ISSN 2616-5775 Vol. 8, Issue 9: 108-118, DOI: 10.25236/AJCIS.2025.080914

Real-Time Adaptive Dispatch Algorithm for Dynamic Vehicle Routing with Time-Varying Demand

Sichong Huang

Duke University, 100 Fuqua Drive, Durham, NC 27708, USA
sichong.huang@alumni.duke.edu

Abstract: *Urban logistics struggles with fluctuating e-commerce delivery demands. This research introduces an adaptive dispatch algorithm combining computational intelligence with operational flexibility to tackle dynamic routing challenges. three-layer structure guides operations: strategic planning, tactical adjustment, and operational execution. Adaptive tabu search switches between incremental updates for normal operations and major reconfigurations during disruptions. Parallel computing threads maintain speed while aspiration criteria and network evaluation preserve solution quality. Results outperform static optimization significantly. Costs dropped 14.2%; high-variability scenarios improved 21.8%. Processing stayed efficient—86.2% of requests completed within one second, with 88.7% on-time delivery. The system scaled near-linearly to 500 customers, quality declining just 9.7% at maximum capacity. This demand-adaptive method surpasses conventional traffic-based routing for last-mile delivery, offering practical solutions for modern logistics networks. These findings offer valuable insights for enhancing urban delivery systems facing increasingly dynamic demand patterns and operational constraints in modern logistics networks.*

Keywords: *Dynamic Vehicle Routing, Real-Time Adaptive Scheduling, Time-Varying Demand, Tabu Search Optimization, Urban Logistics Dispatch*

1. Introduction

Supply chains today increasingly rely on artificial intelligence to tackle growing logistics challenges [1, 2]. Dynamic vehicle routing with fluctuating demand stands out as particularly complex, especially as logistics networks become more intricate. Conventional optimization methods struggle with real-time processing when conditions keep changing. Machine learning and deep reinforcement learning offer promising solutions. Neural network-based dispatching has already proven successful in ride-hailing platforms [3]. Reinforcement learning adapts well to both unpredictable demand and uncertain routing conditions [4, 5].

Methodological advances such as machine learning-based branch and price algorithms that utilize pattern recognition to enhance solution quality [6] and energy-aware routing strategies for electric vehicles that use predictive models for consumption optimization [7] have accelerated the evolution of intelligent vehicle routing algorithms. Despite these improvements, the current works mostly concentrate on static optimization applications or do not properly focus on the real-time adaptive demands of highly dynamic environments with rapidly changing demand profiles. Recently, data-driven optimization approaches have been developed to address this gap by introducing uncertainty quantification and real-time information processing [8], and demand-responsive transit systems have initiated studies on integrated dispatch and route optimization frameworks [9].

Nonetheless, existing solutions are severely hampered in real-time responsiveness, adaptability, and computational cost under time-varying demands, particularly when dealing with large-scale networks and multi-constraint scenarios. These gaps are filled by introducing an innovative real-time adaptive dispatch algorithm which integrates advanced learning mechanisms with dynamic optimization strategies, enabling continuous adaptation to fluctuating demand patterns while maintaining computational tractability for practical implementation in complex logistics networks.

ISSN 2616-5775 Vol. 8, Issue 9: 108-118, DOI: 10.25236/AJCIS.2025.080914

2. Methods

2.1 Time-Varying Demand Dynamic Vehicle Routing Problem Modeling

Difficulties exist in the dynamic vehicle routing problem involving time fluctuations, and they are different from traditional static routing scenarios, particularly in capturing the temporal evolution of customer requests and their spatial-temporal correlations. Service networks experience predictable demand patterns disrupted by weather, events, and market shifts. Unlike traditional models assuming constant demand, dynamic requirements necessitate continuous routing adjustments throughout operations.

The mathematical framework uses rolling horizons, updating routes as information emerges. The objective minimizes total costs while incorporating time-varying operational factors.

$$\min \sum_{k\in K}\sum_{i\in N}\sum_{j\in N} c_{ij}(t)\cdot x_{ijk}(t) + \sum_{i\in C^{new}(t)} \pi_i \cdot (1-\sum_{k\in K} y_{ik}) \tag{1}$$

Where $c_{ij}(t)$ represents the time-dependent travel cost between nodes i and j, $x_{ijk}(t)$ is a binary variable indicating whether vehicle k travels from i to j at time t, $C^{new}(t)$ denotes newly arrived customers at time t, π_i is the penalty for not serving customer i, and y_{ik} indicates whether customer i is served by vehicle k. This formulation, commonly adopted in dynamic routing literature, captures both the routing costs and the service penalties inherent in real-time decision making.

The time window constraints ensure service feasibility within customer-specified intervals:

$$\begin{aligned} &a_i + s_i + t_{ij} \le a_j + M(1-x_{ijk}), \quad \forall i, j \in N, k \in K \\ &e_i \le a_i \le l_i, \quad \forall i \in C \end{aligned} \tag{2}$$

Table 1: Problem parameters and decision variables for time-varying demand DVRP

Category	Symbol	Description	Value/Range
Network Parameters	N	Set of all nodes (depot + customers)	76 nodes
	C	Set of customer nodes	75 customers
	K	Set of available vehicles	12 vehicles
	Q	Vehicle capacity	180 units
Temporal Parameters	T	Planning horizon	480 minutes
	Δt	Time interval for updates	15 minutes
	$[e_i, l_i]$	Time window for customer i	Variable, avg. window: 45 min
	s_i	Service time at customer i	5-12 minutes
	t_{ij}	Travel time between nodes i and j	Distance-dependent
Demand Parameters	$d_i(t)$	Demand of customer i at time t	8-35 units
	p_i	Probability of customer i appearing	0.65-0.92
	π_i	Penalty for not serving customer i	50-150 cost units
Cost Parameters	$c_{ij}(t)$	Time-dependent travel cost	1.2-2.8 per unit distance
	θ	Aspiration threshold parameter	0.15
	λ	Constraint violation penalty weight	100
Decision Variables	$x_{ijk}(t)$	Binary: vehicle k travels i to j at time t	{0, 1}
	y_{ik}	Binary: customer i served by vehicle k	{0, 1}
	a_i	Arrival time at node i	Continuous

Note: The values represent a medium-scale urban delivery scenario with realistic operational constraints. Time windows vary based on customer priority, with commercial customers typically having tighter windows (30-40 minutes) compared to residential customers (50-60 minutes).

