university of groningen

faculty of science and engineering

Exploring the Limitations of LLMs Logical Reasoning Capabilities Using Knights and Knaves Puzzles

Jonas Scholz

university of groningen

faculty of science and engineering

University of Groningen

Exploring the Limitations of LLMs Logical Reasoning Capabilities Using Knights and Knaves Puzzles

Master's Thesis

To fulfill the requirements for the degree of
Master of Science in Artificial Intelligence
at University of Groningen under the supervision of
Prof. dr. Bart Verheij (Artificial Intelligence, University of Groningen)
and
dr. Cor Steging (Artificial Intelligence, University of Groningen)

Jonas Scholz (s4287797)

July 20, 2025

Contents

Note on the usage of AI

This thesis text was produced without the use of any generative AI. The only external tool used for the creation of the text was the grammar checker Grammarly.

Abstract

Large Language Models (LLMs) are becoming widespread and further reasoning models are being created to improve the performance of LLMs on coding, logic and math benchmarks. Both LLMs and reasoning models seem to exhibit logical reasoning capabilities on some benchmarks. The key issues with these are twofold: firstly, data contamination due to the benchmarks being static and secondly they often do not have scaling complexity, which makes testing reasoning capabilities challenging. To address these limitations, this paper introduces a dynamic benchmark based on the Knights and Knaves Puzzle. It can generate puzzles with a variable number of people in the puzzle and per statement. These two factors are used to control the complexity of the puzzle. Further, variance was added by having exchangeable names, type labels and places to combat data contamination. This benchmark was used to evaluate the reasoning capabilities of contemporary LLMs, using the accuracy on the puzzles as a proxy to determine the reasoning capabilities. The results showed that Gemini 1.5 Pro performed the worst, with a halving of accuracy from the least to the most complex puzzle. In contrast, the reasoning models showed better performance and only showed a slight decrease in accuracy as the number of people in the puzzle decreased. This seems to show that state-of-the-art models are capable of performing well on logical reasoning tasks though the performance does decrease as puzzle complexity increases. Though further tests are needed to determine if that pattern holds or there is a hard breaking point as complexity rises and to determine if the models are actually reasoning by examining the reasoning steps. Overall, this shows that the introduced benchmark is capable of evaluating LLMs and as the complexity is scalable it should mean that it will relevant as more capable LLMs are created, while also side stepping data contamination issues due to the ability of creating variations not only in the logical framework, but also in the translation to natural language.

1 Introduction

LLMs are becoming more widespread for a variety of uses, such as text summarization, coding, but also to assist in decision making for example as a customer support agent[1]. OpenAI alone had 400 Million active users per week in Feburary of 2025[2], this means that it is important to look at the limitations of such systems as this allows for better usage of them.

This is particularly true for the new reasoning models introduced by organizations such as OpenAI. These models are designed to handle more complex problems and operate with greater autonomy by being trained to use tools and to “think” before producing an output[3]. OpenAI claims that these models are capable of complex reasoning using techniques like chain-of-thought reasoning. To prove this, these models demonstrate strong performance on math, coding and further benchmarks. However, while these benchmarks suggest reasoning capabilities, they do not evaluate reasoning directly. These reasoning capabilities belong to a variety of emergent abilities that occur once LLMs become big enough (Wei, Tay, et al., 2022). The performance of LLMs on these different benchmarks, therefore, seems to indicate that the models are reasoning to some extent (Wei, Wang, et al., 2022).

There are some benchmarks that look at reasoning, for example, LogiGLUE (Luo et al., 2024), which is a combined benchmark based on other publicly available benchmarks.

Due to the fact that LLMs use vast amount of training data, a lot of the required data is taken from the internet which can lead to data contamination. This happens when the benchmark or the data source from which the benchmark gets added to the training data of an LLM. THis can then lead to erronously high performance on the benchmark by the affected LLM, as the LLM already “knew” the answer. Data contamination can be mitigated by a variety of strategies such as dynamic benchmarks and private benchmarks as seen in the paper by Xu et al. (2024). One such strategy is the creation of a dynamic benchmark, which can generate specific questions based on some input parameter, which can mitigate the data contamination as a new dataset can be created.

Another challenge in evaluating reasoning is that overly simple questions may be answered correctly through memorization or pattern matching rather than genuine reasoning and also that even complex problems at the moment will become easier in the future with better hardware and architectures. Therefore, to rigorously assess reasoning capabilities, particularly in more complex scenarios, benchmarks that not only avoid data contamination but also scale in difficulty are needed.

An attempt at such a benchmark was done by Steging et al. (2025). In their paper they decribe how they created a benchmark with scaling complexity based on linear argumentation graphs to test the reasoning capabilities of LLMs. They found that while regular LLMs did not perform well on these problems the reasoning model o1-mini-preview did perform perfectly on them and only on non-linear argumentation graphs did not perform perfectly. This shows that there is a need for a benchmark that is more complex.

This paper focuses on logical reasoning as a well-defined and measurable subset of reasoning in general. Logical reasoning is especially suitable for this purpose because it allows for controlled complexity and the generation of unambiguous solutions.

From this the research question can be formulated: What are the limitations of the logical reasoning capabilities of LLMs as the complexity of the task increases?

To investigate the research question, I propose a dynamic benchmark with scalable logic which can be used to determine the logical reasoning capabilities of LLMs. In this benchmark the Knights and

[1]https://blogs.nvidia.com/blog/what-are-large-language-models-used-for/

[2]https://www.reuters.com/technology/artificial-intelligence/openais-weekly-active-users-surpass-400-million-2025-02-20/

[3]https://openai.com/index/introducing-o3-and-o4-mini/

Knaves puzzle is used, which was first introduced in a book by Smullyan (1978). In this type of puzzle, there are two types of people, namely Knights who always tell the truth and knaves who always lie. The reader encounters multiple speakers and needs to determine the type of each speaker based on their statements. Example 1 shows one problem of this type.

> An island is inhabited only by Knights and Knaves, Knights always say the truth and Knaves always lie. You encounter two people. The first person says: ”The other person is a Knave.”, the second person says: ”We both are of the same type.”. What are the types of the two people?

Example 1: An example of a knights and knaves puzzle.

In order to scale the complexity of the puzzle, one can add more people to the puzzle, as this can create longer reasoning chains before the reader is able to determine all types. Furthermore, adding more people per statement might also increase the complexity, as it might mean that more people need to be checked before it becomes possible to determine the type of the speaker. By dynamically generating such puzzles with varying levels of complexity, the logical reasoning capabilities of LLMs under controlled conditions can be evaluated more effectively.
To answer the research question, this paper will first touch on what reasoning is and what other benchmarks have done so far and why they are not sufficient. Then the paper explains Knights and Knaves puzzles. Before going over how the dynamic benchmarks works. Finally, the benchmark is tested on the models Gemini 1.5 Pro, o1-mini and o4-mini.

2 Background Literature

2.1 Reasoning

On September 12, 2024, OpenAI introduced a new category of LLMs, called reasoning models[4]. They claimed that these models can solve logical problems, math and other complex issues. While it seems intuitively clear that complex issues require some kind of reasoning, mirroring how humans typically solve these issues, it is necessary to define what is meant by reasoning.
To establish an initial working definition, dictionary sources offer a useful starting point. In the Merriam-Webster dictionary, reasoning is defined as “the use of reason” and especially “the drawing of inferences or conclusions through the use of reason”[5]. Further, reason is defined as “a statement offered in explanation or justification” or “a rational ground or motive”[6]. These definitions suggest that reasoning involves using some facts with appropriate transitions to get to possibly new facts (the conclusion). Reasoning, therefore, can be characterized as the inference from information to a conclusion.

2.1.1 Human Reasoning

There seem to be different approaches to defining reasoning in humans one of which is a rule-following theory, which is held by some researchers. This idea, which is critically discussed in the

[4]https://openai.com/index/introducing-openai-o1-preview/
[5]https://www.merriam-webster.com/dictionary/reasoning
[6]https://www.merriam-webster.com/dictionary/reason

paper by Boghossian (2014), proposes that humans actively apply inference rules on the contents of their premises to come to a conclusion. These rules need not be explicitly formulated in the mind of the human. This theory gives a reason as to why, intuitively, humans feel like the premises are the reason for their conclusion, but are not necessarily be able to say exactly what they did to follow these rules.

While this theory might seem plausible there is a significant flaw that Boghossian (2014) points out, namely that in order to be able to actively follow a rule, reasoning is required to apply a rule correctly. This would, per the rule-following theory, lead to the application of another rule, which would lead to the initial situation again and therefore an infinite recursion, which is implausible. For the theory to be valid, the author supposes that there must be a way to follow rules without reasoning.

Some researchers think that the rule-following theory is not sufficient to explain reasoning, but that reasoning is aimed at achieving specific fitting attitudes, which leads to their theory of reasoning, which is the goodness-fixing theory (McHugh & Way, 2018). In their paper they explain that fitting attitudes should be considered as people trying to get things right. In their theory they therefore include only activities that facilitate getting to fitting attitudes to reasoning. This could mean that only rules that fit the intended aim of the reasoning should be considered part of the reasoning.

2.1.2 AI Reasoning

In the AI domain, the main focus is on building systems that exhibit reasoning capabilities, though often the focus is on reasoning in specific fields such as logical reasoning or mathematical reasoning. One main way to create systems that can reason is to create systems that can apply rules to some information to get to further conclusions. An obvious way to create systems that can reason correctly is to base the systems on a logical framework which are proven to be consistent, such as default logic (Delgrande, 1988). This would then allow the system to use rules such as the modus ponens to reason about its knowledge and also update its knowledge if the system got relevant information. Some issues with such systems could be that while such systems reason correctly according to the framework it might be non-trivial to transform the data into the correct logical format, especially if the data is natural language.

Another way to build reasoning systems is a data-driven approach, these systems usually learn directly from the data, and therefore do not have translation issues that are as big as the previous systems can have. An example of such a system could be an LLM, which can take natural language as input and will output natural language again. This means that it can process the data and, with enough data, can even show forms of for example mathematical reasoning (Wei, Wang, et al., 2022). Given the nature of LLMs reasoning is often mentioned when the model is prompted with chain-of-thought (CoT) reasoning, which encourages the model to produce intermediate results before the final solution. This is thought to emulate the process humans use to reason, though the main goal is to achieve better results and the question of if the models are actually reasoning is not considered (Wei, Wang, et al., 2022).

2.1.3 Reasoning Models

While OpenAI was first to release their o1 reasoning model they have not given a detailed explanation of how the model works. In contrast, the paper from Deepseek about their Deepseek-R1 reasoning model (DeepSeek-AI et al., 2025) offers such an explanation. In their paper, the authors describe that they used CoT to prime the model for human-readable “thinking”.

This “thinking” process is the same as the normal token generation process. However, the “thinking”

tokens are surrounded by special tokens, which are then hidden from the user by default. This means that in essence the reasoning models can be considered to be LLMs that are trained to use CoT to come to their answers.
As with other LLMs, the authors look at reasoning for different domains, such as mathematical reasoning or common-sense reasoning. Nevertheless, the main focus is on the performance of the model on the reasoning benchmarks, and the actual reasoning process is not examined in sufficient detail.

2.1.4 Conclusion

Initially, in this paper, the definition that reasoning is the process of getting from facts using appropriate transitions to conclusions was used as a working definition. This aligns well with the rule-following theory in human reasoning where the application of rules is seen as reasoning. The goodness-fixing theory also goes in that direction, but adds that there needs to be an aim toward reasoning and therefore restricts rule usage towards achieving that aim.
In the AI and Reasoning field, the logic-based systems are, as the name implies, based on logic and therefore rule-based. This means that they align well with reasoning from the rule-based theory. The reasoning in LLMs was mainly about correct answers in different domains, and the actual reasoning seems to be secondary. Nevertheless, using reasoning techniques such as CoT, which adds intermediate steps to the answer, can lead to better results.
So while LLMs depart from the strict rule-following theory, this theory still seems like an appropriate definition to start from for defining reasoning, as it makes the process clearer. The issue with that definition is its descriptive nature, which does not help with evaluating reasoning. To address this, some aspects of the goodness-fixing definition can be included, especially adding an aim for the reasoning allows for better evaluation as the rule's usefulness towards the aim can be evaluated.
This leads to the definition of reasoning that will be used in this paper: Reasoning is the process of applying appropriate rules that transform from premises to conclusions (which may be new premises) such that the aim of the reasoner is achieved. In this paper the aim is the solving of logical puzzles, which means that better reasoning should lead to consistently better solutions.

2.2 Benchmarks

With the definition of reasoning, the next important topic for this paper are other benchmarks for LLMs. There are two main ways to look at reasoning, firstly directly looking at the reasoning steps (genuine reasoning) or secondly looking only at the result of the reasoning (apparent reasoning) to determine the reasoning.
Firstly, the paper "GPT-4 can't reason" by Arkoudas (2023) examines GPT-4 from OpenAI on 21 diverse problems, including arithmetic, graph coloring and elementary logic. The authors define reasoning differently in their paper, namely by seeing it as a justification for a conclusion, this is similar to the way reasoning is defined for this paper as a rule-following towards an aim. So given this definition the authors perform a qualitative analysis of the performance of GPT-4 on these problems and show that the model has internal inconsistencies in their responses, such as replying that there are seven instances and then providing a list which only includes four instances. This leads the author to conclude that GPT-4 cannot reason.
In contrast, the paper by H. Liu et al. (2023) focuses on the performance of the models in benchmarks for multiple-choice questions and natural language inference. The multiple-choice benchmarks are based on question which test logical reasoning in humans and are taken from real exams. The natural language inference benchmarks consist of two sentences, and the model must determine if the second

sentence is an entailment, a contradiction or neutral to the first sentence. The authors show that the LLMs outperform fine-tuned models, but that the LLMs perform significantly worse in out-of-distribution datasets.

This finding is further supported by Mirzadeh et al. (2024), who introduce a new arithmetic benchmark based on the GSM8K benchmark. Their benchmark questions follow the same format as the GSM8K, but they allow for the variation of the names of the people and other labels in the problem description, as well as a variation in the actual values. This design allows them to show that with simple changes to the values and labels, without increasing the complexity of the problem the tested LLMs perform worse than compared to the original GSM8K, which they attribute to possible data contamination due to the prevalence of GSM8K.

A similar approach is taken by Steging et al. (2025), who developed a dynamic benchmark of scaling complexity: In this benchmark, the model must assess whether a person's statement should be believed based on the statements of multiple people. This is done by creating argumentation graphs, which are translated into natural language with randomly chosen names and subjects. This design helps to prevent possible data contamination from occurring, which might be the case with fixed benchmarks. The authors found that most models had problems with even-length attack chains. Performance worsens further when non-linear attack chains are introduced, and even reasoning models fail to solve all problems correctly in that case.

As shown by these papers, it remains unclear to what extent the models can reason. A promising approach to test this is the creation of a dynamic benchmark, as this can reduce issues regarding data contamination, which allows the benchmark to stay relevant. Finally, while Steging et al. (2025) also created a benchmark to test reasoning, their benchmark complexity is polynomial, which might be sufficient for current models, but benchmarks based on higher complexity classes might scale better in case models become more powerful in the future.

2.3 Knights and Knaves Puzzles

To achieve a dynamic benchmark with higher complexity Knights and Knaves puzzles are a good candidate, as they have NP-complete computational complexity (see Appendix A).

A possible strategy used by humans to solve such puzzles is by starting to assign a type to the first person. Then, based on that, the statement of the person is evaluated and based on that, other people are assigned types. If that leads to a contradiction, then the first person has the other agent type. If all people were assigned an agent type without a contradiction, then a solution was found. If not all people were assigned a type, then this procedure can be repeated, by selecting the type of the first unassigned person. This will eventually lead to a solution or the conclusion that the puzzle does not have a solution. This method, while workable, will, as the number of people in the puzzle increases, lead to more nesting of assumptions, which can make the solution harder to track.

