Swiggy Route Management: A Reinforcement Learning

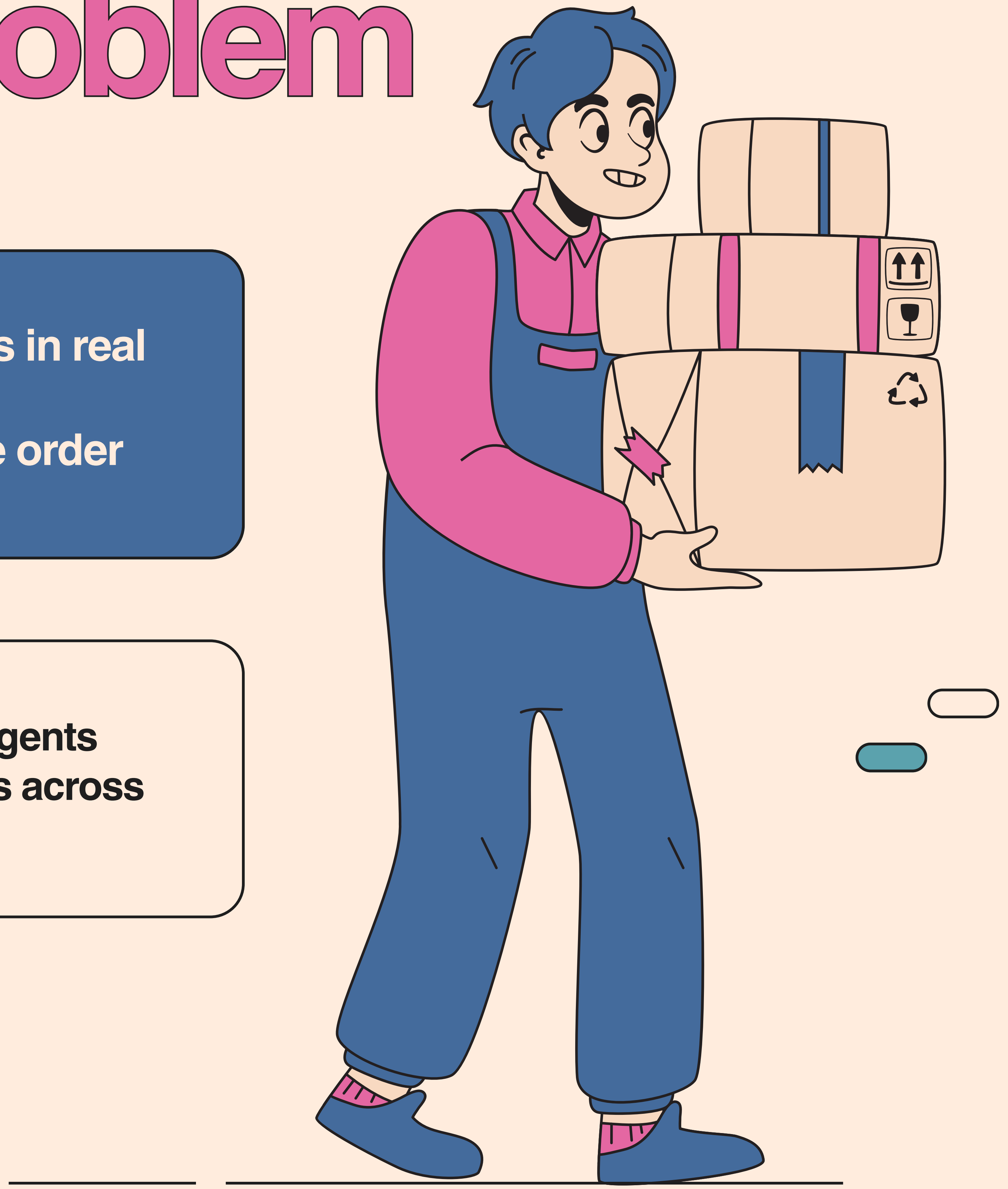
Course : Reinforcement Learning (STAI208)

