RESEARCH PAPER

Fair Benchmarking of Deep Reinforcement Learning Against Metaheuristics and Mixed-Integer Programming for Electric-Vehicle Charging Cost Scheduling

Harshal Shelar[1] and Dr. Prakash Burade[2]
[1]Department of Electrical Engineering, Sandip University, Nashik, India
[2]Department of Electrical Engineering, Sandip University, Nashik, India
Harshalshlr77@gmail.com

Abstract
Coordinated charging of electric-vehicle (EV) fleets under time-of-use tariffs is a constrained cost-minimisation problem, yet the literature compares reinforcement learning (RL), metaheuristics, and exact mathematical programming on inconsistent instances, constraints, and metrics, making head-to-head claims hard to trust. This paper defines one evaluation protocol in which mixed-integer linear programming (MILP), particle swarm optimisation (PSO), a genetic algorithm (GA), and a Fitted-Q deep RL agent solve the same 30-EV overnight instance under identical per-vehicle power limits, an aggregate station limit, and energy-delivery deadlines, and are scored on the same three axes: cost, computational overhead, and constraint satisfaction. Solved exactly, MILP attains the global optimum of $73.31 (a 69.33% reduction over uncontrolled charging) in 0.036 s and serves as the optimality reference. The Fitted-Q agent reaches $80.81 — within 10.23% of the optimum — while charging every vehicle on time and respecting the station limit, producing a full-day schedule in 0.029 s after offline training. The untuned metaheuristics are less competitive: PSO and GA leave gaps of 37.59% and 51.55% and, decisively, satisfy the energy-delivery requirement for only 43.3% and 56.7% of the fleet. The results show no single method dominates: MILP wins under a trusted forecast, RL trades offline training for near-optimal millisecond decisions, and off-the-shelf metaheuristics struggle with constraint feasibility at fleet scale. The protocol and its open implementation give a reproducible reference point for future comparisons.
Keywords: Electric Vehicle Charging · Reinforcement Learning · Particle Swarm Optimisation · Genetic Algorithm · Mixed-Integer Linear Programming · Benchmarking.
How to cite this article: Shelar H, Burade P., Fair Benchmarking of Deep Reinforcement Learning Against Metaheuristics and Mixed-Integer Programming for Electric-Vehicle Charging Cost Scheduling. Int J Drug Deliv Technol. 2026;16(89s): 345-355; DOI: 10.25258/ijddt.16.89s.36

Introduction

Electrification of road transport is shifting a large, controllable load onto distribution networks that were never sized for it. When drivers plug in on arrival and draw full rated power until their batteries fill, the aggregate demand of even a modest fleet spikes sharply, overlapping the residential evening peak and, in many cases, exceeding the rated capacity of the local transformer [1, 2]. Modelling studies report that unmanaged charging can raise local peak demand by roughly a fifth, accelerating asset ageing and forcing premature reinforcement [1]. The same flexibility that makes EV load problematic, however, makes it valuable: a parked vehicle that need only be full by morning can absorb energy whenever it is cheapest and least stressful for the grid.

Exact mathematical programming is the natural first response. Cast as a linear or mixed-integer linear program (MILP), EV charging admits a provably optimal solution given a forecast of arrivals, departures, energy needs, and prices, and a simplex or branch-and-cut solver returns it efficiently for realistic fleet sizes [3, 4, 5, 6]. The catch is structural rather than computational: the optimum is only as good as the forecast it assumes, and re-solving is required whenever conditions change. A second family sidesteps the need for a tractable model. Metaheuristics — particle swarm optimisation (PSO) [7, 8, 9], genetic algorithms (GA) [10, 11], and their hybrids [12] — treat the schedule as a black-box vector improved by population-based stochastic search, and are popular because they are simple to implement and impose no convexity requirements.

More recently, deep reinforcement learning (RL) has been proposed to learn a charging policy directly from interaction, avoiding both the forecasting burden of MILP and the per-instance search of metaheuristics [13, 14]. Value-based agents such as the deep Q-network [15] and actor-critic methods such as DDPG [16] and soft actor-critic [17] have been adapted to EV scheduling with reported bill reductions of twenty percent or more [18, 19, 20], and safe-RL variants have begun to address the hard constraints that practical charging imposes [21]. The appeal is a policy that, once trained, maps the current state to a charging decision in a single forward pass.

What the literature does not provide is a clean comparison. Studies validate each method on its own instance, with its own constraint set, its own tariff, and its own definition of success — cost in one paper, peak reduction in another, user waiting time in a third. A method that looks superior may simply have faced an easier problem. Where comparisons exist, they are usually within a family rather than across the three paradigms, and the computational cost of training is rarely placed on the same footing as the solve time of

an optimiser. Hence we don't have a reliable answer to which approach delivers the lowest cost, at what computational price, and at what cost to constraint feasibility for a given charging instance.

The specific contributions of this paper are as follows.

(1) We define a single, reproducible benchmarking protocol for EV charging cost scheduling in which MILP, PSO, GA, and a deep RL agent operate on identical instances, constraints, and metrics.
(2) We implement all four methods on a common 30-EV overnight instance under a three-tier time-of-use tariff and evaluate every solution with one scoring function covering cost, computational overhead, and constraint satisfaction.
(3) We quantify the cost-versus-runtime-versus-feasibility trade-off across the three paradigms, showing that the RL agent reaches within a few percent of the MILP optimum with full feasibility and millisecond inference, whereas untuned PSO and GA satisfy the energy-delivery constraint for barely half of the fleet.
(4) We release the simulation harness, solved schedules, and metric logs so the benchmark can be reproduced and extended.