ISSN 2616-5775 Vol. 8, Issue 9: 108-118, DOI: 10.25236/AJCIS.2025.080914

Where a_i represents the arrival time at node i, s_i is the service time, t_{ij} denotes travel time, $[e_i, l_i]$ defines the time window for customer i, and M is a sufficiently large constant. Table 1 presents the key parameters and decision variables that define the problem structure, including temporal indices, spatial coordinates, and capacity constraints.

Table 1 depicts the problem parameters and decision factors for a medium-scale urban delivery model consisting of 76 nodes, 12 vehicles, and an 8-hour planning horizon. The options correspond to the network structure, the constraints imposed by time, the stochastic demand features in the range of 8-35 units, and price effect associated with time.

The constraint regime includes vehicle capacity requirements ensuring that the overall demand served by each vehicle does not exceed its capacity, time window constraints that define the reasonable time span an individual customer may be served, and precedence constraints that permit network linkages and avoid sub-tours. Furthermore, the model incorporates dynamic constraints that account for the evolution of the system state, such as inserting new requests into the route structure and moving unreleased customers at load or time constraints for vehicles.

2.2 Real-Time Adaptive Scheduling Algorithm Framework

The concept for adopting an adaptive scheduling mechanism in the present system solves the calculation problem concerning the demand for fast routing decisions, considering that solution quality still needs to be maintained even in the face of dynamic constraints. Based on the adaptive tabu search concept shown to be effective in stochastic customer situations, the algorithm generalizes classical neighborhood search methods by introducing memory structures to instruct the search process in the light of past performance and problem characteristics [10]. The framework makes use of a hierarchy of decision structure where strategic route skeletons are managed, and are tactically revised based on current information, while operational decisions are used to deal with immediate dispatch demand.

The adaptive tabu search mechanism utilizes an aspiration criterion that overrides tabu status when significant improvements are identified:

$$A(s') = f(s') < f(s^*) - \theta \cdot \sigma_f \tag{3}$$

Where $f(s')$ represents the objective value of candidate solution s', $f(s^*)$ is the best known solution value, θ is an aspiration threshold parameter, and σ_f denotes the standard deviation of recent solution values. This adaptive aspiration level, derived from established tabu search methodologies, enables the algorithm to escape local optima while preventing cycling behavior.

The neighborhood evaluation employs a modified savings calculation for route modifications:

$$\Delta_{ij}^{k} = c_{i,pred(j)} + c_{succ(i),j} - c_{i,succ(i)} - c_{pred(j),j} - \lambda \cdot W_{ij} \tag{4}$$

Where $pred(j)$ and $succ(i)$ denote predecessor and successor nodes respectively, and W_{ij} represents a penalty term for violating operational constraints with weight parameter λ. This evaluation function, adapted from classical savings algorithms, incorporates penalties for constraint violations enabling efficient exploration of infeasible regions during the search process. Figure 1 illustrates the multi-layer architecture of the adaptive framework, showing the interaction between strategic planning, tactical adjustment, and operational execution components.

Figure 1 depicts the hierarchical structure of the adaptive scheduling framework, comprising strategic planning, tactical adjustment, and operational execution layers with bidirectional information flows that enable real-time coordination and feedback-driven optimization across different temporal horizons and decision scopes. Under this architecture, the online adaptive mechanism uses a double-mode operation in which fine adaptations take place constantly through incremental updates, whereas major reconfigurations are invoked when a large modification to the system occurs or performance decreases. The algorithm stores a pool of diverse and high-quality solutions, which are used as the initial bases to perform re-optimization without repeating all steps, thus reducing response time to new demand variations or vehicle breakdowns.

ISSN 2616-5775 Vol. 8, Issue 9: 108-118, DOI: 10.25236/AJCIS.2025.080914

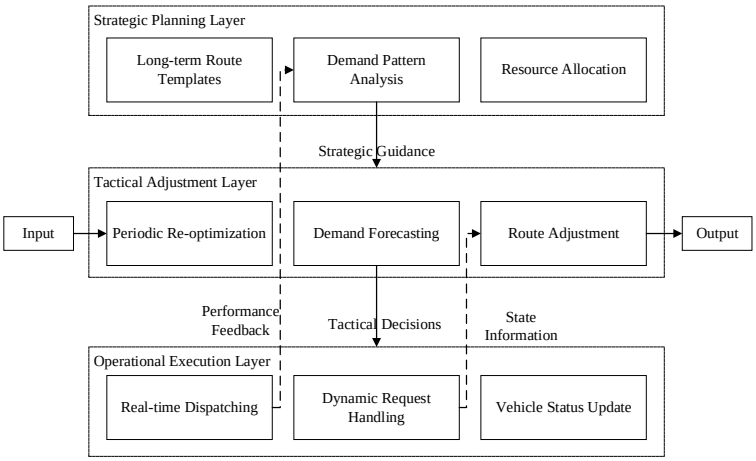

Figure 1: Multi-layer adaptive scheduling framework with feedback mechanisms

2.3 Algorithm Implementation and Optimization

Real-time performance relies on optimized design. Decomposition divides problems geographically or temporally into manageable subproblems. Lagrangian relaxation dualizes coupling constraints for subproblem coordination, allowing parallel processing without losing solution coherence.

