Undergraduate Thesis

Genetic Algorithms to Generate Heuristics about Human Syllogistic Reasoning

Maya Schöchlin

Examiner: Prof. Dr. Marco Ragni
Advisers: Daniel Brand, Nicolas Riesterer

Albert-Ludwigs-University Freiburg
Faculty of Engineering
Department of Computer Science
Chair for Cognitive Computation

October 04th, 2018

Writing period

04. 07. 2018 – 04. 10. 2018

Examiner

Prof. Dr. Marco Ragni

Advisers

Daniel Brand, Nicolas Riesterer

Declaration

I hereby declare, that I am the sole author and composer of my thesis and that no other sources or learning aids, other than those listed, have been used. Furthermore, I declare that I have acknowledged the work of others by providing detailed references of said work.
I hereby also declare, that my Thesis has not been prepared for another examination or assignment, either wholly or excerpts thereof.

Place, Date	Signature

Abstract

Goal of this undergraduate thesis was the generation of easy and human-understandable boolean logic formulas to predict human syllogistic reasoning. To achieve this, syllogistic problems and answers were encoded in binary, and used in a genetic algorithm to generate circuits to match them. I showed that genetic algorithms can generate theories that predict human answers with an accuracy that comes close to the most frequent answer strategy. I also compared the generated formulas to state-of-the-art theories of the syllogism.

Zusammenfassung

Das Ziel dieser Bachelorarbeit war die Generierung einfacher und verständlicher boole'scher Logik-Formeln, mit denen man menschliches syllogistisches Schlussfolgern nachvollziehen kann. Für dieses Ziel habe ich syllogistische Probleme und Antworten binär kodiert und einem genetischen Algorithmus übergeben, der dazu passende boole'sche Schaltkreise generiert hat. Ich habe gezeigt dass der Algorithmus Theorien finden kann die menschliche Antworten mit einer Genauigkeit die der häufigsten Antwort nahe kommt voraussagen kann. Ich habe außerdem die generierten Formeln mit bisherigen syllogistischen Theorien verglichen.

Contents

List of Figures

List of Tables

1 Introduction

Human reasoning has long been subject to research in psychology and neuroscience. It has led to many new questions, seeing as the percentage of logically correct conclusions can vary widely from subject to subject, depending on factors like educational background and measured level of intelligence. It is therefore integral to understand how these conclusions are generated in the human mind if the matter is to understand human cognition.

In cognitive science, a common domain for testing and analyzing deductive reasoning are so-called syllogisms (Greek for '*logical conclusion*'), encodings of two logical premises and their conclusion, using universal and existential quantifiers, and negation. In total, there are 64 syllogisms, to each of which nine possible answers can be given. Of course, not every answer will be correct for every syllogism. The various answers humans are more or less likely to give for different syllogisms can give interesting insights into deductive reasoning. As Khemlani et al. succinctly summarize [1], several theories for understanding and categorizing how humans reason over syllogsims have been proposed in the field of psychology in the last hundred years.

However, these theories focus more on deconstructing the processes that are executed in the human brain during reasoning, whilst not concerning themselves much with prediction. Since this is a rather bottoms-up approach, a top-down approach could lead to new insights into human reasoning processes: Firstly find heuristics

that make good predictions about human answers for syllogisms, and then analyze these heuristics for commonalities. The ability to assess the quality of such generated theories first-hand by letting them predict a certain number of human answers is another advantage of this approach. A straightforward idea for making a predictive approach work would be to train a deep neural network to make the predictions. But as they often have a complex weight-space, are high-dimensional, and not symbolic, there is no easy way to generate simple, human-readable heuristics with them.

Genetic algorithms [2] are a better approach to finding these heuristics, as they can find a variety of good solutions by making use of evolutionary processes, which combine successful solution candidates and exclude unsuccessful ones through selection. They are comparatively easy to implement, and require less input data and tuning than other machine learning approaches, which is advantageous for the problem at hand as it does not give a lot of input data to begin with. Genetic algorithms are capable of generating logical formulas for a given problem, for example by developing a boolean circuit [3] to fit the problem's parameters. Because the circuits can be reversely transferred into logical formulas again and are as such easy to interpret for humans, they make a good approach for finding simple predictive heuristics.

The goal of this thesis is to model human syllogistic reasoning using formal logic, and optimize the resulting formulas for simplicity. The results offer two contributions to the current state of the art: highly interpretable models and highly predictive performance. They also show that it is possible to generalize an approach that uses genetic algorithms with boolean circuits in similar problems.

This thesis consists of seven chapters. Following this chapter will be a brief overview over recent research in the concerned topic in chapter 2, as well as a background section in chapter 3 to introduce all neccessary concepts that were used. In chapter 4, the theoretical foundation of my approach is presented, and chapter 5 describes

the proceedings of experiments. Chapter 6 gives a short evaluation of the results by usage of an online benchmark. Chapter 7, finally, contains the conclusion.

2 Related Work

Genetic algorithms have been used before to solve complex search problems with a large problem space, and often shown to be able to compete with state-of-the-art search techniques [4, 5, 6, 7, 8]. The best results genetic algorithms have achieved were often through usage of a combined approach with other search methods, where the genetic algorithm was not tasked primarily with optimization but with diversifying the exploration space [4, 5, 6]. For the generation of boolean circuits especially, genetic algorithms have been used in the context of generating circuits to satisfy given truth tables. This has proven largely successful, however a scalability problem made processing large truth tables difficult due to a strong increase in processing time [9, 10, 11]. Soleimani et al. [11] especially propose separating the problem space of the circuit into parts to decrease the complexity of evolution.

Theories of the syllogism vary widely in both their method and accuracy, as summarized in the article by Khemlani et al. [1]. As said before, they are not designed to predict human answers in the first place, which is why assessment of their predictive power is difficult. I will go into more detail about popular psychological theories in chapter 3. Machine learning or optimization approaches to generate theories about syllogistic reasoning are few and far in between, but Ragni et al. propose one such an approach [12]. Their idea is to take cognitive theories and combine them with a data driven approach of optimizing prediction. To achieve this they take a set of human responses and then use an autoencoder to generate a precise heuristic for that

data. They also propose a portfolio of heuristics which they use in a adaption-based approach. With this approach they can assess theories in terms of accuracy, which is useful in the sense that it makes it easier to compare predictive approaches with other theories. By maintaining a portfolio of different heuristics of a theory and keeping record of each heuristic's relative past accuracy, they let each heuristic give a weighted vote for a final prediction [13]. With this approach they are able to compare and surpass tested psychological theories in terms of accuracy.

3 Background

3.1 Boolean Circuits

Boolean circuits are a theoretical or technical method to realize boolean logic formulas, as first proposed by George Boole in 1847 [3]. They take one or more inputs that can assume either the value of 1 or 0, or alternatively $True$ or $False$, and consist of logic gates such as AND, OR, XOR, and NOT. With these gates the circuits evaluate the inputs. The output or several outputs contain likewise a value of 1 or 0 again, which represents the result of the inputs after they were evaluated by all gates in the circuit.

Table 1: Different logic gates as building blocks of a logic circuit.

A	¬ A
0	1
1	0

(a) NOT-Gate

A	B	A ∧ B
0	0	0
0	1	0
1	0	0
1	1	1

(b) AND-Gate

A	B	A ∨ B
0	0	0
0	1	1
1	0	1
1	1	1

(c) OR-Gate

A	B	A ⊕ B
0	0	0
0	1	1
1	0	1
1	1	0

(d) XOR-Gate

There are several logic gates which will be used in this work to build boolean circuits. An AND-gate takes two inputs and returns 1 or $True$ only if both of the inputs were also 1, else 0. An OR-gate returns 1 if either of the inputs were 1. An

XOR-gate returns 1 if exclusively one of the inputs holds the value 1, the other one 0. A *NOT*-gate is a gate with only one input, and it returns the opposite of the input that it was given. The detailed function of several gates can be seen in Table 1. Putting the gates together to form a tree-like network, it is possible to build any logical formula that can be evaluated with boolean logic. Such formulas are, if not too long, easily understood by humans or can be made understandable, for example by visualizing them with a Venn diagram [14].