2. Literature Review

Research on EV charging optimisation clusters into three methodological families. Each is treated in turn before the gap in comparability is drawn out.

2.1 Exact Mathematical Programming

When arrivals, departures, energy demands, and prices are known ahead of time, charging scheduling is naturally a linear or mixed-integer linear program. Diaz-Londono et al. [3] formulate parking-lot energy management as both an LP and a MILP and show that the linear relaxation already captures most of the achievable saving when charging power is continuous. Sharma and Panda [22] embed a MILP in a moving-horizon loop so the optimum is recomputed as new vehicles arrive, showing that exact optimisation can react to uncertainty at the cost of repeated solves. For richer settings, binary variables capture plug-in status: Bignucolo et al. [4] minimise charging time within a bounded cost overrun, and Cvetkovic et al. [5] extend the formulation to vehicle-to-everything operation with on-site PV and storage. Datta and Bhakar [6] compare a MILP against metaheuristics for prosumer microgrid operation, reporting that the exact method dominates on cost but that the constraint count inflates the model dimension. The collective limitation is epistemic: the optimum presupposes a forecast, degrades when that forecast is wrong, and must be regenerated whenever the instance changes.

2.2 Metaheuristic Search

Metaheuristics trade optimality guarantees for modelling freedom. Particle swarm optimisation, introduced by Kennedy and Eberhart [7], has been applied to EV scheduling in many forms: Cui et al. [8] coordinate charge/discharge across a microgrid, Yang et al. [9] propose an improved multi-objective PSO for combined cost and load-fluctuation objectives, and Yin et al. [23] add competitive-learning and local-search operators to escape local optima. Genetic algorithms, whose foundations were laid by Goldberg and Holland [24], occupy similar territory: Kumar et al. [11] hybridise a GA with simulated annealing for resilient station placement, and Tariq et al. [10] combine PSO and a GA in a two-stage coordination scheme. Hybrids of metaheuristics with learning are also emerging, such as the Q-learning-guided PSO of Liu et al. [25]. Comparative metaheuristic studies are common [12], but they compare metaheuristics only against one another. A recurring theme, acknowledged even by their proponents [9, 23], is premature convergence and poor handling of hard coupled constraints, typically mitigated by penalty terms that blur the line between feasible and infeasible solutions.

2.3 Reinforcement Learning

Deep RL reframes scheduling as sequential decision-making, learning a policy from interaction rather than solving a fixed model [26]. Building on the deep Q-network [15] and continuous-control methods such as DDPG [16] and soft actor-critic [17], several groups have demonstrated learned charging policies. Lin et al. [18] minimise station operating cost while preserving service quality, and Sweeney et al. [20] report household bill reductions above twenty percent under time-of-use pricing. Lee and Choi [27] cast multi-EV coordination as a multi-agent problem, Shojaeighadikolaei et al. [28] apply multi-agent RL to demand response more broadly, and Alfaverh et al. [29] use DRL for vehicle-to-grid frequency regulation. Chen and Shi [19] combine soft actor-critic with an augmented-Lagrangian term to keep state-of-charge within bounds, while Li et al. [21] enforce charging deadlines explicitly through safe RL. The critical review by Qiu et al. [14] catalogues this breadth while warning that reward design, sample efficiency, and the absence of standard benchmarks remain unresolved. The strength of RL is forecast-free single-pass decisions; its weaknesses are the offline training burden and a sensitivity to reward shaping that is hard to compare across studies.

2.4 Gap Identification

Three observations follow. First, each family is usually validated in isolation, on a bespoke instance with a bespoke constraint set and success metric, so apparent superiority may reflect an easier problem. Second, where comparisons exist they stay within a family, and the rare cross-family studies [6] still report cost without placing training time, solve time, and constraint feasibility on a common footing. Third, the computational accounting is inconsistent: an RL paper

reports millisecond inference but omits hours of training, while a MILP paper reports solve time but assumes a perfect forecast for free. There is, in short, no protocol under which deep RL, classical metaheuristics, and exact programming solve the same EV charging instance, honour the same constraints, and are judged by the same yardsticks. This gap motivates the formulation in Section 3 and the unified protocol built around it.

3 Problem Formulation

3.1 Problem Statement

We fix one charging instance and one constrained optimisation, and require every method to address it without modification. Consider a facility serving a fleet of N EVs over a horizon discretised into T equal slots of length Δt. Vehicle i arrives at slot $t_{a,i}$, departs at slot $t_{d,i}$, and must receive a requested energy $E_{req,i}$ before departure. The decision variable is the charging power $x_{i,t}$ (kW) drawn by vehicle i in slot t; electricity is billed at a time-varying tariff c_t (USD/kWh).

3.2 Objective Function

The objective minimises the total electricity cost of serving the fleet over the horizon:

$$min\ J\ =\ \sum_{i=1}^{N}\sum_{t=1}^{T} c_t\ \cdot\ x_{i,t}\ \cdot\ \Delta t \qquad (1)$$

Here $c_t \cdot x_{i,t} \cdot \Delta t$ is the cost of energy delivered to vehicle i in slot t. Because c_t is higher at peak and lowest overnight, minimising J pushes charging toward cheap slots subject to the constraints below.

