OptiMUS: Scalable Optimization Modeling with (MI)LP Solvers and LLMs*

Presented by Ravi B. Sojitra @ Stanford's REFORM Reading Group

Authors: Ali AhmadiTeshnizi, Wenzhi Gao, Madeleine Udell

*Actually, I am presenting the newer version by
Ali AhmadiTeshnizi , Wenzhi Gao, Herman Brunborg,
Shayan Talaei, Connor Lawless, and Madeleine Udell.

Summary: AI + hard coded workflow goes far.

To increase adoption of optimization modeling, they want to remove frictions related to tasks that are cycled through during problem formulation. For example,

1. Identifying parameters, constraints, and objectives from documentation and data;
2. Constructing formal mathematical models;
3. Reformulating the models so they may be solved efficiently;
4. Selecting solvers; and
5. Generating code.

To address these, they contribute a framework for AI agentic systems and evaluation data.

OptiMUS 0.3 is an example instantiation of this framework.

Real Example: McDonald’s China used an MILP.

Tang and Wang et al. (2025) designed a supply chain network to reduce costs and emissions by formulating a Mixed Integer Linear Program (MILP) that commercial solvers can handle.

informs
https://pubsonline.informs.org/journal/ijaa

INFORMS JOURNAL ON APPLIED ANALYTICS
Vol. 55, No. 1, January–February 2025, pp. 36–47
ISSN 2644-0865 (print), ISSN 2644-0873 (online)

THE FRANZ EDELMAN AWARD
Achievement in Operations Research

McDonald’s China Adopts Operations Research for Network Design

Shouwei Tang,[a] Lei Wang,[a,*] Yun Shi,[b] Andy Li,[b] Kevin Lin,[c] Chen Xiang,[a] Sophia Niu,[a] Shaofeng Zhou,[a] Ming Liu,[c] Hank Tang[c]

[a] Optimization Analytics Technology Pte. Ltd., Singapore 199591; [b] McDonald’s China, Shanghai 201103, China; [c] Xianhui Logistics, Shanghai 201801, China

*Corresponding author

Contact: shouwei.tang@oatservice.com (ST); wendy.wang@oatservice.com, http://orcid.org/0009-0006-3838-4953 (LW); jim.shi@cn.mcd.com (YS); andy.li@cn.mcd.com (AL); kevin.lin@havi-cn.com (KL); chen.xiang@oatservice.com (CX); sophia.niu@oatservice.com (SN); shaofeng.zhou@oatservice.com (SZ); ming.liu@havi-cn.com (ML); hank.tang@havi-cn.com (HT)

Real Example: McDonald’s China used an MILP.

Tang and Wang et al. (2025) designed a supply chain network to reduce costs and emissions by formulating a Mixed Integer Linear Program (MILP) that commercial solvers can handle.

