Can An LLM Elicit Information From Users In Simple Optimization Modelling Dialogues?

Yelaman Abdullin[1], Diego Molla-Aliod[1], Bahadorreza Ofoghi[2], Vicky Mak-Hau[2], John Yearwood[2],

[1] School of Computing, Macquarie University, Sydney, Australia
[2] School of Information Technology, Deakin University, Melbourne, Australia
Correspondence: yelaman.abdullin@hdr.mq.edu.au

Abstract

For a natural language dialogue system to engage in a goal-oriented conversation, it must elicit information from a user. Research on large language models (LLMs) often focuses on aligning them with user goals. Consequently, studies show these models can serve as chat assistants and answer the user questions. However, their information-elicitation abilities remain understudied. This work evaluates these abilities in goal-oriented dialogues for optimisation modelling. We compare two GPT-4-based settings that generate conversations between a modeller and a user over NL4Opt, a collection of simple optimisation problem descriptions, and analyse the modeller's information elicitation. In the first, the modeller LLM has access to problem details and asks targeted questions, simulating an informed modeller. In the second, the LLM infers problem details through interaction — asking clarifying questions, interpreting responses, and gradually constructing an understanding of the task. This comparison assesses whether LLMs can elicit information and navigate problem discovery without prior knowledge of the problem. We compare modeller turns in both settings using human raters across criteria at the whole-dialogue and turn levels. Results show that a non-informed LLM can elicit information nearly as well as an informed one, producing high-quality dialogues. In particular, the success levels of both agents in the system without modeller access to the problem details are comparable to those in a system with full access. Dialogues rate well on coherence, and a post-annotation error analysis identified useful types for improving quality. GPT-4's capability to elicit information in optimisation modelling dialogues suggests newer LLMs may possess even greater capability.

1 Introduction

Mathematical optimisation is an essential cornerstone of modern business decision making, enabling organisations to allocate resources efficiently, minimise costs, and maximise profits in various domains, including supply chain management, production planning, financial portfolio optimisation, and logistics (Bertsimas and Tsitsiklis, 1997). Traditionally, the formulation of optimisation problems is conducted through iterative consultations between optimisation specialists and domain experts, involving multiple rounds of information gathering, clarification, and refinement.

The emergence of large language models (LLMs) presents a compelling opportunity to automate and enhance the information elicitation process through a goal-oriented dialogue system. This dialogue system simulates the optimisation specialist who interviews the domain expert, with the aim to obtain the necessary information for the formulation of the optimisation problem. Much work has been done in dialogue. The Association for Computational Linguistics has a special interest group (Lemon et al., 2022) with this general focus and, more recently, also on goal-oriented dialogue, generation, and evaluation. There has also been considerable work on dialogue systems underpinning chatbots and their evaluation (Fellows et al., 2022). However, the application of LLMs to optimisation modelling dialogue represents a particularly challenging and understudied area. In this work, we investigate whether large language models (LLMs) can effectively engage in information elicitation within the context of optimisation modelling — a structured and high-stakes form of goal-oriented dialogue.

This setting, optimisation modelling, serves as a good testbed for evaluating the information-elicitation abilities of dialogue agents because it requires the modeller to uncover precise, structured information from a user who may not initially express their problem in formal terms. Unlike general chit-chat or even some other forms of goal-oriented dialogue (e.g., booking a restaurant), optimisation dialogues demand both domain expertise and a disciplined questioning strategy to arrive at a well-formed mathematical representation. The modeller or modelling agent interacts with a user who wants to optimise their business or some aspect of their business. The modelling agent engages in a conversation with the user about the nature of their business to find out what exactly they want to improve and optimise. The modelling agent has expert knowledge of mathematical optimisation and how to lead modelling conversations to ascertain the nature of the applicable mathematics and the particular requirements to be solicited. In this work, we focus on the dialogue task of acquiring a first take on the nature of the problem as a linear programming (LP) problem and on obtaining the required information.

To assess the information elicitation ability of the modelling agent, we automatically generate synthetic dialogues between the modeller and user. Two methods, one where the modeller has access to the problem description, and another where it does not, were employed to automatically generate these dialogues as described in Section 3. We use text descriptions of simple business optimisation problems from the NL4OPT dataset (Ramamonjison et al., 2022, 2023), and this is considered the base information for the business situation and problem.

The central question we address is whether an LLM, acting as a modeller without prior access to the full problem description, can elicit all the relevant information through dialogue alone — mirroring how a real-world modeller would need to interact with a client who cannot formally specify their problem upfront. In this paper, the focus is on the extent to which the modelling conversation led by the modelling agent can capture the information on the business and required optimisation to effectively build the mathematical model. This information should be contained in the answers that a user provides and that the modeller summarises at the end of the dialogue. We then conduct a human evaluation of the quality of the synthetic dialogues and assess the elicitation abilities of the modeller. We also identify areas for potential improvement in the dialogues in general, and the modeller agent in particular.

Novel contributions of our methodology include: (1) the systematic evaluation of LLM information elicitation capabilities in optimisation modelling contexts, and (2) an evaluation via simulating modelling conversations without prior problem knowledge.

2 Background

2.1 Linear Programming

Linear programming (LP) is a branch of applied mathematics that provides an optimised solution to a mathematical model that comprises a set of decision variables, an objective function to be maximised or minimised, and a set of linear constraints. An LP is a) an integer linear program (ILP) if all decision variables are integers; b) a binary integer linear program if all decision variables are binary; and c) a mixed-integer linear program (MILP) is the set of decision variables is a mix of integer, binary, and rational variables. The objective is what "drives" the optimisation of an LP: one either minimises a linear combination of the decision variables or maximises it, and different linear combinations of the decision variables. A constraint represents a requirement from the user, typically expressed as a linear combination of decision variables (on the left-hand side) that is either $\geq$, $\leq$ or $=$, and a constant (on the right-hand side).