Looking at Example 1, the method can be applied by assigning the first person the knave type. As the first person says: "The other person is a knave.", this means that the other person must be a knight, as the first person is a liar. Looking at the statement of the second person: "We both are of the same type.", this cannot be true as the first person is a knave and the second person is a knight. This means that the second person cannot be a knight, which leads to a contradiction. Therefore, the first person is a knight, which means that, given the first person's statement, the second person is a knave. This fits with the statement given by the second person, as it is false and they are a liar.

Apart from the computational complexity Knights and Knaves puzzles, also offer another interesting aspect, namely, lying and the uncertainty about who is a liar. Lying is a fairly interesting concept not only from a philosophical angle, but also from a logical perspective, as for example in the paper

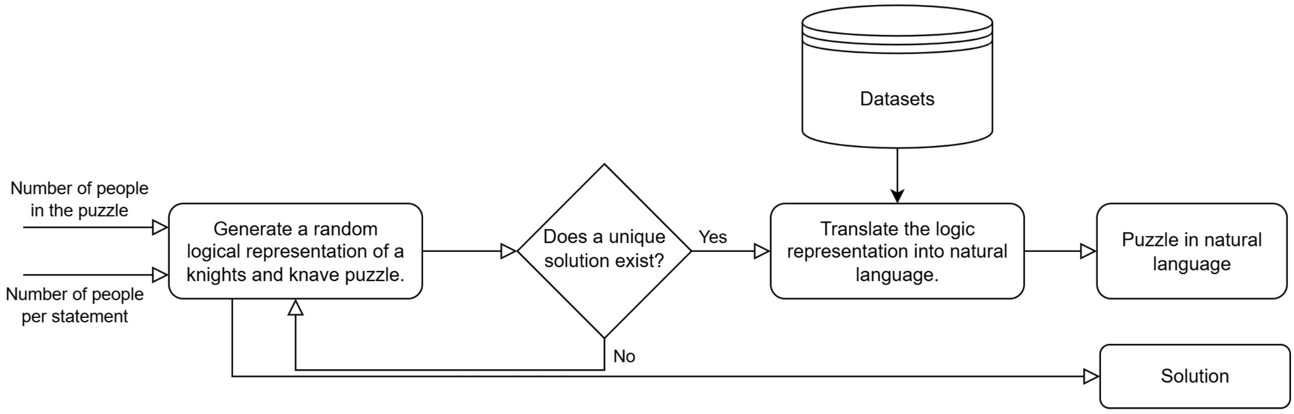

Figure 1: Pipeline to create a knights and knaves puzzle.

"Dynamics of lying" by van Ditmarsch (2014), lying and possible receivers are modelled using dynamic logic. For this the author makes use of Kripke models which are a way to represent epistemic logic puzzles, by having different worlds and relations between worlds depending on the ability of the modelled agents to differentiate them. The author defines how conditional action models can be used to update Kripke Models, which is might allow for the modelling of the Knights and Knaves Puzzle.
The Knights and Knaves Puzzles are capable of creating interesting lying constructs, such as the Liars Paradox, through the sentence: "I am a knave" which, as the name implies, is a paradoxical statement. However, there are still some aspects of lying that are not really covered. For example, an important part of lying is the intent to deceive as outlined by van Ditmarsch, van Eijck, Sietsma, and Wang (2012) who analyze the game Liar's dice. They argue that the intent to deceive may be missing, as no agent is truly deceived; deception is always considered a possibility in this context. This aspect, therefore, is likely to be missing as well from the Knights and Knaves puzzle, as the reader is tasked with finding the liars and is therefore considering any statement to potentially be a lie. This creates a setting that differs from real-world scenarios, where a general presumption of truth may be more common.
Looking at other formalizations of lying and puzzles there is a paper by F. Liu and Wang (2013), which formalizes the Knights and Knaves Puzzles by using an extension of Public announcement logic with added agent types. This not only allows for the automatic solving of such riddles, but also allows for the automatic generation of such puzzles by using the underlying logic and then translating the logic to natural language. This can allow for the creation of the dynamic benchmark as everything needed for the puzzle can be generated automatically.

3 Methods

As previously discussed, there is a need for a new dynamic benchmark. This benchmark is going to use Knight and Knaves puzzles to test the LLMs. For ease of integration and testing, the programming language Python was used to implement the code that can create this dynamic benchmark. The code for the benchmark can be found on Github[7]
The benchmark follows a structured pipeline consisting of three main stages to generate a puzzle (see Figure 1). Firstly, generating the logic of the puzzle and the solution. Secondly, checking the solution to determine if it is unique, and finally using the logic representation of the puzzle to create a natural

[7] https://github.com/jojoscholzgmailcom/knights_and_knaves_benchmark/tree/main

$$
\begin{aligned}
start &: speech \\
speech &: S_{ID}\, sentence \\
sentence &: sentence \wedge sentence \\
&\mid literal \\
literal &: \neg T_{ID}(ID) \\
&\mid T_{ID}(ID)
\end{aligned}
$$

(a) Standard logic notation.

$$
\begin{aligned}
start &: speech \\
speech &: S_{ID}\, sentence \\
sentence &: sentence \,\&\, sentence \\
&\mid literal \\
literal &: \,-T_{ID,ID} \\
&\mid T_{ID,ID}
\end{aligned}
$$

(b) Custom notation used in the project.

Figure 2: The grammar that the generator can generate in standard logic notation and the notation used in the project.

language version of the puzzle. This pipeline can be repeated as many times as needed to create a dataset of the appropriate size. To achieve this, a generator, a solver and a translator are needed.

3.1 Generator

To generate a puzzle, the number of people, the number of individuals involved in each utterance, and a random seed can be specified. The seed controls randomness and ensures reproducibility.
Initially, a random assignment of types to the people is chosen, which is the target solution. Then, to ensure that the generator generates solvable solutions the program generates a random assignment of who each person should talk about, according to three constraints:

- Everyone talks about the same number of people
- Everyone is being talked about
- No one talks about themselves

The first constraint needs to exist as the number of people per statement is one of the factors to control the complexity of the puzzle. The second constraint ensures that the number of people in the puzzle increases the complexity of the puzzle, instead of only a subset of people being talked about by all people, which would reduce complexity to a puzzle of the number of people in the subset.
The importance of the third constraint becomes clear when examining the possible self-referential statements. For example, if the speaker says, “I am a Knave.” then by definition they cannot be a knight, as a knight could not truthfully claim to be a knave. However, this sentence can also not be uttered by a Knave, as then they would say the truth, which by definition they cannot. The only way this sentence is valid is if a conjunction connects it to at least another false statement. The other self-referential statement “I am a knight” does not provide any information, as both the knight can say it truthfully and the knave can say it, as it is a lie. Due to their limited utility, such self-referential statements are excluded from the generation process.
After this setup, the generator proceeds to construct the logical statements. To do this, the generator iterates over all people and for each one, it generates the logical statement representing their utterance. To generate this logical statement, a reduced grammar compared to the solver is used, as this allows for easier translation to natural language. The produced grammar was constrained such that every logical statement only includes conjunctions, negations, type assignments and the speech operator,

which results in grammars as shown in Figure 2. These are essentially logical statements consisting of a speech operator followed by a conjunction of literals, with the constraint of preventing reflexive statements such as ”I am a knight”.
This grammar can then be generated as shown in Algorithm 1. First, negations are randomly assigned to the type assignments (literals). Then the algorithm checks the type of the current speaker, as if the current speaker is a knave, at least one literal must be a lie (see Lines 9-17), but otherwise all statements must be true.
Once the truth value for each literal has been determined, then the literal can be generated as seen in Algorithm 2. Then, if previously negation was assigned to this literal, then it is applied to the truth value. The truth value determines which type should be selected. If the truth value is true, a matching type to the solution is selected. Otherwise, a randomly selected non-matching type is selected. Finally, this function returns both a string matching the selected things as well as a corresponding syntax tree, which can be used to easily traverse the generated literal.
This output is then used to create a string and a syntactic tree for the whole logical statement, which is the output of Algorithm 1.
Then the logical statements from each person are collected to create the overall puzzle. Example 2 shows an example of the logical representation of a puzzle that can be generated, though as stated in the caption the labels for the people were replaced with characters to make it easier to differentiate them from the types. Further, the solution for the generated puzzle is supplied, though at this stage it is not clear if the solution is the only one. In addition to the information shown in the example other information is also included, such as the number of people in the puzzle and the number of people per statement. This generated puzzle is then passed to the solver to determine if the generated puzzle has a unique solution. If that is not the case, then a new puzzle is generated until a puzzle with a unique solution is found.

3.2 Solver

The solver is designed to be both flexible as well as reasonably efficient. To achieve this there are two main components, one which translate from a logic string to a Python function and the second which uses these Python functions to determine all solutions. This second component is based on a Kripke world framework, where each world corresponds to a possible complete assignment of types to people, which allows for the easy evaluation of all possibilities and allows for easy integration of epistemological logic.

3.2.1 Translation

The translation from logical sentences to Python functions is important as it ensures that the solving is more efficient, as the same logical sentences are tested on multiple worlds, which would lead to unnecessary overhead if every test required direct handling of the logical sentences. To translate from logical sentences to Python functions the PLY [8] library is used. This library allows for the creation of custom tokenizers and parsers.
The tokenizer is used to tokenize the logical statement string into separate terminal tokens, such as numbers and operators. To reduce the complexity of the tokenizer and make the logical sentences more easily typeable, all operators where chosen as single ASCII characters (see Figure 3b).
The parser is a Lookahead left-right(LALR) parser which can be defined using the rules seen in Figure 3b. On the left hand side of the colon are the name of the non-terminal tokens and on the

[8]https://pypi.org/project/ply/

Algorithm 1: Generate Logical Statement

```
Data: isKnight which determines if the speaker tells the truth
allowedPeople determines which people the speaker talks about
solution is the type assignment to everyone
negationChance determines how likely it is that a type assignment is negated
Result: The logical statement (resultString) in text and node form (andNode).
resultString ← empty string;
node ← null;
prevNode ← null;
currTruthVal ← isKnight;
falseAtoms ← 0;
numNegations ← 0;
negationArray ← a random array filled with true and false with the same length as solution;
for loopIndex, person in allowedPeople do
    if not isKnight then
        currTruthVal ← random value between true and false using a uniform distribution;
        if not currTruthVal then
            falseAtoms ← falseAtoms + 1
        end
        if currTruthVal and loopIndex + 1 = the length of allowedPeople and
          falseAtoms = 0 then
            currTruthVal ← random value between true and false using a uniform
              distribution;
        end
    end
    isNegationAllowed ← negationArray[loopIndex];
    atomStr, node, hasNegation ←
      GenerateLiteral(currTruthVal, person, solution, negationChance);
    numNegations ← numNegations + hasNegation;
    andNode ← null ;
    if prevNode is not null then
        andNode ← SyntaxNode containing an identifier for AND and has two child nodes,
          namely node and prevNode;
        resultString ← resultString + "&" ;
        prevNode ← andNode;
    else
        prevNode ← node;
    end
    resultString ← resultString + atomString ;
end
```

Algorithm 2: Generate Literal

Data: *truthValue* which determines if the atom should be the truth
person is the selected person
solution is the type assignment to everyone
useNegation indicates if negation should be used
negationChance determines how likely it is that a type assignment is negated
Result: The logical type statement in text(*resultString*), node form (*node*) and if negation was used (*hasNegation*).

resultString $\leftarrow$ empty string;
node $\leftarrow$ *null*;
ancestor $\leftarrow$ *null*;
hasNegation $\leftarrow 0$;
if *useNegation* **then**
  hasNegation $\leftarrow 1$;
  resultString $\leftarrow$ " $-$ ";
  truthValue $\leftarrow$ *not truthValue*;
  ancestor $\leftarrow$ SyntaxNode containing an identifier for NOT;
end
chosenType $\leftarrow$ *solution*[*person*];
if *not truthValue* **then**
  chosenType $\leftarrow$ a random type that is not the type of the person given the solution;
end
resultString $\leftarrow$ *resultString* + "*T*" + person + , + chosenType;
node $\leftarrow$ SyntaxNode containing an identifier for Type assignment and has the person and chosenType as data fields;
if *ancestor is not null* **then**
  ancestor add *node* as a child node *node* $\leftarrow$ *ancestor*
end

right hand side is what the non-terminal tokens can be replaced with. The pipe (|) denotes different possible replacements. The non-terminal ”start” token is the non-terminal token from which all valid phrases can be generated, though implementation-wise this type of parser is implemented bottom up, which means that tokens are combined to non-terminal tokens, and if the input is in the grammar then only the non-terminal ”start” token should be left.
As can be seen in Figure 3 the accepted grammar is a logical sentence representing an utterance of a person. The speech non-terminal token can be replaced by the speech token, as well as an ID, which is a positive natural number and finally a non-terminal token representing the logical sentence. The ID is a numeric identifier for which person is speaking the logical sentence.
The sentence non-terminal token can be replaced by the standard logical operators in combination with further sentence tokens. As can be seen in the figure, there are two rules regarding negation. This is done to optimize the translation to Python code, as double negation can be disregarded. Furthermore, the knowledge operator (K) can be used to incorporate epistemological logic into the sentences, as before the ID is used to identify the agent that knows the logical statement.
The final component in the grammar is the type assignment rule, which functions as a logical atom. In this grammar, it represents the statements of the form: ”Agent x has type y”, where x is the identifier of an agent and y is a valid agent type. These are represented in the grammar by the ID token, where

```
Sa -Tb,1&-Tc,1
Sb -Ta,0&Tc,1
Sc Ta,0&-Tb,0
Solution: [a: 1, b: 0, c: 0]
```

Example 2: An example of the generated logic representation of a knights and knaves puzzle. In this example the zero corresponds to a knave (liar), while the 1 corresponds to a knight (truth-teller). Further, for clarity the labels for the individual people were changed from numbers (0,1,2) to characters (a,b,c).

$$
\begin{aligned}
start &: speech\\
speech &: S_{ID}\, sentence\\
sentence &: sentence \wedge sentence\\
&\mid sentence \vee sentence\\
&\mid sentence \rightarrow sentence\\
&\mid sentence \leftrightarrow sentence\\
&\mid \neg\neg sentence\\
&\mid \neg sentence\\
&\mid (sentence)\\
&\mid K_{ID}\, sentence\\
&\mid speech\\
&\mid T_{ID}(ID);
\end{aligned}
$$

(a) Standard logic notation.

$$
\begin{aligned}
start &: speech\\
speech &: S_{ID}\, sentence\\
sentence &: sentence\,\&\,sentence\\
&\mid sentence \mid sentence\\
&\mid sentence > sentence\\
&\mid sentence = sentence\\
&\mid --sentence\\
&\mid -sentence\\
&\mid (sentence)\\
&\mid K_{ID}\, sentence\\
&\mid speech\\
&\mid T_{ID,ID};
\end{aligned}
$$

(b) Custom notation used in the project.