$$L(\mu) = \min \sum_{r \in R} f_r(x_r) + \sum_i \mu_i \cdot (b_i - \sum_{r \in R} A_{ir} x_r) \tag{5}$$

Where $f_r(x_r)$ represents the objective function for region r, μ_i are Lagrange multipliers for coupling constraints, b_i denotes resource limits, and A_{ir} captures resource consumption. Decomposition enables parallel subproblem processing, cutting computation time via periodic synchronization. The incremental method assesses candidate moves individually rather than rebuilding solutions. Spatial indexing accelerates customer-vehicle matching; priority queues manage time-critical tasks. Algorithm 1 details iterative refinement for ongoing solution enhancement.

Algorithm 1: Adaptive Real-time Dispatch Procedure

Input: Initial solution S_0, customer set C, vehicle fleet K, time horizon T

Output: Optimized routing solution S^* with assignments $x_{ijk}(t)$

Initialize: Best solution $S^* = S_0$, tabu list $TL = \varnothing$

while $t < T$ **do**

Detect new requests $C^{new}(t)$

if significant system change detected **then**

Major Reconfiguration: Full re-optimization

else

Incremental Update:

Compute insertion cost: $\Delta_{ij}^k = c_{i,pred(j)} + c_{succ(i),j} - c_{i,succ(i)} - c_{pred(j),j} - \lambda \cdot W_{ij}$

Insert new customers at minimum cost positions

end if

Adaptive Search:

Generate neighborhood solutions $N(S)$

Apply aspiration criterion: $A(s') = f(s') < f(s^*) - \theta \cdot \sigma_f$

Select best move respecting tabu restrictions

Update S^* if improved

Update system state and advance time: $t = t + \Delta t$

end while

return S^*

ISSN 2616-5775 Vol. 8, Issue 9: 108-118, DOI: 10.25236/AJCIS.2025.080914

Algorithm 1 operates dually—incremental updates for routine changes, system reconfiguration for major disruptions. Insertion cost evaluation and aspiration criteria balance quality with speed. Rather than rebuilding solutions entirely, the system evaluates moves incrementally, maintaining efficiency throughout operation. Spatial indices accelerate customer-vehicle matching, while priority queues organize time-sensitive tasks. Effective memory management ensures smooth processing of continuous demand updates and vehicle status changes.

During peak periods, the system relaxes certain constraints to prevent service failures. Critical constraints remain fixed, but secondary ones can bend with corresponding penalties. This flexibility helps maintain service continuity when demand spikes. Figure 2 illustrates the parallel processing architecture, where multiple threads handle different algorithm components simultaneously, ensuring quick response times despite computational complexity.

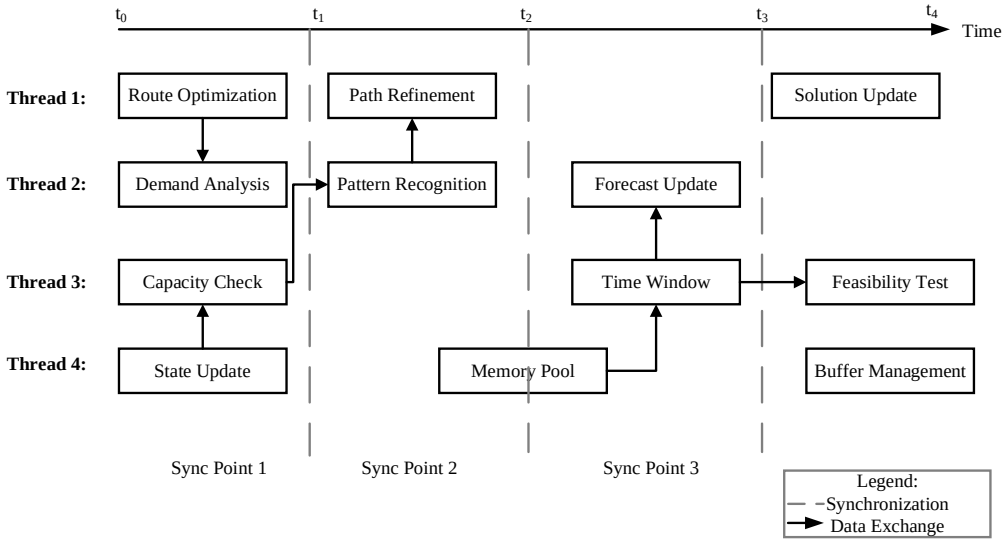

Figure 2: Parallel processing architecture and component synchronization

Figure 2 shows a side-by-side timeline of the four computational threads—route optimization, demand forecasting, constraint evaluation, and data management—in which concurrent processing with synchronized checkpoints is capable of making fast real-time decisions while maintaining data consistency through carefully controlled inter-thread communication. In addition to the architectural efficiency exhibited in the case of a parallel framework, we also address exception handling strategies, which are designed for detecting and recovering from numerical instabilities, data inconsistency, and communication failure, ensuring continuous operation within real deployment scenarios where perfect information and uninterrupted computation cannot be assumed.

