

A given game session keeps track of how many scopes have been opened, and which expressions have so far been derived in the current as well as lower scopes.

The engine can also be asked for a set of applicable rules, given a list of expressions to be operated on, these expressions must be available in the current scope for the action to be valid. Some rules may need additional information, such as supplying an arbitrary expression for disjunction introduction. A completed rule object can be created using the original proposed rule object and such an expression.

One of these rule objects can then be passed back to the engine, resulting in a game state transition, as whatever new expression generated by applying the rule is added to the available expressions in the given scope.

Another way of transition the game into a new state is by making an assumption, and effectively opening a new scope, a scope is closed by selecting an expression and applying the implication introduction rule on it.

4.2 Ensuring Correctness

Ensuring that Gentzen's Quest successfully emulates formal logic is done by analysing the game mechanics in respect to the chosen logic calculus, as well as testing the critical parts of the implementation.

4.2.1 Evaluating Logical Equivalency

A given gameplay session should be logically equivalent to a given proof in natural deduction. See section 1.1.3 for details on this definition and section 2: Theory for details on natural deduction.

Since the start state of the game is simply a set of hypotheses, and every state transition in the game emulates applying inference rules, and opening and closing scopes in a natural deduction proof, it can be argued that the entirety of a game session which leads to procuring the goal expression also emulates an entire proof in natural deduction using either constructive or classical axioms depending on the game level in question.

4.2.2 Junit Tests

The library JUnit is used to test critical parts of the engine by doing a few assertions of selected parts of the code to indicate correctness of behaviour. This is important for determining if the implementation follows the specification of logical equivalence.

An important building block in our game engine is getLegalRules which will given a set of expressions return allowed rules to apply. Listing 4.1 shows part of the code that test the getLegalrules implementation, specifically for conjunction introduction. All rules have been tested with the same premises and principle and the implementation can be found in the Github repository [16]. Expressions are randomly generated in size and form, i.e, the types and numbers of operands and operators in the expressions is randomly generated. These expressions are added to a set and getLegalRules are applied to this set, which in turn returns a set of rules. An assertion is then made to make sure the rule we are testing exist in this set of rules. After that all possible rules containing the generated expressions are generated and from them all rules which are always allowed to be applied are removed. If the rule we are testing does not exist in the set of rules which are always allowed to be applied it is also removed. Any rules that would be applicable because of the randomness of the expression generation are also removed. Now this new set of rules only containing rules which should not be allowed is compared with the set returned from getLegalRules to ensure there is no intersection between these two sets.

Listing 4.1: Part of getLegalRulesTest

```
for(int reps=0;reps<100000;reps++) {
    Expression expression1 = generateExpression(0.2);
    Expression expression2 = generateExpression(0.2);
    ArrayList<Expression> expressions = new ArrayList<>();
    expressions.add(expression1);
    expressions.add(expression2);
    Rule correctRule = new Rule(RuleType.CONJUNCTION_INTRODUCTION,
        expression1, expression2);
    List<Rule> rules = getLegalRules(null, expressions);
    assertTrue(rules.contains(correctRule));
    List<Rule> incorrectRules =
        generateAllRules(expressions,false);
    incorrectRules.removeAll(generateAlwaysPossibleRules(expressions,null));
    // If the two generated expressions are an implication
        expression and the "correct" expression to apply
        implication elimination that rule is removed
    if(expression1 instanceof Implication &&
            ((Implication)
                expression1).getOperand1().equals(expression2) ){
        Rule correctRule3 = new
            Rule(RuleType.IMPLICATION_ELIMINATION,expressions);
        incorrectRules.remove(correctRule3);
    }
    if(expression2 instanceof Implication &&
            ((Implication)
                expression2).getOperand1().equals(expression1) ){
        ArrayList<Expression> reversedExpressions = new
            ArrayList<>(expressions);
        Collections.reverse(reversedExpressions);
        Rule correctRule3 = new
```

```
                Rule(RuleType.IMPLICATION_ELIMINATION,reversedExpressions);
            incorrectRules.remove(correctRule3);
        }
        assertTrue(Collections.disjoint(rules,incorrectRules));
    }
```

4.3 Evaluating the Game by User-Testing

The formulation of the user-testing mainly consists of asking questions that answer the evaluation of Gentzen's Quest (see section 1.1). Data on how much time spent on each level is also gathered as to get a better understanding of which levels include concepts or rules which are hard for the user to understand. As experience with formal logic was deemed to be relevant, data is also gathered and presented about that.

4.3.1 Selection of Test Subjects

The game was tested on readily available people to the project group, this included but were not limited to friends, family and classmates of the project group. The test subjects were all in the age group of 20-30 years old and they were all university students. None of the test subjects were foreign to smart phones or touch based mobile applications.