Figure 3: A more extensive grammar that the parser can parse compared to the logic sentences in the generator in standard logic notation and in the notation used in the project.

the first token corresponds to x and the second one to y.
This concludes the input part of the translation. The output of the translation, as previously stated, is a Python function generated based on the input, such that it is functionally equivalent to the original logic statement. This function is automatically generated and is used to evaluate whether a specific world (a configuration of type assignments) satisfies the logical statement.
The generated Python function takes several arguments: the current world, the neighbourhood mapping, a list of possible types, and a type behaviour mapping.
A world is an assignment configuration of types (for example, Knights or Knaves) to all people. The neighbouring mapping determines given the current world and an agent, which worlds are indistinguishable from the current one from the agent's perspective, which allows for the modelling of epistemic uncertainty. The behaviour mapping is defined per agent type and maps from every possible type to a function which takes a boolean argument and returns a boolean output. This function allows for the specification of how every type should be treated logically. For example, the knave, who always lies, will have a function which returns the negation of the input. The knight has a function which returns the input, as they are always telling the truth and therefore the said logical statement

does not need to be changed.
The final Python function will return a boolean indicating whether the current assignment of people to types satisfies the logic sentence encoded in the function.
To generate this Python function, a string containing a Python function is created, which is then executed to add the function to the Python program. For this, we create the appropriate Python string fragment for each rule. For most operators, such as conjunction and disjunction this is a straightforward process, as Python supports equivalent logical operators. In the case of the implication operator ($\rightarrow$) it can be easily converted into an equivalent logical statement using only basic logic operators: $A \rightarrow B$ into $\neg A \vee B$.
There are some operators that require more consideration: The knowledge operator must evaluate the logical sentence across all possible neighbours of the current world, given the agent specified with the knowledge operator.
The speech operator uses the function from type behaviour mapping, given the speaker, to determine how to evaluate the logical statement.
The type assignment is a simple check on whether the specified person has the specified type in the current world.
Lastly, contrary to its name the start token is the last rule that is used as the parser works from the bottom up. This means that it can be used to generate the appropriate function definition and then dynamically add the function to the Python program.

3.2.2 Solving

With the translation done, the generated Python functions can be used to find the solution/-s to the Knights and Knaves puzzle. For this purpose, the world framework is used. This means that initially, all worlds will be generated given the number of people and a list of all types, as well as all the neighbourhood connections. In the current implementation, the solver will only generate reflexive connections, as in this setup, it is assumed that every person (except the reader) knows the types of all people.
Once the set of worlds is created, the previously generated Python functions are applied sequentially. The functions will produce a boolean for each world, indicating which worlds are viable. After each function is applied all worlds which were not viable are eliminated, and the next function is applied to the remaining worlds, until all functions have been applied. The remaining worlds represent all valid solutions to the puzzle.

3.3 Translation

Once the logical generation of the puzzle is finished and a puzzle with a unique solution was found, then the logical part can be used to generate the plain English puzzle description. For this the generated syntax tree can be used to generate the English description.
To create some variation in the puzzle in the translation there are four main levers: firstly, changing the names of the people, secondly, changing the labels of the types, thirdly, the location of the puzzle and lastly, adding optional sentence connectives.
For the first one, a dataset containing the 1000 most popular baby names in the US between 1880 and 2008 was used, which was already collected by Hadley Wickham in a Github repository[9]. From this dataset, all unique names were extracted which resulted in a name pool with 6872 unique names from which random names can be chosen.

[9] https://github.com/hadley/data-baby-names/tree/master

For the type variation, job descriptions were used to randomly replace the labels of knights and knaves out of a pool of 45 job descriptions. For the location variation the location was randomly selected from a pool of 10 locations.
Lastly, as an optional way to possibly increase variation in the English problem description, sentence connectives such as “Furthermore”, “Additionally”, “Moreover”, “Also” and “In addition” can be added to the individual utterances of the people. This might also help some LLMs to consider the whole sentence as a singular logical statement and therefore be able to apply for example negation correctly.
The English puzzle description is made up of three parts: first, the general description of the puzzle, the utterances of the people and finally, the question and possible prompting for specific output formats. This can be seen in Example 3, which shows what such a generated Puzzle looks like.
The general description is fairly similar, with only the names of the people, labels of the subjects and location being changed.
The utterances are the main variable part of the English text as they are translated from the previously generated logical statement. In this translation the literals that are connected by conjunctions are translated as separate English sentences, which means that every utterance is composed of multiple sentences. Each literal is fairly simple to translate as it is either a simple sentence, such as ”John is a knight”, or at most a simple sentence with negation, such as ”John is not a knight”.
Finally, the last part of the puzzle description is comprised of the task, namely, asking the reader to identify the subject types of the people described in the puzzle. As well as prescribing the LLM to format their output in a specific way to make automatic solution detection easier.

There is a country in which two types of people live, chefs and hairdressers. Chefs lie and hairdressers tell the truth. A logical statement is false if at least one side of a conjunction is false. Every utterance should be considered one logical statement. You encounter 3 inhabitants of the country, they are called Kelsi, Randall and Duff.
Kelsi says: ’Duff is not a hairdresser and moreover, Randall is not a hairdresser.’
Randall says: ’Duff is a hairdresser and additionally, Kelsi is not a chef.’
Duff says: ’Randall is not a chef and also, Kelsi is a chef.’
What are the types of the different people?
Please end your answer in the format:
name : type;

Example 3: An example of a generated Knights and Knaves Puzzle, where the green part is the general description, the blue part are the utterance of the people and the purple part is the question as well as the answer formating instructions. This example is based on the logical representation shown in Example 2.

3.4 Parser

In the previous sections, all the components needed to create a benchmark were explained. The only missing part to use the benchmark is a parser which can automatically determine whether the answer

given by an LLM is correct. This can be done by leveraging the requested answer format. However, even if this formatting is given, the output of LLMs does not always conform to the correct formatting, so the parser needs to be a bit more complex to account for that.
The parser accounts for this deviation from the requested answer format by allowing for the shortening of type descriptions to single words or characters, as long as the assigned types remain distinguishable. Example 4 shows a case in which Gemini 1.5 Pro shortens the types to single characters. Further, the parser also allows shortening of the names to the first character, provided that differentiation is still possible. Further, some deviations of separator symbols are accounted for, and some non-essential characters are ignored. Example 5 shows an answer with unnecessary characters and without line breaks after each assignment.

```
[...]
Celestine : F;
Harl : A;
Shantel : A;
Philip : A;
```

Example 4: An example of a part of an answer of Gemini 1.5 Pro, in which the final answer format is not completely correct, as the types were reduced to single characters.

```
[...]
$boxed\{Abagail : artist; Hosteen : artist; Rhonda : photographer;}
```

Example 5: An example of a part of an answer of Gemini 1.5 Pro, in which the final answer format is not completely correct, as the assignments are all in the same line and surrounded by unnecessary characters.

Once the answer is extracted, then the number of correct assignments is counted, if all assignment of people to types where correct then this puzzle is marked as correctly answered, otherwise it is considered incorrectly answered. Both of these values are stored alongside the answer of the model for further analysis.

4 Experimental Setup

To test the effectiveness of the new benchmark three datasets are generated and tested on the LLMs Google's Gemini 1.5 Pro (gemini-1.5-pro-002), OpenAI o1-mini (o1-mini-2024-09-12) and OpenAI o4-mini (o4-mini-2025-04-16). For all the models, a temperature of 1 was used and for the models from OpenAI a medium reasoning effort.
All datasets include puzzles of different complexities, determined by the number of people in the puzzle and per statement.
The first two datasets have identical logic representations and only the agent type labels are different with one having random labels and the other the original knight and knave labels. This dataset includes puzzles with three up to and including five people in the puzzle. For each number of people in the puzzle, the number of people was varied from two up to (not including) the number of people

in the puzzle. This results in six combinations of people in a puzzle and people per statement. For each of these combinations, 100 unique puzzles were generated which means that each dataset had 600 unique puzzles.
The third dataset has random agent type labels and has puzzles from six up to and including 17 people in the puzzle, thought 14 and 15 people are excluded due to budget constraints. The dataset includes all numbers people per statement between two and the number of people in the puzzle. This means that there are 87 combinations, each having 100 unique puzzles for a total of 8700 unique puzzles in the dataset.
Some initial prompt engineering was performed using Gemini 1.5 Pro to ensure that the puzzle description was as clear as possible, the exact variations tested are explained in Appendix B.
All three models are tested on the first dataset, only Gemini 1.5 Pro was tested against the second dataset with the original labels to test if there was potential data contamination with regards to knights and knaves puzzles. Lastly, the best model was then tested on the third dataset. For this model the results of the first and third dataset will be combined in the results to create more clarity.

5 Results

Figure 4 shows heatmaps displaying the accuracy of Gemini 1.5 Pro given the number of people in the puzzle and the number of people per statement for puzzles up to five people. Figure 4a the accuracy is achieved on puzzles with randomized agent labels and in Figure 4a the accuracy is achieved on puzzles with the original agent labels (knight and knave).
In Figure 4a, it can be seen that Gemini 1.5 Pro performs well for the least complex configuration with an accuracy of 70%. However, there is a steep decline as the number of people increases from three to four, to at best an accuracy of 48%. This steep decline in accuracy cannot be seen when going from four to five people in a puzzle as the highest accuracy for five people in a puzzle is 43%. For the number of people per statement, there is no clear trend, as for four people in the puzzles, fewer people per statement lead to a slightly higher accuracy and for five people in a puzzle, the highest accuracy can be observed with the middle amount of people per statement.
In contrast, Figure 4b shows the performance of Gemini 1.5 Pro on the same dataset, but with the type labels being knights and knaves. In this figure, it can be seen that the accuracy for the simplest configuration is slightly lower, with 66%. Though there is a drastic improvement in accuracy to 62% for the configuration with four people in the puzzle and two people per statement, though the other configuration with four people in the Figure performs the same way as with randomised labels. Further, for five people the model performs worse than with randomised labels with only a maximum accuracy of 32%.
Figure 5 shows a bar graph showing the accuracy already shown in Figure 4a, but it also show the baseline guessing accuracy as a red line. In this Figure, it can be seen that the accuracy of Gemini 1.5 Pro is above baseline for all combinations.
Figure 6 shows a heatmap of the accuracy of the model o1-mini given the number of people in the puzzle and the number of people per statement for puzzles up to five people. In the Figure, it can be seen that the reasoning model o1-mini performs near perfectly for the puzzles with fewer people in them, with an accuracycy of 98%. This accuracy decreases slightly as more people are added to the puzzle, reaching the lowest accuracy for five people of 90%.
Figure 7 shows a heatmap of the accuracy of the model o4.mini given the number of people in the puzzle and the number of people per statement for puzzles with up to 17 people. The model has an accuracy of 99% initially, but as the number of people increases the accuracy trends downwards.

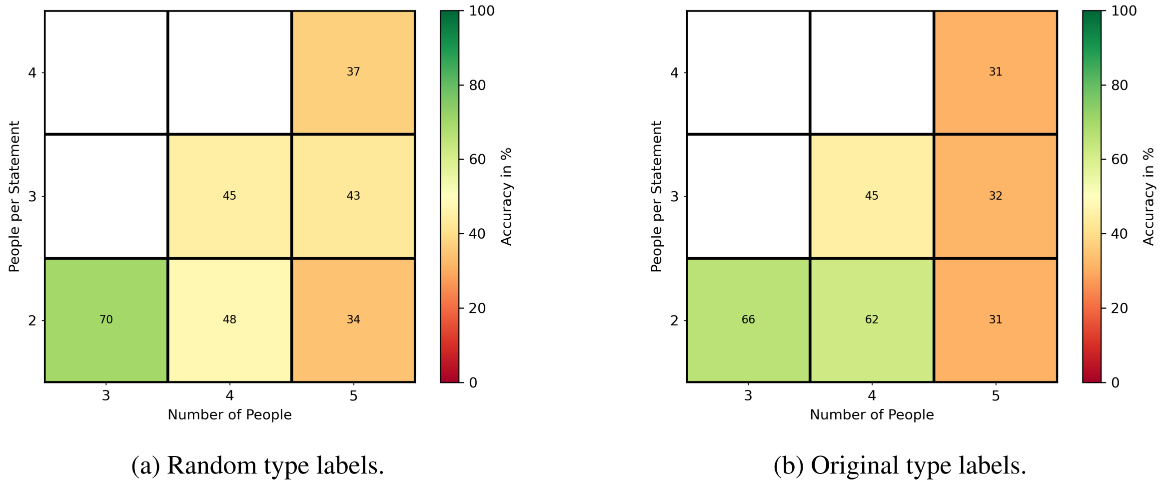

Figure 4: This Figure shows the accuracy of Gemini 1.5 Pro given the number of people in the puzzle and the number of people per statement for puzzles. Figure a) shows the accuracy from with random type labels, and Figure b) shows the accuracy with the original type labels.

Despite this, puzzles with the maximum or minimum amount of people in a statement did not drop below 80%, while the lowest accuracy is achieved by the combination of seventeen people in the puzzle and twelve people per statement, with an accuracy of 60%.
Figure 8 shows a heatmap of the mean number of reasoning tokens used by o4-mini to answer the puzzles given the number of people in the puzzle and the number of people per statement. This shows that as the number of people in the puzzle increases, the number of reasoning tokens increases. Also, in general as the number of people in each statement increases, the number of reasoning tokens increase as well, though there are exceptions, which can be seen for example for the puzzles with 17 people, in which the second-highest reasoning token amount can be seen for twelve people per statement.
In Figure 9 a bar graph showing the mean accuracy of Gemini 1.5 Pro, o1-mini and o4-mini for the first dataset can be seen. In this Figure it can be seen that Gemini 1.5 Pro has a mean accuracy of 46%, o1-mini has a mean accuracy of 93% and o4-mini has a mean accuracy of 97%. Further, Gemini 1.5 Pro shows high variability while o1-mini and o4-mini have a smaller deviation.

5.1 Qualitative Results

On top of looking at the quantative results it is useful to look at the actual responses for this I have randomly looked at two correctly solved puzzles and two randomly selected incorrectly solved puzzles per model. For the sake of brevity I will only show two examples in this section and I have included the other examples in Appendix C. Especially all examples of o1-mini have been left out as they behave similar to the example from o4-mini.
Example 6 shows the correct answer of the reasoning model o4-mini. In the example it can see that, because o4-mini has reasoning tokens which are not visible to the user. Because of this, the "reasoning" process of the model can not be assessed, but only the final result. This means that the reasoning process cannot be directly analyzed.
In Example 7, it can be seen that Gemini 1.5 Pro first correctly transforms the statements into representations without negations. Then the model assigns the first person an agent type and tests the

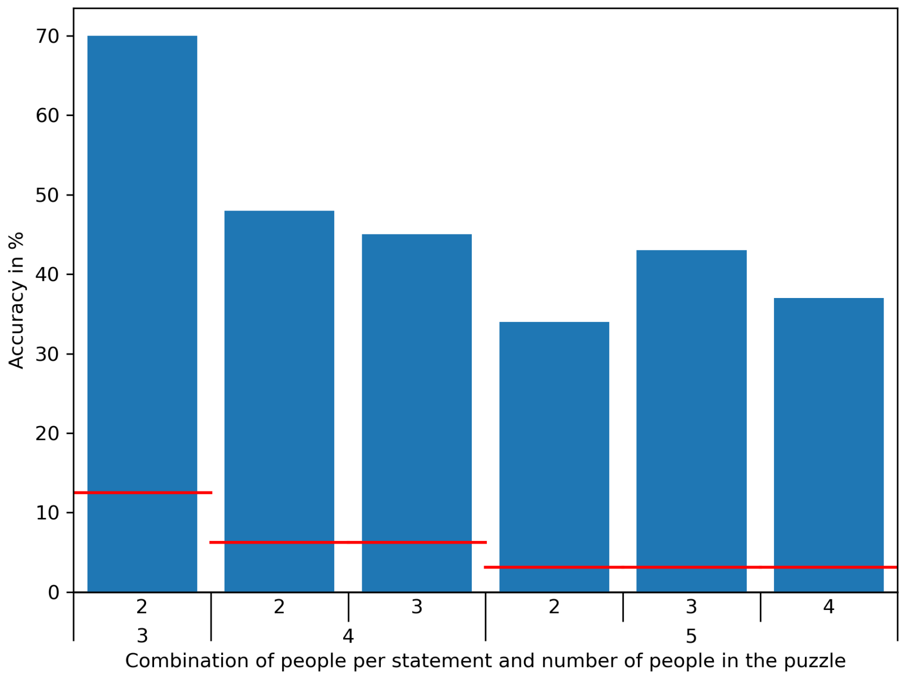

Figure 5: This Figure shows the accuracy of Gemini 1.5 Pro given the number of people in the puzzle and the number of people per statement for puzzles. The upper number on the x axis is the number of people per statement, while the lower number is the number of people in the puzzle. Further, the red line indicates the baseline if the model is guessing an answer.