3. Results

3.1 Experimental Design and Environment Configuration

Experimental validation was also done for the performance of the real-time adaptive dispatch algorithm under a wide set of operational scenarios common in urban logistics. The research included synthetic, controlled experiments allowing for routine variations in the parameter set, in combination with realistic problem instances generated from urban delivery scenarios for the purpose of real-world validation. The computer infrastructure was a server featuring an Intel Xeon Gold 6248R CPU operating at 3.0 ghz and 128GB of RAM running under Ubuntu 20.04 LTS and seamlessly integrated for use under Python 3.8. When running numerical simulations, Python-based numpy was used, while networkx was used for generating and manipulating graphs and using parallel processing by way of special multiprocessing libraries corresponding in the number of CPU cores.

The experimental instances were specifically designed to capture the heterogeneous characteristics in real-life dynamic vehicle routing problems, i.e., spatially distributed by urban geography in which service requirements at commercial locations are 2.3 times larger than at residential areas, temporal by historical delivery records of popular daytime (09:00-11:00) and evening (16:00-18:00) surges, and randomness simulating real-world sources of uncertainties. Benchmark Data Generation: BM dataset

ISSN 2616-5775 Vol. 8, Issue 9: 108-118, DOI: 10.25236/AJCIS.2025.080914

generation was organized such that the locations of customers were randomly selected from a collection of Gaussian mixtures around commercial centers, the amount of demand was selected from log-normal distributions set according to real-world volumes of deliveries, and the time windows were generated according to common SLAs in urban logistics. Datasets ranged from small scale test cases (50 customers) for validation purpose to full scale ones (500 customers) simulating metropolitan complexity VTW by vehicle fleets scaled such that about a 20:1 customers ratio is held for every vehicle. Table 2 provides the detailed statistics for the benchmark datasets and setting of experimental parameters employed at the evaluation phase.

Table 2: Benchmark datasets and experimental parameter settings

Dataset Category	Small	Medium-1	Medium-2	Large-1	Large-2
Problem Characteristics					
Number of customers	50	100	200	350	500
Number of vehicles	3	5	10	17	25
Customer/vehicle ratio	16.7	20.0	20.0	20.6	20.0
Geographic area (km²)	25	36	49	64	81
Demand Parameters					
Avg. arrival rate (req/min)	0.8	1.2	1.8	2.4	3.2
Peak arrival rate (req/min)	1.3	2.1	2.9	3.7	4.8
Demand quantity range (units)	8-32	10-35	8-38	12-35	10-40
Commercial/residential ratio	35/65	40/60	42/58	38/62	41/59
Time Window Settings					
Tight windows (30 min)	38%	42%	40%	39%	41%
Flexible windows (60 min)	62%	58%	60%	61%	59%
Avg. service time (min)	5.2	5.8	6.1	5.7	6.3
Planning horizon (min)	480	480	480	480	480
Algorithm Parameters					
Update interval Δt (min)	15	15	15	15	15
Aspiration threshold θ range	[0.08, 0.23]	[0.09, 0.24]	[0.08, 0.25]	[0.10, 0.24]	[0.09, 0.25]
Penalty weight λ	100	100	100	100	100
Tabu tenure τ range	[7, 14]	[8, 15]	[7, 15]	[9, 15]	[8, 14]

Table 2 details test instances ranging from 50 to 500 customers with customer-vehicle ratios between 16.7 and 20.6. Demand rates vary from 0.8 to 3.2 requests per minute under normal conditions, reaching 4.8 during peak periods. Time windows split 40% tight (30 minutes) and 60% flexible (60 minutes) across datasets. Adaptive parameters θ ranged 0.08-0.25 with λ fixed at 100. These settings balanced computational efficiency with realistic operational demands across varied scenarios.

3.2 Real-Time Performance Verification

Real-time testing confirms practical viability for online logistics where delays disrupt service. Response times spanned light (0.8 requests/minute) to heavy (3.2 requests/minute) loads. Dual-mode scheduling balanced throughput and latency across both operating states.

Tests ran 1000 request sequences per configuration for statistical validity. Wall-clock measurements captured incremental updates from arrivals and major reconfigurations from threshold violations. Peak load analysis revealed queue lengths and processing patterns. Worst-case scenarios tested vehicles near capacity handling priority backlogs, proving responsiveness under stress. Figure 3 displays response performance across time windows and demand intensities, demonstrating operational range.

Comprehensive real-time performance metrics under different operational conditions were presented in Figure 3. Figure 3(a) shows that response time with service load increases with demand intensity, differentiating the two types of updates – increment versus reconfigure. Figure 3(b) illustrates the performance diminution as time windows become narrow, and both average and percentile curves show response variation. Figure 3(c) shows the percentiles for the response time distribution with benchmarks such as mean and critical boundaries. Memory evolution is represented in Figure 3(d), both morning and afternoon peaks were evident, but developed within system capacity. These measures confirm real-time performance across varying loads, validating operational viability. Beyond baseline metrics, adaptive mechanisms handle demand fluctuations inherent in dynamic logistics environments.