In the book *UX Design for Mobile* [17], it is stated that three to five testers are enough to identify any potential fallacies in the application's user interface. Eight users participated in the test which was deemed sufficient to provide reasonable amount of data to draw conclusions from even though more testers would provide extra reliability.

4.3.2 Gameplay Testing

The parts of the game that is subject to testing is the tutorial levels and a few intermediate and hard levels. The way the progression system works in the game forces the testers to complete a majority of levels in each category before progressing to the next one. This way it could be ensured that the testers would primarily test the tutorial levels, that were of most interest since they introduce all the core mechanics of the game. The game also recorded how much time each tester spent on a level, which could be subject to analysis. A guideline for total time spent testing was set to 20 minutes.

4.3.3 Test Procedures

The tests were conducted and administrated by members of the project group. Each project member conducted tests individually on test subjects of their own choice. All tests conducted followed the same guidelines described below in order to get comparable results.

At the start of the tests, the test administrator (a project member conducting the test) briefly explained how the test would be conducted. The administrator instructed the test subject that it would be imperative that they read the level instructions carefully since instructive animations were not implemented in the application.

Communication between the tester and test administrator was allowed if the tester needed help with clarification or if translation of the instructions in the game was needed. No help or guidance when it came to actual gameplay was allowed.

Communication was also allowed if any bugs occurred during the testing, the administrator would then let the tester know that a bug occurred and instruct the tester to restart the level.

After the gameplay was finished the testers answered the questionnaire.

4.3.4 Questionnaire

The questionnaire was short and open-ended. First, the testers were asked about their prior knowledge in formal logic. Then they were asked how easy and how fun the tester thought the game. The testers were asked to answer both questions on a one to five scale. Both of these questions were followed up with an open discussion about their arguments to why they gave that answer. The questionnaire finished with a last open question where the tester could give any input they felt fitting.

Questions in the Questionnaire

1. Any prior knowledge in formal logic? Please specify as comment
2. Could you easily understand the game? Rank 1-5 (very hard .. very easily)
    - What was hard to understand? Please specify as comment
3. Did you enjoy the game? Rank 1-5 (not at all .. very much)
    - What part was boring/fun? Please specify as comment
4. Any other comments?

4.3.5 Analysis of Test Results

When the data has been gathered from the user testing, both the quantitative and qualitative data is compiled and analysed. From the quantitative data, i.e. the scores given by the testers and the time for completion of levels, averages for each level are found among the testers with previous knowledge of formal logic and those without. These averages are then plotted as to indicated any occurring patterns.

From the qualitative data, i.e. the comments made and the answers to the question, core ideas are identified by seeing recurring statements. The number of testers that commented similarly to each core idea is then counted as to measure the prevalence of the opinion that the core idea represents.

4.4 Design Decisions

The goal of the game design is to present each mechanic in a clear and understandable way, in order to enable the player to easily interact with the gameplay. This is important for the evaluation of the concept of formal logic as a smartphone game as it will influence how the user perceives these game mechanics, both in terms of how easy it is to understand and how enjoyable it is to play. The main idea behind the design is to make visual elements easily distinguishable, consistent and clear, which is based on the principles of game design outlined in section 2.2.

4.4.1 Main Interface

As described in chapter 3, the main interface is the screen that appears for each level. This interface features a visual division through use of different colours creating a clear distinction between different elements. Furthermore, the main interface shows a few icons that symbolise different actions the user can take. These are the trashcan, used to discard expressions on the board, the question mark, to view the level description text, the briefcase, to open the inventory and the thought-bubble, to create assumptions. These symbols were designed to be self explanatory in what they represent. However, the mechanics of their actions are further explained in text.

4.4.2 Proposition Design

The visual representation of a logical proposition, which within the game is called an expression, is designed using consistent symbols. Each operator is associated with a symbol and a colour, and this representation is the same throughout the interface as to provide consistency, which is important according to Dalmau's conclusions discussed in 2.2.2. The symbols for conjunction and disjunction were chosen quite

arbitrarily, while the implication symbol more closely mimics the traditional representation of an arrow. Similarly, each atomic proposition used features a distinct colour and symbol, while absurdity is simply represented by the colour red and no symbol which distinguishes it.

When combining operators and atomic propositions in order to create larger more complex propositions, scale and rotation is used to show the level of nested expressions. By rotating expressions and making their size smaller, screen space is used more efficiently, which is important when the application is meant to be played on the small screen of a mobile device, as discussed in section 2.2. This rotation is done in such a way that the first expression is to the top or left and the last is to the bottom or right which is consistent with the way text is read making the representation feel natural.

4.4.3 Interacting with Propositions

Each proposition on the main gameboard can be selected and deselected through a simple touch on the screen. When this happens, feedback is provided to the user through the change of colour to mark which propositions are selected. A number also appears on the selected proposition in order to show the user in which order the propositions were selected which is important in the application of some rules.