	Description
Sets and indices	
I	Set of all potential sites and existing DC locations with index i
J	Set of all existing DCs with index j, a subset of I
F	Set of factories (suppliers) with index f
D	Set of demand points with index d
SKU	Set of SKU groups (aggregated SKUs) with index s
T	Set of time periods (years) with index t
P	Set of storage areas in a DC with index p
K	Set of integer numbers starting from one, with index k
Parameters[a]	
α_i	One-time building investment cost for DC i in RMB yuan
β_i	Fixed operational cost per time period for DC i in RMB yuan
$\gamma_{d,t,s}$	Demand of demand point d per time period t for SKU group s in cases
$\delta_{f,t,s}$	Supply capacity of factory f per time period t for each SKU group s in cases
ε_s	Storage area of SKU group s in cases
$\zeta_{i,t,p}$	DC volume capacity of storage area p in DC i per time period t in cases
η_p	Unit storage cost per day for storage area p in RMB yuan
$\theta_{i,s}$	Per-unit handling cost for SKU s in DC i in RMB yuan
ι_s	Current safety stock cost in number of days of demand for SKU s
κ_S	Current cycle stock level in number of days of demand for SKU s
$\lambda_{f,i,s}$	Unit inbound transportation cost from factory f to DC i for SKU s in RMB yuan
$\mu_{i,d,s}$	Unit outbound transportation cost from DC i to demand point d for SKU s in RMB yuan
v	Number of days in a time period.
ξ	Number of DCs in baseline setting (i.e., number of McDonald’s China DCs in 2020)
$\pi_{d,t,s}$	The ratio (percentage) between SKU demand volume and total demand volume for each demand point d, time period t, and SKU s
Decision variables	
$x_{i,t}$	Binary variable, one if DC i will be opened at beginning of time period t, and zero otherwise
$y_{i,t}$	Binary variable, one if DC i will be closed at beginning of time period t, and zero otherwise
$z_{i,t}$	Binary variable, one if DC i exists during time period t, and zero otherwise
n_t	Integer variable, total number of DCs
$m_{k,t}$	Binary variable, one if the number of DCs in time period t is k
$if_{f,i,s,t}$	Nonnegative continuous variable, inbound flow volume from factory f to DC i of SKU s in year t
$of_{i,d,t}$	Nonnegative continuous variable, outbound flow volume from DC i to demand point d of SKU s in year t
$ss_{s,t}$	Nonnegative continuous variable, total safety stock volume for SKU s in year t
$cs_{s,t}$	Nonnegative continuous variable, total cycle stock volume for SKU s in year t

[a] The exchange rate from USD to RMB yuan is 1:7.12 as of this writing.

Objective Function

The mathematical model’s objective function is to minimize the total cost, which comprises the following components:

$$z_1 = \sum_{f\in F, i\in I, s\in SKU, t\in T} if_{f,i,s,t} * \lambda_{f,i,s}$$

$$z_2 = \sum_{i\in I, d\in D, s\in SKU, t\in T} of_{i,d,t} * \mu_{i,d,s} * \pi_{d,t,s}$$

$$z_3 = \sum_{f\in F, i\in I, s\in SKU, t\in T} if_{f,i,s,t} * \theta_{i,s}$$

$$z_4 = \sum_{s\in SKU, t\in T} (ss_{s,t} + cs_{s,t}) * \eta_{\varepsilon_s} * v$$

$$z_5 = \sum_{i\in I, t\in T} \alpha_i * x_{i,t}$$

$$z_6 = \sum_{i\in I, t\in T} \beta_i * z_{i,t}.$$

The objective function is the amalgamation of the components above:

$$z = z_1 + z_2 + z_3 + z_4 + z_5 + z_6.$$

Constraints

$$\sum_t x_{i,t} \le 1 \quad \forall i \in I \tag{A.1}$$

$$z_{i,t} = \sum_{t1\in T, t1 \le t} (x_{i,t1} - y_{i,t1}) \quad \forall i \in I, t \in T \tag{A.2}$$

$$\sum_{t\in Ti, t\neq 1} x_{j,t} = 0 \quad \forall j \in J \tag{A.3}$$

$$\sum_{f\in F, s\in SKU} if_{f,i,s,t} \le Big\_M * z_{i,t} \quad \forall i \in I, t \in T \tag{A.4}$$

$$\sum_f if_{f,i,s,t} = \sum_d of_{i,d,t} * \pi_{d,t,s} \quad \forall i,s,t,\ \forall i \in I, t \in T, s \in SKU \tag{A.5}$$

$$\sum_{i\in I} of_{i,d,t} = \Sigma_s \gamma_{d,t,s} \quad \forall d \in D, t \in T \tag{A.6}$$

$$\sum_{i\in I} if_{f,i,s,t} \le \delta_{f,t,s} \quad \forall f \in F, s \in SKU, t \in T \tag{A.7}$$