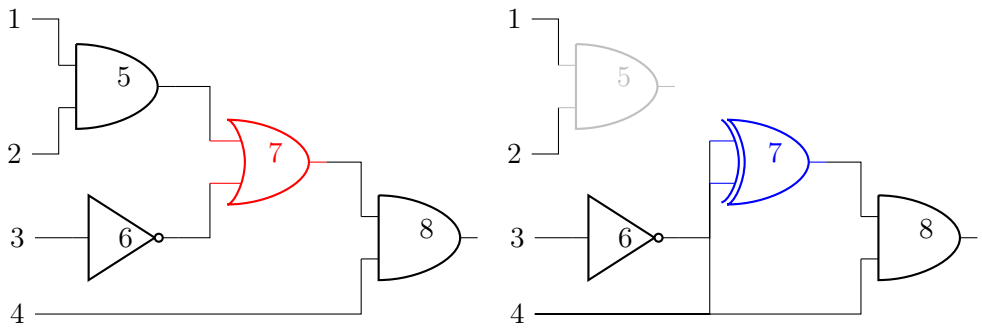
3.2 Hill Climbing

Hill climbing is an algorithm type that is used for optimization and local search. It begins with an optimum or minimum as the goal (depending on how the goal function is defined [15]), and a starting solution that may be arbitrarily far away from it. In a continuous iteration, the algorithm makes local changes to try and see if the new solution is closer to the desired goal than the previous one, until it either reaches said goal or alternatively ends up in a local extremum, which is a solution that cannot be improved on further by any direct, local changes, but is not the desired goal as specified in the function either [16].

3.3 Genetic Algorithms

Genetic algorithms are a variant of machine learning which are frequently used in optimization or search problems. They use a concept that is based on biological mechanisms in genetic inheritance as first proposed by Darwin [17], where processes that take place during the copying of genetic sequences result in the creation of new combinations. With evolutionary selection the genes that prove advantageous then

continue to spread over generations of a population. For genetic algorithms, this principle is used to continually improve a solution candidate towards a given goal or ruleset by drawing on strategies that combine randomization and combination of already good solutions [2]. As they go into a search space without expectations or previous knowledge that could bias their approach, this makes them a good candidate for finding new directions in problems that were previously only looked at by humans. As Clinton Sheppard's library for generating boolean circuits with genetic algorithms [18, 19] will be used primarily in this work to generate formulas, a brief explanation about how genetic algorithms function follows, using this concrete library as an example.



(a) Before Mutation **(b)** After Mutation

Figure 1: Process of a single mutation step for boolean circuits. In 1a, a random index is selected as the gate to be mutated. This is the gate marked in red. Per randomization, a new gate to replace the old one is then chosen. The inputs of this new gate are also chosen by randomization. The result can be seen in 1b, where the new gate is marked blue. The greyed out gate is one whose output is unused now, which is why it will not appear in the result circuit. It can however still be chosen as an input for a following mutation.

A genetic algorithm always starts with a set of rules, which in our case are the different input bits of the boolean circuit, and the corresponding output bits the circuit should produce for each of those inputs. Firstly the genetic algorithm generates

a random guess at a solution, which should fulfill the rule set. This guess is then evaluated by a fitness function which tells the algorithm how good the guess was. In case of the boolean circuit, this fitness is evaluated by building and testing the formula on all rule inputs, and then checking how many of the circuit's answers match the output that was given for that specific input in the rules. The more outputs are correct, the better the formula. A correct solution is given only if all inputs match the outputs for the formula.

After the solution is created and evaluated, the next step is to generate a second solution attempt that is hopefully an improvement of the first one. This is achieved by using heuristics based on biological mechanisms of evolution, which define the genetic algorithm as an evolutionary procedure. Mutation and crossover are some of the most frequently used heuristics. For mutation, a new guess is generated by taking the current one and changing parts of it randomly, based on the biological principle of a change in the nucleotide sequence of a chromosome, which typically occurs due to errors during replication of the sequence [20]. In case of the boolean circuit, the mutation is done by randomly replacing any gate in the circuit with a new, randomly selected gate as depicted in Figure 1. The genetic algorithm compares which of the two guesses has a better fitness, and remembers the one that is best. It continues generating new guesses and comparing them until a solution that matches all rules is found.

As observed in chapter 2, genetic algorithms often perform better when they are used in a hybrid approach. For this purpose, Sheppard's library uses a hill climbing algorithm in combination with the genetic one. This has the purpose of making the algorithm work towards an optimal length of the formula, which the genetic algorithm itself does not care about. When comparing the new mutation with the previous best one, the genetic algorithm looks only if more rules are satisfied than were previously, but with hill climbing the algorithm can now also work towards optimizing

the size of the circuit. This means it will compare the number of satisfied rules as well as the number of gates, as a circuit size as small as possible is the desired outcome.

Figure 2 shows an overview over the entire genetic algorithm cycle.

3.4 Theories of the Syllogism

Syllogisms are a domain of deductive reasoning. They usually consist of two premises in the form of 'All A are B', 'Some X are not Y' and similar ones, making statements about three groups in total who are in relation to each other. Based on those statements a logical conclusion must be drawn, in the same form as the premises. Some syllogisms allow several correct conclusions, and for others there does not exist any valid conclusion. Syllogisms have been used since ancient Greece to model logical thinking as summarized by Smith et al. [21], and since they are easily understood by humans, the answers humans propose for each of the problems may be helpful in understanding humans' reasoning processes.

Table 2: Abbreviations for the four syllogistic moods.

Abbreviation	Meaning
A	All X are Y.
E	No X are Y.
I	Some X are Y.
O	Some X are not Y.

There are four different types of statements in syllogistic reasoning, called moods, that each contain an existential or universal quantifier as well as a positive or negated statement. The moods are traditionally abbreviated with the letters A, E, I, and O, as seen in Table 2. It is also relevant for syllogisms to differentiate in what configuration the statements appear, as well as the sets they argue about. The

Table 3: The four syllogistic figures.

Figure	Structure	Example
1	A — B B — C	All corgis are dogs. All dogs are mammals.
2	B — A C — B	Some dogs are corgis. Some mammals are dogs.
3	A — B C — B	All corgis are dogs. Some mammals are dogs.
4	B — A B — C	Some dogs are corgis. All dogs are mammals.

Table 4: Abbreviations for the directions of syllogistic answers. The direction always refers to the ordering of the sets A and C of the syllogistic premises, as they appear in the answering mood.

Abbreviation	Direction
ac	Corgis → Mammals
ca	Mammals → Corgis

different possible types of configurations are called figures and can be viewed in Table 3. As there is four moods and four figures which can be combined in any way in syllogistic reasoning, this gives a total of 64 possible syllogisms. The answers to the syllogisms consist again of a mood, as well as a direction of the relation. For example if a syllogism were to consist of the statements 'All penguins are flightless animals.' and 'All penguins are birds.', which would make it the syllogism $AA4$, a possible conclusion would be 'Some birds are flightless animals.'. As 'flightless animals' is, going from the figure, set A of the statement, and 'birds' is set C, the conclusion is in backwards direction. Directions of the answer are abbreviated *ac* and *ca* respectively, as seen in Table 4. If no conclusion for the answer is possible, the abbreviation NVC is used which is short for 'no valid conclusion' [22, 1].

In the field of psychology, many studies have been conducted and theories created to try and explain human syllogistic reasoning. Some of those theories are heuristic and focus more on explaining why humans do not reason deductively and may give incorrect answers, others are based on modelling the process of human thinking [1]. Most of the theories are focused on explaining answers that may occur instead of directly predicting them, however they are still of good use in evaluating and understanding the formulas that are generated by the genetic algorithm.

The first such theory that was proposed in modern psychologic research, and also a good example for a heuristic theory, is the atmosphere theory as first published by Sells et al. [23, 24]. It proposes that the existential quantifier as well as the presence of negativity gives the syllogism a so-called atmosphere, which influences a majority of people to include those properties in their proposed answer for the syllogism. Otherwise, atmosphere theory proposes a bias to non-negative, universal answers. Table 5 gives a brief overview over these heuristics. The atmosphere theory has some weaknesses, namely that it cannot say anything about the direction of the conclusion. It also does not take into account the figure of the syllogism for reaching the conclusion, and it does not factor in NVC answers. However, it still makes a good prediction about universality and negativity for a majority of answers. Another example for a heuristic theory is the matching theory by Wetherick et al. [25], which is somewhat analogous to the atmosphere theory and proposes that individuals answer with the mood of the premise which describes the existence of fewer entities.