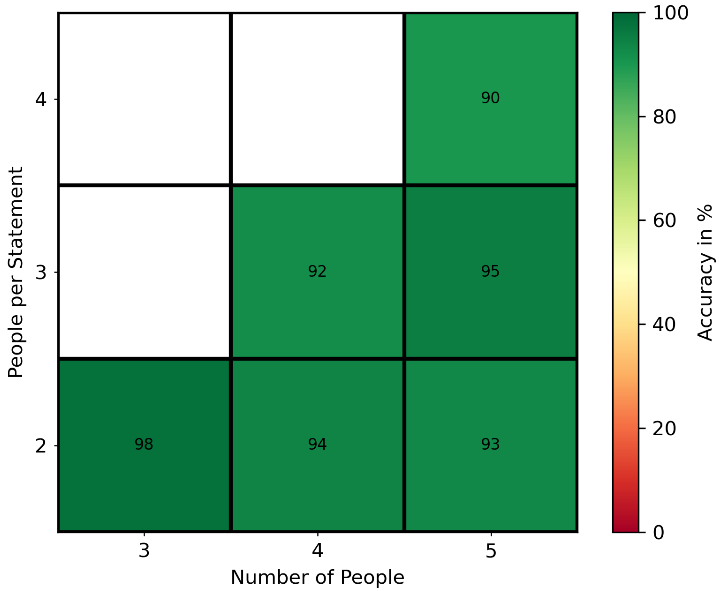

Figure 6: This Figure shows the accuracy of o1-mini given the number of people in the puzzle and the number of people per statement.

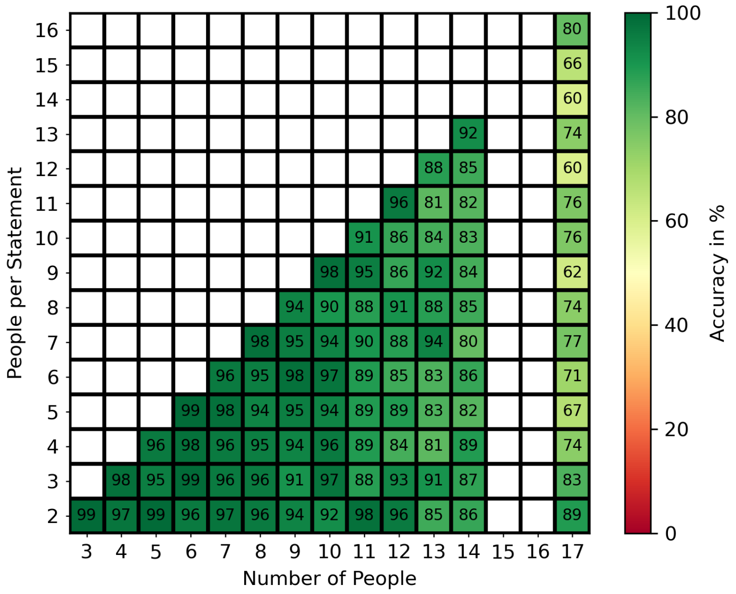

Figure 7: This Figure shows the accuracy of o4-mini given the number of people in the puzzle and the number of people per statement.

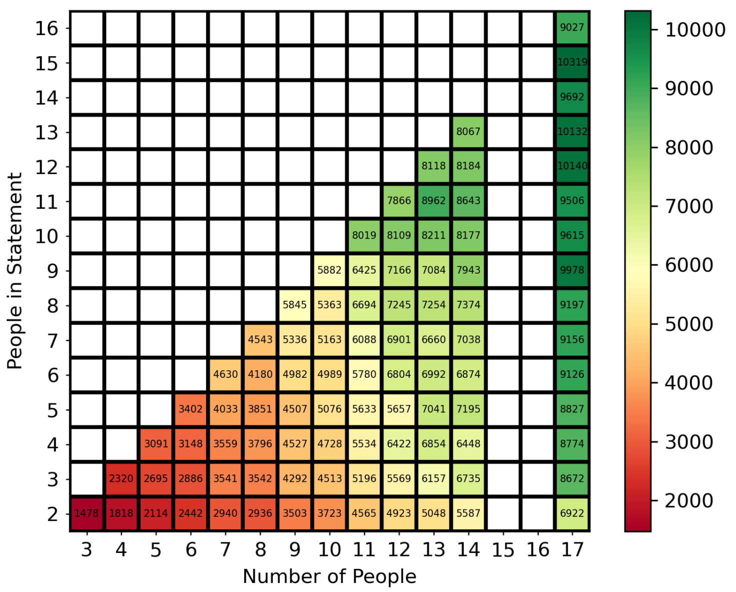

Figure 8: This Figure shows the mean reasoning tokens of o4-mini given the number of people in the puzzle and the number of people per statement.

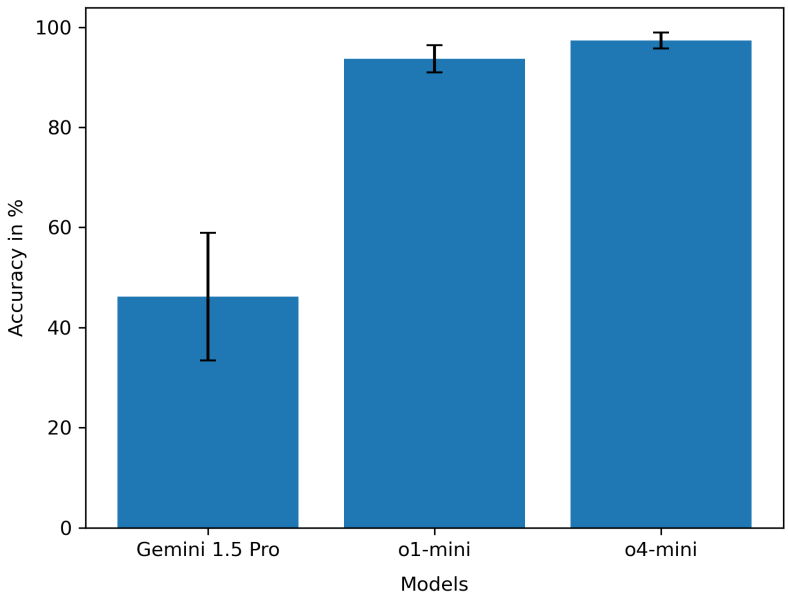

Figure 9: This Figure shows the mean accuracy of the models Gemini 1.5 Pro, o1-mini and o4-mini given the puzzles in the first dataset (600 unique puzzles). The black bars indicating the standard deviation. The accuracy is calculated per combination of people in the puzzle and people per statement, then the mean is calculated over the different combinations.

> Q: There is a country in which two types of people live, chefs and hairdressers. Chefs lie and hairdressers tell the truth. A logical statement is false if at least one side of a conjunction is false. Every utterance should be considered one logical statement. You encounter three inhabitants of the country, they are called Kelsi, Randall and Duff.
> Kelsi says: 'Duff is not a hairdresser and moreover, Randall is not a hairdresser.'
> Randall says: 'Duff is a hairdresser and additionally, Kelsi is not a chef.'
> Duff says: 'Randall is not a chef and also, Kelsi is a chef.'
> What are the types of the different people?
> Please end your answer in the format:
> name : type;
>
> A: Kelsi : hairdresser;
> Randall : chef;
> Duff : chef;

Example 6: An example of a question and correct response of o4-mini.

Q: There is a continent on which two types of people live, detectives and doctors. Detectives lie and doctors tell the truth. A logical statement is false if at least one side of a conjunction is false. Every utterance should be considered one logical statement. You encounter 3 inhabitants of the continent, they are called Ephram, Lynwood and Verlie.
Ephram says: 'Verlie is a detective and additionally, Lynwood is not a doctor.'
Lynwood says: 'Verlie is a doctor and moreover, Ephram is not a detective.'
Verlie says: 'Lynwood is a doctor and additionally, Ephram is a detective.'
What are the types of the different people?
Please end your answer in the format:
name : type;

A: Let's denote the statement ""'X is a detective"" by D(X) and ""'X is a doctor"" by C(X). Since there are only two types of people on the continent, if someone is not a detective, they must be a doctor, and vice-versa.

Ephram says: D(Verlie) and not C(Lynwood), which is equivalent to D(Verlie) and D(Lynwood). Lynwood says: C(Verlie) and not D(Ephram), which is equivalent to C(Verlie) and C(Ephram). Verlie says: C(Lynwood) and D(Ephram).

Let's consider the possibilities for Ephram.

Case 1: Ephram is a detective. This means he lies. So his statement ""'D(Verlie) and D(Lynwood)"" is false. This means either D(Verlie) is false or D(Lynwood) is false (or both).

Subcase 1a: D(Verlie) is false, so Verlie is a doctor. Then Verlie tells the truth, so C(Lynwood) and D(Ephram) are true. This matches our assumption that Ephram is a detective. Since Verlie is telling the truth, Lynwood is a doctor. **So we have: Ephram is a detective, Lynwood is a doctor, and Verlie is a doctor. This fits with Lynwood's statement, which would be true.**

Subcase 1b: D(Lynwood) is false, so Lynwood is a doctor. Then Lynwood tells the truth. Thus C(Verlie) and C(Ephram) are true. But we assumed Ephram is a detective, which contradicts C(Ephram). This subcase leads to a contradiction.

Case 2: Ephram is a doctor. Then he tells the truth. So D(Verlie) and D(Lynwood) are true. Verlie is a detective, so they lie. Therefore, at least one of C(Lynwood) or D(Ephram) must be false. Since we assume Ephram is a doctor, D(Ephram) is false. So this means C(Lynwood) might be true or false. However, since D(Lynwood) is true (from Ephram's statement), Lynwood is a detective. Since Lynwood is a detective, Lynwood lies. Thus, the statement C(Verlie) and C(Ephram) must be false. **This is consistent with the fact that D(Verlie) is true, meaning C(Verlie) is false. But, we assume C(Ephram) is true. This creates a contradiction.**

Therefore, only Case 1a is valid.

Ephram : detective;
Lynwood : doctor;
Verlie : doctor;

Example 7: An example of a question and incorrect response of Gemini 1.5 Pro.

consequence of that. During that, it makes some mistakes, which are highlighted in the example. In the first case, it neglects to properly check if all statements are fulfilled and just concludes that this is the case. In the other highlighted case, the reasoning is not clear as the first sentence is correct, but then the following sentence is not related to the previous sentence and incorrect, which leads the model to provide the wrong answer.

6 Discussion

Figure 9 shows that the reasoning models o1-mini and o4-mini are significantly outperforming the non-reasoning model Gemini 1.5 Pro.
As shown in Figures 6 and 7, OpenAI's reasoning models perform consistently well on the benchmark, with accuracies exceeding 90% for all configurations up to and including five people. This shows that there is only a gradual decline in performance as the complexity of the puzzle increases. Further in Figure 7, it can be seen that this gradual decline in performance for o4-mini is continued as the complexity of the model increases.
In contrast, Gemini 1.5 Pro exhibits a sharp decline in performance as complexity rises, as illustrated in Figure 4a. While performance is relatively strong on the simplest puzzles (70% accuracy), it drops to as low as 34% as the number of people, and therefore the complexity, increases. This sharp decline indicates that Gemini 1.5 Pro's reasoning capabilities do not scale well when faced with more complex puzzles.
One possible explanation for this performance drop can be found in Example 7, in which it can be seen that the model does not check all relevant conditions or makes unexplained invalid reasoning steps. If it is assumed that there is a baseline chance that the model makes this kind of mistake, then as the complexity rises, the number of reasoning steps required rises, which increases the probability that such a mistake occurs.
In Figure 4b, it can be seen that even with the original type labels being restored, there is no overall performance uplift. This might suggest that Gemini 1.5 Pro did not have a lot of Knights and knaves puzzles in its training sample, and therefore that the original labels could not give the model additional context to better solve the puzzle.
Further, in Figure 5 it can be seen that Gemini 1.5 Pro performs above baseline for all tested complexities. As this model was the worst performing one this means that all of the other models also perform above baseline, which suggests that the models are not just guessing the answer.
For o1-mini and Gemini 1.5 Pro, the models perform better with three people per statement compared to fewer or more people per statement. For o4-mini, this does not hold, as in general, the most or least amount of people lead to better performance. A possible explanation for both could be that both very long and very short statements might be beneficial, depending on the type of speaker. In the case that the speaker is a truth-teller, then fewer people per statement are beneficial, as this reduces the number of people that are need to be checked to determine the type of the speaker. On the other hand, if the speaker is a liar, then it might be beneficial if the statement has more people, as this increases the likelihood that more speakers are mentioning the speaker and therefore can be identified as liars once the original speaker is identified. Also, on average a liar is faster to identify, as type assignments up to the first lie need to be checked. So longer statements will have less of an effect on the number of type assignment checked, compared to having to check all type assignments in a statement to determine a speaker is a knight. Given this explanation both contradictory could make sense as for o1-mini and Gemini 1.5 Pro the three different numbers of people per statement for five people might not lead to a sufficient distance from the extremes to create a noticeable difference, while for o4-mini, more combinations were tested and therefore such an effect might be possible.
While this explanation might account for the general trend of the results, it does not explain the fluctuations seen in o4-mini, where some combinations which are closer to the extremes produce worse results compared to more centered combinations. An example of this are ten people in a statement compared to 14 people in a statement.
Overall, the results indicate that puzzle complexity affects model performance, though to different degrees depending on the model. There is a clear trend that as the number of people in the puzzle

	Problem	Evaluation	Basis	Found Limitation	Complexity	Problem based on natural language
Apple's paper	Four different problems which involve moving from an initial position to a goal position	Intermediate Results and Final Result	Based on simulator	Yes	Varied complexity from linear to exponential	No
Steging's paper	Determine if the first person told the truth based on the statements of others	Final Result	Based on argumentation graphs	Yes	Polynomial	Yes
This Paper	Knights and Knaves Puzzle	Final Result	Based on a logical framework	No	NP-complete	Yes

Table 1: The different papers' approaches to examine the logical reasoning capabilities of LLMs. In this table, the Apple paper refers to "The illusion of thinking"(Shojaee et al., 2025) and the Steging's paper to "Parameterized argumentation-based reasoning tasks for benchmarking generative language models" (Steging et al., 2025).

increases the performance of the models decreases. For the number of people in each statement, it can be seen that for o4-mini the extremes perform better, while for the other two, the middle performs better, though for these models, fewer combinations were tested.

Looking at similar papers, Steging et al. (2025) and the non-peer-reviewed paper from Apple by Shojaee et al. (2025) both created dynamic benchmarks to test the reasoning capabilities of LLMs. Although their methods differ slightly, as seen in Table 1, both papers reveal limitations in the reasoning capabilities of LLMs. In contrast, this paper shows the best model performing well with only a slight drop in performance.

