Optimization Modeling and Verification from Problem Specifications using a Multi-agent Multi-stage LLM Framework

38th Annual AAAI Conference on Artificial Intelligence

Mahdi Mostajabdaveh[1], Timothy TL Yu[1], Rindra Ramamonjison[1], Giuseppe Carenini[2], Zirui Zhou[1], and Yong Zhang[1]

1. Huawei Technologies Canada
2. University of British Columbia

HUAWEI

UBC

Introduction

1. Operations Research (OR) improves efficiency in many fields but is hindered by its time-consuming and complex nature.
2. OR projects need specialists to transform problem details into mathematical models and then code, which is stored in two formats:
3. Can Large Language Models (LLMs) translate natural language descriptions of an optimization problem into an optimization model (NL2OPT) ?

Our contribution :

- Novel multi-agent modeling framework that
  - leverages relations identifier agents
  - multi-agent verification mechanism,
  - Independent from optimization solvers execution.
- Straightforward and practical evaluation framework
- Test and evaluate different LLM prompting approaches, providing insights about their effectiveness in enhancing accuracy

NL2OPT Task definition

Translating natural language descriptions into optimization models (**NL2OPT** for short)

- **Input:** Problem description in natural language, whether structured or unstructured.
- **Output**: Mathematical formulation in LaTeX or modeling code.

Please see an example of our dataset in the following section.

Description

Due to the recent surge in COVID-19 cases centered around Yellow Park, as the city's transportation manager, you have been asked to create a plan. The Circle Plaza station, a bustling hub close to multiple tourist attractions, is a region at risk of becoming a super-spreader site due to its high amount of foot traffic. To prevent the spread of the virus from Yellow Park to the Circle Plaza station, the city has decided to temporarily shut down some subway lines. This was proposed to ensure that no subway route exists between Yellow Park and Circle Plaza. While public health is important, shutting down subway lines has its repercussions. There is an impact on city finances. Specifically, impacting ticket sales, staffing, and operational costs. Therefore, you need to make a decision that ensures public safety while also being economically feasible. You have estimated the financial impact associated with closing each track. Your task is to identify which subway routes to shut down, minimizing the overall costs, while ensuring that travellers cannot travel by subway between Yellow Park and the Circle Plaza.

Problem Specifications

- Set of subway stations
- Set of subway routes each connecting two sequential stations
- The cost to shut down a subway route
- Decision to close a specific subway route between a pair of stations
- Decision to include a specific station in the set of stations remain connected to Yellow Park station
- Minimize the cost of closing subway routes to make sure that people cannot travel by subway between Yellow Park and Circle Plaza.
- The Yellow Park station (the Source) is definitely included in the set of stations on the Yellow Park station side, and Circle Plaza station (the Sink) is not.
- If the subway route between two specific stations is closed, then the head station must be included in the set of stations on the Yellow Park side and the tail station is not.

Model Code in Zimpl

```
# Sets:
## Set of subway stations
set STATION;
## Set of subway routes each connecting two sequential stations
set ROUTE[STATION, STATION];
# Parameters:
## The cost to shut down a subway route
param cost[ROUTE];
# Variables:
## Decision to close a specific subway route between a pair of stations
var Z[ROUTE] binary;
## Decision to include a specific station in the set of stations remain connected to
Yellow Park station
var Y[STATION] binary;
# Objective:
## Minimize the cost of closing subway routes to make sure that people cannot travel
by subway between Yellow Park and Circle Plaza.
minimize obj: sum <i, j> in ROUTE do cost[i, j] * Z[i, j];
# Constraints:
## The Yellow Park station (the Source) is definitely included in the set of stations
on the Yellow Park station side, and Circle Plaza station (the Sink) is not.
subto source_sink: Y[first(STATION)] - Y[last(STATION)] >= 1;
## If the subway route between two specific stations is closed, then the head
station  must be included in the set of stations on the Yellow Park side and the tail
station is not.
subto route_cut: forall <(i, j)> in ROUTE do Z[i, j] - Y[i] + Y[j] >= 0;
```

Problem Specification

- **Elements**:These encompass the actors within the system
- **Data Parameters:** Related elements and define the input data to the system.
- **Decision Activities:** These are direct actions within the system requiring decision-making, associated with specific entities.
- **Calculations:** Representing auxiliary variables in optimization problems, calculations derive their values from other decisions.
- **Objective Criterion:** Outlines the goal or metric to be minimized or maximized by the business owner.
- **Requirements:** These are the regulations or limitations that must be adhered to in the optimization problem, defining the conditions for an acceptable solution.