An example for a model-based theory is the mental models theory, which was developed by Johnson-Laird et al. [26, 27]. It proposes that humans first construct a mental representation of the problem to depict the relations of the different sets of the syllogism. The arrangements are presented spatially in a way to fit with the premises, and from this spatial model is then drawn a conclusion. If this conclusion

Table 5: A summarized version of the heuristics of the atmosphere theory.

Precondition	Answer
Both premises are univeral and affirmative.	A
Both premises are universal and one or both are negative.	E
One or both premises are existential and both are affirmative.	I
One or both premises are existential and one or both are negative.	O

seems implausible, people attempt to falsify it by finding a possible counterexample that fits with their model. If this succeeds, they search for an alternative model of the premises and try to draw another conclusion. This step is repeated until either a valid conclusion is found or the individual gives up and draws the conclusion of 'no valid conclusion'. An example of the reasoning process can be seen in Figure 3. A study by Johnson-Laird et al. on the subject however has shown that only a select people include the step of falsification, and even fewer attempt to construct an alternative model, but those who do attempt to use the falsification step in their deduction process are more likely to give correct answers [28]. Polk and Newell propose a variant of the mental model theory called the verbal models theory [29], which does not include the falsification step. The argue that instead of spatial, the model is constructed in verbal terms, and that people do not falsify, but instead only re-construct the model if no conclusion can be drawn from the previous attempt. With this model too, the answer 'no valid conclusion' is given after enough failed attempts to draw a conclusion.

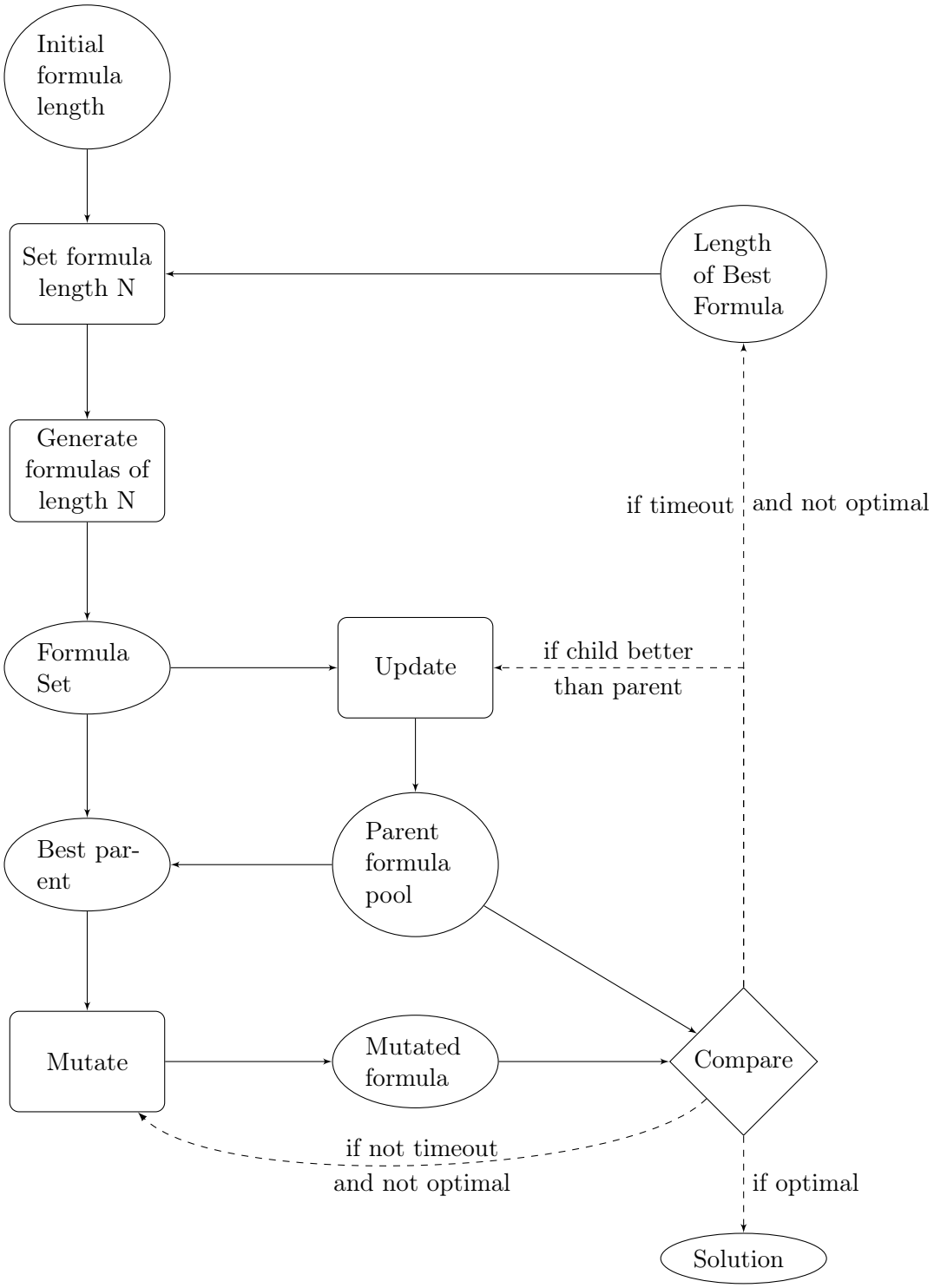


Figure 2: Cycle of the genetic algorithm to generate circuits. A set of potential parent solutions is randomly generated and continuously mutated until a timeout is reached. The best solution serves as a basis for the following iteration.

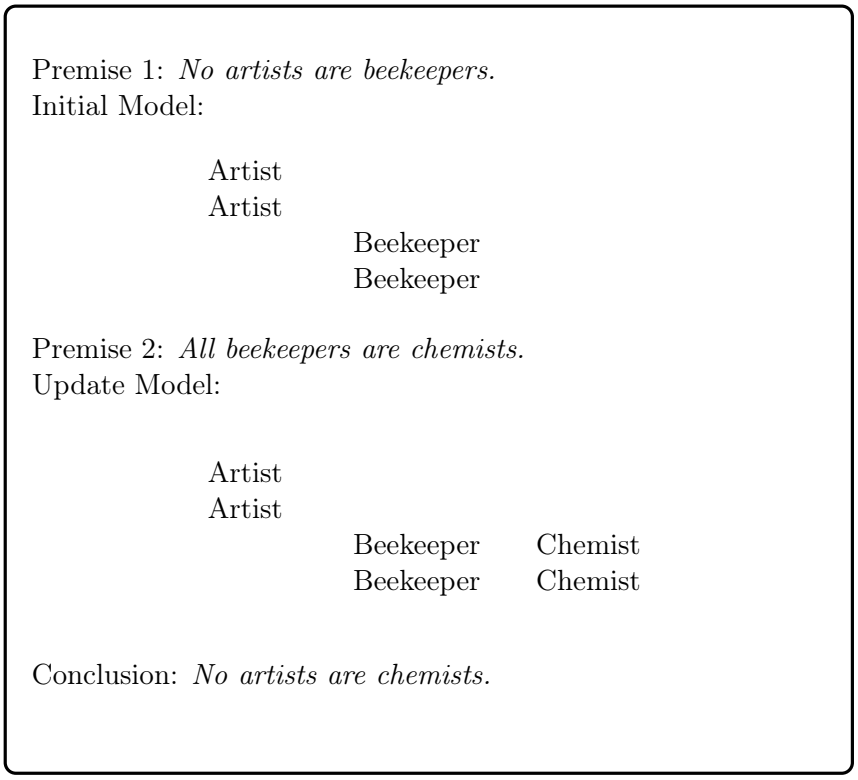


Figure 3: An example to outline the process of reasoning as proposed by the mental models theory. The individual first arranges the two groups spatially to fit the premises, and then draws a conclusion from the resulting model, which is incorrect in this case. Depending on the individual, a validation process may follow, where an attempt to falsify the conclusion and a subsequent try to construct a new model are made.