3.3 Constraints

Each EV must receive its full requested energy before departure:

$$\sum_{t=t}^{t} x_{i,t}\ \cdot\ \Delta t\ =\ E_{req,i},\ \ \forall i \qquad (2)$$

the sum running over the presence window from $t_{a,i}$ to $t_{d,i}$. Charging power is bounded by the connector rating and is zero when the vehicle is absent:

$$0\ \leq\ x_{i,t}\ \leq\ P_{max}\ \cdot\ a_{i,t},\ \ \forall i,t \qquad (3)$$

where $a_{i,t} \in \{0, 1\}$ is the availability indicator and P_{max} the maximum charging power. The aggregate draw must not exceed the station limit in any slot:

$$\sum_{i=1}^{N} x_{i,t}\ \leq\ P_{station},\ \ \forall t \qquad (4)$$

Constraint (4) is the coupling that makes the problem hard: it links all vehicles present in a slot, so a feasible schedule must allocate shared capacity across competing deadlines. The tariff is piecewise-constant across three bands, $c_{peak} > c_{mid} > c_{off}$, applied over peak, mid-peak, and off-peak hours (numeric values in Section 5). The problem (1)–(4) is a linear program when $x_{i,t}$ is continuous, so an exact solver returns its global optimum and provides the reference against which every other method is measured.

4 System Model and Solution Methods

All four methods solve the formulation of Section 3 from different directions: MILP attacks the static linear program, the metaheuristics search the flattened schedule vector, and the RL agent treats the horizon as a sequential decision process. The sequential model is described first because it underpins the proposed agent.

4.1 Markov Decision Process Formulation

The facility is modelled as a Markov decision process traversed slot by slot [26]. At slot t the agent observes a state s_t, selects an aggregate charging action a_t, and receives a reward r_t before the system advances. **State.** Six normalised features: time-of-day fraction t/T; share of vehicles present; fraction of total requested energy still undelivered; an urgency signal summing each present vehicle's remaining energy divided by its hours-to-departure; normalised current price; and a sinusoidal encoding of the slot index. The urgency feature lets a single aggregate policy respect individual deadlines. **Action.** The scalar $a_t \in [0, 1]$ sets the fraction of station capacity to draw; the requested power is distributed among present vehicles by deficit-over-deadline priority and clipped to each vehicle's rated power and residual need, so constraints (3) and (4) hold by construction. **Reward.** The reward penalises spending and unmet demand:

$$r_t\ =\ -(\ c_t\ \cdot\ L_t\ \cdot\ \Delta t\ +\ \lambda\ \cdot\ S_t\ ) \qquad (5)$$

where L_t is the realised station load, the first term is the slot energy cost from (1), and S_t is the energy shortfall of any vehicle departing at the end of slot t (with a terminal sweep at $t = T$). The penalty weight λ is large enough that stranding a vehicle never pays. Maximising the discounted return is therefore equivalent to minimising cost (1) subject to deadline (2).

4.2 Proposed Fitted-Q Agent

We solve the MDP with Fitted-Q-Iteration, a batch value-based method matched to the small state dimension and deterministic dispatch rule. A multilayer-perceptron regressor $Q(s, a; \theta)$ approximates the action-value function over eleven discretised action levels $\{0, 0.1, \ldots, 1.0\}$. The agent collects transitions under an ε-greedy policy and periodically refits Q to the bootstrapped Bellman target $y = r + \gamma \max_{a'} Q(s', a'; \theta)$. Exploration decays linearly from $\varepsilon = 1.0$ to 0.02 over the first two-thirds of training. At deployment the policy is greedy, so a full-day rollout costs a single sweep over the horizon — the source of the millisecond inference reported in Section 6.

4.3 Algorithm

Algorithm 1 lists the training and deployment procedure for the proposed agent.

Algorithm 1. Fitted-Q agent for EV charging cost scheduling

```
Input: instance (a_{i,t}, E_req,i, t_d,i, c_t); discount gamma;
episodes G; action set A = {0, 0.1, ..., 1.0}
Output: greedy charging schedule x*[i,t]
1:  initialise Q-network theta; replay buffer B <- empty
2:  for g = 1 to G do
3:      reset env; delivered[i] <- 0; s <- observe()
4:      eps <- max(0.02, 1 - g / (0.65 G))
5:      for t = 1 to T do
6:          a <- random in A w.p. eps, else argmax_a Q(s,a;theta)
7:          target_kw <- a * P_station
8:          split target_kw to present EVs by deficit/hours-left;
9:              clip each to P_max and residual need  (enforces (3),(4))
10:         update delivered[]; cost L_t; shortfall S_t
11:         r <- -(c_t * L_t * dt + lambda * S_t)          (Eq. 5)
12:         s' <- observe(); store (s,a,r,s') in B; s <- s'
13:     end for
14:     every 20 episodes: refit Q on B to Bellman targets
15: end for
16: greedy rollout: for t=1..T, a<-argmax_a Q(s,a;theta);
17:     apply dispatch rule; record x*[:,t]
18: return x*[i,t]
```

The dispatch rule in lines 8–9 guarantees feasibility of (3) and (4) at every step, so the only constraint the agent must learn to honour is the energy deadline (2), enforced through the shortfall penalty.

