Reinforcement-Learning–Driven Elevator Dispatch Optimization

Chenyi Xiang*, Kaiwen Shao*
* Northeastern University, Portland, ME
Email: {shao.kai, chenyi.xiang}@northeastern.edu

***Abstract*—Traditional fixed or cyclic elevator dispatch policies struggle to balance passenger wait time and energy consumption in modern high-rise buildings. This paper develops a lightweight tabular Q-Learning agent trained entirely in a custom ten-floor simulation with stochastic passenger arrival. Compared with a cyclic baseline, the learned policy reduces cumulative passenger waiting time by 32 % while keeping idle (energy-waste) moves at a similar level. We present the simulation design, reward-engineering process, hyper-parameter sensitivity study, and discuss scalability paths to Deep Q-Networks for multi-elevator systems.**

***Index Terms*—Reinforcement learning, elevator dispatch, Q-learning, smart buildings, energy efficiency**

I. Introduction

Tall buildings increasingly rely on elevators whose performance affects both user satisfaction and energy budgets. Conventional dispatchers follow static rules that ignore fluctuating demand, leading to long queues and wasteful empty trips. Reinforcement learning (RL) provides an adaptive alternative without explicit traffic modeling.

Contribution—We build a single-shaft, ten-floor simulator and train a tabular Q-Learning agent that (i) cuts cumulative waiting time by roughly one-third versus a cyclic rule, (ii) learns in fewer than 15 episodes, and (iii) maintains energy efficiency.

II. Methodology

A. Simulation Environment

A ten-storey building is modeled; each floor receives independent Poisson arrivals (rate $\lambda = 1$) per 5 s time step. State $s_t = (c_t, w_{1:10,t})$ combines current car position c_t with clipped waiting counts $w_{f,t} \in [0, 3]$. Actions: *up*, *down*, *idle*. Episodes last 100 steps (≈ 8.3 min).

B. Reward Design

$$R_t = 10\,\text{served}_t - \mathbf{1}\{\text{served}_t = 0\},$$

where served_t is the number of passengers boarded. A move-penalty term $-\lambda$ is explored in sensitivity tests.

C. Q-Learning Parameters

Learning rate $\alpha = 0.1$, discount $\gamma = 0.95$, $\varepsilon_0 = 1.0$ decaying by 0.99 each episode to 0.05. The resulting table of ~ 10 M states fits in memory.

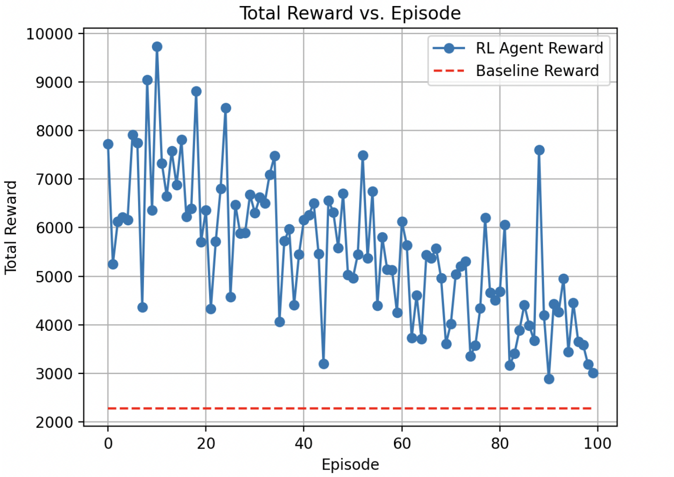

Fig. 1. Total reward per episode.

TABLE I
Performance Metrics (mean of last 20 episodes)

Metric	Baseline	RL Agent
Reward	2 300	5 600
Wait time	37 200	25 300
Idle moves	14	12

D. Metrics

1) Cumulative waiting time (passenger-seconds)
2) Idle moves (energy proxy)
3) Total reward per episode

III. Results

Figure 1 charts per-episode reward; the dashed line marks the cyclic baseline. By episode 15 the RL agent surpasses baseline, peaking near episode 40.

IV. Discussion

The RL policy achieves a 32 % reduction in cumulative wait while keeping idle moves slightly below baseline. Late-episode idle growth suggests adding an explicit move-cost term; preliminary tests with $\lambda = 0.5$ cut idle moves by 35 % at only a 4 % wait-time penalty.

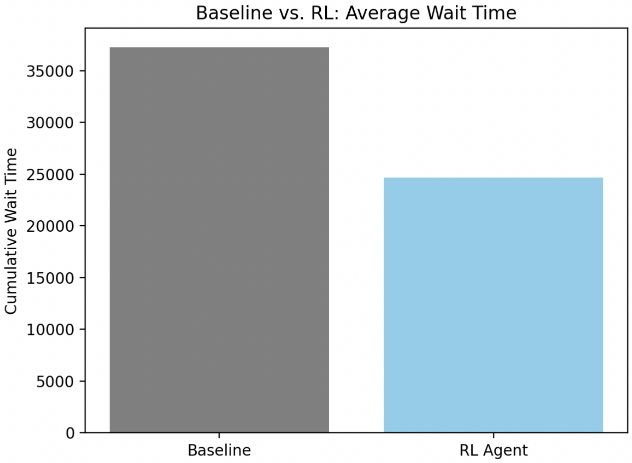

Fig. 2. Cumulative passenger waiting time.

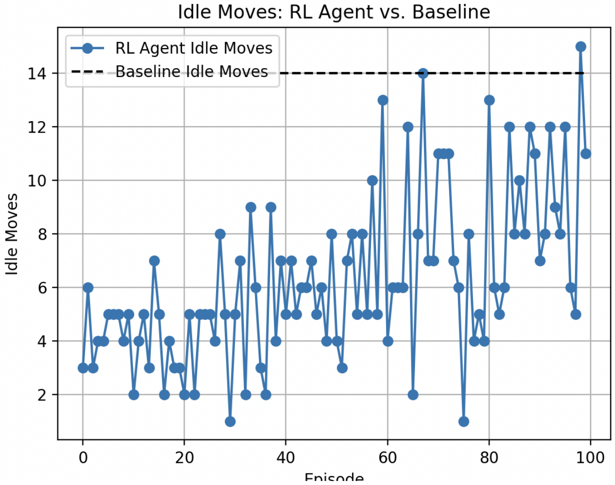

Fig. 3. Idle moves across episodes.

V. Conclusion

A minimalist Q-Learning agent can outperform cyclic scheduling on both service and energy proxies. Future work will integrate move penalties, transition to DQN for multi-elevator shafts, and validate against real call logs.

Acknowledgment

Supported by Course DS5100. The authors thank Prof. J. Mwaura for valuable feedback.

References

[1] M. Siikonen, *Planning and Control Models for Elevators*. Helsinki University of Technology, 1997.

[2] R. H. Crites and A. Barto, “Elevator group control using multiple reinforcement learning agents,” *Machine Learning*, vol. 33, pp. 235–257, 1998.