4 Approach

4.1 Problem Definition

The goal of my work was to find boolean formulas to predict human answers for the syllogisms that are both easily understandable and accurate. The problem is therefore how to encode the syllogisms and answers in bit-representation, and represent the given data as rulesets in a way that can be used by a genetic algorithm to generate boolean circuits.

4.2 Matching Answers with Input

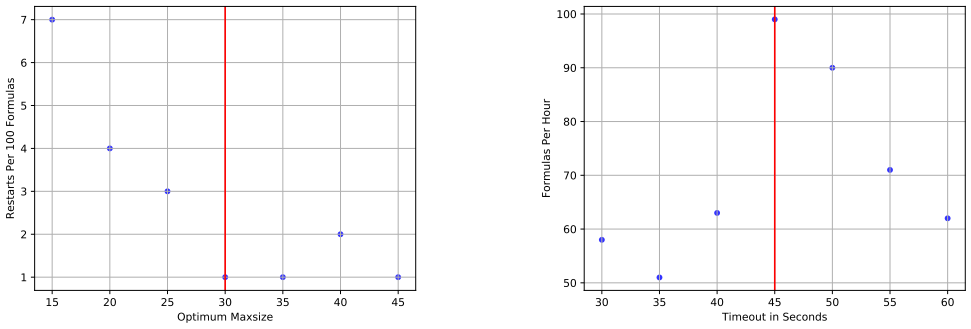
Opposed to most syllogistic theories, boolean circuits do not accept several answers for the same input. They are deterministic and only one answer can be given for any problem. This raises the first question of how to deal with differing answers for a syllogism. With this limitation, two different methods for generating a theory with genetic algorithms were tested. The first one was to calculate the most frequent answer (MFA) for each syllogism and exclude all other answers from the boolean ruleset. The second one was to maintain a set of formulas rather than only one, where a matching occurs if one formula within the theory can predict the answer. For the latter, the goal was to find a set of formulas as minimal as possible.

4.3 Dividing the Problem

One drawback of genetic algorithms is that their processing time can grow exponentially with increasing problem size. After some test runs with a full 64-syllogisms problem set, none of which terminated after several days (for details of the system used please see chapter 5), I split up the problem in smaller chunks. I parted the problem by figure of the syllogisms to have a different formula for each figure, instead of one that covers all figures. I also split it up by answer to have only one output bit per circuit and therefore formula. This gives a theory with one formula per property of the answer (i.e. universality, negativity...) and per figure, resulting in a total 16 formulas for the MFA theory, and a set of sixteen formulas per person for the combination approach. Because of the hybrid approach with hill climbing, the algorithm still has a chance of getting stuck in a local extremum, at which point it needs to be restarted. For this reason, further optimization of the hyperparameters was performed as described in the following section. Yet in general this split up showed to be advantageous for later evaluation, as the partition made it easier to compare the resulting formulas with other theories.

4.4 Expected Size and Timeout

The algorithm uses hill climbing to optimize the size of the circuits that are generated by the genetic exploration, which itself only checks whether a circuit satisfies all input rules. With this hybrid optimization process it is ensured that the solution formula is as short as possible, as well as satisfying all rules. Because hill climbing always requires a goal function, in this case an optimum, to work towards, a value of 'expected optimal size' needs to be determined pre-emptively before the start of computation. In this case, the size is measured by the number of gates and inputs included in the circuit. If a formula generated by the algorithm fits all



(a) Restarts Per Optimum Size

(b) Algorithm Speed Per Timeout

Figure 4: Parameter tuning for the genetic algorithm. 4a shows a range of different optimum sizes as goal for the hill climbing. Optimums were calculated in a distance of five for the expected number of gates of the circuits. 4b shows the different timeout length to minimize the amount of times the algorithm gets stuck in a local maximum. The red lines show the chosen values of the parameters that were used for the experiments.

rules but does not meet the 'expected size' requirement, the genetic algorithm does not accept the formula, and instead continues to mutate. Because the algorithm does not restrict the generation of bigger circuits but only rejects in hindsight the circuits that were already generated, it may happen that processing time increases with a smaller optimum, depending on the complexity of the problem. Likewise, the addition of hill climbing brings with itself the risk of steering into a local extremum, at which point the algorithm becomes stuck indefinitely and has to be restarted. This means that for a smooth running of the experiments, which was neccessary due to time constraints, the optimum size has to be chose in such a way that the algorithm can find solutions that fit the optimum requirement with a good probability. Figure 4a shows the relation between the size of the optimum and the amounts of neccessary restarts. For my experiments with MFA I empirically determined the smallest optimum size that did not require restarting. For the experiments with the formula sets I raised the optimum size to the smallest rate that minimized the amount of neccessary restarts. All resulting optimum sizes can be seen in Table 6.

Table 6: Chosen optima for the different experiments. Sizes were chosen according to a fast performance in formula calculation. As NVC formulas proved during tests to have a longer relative calculation time than the other properties, optima were each set one step (plus five gates expected size) higher for this bit, to have homogenous calculation time.

Method	Universal	Negative	Direction	NVC
MFA	20	20	20	25
Formula Sets	30	30	30	35

Another parameter of the algorithm was a timeout included in the mutation process. The author of the utilized program library states that the longer the mutation takes, the more likely it is that the algorithm will produce an unoptimal mutation [18]. Thus the timeout is there to ensure that too long solution propositions will not be generated, which speeds up the process. Combined with hill-climbing however this brings the risk of steering the algorithm into a local maximum if the timeout is too short. Therefore it was neccessary to test out good timeout values before proceeding with the experiments. Figure 4b shows the number of formulas that can be processed in an hour per different timeout lengths. For my experiments, I have chosen a timeout of 45 seconds.

4.5 Encoding

For the final encoding of the syllogisms, a bitwise encoding in the order of $Figure - Premise1 - Premise2$ was chosen. The different properties of each syllogism were encoded in the order of first the universal quantifier, and secondly the negativity quantifier. In the answer, the direction was encoded as the third bit, *ac* or the forward direction representing 1 and *ca* or the backward direction representing 0. The answer was added a fourth bit to accommodate NVC answers in the ruleset. For the generation of formulas I split the ruleset up in NVC-answers-included and

non-NVC sets, the first of which was used to train only the answers that resulted in NVC, and the second one all other answers. In the prediction tool NVC was therefore prioritized above all other answers, which means that if a formula predicts NVC it does not matter what all other formulas predict because NVC outvotes all of the other bits. This rule separation is to ensure that no unneccessary information from the NVC calculations will flow into the generation of the first three bits, seeing as the first three bits could technically be either 0 or 1 when the answer is NVC, but such a state is not possible in a boolean circuit. An overview of the final encodings can be seen in Table 7.

Table 7: Bitwise encoding of the syllogisms into binary. Ordering for the problems was $Figure-Premise1-Premise2$, and for the answers $Mood-Direction-NVC$. In the case of NVC, only the last bit was set to 1 and all others to 0.

Figure	Encoding
1	00
2	01
3	10
4	11

(a) Figures

Mood	Universal	Negative
A	1	0
E	1	1
I	0	0
O	0	1

(b) Moods

Direction	Encoding
ac	1
ca	0

(c) Direction

5 Experiments

5.1 Experimental Setup

The genetic algorithm used was taken from Clinton Sheppard's library of genetic algorithms [18, 19], chapter 16 genetic algorithms with boolean circuits. All experiments were run and evaluated on a *Windows 10* system with an *Intel® Core™i5-3320M* dual-core processor with 2.60 GHz frequency and 8GB RAM. For both the experiments and the evaluation Python 3.6 was used.

5.2 Most Frequent Answer

A natural approach to predicting the syllogistic reasoning of a group of humans is to find the most frequent answers (MFA) of any given syllogisms, and generate circuits to predict them. To achieve this a dataset of 139 people, who each answered logical questions for all 64 syllogisms, was used. Then, a training set of 100 people was randomly selected, whose answers were used as the basis for calculating MFA outputs for each syllogistic input. These formed the genes for the genetic algorithm. The resulting formulas that were generated were evaluated by their accuracy with the remaining 39 people as the test set. The performance of each formula can NVC seen in Table 8.