$$\sum_{s\in SKU \mid \varepsilon_s = p, f\in F} of_{f,i,s,t} \le \zeta_{i,p} \quad \forall i \in I, p \in P, t \in T \tag{A.8}$$

$$\sum_{i\in I} z_{i,t} = n_t \quad \forall t \in T \tag{A.9}$$

$$\sum_{k\in K} k * m_{k,t} = n_t \quad \forall t \in T \tag{A.10}$$

$$\sum_{k\in K} m_{k,t} = 1 \quad \forall t \in T \tag{A.11}$$

$$cs_{s,t} = \sum_{d\in D} \kappa_s * \gamma_{d,t,s}/v \quad \forall s \in SKU, t \in t \tag{A.12}$$

$$ss_{s,t} * sqrt(\xi) = \sum_{k\in K} sqrt(k) * m_{k,t} \sum_d l_s * \gamma_{d,t,s}/v \quad \forall s \in SKU, t \in T. \tag{A.13}$$

How do you arrive at good problem formulations?

Assume that this is a good formulation. What did the authors do to arrive at this formulation?

Why these variables?

	Description
Sets and indices	
I	Set of all potential sites and existing DC locations with index i
J	Set of all existing DCs with index j, a subset of I
F	Set of factories (suppliers) with index f
D	Set of demand points with index d
SKU	Set of SKU groups (aggregated SKUs) with index s
T	Set of time periods (years) with index t
P	Set of storage areas in a DC with index p
K	Set of integer numbers starting from one, with index k
Parameters[a]	
α_i	One-time building investment cost for DC i in RMB yuan
β_i	Fixed operational cost per time period for DC i in RMB yuan
$\gamma_{d,t,s}$	Demand of demand point d per time period t for SKU group s in cases
$\delta_{f,t,s}$	Supply capacity of factory f per time period t for each SKU group s in cases
ε_s	Storage area of SKU group s in cases
$\zeta_{i,t,p}$	DC volume capacity of storage area p in DC i per time period t in cases
η_p	Unit storage cost per day for storage area p in RMB yuan
$\theta_{i,s}$	Per-unit handling cost for SKU s in DC i in RMB yuan
ι_s	Current safety stock cost in number of days of demand for SKU s
κ_S	Current cycle stock level in number of days of demand for SKU s
$\lambda_{f,i,s}$	Unit inbound transportation cost from factory f to DC i for SKU s in RMB yuan
$\mu_{i,d,s}$	Unit outbound transportation cost from DC i to demand point d for SKU s in RMB yuan
v	Number of days in a time period.
ξ	Number of DCs in baseline setting (i.e., number of McDonald's China DCs in 2020)
$\pi_{d,t,s}$	The ratio (percentage) between SKU demand volume and total demand volume for each demand point d, time period t, and SKU s
Decision variables	
$x_{i,t}$	Binary variable, one if DC i will be opened at beginning of time period t, and zero otherwise
$y_{i,t}$	Binary variable, one if DC i will be closed at beginning of time period t, and zero otherwise
$z_{i,t}$	Binary variable, one if DC i exists during time period t, and zero otherwise
n_t	Integer variable, total number of DCs
$m_{k,t}$	Binary variable, one if the number of DCs in time period t is k
$if_{f,i,s,t}$	Nonnegative continuous variable, inbound flow volume from factory f to DC i of SKU s in year t
$of_{i,d,t}$	Nonnegative continuous variable, outbound flow volume from DC i to demand point d of SKU s in year t
$ss_{s,t}$	Nonnegative continuous variable, total safety stock volume for SKU s in year t
$cs_{s,t}$	Nonnegative continuous variable, total cycle stock volume for SKU s in year t

[a]The exchange rate from USD to RMB yuan is 1:7.12 as of this writing.

Why this sum of costs?

