Evaluation and Discussion of LLM-based Reasoner and Optimization-based Model for Service Placement

1 Evaluation Overview

This document reports the evaluation results of the LLM-based Reasoner and the optimization-based model for the service placement problem.

Three datasets were employed in the evaluation: Dataset 1 represents a balanced and feasible configuration, Dataset 2 captures a resource-constrained scenario with conflicting requirements, and Dataset 3 corresponds to a medium-scale case where feasibility is maintained but complexity increases.

The code and the datasets can be found in the following GitHub repository: Reasoner_Agent.

Table 1 summarizes the outputs obtained from both models on each dataset.

Table 1: Evaluation results of the LLM-based and optimization-based models.

Model	Dataset 1	Dataset 2	Dataset 3
LLM-based	{ "s1":"g1", ”s2”:”g2”, ”s3”:”n4” }	{ "s2":"g1", ”s1”:”g2”, ”s4”:”n2”, ”s6”:”n2” }	{ "s1":"g2", ”s2”:”g1”, ”s3”:”n2” }
Optimization-based		No result generated	{ "s1":"n3", ”s2”:”n3”, ”s3”:”n3” }

2 Discussion

This section discusses the evaluation results summarized in Table 1.

For Dataset 1, both the LLM-based reasoner and the mathematical solver converge to the same placement. Resource capacities, latency, and bandwidth constraints align such that only one configuration (or a set of equivalent configurations) simultaneously satisfies all requirements while minimizing the objective function.

For Dataset 2, the MILP solver yields no feasible solution. The joint service demands, node capacities, and QoS requirements render the problem infeasible under strict constraint enforcement. Specifically: (i) certain intentions i, such as Heavy workload ($i6$), require simultaneous execution of all services, but their aggregate CPU, memory, and bandwidth demands exceed the available capacity of any node; (ii) bandwidth requirements for some intentions (e.g., $i5$ or $i6$) surpass the provisioned capacity of many nodes; and (iii) latency constraints further reduce feasible options, as nodes that meet resource constraints may fail to satisfy latency requirements. In contrast, the LLM-generated plan relaxes or interprets constraints more flexibly. It may tolerate slight resource over-utilization, accept marginally higher latency, or prioritize full placement over strict adherence to all limits. This heuristic adaptability enables the LLM to propose a feasible configuration even in cases where the MILP solver identifies infeasibility.

For Dataset 3, the divergence between approaches arises from their treatment of objectives and constraints. The MILP solver rigorously enforces all constraints while minimizing total latency, which often leads to clustering services on specific nodes (e.g., placing s_1, s_2, s_3 on n_3 and s_4, s_5, s_6 on n_2). The LLM, by contrast, employs heuristic reasoning that balances constraint satisfaction with load distribution and redundancy. By allowing small deviations from strict resource or QoS bounds, it generates a more distributed placement (e.g., s_2 on g_1, s_1 on g_2, s_4, s_5 on n_1, and s_3, s_6 on n_2). While this solution may not achieve the mathematically minimal latency, it could offer greater robustness in practice.