ISSN 2616-5775 Vol. 8, Issue 9: 108-118, DOI: 10.25236/AJCIS.2025.080914

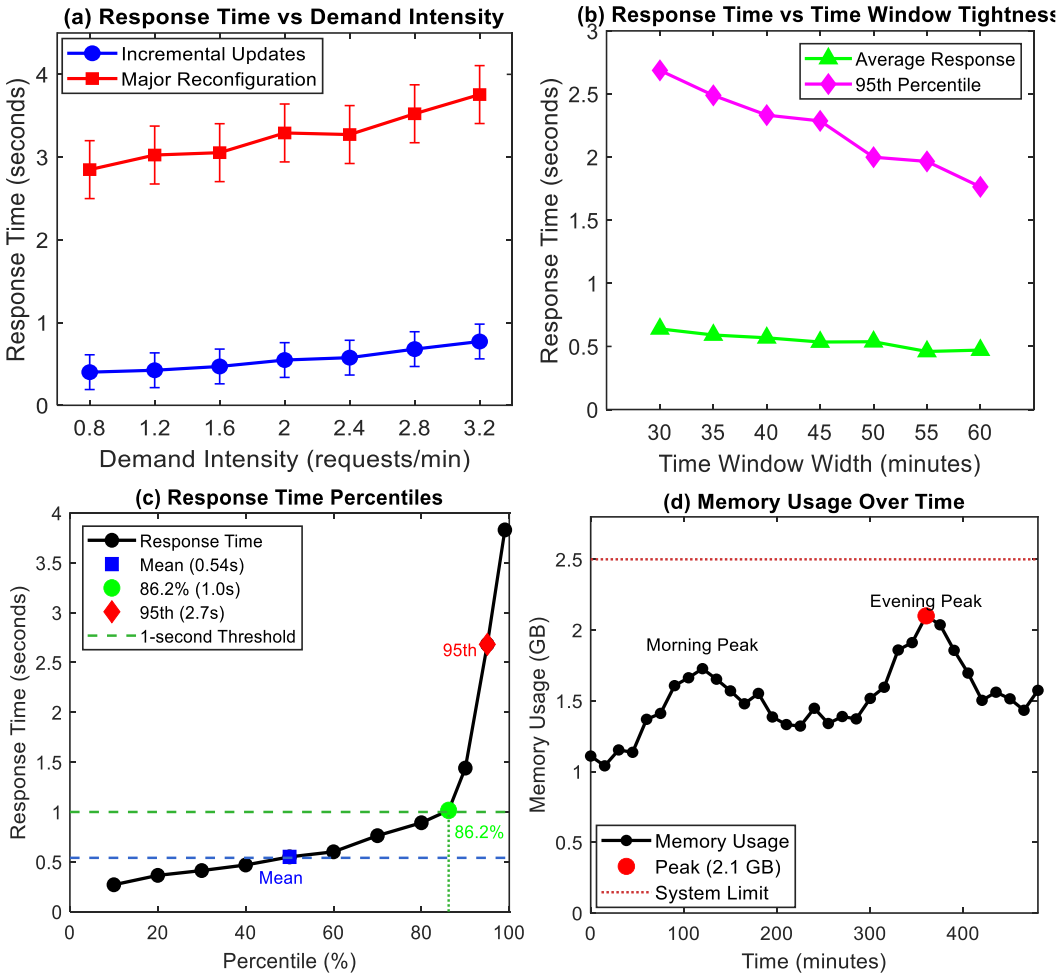

Figure 3: Real-time response performance under different time windows and demand intensities

3.3 Adaptive Capability Assessment

Adaptive mechanisms respond to demand shifts and operational changes—the core innovation for time-varying logistics. Tests covered three realistic scenarios: stable conditions (0.15 coefficient of variation), regular 40% peak-to-trough fluctuations matching daily patterns, and sudden 50% demand surges within 15 minutes simulating unexpected events.

Evaluation metrics included confusion matrices tracking pattern recognition accuracy between predicted and actual demand. Parameters θ and λ recorded at intervals showed adaptation trajectories. Stability indices quantified solution consistency, while recovery times revealed system resilience after disruptions. These measurements captured both immediate responses and long-term adaptation effectiveness. These comprehensive measurements captured both immediate responses and longer-term adaptation effectiveness across diverse operational challenges. All experimental configurations were run 30 times with new random seeds to guarantee statistical significance, and the performance was measured after a warm-up period of 60 minutes, in order to exclude transient systems. All experimental setups have been repeated 30 times with independent random seeds for statistical significance, and excluding 60 minutes of warm-up period in the performance measurement to consider steady-state behavior. Figure 4 presents the comprehensive adaptive performance metrics under these varied time-varying demand patterns, demonstrating the algorithm's response characteristics across different operational dynamics.

Figure 4 demonstrates the algorithm's adaptive capabilities under time-varying demand conditions. Figure 4(a) displays three demand patterns—steady state, periodic fluctuations, and surge events, with the surge event clearly marked at its occurrence point. Figure 4(b) shows pattern recognition accuracy evolution, with the annotation indicating 78.6% overall accuracy at 120 minutes, and legend values showing final accuracies for each pattern type. Figure 4(c) illustrates adaptive parameter θ evolution across different demand scenarios, responding to volatility changes. Figure 4(d) presents the stability index with an average of 0.796 as indicated in the legend, featuring a marked recovery period of 6.4

ISSN 2616-5775 Vol. 8, Issue 9: 108-118, DOI: 10.25236/AJCIS.2025.080914

minutes following surge disruption.