Objective Function

The mathematical model's objective function is to minimize the total cost, which comprises the following components:

$$z_1 = \sum_{f\in F, i\in I, s\in SKU, t\in T} if_{f,i,s,t} * \lambda_{f,i,s}$$

$$z_2 = \sum_{i\in I, d\in D, s\in SKU, t\in T} of_{i,d,t} * \mu_{i,d,s} * \pi_{d,t,s}$$

$$z_3 = \sum_{f\in F, i\in I, s\in SKU, t\in T} if_{f,i,s,t} * \theta_{i,s}$$

$$z_4 = \sum_{s\in SKU, t\in T} (ss_{s,t} + cs_{s,t}) * \eta_{\varepsilon_s} * v$$

$$z_5 = \sum_{i\in I, t\in T} \alpha_i * x_{i,t}$$

$$z_6 = \sum_{i\in I, t\in T} \beta_i * z_{i,t}.$$

The objective function is the amalgamation of the components above:

$$z = z_1 + z_2 + z_3 + z_4 + z_5 + z_6.$$

Why these constraints?

Constraints

$$\sum_t x_{i,t} \le 1 \quad \forall i \in I \tag{A.1}$$

$$z_{i,t} = \sum_{t1\in T, t1\le t} (x_{i,t1} - y_{i,t1}) \quad \forall i \in I, t \in T \tag{A.2}$$

$$\sum_{t\in Ti, t\neq 1} x_{j,t} = 0 \quad \forall j \in J \tag{A.3}$$

$$\sum_{f\in F, s\in SKU} if_{f,i,s,t} \le Big\_M * z_{i,t} \quad \forall i \in I, t \in T \tag{A.4}$$

$$\sum_f if_{f,i,s,t} = \sum_d of_{i,d,t} * \pi_{d,t,s} \quad \forall i,s,t,\ \forall i \in I, t \in T, s \in SKU \tag{A.5}$$

$$\sum_{i\in I} of_{i,d,t} = \Sigma_s \gamma_{d,t,s} \quad \forall d \in D, t \in T \tag{A.6}$$

$$\sum_{i\in I} if_{f,i,s,t} \le \delta_{f,t,s} \quad \forall f \in F, s \in SKU, t \in T \tag{A.7}$$

$$\sum_{s\in SKU | \varepsilon_s = p, f\in F} of_{f,i,s,t} \le \zeta_{i,p} \quad \forall i \in I, p \in P, t \in T \tag{A.8}$$

$$\sum_{i\in I} z_{i,t} = n_t \quad \forall t \in T \tag{A.9}$$

$$\sum_{k\in K} k * m_{k,t} = n_t \quad \forall t \in T \tag{A.10}$$

$$\sum_{k\in K} m_{k,t} = 1 \quad \forall t \in T \tag{A.11}$$

$$cs_{s,t} = \sum_{d\in D} \kappa_s * \gamma_{d,t,s}/v \quad \forall s \in SKU, t \in t \tag{A.12}$$

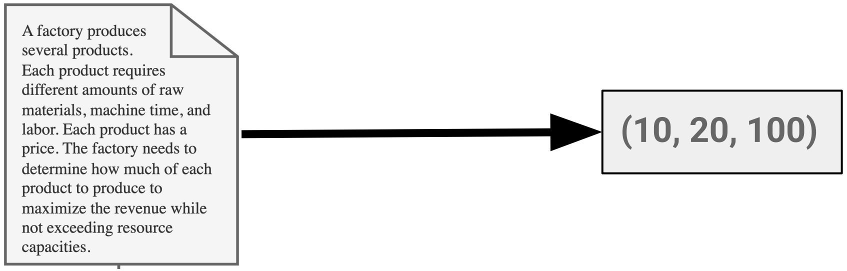
$$ss_{s,t} * sqrt(\xi) = \sum_{k\in K} sqrt(k) * m_{k,t} \sum_d l_s * \gamma_{d,t,s}/v \quad \forall s \in SKU, t \in T. \tag{A.13}$$

There are many pain points in MILP modeling.