By: Akhila Macherla

Introduction: The Problem

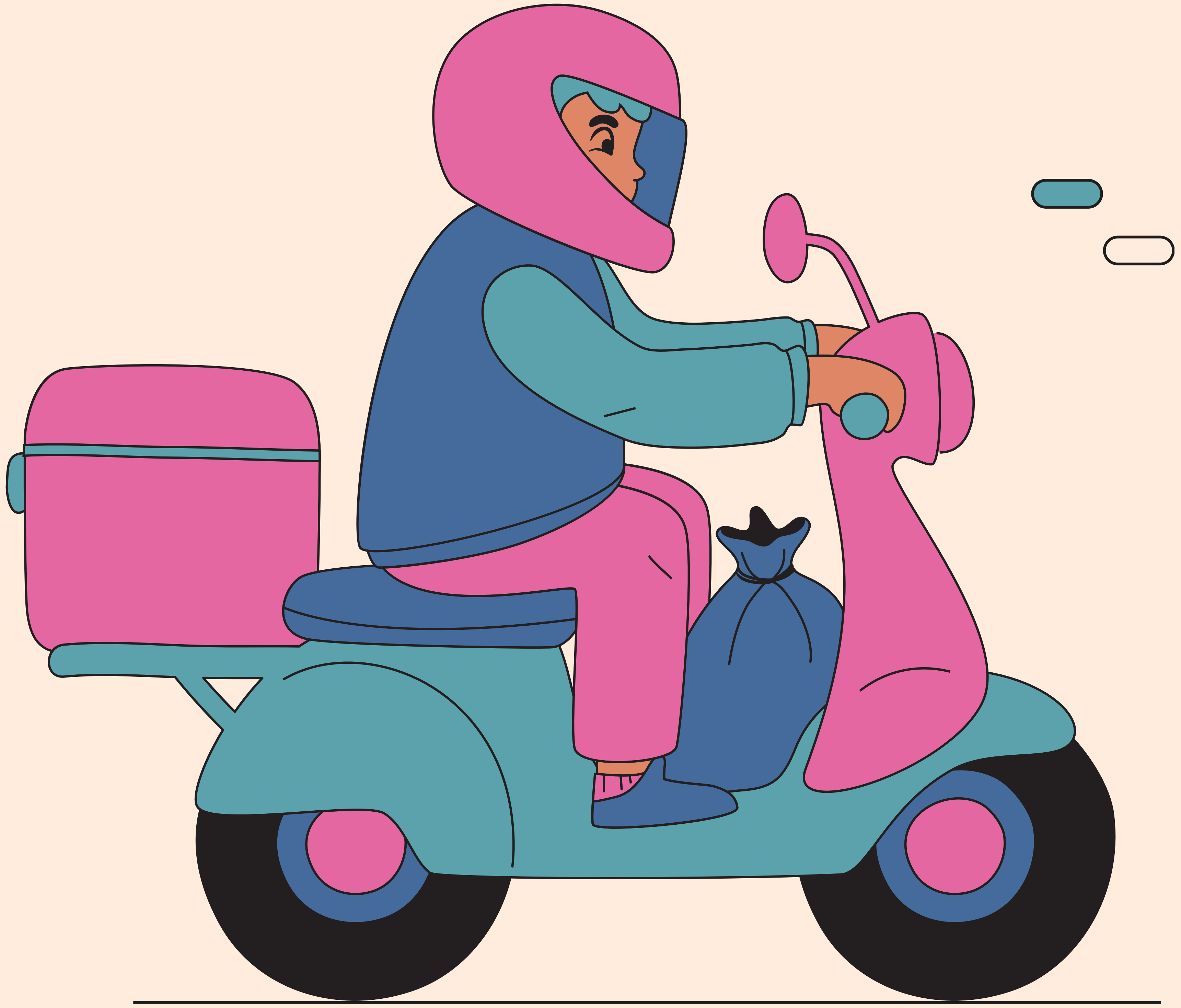
Problem Statement:
How can Swiggy dynamically assign delivery routes to its agents in real time to maximize efficiency, speed, and customer satisfaction considering the parameters of traffic, weather and unpredictable order flow?

Context:
Swiggy, a leading food delivery platform, must manage how delivery agents (riders) pick up orders from restaurants and deliver them to customers across a city.



