OptiMUS-0.3: Using LLMs to model and solve optimization problems at scale

Journal version of ICML'24 paper

Ali AhmadiTeshnizi, Wenzhi Gao, Herman Brunborg, Shayan Talaei, Connor Lawless, Madeleine Udell

Stanford CS/MS&E 331

Automating the modeling bottleneck

Integer programming powers decision-making in operations

- E.g., power system scheduling, medical resource allocation, …

Expertise barrier [Gurobi '23]:

- 81% of Gurobi users hold advanced degrees
- 49% have formal training in operations research

Small firms, municipalities, NGOs lack modeling expertise

- Leads to missed opportunities in efficiency

Goal: automate modeling to democratize optimization

Challenges

- **Long problem descriptions**
  - Real specs can span dozens of pages → more modeling errors
- **Large problem data**
  - Industrial problems involve massive data tables
- **Hallucination**
  - LLMs invent constraints or API calls
  - Hard to detect: code may run but model logic is wrong
- **Poor model quality**
  - Solve time depends on formulation structure
  - LLMs rarely exploit modeling tricks used by experts

Dataset

355 problems: 287 easy LPs, 68 hard LP/MILPs

- Easy: short text, scalar params
- Hard: long, multi-dimensional

Each instance includes text, LaTeX, code, and solution

Covers domains like scheduling, routing, energy, and retail

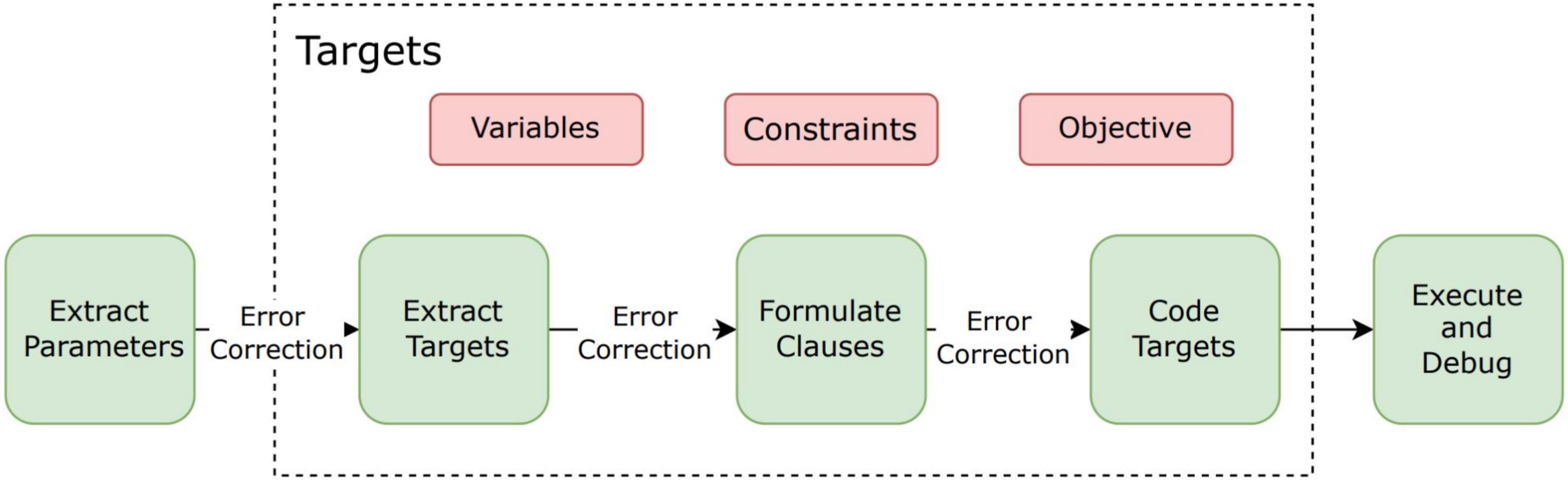
Guarded release to prevent leakage

Components of an integer program

maximize	$\boldsymbol{c} \cdot \boldsymbol{z}$
subject to	$A\boldsymbol{z} \leq \boldsymbol{b}$
	Some variables must be integral