When a human LP practitioner formulates a model, they will typically have a conversation with the user to elicit all the required information. Information such as what the decisions are (for example, whether or not to perform a task, how much of a resource one can consume, or the number of products to produce); what the goals are (for example, to minimise cost, to maximise profit, to minimise the travel time); or what restrictions are involved (such as: a lower limit that must be met, an upper limit that must not be exceeded, or some proportion requirements must be followed).

Figure 1 shows a constructed example of a successful conversation for eliciting information related to a simple LP model. The goal of this paper is to simulate these dialogues automatically. At the end of the dialogue, the modeller will write a summary of the problem that lists all the constraints that have been elicited. Figure 4 shows the output generated by the system described in Section 3, from a problem description shown in Figure 4.

Real-life problems are much harder. The choice of initial data is constrained by the lack of sets of more complex problems at the time of developing this research. Some examples of problems are course timetabling (Esmaeilbeigi et al., 2022), satellite communications dynamic resource allocation (Gaudry et al., 2023), kidney exchange optimisation (Lam and Mak-Hau, 2020), and pilot training (Mak-Hau et al., 2021). More importantly,

Modeller: *What kind of decision do you need to make for your business?*
User: *I make coffee mugs, and need to decide how many to make for each size: small, medium, and large.*
Modeller: *What is the goal of your business?*
User: *I want to make as much profit as I possibly can*
Modeller: *What are the profits associated with the different sizes?*
User: *I can make a profit of $2 for a small mug, $3 for a medium mug, and $4 for a large mug.*
Modeller: *Are there anything that limits how many mugs you can make?*
User: *There are two limiting factors. First, the availability of glass*
Modeller: *How much glass is required for each size of the mugs, and what is the resource you have?*
User: *Each small mug requires 10 oz of glass, each medium mug requires 12 oz of glass, and each large mug requires 15 oz of glass. There are 1000 oz of glass available in my factory.*
Modeller: *So what is the other limiting factor?*
User: *The plastic used for the lid. A small mug requires 2 units of plastic, a medium mug requires 3, and a large mug requires 3.5. There are 835 units of plastic available in my factory.*
Modeller: *Any other restrictions or business requirements?*
User: *Nope.*

Figure 1: Constructed example of a successful conversation between a modeller and a user.

a user may not be able to articulate their business requirement clearly, and some guidance will be needed from the human modeller.

2.2 Goal-Oriented Conversational Agents

Conversational agents (CAs) are software agents that engage in interactions with users through written or spoken natural language. Goal-oriented CAs are a specialised kind of CA engineered to engage in natural language dialogues with humans, with the primary objective of accomplishing specific tasks. Developing sophisticated goal-oriented CAs is challenging, drawing notable attention from industry leaders due to rising market demand and wide domain applications. For example, modern hospitals employ artificial intelligence (AI) doctors, assisted by goal-oriented CAs, to aid patients in their medical care (Valizadeh and Parde, 2022).

In the past, goal-oriented CAs have been developed for a diverse range of objectives, ranging from group discussion (Hogan et al., 2021) to teaching business process models (Rooein et al., 2022). Prior research in the domain of goal-oriented CAs can be categorised into two main groups: pipeline approaches and end-to-end methods. The development of pipeline-based goal-oriented CAs relied on conventional natural language processing (NLP) techniques, including the utilisation of off-the-shelf tools for keyword identification and dialogue state tracking (Williams et al., 2014).

In contrast, end-to-end methods construct the system using a single model that directly takes a natural language command as input and generates a natural language response as output (Peng et al., 2022). Most end-to-end systems use neural models to perform dialogue generation. The progress in constructing end-to-end trainable goal-oriented CAs was limited by the scarcity of suitable data.

The assessment of the quality of dialogues generated by goal-oriented conversational agents has focused on two key aspects: (i) the cost associated with dialogue generation, and (ii) the extent to which the conversational goals are comprehensively addressed. Dialogues generated by CAs are often evaluated by human annotators, where the evaluation criteria mostly cover goal achieved rate, irrelevant and redundant generation rate, and user satisfaction level. For example, in (Shi et al., 2019), evaluators are asked to rate the dialogues generated by CAs, representing an agent and a human simulator agent. Apart from the human evaluation of the dialogues generated by the CAs, simulated evaluation has also been widely used by adapting a similar set of criteria. The structured nature of goal-oriented dialogues facilitates simulated evaluations. Recent goal-oriented CA platforms, such as ConvLab (Lee et al., 2019) and PyDial (Ultes et al., 2017), have utilised this approach.

In recent times, the rise of advanced language models like OpenAI's[1] Generative Pre-trained Transformer (GPT) has created opportunities to automate the evaluation of language generation systems, including the goal-oriented CAs (Lin and Chen, 2023; Liu et al., 2023). (Abdullin et al., 2023) applies LLMs for generating goal-oriented synthetic dialogues for optimisations, and (Mirabi et al., 2025) studies types of noise in goal-oriented dialogues.

3 Methods

To test the information elicitation ability of an LLM, we generated dialogues using problem descriptions with two methods: the dual-agent method and a single-completion method.

The natural language problem descriptions taken from the NL4Opt dataset serve as the foundational material for our synthetic dialogues. The primary

[1]https://openai.com/

aim of the setup is to facilitate a dialogue where the **modeller** systematically extracts key information about a given LP problem by interacting with the **user**, which is designed to simulate a user's behaviour by impersonating a person described in the problem statement. The setup maintains the integrity of the original problem statement using the methods described below.

Our goal is to assess whether LLMs can effectively perform the role of a modeller — eliciting the necessary information from a user in order to formulate an optimisation problem. The key difference between the two methods lies in the level of access to the problem description. In the dual agent method, the modeller agent does **not** have direct access to the problem description and must generate a meaningful dialogue while navigating uncertainty, particularly focusing on eliciting key information through the conversation. In contrast, the single-completion method allows the modeller to predetermine questions based on the full problem description. In both methods, the user agent has access to the problem description, so that it can answer the questions asked by the modeller agent.