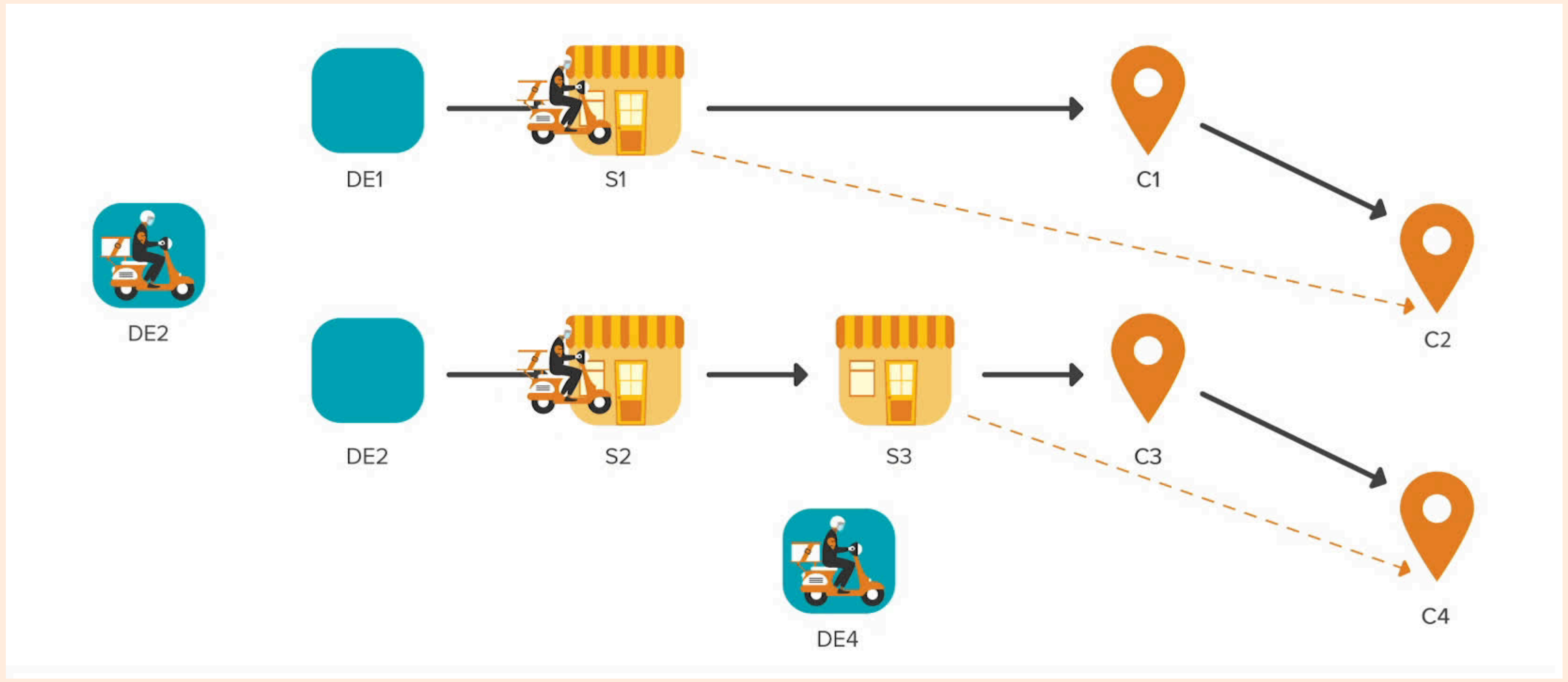
Traditional Method

- The Swiggy route assignment uses an algorithm that is similar to Fully Greedy.
- The Algorithm uses Google Maps API to assign routes based on Estimated Time of Arrival (ETA).
- Assigns the nearest orders and restaurants.
- Assigns the Shortest route without considering traffic or future consequences.
- Fixed rules like Max of 2 orders per Trip which restricts Optimization and Efficiency.
- No involvement of Learning and using past data to improve Over Time.



Why the problem is Significant

- Urban delivery is dynamic: Traffic, sudden order spikes, and city events impact delivery times.
- Current system is limited:
  - Only looks for nearest pick-up/delivery (greedy).
  - Ignores traffic fluctuations and future order opportunities.
  - Uses rigid rules (e.g., only 2 orders per trip), limiting flexibility.
- Business impact: Less efficient routes, more delays, reduced customer satisfaction, and higher costs.



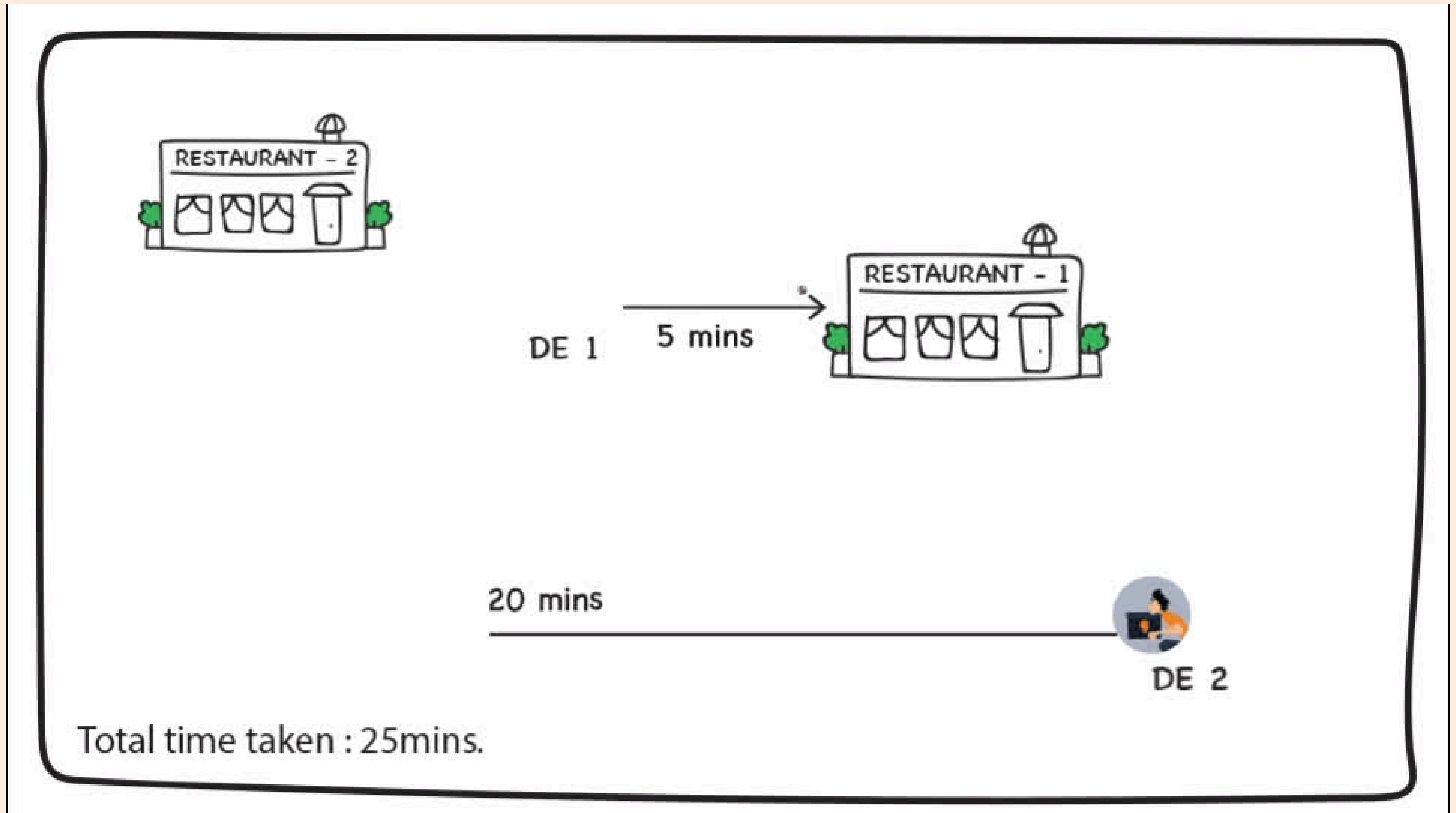
- This Gif shows how the current algorithm allows the DE to usually only carry a limited number of orders (capacity)