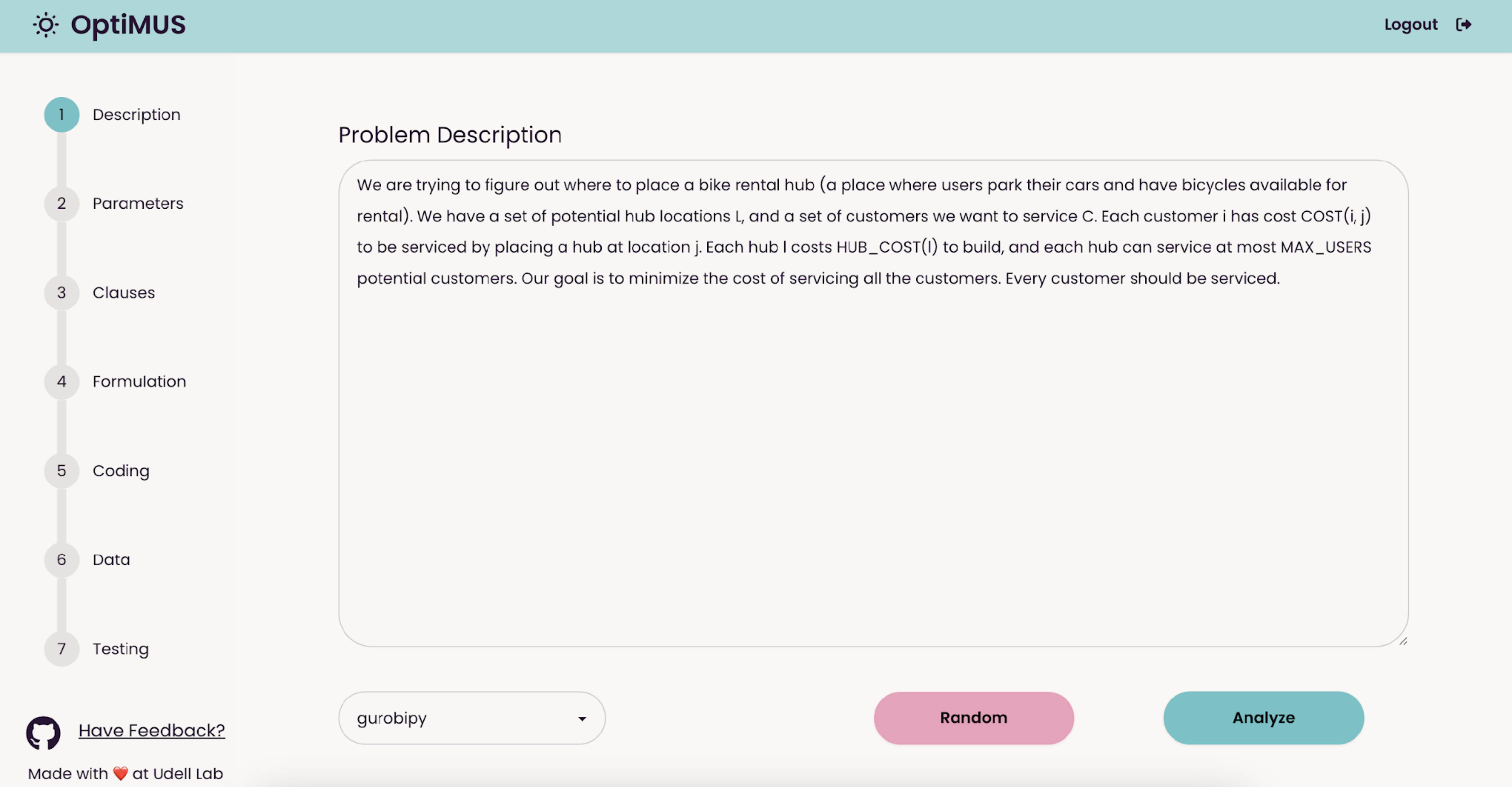
- *Parameters:* $\boldsymbol{c}, A, \boldsymbol{b}$
- *Clauses:* Objective, constraints
- *Variables:* $\boldsymbol{x}$