This distinction reflects a contrast between real-world modelling conversations, where the modeller must uncover the problem details interactively, and idealised settings where the problem is already specified in full. By comparing the two, we examine the LLM's capacity to function in realistic modelling scenarios that require adaptive questioning, hypothesis testing, and clarification.

The dual-agent method for generating synthetic dialogues utilises a dual-agent setup similar to (Abdullin et al., 2023; Mirabi et al., 2025), where each agent is powered by an LLM to simulate user-modeller interactions centred on LP problems, plus a third LLM that acts as a comparing component and informs one of the agents as described below. The system architecture builds upon OpenAI's GPT-4 (OpenAI, 2023), and the conversational content is generated by having the two agents interact with each other, emphasising natural progression and information extraction from given problem statements.

The single-completion method, while serving as a high baseline, follows a similar approach to the one described by Hong et al. (2023), where an LLM generates the entire conversation in one go based on a problem description. This method serves as a useful benchmark to measure the upper bound of the system's performance when full information is available.

Figure 4 in Appendix B shows an example of a generated dialogue using a problem description that is shown on Figure 3. This text description is not part of the NLP4Opt dataset, and has been included here for comparison with Figure 1.

The specific prompts used for the LLMs are included in Appendix C.

3.1 Question Generation And Prompt Engineering

An effective dialogue system's essence, especially focusing on goal-oriented tasks, lies in its ability to ask relevant questions that elicit the necessary information. The modelling agent is carefully crafted, using prompt engineering techniques, to play this pivotal role in our dual-agent setup. For our modelling agent, prompt engineering is employed to refine interaction dynamics, and to elicit information.

3.1.1 Refine Interaction Dynamics

By adeptly structuring the prompts, we encourage the modelling agent to ask questions in a sequential manner, ensuring that the dialogue evolves logically and comprehensively. This approach aids in segmenting the LP problem statement step-by-step, creating a natural conversational flow. We also incorporate a structured one-shot example explaining how summaries should be constructed. This addition ensures the modelling agent has a clear guideline for the desired output format and content, promoting more accurate and standardised summaries.

3.1.2 Elicit Information

Using tailored prompts, the modelling agent is guided to ask more detailed questions, encouraging the user agent to reveal all critical aspects of the LP problem. It ensures that no critical detail is left untouched, thus enhancing the completeness of the conversation.

3.2 Question Answering and Feedback Integration

The user agent is prompted to answer questions based on a predefined problem statement. To make the dialogue more natural and contextually grounded, the agent is configured to impersonate the individual mentioned in the original problem statement. This design choice enhances the genuineness of the interaction, creating a more realistic

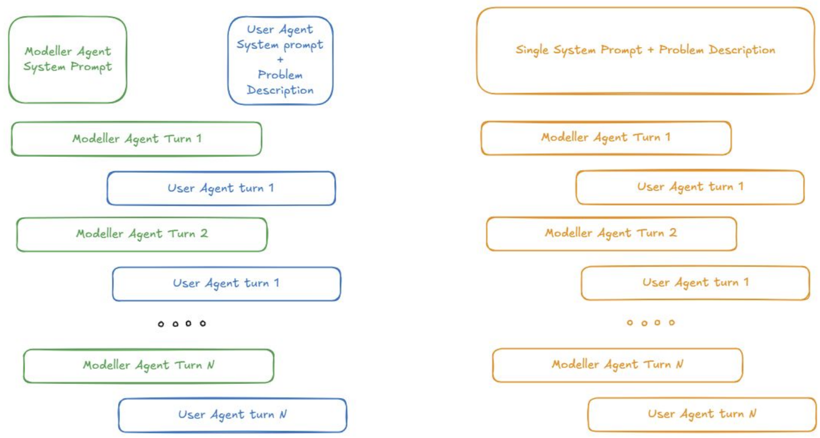

Figure 2: The diagram shows two different approaches. Dual agent method (left diagram) uses two different system prompts, and agents interact with each other. Single Completion method uses a single prompt and the entire conversation is generated at once.

user-agent dialogue experience. Also, a component within the user agent (henceforth the **comparing component**) compares the modelling agent's summary with the original problem statement using an LLM whenever a summary is generated. This comparison aims to identify mistakes in the generated summary and provide real-time feedback to the user agent, enabling the user agent to naturally correct and guide the modelling agent, emulating real-world interactions.

3.3 System Prompt Reinforcement

While the first system prompt initialises the dialogue, supplementary system prompts are injected at each turn. These prompts remind the LLM agents of specific instructions that enhance the dialogue's coherence and consistency.

4 Evaluation

To assess dialogues created using our two methods, we employed human annotators. We selected 25 problem descriptions from the NL4OPT dataset and generated two types of dialogues for each problem, resulting in a total of 50 dialogues. Five annotators were employed, with each dialogue evaluated by three annotators to ensure broader coverage and consistency. To ensure a balanced distribution, each annotator assessed an equal number of dialogues: a total of 30 dialogues, with 15 of each type covering 25 distinct problems. This distribution is detailed in Table 1.

The annotations were performed using the EZ-CAT tool (Guibon et al., 2022), which supports both turn-level and dialogue-level annotations.

For the turn-level annotations, we used: *Coherence, Fluency, Informativeness, and Quality*. For the dialogue-level annotations, we used *Overall Coherence, Modeller Agent Success, User Agent Success*. Detailed descriptions of the annotation criteria are provided in Appendix A.1.

The annotators were five computer science students who underwent a training session before starting the actual annotations. They were provided with detailed guidelines and received feedback during the training phase to ensure consistency in their evaluations.

The results of the human annotations were then analysed to compare the performance of the two dialogue generation methods. We calculated inter-rater reliability to understand the agreement among the annotators and used the scores to assess the overall quality of the dialogues generated by each method.