This feedback tells the user that something has happened as well as provides information on the current state of the game, making it clear at all times what is currently happening. Providing feedback to player action is one of the principles outlined by both Dalmau and Baron discussed in sections 2.2.2 and 2.2.3

4.4.4 Rule Design

When a proposition is selected a set of rules is displayed to the user. Instead of arbitrary symbols for the rules that the user must memorise, the visual representation of each rule attempts to show what will happen when the rule is applied. This is done through the use of the operator symbols which are consistent with the representation of propositions, as well as with numbers indicating the selected propositions. Furthermore, a hammer-like symbol is used to represent when the user has to create or *build* an arbitrary proposition in order to apply the rule.

4.4.5 Level Design

Each level has been designed in respect to a progression in which new concepts are introduced and the difficulty is increased. This progression is defined by placing the levels in different categories and making only some of the levels available to the player at the start. By encouraging the user to play the levels in a specific order, the game mechanics can be explained without overwhelming the user. To further

prevent the risk of presenting too much information at once, certain mechanics such as some rules and the ability to make an assumption are locked until the later levels.

The idea to divide the game into levels is in accordance with the four methods that promote the state of flow, discussed in section 2.2.3. Through the division into levels, the player faces clear and concrete goals. By ordering the levels, the actions demanded from the player should be within the person's capabilities as one concept is introduced at a time. Furthermore, by dividing the game into distinct parts the game can be played in short sessions which is preferable on a mobile device, as discussed in 2.2.1.

5

Results of Game Testing

This chapter describes the results of testing done on the game. This includes user testing to get a grasp of what people thinks of the game, as well as Junit tests to better assure that the underlying logic engine is correct.

5.1 Results of User Testing

This section describes the responses received from the user tests as well as the time it took for them to complete the levels and their responses to the questionnaire. The questionnaire can be found in section 4.3.4 and the testing procedure is described in section 4.3.

5.1.1 Results of Level Time and Completion

Figure 5.1 shows the average time each level took to be completed by the testers. The data is divided between those who considered themselves to be experienced in propositional logic and those who didn't. If the tester could not finish the level, their result on that level is disregarded in figure 5.1.

Figure 5.2 shows the percentage of the experienced and the non-experienced testers that completed each level. Table 5.1 describes the level numbers and their corresponding name which is how they're presented in the application.

Table 5.1: Level number and corresponding level name

Level	Level name
1.1	Conjunction Introduction
1.2	Conjunction Elimination
1.3	Conjunction Introduction & Conjunction Elimination
1.4	Inventory Use
1.5	Conjunction Introduction & Conjunction Elimination 2
1.6	Conjunction Introduction & Conjunction Elimination 3
2.1	Implication Elimination
2.2	Implication Elimination 2
2.3	Implication Introduction
3.1	Disjunction Introduction
3.2	Disjunction Elimantion
3.3	Disjunction Elimination 2
4.1	Absurdidy
4.2	Absurdity Elimination
5.1	Deep Implication

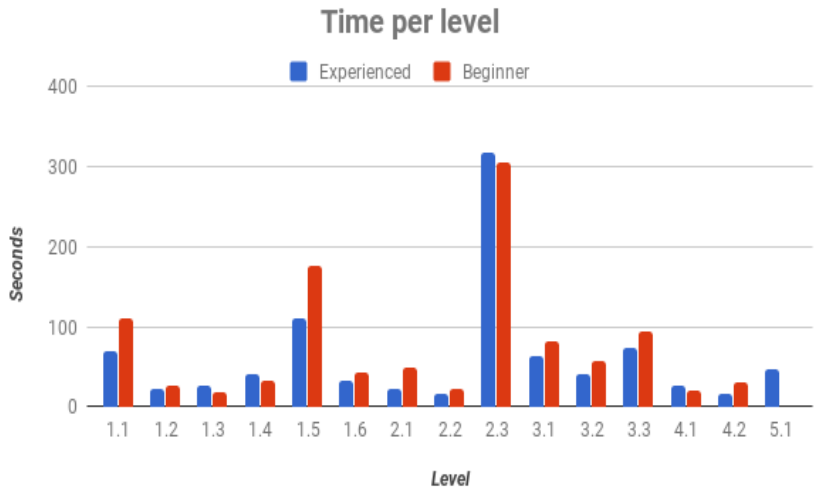


Figure 5.1: The amount of time on average it took for the testers to complete the levels, categorised by experience.