4.4 Baseline Methods

MILP. The continuous linear program (1)–(4) is solved exactly with the HiGHS dual-revised-simplex solver [30] through the SciPy interface, giving the reference cost and optimality gaps. **PSO.** Particle swarm optimisation [7] searches the flattened schedule vector; fifty particles are initialised to feasible uniform profiles and updated with inertia 0.72 and coefficients 1.49, with availability masking and box-clipping for (3) and penalties for (2) and (4). **GA.** A real-coded genetic algorithm [24] with a population of sixty evolves the same vector by tournament selection, blend crossover, Gaussian mutation, and two-elite preservation, with the identical penalised fitness. **Uncontrolled baseline.** Every vehicle charges at full rated power from arrival until full, ignoring price and the station limit; this defines the 0% cost-reduction reference.

5 Experimental Setup

5.1 Instance and Tariff

The benchmark models an overnight residential/community charger serving **30 EVs** over a **24-hour** horizon from noon to noon, discretised into **96 slots** of 15 minutes. Arrivals are drawn from a late-afternoon Gaussian and departures from an early-morning Gaussian, so each presence window spans the evening peak, the overnight off-peak trough, and part of the next morning. Requested energies are sampled uniformly in **15–40 kWh** and capped at what each stay can deliver. The three-tier tariff charges \$**0.32**/kWh at peak (16:00–21:00), \$**0.16**/kWh mid-peak, and \$**0.08**/kWh off-peak (23:00–06:00). The station limit is **110 kW** and each connector is rated **7.2 kW**. Fig. 1 shows the benchmarking framework and Fig. 2 the generated instance against the tariff.

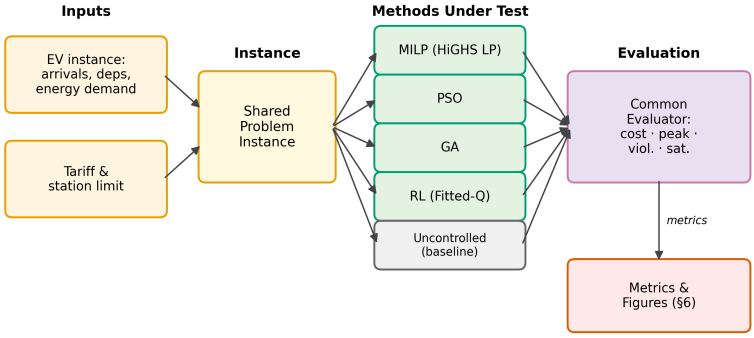


Fig. 1. Unified benchmarking framework. A shared instance and a common evaluator ensure MILP, PSO, GA, the RL agent, and the uncontrolled baseline are scored on identical data, constraints, and metrics.

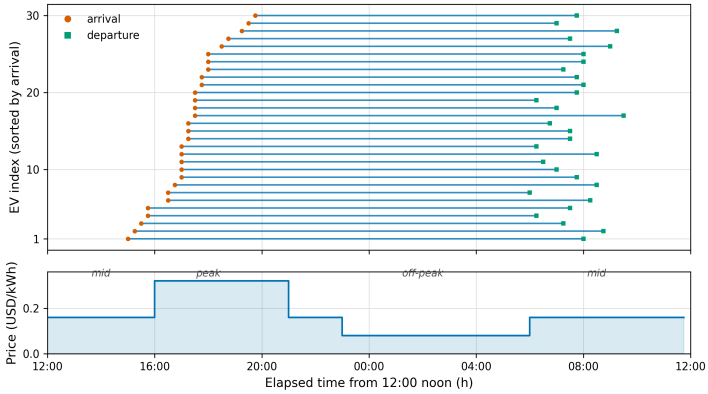


Fig. 2. The benchmark instance. Top: arrival (circles) and departure (squares) windows of the 30 EVs, sorted by arrival. Bottom: the three-tier time-of-use
tariff over the horizon, with the off-peak trough overlapping every vehicle's overnight stay.

5.2 Implementation, Parameters and Metrics

All methods are implemented in Python 3.12 using NumPy, SciPy (HiGHS for the LP), and scikit-learn (the MLP regressor for the value function), single-threaded on a standard CPU with no GPU. A fixed random seed governs instance generation and every stochastic method, so the study is reproducible. Table 1 lists the parameters. Three metrics are applied identically to every solution: **cost** (objective *J*, reported absolutely, as a reduction over the uncontrolled baseline, and as an optimality gap against MILP); **computational overhead** (wall-clock time, separating one-off training from per-instance inference for RL); and **constraint satisfaction** (percentage of vehicles fully charged by departure, and the number of station-limit violations). A method is credible only if it scores well on all three at once.

Table 1. System and algorithm parameters used throughout the study.

Parameter	Symbol	Value
Number of EVs	N	30
Scheduling horizon	T·Δt	24 h (96 slots)
Slot duration	Δt	0.25 h (15 min)
Max power per EV	P_max	7.2 kW
Station power limit	P_station	110 kW
Energy requirement	E_req	15–40 kWh
Off-peak / mid / peak price	c_t	$0.08 / 0.16 / 0.32 per kWh

PSO swarm / iterations	—	50 / 150
PSO inertia / c1 / c2	—	0.72 / 1.49 / 1.49
GA population / generations	—	60 / 150
GA mutation probability	—	0.05
Fitted-Q network	—	2×64 hidden, ReLU
Learning rate / discount	γ	1×10^{-3} / 0.98
Training episodes / ε	—	400 / 1.0→0.02

6 Results and Discussion

Results are organised around the objective of Section 3. Section 6.1 evaluates cost; Section 6.2 constraint satisfaction; Section 6.3 computational overhead; Section 6.4 convergence; and Section 6.5 the combined picture. Every figure is generated from the solved schedules; headline numbers appear in Table 2.