5 Results

5.1 Quality Of Human Annotations

To assess the quality and consistency of the annotations provided by our human annotators, we calculated pairwise Weighted Cohen's kappa coefficients for both dialogue-level and turn-level annotations. Weighted Cohen's kappa is a well-established statistical measure that evaluates inter-rater agreement (Cohen, 1968).

The pairwise Cohen's kappa results for turn-level annotations show varying levels of agreement among annotators, with overall low kappa values indicating limited consistency. This trend

Problem	R_1	R_2	R_3	R_4	R_5	Problem	R_1	R_2	R_3	R_4	R_5
1	2	2	1, 2	1	1	14	1	2	2	1, 2	1
2	2	2	1, 2	1	1	15	1	2	2	1, 2	1
3	2	2	1, 2	1	1	16	2	1, 2	1	1	2
4	2	2	1, 2	1	1	17	2	1, 2	1	1	2
5	2	2	1, 2	1	1	18	2	1, 2	1	1	2
6	1, 2	1	1	2	2	19	2	1, 2	1	1	2
7	1, 2	1	1	2	2	20	2	1, 2	1	1	2
8	1, 2	1	1	2	2	21	1	1	2	2	1, 2
9	1, 2	1	1	2	2	22	1	1	2	2	1, 2
10	1, 2	1	1	2	2	23	1	1	2	2	1, 2
11	1	2	2	1, 2	1	24	1	1	2	2	1, 2
12	1	2	2	1, 2	1	25	1	1	2	2	1, 2
13	1	2	2	1, 2	1						

Table 1: Distribution of dialogues among annotators. R_x represents rater x. Numbers in cells (1 and/or 2) represent the type of dialogue.

	R_1	R_2	R_3	R_4	R_5
Quality					
R_1	1.00				
R_2	-0.02	1.00			
R_3	-0.05	0.00	1.00		
R_4	0.11	0.39	-0.08	1.00	
R_5	0.01	0.29	0.01	0.25	1.00
Informativeness					
R_1	1.00				
R_2	0.01	1.00			
R_3	0.11	-0.05	1.00		
R_4	0.13	0.46	-0.10	1.00	
R_5	0.03	0.29	0.00	0.26	1.00
	R_1	R_2	R_3	R_4	R_5
Coherence					
R_1	1.00				
R_2	-0.16	1.00			
R_3	-0.01	0.01	1.00		
R_4	0.01	0.36	-0.00	1.00	
R_5	-0.03	0.17	0.00	0.26	1.00
Fluency					
R_1	1.00				
R_2	-0.03	1.00			
R_3	-0.02	-0.04	1.00		
R_4	-0.10	0.08	0.03	1.00	
R_5	-0.03	0.04	0.00	-0.06	1.00

Table 2: Pairwise Cohen's kappa for turn-level annotations. R_x represents rater x.

persists across individual dimensions of Quality, Informativeness, Coherence, and Fluency, as shown in Table 2. The low agreement may reflect the subjective nature of evaluating turn-level dialogue quality, where different annotators have varying interpretations of factors like coherence or fluency in individual dialogue turns.

The dialogue-level annotations also showed varying levels of agreement among annotators, though generally higher than the turn-level annotations. The kappa values suggest moderate agreement on certain dimensions, particularly Modeller Agent Success, User Agent Success, and Overall Coherence, as shown in Table 3.

Agreement between the human annotators across the dialogue rounds for a set of different problems from NL4OPT is assessed, and the overall scores are used to provide insights into the quality of the modelling dialogue. The annotations are used to identify areas for potential improvement in the dialogues and potential refinement of the modeller agent.

5.2 Quality Of The Generated Dialogues

Figure 4 shows an example of a dialogue generated by the system described in Section 3, and we can observe that, overall, the dialogue reads well. In fact, the results of the human annotations (Tables 4 and 5) show a relatively good quality of the dialogue turns with average scores above 4 across all criteria.

For turn-level annotations, both methods (dual-agent, single-completion) performed similarly. Both methods demonstrated strong coherence and fluency, with nearly identical scores, highlighting the effectiveness of our system in generating readable and coherent dialogues.

At a dialogue level, the modelling agent in a dual-agent method is rated successful, with a mean of 4.39 and a median of 5, indicating that the questions generated by the modeller are effective in eliciting the necessary information. However, the single-completion method, in which the LLM had access to the full problem description, consistently achieved higher scores in all categories, with less variability in the ratings. For example, single-completion dialogues scored a mean of 4.72 and a median of 5 for overall coherence. Interest-

	R_1	R_2	R_3	R_4	R_5	R_1	R_2	R_3	R_4	R_5	R_1	R_2	R_3	R_4	R_5
	Modeller Success					**User Success**					**Overall Coherence**				
R_1	1.00					1.00					1.00				
R_2	0.00	1.00				0.08	1.00				-0.05	1.00			
R_3	0.00	-0.03	1.00			0.57	0.51	1.00			-0.18	0.25	1.00		
R_4	0.00	0.59	0.20	1.00		0.00	0.51	0.14	1.00		-0.18	0.30	0.08	1.00	
R_5	0.00	0.65	0.59	0.30	1.00	0.22	0.58	0.36	0.06	1.00	0.01	0.75	0.45	0.35	1.00

Table 3: Pairwise Cohen's kappa for dialogue-level annotations. R_x represents rater x.

	Dual-Agent			Single-Completion		
	Mn	Md	Std	Mn	Md	Std
Quality	4.51	4.70	0.49	4.64	4.75	0.35
Informat.	4.44	4.60	0.51	4.54	4.64	0.36
Coherence	4.78	4.83	0.22	4.85	4.94	0.24
Fluency	4.83	4.93	0.26	4.80	4.93	0.30

Table 4: Annotation scores (Mean, Median and Standard Deviation) average across turns for turn-level criteria (Quality, Informativeness, Coherence, Fluency) over all problems in the test set on a scale of 1 to 5.