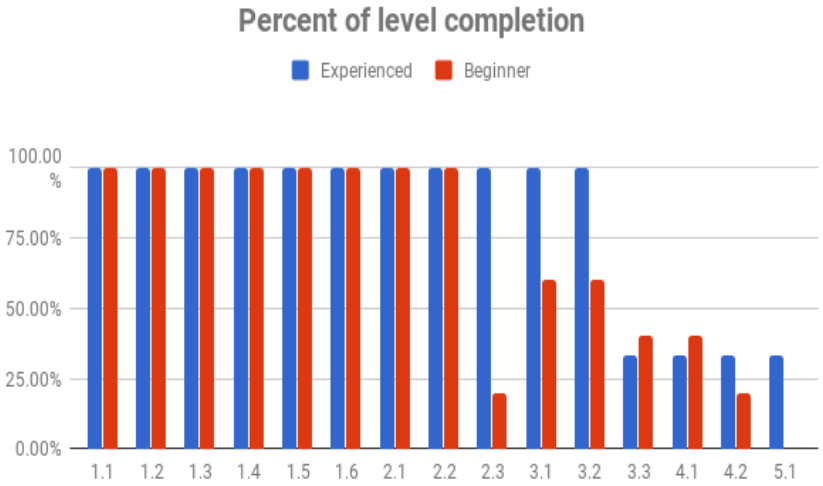


Figure 5.2: Percentage of the testers that finished each level

5.1.2 Results of the User Questionnaire

The testers were also asked to grade the game on a scale from one to five on how they enjoyed the game and how well they could understand it. The results are presented in figures 5.3 and 5.4, where again the testers' answers are divided by previous experience.

Furthermore, the testers were asked to comment on their thoughts and experience while playing the game. All these comments are shown in appendix B. From these comments, the following core ideas were identified. The number in parenthesis displays the number of testers who made a similar comment.

- Negative remarks on the explanation of the game mechanics. (8/8)
- Negative remarks on parts of the graphical representation. (3/8)
- Positive remarks on some parts of the graphical representation. (2/8)
- Negative remarks on the difficulty curve. (6/8)
- Negative remarks on level 2.3, implication introduction. (6/8)
- Negative remarks regarding the game concept. (1/8)
- Positive remarks regarding the the game concept. (5/8)
- Negative remarks regarding the enjoyment of the game as a result of poor descriptions of game mechanics. (4/8)

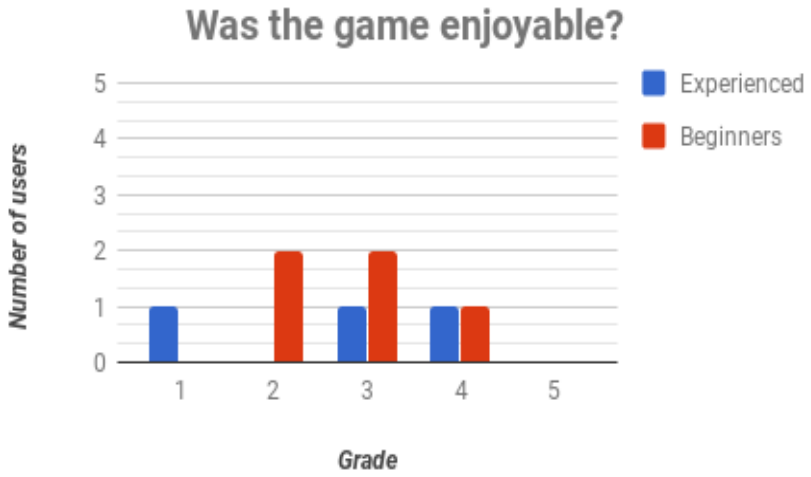


Figure 5.3: The testers' grade of enjoyment of the game

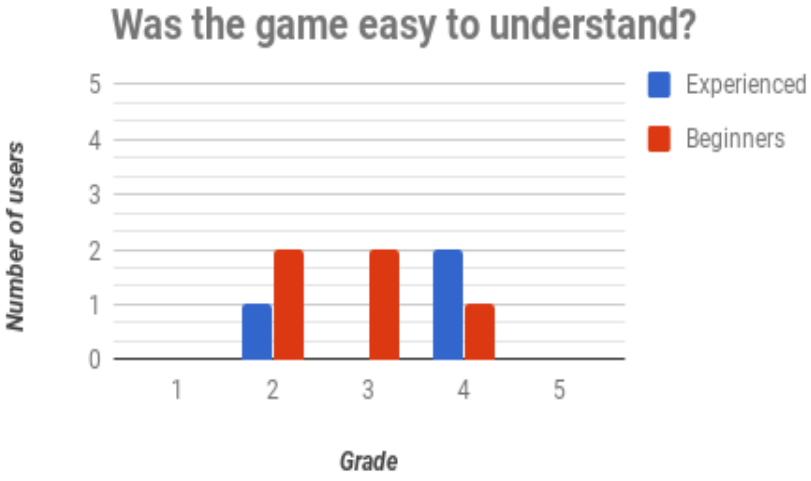


Figure 5.4: The testers' grade of comprehension of the game

5.2 Result of Junit Tests

This section presents the results from the Junit tests done on the code. Source code for all the Junit tests can be found in the github repository [16].