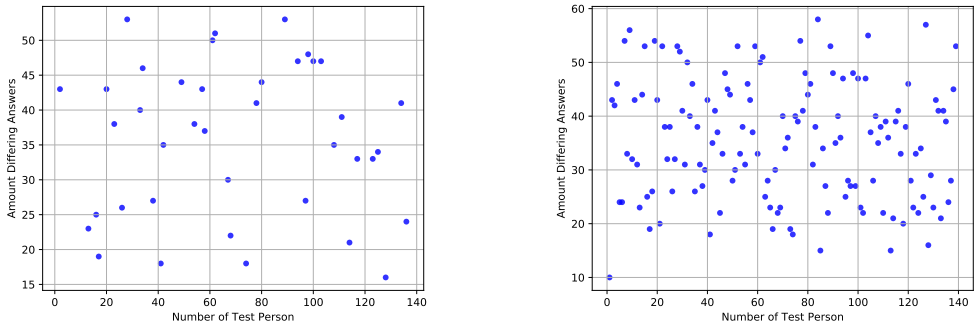
Table 8: Percentage of correctly predicted answers for each MFA formula. Performance was calculated based on the answers of a testing set of size 39.

Bit	Figure 1	Figure 2	Figure 3	Figure 4
Universal	87 %	84 %	77 %	82 %
Negative	72 %	78 %	79 %	76 %
Direction	69 %	77 %	64 %	70 %
NVC	67 %	69 %	66 %	62 %

Figure 5 shows the amount of answers per person, out of the 64 syllogisms, that differ from the predicted answers of MFA. As can be seen in both graphs, the accuracy varies wildly and there is no person in either the testing or training group that can fully be predicted with the MFA. In total, MFA achieves only 44 % accuracy of prediction, and is therefore not a very good tool to get a full understanding of how human thinking solves the syllogism. However, it can give a good indication about what a large group of people answer. The question that arises is whether it will be possible to group people together based on differing answers, and gain insight by predicting the answers after having identified what hypothetical answering group a person belongs to.

5.3 Prediction Sets

Instead of using just one formula another option is to maintain a whole set of formulas per bit and figure, and then attempt to gain a good coverage of the different answers with this set. This formulates a different approach where the goal is not to predict as many answers as possible with a single formula, but find one or more formulas within the set that can predict the given answer. The question is whether it is possible to form clusters of people with such sets, where disjoint or at least roughly disjoint groups of people can be matched with certain combinations of formulas within the



(a) Testing Group **(b)** Total Group

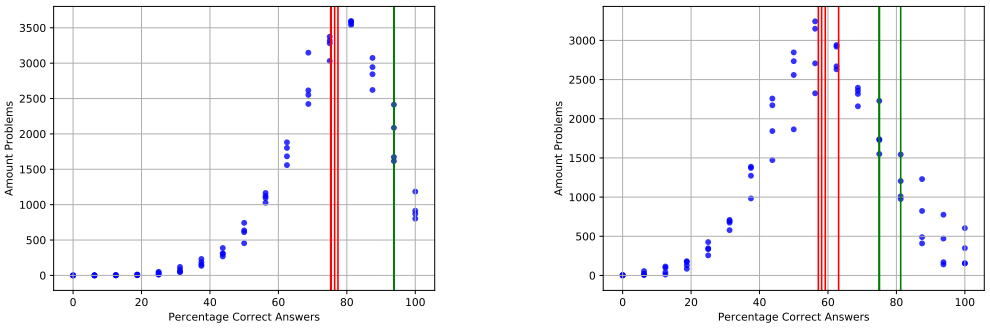
Figure 5: Amount of deviations from MFA prediction per test person. 5a shows only the testing group, 5b includes those people whose data was used to train MFA.

sets. If yes, then it may be possible to predict answers partially, after asking targeted questions to determine which cluster the person belongs to.

Table 9: Figure 1 lower bounds. The boundaries were calculated to cover 25 % of the total number of answers, and rounded up if there was not found a bound that hit the number exactly.

Bit	Figure 1	Figure 2	Figure 3	Figure 4
Universal	$\frac{15}{16}$	$\frac{15}{16}$	$\frac{15}{16}$	$\frac{15}{16}$
Negative	$\frac{13}{16}$	$\frac{13}{16}$	$\frac{13}{16}$	$\frac{13}{16}$
Direction	$\frac{12}{16}$	$\frac{13}{16}$	$\frac{12}{16}$	$\frac{12}{16}$
NVC	$\frac{14}{16}$	$\frac{14}{16}$	$\frac{13}{16}$	$\frac{13}{16}$

To find such sets a first necessary step is to find a measure for the quality of a formula. Obviously this can be measured by how many answers the formula can accurately predict within the test group. But if only the people that match with the formula in all 16 answers are taken into account, it will be difficult to find a good

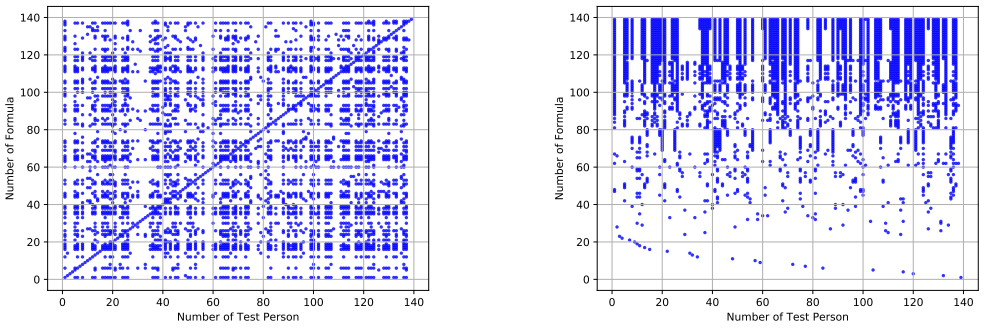


(a) Universal Distribution **(b)** Direction Distribution

Figure 6: Selected total distributions. The red lines show the mean performance of formulas for the given bit and figure, the green lines show the calculated lower bound according to Table 9.

coverage of the group without using too many formulas. Therefore the approach in this work was to calculate a performance mean of each formula that represents the average percentage of correctness in prediction of the entire test set, and then define a lower bound to represent the minimum required correctness percentage. A formula then "covers" a person if all answers the person gives reach a total accuracy that lies above the minimum required correctness border. The lower bounds were chosen high enough to still give a good accuracy for the formulas, and give a none-too-large coverage of the testing group so it will be possible to form somewhat distinguishable sets. The bounds were chosen according to Table 9 to account for rounded $\frac{1}{4}$ of all possible answers for each formula, as can be seen in Figure 6.

Figure 7 shows the performance of all 139 formulas predicting the first syllogistic figure's universal parameter. The blue line in the middle depicts the identity of each formula and therefore indicates the correctness of the calculation. Seeing as each formula should have 100 % accuracy for the person it was generated from, it of course has to cover said person. From the unordered graph it is evident already that groups of similar coverages can be formed, and even more so in the ordered graph, which shows that there are actually many formulas that cover an identical to



(a) Unordered total performance

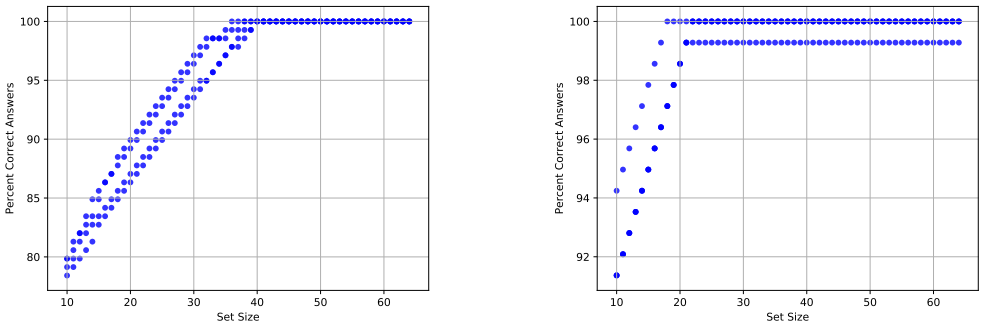
(b) Ordered total performance

Figure 7: Total performance of figure 1 universal formulas. Using the given lower bounds, this graphics shows the total coverage of the test group for the different formulas, meaning in the case of figure 1 universal, a formula covers a person if 90 % or more of their answers are correctly predicted by the formula. 7a shows the formulas unordered, enumerated according to the person they were generated from. 7b shows the formulas in descending order of how many people they cover.

near-identical set of people. To factor these out of the set calculation, I first grouped all the formulas with identical coverage together to get a set of unique formula types. Going from the results of the MFA algorithm, I then formed clusters by order of descending coverage size of the formulas. The idea was that if MFA covers only one group of typical answers for people, then there can be found smaller formulas that will cover the rest.