https://bytes.swiggy.com/assignment-routing-optimization-for-swiggy-instamart-delivery-part-i-2e8fb3115463

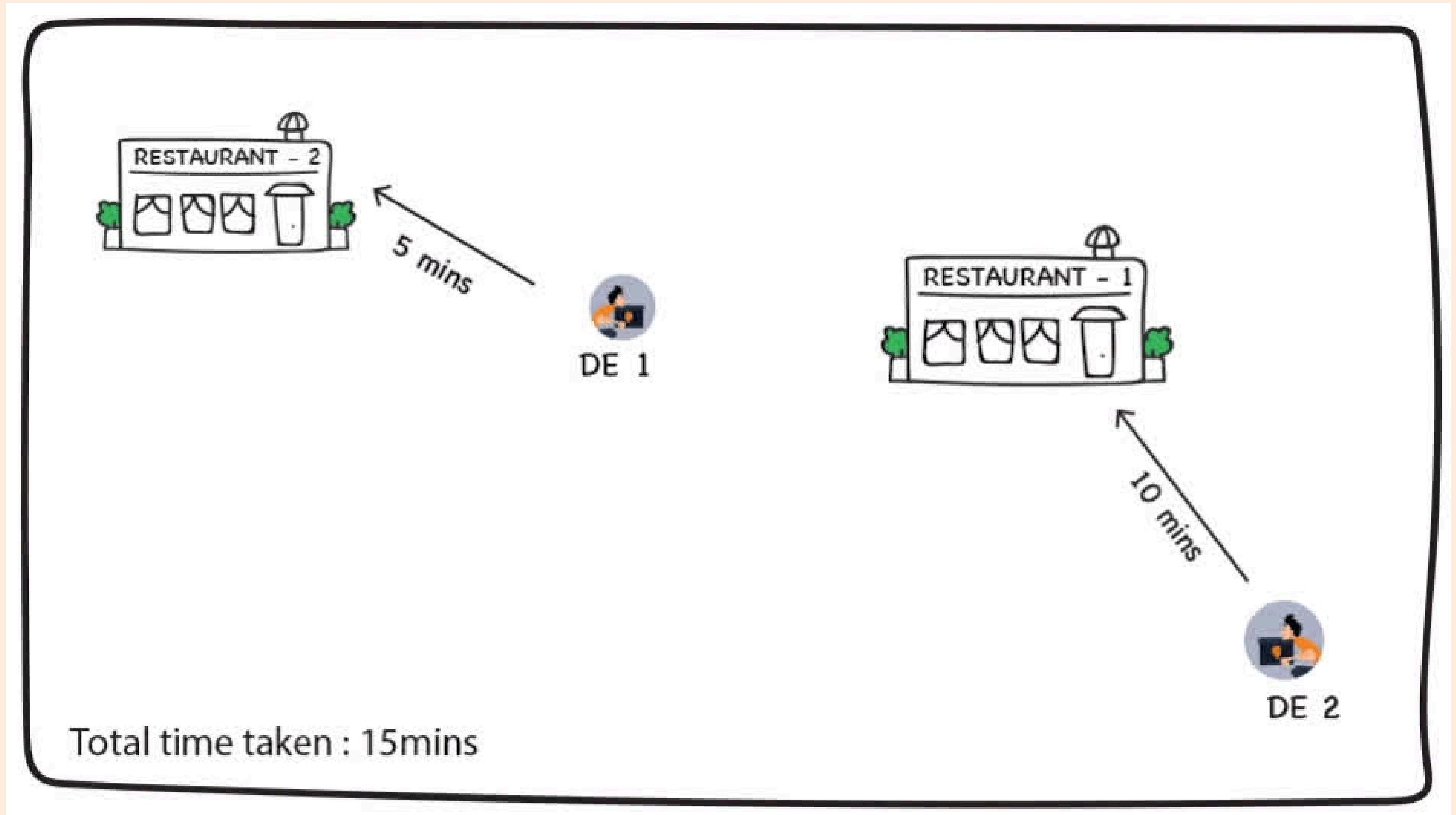
Why Reinforcement Learning?