As can be seen in figure 5.5 all inference rules passed their respective tests.

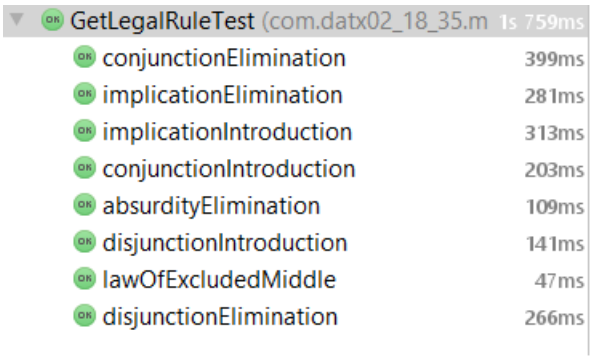


Figure 5.5: Results of Junit tests. Green "ok" symbol means that the test passed.

6

Discussion and Conclusion

As stated in section 1.1 the evaluation satisfaction criteria aims to evaluate how fun and easily understandable the concept behind Gentzen's Quest is and whether it's logically equivalent to formal proofs. This chapter discusses the results and reflects on the concept as a whole.

6.1 Evaluating Gentzen's Quest

This section will discuss the results from user testing and the conclusions made from the data. Furthermore, the section will also discuss the proof equivalence of the game.

6.1.1 Conclusions From User Testing

Overall, the user testing pointed towards that the game has some room for improvement. The issues identified were due mostly to difficulties in introducing and explaining fairly complex ideas.

In figure 5.1 we see clear spikes in time consumption for certain levels, namely level 1.1, 1.5, and 2.3. The first level has a long text description and this in combination with being the first level could explain the longer time. Level 1.5 challenges the player in considering how propositions will be rotated when combined, which could explain the increased difficulty. Finally, level 2.3, *Implication introduction* introduces the concept of scopes and assumptions. The testers found these concepts hard to understand, as shown by the comments made, as well as in the data of which levels were completed (figure 5.2).

Testers commented in general on the varying difficulty of the levels, and how it hindered the enjoyment of the game. From the testing results it is clear that the level on implication introduction has not been explained well enough as many of the testers failed to complete it, especially among those who did not have a background in formal logic.

Despite varying difficulties in the levels and poor explanations of game mechanics, several of the testers were positive towards the game concept. More specifically,

testers commented that pondering on logic puzzles is a satisfying activity, and that this concept would work well as intellectual exercise. These specific comments can be found in the raw data presented in appendix B.

The scores given by the testers on the enjoyment of the game are quite varied. Taking in consideration the comments made, several testers believed that the enjoyment of the game was a result of poor descriptions and weak explanation of the game's mechanics.

6.1.2 Validity of Testing Results

It is well worth noting again that the testing was performed by people with relation to the authors, and so it brings up the question of bias concerning the questionnaire. The results could be skewed because of this. The users were also limited in terms of time, which could have affected both their results as well as their enjoyment negatively.

Many of the testers came from a similar background, namely university students. As a result the significance of the conclusions from the results is limited. A broader selection of testers would give a better understanding of the strengths and weaknesses of the game.

6.1.3 Logical Correctness

In previous sections, the game mechanics in Gentzen's Quest are described to emulate inference rules which means a game session corresponds to proofs in formal logic. However, this conclusion is based on the assumption that the implementation is entirely correct. This has not been rigorously proven and cannot be done without a complete analysis of the code.

Such an analysis could be performed by implementing the core model in a formal verification language like *Dafny*[1] but was not deemed a priority due to time constraints. Instead, the conclusion is supported by the fact that the Junit testing and testing of game functionality gives no indication of logical faultiness. As such the conclusion can be made that the game likely upholds proof equivalency.

6.2 Evaluating the Concept of Gamification of Formal Logic

As the discussion and description of Gentzen's Quest has shown, it is feasible to create a game based on formal logic.

[1] See: https://github.com/Microsoft/dafny

First of all, creating game mechanics that emulates proof calculus is possible and has been done to a successful degree in Gentzen's Quest. This concludes the fulfilment of satisfaction criterion 3: "The game play should be logically equivalent to a formal proof". The choice of logic system proved to work well as an exploratory puzzle game as the testers were positive towards this concept.

The difficulty in creating a formal logic game, and where Gentzen's Quest fails, is to effectively teach and explain the complex game mechanics. The results indicate that Gentzen's Quest was confusing and hard to play, the question is whether this is because of a lacklustre tutorial and level design or because of formal logic being a complex topic. This question could be answered by testing other implementations attempting to better explain the game mechanics and determining if the concept of formal logic as a game is still seen as too complex.

From the results we can neither conclude the fulfilment of the satisfaction criterion 2: "The game should be easily understandable", nor can we rule out the opposite, as the concept might not be the culprit.