Table 10: Difference rates with best coverages per figure. All rates were calculated with a set size of 10.

Bit	Figure 1	Figure 2	Figure 3	Figure 4
Universal	0.2	0.1	0.1	0.45
Negative	0.15	0.1	0.15	0.1
Direction	0.2	0.3	0.25	0.2
NVC	0.35	0.45	0.45	0.25

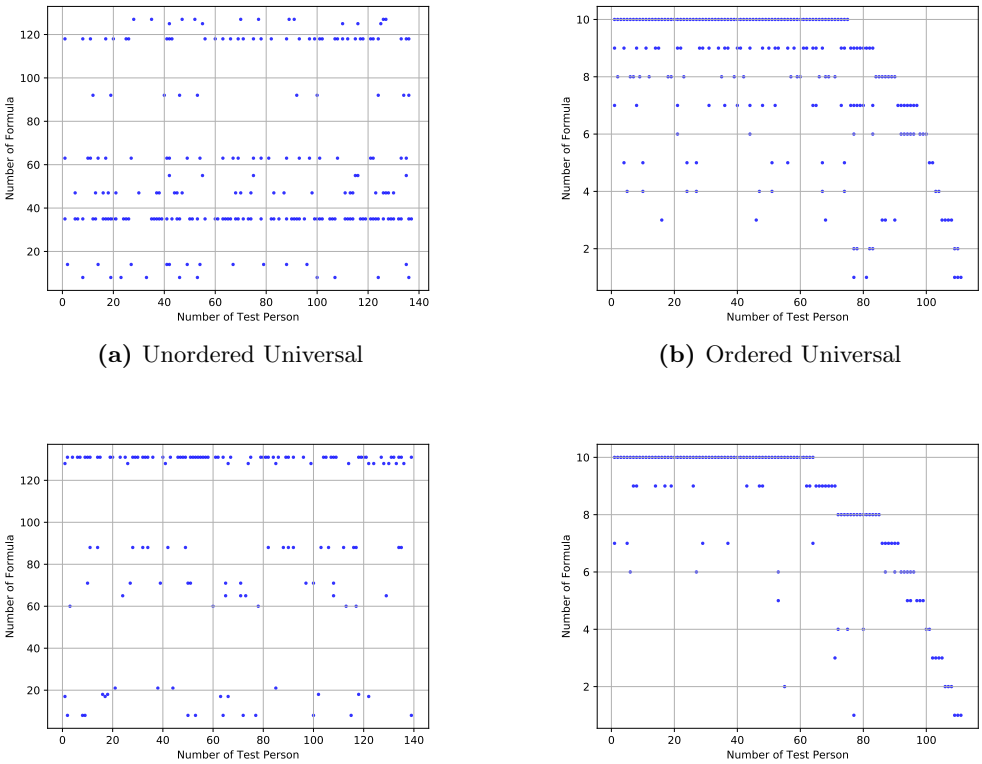


(a) Universal **(b)** Negative

Figure 8: Growth of coverage per set size. Differing percentages for all figures are included, starting from a set size of ten.

To find a set of formulas with good coverage one firstly needs to determine a good set size. Figure 8 compares the prediction coverage of the testing group achieved with different numbers of formulas. As evident, trying to achieve a coverage close to 100 % lets the set size grow linearly, as was to be expected as there are a select number of unusual people, which are only covered by a small number of formulas covering little else in return. As my goal was to form a set with a small size for easier comparison purposes, I first settled for a coverage of around 80 % of people, which was reached by using a set size of 10. Also to be determined before further evaluation was the difference rate between the formulas, meaning the percentage of people covered by exclusively one formula within the set. For a clustering of people according to the formulas, a large difference rate is needed, but depending on the coverage of the formulas this may again result in a bad total coverage of the set. Table 10 shows the difference rates with the best performance of all figures and bits. As can be seen already, clustering may be hard to achieve with this method for most formula types.

In Figure 9 this can be seen even more clearly. While for the figure 1 NVC formulas it was possible to build somewhat disjunct sets of people for the formulas, this was not readily so for the universal formulas and most other formula types. People don't



(a) Unordered Universal

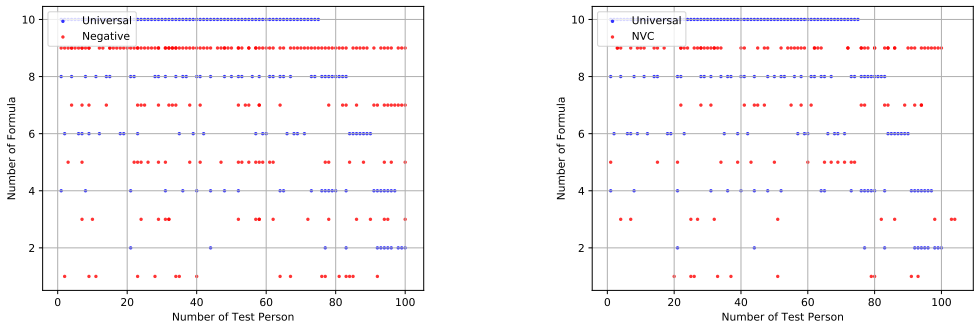
(b) Ordered Universal

(c) Unordered NVC

(d) Ordered NVC

Figure 9: Figure 1 best coverage with a set size of 10 for selected bits. This graphic shows the performance of the best cover formulas for universal and NVC each. 9b and 9d respectively order the people along the abscissa so that people who belong to the same formula cover set will be closer together, for better comparison.

really belong into distinct behavioral groups with similar answering patterns, and instead will give a more individually determined combination of answers. Comparing the formulas for different bits of a figure with each other yields similar results, as shown in Figure 10. Here too a disjunct grouping of the different sets is not easily possible, as there are many candidates who fit into the clustering for one group, but fall outside of it for a similar group of a different bit. This gives a decided indication that clustering people into different answer groups is not possible, and fits with what research has shown so far in this area.



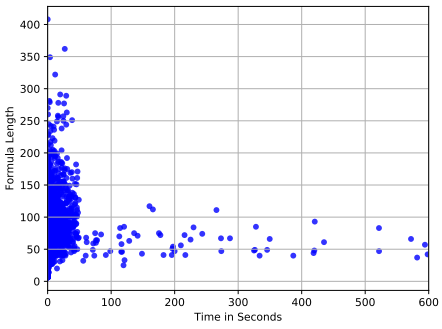
(a) Comparison Universal vs. Negative

(b) Comparison Universal vs. NVC

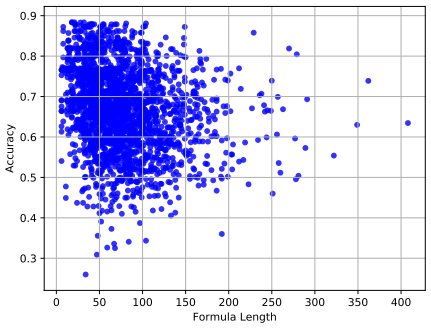
Figure 10: Figure 1 comparison between bit sets. This graphic shows the performance of the best cover formulas for universal and negative bits, and universal and NVC in comparison. For easier visibility, only the first five sets per bit are shown in this graph.