	Dual-Agent			Single-Completion		
	Mn	Md	Std	Mn	Md	Std
Overall Coh.	4.31	5.0	0.9	4.72	5.0	0.45
Modeller S.	4.39	5.0	0.91	4.65	5.0	0.55
User S.	4.21	4.0	0.83	4.63	5.0	0.52

Table 5: Annotation scores (Mean, Median and Standard Deviation) for dialogue-level criteria (Overall Coherence, Modeller Success, User Success) over all problems in the test set on a scale of 1 to 5.

ingly, single-completion dialogues are shorter on average, with 14.96 turns compared to 19.52 turns in the dual-agent method. These observations suggest that, while the dual-agent method performs well in generating meaningful dialogue under uncertainty, having access to full information (as in single-completion) naturally results in more coherent and well-structured conversations.

In the post-annotation process, we identified several types of errors made in the dialogue turns. This section summarises these errors. Whenever possible, we will use examples from Figure 4.

5.2.1 Text Too Casual

In general, the tone of the messages made both by the modelling agent and the user was very casual, and this might lead to vague or ambiguous statements, inaccuracies, or statements that are in poor English. For example, *The day for picking strawberries lasts 8000 minutes*. This sounds awkward, and in fact a day has only 1,440 minutes.

5.2.2 Uninformative First Message

The first message made by the modeller was often very vague, not very informative, and out of context; it might be misinterpreted. For example, in Figure 4: *Hello! I'm OptiMouse, here to help you make the best decision for your situation. Could you tell me a bit about what you're trying to achieve?* This could be misinterpreted by someone who does not know what the system (named OptiMouse) actually does.

5.2.3 Unclear Decision Variables

The user often failed to specify the exact nature of the decision variables, and the modeller did not follow up to clarify this ambiguity. For example, in Figure 4, the user said *We make and sell small, medium, and large mugs*, but would not say that the user wants to know what the *number* of each type of mug is required to be. Then, the modeller would never ask or confirm that the objective function must contain the number of mugs of each type.

5.2.4 Generic Questions

The modeller would sometimes ask questions that are too generic, or just not relevant to the particular optimisation problem. The user, then, would attempt to answer the question without pointing out that the question may not be relevant or helpful.

5.2.5 Complex Answers

The modeller would sometimes ask questions that elicit several constraints, not just one. The user would then give too much information. For example, in Figure 4, the question *Could you tell me how much glass and plastic is needed to make each size of mug?* leads to a complex answer by the user.

5.2.6 Fixation On Non-pertinent Topic

After receiving negative answers several times, the modeller would keep asking the same kinds of questions, even though it is apparent that these questions are not pertinent. Conversely, the user would not give helpful information to indicate that the modeller might be confused.

Investigating these errors is useful in relation to the prompt engineering used and described in Section 3.1. The modelling agent needs to introduce itself clearly, indicating the kind of modelling and expertise it brings. It is engaging to use easy-to-understand language, but the language should not be casual as modelling conversations have to be executed carefully and with precision. The modelling agent prompts encourage asking questions in a sequential manner, guiding the dialogue logically and thoroughly; however, this error analysis suggests that better guidance could be given here by explaining that the better directions for the next question may use clues from what has been given in the prior answers from the user agent. This may help to give direction and priority to the most pertinent questions. Consistent with the errors noted, the modelling agent rarely reminds the user to answer simply, providing one piece of information, rather than giving several model components in one response. The inappropriate user answers suggest that the user should be more strongly encouraged to answer the direct question, but if there is no relevant answer to this, then it should be helpful and provide the closest relevant answer. Fixation on a non-pertinent topic may also see improvement through prompting that encourages the modelling agent to reflect on the answers provided so far in the conversation, as well as prompting the user agent to be helpful in reminding the modelling agent of the objective and the decision variables.

6 Conclusion And Future Work

Mathematical modelling proceeds through dialogues between a modeller and a user with a business problem. This paper uses the NL4Opt dataset of optimisation problems to study simulated conversations between a modelling agent and a user agent. In a dual-agent variant, only the user agent has the problem description. A baseline variant uses a single-completion method, where an LLM generates the dialogue in one pass, with both conversants having full access to the problem description.

Human evaluations reveal insights into each method's strengths, showing that the modelling agent in the dual-agent system performs as successfully as in the baseline system. Despite knowing less about the problem description, it effectively gathers necessary information. However, dialogue scores vary more, especially in coherence and success of the agents, in the dual-agent system compared to the single-completion method. The dual-agent approach performed well overall, scoring positively in terms of dialogue quality. This variability may stem from the single-completion method, where the LLM controls both questions and answers, creating a more consistent dialogue structure.

Human annotations show that the dual-agent system effectively elicits crucial information, despite the modeller's lack of prior knowledge about the problem. However, these findings must be interpreted with an important limitation. The low inter-rater reliability raises questions about the consistency of human evaluation scores and limits our confidence in drawing strong comparative conclusions. Notably, while the overall kappa values are low, annotators still gave high scores across all criteria. This suggests the dialogues are of high quality, allowing us to answer positively the central question: "Can an LLM elicit information from users?". The negative kappa values are close to zero, indicating that agreement is close to random. Despite this limitation, the high scores suggest the dialogues were of sufficient quality to provide initial evidence that LLMs can engage in meaningful information elicitation for optimisation modelling, warranting further study with better evaluation methods and a broader scope.

Future research directions include expanding beyond linear programming to more complex optimisation problems, generating dialogues with real human users rather than synthetic agents, and developing more sophisticated evaluation metrics that better capture the nuances of information elicitation quality. Additionally, investigating domain-specific prompt engineering techniques and exploring the capabilities of newer language models could further enhance the effectiveness of LLM-based optimisation dialogue systems. Finally, we plan to incorporate the ability to generate the optimisation model and run an LP solver, so that we can assess the ability of the system to solve the optimisation problem.