- Sequential Decision Making: RL plans each move considering its effect on future deliveries, optimizing the whole route sequence.
- Exploration vs. Exploitation: RL tries new routing strategies to discover better paths while using known efficient routes, improving over time.
- Optimization Over Time: RL maximizes long-term success (on-time deliveries, efficient batching) rather than just immediate shortest paths.
- Uncertainty Handling: RL adapts to unpredictable traffic, weather, and order changes by learning robust routing policies from feedback.

Disadvantage of Not using RL



No JIT - Immediate Assignment



JIT Assignment with swaping

The core problem illustrated here is that immediately assigning the “nearest” agent to each order, without considering potential near-future orders, can lead to globally poor outcomes—even if each single assignment appears optimal at the moment.

MDP Formulation for Swiggy's Route Management

Traditional shortest-path algorithms struggle to adapt real time. Reinforcement Learning (RL) can dynamically learn and optimize routes.

MDP Components:

Agent: Swiggy route management system.

Environment: Urban delivery area (map of roads, traffic signals, restaurants, customer homes, traffic density).

States (S): A state is a snapshot of the agent's context, such as:

- Current location (e.g., [Koramangala, Lat: 12.93, Long: 77.61])
- Orders in queue with delivery deadlines
- Traffic density on each road
- Time of day

Example: S = (Location=R1, Orders=[O1,O2], Traffic=T1, Time=t)

Actions (A): Possible movements/decisions:

- Choose the next location to visit (e.g., go to O1's customer, or restaurant R2)
- Pick up or deliver an order

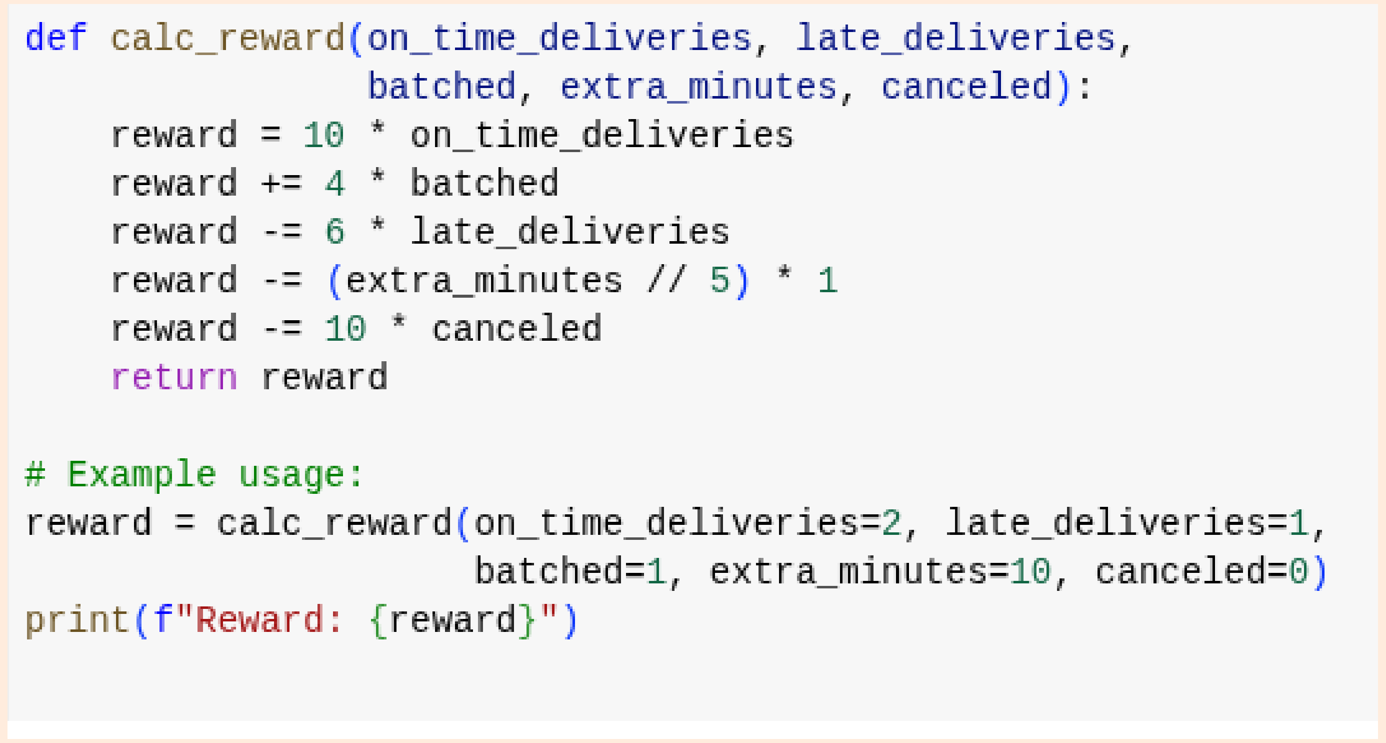
Example: A = {pickup O1, deliver O1, move to R2}