- Identifying decision variables and parameters in documentation, data (e.g. dimensions)
- Mathematical problem formulation (e.g. constraints, objective function)
- Performance/efficiency (e.g. adding slack, relaxations)
- Algorithms (e.g. column or row generation)
- Programming (the actual code)
- Sensitivity analysis, visualization (to understand the solution, robustness)
- Debugging (e.g. Identify, assess, diagnose infeasibility or redundant constraints)
- Iteration: What if I reroute…what if I buy more.. What if I use supplier X instead?
- (i) verifying the irreducible infeasible subset, (ii) activating/deactivating constraints, and (iii) relaxing constraints by adding slack variables.

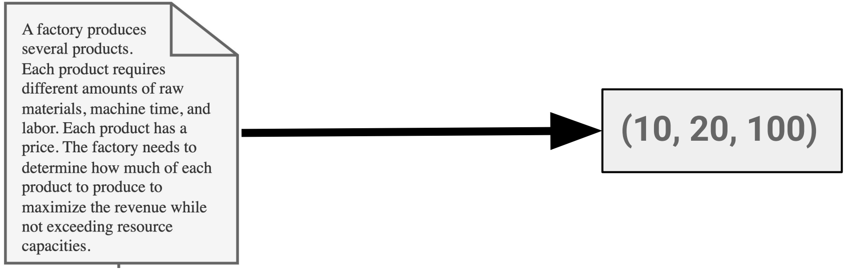
OptiMUS maps NLP goals to quantitative choices.

Vision: Provide a business problem in natural language and receive optimal solutions.



OptiMUS maps NLP goals to quantitative choices.

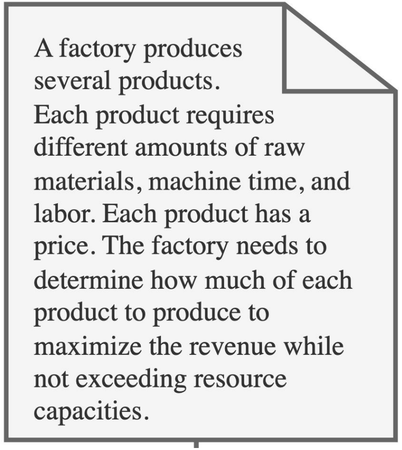
Vision: Provide a business problem in natural language and receive optimal solutions.



The OptiMUS architecture has four ingredients.

- States
- Components
- Error Correction
- Structure Detection

OptiMUS constructs states with useful details.