As discussed in section 6.1.1 the enjoyment gained from playing the game was attributed to the game concept, the majority of negative remarks regarding the enjoyment of playing was directed at poor descriptions. This indicates that the concept might fulfil satisfaction criterion 1: "The game should be enjoyable".

6.3 Further Development of Gentzen's Quest

This section will discuss possible features that could be implemented in the game Gentzen's Quest in the future, but were not done during the project due to time restrictions.

6.3.1 Tutorial Design

As the results concluded, a major complaint was that the explaining descriptions for the levels were lacking. Considering the strategy for tutorials outlined in section 2.2.4, it can be argued that the tutorial levels cater more to an "explorative acquisition player" as the early levels can be completed by exploring the game mechanics. Because of this, improvement can be made by further developing the descriptions as to guide the "modelling acquisition players", or implement a more visual demonstration of the mechanics, for example through animations which the Android official documentation recommends [18].

6.3.2 Level Design

User tests concluded there was a problematic varying difficulty of the levels and that it hindered the enjoyment of the game. Section 2.2.3 describes a few principles to be

followed for designing levels successfully. As the user testing showed, the levels in Gentzen's Quest failed to do so, as many levels were found frustrating due to their difficulty. The main flaw of Gentzen's Quest's level design could be that the game demands actions beyond the player's capabilities.

Further developing the levels as to provide a good balance between introducing new concepts and building on already established mechanics can be a way to improve the game experience. Care must also be taken to ensure that aspects of the underlying logic, that are often considered complicated, is explained in a clear and understandable way.

6.4 Ethical Implications

The conclusion drawn is that the application has little to no implication on ethical aspects as well as no considerable implication on society. However it can be argued that the game may provide value to society by being able to be used as an educational resource, and on an individual level, improve cognitive ability, albeit in a limited capacity.

6.5 Conclusion

In conclusion Gentzen's Quest is an implementation that shows the potential of the concept of gamification of formal logic. The game manages to translate logical calculus into game mechanics and likely maintains proof equivalence. The biggest flaw with Gentzen's Quest is the poor explanation, as this is the motivation behind most negative remarks regarding the enjoyment and understandability of the game.

The concept is therefore considered to have potential of being an enjoyable experience with intuitive gameplay if the game mechanics can be explained in a successful way. Perhaps also, other ways of implementing the game mechanics can be considered in order to improve the ease of understanding the game.

Bibliography

[1] Merriam-Webster. Dictionary definition of logic, 2018-01-19. URL https://www.merriam-webster.com/dictionary/logic.

[2] R.Renjith Kumar. Effectiveness of formal logic course on the reasoning skills of students in nizwa college of technology, oman. *Journal of Education and Practice*, 8(7), Apr 2017. ISSN 2222-1735. URL http://iiste.org/Journals/index.php/JEP/article/view/36006.

[3] Dan Bouhnik and Yahel Giat. Teaching high school students applied logical reasoning. *Journal of Information Technology Education: Innovations in Practice*, 8, 2009. ISSN 2165-3151. URL https://www.learntechlib.org/p/111713/.

[4] Jonathan Seller Barbara Boucher Owens. A study of the relationship between performance in propositional logic and computer science. 1996. URL http://ccscjournal.willmitchell.info/Vol11-95/No7/Barbara%20Boucher%20Owens.pdf.

[5] Merriam-Webster. Dictionary definition of gamification, 2018-04-27. URL https://www.merriam-webster.com/dictionary/gamification.

[6] Fengfeng Ke. A qualitative meta-analysis of computer games as learning tools. 2009-01. URL https://www.researchgate.net/publication/237267086_Chapter_I_A_Qualitative_Meta-Analysis_of_Computer_Games_as_Learning_Tools.

[7] Tzu-Chien Liu a Yao-Ting Sung a, Kuo-En Chang b. The effects of integrating mobile devices with teaching and learning on students' learning performance: A meta-analysis and research synthesis. *Computers & Education*, 2016-03. URL https://doi.org/10.1016/j.compedu.2015.11.008.

[8] Mark Ryan Michael Huth. *Logic in Computer Science*, chapter Propositional Logic. Cambridge University Press, 2004. ISBN 0511555660.

[9] Joan Moschovakis. Intuitionistic logic. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, spring 2015 edition, 2015.

[10] Mauricio Ayala-Rincón and Flávio L. C. de Moura. *Derivation and Proofs in the Propositional Logic*, pages 1–41. Springer International Publishing, Cham, 2017.

ISBN 978-3-319-51653-0. doi: 10.1007/978-3-319-51653-0_1. URL https://doi.org/10.1007/978-3-319-51653-0_1.

[11] Scolastici Claudio and David Nolte. *Mobile Game Design Essentials*, pages 324–329. Packt Publishing, 2013. ISBN 9781849692991.