Ethical Considerations And Limitations

The base problems used for the generation of the dialogues in this paper are publicly available at the NLP4Opt Competition (Ramamonjison et al., 2023). To our knowledge, these base problems were generated ethically. These consist of simple LP problems from the NLP4Opt dataset; future work will address more complex types.

Our dialogues were not automatically checked for toxicity, but visual inspection of a small sample revealed none.

A limitation is the low agreement among the human annotators. However, each annotator's high scores indicate the dialogues are of good quality.

Additionally, it is important to note that the experiments described in this paper employed the 2023 version of GPT-4. More advanced Large Language Models (LLMs) were not evaluated due to the significant costs associated with human evaluation. Nevertheless, newer LLMs likely possess greater capability to elicit information from users.

Although employing LLMs to generate synthetic dialogues provides advantages in terms of control and scalability, the use of an LLM as the "user agent" may not adequately capture the complexity, unpredictability, and diversity of real human users. Synthetic users, while consistent and controllable, lack the uncertainty, emotional responses, varied knowledge, and communication patterns of real human interactions in optimisation modelling.

Acknowledgements

This work was partially funded by the Australian Research Council, Australia through the Discovery Project 2022 (grant number DP220101925).

References

Yelaman Abdullin, Diego Molla, Bahadorreza Ofoghi, John Yearwood, and Qingyang Li. 2023. Synthetic dialogue dataset generation using LLM agents. In *Proceedings of the Third Workshop on Natural Language Generation, Evaluation, and Metrics (GEM)*, pages 181–191, Singapore. Association for Computational Linguistics.

Dimitris Bertsimas and John N Tsitsiklis. 1997. *Introduction to linear optimization*, volume 6. Athena scientific Belmont, MA.

Jacob Cohen. 1968. Weighted kappa: Nominal scale agreement provision for scaled disagreement or partial credit. *Psychological bulletin*, 70(4):213.

Rasul Esmaeilbeigi, Vicky Mak-Hau, John Yearwood, and Vivian Nguyen. 2022. The multiphase course timetabling problem. *European Journal of Operational Research*, 300(3):1098–1119.

Ryan Fellows, Hisham Ihshaish, Steve Battle, Ciaran Haines, Peter Mayhew, and J.Ignacio Deza. 2022. Task-oriented dialogue systems: Performance vs quality-optima, a review. In *Computer Science and Information Technology*. Academy and Industry Research Collaboration Center (AIRCC).

Angus Gaudry, Ryan Li, and Vicky Mak-Hau. 2023. FlexBeamOpt: Hybrid solution methodologies for high-throughput GEO satellite beam laydown and resource allocation. *International Journal of Satellite Communications and Networking*.

Ga"el Guibon, Matthieu Labeau, Luce Lefeuvre, and Chlo'e Clavel. 2022. Ezcat: an easy conversation annotation tool. In *Proceedings of the 13th Language Resources and Evaluation Conference*, Marseille, France. European Language Resources Association.

Kevin Hogan, Annabelle Baer, and James Purtilo. 2021. Diplomat: A conversational agent framework for goal-oriented group discussion. In *International Conference on Group Decision and Negotiation*, pages 143–154. Springer.

Joey Hong, Sergey Levine, and Anca Dragan. 2023. Zero-shot goal-directed dialogue via rl on imagined conversations. *Preprint*, arXiv:2311.05584.

Edward Lam and Vicky Mak-Hau. 2020. Branch-and-cut-and-price for the cardinality-constrained multi-cycle problem in kidney exchange. *Computers Operations Research*, 115:104852.

Sungjin Lee, Qi Zhu, Ryuichi Takanobu, Zheng Zhang, Yaoqin Zhang, Xiang Li, Jinchao Li, Baolin Peng, Xiujun Li, Minlie Huang, and Jianfeng Gao. 2019. ConvLab: Multi-domain end-to-end dialog system platform. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 64–69, Florence, Italy. Association for Computational Linguistics.

Oliver Lemon, Dilek Hakkani-Tur, Junyi Jessy Li, Arash Ashrafzadeh, Daniel Hernández Garcia, Malihe Alikhani, David Vandyke, and Ondřej Dušek, editors. 2022. *Proceedings of the 23rd Annual Meeting of the Special Interest Group on Discourse and Dialogue*. Association for Computational Linguistics, Edinburgh, UK.

Yen-Ting Lin and Yun-Nung Chen. 2023. LLM-eval: Unified multi-dimensional automatic evaluation for open-domain conversations with large language models. In *Proceedings of the 5th Workshop on NLP for Conversational AI (NLP4ConvAI 2023)*, pages 47–58, Toronto, Canada. Association for Computational Linguistics.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. G-eval: NLG evaluation using gpt-4 with better human alignment. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2511–2522, Singapore. Association for Computational Linguistics.

Vicky Mak-Hau, Brendan Hill, David Kirszenblat, Bill Moran, Vivian Nguyen, and Ana Novak. 2021. A simultaneous sequencing and allocation problem for military pilot training: Integer programming approaches. *Computers Industrial Engineering*, 154:107161.

Sara Mirabi, Bahadorreza Ofoghi, John Yearwood, Diego Molla Aliod, and Vicky Mak-Hau. 2025. Investigating answer validation using noise identification and classification in goal-oriented dialogues.

OpenAI. 2023. GPT-4 technical report. *ArXiv*, abs/2303.08774.

Baolin Peng, Michel Galley, Pengcheng He, Chris Brockett, Lars Liden, Elnaz Nouri, Zhou Yu, Bill Dolan, and Jianfeng Gao. 2022. Godel: Large-scale pre-training for goal-directed dialog. *arXiv preprint arXiv:2206.11309*.

Rindra Ramamonjison, Haley Li, Timothy Yu, Shiqi He, Vishnu Rengan, Amin Banitalebi-dehkordi, Zirui Zhou, and Yong Zhang. 2022. Augmenting operations research with auto-formulation of optimization models from problem descriptions. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 29–62, Abu Dhabi, UAE. Association for Computational Linguistics.