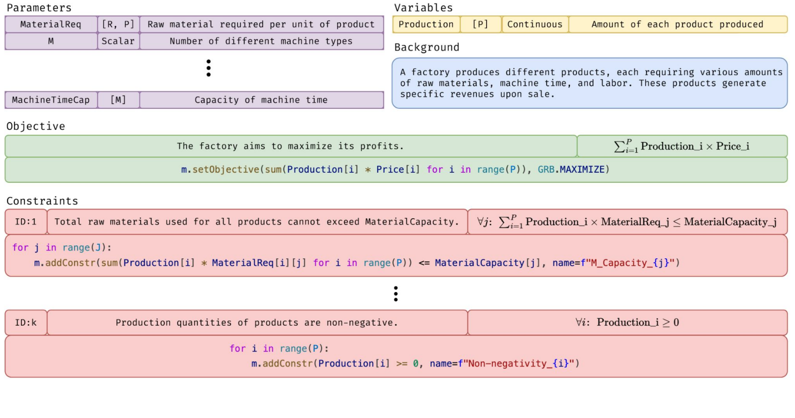
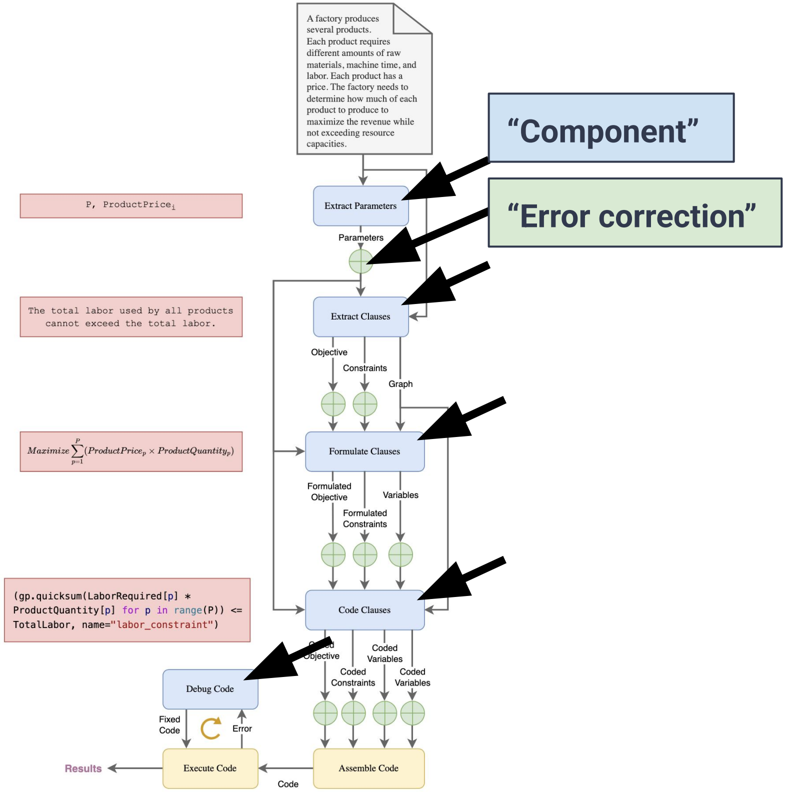


Figure 2 A completed state for a factory production optimization problem

1. **Background/Context**
2. **Parameters**: Symbol, shape, and definition.
3. **(Decision) variables**: Symbol, shape, definition, and type (binary, continuous, or integer)
4. **Clauses (objective function, constraints)** in written in English, LaTeX, AND code.
5. **Connection graph**: Matrix with rows representing clauses and columns variables.

OptiMUS has 5 components (key workflow steps).



Its Error Correction includes reformulation.

They use three tricks.

1. Reflective prompts: Prompt with prior LLM output asking LLM to correct it if necessary.

2. Confidence based feedback: LLMs rates its confidence (1-5); if <5, request feedback.

3. Code debugging: LLM is allowed to make changes it sees fit to make the code run.

Its Error Correction includes reformulation.

They use three tricks.

1. **Reflective prompts**: Prompt with prior LLM output asking LLM to correct it if necessary.

2. **Confidence based feedback**: LLMs rates its confidence (1-5); if <5, request feedback.

3. **Code debugging**: LLM is allowed to make changes it sees fit to make the code run.

Are units the same for both sides of this constraint?

$(p_a + x_a) \cdot d_a \cdot (1 + e_a \cdot \frac{x_a}{p_a}) \leq m_a, \forall a \in A$?

... Left-hand side (LHS):

- $(p_a + x_a)$ represents the new price for article a, which is in euros (€).

- d_a represents the sales forecast (demand) for article a for the next twelve months at the current price, which is in units of the article.

- $(1 + e_a \cdot \frac{x_a}{p_a})$ is a unitless factor ...

Therefore, the unit of the left-hand side is: **euros (€) × units of the article**

Right-hand side (RHS):

- m_a represents the maximum production volume for article a, which is in units of the article (e.g., number of shirts, pants, etc.).