NL2OPT Difficulty Criteria

Problem Description		Modeling Complexity	
Characteristics	**Description**	**Characteristics**	**Description**
Problem Name	Whether explicitly mentioning the name of a known optimization problem.	Individual Variables	Whether if variables or parameters are defined individually
All Components	Whether all the sets, parameters, variables, constraints and objectives are defined explicitly with keywords identifying them.	Sets	Whether sets are required to model the problem
Components Relations	Whether the relationship between the model components are defined explicitly.	Number of variables and parameters	The number of variables types and parameters in the model
Implicit components	Whether implicit model components such as calculations and structural constraints are defined explicitly or not.	Number of constraint	The number of constraint types in the model
Data Abstraction	Whether the data parameters and sets have explicit numerical values or not.	Number of objectives	Whether the model is single objective or multi-objective.
Math symbols	Whether math symbols are used in the description	Indices with condition	Whether there are conditions on indices for constraints and summations.
OR jargons	Whether OR terms are used or not	Logical constraints	Constraints that define the logical relations between variables
Generic or Specific	Whether the description is generic or have a specific context		

Table 1: Criteria for categorizing Optimization problem for NL2OPT task input respect to the description ambiguity and modeling complexity

Our Dataset

Dataset contains 70 instances design and verified by OR experts

Description Abstraction

- No problem name,
- No OR jargon,
- No numerical value,
- No math symbols
- Specific Context

Complexity of the Mathematical Model

- LP, MILP, QP
- Numbers of Set and Variable <= 5
- Number of Parameters and constraints <= 8
- Covering 15 application domain

Multi-agent LLM framework

Relations Identifier:
Identifies the parameters and variables that should participate in the expression of each constraint or objective.

Consistency Verifier:
Checking the consistency of the generated model.

Relations Verifier:
Use the relations extracted by the identifier to check the generated model.

Indices Verifier:
Ensuring that the logical structure of constraints is sound and the indices are correct.

Meaning Alignment Verifier:
Verifying that the generated objectives and constraints align with the natural language input.