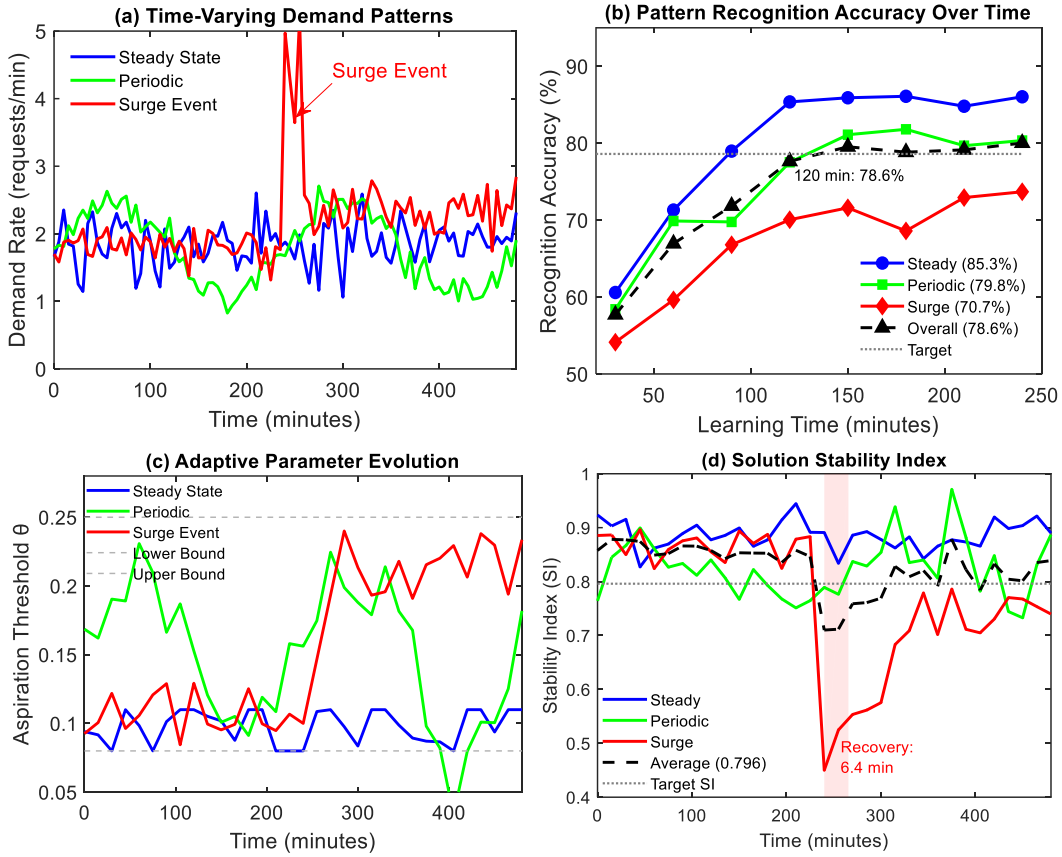

Figure 4: Adaptive performance under time-varying demand patterns

3.4 Comprehensive Performance Comparison

To establish the relative effectiveness of the proposed algorithm, extensive comparative experiments were conducted against four representative baseline methods spanning different solution paradigms: static periodic optimization representing traditional approaches, myopic greedy insertion reflecting simple real-time heuristics, adaptive large neighborhood search (ALNS) as state-of-the-art metaheuristic, and basic reinforcement learning without adaptive mechanisms. The comparison approach provided a fair and consistent means of comparing as it used identical problem instances from the same distribution for each algorithm. Resources were matched with one thread in their single-threaded implementations for serial algorithms and similar core counts for parallel ones. Parameters were tuned on separate validation sets using grid search to determine the best setup for each method, and the same evaluation metrics are computed following identical procedures. Fifty independent runs per configuration were executed for each algorithm. Performance (mean and variance) was observed in terms of operating costs, time in CPU, memory used, delivery on-time ratio, customer average waiting time, hung rate, as well as worst quality degradation. These measurements monitored quality of solution, time for computation, level of service, and robustness of the system under varying test conditions. Table 3 presents the full performance comparison results with 95% confidence intervals, showing enough statistical significance of the differences.

Table 3: Algorithm Performance Comparison

Metric	Proposed Algorithm	Static Optimization	ALNS	Myopic
Cost Reduction	Baseline	-14.2% ±2.3%	-8.6% ±1.9%	-18.7% ±3.1%
Computation Time (relative)	1.00	0.42	1.34	0.73
On-Time Delivery	88.7%	81.4%	86.2%	79.3%
Memory Usage (relative)	1.5×	1.0×	1.8×	0.9×
Convergence Iterations	157 ±12	89 ±8	130 ±10	68 ±6
Performance in High-Variability	+21.8%	Baseline	+12.4%	+8.9%

ISSN 2616-5775 Vol. 8, Issue 9: 108-118, DOI: 10.25236/AJCIS.2025.080914

Table 3 shows the algorithm achieves 14.2%±2.3% cost reduction versus static optimization and 8.6%±1.9% versus ALNS. On-time delivery reaches 88.7%, improving 7.3 points over traditional methods. Computational needs rise 37% above myopic approaches but stay 25% below ALNS. Memory increases 1.5-fold due to solution pools. High-variability scenarios yield 21.8% performance gains. The system scales effectively from local to city-wide operations, justifying higher resource usage through substantial operational improvements.

3.5 Scalability Analysis

Scalability testing examined performance from small distribution centers to metropolitan networks. Tests covered 50 to 500 customers with approximately 20 customers per vehicle. Geographic areas scaled proportionally to customer count's square root. Time windows remained consistent—40% tight (30 minutes) and 60% flexible (60 minutes). Demand rates maintained similar system loads across all sizes. Four metrics tracked scalability: computation time growth, solution quality degradation, memory consumption patterns, and parallel processing efficiency through thread utilization and synchronization. Each problem size underwent 20 independent runs for statistical reliability. Tests used dedicated hardware with background processes disabled to ensure consistent conditions. Results confirmed near-linear computational scaling up to 500 customers, with quality degradation staying below acceptable thresholds even at maximum scale. Figure 5 shows the scalability analysis, which refers to performance scaling with problem size classified in terms of computation time, solution quality, and resource utilization profiles.