The unit of the right-hand side is: **units of the article**

... this inconsistency suggests an error in the formulation of Constraint 5. To correct this, we should ... here is the corrected constraint:

$$d_a \cdot (1 + e_a \cdot \frac{x_a}{p_a}) \leq m_a, \forall a \in A$$

Figure 3 **OptiMUS-0.3 can fix its constraint modeling errors when prompted "*Are units the same for both sides of C?*"**

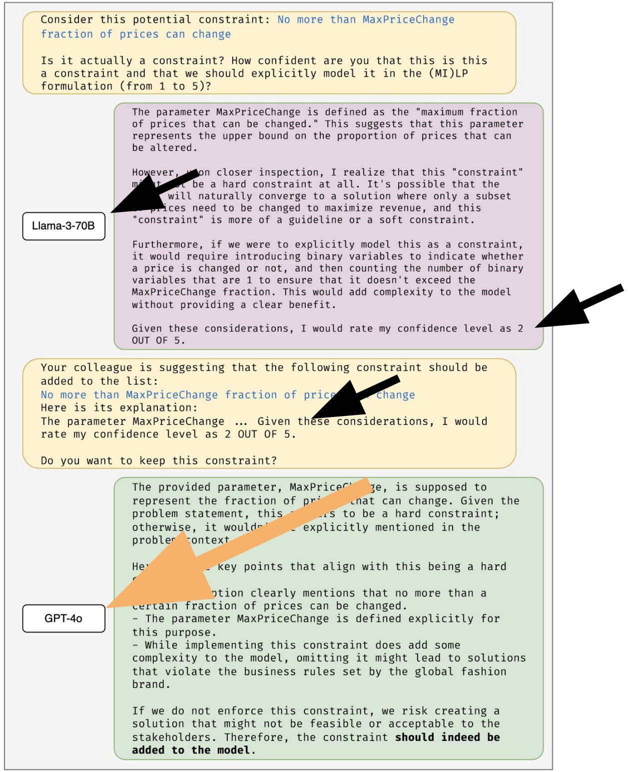
Its Error Correction includes reformulation.

They use three tricks.

1. **Reflective prompts**: Prompt with prior LLM output asking LLM to correct it if necessary.

2. **Confidence based feedback**: LLMs rates its confidence (1-5); if <5, request feedback.

3. **Code debugging**: LLM is allowed to make changes it sees fit to make the code run.



Its Error Correction includes reformulation.

They use three tricks.

1. Reflective prompts: Prompt with prior LLM output asking LLM to correct it if necessary.

2. Confidence based feedback: LLMs rates its confidence (1-5); if <5, request feedback.

3. Code debugging: LLM is allowed to make changes it sees fit to make the code run.

Its Error Correction includes reformulation.

They use three tricks.

1. Reflective prompts: Prompt with prior LLM output asking LLM to correct it if necessary.

2. Confidence based feedback: LLMs rates its confidence (1-5); if <5, request feedback.

3. Code debugging: LLM is allowed to make changes it sees fit to make the code run.

Table 3 Ablation studies on OptiMUS-0.3

	Easy	Hard
Importance of Different Components		
w/o Debugging	73.2%	26.7%
w/o Extraction EC	86.7%	60.5%
w/o Modeling EC	83.8%	65.7%
w/o LLM Feedback	86.6%	68.4%
OptiMUS-0.3	**86.6%**	**73.7%**
Performance with Different LLMs		
LLaMa3.1-70B-Instruct	70.4%	31.5%
GPT4-o	**86.6%**	**73.7%**

OptiMUS maintains a list of structures detected.

It maintains a pool of problem structures that can be solved efficiently with existing solvers.

E.g. For the traveling salesman problem, use Concorde instead of MLP.

Notifies user when their problem structure matches one in the pool & recommends a solver.