6.1 Cost Reduction

Against the uncontrolled baseline of $**239.02**, solving the linear program exactly yields the global optimum of $**73.31** — a **69.33%** reduction. Because every vehicle's stay straddles the off-peak trough, almost the entire bill can be avoided by deferring charging from the evening peak to the small hours, and the optimiser exploits this fully. The Fitted-Q agent captures most of the same saving, reaching $**80.81** (a **66.19%** reduction), just **10.23%** above the optimum. The metaheuristics trail: PSO settles at $**100.88** and GA at $**111.11**, leaving gaps of **37.59%** and **51.55%**. These figures flatter the metaheuristics, since part of their apparent cheapness comes from not delivering all the energy requested, as Section 6.2 shows.

The mechanism is visible in the dispatch profiles of Fig. 3. The uncontrolled policy charges on arrival and spikes to **187.2 kW**, far above the **110.0 kW** limit. MILP and the RL agent flatten the profile and shift the bulk of the energy into the off-peak window, saturating the station limit overnight when electricity is cheapest. PSO and GA shift load only partially, leaving ragged profiles — precisely why their costs are higher. The agent reproduces the optimiser's strategy without ever being told the price schedule analytically.

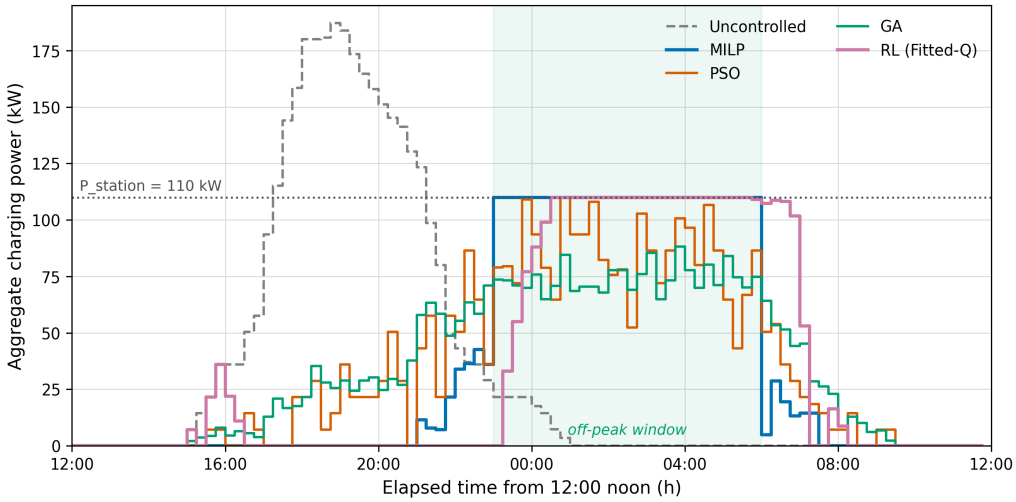


Fig. 3. Aggregate station load. Uncontrolled charging (grey) spikes to 187 kW; MILP (blue) and RL (pink) shift load into the off-peak window and respect the 110 kW limit, while PSO (orange) and GA (green) shift only partially.

6.2 Constraint Satisfaction

Constraint (2) requires every vehicle to leave fully charged and (4) forbids exceeding the station limit. Here the methods diverge most sharply. MILP and the Fitted-Q agent both satisfy the energy deadline for **100%** of the fleet and never breach the station limit. PSO and GA fully charge only **43.3%** and **56.7%** of vehicles. The uncontrolled baseline charges everyone but overloads the station in **16** slots — infeasible in practice. Fig. 4 resolves the per-vehicle picture: MILP and RL deliver 100% to essentially every vehicle, whereas PSO and GA fall well short for a substantial fraction, some receiving little more than 60% of what they requested.

The difficulty is the dimensionality of the coupled feasible region. With 30 vehicles over 96 slots the search space has thousands of dimensions, and the energy-delivery equalities (2) define a thin feasible manifold that penalty-based search struggles to stay on. MILP enforces (2) as a hard equality and the RL agent enforces it through a dispatch rule plus a large shortfall penalty — structural devices the metaheuristics, as commonly formulated, lack. The practical lesson is that the popularity of PSO and GA does not translate into reliable feasibility at fleet scale without substantial problem-specific repair operators.

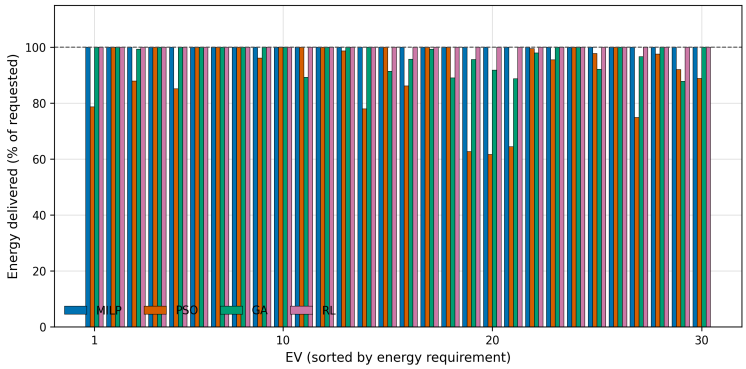


Fig. 4. Per-vehicle energy delivered as a percentage of the request, sorted by requirement. MILP (blue) and RL (pink) meet essentially every deadline; PSO (orange) and GA (green) under-deliver to a large fraction of the fleet.