Regarding the Apple paper, they observed a collapse in the amount of reasoning tokens used by the models as the models grew past some threshold of complexity. In comparison, as seen in Figure 8, this does not occur in this paper. This might suggest that the complexity of this benchmark was not high enough, especially as the number of reasoning tokens is still below some of the breaking points of some of the models tested in the Apple paper. Though checking the model with more complex puzzles allows for the better analysis on the robustness of the models capabilities even if no hard limit was found. A potential avenue of research could be the further scaling of the benchmark to test if a similar breaking point exists, or if the observed behaviour continues.

Another key difference between the Apple paper and this paper is the kind of puzzle that is used. Their puzzles like for example Tower of Hanoi, are mainly about representation manipulation to get to the solution, while the Knights and Knaves puzzle has a strong linguistic component in it. Given the nature of large language models, there might be a bias towards more language oriented puzzles.

This leads to the results by Steging et al. (2025). They showed that most models could not reliably solve linguistic puzzles based on linear argumentation graphs, and while the tested reasoning model, o1-preview, could solve them perfectly, it could not solve the puzzles based on non-linear argumentation graphs perfectly anymore. While these results showed some clear limitations of the LLMs, most of the tested models are not reasoning models, so comparing the results from that paper with the non-reasoning model Gemini 1.5 Pro, it can be seen that the model is not scaling well with increased complexity. For the reasoning model, o1-preview, the authors do not examine how the performance scales with increasing complexity, so it is hard to compare them to the results of this paper. Though both papers show that even reasoning models are not capable of solving the puzzles perfectly. It seems contradictory that their paper showed clear limitations of LLMs, while this paper seemingly did not, especially as both use linguistic puzzles and their puzzle complexity scales more slowly, this might be explained by the different model selection.
Another difference between this paper and the Apple paper by Shojaee et al. (2025) is that this paper only looks at the final result, while their paper also examines the intermediate results. This might be a weak point in this benchmark as while it is reasonable to assume that good reasoning produces good answers, the reverse is not necessarily true, a correct answer does not guarantee that the reasoning behind it was logically sound. This issue is especially pronounced in OpenAI's reasoning models, where the internal reasoning tokens are not accessible to the user, which makes it impossible to determine the reasoning steps. A possibility to address this could be the use of other reasoning models, such as Deepseek, with accessible reasoning tokens, which would allow for a qualitative analysis of the reasoning steps. Though due to time constraints the experiment could not be run on the model as the reply from the model for 600 puzzles would likely taken a week or longer, based on some initial small scale tests.
Finally, further research could investigate the effect observed regarding the number of people per statement, to investigate whether the agent type of the people influences the complexity of the puzzle in regard to the number of people in each statement. In particular, testing puzzles with solutions consisting only of knights or only of knaves could clarify whether the predicted effect holds.

6.1 Conclusion

In summary, some LLMs, particularly OpenAI's reasoning models, appear capable of handling structured logical reasoning tasks, and do so robustly across increasing levels of complexity, as demonstrated by their consistently high accuracy though performance does drop with increased complexity. However, this dynamic benchmark demonstrates that even reasoning models are incapable of solving these problems perfectly, which shows that there is value in this benchmark with scaling complexity, as it allows for an analysis of the reasoning capabilities as problems become more complex and it allows the benchmark to evaluate future more capable LLMs. However, due to the limitations of current model transparency, it remains unclear whether these models are following coherent, rule-based reasoning processes, or simply generating correct answers through other means, such as tool use. Clarifying the reasoning steps is still an important goal that should be investigated in future research.

References

Arkoudas, K. (2023, August). GPT-4 Can't Reason. (arXiv:2308.03762). Retrieved from http://arxiv.org/abs/2308.03762 (arXiv:2308.03762) doi: 10.48550/arXiv.2308.03762

Boghossian, P. (2014, May). What is inference? *Philosophical Studies*, *169*(1), 1–18. (Company: Springer Distributor: Springer Institution: Springer Label: Springer number: 1 publisher: Springer Netherlands) doi: 10.1007/s11098-012-9903-x

DeepSeek-AI, Guo, D., Yang, D., Zhang, H., Song, J., Zhang, R., ... Zhang, Z. (2025, January). DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. (arXiv:2501.12948). Retrieved from http://arxiv.org/abs/2501.12948 (arXiv:2501.12948 [cs]) doi: 10.48550/arXiv.2501.12948

Delgrande, J. P. (1988, August). An approach to default reasoning based on a first-order conditional logic: Revised report. *Artificial Intelligence*, *36*(1), 63–90. doi: 10.1016/0004-3702(88)90079-3

Karp, R. M. (1972). Reducibility among Combinatorial Problems. In *Complexity of computer computations* (p. 85–103). Springer, Boston, MA. Retrieved from https://link.springer.com/chapter/10.1007/978-1-4684-2001-2_9 doi: 10.1007/978-1-4684-2001-2_9

Liu, F., & Wang, Y. (2013, March). Reasoning About Agent Types and the Hardest Logic Puzzle Ever. *Minds and Machines*, *23*(1), 123–161. doi: 10.1007/s11023-012-9287-x

Liu, H., Ning, R., Teng, Z., Liu, J., Zhou, Q., & Zhang, Y. (2023, May). Evaluating the Logical Reasoning Ability of ChatGPT and GPT-4. (arXiv:2304.03439). Retrieved from http://arxiv.org/abs/2304.03439 (arXiv:2304.03439) doi: 10.48550/arXiv.2304.03439

Luo, M., Kumbhar, S., Shen, M., Parmar, M., Varshney, N., Banerjee, P., ... Baral, C. (2024, March). Towards LogiGLUE: A Brief Survey and A Benchmark for Analyzing Logical Reasoning Capabilities of Language Models. (arXiv:2310.00836). Retrieved from http://arxiv.org/abs/2310.00836 (arXiv:2310.00836 [cs]) doi: 10.48550/arXiv.2310.00836

McHugh, C., & Way, J. (2018, January). What is Reasoning? *Mind*, *127*(505), 167–196. doi: 10.1093/mind/fzw068

Mirzadeh, I., Alizadeh, K., Shahrokhi, H., Tuzel, O., Bengio, S., & Farajtabar, M. (2024, October). GSM-Symbolic: Understanding the Limitations of Mathematical Reasoning in Large Language Models. (arXiv:2410.05229). Retrieved from http://arxiv.org/abs/2410.05229 (arXiv:2410.05229) doi: 10.48550/arXiv.2410.05229

Shojaee, P., Mirzadeh, I., Alizadeh, K., Horton, M., Bengio, S., & Farajtabar, M. (2025). *The Illusion of Thinking: Understanding the Strengths and Limitations of Reasoning Models via the Lens of Problem Complexity.* Retrieved from https://ml-site.cdn-apple.com/papers/the-illusion-of-thinking.pdf

Smullyan, R. M. (1978). *What is the name of this book: The riddle of dracula and other logical puzzles*. Prentice Hall, Hoboken.

Steging, C., Renooij, S., & Verheij, B. (2025, May). Parameterized Argumentation-based Reasoning Tasks for Benchmarking Generative Language Models.. Retrieved from https://arxiv.org/abs/2505.01539v1 van Ditmarsch, H. (2014, March). Dynamics of lying. *Synthese*, *191*(5), 745–777. (Company: Springer Distributor: Springer Institution: Springer Label: Springer number: 5 publisher: Springer Netherlands) doi: 10.1007/s11229-013-0275-3

van Ditmarsch, H., van Eijck, J., Sietsma, F., & Wang, Y. (2012). On the Logic of Lying. In *Games, actions and social software* (p. 41–72). Springer, Berlin, Heidelberg. Retrieved from https://link.springer.com/chapter/10.1007/978-3-642-29326-9_4 doi: 10.1007/978-3-642-29326-9_4

Wei, J., Tay, Y., Bommasani, R., Raffel, C., Zoph, B., Borgeaud, S., ... Fedus, W. (2022, October). Emergent Abilities of Large Language Models. (arXiv:2206.07682). Retrieved from http://arxiv.org/abs/2206.07682 (arXiv:2206.07682) doi: 10.48550/arXiv.2206.07682

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., ... Zhou, D. (2022, January). *Chain-of-Thought Prompting Elicits Reasoning in Large Language Models.* Retrieved from https://arxiv.org/abs/2201.11903v6

Xu, C., Guan, S., Greene, D., & Kechadi, M.-T. (2024, June). Benchmark Data Contamination of Large Language Models: A Survey. (arXiv:2406.04244). Retrieved from http://arxiv.org/abs/2406.04244 (arXiv:2406.04244 [cs]) doi: 10.48550/arXiv.2406.04244

Appendices

A Knights and Knaves computational complexity

To determine the computational complexity of Knights and Knaves puzzles two languages are defined: *Knights-SOL* and *Knights-USOL*, which can be seen in the Equations 1 and 2.

$$Knights\text{-}SOL = \{\langle\delta\rangle | \delta \texttt{ is a solvable knights and knaves puzzle}\} \tag{1}$$

$$Knights\text{-}USOL = \{\langle\delta\rangle | \delta \texttt{ is a uniquely solvable knights and knaves puzzle}\} \tag{2}$$

As can be seen from the definitions *Knights-USOL* is a subclass of *Knights-SOL* as it just introduces an additional constraint. Therefore we will first focus on the larger class of puzzles.
The language *Knights-SOL* is NP-complete. To prove this it needs to be shown that it is in the nondeterministic polynomial (NP) complexity class and that there is a polynomial reduction from another NP-complete language to this language.
To show that *Knights-SOL* is in NP, for that it needs to fit in the definition of NP. The NP complexity class is defined as the class of languages verifiable in polynomial time by a deterministic Turing machine. Verifiable means that, given a solution, it can be verified that the solution is correct. This is trivially done in this case, as given a solution, all type assignments and speech operators now have a clear meaning, and therefore it becomes a set of simple logical sentences that all need to be true. As problems such as SAT are in NP any logic sentence is trivially solvable in polynomial time.
The second property of *Knights-SOL* that needs to be proven is that there is a polynomial reduction from another NP-complete language to this language. For this the polynomial reduction from 3-SAT to *Knights-SOL* was chosen. This is because it has an easier structure compared to similar languages such as SAT, but is proven to be NP-complete (Karp, 1972). 3-SAT is restricted to logic formulas that are three boolean variables connected by disjunctions, and this set of three is connected to others by conjunctions, as seen in Equation 3.

$$(a \vee \neg b \vee c) \wedge (a \vee b \vee \neg c) \wedge (\neg a \vee b \vee \neg c) \tag{3}$$

To reduce from 3-SAT to *Knights-SOL* we want to create logical representations of knights and knaves puzzles that have at least one solution, if and only if the given 3-SAT formula is satisfiable. To achieve this the 3-SAT formula needs to be translated into multiple logical utterances that confirm to the grammar of knights and knaves puzzles. The assignment of values to boolean variables in 3-SAT will be replaced with the assignment of types to speakers in the knights and knaves puzzle.
The knights and knaves puzzles produced by the generator (see Section 3.1), do not contain disjunctions, so to translate from 3-SAT we need to convert the disjunctions to conjunctions. This can be done using the De Morgan's law, which says that $\neg(A \wedge B)$ is equivalent to $\neg A \vee \neg B$. To make use of the De Morgan's law means that we need to be able to negate conjunctions of literals, which can be done using the speech operator (S_{ID}) if the speaker is a knave (liar), as from a logical perspective a lie is the negation of the said statement.
To ensure that the speakers are knaves, we can make use of the Liar's paradox, which in the case of the knights and knaves puzzle is the sentence "I am a knave" ($S_x T x, 0$ where x is the identifier of the speaker). This sentence cannot be said by a knight (truth-teller) as it would be a lie, and it can only be said by a knave in conjunction with at least one other sentence that is a lie as the liar would otherwise say a truthful statement. This makes it ideal to ensure that the speaker can only be assigned the knave

type, while not adding a lie to the conjunction which would make the said statement trivially a lie regardless of the truth values of the other terms.

To reduce 3-SAT to *Knights-SOL*, first, all of the boolean variables are extracted and rewritten as speech sentences. This is done by writing this sentence: $S_xTx,1$, where x is the original variable name and is equivalent to the sentence: "I am a knight.". This sentence can be said by both knights and knaves. This makes the sentence ideal for representing atoms as they can be either true or false.

Then each set of disjunctions in the original input (3-SAT formula) can be translated into a utterances of a speaker. This is done by translating every boolean variable to a type assignment where the person in the type assignment corresponds to the speaker that was previously created for the boolean variable. The agent type identifier in the type assignment is set to a knave, as this corresponds to a negation, which is necessary as the De Morgan Law is used which requires the negation of each element. If the input has negation of an atom (for example $\neg a$) than this is kept in the new speech sentence (for example $\neg T_{a.0}$). Finally, the additional type assignment for the Liar's Paradox sentence is added to ensure that the speaker can only be a knave.

This reduction is polynomial as the expansion of the input is bound by a constant factor, as the translation of the individual parts is independent of the length of the input. This is because as seen above every character in the input has a simple translation and the atoms have an additional sentence, which is still bound by a constant factor. This means that the reduction is polynomial, which means that *Knights-SOL* is NP-complete.

The previous reduction was from 3-SAT to *Knights-SOL* with reflexive type assignments, though the actual grammar (see Section 3.1) does not allow for the reflexive assignments therefore we need to create a reduction from the grammar with reflexive types to one without. For this, we only need to show how to transform the two reflexive type assignments, as all others are the same.

The first one is the logical sentence "$T_{x,1}$" where x is the identifier of the speaker of the type assignment, which translates to "I am a knight", to remove the reflexive type assignment, another new speaker y who says $S_yT_{x,1}$ can be added and the logical statement "$T_{x,1}$" can be replaced with "$T_{y,1}$". This is equivalent, as both the initial and the changed statements are true regardless of which type the speaker x has.

The second logical sentence $T_{x,0}$ where x is the identifier of the speaker of the type assignment, which translates to "I am a knave". To remove the reflexive type assignment, another new speaker y who says $S_yT_{x,0}$ can be added and the logical statement "$T_{x,0}$" can be replaced with "$T_{y,1}$". Both the reflexive and non-reflexive versions are equivalent, as both lead to contradictions regardless of the type of x. This translation is polynomial, because at worst the original input consists of only reflexive type assignments, in which case the input would roughly double, which is still polynomial. This means that there is a polynomial reduction from the language with reflexive assignments to the one without, which means that the language without reflexive assignments is also NP-complete (same reason as before for why it is in NP).

For the *Knights-USOL* it does not necessarily hold that it is in NP, because the solution is not a sufficient certificate to ensure that it is a unique solution. This is because from a single solution it is impossible to tell if it is the only solution. Given that we can look at which other complexity class might be appropriate. Though this language might be in co-NP, as this only requires that there is a polynomial-length certificate that allows one to determine in polynomial time that an input belongs to the complement of the language. This can be shown fairly easily, as the invalid inputs are easily checked in polynomial time by running a parser on the input and then accepting all invalid inputs. For the valid inputs, a certificate containing two distinct solutions is sufficient to verify that the language does not have a unique solution. Therefore, *Knights-USOL* is in co-NP.

B Prompt Engineering

Some initial tests revealed that there might be some ambiguity in the original puzzle description, therefore, additional translation variations were created in order to determine if that has a significant effect on the performance. These variations are explained in detail in Table 2.