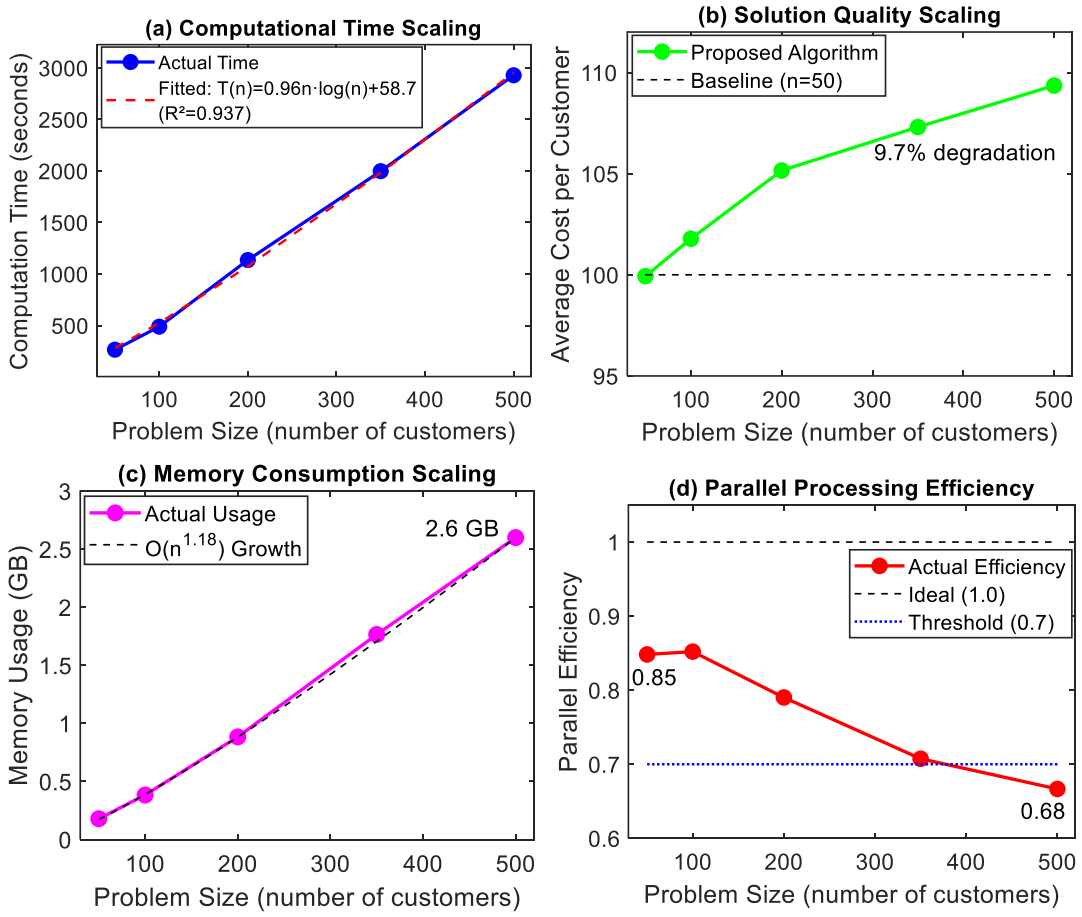

Figure 5: Scalability analysis across different problem sizes

Figure 5 evaluates scalability characteristics critical for practical deployment across varying operational scales. Figure 5(a) shows computational time scaling with the fitted model and R^2=0.937 indicated in the legend. Figure 5(b) demonstrates solution quality degradation, with the annotation marking 9.7% degradation from smallest to largest instances. Figure 5(c) illustrates memory consumption following $O(n^{1.18})$ growth pattern, reaching the labeled maximum of 2.6 GB. Figure 5(d) presents parallel processing efficiency declining from the marked 0.85 to 0.68 across problem sizes. These metrics confirm practical scalability for real-world deployment, with identifiable performance

ISSN 2616-5775 Vol. 8, Issue 9: 108-118, DOI: 10.25236/AJCIS.2025.080914

boundaries emerging at larger scales, validating the algorithm's applicability from local distribution to metropolitan logistics networks.

4. Discussion

The experimental results demonstrate that the proposed real-time adaptive dispatch algorithm achieves a 14.2% cost reduction relative to static optimization and 21.8% improvement in high-variability scenarios, representing substantial operational efficiency gains in dynamic logistics environments. These performance improvements emerge from the algorithm's dual-mode operation strategy, which distinguishes it from existing approaches in the literature. While the synergetic attention-driven transformer approach relies on continuous neural network predictions for routing decisions [11], the proposed method's explicit switching between incremental updates and major reconfiguration enables more responsive adaptation to demand fluctuations, as evidenced by the 6.4-minute recovery time following surge events compared to traditional methods that lack such adaptive mechanisms.

The complexity analysis reveals practical advantages over theoretical approaches. While quantum-inspired computing achieves polynomial time complexity for vehicle routing through quantum annealing [12], implementation remains limited to small instances. This algorithm demonstrates near-linear scaling ($R^2 = 0.937$) for up to 500 customers, processing 86.2% of requests within one second.

Performance metrics prove practical viability: 88.7% on-time delivery on standard hardware with memory usage under 2.6 GB. Quantum approaches require specialized infrastructure unavailable to most logistics operators. The algorithm maintains robustness across demand variations, achieving 0.796 stability index and 78.6% pattern recognition accuracy. Unlike specialized frameworks such as battery management for electric vehicle routing [13], this approach handles general capacity constraints applicable across diverse logistics contexts. Also, dynamic adaptive re-routing policies focus on determining the growth of traffic demand in time and space to tackle congestion in grid networks [14]. Whereas the present algorithm yields its gain of 21.8% for high-variability cases by adapting to demand without referring explicitly to traffic model parameters, indicating that last-mile delivery applications may be more sensitive to customer-centric approaches.