6.3 Computational Overhead

Computational cost separates the methods along a different axis. MILP returns the global optimum in **0.036 s**; the sparse structured linear program is dispatched in milliseconds. PSO and GA take **1.13 s** and **2.67 s**, two to three orders of magnitude slower and still short of feasibility. The RL agent is most expensive to prepare — training takes **28.0 s** — but that cost is paid once, offline. At deployment the trained policy produces a complete 24-hour schedule in **0.029 s**, comparable to MILP's solve time and without a forecast or fresh optimisation. The accounting thus depends on the use case: for a one-shot day-ahead plan with a trusted forecast, MILP is unbeatable; for repeated real-time decisions, RL amortises its training into near-free inference, whereas MILP and the metaheuristics must re-solve from scratch each time.

6.4 Convergence and Learning Dynamics

Fig. 5 contrasts how the iterative methods approach their solutions. The metaheuristic traces (left) show the familiar shape of population search: a rapid early drop followed by a long, slow tail. PSO descends faster than GA and reaches a lower final value, but neither comes close to the MILP optimum (dashed) within the iteration budget. The RL training curve (right) is volatile and deeply negative early on, when the exploring agent strands vehicles and pays the shortfall penalty; as ε decays and the value function sharpens, reward climbs and stabilises. The dip-then-recover shape is characteristic of value-based learning under a heavy constraint penalty — the agent must first discover that under-charging is catastrophic before it learns when to charge cheaply.

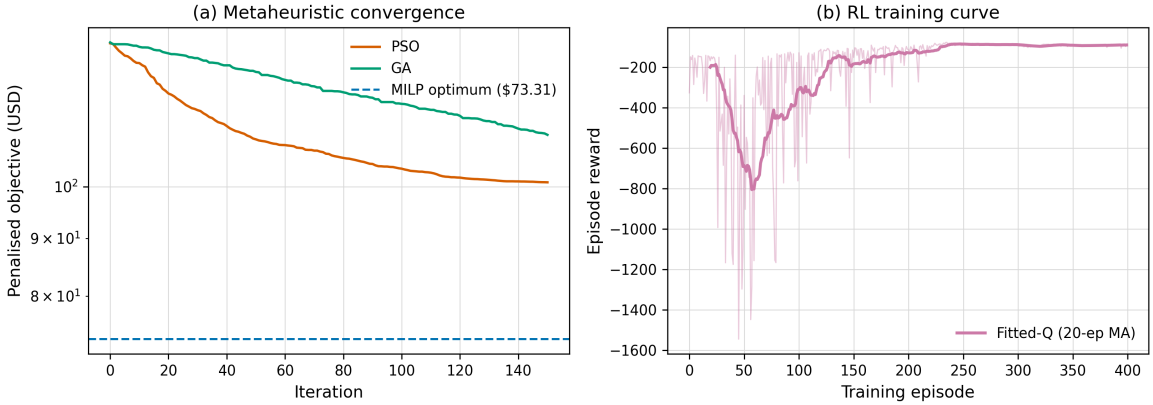


Fig. 5. Convergence behaviour. (a) PSO and GA penalised-objective traces against the MILP optimum (dashed); both plateau above it. (b) RL episode reward (raw and 20-episode moving average); reward recovers from an early exploratory dip and stabilises.

6.5 Combined Comparison

Table 2 collects the three axes for all methods. No single method dominates on every axis, but the methods occupy clearly separated regions of the trade-off space, plotted in Fig. 6 (optimality gap versus runtime, marker size proportional to constraint satisfaction). MILP sits in the ideal corner under the assumption of a perfect forecast and a re-solve per instance. The RL agent sits at low gap and full feasibility, at a preparation cost that collapses to near-zero at deployment. PSO and GA occupy the unfavourable middle: moderate runtime, large gaps, and small markers signifying poor feasibility.

Table 2. Headline comparison across the three evaluation axes. Best feasible values in bold; the optimality gap is measured against the MILP optimum.

Method	Cost ($)	Reduction	Gap	Peak (kW)	Charged
Uncontrolled	239.02	—	—	187.2	100.0%
MILP	**73.31**	**69.33%**	**0.00%**	110.0	**100.0%**
PSO	100.88	57.80%	37.59%	109.9	43.3%
GA	111.11	53.51%	51.55%	88.3	56.7%
RL (Fitted-Q)	**80.81**	**66.19%**	**10.23%**	110.0	**100.0%**