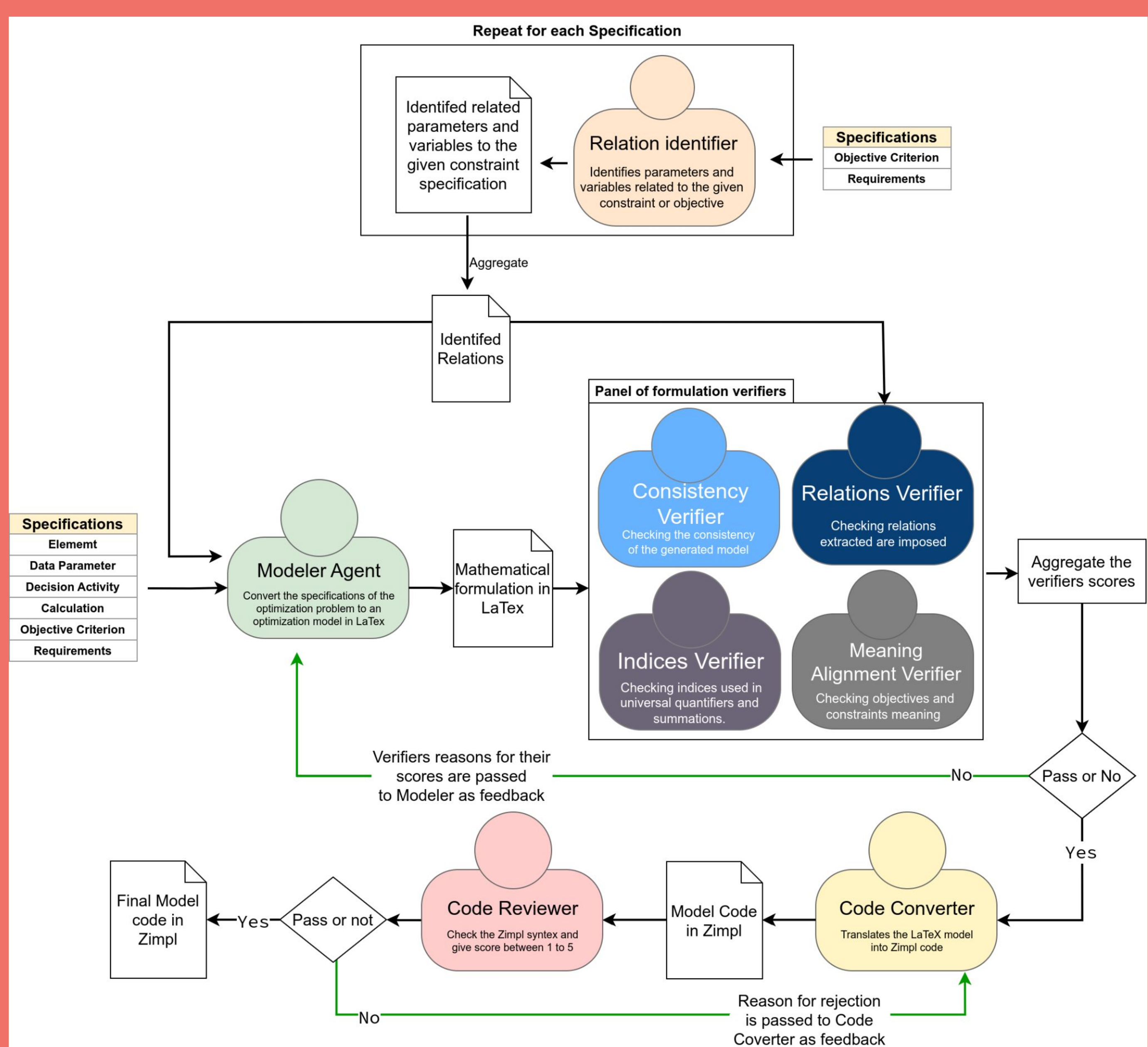

Figure 1: How the LLM agents interacting with each other .

Evaluation Pipeline

To evaluate the generated mathematical model and code we proposed a mutli step human involved evaluation

- Step1 : Embadening based model component matching
- Step2: Embadening based model expression matching
- Step3: Manual evaluation : The components that are flagged for inequality or parsing issues are passed to a human expert for review.
- Evaluation metric: Component based F1 Score

Experimental results

We compare our approach on 70 real-world optimization problem with prompting LLMs directly with popular prompting approaches .

				Component Exact-match Accuracy					
Strategy	Multi-turn	Spec input	Desc input	Set	Param	Var	Obj	Constraint	Avg
DESC2MODEL			✓	0.821	0.633	0.448	0.200	0.108	0.529
SPEC2MODEL		✓		**1.000**	0.889	**0.829**	0.586	0.472	0.747
MULTI-TURN	✓	✓		**1.000**	0.832	0.770	0.500	0.426	0.712
MULTI-TURN + DESC	✓		✓	**1.000**	**0.893**	0.789	0.600	0.458	0.751
MULTI-TURN + SPEC	✓	✓		**1.000**	0.881	0.789	0.571	0.463	0.746
OUR APPROACH		✓		**1.000**	0.873	0.786	**0.804**	**0.689**	**0.808**

Conclusion and Discussion

Structured Input Significantly Impacts Accuracy

This improvement can be attributed to the structured nature of problem specifications, Notably, this achievement is realized even though the specifications exclude mathematical symbols, explicit OR keywords, problem names, and are tailored to specific contexts.

LLMs have better accuracy when they have a comprehensive view of the optimization problem

In the MULTI-TURN method, we provide only one specification to the LLM at a time, limiting its ability to perceive the optimization problem in its entirety, which in turn affects accuracy.

Using several task-specific LLM verifiers with a feedback mechanism can increase accuracy

The difference between the third and second rows in Table 4 illustrates that having specialized verifiers can significantly enhance the accuracy of NL2OPT for more complex model components (e.i., objectives and constraints).

Limitation and Future Work

- Our framework can be enhanced to focus on generating not only accurate but also efficient optimization models, moving beyond merely ensuring correctness which can be achieved by training models to determine which generated optimization models are suitable for generic optimization solvers.
- Additionally, our dataset can be expanded to encompass more complex optimization problems or to have more ambiguous problem descriptions as input.

References

1. AhmadiTeshnizi, A.; Gao, W.; and Udell, M. 2023. OptiMUS: Optimization Modeling Using mip Solvers and large
2. language models. arXiv preprint arXiv:2310.06116
3. Beldiceanu, N.; and Simonis, H. 2012. A model seeker: Extracting global constraint models from positive examples. In International Conference on Principles and Practice of Constraint Programming, 141–157. Springer.
4. Nijkamp, E.; Pang, B.; Hayashi, H.; Tu, L.; Wang, H.; Zhou, Y.; Savarese, S.; and Xiong, C. 2022. Codegen: An open large language model for code with multi-turn program synthesis. arXiv preprint arXiv:2203.13474
5. Ramamonjison, R.; Yu, T.; Li, R.; Li, H.; Carenini, G.; Ghaddar, B.; He, S.; Mostajabdaveh, M.; BanitalebiDehkordi, A.; Zhou, Z.; et al. 2022b. NL4Opt Competition: Formulating Optimization Problems Based on Their Natural Language Descriptions. In NeurIPS 2022 Competition Track, 189–203. PMLR