MDP Formulation for Swiggy's Route Management

MDP Components Continuation:

Reward (R): Feedback to the agent:

- +10 for on-time delivery
- -6 for late delivery
- - (extra mins / 5) * 1 for longer route or idle time
- +5 for delivering multiple orders efficiently
- -10 for canceling the orders



```python
def calc_reward(on_time_deliveries, late_deliveries,
                batched, extra_minutes, canceled):
    reward = 10 * on_time_deliveries
    reward += 4 * batched
    reward -= 6 * late_deliveries
    reward -= (extra_minutes // 5) * 1
    reward -= 10 * canceled
    return reward

# Example usage:
reward = calc_reward(on_time_deliveries=2, late_deliveries=1,
                     batched=1, extra_minutes=10, canceled=0)
print(f"Reward: {reward}")
```

State Transitions & Probabilities in Swiggy Route Optimization

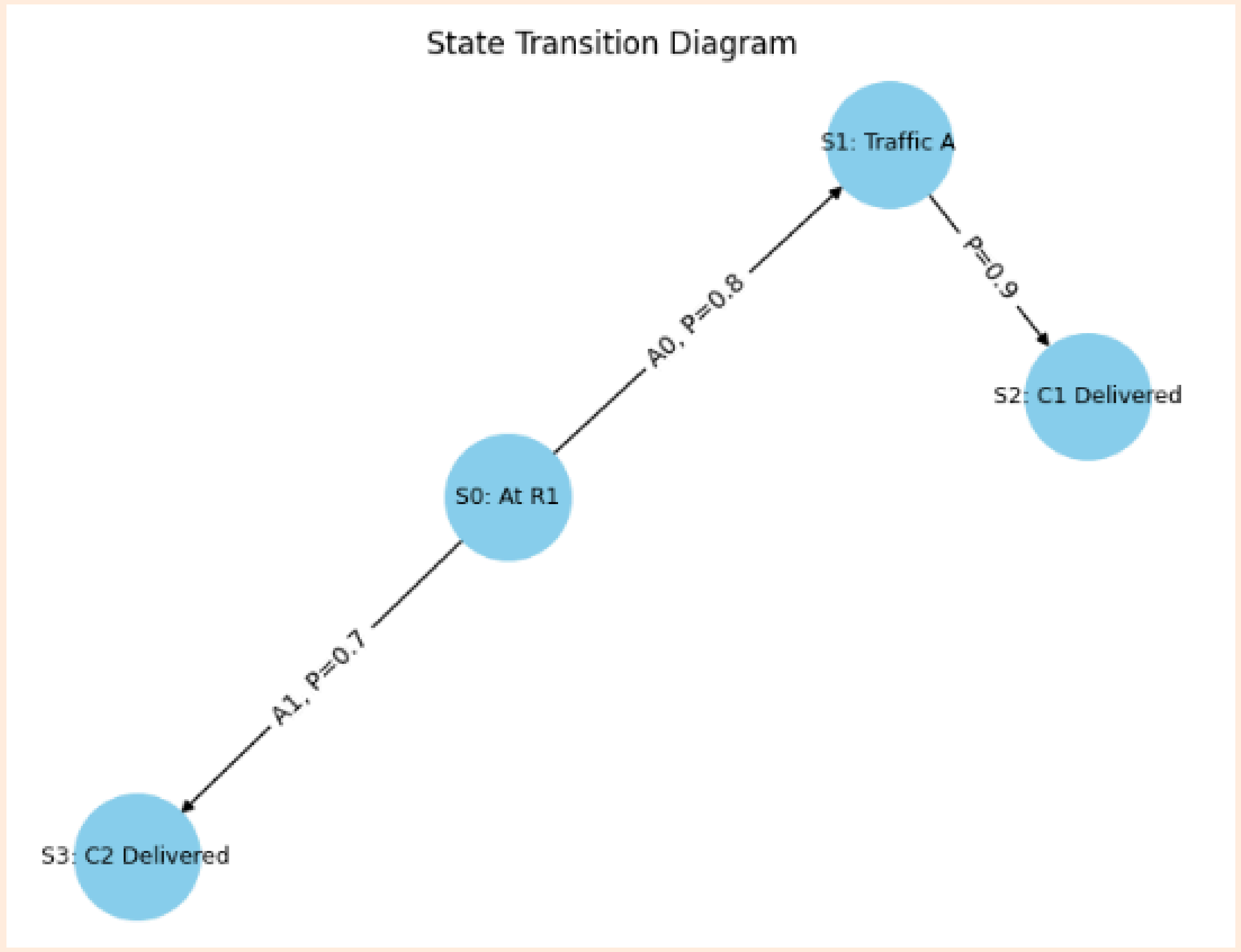
State-Transition Example:

States:

- S0: At restaurant R1 with 2 orders
- S1: At traffic signal A (en route to customer C1)
- S2: At customer C1's location
- S3: At customer C2's location

Actions:

- A0: Move from R1 to C1 (via path A)
- A1: Move from R1 to C2 (via path B)



From State	Action	To State	Probability	Explanation
S0	A0	S1	0.8	Reaches signal A en route to C1 (Path A, normal)
S0	A1	S3	0.7	Reaches C2 directly (Path B, normal)
S1	A0	S2	0.9	Successfully moves from signal A to C1
S1	A0	S1	0.1	Stuck at signal A, possibly delay/traffi

RL Learning & Efficiency

Why RL Works:

- The delivery agent interacts with the city environment.
- Learns from delays, traffic, order deadlines.
- RL uses past experience (episodes) to learn which routes/actions yield maximum reward.