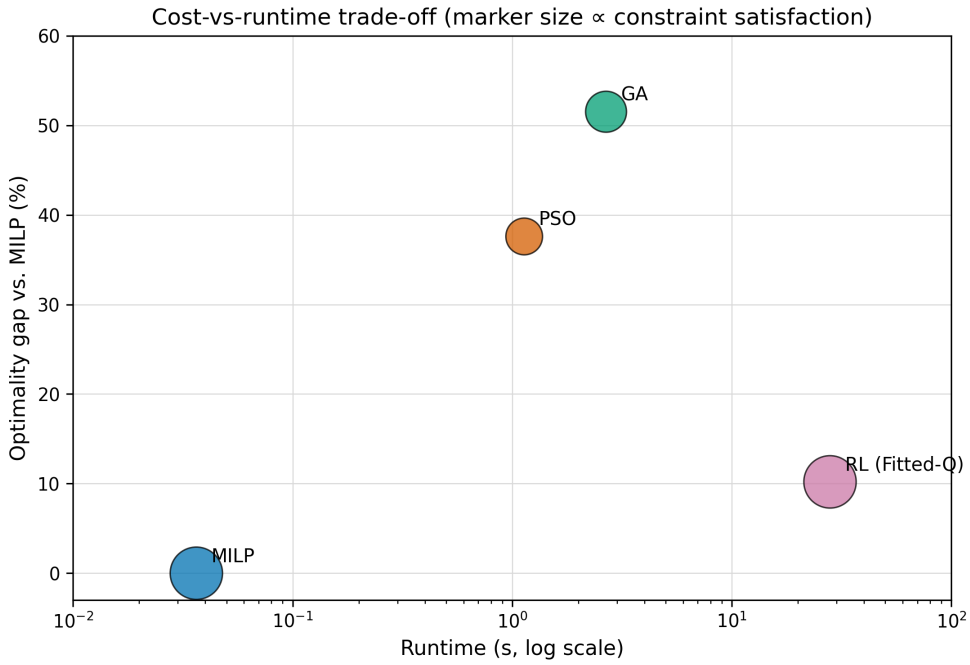


Fig. 6. Cost-versus-runtime trade-off; marker area is proportional to the percentage of vehicles fully charged. MILP and RL combine small gaps with full feasibility; PSO and GA show larger gaps and smaller, less-feasible markers.

Two qualifications keep these conclusions honest. First, the metaheuristics were run with standard, untuned operators and penalty-based constraint handling; bespoke repair operators would narrow their gaps, at the price of the problem-specific engineering their proponents cite as a drawback of exact methods. Second, MILP's dominance rests on a perfect forecast — once arrivals or prices become uncertain, its open-loop optimum degrades and the forecast-free RL policy becomes comparatively more attractive. The benchmark measures an idealised deterministic instance; the relative standings under uncertainty are exactly what the released protocol is designed to let others probe.

7 Conclusion

This paper put deep reinforcement learning, particle swarm optimisation, a genetic algorithm, and mixed-integer linear programming on the same 30-EV charging instance under identical constraints and metrics, replacing the incomparable claims of the existing literature with one controlled experiment. Solved exactly, MILP achieved the global optimum of $**73.31** (a **69.33%** reduction) in **0.036 s** with full constraint satisfaction; the Fitted-Q agent came within **10.23%** of that optimum while charging every vehicle on time and deciding in **0.029 s** after offline training; and the untuned metaheuristics satisfied the energy-delivery requirement for only **43.3%** (PSO) and **56.7%** (GA) of the fleet. The right method depends on the deployment regime rather than on any blanket superiority: exact programming wins under a trustworthy forecast, learned policies win when fast forecast-free decisions recur across many days, and off-the-shelf metaheuristics need substantial constraint-handling engineering before they are competitive at fleet scale. The study's limitations — a single deterministic instance, a continuous-power relaxation, and no vehicle-to-grid discharge — point to the natural next steps: stress-testing under stochastic arrivals and price forecasts, extending to bidirectional V2G and battery-degradation costs, scaling to hundreds of vehicles with multiple coupled transformers, and benchmarking continuous-action agents such as soft actor-critic against the value-based agent used here. The open harness and metric logs are provided so these extensions can be measured on the same footing.

References

1. Wang, J., Zhou, Y., et al.: Impact of electric vehicle charging demand on clean energy regional power grid control. Energy Inform. 8, 38 (2025). https://doi.org/10.1186/s42162-025-00538-0
2. Al-Saadi, M., Bhattacharyya, B., Al-Greer, M.: A review of the electric vehicle charging technology, impact on grid integration, policy consequences, challenges and future trends. Energy Rep. 12, 5571–5594 (2024). https://doi.org/10.1016/j.egyr.2024.10.062
3. Diaz-Londono, C., Ruiz, F., Gruosso, G.: Optimal strategy for electric vehicle parking lot energy management using LP and MILP techniques. Sustain. Energy Grids Netw. 36, 101176 (2023). https://doi.org/10.1016/j.segan.2023.101176
4. Bignucolo, F., Bertoldi, A., Coppo, M., Turri, R.: Minimum-time MILP-based EV charging scheduling for prosumers. Electr. Power Syst. Res. 247, 111819 (2025). https://doi.org/10.1016/j.epsr.2025.111819
5. Cvetkovic, M., Pavic, I., Capuder, T.: Electric vehicle charging station power supply optimization with V2X capabilities based on mixed-integer linear programming. Sustainability 15(22), 16073 (2023). https://doi.org/10.3390/su152216073
6. Datta, P., Bhakar, R.: Energy regulation of EV using MILP for optimal operation of incentive based prosumer microgrid with uncertainty modelling. Int. J. Electr. Power Energy Syst. 131, 107094 (2021). https://doi.org/10.1016/j.ijepes.2021.107094
7. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings of the IEEE International Conference on Neural Networks, vol. 4, pp. 1942–1948 (1995). https://doi.org/10.1109/ICNN.1995.488968
8. Cui, Z., Yuan, K., Lou, Q.: Particle swarm optimisation for scheduling electric vehicles with microgrids. In: 2020 IEEE/IAS Industrial and Commercial Power System Asia, pp. 1–6 (2020). https://doi.org/10.1109/ICPSAsia48933.2020.9208476
9. Yang, L., Liu, Y., Sun, J., et al.: Multi-objective comprehensive charging/discharging scheduling strategy for electric vehicles based on the improved particle swarm optimization algorithm. Front. Energy Res. 9, 811964 (2022). https://doi.org/10.3389/fenrg.2021.811964
10. Tariq, M.A., Kazmi, S.A.A., Aman, M.M., et al.: A two-stage multi-agent EV charging coordination scheme for maximizing grid performance and customer satisfaction. Sensors 23(5), 2925 (2023). https://doi.org/10.3390/s23052925
11. Kumar, B.A., Jyothi, B., Singh, A.R., et al.: Hybrid genetic algorithm-simulated annealing based electric vehicle charging station placement for optimizing distribution network resilience. Sci. Rep. 14, 7637 (2024). https://doi.org/10.1038/s41598-024-58024-8
12. Bouakkaz, A., Gil Mena, A.J., Haddad, S., Ferrari, M.L.: Optimal electric vehicle charging and discharging scheduling using metaheuristic algorithms: V2G approach for cost reduction and grid support. J. Energy Storage 90, 111935 (2024). https://doi.org/10.1016/j.est.2024.111935
13. Boglou, V., Karavas, C.S., Karlis, A., Arvanitis, K.: An intelligent decentralized energy management strategy for the optimal electric vehicles charging in low-voltage islanded microgrids. Int. J. Energy Res. 46(3), 2988–3009 (2022). https://doi.org/10.1002/er.7358
14. Qiu, D., Wang, Y., Hua, W., Strbac, G.: Reinforcement learning for electric vehicle applications in power systems: a critical review. Renew. Sustain. Energy Rev. 173, 113052 (2023). https://doi.org/10.1016/j.rser.2022.113052
15. Mnih, V., Kavukcuoglu, K., Silver, D., et al.: Human-level control through deep reinforcement