Version	Effect	Example
1.1	Baseline	There is a country in which two types of people live, chefs and hairdressers. Chefs lie and hairdressers tell the truth. You encounter 3 inhabitants of the country, they are called Kelsi, Randall and Duff. Kelsi says: 'Duff is not a hairdresser. Moreover, Randall is not a hairdresser.' Randall says: 'Duff is a hairdresser. Additionally, Kelsi is not a chef.' Duff says: 'Randall is not a chef. Also, Kelsi is a chef.' What are the types of the different people?
1.2	Add highlighted sentence. Adds “always” to type description.	There is a country in which two types of people live, chefs and hairdressers. Chefs always lie and hairdressers always tell the truth. **Someone lies as long as at least a single sentence is a lie.** You encounter 3 inhabitants of the country, they are called Kelsi, Randall and Duff. Kelsi says: 'Duff is not a hairdresser. Moreover, Randall is not a hairdresser.' Randall says: 'Duff is a hairdresser. Additionally, Kelsi is not a chef.' Duff says: 'Randall is not a chef. Also, Kelsi is a chef.' What are the types of the different people?
1.3	Add highlighted sentence.	There is a country in which two types of people live, chefs and hairdressers. Chefs lie and hairdressers tell the truth. **Someone lies as long as at least a single sentence is a lie.** You encounter 3 inhabitants of the country, they are called Kelsi, Randall and Duff. Kelsi says: 'Duff is not a hairdresser. Moreover, Randall is not a hairdresser.' Randall says: 'Duff is a hairdresser. Additionally, Kelsi is not a chef.' Duff says: 'Randall is not a chef. Also, Kelsi is a chef.' What are the types of the different people?

1.4	Add highlighted sentence. Adds “always” to type description.	There is a country in which two types of people live, chefs and hairdressers. Chefs always lie and hairdressers always tell the truth. **A logical statement is false if at least one side of a conjunction is false.** You encounter 3 inhabitants of the country, they are called Kelsi, Randall and Duff. Kelsi says: ’Duff is not a hairdresser. Moreover, Randall is not a hairdresser.’ Randall says: ’Duff is a hairdresser. Additionally, Kelsi is not a chef.’ Duff says: ’Randall is not a chef. Also, Kelsi is a chef.’ What are the types of the different people?
1.5	Add highlighted sentence.	There is a country in which two types of people live, chefs and hairdressers. Chefs lie and hairdressers tell the truth. **A logical statement is false if at least one side of a conjunction is false.** You encounter 3 inhabitants of the country, they are called Kelsi, Randall and Duff. Kelsi says: ’Duff is not a hairdresser. Moreover, Randall is not a hairdresser.’ Randall says: ’Duff is a hairdresser. Additionally, Kelsi is not a chef.’ Duff says: ’Randall is not a chef. Also, Kelsi is a chef.’ What are the types of the different people?
1.6	Add highlighted sentences. Adds “always” to type description.	There is a country in which two types of people live, chefs and hairdressers. Chefs always lie and hairdressers always tell the truth. **A logical statement is false if at least one side of a conjunction is false.** **Every utterance should be considered one logical statement.** You encounter 3 inhabitants of the country, they are called Kelsi, Randall and Duff. Kelsi says: ’Duff is not a hairdresser. Moreover, Randall is not a hairdresser.’ Randall says: ’Duff is a hairdresser. Additionally, Kelsi is not a chef.’ Duff says: ’Randall is not a chef. Also, Kelsi is a chef.’ What are the types of the different people?

1.7	Add highlighted sentences.	There is a country in which two types of people live, chefs and hairdressers. Chefs lie and hairdressers tell the truth. **A logical statement is false if at least** **one side of a conjunction is false.** **Every utterance should be considered** **one logical statement.** You encounter 3 inhabitants of the country, they are called Kelsi, Randall and Duff. Kelsi says: 'Duff is not a hairdresser. Moreover, Randall is not a hairdresser.' Randall says: 'Duff is a hairdresser. Additionally, Kelsi is not a chef.' Duff says: 'Randall is not a chef. Also, Kelsi is a chef.' What are the types of the different people?
2.1	Change utterances to single sentences.	There is a country in which two types of people live, chefs and hairdressers. Chefs lie and hairdressers tell the truth. You encounter 3 inhabitants of the country, they are called Kelsi, Randall and Duff. Kelsi says: 'Duff is not a hairdresser and moreover, Randall is not a hairdresser.' Randall says: 'Duff is a hairdresser and additionally, Kelsi is not a chef.' Duff says: 'Randall is not a chef and also, Kelsi is a chef.' What are the types of the different people?
2.2	Change utterances to single sentences. Add highlighted sentence. Adds "always" to type description.	There is a country in which two types of people live, chefs and hairdressers. Chefs always lie and hairdressers always tell the truth. **Someone lies as long as** **at least a single sentence is a lie.** You encounter 3 inhabitants of the country, they are called Kelsi, Randall and Duff. Kelsi says: 'Duff is not a hairdresser and moreover, Randall is not a hairdresser.' Randall says: 'Duff is a hairdresser and additionally, Kelsi is not a chef.' Duff says: 'Randall is not a chef and also, Kelsi is a chef.' What are the types of the different people?

2.3	Change utterances to single sentences. Add highlighted sentence.	There is a country in which two types of people live, chefs and hairdressers. Chefs lie and hairdressers tell the truth. **Someone lies as long as at least a single sentence is a lie.** You encounter 3 inhabitants of the country, they are called Kelsi, Randall and Duff. Kelsi says: ’Duff is not a hairdresser and moreover, Randall is not a hairdresser.’ Randall says: ’Duff is a hairdresser and additionally, Kelsi is not a chef.’ Duff says: ’Randall is not a chef and also, Kelsi is a chef.’ What are the types of the different people?
2.4	Change utterances to single sentences. Add highlighted sentence. Adds “always” to type description.	There is a country in which two types of people live, chefs and hairdressers. Chefs always lie and hairdressers always tell the truth. **A logical statement is false if at least one side of a conjunction is false.** You encounter 3 inhabitants of the country, they are called Kelsi, Randall and Duff. Kelsi says: ’Duff is not a hairdresser and moreover, Randall is not a hairdresser.’ Randall says: ’Duff is a hairdresser and additionally, Kelsi is not a chef.’ Duff says: ’Randall is not a chef and also, Kelsi is a chef.’ What are the types of the different people?
2.5	Change utterances to single sentences. Add highlighted sentence.	There is a country in which two types of people live, chefs and hairdressers. Chefs lie and hairdressers tell the truth. **A logical statement is false if at least one side of a conjunction is false.** You encounter 3 inhabitants of the country, they are called Kelsi, Randall and Duff. Kelsi says: ’Duff is not a hairdresser and moreover, Randall is not a hairdresser.’ Randall says: ’Duff is a hairdresser and additionally, Kelsi is not a chef.’ Duff says: ’Randall is not a chef and also, Kelsi is a chef.’ What are the types of the different people?

2.6	Change utterances to single sentences. Add highlighted sentences. Adds “always” to type description.	There is a country in which two types of people live, chefs and hairdressers. Chefs always lie and hairdressers always tell the truth. **A logical statement is false if at least one side of a conjunction is false.** **Every utterance should be considered one logical statement.** You encounter 3 inhabitants of the country, they are called Kelsi, Randall and Duff. Kelsi says: ’Duff is not a hairdresser and moreover, Randall is not a hairdresser.’ Randall says: ’Duff is a hairdresser and additionally, Kelsi is not a chef.’ Duff says: ’Randall is not a chef and also, Kelsi is a chef.’ What are the types of the different people?
2.7	Change utterances to single sentences. Add highlighted sentences.	There is a country in which two types of people live, chefs and hairdressers. Chefs lie and hairdressers tell the truth. **A logical statement is false if at least one side of a conjunction is false.** **Every utterance should be considered one logical statement.** You encounter 3 inhabitants of the country, they are called Kelsi, Randall and Duff. Kelsi says: ’Duff is not a hairdresser and moreover, Randall is not a hairdresser.’ Randall says: ’Duff is a hairdresser and additionally, Kelsi is not a chef.’ Duff says: ’Randall is not a chef and also, Kelsi is a chef.’ What are the types of the different people?
3.1	Change utterances to single sentences without sentence connectives. Add highlighted sentence.	There is a country in which two types of people live, chefs and hairdressers. Chefs lie and hairdressers tell the truth. You encounter 3 inhabitants of the country, they are called Kelsi, Randall and Duff. Kelsi says: ’Duff is not a hairdresser and Randall is not a hairdresser.’ Randall says: ’Duff is a hairdresser and Kelsi is not a chef.’ Duff says: ’Randall is not a chef and Kelsi is a chef.’ What are the types of the different people?

3.2	Change utterances to single sentences without sentence connectives. Add highlighted sentence. Adds “always” to type description.	There is a country in which two types of people live, chefs and hairdressers. Chefs always lie and hairdressers always tell the truth. **Someone lies as long as at least a single sentence is a lie.** You encounter 3 inhabitants of the country, they are called Kelsi, Randall and Duff. Kelsi says: ’Duff is not a hairdresser and Randall is not a hairdresser.’ Randall says: ’Duff is a hairdresser and Kelsi is not a chef.’ Duff says: ’Randall is not a chef and Kelsi is a chef.’ What are the types of the different people?
3.3	Change utterances to single sentences without sentence connectives. Add highlighted sentence.	There is a country in which two types of people live, chefs and hairdressers. Chefs lie and hairdressers tell the truth. **Someone lies as long as at least a single sentence is a lie.** You encounter 3 inhabitants of the country, they are called Kelsi, Randall and Duff. Kelsi says: ’Duff is not a hairdresser and Randall is not a hairdresser.’ Randall says: ’Duff is a hairdresser and Kelsi is not a chef.’ Duff says: ’Randall is not a chef and Kelsi is a chef.’ What are the types of the different people?
3.4	Change utterances to single sentences without sentence connectives. Add highlighted sentence. Adds “always” to type description.	There is a country in which two types of people live, chefs and hairdressers. Chefs always lie and hairdressers always tell the truth. **A logical statement is false if at least one side of a conjunction is false.** You encounter 3 inhabitants of the country, they are called Kelsi, Randall and Duff. Kelsi says: ’Duff is not a hairdresser and Randall is not a hairdresser.’ Randall says: ’Duff is a hairdresser and Kelsi is not a chef.’ Duff says: ’Randall is not a chef and Kelsi is a chef.’ What are the types of the different people?

3.5	Change utterances to single sentences without sentence connectives. Add highlighted sentence.	There is a country in which two types of people live, chefs and hairdressers. Chefs lie and hairdressers tell the truth. **A logical statement is false if at least one side of a conjunction is false.** You encounter 3 inhabitants of the country, they are called Kelsi, Randall and Duff. Kelsi says: 'Duff is not a hairdresser and Randall is not a hairdresser.' Randall says: 'Duff is a hairdresser and Kelsi is not a chef.' Duff says: 'Randall is not a chef and Kelsi is a chef.' What are the types of the different people?
3.6	Change utterances to single sentences without sentence connectives. Add highlighted sentences. Adds "always" to type description.	There is a country in which two types of people live, chefs and hairdressers. Chefs always lie and hairdressers always tell the truth. **A logical statement is false if at least one side of a conjunction is false.** **Every utterance should be considered one logical statement.** You encounter 3 inhabitants of the country, they are called Kelsi, Randall and Duff. Kelsi says: 'Duff is not a hairdresser and Randall is not a hairdresser.' Randall says: 'Duff is a hairdresser and Kelsi is not a chef.' Duff says: 'Randall is not a chef and Kelsi is a chef.' What are the types of the different people?
3.7	Change utterances to single sentences without sentence connectives. Add highlighted sentences.	There is a country in which two types of people live, chefs and hairdressers. Chefs lie and hairdressers tell the truth. **A logical statement is false if at least one side of a conjunction is false.** **Every utterance should be considered one logical statement.** You encounter 3 inhabitants of the country, they are called Kelsi, Randall and Duff. Kelsi says: 'Duff is not a hairdresser and Randall is not a hairdresser.' Randall says: 'Duff is a hairdresser and Kelsi is not a chef.' Duff says: 'Randall is not a chef and Kelsi is a chef.' What are the types of the different people?

Table 2: Variations based on the initial baseline, with additional sentences and/or different translation of logic statements.

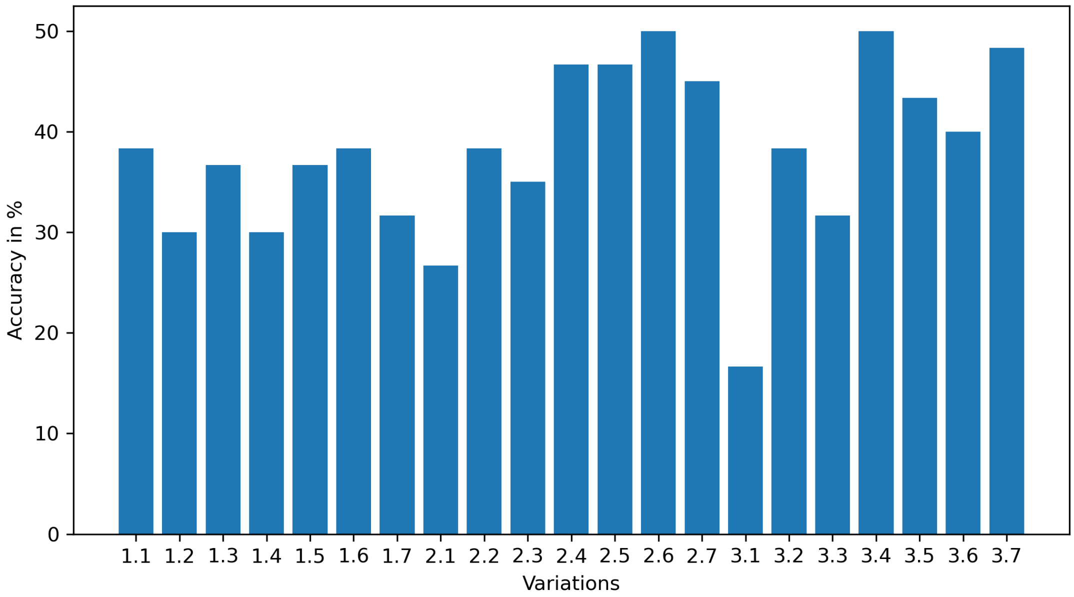

Figure 10: Initial variation testing using Gemini 1.5 Pro, where the y-axis is the accuracy and x indicates the variation used. Each variation is tested on 60 puzzles, with 10 puzzles for each combination between three people in the puzzle and two people per statement, and five people in the puzzle and four people per statement.

Figure 10 is a bar graph that shows the accuracy of the model Gemini 1.5 Pro for different variations of the prompt over a total of 60 unique puzzles. In the figure, it can be seen that the variations 2.6, 3.4, 3.7, 2.4, 2.5 and 2.7 have the highest accuracy of all the tested variations, given the small sample size.

Therefore, these variations should be investigated further, but due to an issue with the parser the initial graph looked different namely, such that 2.6 and 2.4 were not performing well enough to be among the best 6 variations, but instead, 3.2 and 3.5. This therefore means that as seen in Figure 11, the variations 2.5, 2.7, 3.2, 3.4, 3.5 and 3.7 are tested with 600 puzzles each. In this figure, it can be seen that variation 2.7 achieves the highest accuracy among the tested variations, but still with an accuracy below 50%. Therefore, further variations were created to determine whether these improved the performance of the model. The variations are based on variation 2.7 and the exact details of these variations can be seen in Table 3.

Testing these variations on Gemini 1.5 Pro with 600 puzzles results in the bar graph in Figure 12 which shows the accuracy of the model on these variations. In this figure, it can be seen that variant 2.7 still outperformed the other variations, which means that this variant is used for all testing in the results section (Section 5).