Learning Algorithm (Q-learning):

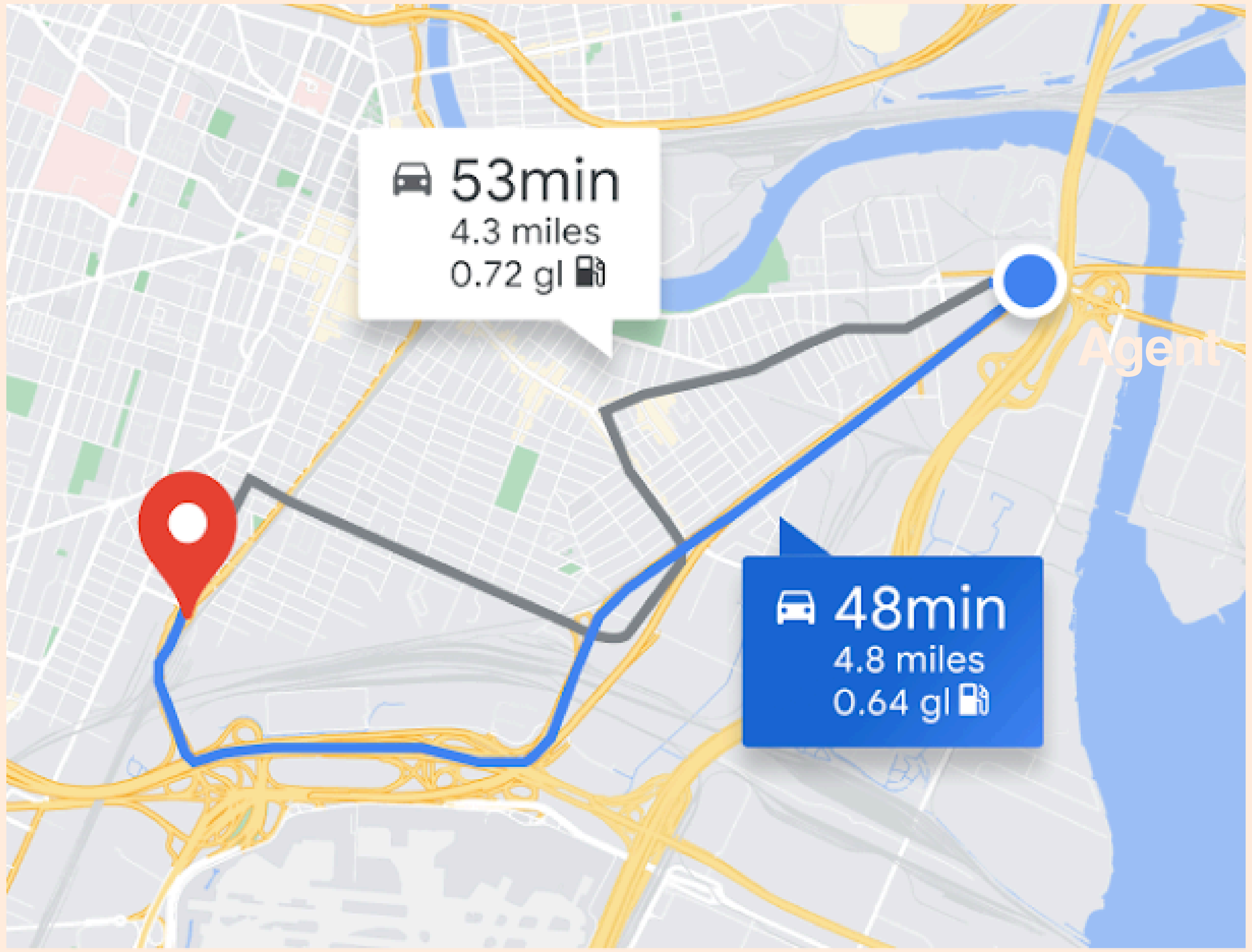
- Q(s,a)←Q(s,a)+α×[Reward+γ×Future Best−Q(s,a)]
- Where:
  - s: current state
  - a: action taken
  - s': resulting state
  - r: reward
  - α: learning rate
  - γ: discount factor

Long-Term Benefits:

- Agent adapts to real-time traffic, weather, peak hours
- Over time, policy π(a|s) improves: selects better actions
- Results in reduced delivery times, higher satisfaction

Example:

- Initially agent takes longest path, gets -5 reward
- Learns that shorter, traffic-avoiding route yields +10
- Over episodes, Q-values converge to optimal path



Practical Challenges for RL in Swiggy Route Management

- Huge state/action space: Too many possible locations, orders, and routes for simple solutions.
- Limited and noisy data: Real-time info on traffic and agents isn't always reliable.
- Instant decisions needed: Assignments must be made in seconds.
- Coordinating many agents: Must optimize the whole fleet, not just one rider.
- Complex rewards: Hard to balance speed, cost, batching, and satisfaction in one formula.
- Adapting to change: City patterns and demand shift constantly.
- Privacy and safety: Must protect sensitive user/agent data.