learning. Nature 518(7540), 529–533 (2015). https://doi.org/10.1038/nature14236
16. Lillicrap, T.P., Hunt, J.J., Pritzel, A., et al.: Continuous control with deep reinforcement learning. In: Proceedings of the 4th International Conference on Learning Representations (ICLR) (2016). https://doi.org/10.48550/arXiv.1509.02971
17. Haarnoja, T., Zhou, A., Abbeel, P., Levine, S.: Soft actor-critic: off-policy maximum entropy deep reinforcement learning with a stochastic actor. In: Proceedings of the 35th International Conference on Machine Learning (ICML), pp. 1861–1870 (2018)
18. Lin, Y., Wei, R., Wang, F., et al.: CASA: cost-effective EV charging scheduling based on deep reinforcement learning. Neural Comput. Appl. 36, 11497–11512 (2024). https://doi.org/10.1007/s00521-024-09530-3
19. Chen, G., Shi, X.: A deep reinforcement learning-based charging scheduling approach with augmented Lagrangian for electric vehicle. arXiv:2209.09772 (2022). https://doi.org/10.48550/arXiv.2209.09772
20. Sweeney, C., Vance, R., Soares, T.: Optimal scheduling of electric vehicle charging with deep reinforcement learning considering end users flexibility. arXiv:2310.09040 (2023). https://doi.org/10.48550/arXiv.2310.09040
21. Li, H., Wan, Z., He, H.: Constrained EV charging scheduling based on safe deep reinforcement learning. IEEE Trans. Smart Grid 11(3), 2427–2439 (2020). https://doi.org/10.1109/TSG.2019.2955437
22. Sharma, S., Panda, K.: Moving horizon-based optimal scheduling of EV charging: a power system-cognizant approach. arXiv:2003.12192 (2020). https://doi.org/10.48550/arXiv.2003.12192
23. Yin, Y., Xie, B., Chen, L., et al.: Electric vehicle charging and discharging scheduling strategy based on local search and competitive learning particle swarm optimization algorithm. J. Energy Storage 42, 103154 (2021). https://doi.org/10.1016/j.est.2021.103154
24. Goldberg, D.E., Holland, J.H.: Genetic algorithms and machine learning. Mach. Learn. 3, 95–99 (1988). https://doi.org/10.1023/A:1022602019183
25. Liu, Y., Wang, J., Li, M.: Optimal scheduling method for electric vehicle charging and discharging via Q-learning-based particle swarm optimization. Energy 318, 134925 (2025). https://doi.org/10.1016/j.energy.2025.134925
26. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. 2nd edn. MIT Press, Cambridge (2018)
27. Lee, S., Choi, D.-H.: Multi-agent deep reinforcement learning approach for EV charging scheduling in a smart grid. Appl. Energy 326, 119942 (2022). https://doi.org/10.1016/j.apenergy.2022.119942
28. Shojaeighadikolaei, A., Ghasemi, A., Jones, K., et al.: Distributed energy management and demand response in smart grids: a multi-agent deep reinforcement learning framework. arXiv:2211.15858 (2022). https://doi.org/10.48550/arXiv.2211.15858
29. Alfaverh, F., Denai, M., Sun, Y.: Optimal vehicle-to-grid control for supplementary frequency regulation using deep reinforcement learning. Electr. Power Syst. Res. 214, 108949 (2023). https://doi.org/10.1016/j.epsr.2022.108949
30. Huangfu, Q., Hall, J.A.J.: Parallelizing the dual revised simplex method. Math. Program. Comput. 10(1), 119–142 (2018). https://doi.org/10.1007/s12532-017-0130-5