Rindranirina Ramamonjison, Timothy Yu, Raymond Li, Haley Li, Giuseppe Carenini, Bissan Ghaddar, Shiqi He, Mahdi Mostajabdaveh, Amin Banitalebi-Dehkordi, Zirui Zhou, and 1 others. 2023. Nl4opt competition: Formulating optimization problems based on their natural language descriptions. In *NeurIPS 2022 Competition Track*, pages 189–203. PMLR.

Donya Rooein, Devis Bianchini, Francesco Leotta, Massimo Mecella, Paolo Paolini, and Barbara Pernici. 2022. aCHAT-WF: Generating conversational agents for teaching business process models. *Software and Systems Modeling*, pages 1–24.

Weiyan Shi, Kun Qian, Xuewei Wang, and Zhou Yu. 2019. How to build user simulators to train rl-based dialog systems. *arXiv preprint arXiv:1909.01388*.

Stefan Ultes, Lina M Rojas Barahona, Pei-Hao Su, David Vandyke, Dongho Kim, Inigo Casanueva, Paweł Budzianowski, Nikola Mrkšić, Tsung-Hsien Wen, Milica Gasic, and 1 others. 2017. Pydial: A multi-domain statistical dialogue system toolkit. In *Proceedings of ACL 2017, System Demonstrations*, pages 73–78.

Mina Valizadeh and Natalie Parde. 2022. The AI doctor is in: A survey of task-oriented dialogue systems for healthcare applications. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6638–6660.

Jason D Williams, Matthew Henderson, Antoine Raux, Blaise Thomson, Alan Black, and Deepak Ramachandran. 2014. The dialog state tracking challenge series. *AI Magazine*, 35(4):121–124.

A Annotation Criteria

A.1 Turn-Level Annotations

At the turn level, the following criteria were used to assess turns:

- **Coherence:** Assess if each turn logically connects with previous and subsequent turns. The turn should continue or respond logically to the topic or question. Check for consistency in details mentioned across turns. Rate Coherence for a given turn from 1 to 5, where 1 is low coherence, and 5 is high coherence.

- **Fluency:** Evaluate the linguistic clarity of each turn. It should use clear, concise language and avoid overly complex or technical terms unless necessary. Rate Fluency for a given turn from 1 to 5, where 1 represents low fluency, and 5 represents high fluency.

- **Informativeness:** Determine the value added by each turn to the dialogue. The turn should provide relevant information, clarify doubts, or advance the conversation meaningfully. Avoid redundant or irrelevant information. A turn should get a low score if it appears redundant in a given context.

- **Quality:** Look at the appropriateness and effectiveness of each turn. It should respond accurately to the preceding turn, provide valid information, and be suitable for the dialogue's context. The turn should contribute to achieving the conversation's objective. If the question cannot be asked due to limited information in the conversation, the agent should get a lower score.

A.2 Dialogue-Level Annotations

At the dialogue level, the following criteria were uses to assess entire dialogues:

- **Overall Coherence:** This criterion assesses the logical flow of the entire dialogue. Check for a structured conversation where each part connects to the next, maintaining thematic and topical continuity. Ensure the dialogue forms a coherent narrative from the introduction of the problem to the conclusion.

- **Modelling Agent Success:** Evaluate how successfully the agent (OptiMouse) performs as an optimisation helper. This includes its effectiveness in providing relevant, accurate advice, understanding and addressing the user's queries, and eliciting information for practical solutions. Overall, how well does the agent do the job of a modeller?

- **User Agent Success:** Evaluate how well the user agent performs through the dialogue. This involves determining whether the user agent provides accurate information based on the problem description and responds to the agent's queries. Overall, how good (and natural) is the user agent at impersonating a business person mentioned in the MILP problem description?

B Examples

HauMak is a mug company and they make mugs of three sizes: small, medium, and large. With each small mug, they can make a profit of $2, with each medium mug, $3, and with each large mug, $4. Each small mug needs 10oz of glass and 2 units of plastic; each medium mug requires 12 oz of glass and 3 units of plastic; each large mug requires 15 oz of glass and 3.5 units of plastic. There are 1000 oz of glass and 835 units of plastic available. HaMak would of course like to maximise their profit, so they have to decide how many mugs of each size to make in order to achieve the goal.

Figure 3: The problem description used to generate the answers.

Modeller: *Hello! I'm OptiMouse, here to help you make the best decision for your situation. Could you tell me a bit about what you're trying to achieve?*
User: *Hello! I own HaMak, a mug company. I want to maximise our profit.*
Modeller: *That's great! To help you with that, could you tell me what types of mugs you sell?*
User: *We make and sell small, medium, and large mugs.*
Modeller: *That's wonderful! Could you tell me how much profit you make from selling each size of mug?*
User: *We make a profit of $2 on each small mug, $3 on each medium mug, and $4 on each large mug.*
Modeller: *That's good to know. Now, is there a limit to how many mugs you can make and sell in a given period?*
User: *The number of mugs we can make is limited by the available materials: 1000 oz of glass and 835 units of plastic.*
Modeller: *I see. Could you tell me how much glass and plastic is needed to make each size of mug?*
User: *Each small mug needs 10oz of glass and 2 units of plastic. Medium mugs require 12 oz of glass and 3 units of plastic. Large mugs require 15 oz of glass and 3.5 units of plastic.*
Modeller: *That's helpful. Now, are there any other restrictions or rules that you need to follow when making and selling your mugs?*
User: *No, there are no other restrictions or rules beyond the material limitations.*
Modeller: *Great! Let me summarize to make sure I've got everything:*
- You want to maximise the profit from selling mugs.
- You sell three types of mugs: small, medium, and large.
- You make a profit of $2 on each small mug.
- You make a profit of $3 on each medium mug.
- You make a profit of $4 on each large mug.
- Each small mug requires 10oz of glass and 2 units of plastic.
- Each medium mug requires 12 oz of glass and 3 units of plastic.
- Each large mug requires 15 oz of glass and 3.5 units of plastic.
- You have at most 1000 oz of glass and 835 units of plastic available for making mugs.