Version	Effect	Example

2.7	New Baseline	There is a country in which two types of people live, chefs and hairdressers. Chefs lie and hairdressers tell the truth. A logical statement is false if at least one side of a conjunction is false. Every utterance should be considered one logical statement. You encounter 3 inhabitants of the country, they are called Kelsi, Randall and Duff. Kelsi says: 'Duff is not a hairdresser and moreover, Randall is not a hairdresser.' Randall says: 'Duff is a hairdresser and additionally, Kelsi is not a chef.' Duff says: 'Randall is not a chef and also, Kelsi is a chef.' What are the types of the different people?
2.8	Changed wording (highlighted). Adds "always" to type description.	There is a country in which two types of people live, chefs and hairdressers. Chefs always lie and hairdressers always tell the truth. A logical statement is false if at least one **element** of a conjunction is false. Every utterance should be considered one logical statement. You encounter 3 inhabitants of the country, they are called Kelsi, Randall and Duff. Kelsi says: 'Duff is not a hairdresser and moreover, Randall is not a hairdresser.' Randall says: 'Duff is a hairdresser and additionally, Kelsi is not a chef.' Duff says: 'Randall is not a chef and also, Kelsi is a chef.' What are the types of the different people?

2.9	Changed wording (highlighted).	There is a country in which two types of people live, chefs and hairdressers. Chefs lie and hairdressers tell the truth. A logical statement is false if at least one **element** of a conjunction is false. Every utterance should be considered one logical statement. You encounter 3 inhabitants of the country, they are called Kelsi, Randall and Duff. Kelsi says: 'Duff is not a hairdresser and moreover, Randall is not a hairdresser.' Randall says: 'Duff is a hairdresser and additionally, Kelsi is not a chef.' Duff says: 'Randall is not a chef and also, Kelsi is a chef.' What are the types of the different people?
2.10	Changed wording (highlighted). Adds "always" to type description.	The following is a logical puzzle. There is a country in which two types of people live, chefs and hairdressers. Chefs always lie and hairdressers always tell the truth. A logical statement is false if at least one **element** of a conjunction is false. Every utterance should be considered one logical statement. You encounter 3 inhabitants of the country, they are called Kelsi, Randall and Duff. Kelsi says: 'Duff is not a hairdresser and moreover, Randall is not a hairdresser.' Randall says: 'Duff is a hairdresser and additionally, Kelsi is not a chef.' Duff says: 'Randall is not a chef and also, Kelsi is a chef.' What are the types of the different people?

2.11	Changed wording (highlighted).	The following is a logical puzzle. There is a country in which two types of people live, chefs and hairdressers. Chefs lie and hairdressers tell the truth. A logical statement is false if at least one **element** of a conjunction is false. Every utterance should be considered one logical statement. You encounter 3 inhabitants of the country, they are called Kelsi, Randall and Duff. Kelsi says: 'Duff is not a hairdresser and moreover, Randall is not a hairdresser.' Randall says: 'Duff is a hairdresser and additionally, Kelsi is not a chef.' Duff says: 'Randall is not a chef and also, Kelsi is a chef.' What are the types of the different people?
2.12	Changed wording (highlighted). Adds “always” to type description.	There is a country in which two types of people live, chefs and hairdressers. Chefs always lie and hairdressers always tell the truth. A logical statement is false if at least one **part** of a conjunction is false. Every utterance should be considered one logical statement. You encounter 3 inhabitants of the country, they are called Kelsi, Randall and Duff. Kelsi says: 'Duff is not a hairdresser and moreover, Randall is not a hairdresser.' Randall says: 'Duff is a hairdresser and additionally, Kelsi is not a chef.' Duff says: 'Randall is not a chef and also, Kelsi is a chef.' What are the types of the different people?

2.13	Changed wording (highlighted).	There is a country in which two types of people live, chefs and hairdressers. Chefs lie and hairdressers tell the truth. A logical statement is false if at least one **part** of a conjunction is false. Every utterance should be considered one logical statement. You encounter 3 inhabitants of the country, they are called Kelsi, Randall and Duff. Kelsi says: ’Duff is not a hairdresser and moreover, Randall is not a hairdresser.’ Randall says: ’Duff is a hairdresser and additionally, Kelsi is not a chef.’ Duff says: ’Randall is not a chef and also, Kelsi is a chef.’ What are the types of the different people?
2.14	Changed wording (highlighted). Adds “always” to type description.	There is a country in which two types of people live, chefs and hairdressers. Chefs always lie and hairdressers always tell the truth. A logical statement is false if at least one **conjunct** of a conjunction is false. Every utterance should be considered one logical statement. You encounter 3 inhabitants of the country, they are called Kelsi, Randall and Duff. Kelsi says: ’Duff is not a hairdresser and moreover, Randall is not a hairdresser.’ Randall says: ’Duff is a hairdresser and additionally, Kelsi is not a chef.’ Duff says: ’Randall is not a chef and also, Kelsi is a chef.’ What are the types of the different people?

2.15	Changed wording (highlighted).	There is a country in which two types of people live, chefs and hairdressers. Chefs lie and hairdressers tell the truth. A logical statement is false if at least one **conjunct** of a conjunction is false. Every utterance should be considered one logical statement. You encounter 3 inhabitants of the country, they are called Kelsi, Randall and Duff. Kelsi says: 'Duff is not a hairdresser and moreover, Randall is not a hairdresser.' Randall says: 'Duff is a hairdresser and additionally, Kelsi is not a chef.' Duff says: 'Randall is not a chef and also, Kelsi is a chef.' What are the types of the different people?
2.16	Changed wording (highlighted). Adds "always" to type description.	There is a country in which two types of people live, chefs and hairdressers. Chefs always lie and hairdressers always tell the truth. A logical statement is false if at least one **literal** of a conjunction is false. Every utterance should be considered one logical statement. You encounter 3 inhabitants of the country, they are called Kelsi, Randall and Duff. Kelsi says: 'Duff is not a hairdresser and moreover, Randall is not a hairdresser.' Randall says: 'Duff is a hairdresser and additionally, Kelsi is not a chef.' Duff says: 'Randall is not a chef and also, Kelsi is a chef.' What are the types of the different people?

2.17	Changed wording (highlighted).	There is a country in which two types of people live, chefs and hairdressers. Chefs lie and hairdressers tell the truth. A logical statement is false if at least one **literal** of a conjunction is false. Every utterance should be considered one logical statement. You encounter 3 inhabitants of the country, they are called Kelsi, Randall and Duff. Kelsi says: 'Duff is not a hairdresser and moreover, Randall is not a hairdresser.' Randall says: 'Duff is a hairdresser and additionally, Kelsi is not a chef.' Duff says: 'Randall is not a chef and also, Kelsi is a chef.' What are the types of the different people?
2.18	Changed wording (highlighted). Adds "always" to type description.	There is a country in which two types of people live, chefs and hairdressers. Chefs always lie and hairdressers always tell the truth. A logical statement is false if at least one **partial statement** of a conjunction is false. Every utterance should be considered one logical statement. You encounter 3 inhabitants of the country, they are called Kelsi, Randall and Duff. Kelsi says: 'Duff is not a hairdresser and moreover, Randall is not a hairdresser.' Randall says: 'Duff is a hairdresser and additionally, Kelsi is not a chef.' Duff says: 'Randall is not a chef and also, Kelsi is a chef.' What are the types of the different people?

2.19	Changed wording (highlighted).	There is a country in which two types of people live, chefs and hairdressers. Chefs lie and hairdressers tell the truth. A logical statement is false if at least one **partial statement** of a conjunction is false. Every utterance should be considered one logical statement. You encounter 3 inhabitants of the country, they are called Kelsi, Randall and Duff. Kelsi says: 'Duff is not a hairdresser and moreover, Randall is not a hairdresser.' Randall says: 'Duff is a hairdresser and additionally, Kelsi is not a chef.' Duff says: 'Randall is not a chef and also, Kelsi is a chef.' What are the types of the different people?

Table 3: Additional variations based on the previously described variation 2.7.

C LLM answers

In this section, the remaining examples described in the Qualitative Results (Section 5.1). As described before the reasoning models only show the final result as seen in Examples 11, 12, 13, 14, 15, 16 and 17. The responses from Gemini 1.5 Pro use a similar strategy compared to what was described before, though as seen in Examples 8, 9 and 10. Though, in these examples the model does not translate the statements to reduce negation, but it generally seems to use the same general strategy.

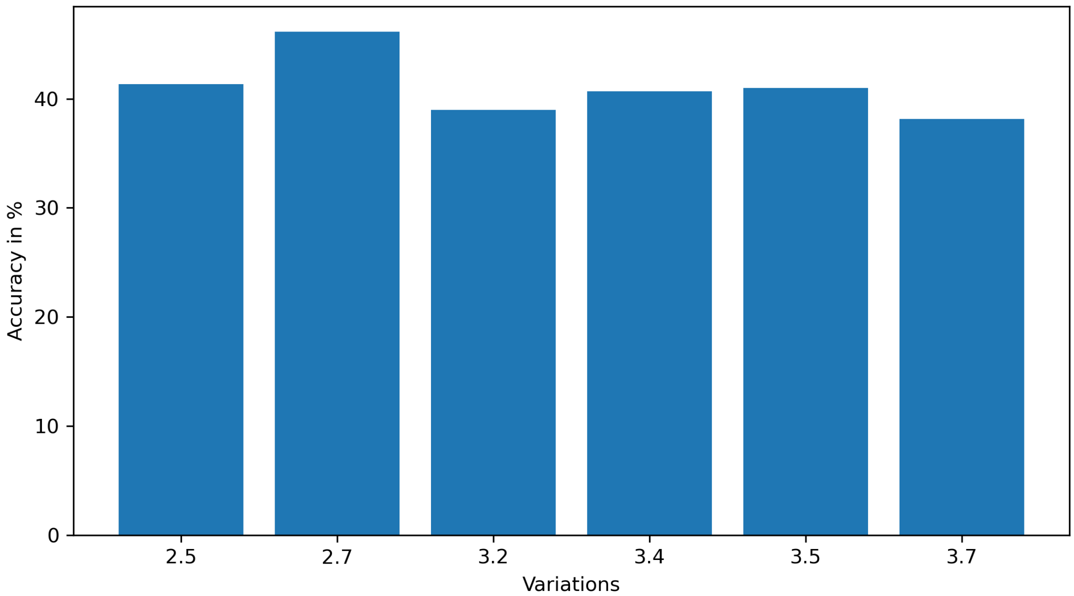

Figure 11: Further variation testing using Gemini 1.5 Pro, where the y-axis is the accuracy and x indicates the variation used. Each variation is tested on 600 puzzles, with 100 puzzles for each combination between 3 people in the puzzle and 2 people per statement, and 5 people in the puzzle and 4 people per statement.

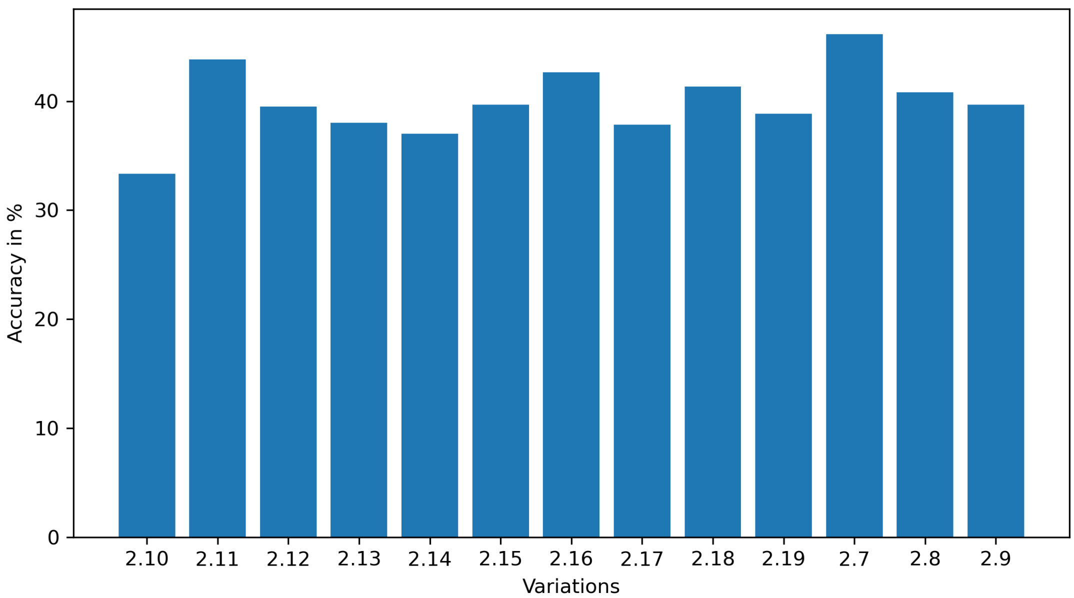

Figure 12: Further variation testing using Gemini 1.5 Pro, where the y-axis is the accuracy and x indicates the variation used. Each variation is tested on 600 puzzles, with 100 puzzles for each combination between 3 people in the puzzle and 2 people per statement, and 5 people in the puzzle and 4 people per statement.

Q: There is a cave in which two types of people live, car mechanics and flight attendants. Car mechanics lie and flight attendants tell the truth. A logical statement is false if at least one side of a conjunction is false. Every utterance should be considered one logical statement. You encounter 5 inhabitants of the cave, they are called Norton, Shayna, Naomi, Litha and Berta. Norton says: 'Berta is a flight attendant and additionally, Litha is a car mechanic and also, Naomi is a car mechanic and moreover, Shayna is a flight attendant.' Shayna says: 'Berta is not a car mechanic and moreover, Litha is not a car mechanic and further, Naomi is not a car mechanic and further, Norton is a car mechanic.' Naomi says: 'Berta is a flight attendant and also, Litha is a car mechanic and also, Shayna is not a car mechanic and moreover, Norton is not a flight attendant.' Litha says: 'Berta is not a car mechanic and additionally, Naomi is a car mechanic and further, Shayna is not a car mechanic and moreover, Norton is not a flight attendant.' Berta says: 'Litha is a flight attendant and additionally, Naomi is a flight attendant and moreover, Shayna is a flight attendant and further, Norton is not a flight attendant.' What are the types of the different people? Please end your answer in the format: name : type;

A: Let's analyze each statement, keeping in mind that conjunctions are false if at least one part is false.

* **Norton:** ""'[...]"" If Norton is a car mechanic (lying), at least one part of this statement must be false. If Norton is a flight attendant (truth-telling), the entire statement must be true.

* **Shayna:** ""'[...]"" If Shayna is a car mechanic (lying), at least one part of this statement must be false. If Shayna is a flight attendant (truth-telling), the entire statement must be true.

* **Naomi:** ""'[...]"" If Naomi is a car mechanic (lying), at least one part of this statement must be false. If Naomi is a flight attendant (truth-telling), the entire statement must be true.

* **Litha:** ""'[...]"" If Litha is a car mechanic (lying), at least one part of this statement must be false. If Litha is a flight attendant (truth-telling), the entire statement must be true.

* **Berta:** ""'[...]"" If Berta is a car mechanic (lying), at least one part of this statement must be false. If Berta is a flight attendant (truth-telling), the entire statement must be true.

Let's assume Norton is a flight attendant (truth-telling). This means Berta is a flight attendant, Litha is a car mechanic, Naomi is a car mechanic, and Shayna is a flight attendant. Let's see if this holds up with the other statements:

* **Shayna (flight attendant):** Berta is not a car mechanic (True), Litha is not a car mechanic (False), Naomi is not a car mechanic (False), Norton is a car mechanic (False). Since Shayna is supposedly telling the truth, this scenario is invalid.

Let's assume Norton is a car mechanic (lying).

* **Shayna:** If Shayna is a flight attendant, then Berta is a flight attendant, Litha is a flight attendant, Naomi is a flight attendant, and Norton is a car mechanic. * **Naomi (flight attendant):** Berta is a flight attendant (True), Litha is a car mechanic (False), Shayna is not a car mechanic (True), Norton is not a flight attendant (True). This contradicts the assumption. * **Shayna:** If Shayna is a car mechanic: At least one part of her statement must be false.