[12] Daniel Sánchez-Crespo Dalmau. Learn faster to play better: How to shorten the learning cycle, 1999. URL https://www.gamasutra.com/view/feature/131799/learn_faster_to_play_better_how_.php.

[13] Stephanie Daul. *Game Design for Learning*, chapter Elements of Gaming. Association for Talent Development, 2014. ISBN 9781562869571.

[14] Baron Shaun. Cognitive flow: The psychology of great game design, 2012. URL http://www.gamasutra.com/view/feature/166972/cognitive_flow_the_psychology_of_.php.

[15] Sheri Graner Ray. Tutorials: Learning to play, 2010. URL https://www.gamasutra.com/view/feature/134531/tutorials_learning_to_play.php.

[16] MAGNUS HARRYSON JOHAN RONNÅS ROBIN ÅSTEDT JONATAN ÖBERG JOHANNES BACKLUND, LUDVIG EKMAN. Github repository for gentzen's quest, 2018. URL https://github.com/raxxor95/DATX02-18-35.

[17] Pablo. Perea and Pau. Giner. *UX Design for Mobile*, chapter User Testing. Packt Publishing, 2017. ISBN 9781787283596. URL https://books.google.se/books?id=LedDDwAAQBAJ.

[18] Official android documentation. Overview of android animations, 2018. URL https://developer.android.com/training/animation/overview.

A

Pre-study: A Search for Games with Similar Concept

A.1 Introduction

Before beginning to work on the project we wanted to confirm the concept of a game abstracting formal logic was original and that no such application already existed. This was achieved in this pre-study by exploring applications about logic.

A.1.1 Aim

The aim of this pre-study is to search for an application with the concept of visually abstracting proof calculus, in order to better know if the project is new and original.

If any games meeting these requirements are found, the aim is also to analyze these games and get a better understanding and change our scope accordingly.

A.1.2 Limitations

Since the main project is limited to Android applications, this will also be the limitation for this pre-study. This choice is discussed in section A.5.

A.2 Method

At first the embedded search engine in Google Store was used, but the results seemed to be affected by the popularity of the applications, and the same names came up in the searches. Therefore, it was decided to redo the search with the search engine Duckduckgo. In order to only get search hits from Google Store the advanced search function ”site:*play.google.com/store/apps*” was used.

The search terms used together with the ”site”-function are listed in A.1. Each time a result was found, further information was looked for in the description of

the application. If a game was especially interesting, it was downloaded and tried out.

All applications that involved logic was included in the result.

Table A.1: Search words used when looking for a game with the concept of mimicking formal proofs. The asterisk (*) is a joker sign, meaning that the word can end in different ways.

Logic*
Proof*
(Natural) Deduction
Proposition*
Conjunction
Disjunction
Induction
Elimination

A.3 Result

14 applications related to logic was found, but no application with the same idea of abstracting logic into a puzzle game.

The applications found can be included in one of these five categories:

- Simple multiple-choice questions about logic
- Building and proving formal propositions
- Calculators for formal logic
- Solving proofs with sequent calculus
- Logical combinations of transistors

The categories above are described in the next part of the result. In the end of each category there is a conclusion to whether it is meeting the requirements in the Aim

A.3.1 Simple Multiple-Choice Questions About Logic

These are multiple-question games, and do not have an logic engine on their own. The names of the applications are listed in A.2.

These multiple-question application does use logic, but there is no visual abstraction. There is also no proof calculus since the user only can give an answer and not make a proof.

Table A.2: The names of the applications found that uses multiple-choice questions about logic. They can all be found at the Google Store.

Learn Reasoning : Logical Reasoning & Cool Maths
Logical Reasoning Test: Practice, Tips & Tricks
Andor
Logic and deduction

A.3.2 Calculators for Formal Logic

These applications, that are listed in Table A.3, can create and evaluate a preposition in different ways.

Table A.3: The names of the applications where the user can create and try out propositions. They can all be found at the Google Store.

Logic Calculator
Truth tables
Math Logic Calculator
Logical Sentences

This category does use formal logic, but there is no abstraction, and no concept of solving proofs.

A.3.3 Building and Proving Formal Propositions

The concept of the applications in this category is to prove formal logic, without any abstraction. The names of these applications are in Table A.4.

Table A.4: The names of the found applications that can be used to prove a proof in formal logic. They can all be found at the Google Store.

Logic-Proof Studio
Logic++
LogicCalc

The user gets a proof including several propositions, and can use any rule of logic on the propositions in order to achieve the goal.

This category uses proof calculus, but does not do an abstraction of logic.

A.3.4 Solving Proofs with Sequent Calculus

There was only one application found that used sequent calculus, and the name of this application is Natural Deduction.

Like in section A.3.3 this category uses proof calculus, but does not abstract it visually.

A.3.5 Logical Combinations of Transistors

These applications, that are listed in Table A.5, makes the user combine transistor gates in various ways to reach the goal.