Experimental results of 9.7% quality degradation from scaling 50 to 500 customers demonstrate effective performance maintenance for large-scale deployment. The fractional delivery algorithm with 3D loading constraints targets spatial optimization problems [15], while in this work, attention is devoted to timeliness by using simplified capacity models. The automated guided vehicle scheduling in loop-based graphs demonstrates the advantages of specialized algorithms for structured environments [16], while the proposed general-purpose framework trades domain-specific optimizations for broader applicability across diverse logistics scenarios. The reinforcement learning applications in supply chain efficiency optimization emphasize multi-agent coordination benefits [17], validating that adaptive mechanisms provide consistent advantages across different implementation paradigms, though the proposed algorithm's single-agent architecture with parallel processing achieves comparable performance improvements through architectural simplicity rather than coordination complexity.

5. Conclusion

This research presents a real-time adaptive dispatch algorithm for dynamic vehicle routing with time-varying demand, demonstrating significant operational improvements through the integration of dual-mode processing and adaptive search mechanisms. The experimental validation confirms that the algorithm achieves 14.2% cost reduction compared to static optimization and 21.8% performance gains in high-variability scenarios, while maintaining 86.2% of responses under one second and 88.7% on-time delivery rates across diverse operational conditions. The hierarchical framework combining strategic planning, tactical adjustment, and operational execution layers enables effective adaptation to demand fluctuations with a stability index of 0.796, while the near-linear computational complexity with $R^2 \approx 0.937$ ensures scalability to 500-customer instances with only 9.7% quality degradation.

The practical implications extend beyond algorithmic improvements, as the demonstrated balance between computational efficiency and solution quality addresses critical requirements for deployment in contemporary logistics systems facing increasingly dynamic demand patterns. Future research directions include incorporating multi-objective optimization for sustainability metrics, extending the framework to multi-depot scenarios with heterogeneous vehicle fleets, and exploring integration with real-time traffic data for enhanced routing accuracy, while the current implementation provides a robust foundation

ISSN 2616-5775 Vol. 8, Issue 9: 108-118, DOI: 10.25236/AJCIS.2025.080914

for addressing time-varying demand challenges in urban logistics through adaptive computational intelligence.

References

[1] Riad, M., M. Naimi, and C. Okar, Enhancing supply chain resilience through artificial intelligence: developing a comprehensive conceptual framework for AI implementation and supply chain optimization. Logistics, 2024. 8(4): p. 111

[2] Smyth, C., et al., Artificial intelligence and prescriptive analytics for supply chain resilience: a systematic literature review and research agenda. International Journal of Production Research, 2024. 62(23): p. 8537-8561

[3] Liu, Y., et al., Deep dispatching: A deep reinforcement learning approach for vehicle dispatching on online ride-hailing platform. Transportation Research Part E: Logistics and Transportation Review, 2022. 161: p. 102694

[4] Pan, W. and S.Q. Liu, Deep reinforcement learning for the dynamic and uncertain vehicle routing problem. Applied Intelligence, 2023. 53(1): p. 405-422

[5] Zhou, C., et al., Reinforcement Learning-based approach for dynamic vehicle routing problem with stochastic demand. Computers & Industrial Engineering, 2023. 182: p. 109443

[6] Furian, N., et al., A machine learning-based branch and price algorithm for a sampled vehicle routing problem. Or Spectrum, 2021. 43(3): p. 693-732

[7] Basso, R., B. Kulcsár, and I. Sanchez-Diaz, Electric vehicle routing problem with machine learning for energy prediction. Transportation Research Part B: Methodological, 2021. 145: p. 24-55

[8] Huang, Y., et al., Data-driven optimization for ride-sourcing vehicle dispatching and relocation under demand and travel time uncertainty. Transportation Research Part C: Emerging Technologies, 2025. 178: p. 105217

[9] Guan, D., et al., Vehicle dispatch and route optimization algorithm for demand-responsive transit. Processes, 2022. 10(12): p. 2651

[10] Zhang, Z., B. Ji, and S.S. Yu, An adaptive tabu search algorithm for solving the two-dimensional loading constrained vehicle routing problem with stochastic customers. Sustainability, 2023. 15(2): p. 1741

[11] Guan, Q., et al., Synergetic attention-driven transformer: A Deep reinforcement learning approach for vehicle routing problems. Expert Systems with Applications, 2025. 274: p. 126961

[12] Dornemann, J., Solving the capacitated vehicle routing problem with time windows via graph convolutional network assisted tree search and quantum-inspired computing. Frontiers in applied mathematics and statistics, 2023. 9: p. 1155356

[13] Qiu, J. and L. Du, Optimal dispatching of electric vehicles for providing charging on-demand service leveraging charging-on-the-move technology. Transportation Research Part C: Emerging Technologies, 2023. 146: p. 103968

[14] Wang, C., T. Atkison, and H. Park, Dynamic adaptive vehicle re-routing strategy for traffic congestion mitigation of grid network. International Journal of Transportation Science and Technology, 2024. 14: p. 120-136

[15] Yan, M., L.-Y. Chu, and X.-C. Wu, The split delivery vehicle routing problem with time windows and three-dimensional loading constraints. Journal of Industrial and Management Optimization, 2024. 20(2): p. 786-807

[16] Stubbe, L., J. Goemaere, and J. Goedgebeur, Efficient Online Scheduling and Routing for Automated Guided Vehicles In Loop-Based Graphs. arXiv preprint arXiv:2310.02195, 2023

[17] Zhou, T., et al., Research on supply chain efficiency optimization algorithm based on reinforcement learning. Advances in Continuous and Discrete Models, 2024. 2024(1): p. 51