Conclusion

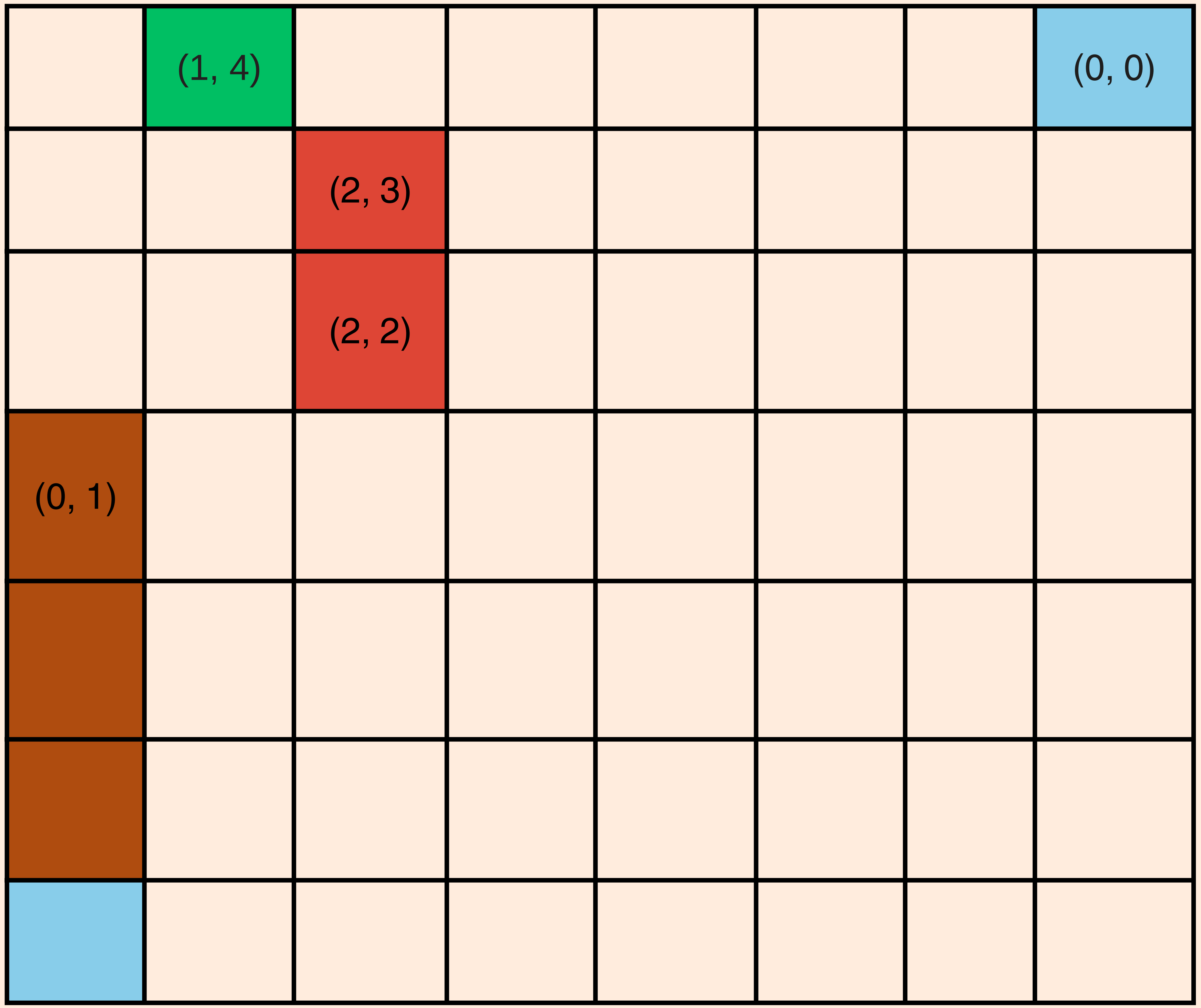
Swiggy's route management is a complex, dynamic problem full of uncertainty and rapid changes, making it an ideal candidate for Reinforcement Learning (RL).

Traditional greedy methods fall short by ignoring traffic fluctuations and smart order batching. By modeling the problem as a Markov Decision Process (MDP), RL learns adaptive, long-term strategies that optimize delivery speed, efficiency, and customer satisfaction.

Although challenges like massive state spaces and real-time requirements exist, these can be overcome through advanced RL techniques, Multi Agent Envinorment, Managing Huge state and action, and careful system design.

With this approach, RL can transform Swiggy's delivery operations to be faster, smarter, and more customer-friendly, setting a new standard for modern urban logistics.

Grid World



Restaurent: Green
Customers: Blue
Traffic: Red
Road Block: Brown

Thank you