OptiMUS pipeline



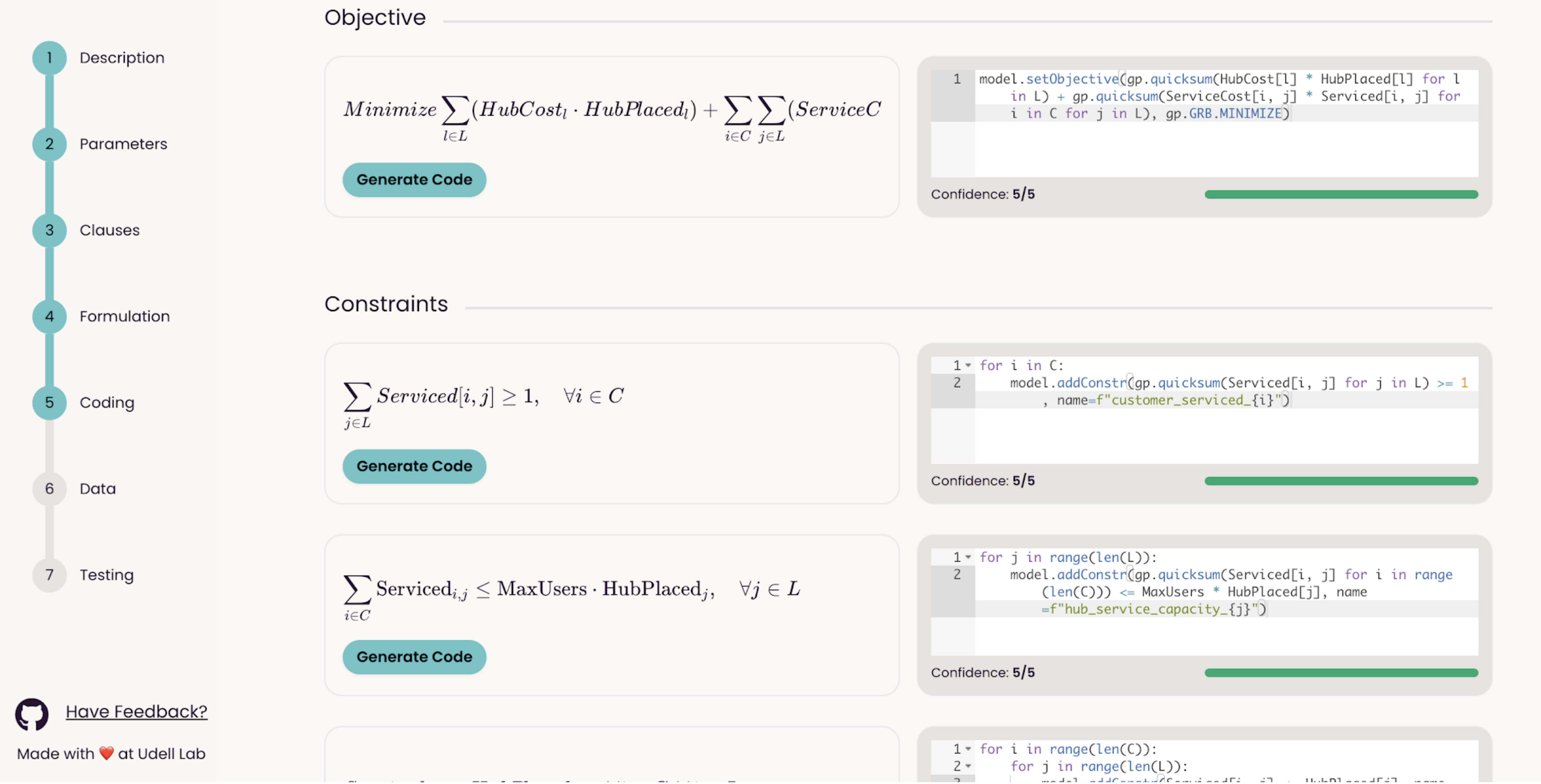
- LLMs at every stage
- Human + solver feedback:
  - Guide iterative LLM corrections and debugging for reliability

Stanford CS/MS&E 331. Figure by Connor Lawless.