5.4 Generated Formulas

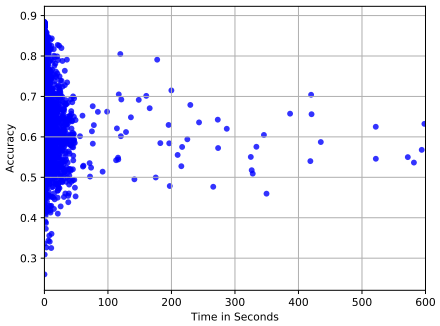
At last the formulas that were generated by the genetic algorithm will be examined a bit closer. Generally, it should be said that performance varies widely for the generation of the formulas. For some test persons it only takes a short processing time and the formulas are easily understandable. For others generation of the circuits takes a long time and frequently needs to be restarted. A supposition was that there was a correlation between these factors, which is why I compared the data of processing time per formula with the length and accuracy, as seen in Figure 11. However, as can be seen, no such correlation exists. Yet I will still only look at the more well performing formulas in this section, as results from the previous section have shown that clustering is not possible, and evaluating all 139 formulas would go beyond the scope here. For each figure and bit I have therefore chosen the best MFA formulas as well as some of the best performing single person formulas, according to the coverage accuracy percentage as calculated in the previous section. The latter were selected with exceptions, as a rare few of the single person formulas were already too long and complex.



(a) Length per processing time



(b) Accuracy per length



(c) Accuracy per processing time

Figure 11: Comparison of processing time, length and accuracy per formula.

Table 11 shows the MFA results which in most cases proved to be the simplest and shortest formulas. Looking at the quantifier bits of figures 1 and 2, it becomes apparent that MFA follows in large parts the atmosphere theory, with small exceptions like for figure 1 where a conclusion will only become negative if exactly one of the base premises is. It is notable that for figure 2 both exclusive and simple OR of the premise negativities is a valid prediction for answer negativity. At first glance this seems contradictory, however when looking at the NVC strategy for figure 2 proposed by MFA, the case where the negativity formulas would contradict each other never happens. The conclusion is always NVC if the second premise starts out

Table 11: Best resulting formulas of MFA encoding. $U1$ denotes whether premise 1 is universal ($True$ if yes, $False$ if no). $U2$ likewise tells the same about premise 2. $N1$ denotes whether premise 1 is negative ($True$ if yes, $False$ if no), and $N2$ does the same with premise 2.

Figure	Universal	Negative	Direction	NVC
1	$U1 \wedge U2$	$N1 \oplus N2$	$\top$	$\neg U2 \vee (N2 \wedge U2)$
2	$(N1 \vee N2) \wedge U2$	$N1 \oplus N2$ and $N1 \vee N2$	$\bot$	$N1 \vee (\neg U1 \wedge \neg U2 \wedge N2)$
3	$\neg(U1 \oplus U2)$	$N1 \vee N2$	$U2$	$(U1 \wedge \neg N1 \wedge U2 \wedge \neg N2) \vee (\neg U1 \wedge \neg U2) \vee (N1 \wedge N2)$
4	$U1 \wedge U2$ and $\neg(U1 \oplus U2)$	$N1 \vee N2$	$\neg N1$	$(\neg U1 \vee N1) \wedge (\neg U2 \vee N2)$

negative. The direction of the first two figures matches largely with those predicted by the verbal models theory, which is to be opposites of each other and homogenous. However, a good alternative strategy for figure 1 was to make the forward direction dependant on the non-negativity of both premises as well as universality. For figure 2, a successfull strategy besides "always predict ca" was to use direction ac only if both premises are E. NVC seems to depend on non-universality of the second statement for figure 1, and both premises for figure 2 as well as negativity of the last premises. Again, it seems to largely match the verbal models theory here. A successfull strategy other than MFA for both figures was to never use NVC at all.

Figures 3 and 4 are more complicated, given the somewhat laxer rules for universality, in the sense that neither or both premises can be universal to have universality in the prediction. Figure 4 allows similar to negativity for figure 2 a somewhat contradictory prediction of both AND and XOR formulas. Yet, in both figures the case that neither premise is universal is not reached, as in these cases NVC again trumps the prediction. Thus no contradiction is given. In total, it thus seems likely that figures 3 and 4 too follow the atmosphere theory regarding the quantifiers. The direction is

more interesting here. Figure 3 suggests ac in the case that premise 2 is universal, figure 4 when premise 2 is not negative. Both also accept 'always ac' and 'always ca' respectively as successful strategies, matching their counterparts from figures 1 and 2. Interestingly enough, the formulas quickly become very complicated after these very simple looking, best performing results, suggesting that they adapt to individualistic behavior rather than represent a feasible, generalizing strategy. The same happens, as expected, with NVC, as evident even from the MFA results already. One can however see in lieu of complexity of the formulas, that absence of universality and presence of negativity seems to be an important factor for NVC in these two figures.

To summarize, in most somewhat successful strategies the quantifiers follow the atmosphere theory or a variation or specification of it, with formulas getting more specific the less accurate they are, but most still containing the MFA formulas in their larger formulas in some manner. For direction and NVC things look different. NVC especially seems to depend on negativity and non-universality, which makes sense seeing as these properties contain less information than their counterparts. However, evaluation for the NVC formulas is complicated as not all of them are variations of each other in the same fashion as the formulas for the quantifiers, and the formulas also quickly become large and complicated, especially for the latter two figures. As direction and NVC appear to also be more easily clusterable, as explained in the previous section, this was to be expected.

5.5 Summary

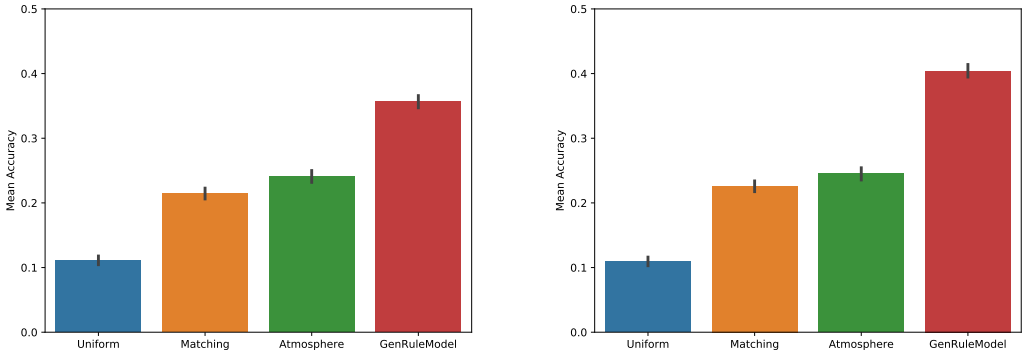
Tests were conducted to assess both the performance for formulas generated on the basis of the most frequent answer of the given data set, as well as for formulas tailored to singular test persons' answers. A clustering with the given methods was not possible, and while the accuracy for the singular properties of an answer (i.e. universal, negative...) could achieve a high overall accuracy, put together the accuracy

could only reach a percentage of 44 %. The best performing formulas generated by the algorithm mostly showed similarity with either the atmosphere or verbal models theory.

6 Evaluation

To test out the actual performance of the formula sets calculated in chapter 5, they were given to a portfolio algorithm inspired by the approach of Riesterer et al. [13]. The approach was adapted for these purposes, combining rules based on a principle similar to portfolios to perform a voting more on an individual basis and make the formula sets adapt to each test person gradually over time. The algorithm starts by initializing weights for each formula and predicts the answers of a person for each syllogism by letting the formulas give a weighted vote. The weights of the formulas are updated continuously to reward correct answers, thus giving formulas that make correct predictions have a higher voting power.

A benchmark of Riesterer et al. [30], also based on their portfolio approach, was used to evaluate the predictions. The adaption was performed online, allowing the benchmark to update weights of successful formulas after each prediction. The test runs did not start with any previous knowledge about an individuum, but instead grew to anticipate the answers of each test person over time. After an individual run was finished, all voting weights were reset to their initial values again. Two different test runs were made, one with the configurations that were calculated in chapter 5 (see Tables 9 and 10) and a set size of 10. The other was conducted with maximum configurations, which is 100 % accuracy required for all formulas in the set and no limit to the set size. The answers were evaluated by a testing set of 139 people. The resulting total accuracies can be seen in Figure 12.