Does this cover everything?

User: *Yes, that's correct! You've summarized our situation perfectly. Thank you for your help. Goodbye!*

Figure 4: Example of generated conversation between a modeller and a user using the dual-agent method.

C LLM Prompts

C.1 Modeller Agent Prompt (dual agent method):

```
YOU ARE "OptiMouse" - A CHATBOT HELPING USERS TO
      FORMULATE FULL OPTIMIZATION PROBLEM
     STATEMENT.
THE USER IS NOT A MATH EXPERT AND HAS NO
     EXPERIENCE WITH MATH AND OPTIMIZATIONS.
DO NOT USE ANY MATHEMATICAL TERMINOLOGY OR
     EXPLANATIONS SUCH AS OBJECTIVE FUNCTION,
     CONSTRAINTS, ETC.

GATHER NECESSARY DETAILS THAT CAN BE MAPPED TO A
      LINEAR PROGRAMMING MODEL.
ENGAGE USERS BY ASKING CLEAR, CONCISE, AND
     SEQUENTIAL QUESTIONS TO RECEIVE INFORMATION
     ABOUT CONSTRAINTS AND OBJECTIVE FUNCTION.
ASK A QUESTION BASED ON THE PREVIOUS INFORMATION
      THAT WILL LEAD TO GETTING A CONSTRAINT OR
     OTHER PARAMETER OF THE MODEL.
THINK DEEPLY SO YOU WILL BE ABLE TO GET FULL
     PROBLEM DETAILS.
ONE QUESTION ALLOWED PER MESSAGE.

PROVIDE A SUMMARY IN BULLET POINTS (SEE EXAMPLE
     DELIMITED BY "====") ONCE YOU HAVE ALL THE
     INFORMATION NEEDED
DO NOT INCLUDE UNKNOWN/NON-FACTUAL CONSTRAINTS
     IN A SUMMARY(For example, 'There's no
     specific requirement on X...', 'There's no
     limit on X...' )
ASK A CLARIFICATION QUESTION BEFORE PROVIDING A
     SUMMARY TO MAKE SURE YOU HAVE ALL THE
     CONSTRAINTS AND AN OBJECTIVE FUNCTION.

EXAMPLE OF A SUMMARY:

====
- A coconut seller has to transport coconuts
     using either rickshaws or ox carts.
- The rickshaws can take 50 coconuts each and
     cost $10 per trip.
- The ox carts can take 30 coconuts each and
     cost $8 per trip.
- The seller has at most $200 to spend on
     transporting the coconuts.
- The number of rickshaws must not exceed the
     number of ox carts.
====
```

```
START THE CONVERSATION WITH A FRIENDLY GREETING,
     INTRODUCING YOURSELF AND ASKING WHAT THE
    USER WOULD LIKE TO OPTIMISE.
```

C.2 User Agent Prompt (dual agent method):

```
YOU ARE AGENT IMPERSONATING THE BUSINESS OWNER
    MENTIONED IN THE PROBLEM STATEMENT(DELIMITED
     BY ```).
BE POLITE.
YOU(THE BUSINESS OWNER) ARE TALKING WITH AN
    EXPERT IN OPTIMIZATIONS.
ACCURATELY PROVIDE INFORMATION AS REQUESTED
    BASED ON THE PROBLEM STATEMENT.
MAKE SURE INFORMATION YOU PROVIDING IS CORRECT
    AND CAN BE FOUND IN THE PROBLEM STATEMENT.
IF THE PROBLEM STATEMENT DOES NOT CONTAIN
    REQUESTED INFORMATION, SIMPLY SAY YOU DON'T
    KNOW THESE DETAILS. (for example, "I'm not
    sure about it, can we skip this")
DO NOT MAKE CALCULATIONS OR INFORMATION
    MANIPULATING. Use facts from the problem (
    for example, question: How many X are
    produced in a day? Answer: I'm not sure, but
     I know that to produce one X, we need Y
    minutes.)
DO NOT MENTION THE PROBLEM STATEMENT ANYWHERE;
    ACT AS IF IT IS YOUR PERSONAL KNOWLEDGE.

THE PROBLEM STATEMENT:
```
{0}
```

START THE CONVERSATION WITH A WARM GREETING
```

C.3 Dialogue Generation Prompt (single completion method)

````
Generate a dialogue between Optimouse(an AI
    Optimization/MILP helper) and a business
    owner from a problem description.
Problem description:
```
{0}
```
The business owner is not familiar with math/
    optimisation and its terminology.
Optimouse should perform information elicitation
     to figure out the business owners problem
    and its details by asking questions and
    engaging in dialogue with the business owner
    .
The business owner should limit avoid revealing
    too much information at once and should
    provide information only in response to
    Optimouse's questions.
Finally, Optimouse gives a summary of the
    problem at the end of dialogue.
Optimouse starts the conversation with a
    friendly greeting, introduction and asking
    what the business is.
EXAMPLE OF A SUMMARY:
====
- You want to maximise the number of coconuts
    that can be transported. # clearly stated
    objective function, always 1st in the
    summary
- You transport coconuts using either rickshaws
    or ox carts. # 2 clearly described decision
    variable names, always 2nd in the summary
- A rickshaw can take 50 coconuts. # separate
- A rickshaw cost $10 per trip. # separate
- An ox cart can take 30 coconuts each. #
    separate
- An ox cart cost $8 per trip. # separate
- You have at most $200 to spend on transporting
     the coconuts. # separate
- The number of rickshaws must not exceed the
    number of ox carts. # separate
====

Format dialogue as a JSON array with key "
    dialog_messages". Where each message has two
     fields "role"(client or agent) and "content
    "
````