Table A.5: The names of the applications found that uses logic gates and transistors. They can all be found at the Google Store.

Smart Logic Simulator
Boole
Switch or not?

This category of applications has a visual abstraction of logic, but not proof calculus. In these applications

A.4 Conclusion

There are few applications with similar concepts available for android mobile devices. This makes the choice of our scope for the project broad and a variety of available ways to explore this concept of abstracting formal logic proofs via a graphical representation.

A.5 Discussion about the Scope of the Pre-Study

After the pre-study was performed we came to the conclusion that more could have been learned if we included all platforms. If we found a game with our concept on another platform we could have analysed that game just as well as an Android based game.

B

Raw Response Data from User Testing

The notes from the open questions about Gentzen's Quest in the questionnaire are gathered in this chapter and comments mentioned by the user during the playing of the game is also included here. The data is mostly quotes, but sometimes paraphrased. A translated summary is available in 5.1.2.

B.1 Comments from User During Testing

En mening om hur symboler roterar var väldigt oklar men blev tydlig efter han hade spelat levelen. Inte tydligt att man kunde deslecta kort. Fattade först inte att implikationssymbolen var annorlunda från conjunction, men det blev tydligt när han testade applicera regeln. Blev förvirrad av assumptions o vilket scope han va på också hur implication introduction fungerade .

Trodde conjunction introduction skulle komma upp efter att man tryckt på en symbol bara. || Trodde inte det var någon skillnad på A&(A&(A&B)) och (A&A)&(A&B) i view.

För mycket text i intro!. "Oklart vad jag gjorde men jag gjorde rätt". Split (vad betyder det? 2 gråa rutor är otydligt). implication elimination, Varför är pilen sne? varför nedscrollad i början. Varför finns det i inventoryn när jag raderade det. Assumption är väldigt otydligt(väldigt frustrerad). Jag kommer inte ihåg vad det här betyde(operatorer). 3 kort regeln, vaför spelar inte ordningen roll nu?

Öppnade inventoryn för att knappen fanns där, Beskrivs in att man kan avselecta kort, I inventoryn så förklaras inte hypotes etc, Svårt att förstår rotation av kort då beskrivningen av det är komplicerad. Conjunction elimination är svårt. Implikation introduktion är komplicerad.

Inventoryn med scopes förklaras inte, Implikation introduktion svårt.

B.2 Did you understand the game?

Svårt i början, men efter ett tag blev det något bättre "Underlig kombination av att ibland bara ge svaret och sen nästa puzzel ska det listas ut allt. En obalanserad stegring av svårighetsgrad riskerar snabbt att tappa spelaren. Att bara ge svaret är helt opedagogiskt. Fina animationer. " Deleteknappen förklades inte, assumption krångligt, mer precisa förklaringar vad alla knappar gjorde, förklara hur man manuevrar i appen förklaras inte, backa ur saker etc.

Lätt fram till implication introduction Alldeles för långa engelska instruktioner med typ fackspråk, Svårt att hålla koll på alla olika ord som introduceras, för snabb progression.

Det gick bra fram till implication introduction. Inventoryn var lite förvirrande, förstod först inte att kort kom in på gameboarden. Vridningen på korten var högst oklart.

B.3 Did you enjoy the game?

Spännande med att spelet är kopplat till kunskap, men med nuvarande brister är det ganska begränsat hur kul det är. Om man vet att det går att lösa kan jag tänka mig att det är roligt, som hjärngympa. Tycker generellt att formel logik bevisföring kan vara roligt som hjärngympa

Typ en 4 i början, men det blev typ en 2/1 i slutet pga av frustration över att man inte förstod vad som skulle göras. Konceptet var roligt! behöver bara bättre introducering av hur man spelar Det var roligt fram tills nivå 9 då det var en grov svårighetökning.

B.4 Other Comments from after the User Completed the Game

Spännande. Info rutor på regel korten önskas och inventoryn.. key saker liksom. Fler mindre info rutor än en stor. Helt walled in environment, varje nytt klick ger liksom ny info (tutorial). Kul ide.

Göra saker mer tydligt vad det innebär. Front-end och design behöver uppdateras och är avgörande om man spelar det eller inte. Konceptet fungerar relativt bra, hur långt kan man gå? Behöva hämta saker från inventoryn är tradigt och inte så intuitivt. Ta bort saker från inventoryn också.

Det skulle behövas en bättre förklaring på sandboxen Man vet inte när det spelar någon roll vilken ordning man klickar på korten. Molnet på assumptionkort kunde ha varit tydligare. Beskrivningarna hade kunnat varit två delade, först på leveln

förklara hur man spelar sen efter man klarat leveln förklara hur logiken fungerade formellt. Animationer hade varit bättre än nuvarande instruktioner.

Spelets grafik var trevligt. Fattade inte i slutändan var det var hon gjorde så att hon fick rätt.