(a) Calculated Configurations **(b)** Maxed-Out Distribution

Figure 12: Performance of selected formula portfolios. Compares accuracies of the testing set (red) with atmosphere (green), matching (yellow) and uniformly random (blue) strategies. 12a shows the results with configurations as calculated in the previous chapter, 12b shows the results with maxed configurations, meaning 100 % accuracy for a formula to be included, no difference rate between formula coverages, and no set size limit.

As can be seen, with as small a set size of 10 formulas each, a significant improvement in prediction accuracy compared to the psychological theories is reached, and without limit on set size the results even come close to MFA accuracy (around 44 %), as calculated in the previous chapter.

7 Conclusion

This work showed that genetic algorithms can create simple and effective formulas for predicting human answers for syllogistic problems, and that genetic algorithms for boolean circuits can be used generally to find solutions for a large space search problem, given that the inputs and outputs are encoded to fit the problem as well as the appropriate hyperparameters. In this work, formula sets were generated as theories to adapt to an individuum in terms of prediction, which was more successful than the compared psychological theories. The formula sets were able to reach accuracy rates of almost 44 %, which is the accuracy reached by the most frequent answer strategy. This marks the question why the individual adaption could not surpass the most frequent answers. For this there could be two reasons, one would be that the algorithm first has to 'learn' most frequent answer before it can surpass it, for which there might not be enough trial time. Another reason could be that a lot of the formulas generated by the genetic algorithm appear themselves to be only variations, or specifications, of the formulas created by most frequent answer. In the future it could be interesting to try out how the individual adaption of the portfolio approach performs given more runs per individuum. Another topic that could be interesting for the future could be to try out how the adaption would fare if the algorithm already had previous knowledge on the basis of most frequent answer, and would adapt its formulas from this pre-knowledge instead of uniformly.

In singular comparison, the formulas performed extremely well, reaching up to

80 % accuracy. This means of course, that to achieve a better total rate for the prediction, the singular formulas would have to reach almost perfect scores nearing 100 %. This raises the further question whether or not it will be possible to make better prediction with a formula that was optimized for all figures and moods simultaneously. However, as one single formula can of course not reach better results than the most frequent answer, the individual adaption strategy has the most promising outlook in this field. Otherwise, another question would be if the strictly deterministic boolean strategy should even be forgone entirely, and a more probabilistic approach could be used.

8 Acknowledgments

First and foremost, I would like to thank...

- my advisers Daniel Brand and Nicolas Riesterer
- my examiner Prof. Dr. Marco Ragni
- Robert Grönsfeld for proofreading
- the Fachschaftswebsite for the LaTeX-Template

Bibliography

[1] S. Khemlani and P. N. Johnson-Laird, "Theories of the syllogism: A meta-analysis," *Psychological bulletin*, vol. 138, no. 3, p. 427, 2012.

[2] M. Mitchell, *An introduction to genetic algorithms*. MIT press, 1998.

[3] G. Boole, *The mathematical analysis of logic*. Philosophical Library, 1847.

[4] B. M. Baker and M. Ayechew, "A genetic algorithm for the vehicle routing problem," *Computers & Operations Research*, vol. 30, no. 5, pp. 787–800, 2003.

[5] S. Ng, S. Leung, C. Chung, A. Luk, and W. Lau, "The genetic search approach. a new learning algorithm for adaptive iir filtering," *IEEE signal processing magazine*, vol. 13, no. 6, pp. 38–46, 1996.

[6] B. Dengiz, F. Altiparmak, and A. E. Smith, "Local search genetic algorithm for optimal design of reliable networks," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 3, pp. 179–188, 1997.

[7] J. Inagaki, M. Haseyama, and H. Kitajima, "A genetic algorithm for determining multiple routes and its applications," in *ISCAS'99. Proceedings of the 1999 IEEE International Symposium on Circuits and Systems*, vol. 6, pp. 137–140, IEEE, 1999.

[8] S. Sadjadi, M. Jafari, and T. Amini, "A new mathematical modeling and a genetic algorithm search for milk run problem (an auto industry supply chain case study)," *The International Journal of Advanced Manufacturing Technology*, vol. 44, no. 1-2, p. 194, 2009.

[9] C. Reis, J. MacHado, and J. B. Cunha, "Logic circuits synthesis through genetic algorithms," *WSEAS Transactions on Information Science and Applications*, vol. 2, no. 5, pp. 618–623, 2005.

[10] S. Karakatic, V. Podgorelec, and M. Hericko, "Optimization of combinational logic circuits with genetic programming," *Elektronika ir Elektrotechnika*, vol. 19, no. 7, pp. 86–89, 2013.

[11] P. Soleimani, R. Sabbaghi-Nadooshan, S. Mirzakuchaki, and M. Bagheri, "Using genetic algorithm in the evolutionary design of sequential logic circuits," *arXiv preprint arXiv:1110.1038*, 2011.

[12] N. Riesterer, D. Brand, and M. Ragni, "A machine learning approach for syllogistic reasoning," in *Proceedings of the 14th Biannual Conference of the German Society for Cognitive Science* (C. Rothkopf, D. Balfanz, R. Galuske, F. Jäkel, K. Kersting, J. Macke, and B. Mohler, eds.), (Darmstadt, Germany), p. 54, Technische Universität Darmstadt, 2018.

[13] N. Riesterer, D. Brand, and M. Ragni, "The predictive power of heuristic portfolios in human syllogistic reasoning," in *Proceedings of the 41st German Conference on AI* (F. Trollmann and A.-Y. Turhan, eds.), (Berlin, Germany), Springer, 2018.

[14] J. Venn, "On the diagrammatic and mechanical representation of propositions and reasonings," *The London, Edinburgh, and Dublin philosophical magazine and journal of science*, vol. 10, no. 59, pp. 1–18, 1880.

[15] W. Gohout, *Operations Research: einige ausgewählte Gebiete der linearen und nichtlinearen Optimierung.* Walter de Gruyter GmbH & Co KG, 2009.

[16] S. J. Russell and P. Norvig, *Artificial intelligence: a modern approach.* Malaysia; Pearson Education Limited, 2016.

[17] C. Darwin, *On the origin of species, 1859.* Routledge, 2004.

[18] C. Sheppard, *Genetic Algorithms with Python.* Austin, Texas: CreateSpace Independent Publishing Platform, 2016.

[19] C. Sheppard, "Genetic algorithms with python, source code." `https://github.com/handcraftsman/GeneticAlgorithmsWithPython`. Accessed: 2018-05-02.

[20] J. H. Holland, *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence.* MIT press, 1992.

[21] R. Smith, "Aristotle's prior analytics," 1989.

[22] I. M. Copi, C. Cohen, and K. McMahon, *Introduction to logic.* Routledge, 2016.

[23] S. B. Sells, "The atmosphere effect: an experimental study of reasoning," *Archives of Psychology (Columbia University)*, 1936.

[24] I. Begg and J. P. Denny, "Empirical reconciliation of atmosphere and conversion interpretations of syllogistic reasoning errors," *Journal of Experimental Psychology*, vol. 81, no. 2, p. 351, 1969.

[25] N. Wetherick and K. Gilhooly, "'atmosphere', matching, and logic in syllogistic reasoning," *Current Psychology*, vol. 14, no. 3, pp. 169–178, 1995.

[26] P. N. Johnson-Laird, "Mental models in cognitive science," *Cognitive science*, vol. 4, no. 1, pp. 71–115, 1980.

[27] P. N. Johnson-Laird, *How we reason*. Oxford University Press, USA, 2006.

[28] P. N. Johnson-Laird and B. G. Bara, "Syllogistic inference," *Cognition*, vol. 16, no. 1, pp. 1–61, 1984.

[29] T. A. Polk and A. Newell, "Deduction as verbal reasoning," *Psychological Review*, vol. 102, no. 3, p. 533, 1995.

[30] N. Riesterer, "Online reasoning for cognition analysis (orca) framework." `https://github.com/CognitiveComputationLab/orca`. Accessed: 2018-09-10.