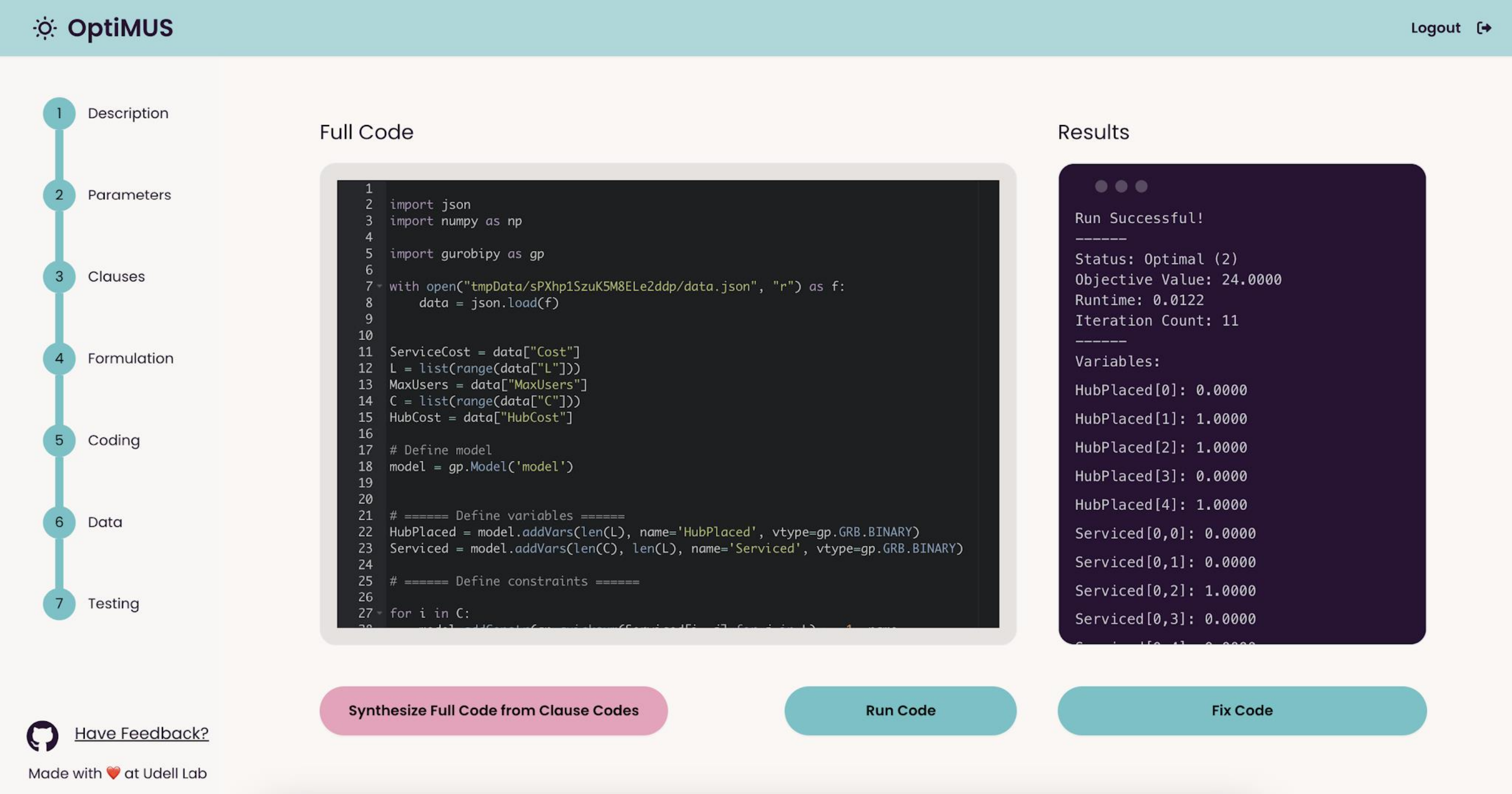


[Stanford CS/MS&E 331](). Figure by Connor Lawless.

Stanford CS/MS&E 331. Figure by Connor Lawless.



Stanford CS/MS&E 331. Figure by Connor Lawless.



Stanford CS/MS&E 331. Figure by Connor Lawless.

Error correction

- **Goal:** Mitigate hallucinations
  - Typical errors: wrong parameters, redundant constraints, invalid code
- **Two correction layers:**
  - *Reflective prompts:* LLM self-checks and revises outputs
  - *Confidence-based feedback:* uncertain results flagged for user review
- Reflective prompting process:
  - Analyzed errors at every modeling stage
  - Designed targeted reflective prompts for each error type
- Substantially lowers modeling error rates

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):