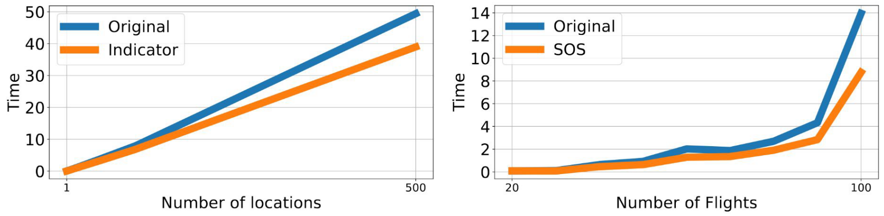


Figure 5 **Impact of Structure Detection on OptiMUS Performance. (Left) Speedup of solving a facility location optimization problem with indicator variables with the structure identified to the solver (indicator) versus with OptiMUS modeling of indicator variable. (Right) Speedup of solving a flight assignment problem with SOS-constraints with the structure identified to the solver (SOS) versus with OptiMUS modeling of the SOS constraints. Full details of problem instances are included in Appendix D.**

For evaluation, they created the NLP4LP dataset.

Table 1 **A comparison on different aspects of complexity for various datasets. The unit for description length is characters**

Dataset	Description Length	Instances (#MILP)	Multi-dimensional Parameters
NL4Opt	518.0 ± 110.7	1101 (0)	×
ComplexOR	497.1 ± 247.5	37 (12)	✓
NLP4LP Easy (Ours)	507.2 ± 102.6	287 (0)	✓
NLP4LP Hard (Ours)	912.3 ± 498.2	68 (18)	✓

355 problems: 22 Dev set problems (12 easy, 10 hard), 332 test set problems (227 easy, 55 hard)

- “Easy” means short problem descriptions, scalar parameters, LPs only.
- “Hard” means long problem descriptions, multidimensional parameters, and both LPs and MILPs.
- Application settings: scheduling, cutting, routing, blending, and packing, and a list of common application domains such as sports, government, retail, agriculture, and energy.

Each problem is accompanied by Code to run the instance and a ground-truth intermediary representation:

- Extracted parameters and targets
- List of clauses of the problem represented in natural language
- LATEX problem statement
- Solution and optimal value

The current results focus on accuracy, efficiency.

	Easy	Hard
Standard	47.3%	33.2%
Reflexion	53.0%	42.6%
CoE	64.2%	49.2%
OptiMUS-0.2	78.8%	68%
OptiMUS-0.3	**86.6%**	**73.7%**

Three Benchmarks: Standard Prompting, Chain of Experts (CoE), Reflexion.

- CoE: A Conductor agent selects a sequence of agents to perform tasks (each is given a prompt and appends its reasoning trace); A “reflection” phase runs and fixes issues.
- Reflexion: “General Purpose framework” for “verbal reinforcement”

Performance Metrics: Accuracy, compilation error (CE) rate, and runtime error (RE) rate.

- Accurate/correct means that (1) the code executes, (2) the optimal value is correct, AND (3) the optimal solution is correct.

Questions

If the goal is to reduce frictions, what are the right outcomes to measure for evaluation?

- Time an individual needs to arrive at a working, efficient, or correct solution?
- Impact of decisions actually made using the solution?

Who is the target user?

- Experts would probably want highly AI to perform highly specific tasks (e.g. parse documents, adding/removing constraints, sensitivity analysis).
- A novice who is considering doing optimization is probably an expert in another area. Does it make more sense to have them use AI to do optimization or to just pay for cloud software tools built for non-technical users?

Summary: GenAI + hardcoded workflow goes far.

To increase adoption of optimization modeling, they want to remove frictions related to tasks that are cycled through during problem formulation. For example,

1. Identifying parameters, constraints, and objectives from documentation and data;
2. Constructing formal mathematical models;
3. Reformulating the models so they may be solved efficiently;
4. Selecting solvers; and
5. Generating code.

To address these, they contribute a framework for AI agentic systems and evaluation data.

OptiMUS 0.3 is an example instantiation of this framework.

Other potentially interesting prior/related work

[Diagnosing Infeasible Optimization Problems Using Large Language Models]

[Large Language Models for Supply Chain Optimization]