Let's consider Berta is a flight attendant:

* **Berta:** Litha is a flight attendant, Naomi is a flight attendant, Shayna is a flight attendant, Norton is a car mechanic. * **Naomi (flight attendant):** Berta is a flight attendant (True), Litha is a car mechanic (False), Shayna is not a car mechanic (True), Norton is not a flight attendant (True). Contradiction.

If Berta is a car mechanic:

* Assume Shayna is a car mechanic, Naomi is a car mechanic, and Litha is a flight attendant. Norton would have to be a car mechanic (from Berta's statement). *** **Litha (flight attendant):** Berta is not a car mechanic (False). This works.**

Therefore: Norton is a car mechanic, Shayna is a car mechanic, Naomi is a car mechanic, Litha is a flight attendant, and Berta is a car mechanic.

Norton : car mechanic; Shayna : car mechanic; Naomi : car mechanic; Litha : flight attendant; Berta : car mechanic;

Example 8: Incorrect answer from Gemini 1.5 Pro. The highlighted text shows errors.

Q: There is a swamp in which two types of people live, nurses and hairdressers. Nurses lie and hairdressers tell the truth. A logical statement is false if at least one side of a conjunction is false. Every utterance should be considered one logical statement. You encounter 5 inhabitants of the swamp, they are called Talmadge, Tammi, Dawn, Eleanor and Christy. Talmadge says: 'Christy is a nurse and moreover, Eleanor is not a hairdresser and moreover, Tammi is a nurse.' Tammi says: 'Christy is not a nurse and additionally, Dawn is a hairdresser and also, Talmadge is a hairdresser.' Dawn says: 'Christy is a hairdresser and also, Eleanor is a hairdresser and further, Tammi is a hairdresser.' Eleanor says: 'Christy is a hairdresser and also, Dawn is a hairdresser and further, Tammi is not a hairdresser.' Christy says: 'Eleanor is not a nurse and moreover, Dawn is not a nurse and also, Tammi is a nurse.' What are the types of the different people? Please end your answer in the format: name : type

A: Let's analyze each statement, keeping in mind that nurses lie and hairdressers tell the truth.

* **Talmadge:** ""'Christy is a nurse and Eleanor is not a hairdresser and Tammi is a nurse.""' If Talmadge is a nurse, this whole statement must be false. If Talmadge is a hairdresser, the whole statement must be true.

* **Tammi:** ""'Christy is not a nurse and Dawn is a hairdresser and Talmadge is a hairdresser.""' If Tammi is a nurse, this whole statement must be false. If Tammi is a hairdresser, the whole statement must be true.

* **Dawn:** ""'Christy is a hairdresser and Eleanor is a hairdresser and Tammi is a hairdresser.""' If Dawn is a nurse, this whole statement must be false. If Dawn is a hairdresser, the whole statement must be true.

* **Eleanor:** ""'Christy is a hairdresser and Dawn is a hairdresser and Tammi is not a hairdresser.""' If Eleanor is a nurse, this whole statement must be false. If Eleanor is a hairdresser, the whole statement must be true.

* **Christy:** ""'Eleanor is not a nurse and Dawn is not a nurse and Tammi is a nurse.""' If Christy is a nurse, this whole statement must be false. If Christy is a hairdresser, the whole statement must be true.

Let's assume Talmadge is a hairdresser. Then Christy is a nurse, Eleanor is not a hairdresser (so Eleanor is a nurse), and Tammi is a nurse.

If Tammi is a nurse, then her statement is false. So, at least one of the following must be true: Christy *is* a nurse, Dawn is *not* a hairdresser, or Talmadge is *not* a hairdresser. We've already established Christy is a nurse and Talmadge is a hairdresser (our assumption). Therefore, Dawn must not be a hairdresser, so Dawn is a nurse.

If Dawn is a nurse, her statement is false. So, at least one of the following must be true: Christy is not a hairdresser, Eleanor is not a hairdresser, or Tammi is not a hairdresser. We know Christy is a nurse and Tammi is a nurse. So either Christy and Tammi are not hairdressers (consistent with them being nurses) and Eleanor is not a hairdresser (consistent with being a nurse), or one of the latter is true which works with the other two being not hairdressers.

If Eleanor is a nurse, her statement is false. So at least one of the following must be true: Christy is not a hairdresser, Dawn is not a hairdresser, or Tammi *is* a hairdresser. We already know Christy is a nurse (not a hairdresser), Dawn is a nurse (not a hairdresser), and Tammi is a nurse (not a hairdresser). So Eleanor being a nurse is consistent.

If Christy is a nurse, her statement is false. So at least one of the following must be true: Eleanor *is* a nurse, Dawn *is* a nurse, or Tammi is *not* a nurse. This is consistent with our findings so far.

Therefore, our assumption that Talmadge is a hairdresser is consistent with all statements.

Final Answer: Talmadge : hairdresser; Tammi : nurse; Dawn : nurse; Eleanor : nurse; Christy : nurse;

Example 9: Correct answer from Gemini 1.5 Pro.

Q: There is a swamp in which two types of people live, soldiers and maids. Soldiers lie and maids tell the truth. A logical statement is false if at least one side of a conjunction is false. Every utterance should be considered one logical statement. You encounter 4 inhabitants of the swamp, they are called Joleen, Jonell, Selene and Racquel. Joleen says: 'Racquel is not a maid and also, Selene is not a soldier and additionally, Jonell is a soldier.' Jonell says: 'Racquel is not a soldier and additionally, Selene is not a maid and additionally, Joleen is not a maid.' Selene says: 'Racquel is a soldier and additionally, Jonell is not a maid and additionally, Joleen is not a maid.' Racquel says: 'Selene is not a maid and also, Jonell is a maid and also, Joleen is not a maid.' What are the types of the different people? Please end your answer in the format: name : type

A: Let's represent the statement ""X is a soldier"" as S(X) and ""X is a maid"" as M(X). Since there are only two types of people, M(X) is equivalent to not S(X), and S(X) is equivalent to not M(X).

* **Joleen:** ¬M(Racquel) ∧ ¬S(Selene) ∧ S(Jonell) * **Jonell:** ¬S(Racquel) ∧ ¬M(Selene) ∧ ¬M(Joleen) * **Selene:** S(Racquel) ∧ ¬M(Jonell) ∧ ¬M(Joleen) * **Racquel:** ¬M(Selene) ∧ M(Jonell) ∧ ¬M(Joleen)

Now, let's analyze each person's statement:

1. **Assume Joleen is a maid.** Then her statement is true. This means Racquel is not a maid (so a soldier), Selene is not a soldier (so a maid), and Jonell is a soldier. Let's see if this holds up with the other statements: * **Jonell (soldier):** ""Racquel is not a soldier"" (False). The conjunction is false, which is consistent with Jonell being a soldier (and therefore lying). * **Selene (maid):** ""Racquel is a soldier and Jonell is not a maid and Joleen is not a maid."" (False because Joleen *is* a maid). This contradicts our initial assumption, so Joleen cannot be a maid.

2. **Assume Joleen is a soldier.** Then her statement is false. This means at least one of the following must be true: Racquel *is* a maid, Selene *is* a soldier, or Jonell *is* a maid. * **Consider the case where Racquel is a maid:** * **Jonell:** ¬S(Racquel) ∧ ¬M(Selene) ∧ ¬M(Joleen). If Jonell is a maid, this statement is true. This means Racquel is a maid (consistent), Selene is a soldier, and Joleen is a soldier (consistent). * **Selene (soldier):** S(Racquel) ∧ ¬M(Jonell) ∧ ¬M(Joleen). This statement would be false (Racquel isn't a soldier), consistent with Selene being a soldier. * **Racquel (maid):** ¬M(Selene) ∧ M(Jonell) ∧ ¬M(Joleen). This statement would be true (Selene is a soldier, Jonell is a maid, and Joleen is a soldier), consistent with Racquel being a maid.

This scenario works! Let's summarize: Joleen is a soldier, Racquel is a maid, Jonell is a maid, and Selene is a soldier.

Joleen : soldier; Jonell : maid; Selene : soldier; Racquel : maid;

Example 10: Correct answer from Gemini 1.5 Pro.

Q: There is a planet on which two types of people live, musicians and astronauts. Musicians lie and astronauts tell the truth. A logical statement is false if at least one side of a conjunction is false. Every utterance should be considered one logical statement. You encounter 4 inhabitants of the planet, they are called Mikalah, Danny, Cari and Vidal. Mikalah says: 'Vidal is not a musician and additionally, Cari is not an astronaut and also, Danny is not an astronaut.' Danny says: 'Vidal is not an astronaut and further, Cari is not an astronaut and additionally, Mikalah is not an astronaut.' Cari says: 'Vidal is not a musician and further, Danny is an astronaut and further, Mikalah is not an astronaut.' Vidal says: 'Cari is an astronaut and also, Danny is not a musician and moreover, Mikalah is not an astronaut.' What are the types of the different people? Please end your answer in the format: name : type

A: Mikalah : musician;
Danny : astronaut;
Cari : musician;
Vidal : musician;

Example 11: Correct answer from o1-mini.

Q: There is a continent on which two types of people live, fighter pilots and butchers. Fighter pilots lie and butchers tell the truth. A logical statement is false if at least one side of a conjunction is false. Every utterance should be considered one logical statement. You encounter 5 inhabitants of the continent, they are called Simeon, Sasha, Donal, Fiona and Augustus.
Simeon says: 'Augustus is not a butcher and also, Fiona is a fighter pilot.'
Sasha says: 'Fiona is a butcher and also, Simeon is a butcher.'
Donal says: 'Augustus is a fighter pilot and also, Sasha is a butcher.'
Fiona says: 'Augustus is not a butcher and also, Simeon is a butcher.'
Augustus says: 'Fiona is not a butcher and additionally, Donal is not a butcher.'
What are the types of the different people? Please end your answer in the format: name : type

A: Based on the logical analysis of each statement and the types of inhabitants, the classifications are as follows:
‘‘‘
Simeon : Fighter Pilot;
Sasha : Fighter Pilot;
Donal : Fighter Pilot;
Fiona : Fighter Pilot;
Augustus : Butcher;

Example 12: Correct answer from o1-mini.

Q: There is an earldom in which two types of people live, miners and reporters. Miners lie and reporters tell the truth. A logical statement is false if at least one side of a conjunction is false. Every utterance should be considered one logical statement. You encounter 5 inhabitants of the earldom, they are called Kirby, Ernestine, Harvey, Vicente and Achsah. Kirby says: 'Vicente is a reporter and also, Ernestine is a miner.' Ernestine says: 'Vicente is not a miner and moreover, Harvey is a reporter.' Harvey says: 'Vicente is a miner and also, Kirby is not a reporter.' Vicente says: 'Achsah is a reporter and additionally, Ernestine is a miner.' Achsah says: 'Vicente is not a miner and also, Harvey is a reporter.' What are the types of the different people? Please end your answer in the format: name : type

A: After analyzing the statements and considering the logical constraints, the consistent assignments of types are as follows:
- **Kirby** is a **reporter**.
- **Ernestine** is a **miner**.
- **Harvey** is a **miner**.
- **Vicente** is a **miner**.
- **Achsah** is a **miner**.
Final Assignments:
‘‘‘
Kirby : reporter;
Ernestine : miner;
Harvey : miner;
Vicente : miner;
Achsah : miner;

Example 13: Incorrect answer from o1-mini.

Q: There is an island on which two types of people live, astronomers and astronauts. Astronomers lie and astronauts tell the truth. A logical statement is false if at least one side of a conjunction is false. Every utterance should be considered one logical statement. You encounter 5 inhabitants of the island, they are called Tyron, Ester, Deyanira, Renata and Margarita. Tyron says: 'Margarita is not an astronaut and also, Renata is an astronomer and moreover, Deyanira is not an astronaut.' Ester says: 'Renata is an astronaut and additionally, Deyanira is not an astronaut and also, Tyron is not an astronaut.' Deyanira says: 'Margarita is an astronomer and also, Renata is an astronomer and moreover, Tyron is an astronaut.' Renata says: 'Margarita is not an astronomer and further, Ester is not an astronomer and moreover, Tyron is an astronomer.' Margarita says: 'Renata is an astronaut and moreover, Deyanira is an astronomer and additionally, Ester is not an astronomer.' What are the types of the different people? Please end your answer in the format: name : type

A: Tyron : astronaut;
Ester : astronomer;
Deyanira : astronomer;
Renata : astronomer;
Margarita : astronomer;

Example 14: Incorrect answer from o1-mini.

Q: There is a house in which two types of people live, make-up artists and computer programmers. Make-up artists lie and computer programmers tell the truth. A logical statement is false if at least one side of a conjunction is false. Every utterance should be considered one logical statement. You encounter 3 inhabitants of the house, they are called Christian, Starr and Sharee. Christian says: 'Sharee is not a computer programmer and also, Starr is a make-up artist.' Starr says: 'Sharee is a computer programmer and moreover, Christian is a computer programmer.' Sharee says: 'Starr is not a make-up artist and also, Christian is not a computer programmer.' What are the types of the different people? Please end your answer in the format: name : type

A: Christian : computer programmer;
Starr : make-up artist;
Sharee : make-up artist;

Example 15: Correct answer from o4-mini.

Q: There is a continent on which two types of people live, car mechanics and cashiers. Car mechanics lie and cashiers tell the truth. A logical statement is false if at least one side of a conjunction is false. Every utterance should be considered one logical statement. You encounter 5 inhabitants of the continent, they are called Nita, Nora, Inez, Marland and Oral. Nita says: ’Oral is a car mechanic and also, Marland is not a car mechanic and additionally, Inez is a car mechanic and moreover, Nora is not a car mechanic.’ Nora says: ’Oral is not a cashier and moreover, Marland is a car mechanic and further, Inez is not a car mechanic and moreover, Nita is not a car mechanic.’ Inez says: ’Oral is not a cashier and moreover, Marland is not a car mechanic and also, Nora is a cashier and additionally, Nita is not a car mechanic.’ Marland says: ’Oral is not a cashier and further, Inez is not a cashier and further, Nora is not a cashier and further, Nita is not a car mechanic.’ Oral says: ’Marland is a car mechanic and further, Inez is not a cashier and also, Nora is not a cashier and moreover, Nita is a car mechanic.’ What are the types of the different people? Please end your answer in the format: name : type

A: Nita : car mechanic;
Nora : car mechanic;
Inez : car mechanic;
Marland : car mechanic;
Oral : car mechanic;

Example 16: Incorrect answer from o4-mini.

Q: There is an island on which two types of people live, astronomers and astronauts. Astronomers lie and astronauts tell the truth. A logical statement is false if at least one side of a conjunction is false. Every utterance should be considered one logical statement. You encounter 5 inhabitants of the island, they are called Tyron, Ester, Deyanira, Renata and Margarita. Tyron says: ’Margarita is not an astronaut and also, Renata is an astronomer and moreover, Deyanira is not an astronaut.’ Ester says: ’Renata is an astronaut and additionally, Deyanira is not an astronaut and also, Tyron is not an astronaut.’ Deyanira says: ’Margarita is an astronomer and also, Renata is an astronomer and moreover, Tyron is an astronaut.’ Renata says: ’Margarita is not an astronomer and further, Ester is not an astronomer and moreover, Tyron is an astronomer.’ Margarita says: ’Renata is an astronaut and moreover, Deyanira is an astronomer and additionally, Ester is not an astronomer.’ What are the types of the different people? Please end your answer in the format: name : type

A: A consistent assignment (assuming “astronomers” tell the truth and “astronauts” lie) is:
Tyron : astronaut; Ester : astronaut; Deyanira: astronaut; Renata : astronomer; Margarita: astronaut.

Example 17: Incorrect answer from o4-mini.