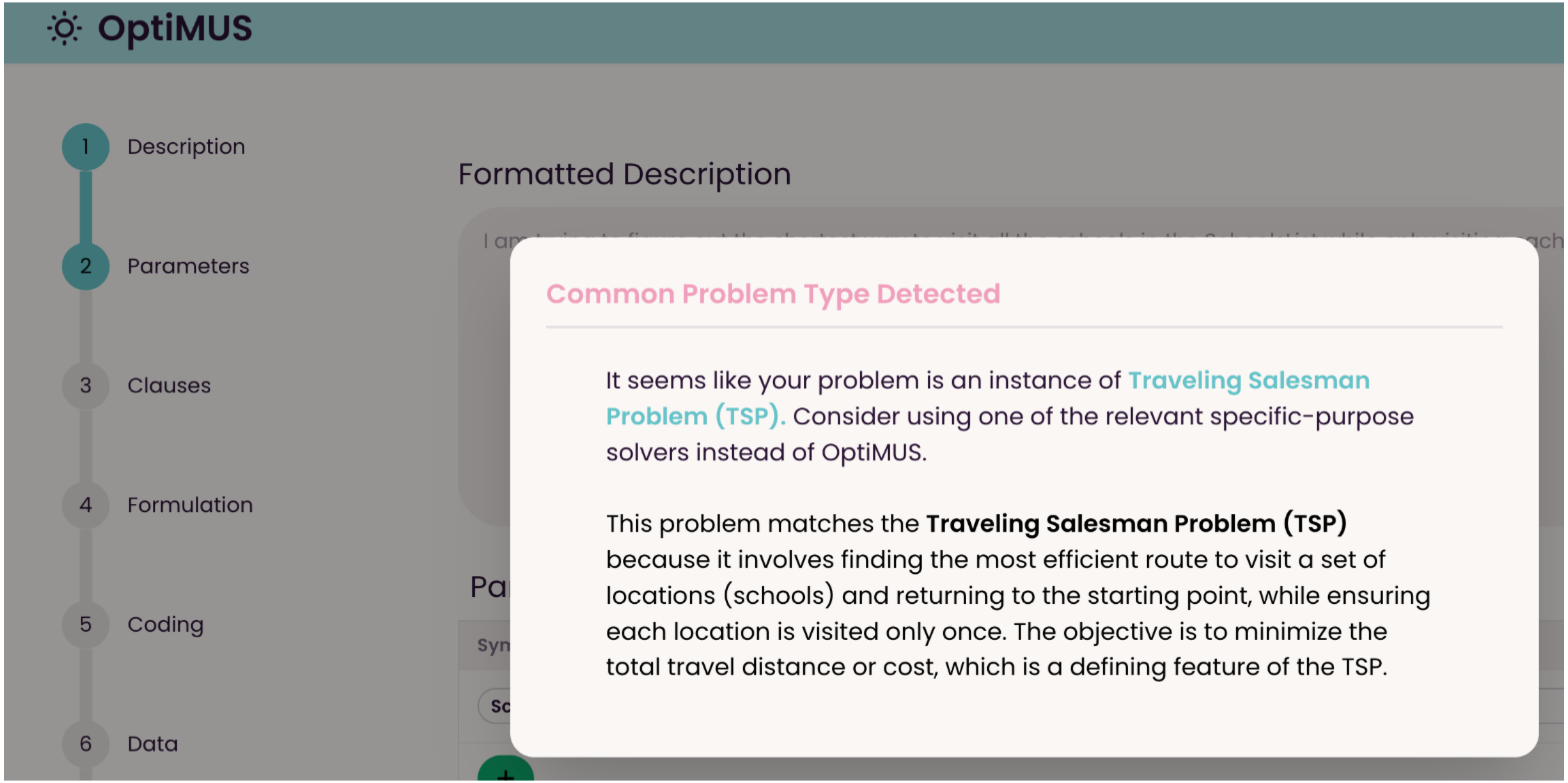
- m_a represents the maximum production volume for article a, which is in units of the article (e.g., number of shirts, pants, etc.).

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:

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

Identifying special problems



Stanford CS/MS&E 331. Figure by Connor Lawless.

Structure detection agent

- **Goal:** Identify and exploit special structures
  - Enhances solver performance and simplifies formulations
- Common structures:
  - Special Ordered Sets (SOS)
  - Indicator and semi-continuous variables
  - Piecewise-linear constraints
- Appear in ~10% of NLP4LP problems
- **Method:**
  - Iterates through known structures
  - LLM decides whether structure applies, then reformulates

	LLM	NL4OPT	NLP4LP	IndustryOR
Methods based on direct prompting				
Standard	GPT-4o	47.3%	33.2%	28.0%
Standard	o1	> 95%	68.8%	44.0%
Reflexion	GPT-4o	53.0%	42.6%	–
Methods based on fine-tuning LLMs				
LLMOPT	Qwen1.5-14B	93.0%*	83.8%*	46.0%*
ORLM	Deepseek-Math	86.5%*	72.9%*	38.0%*
Methods based on agentic frameworks				
CoE	GPT-4o	64.2%	49.2%	–
OptiMUS-0.2	GPT-4o	78.8%	68.0%	–
OptiMUS-0.3	GPT-4o	86.6%	73.7%	37.0%
OptiMUS-0.3	o1	–	80.6%	46.0%

Takeaways:

- Decomposition frameworks out-perform LLMs alone
  - Especially with cheaper models
- Fine-tuning adds a performance increase
  - But OptiMUS is competitive without fine-tuning