SIRA: SELF-IMPROVING RECOMMENDATION AGENT UNDER DEMAND SHIFTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Conversational agents increasingly sell products and services, and to be profitable, an agent must (i) find out what each customer wants, (ii) offer what the customer will buy, and (iii) complete the purchase before the customer loses patience. No benchmark measures profit in sales conversations that require all three. We introduce AEROSELL, a benchmark that scores the profit an airline keeps when an agent sells flight add-ons to simulated customers. Each customer has a hidden type that determines what they buy, so the agent must learn which questions reveal the type and what each type buys as demand shifts. Each turn, the agent makes one of three tool calls, to ask a question, offer an add-on, or complete the purchase. Unlike language-model customers, which can be persuaded, AEROSELL customers react only to these calls, making results reproducible and the best profit exactly computable. Neither prompting a language model nor fine-tuning it with standard recipes makes it a profitable seller. A hand-written sales strategy does better by asking for the facts each add-on requires and keeping a turn to complete the purchase, but ranks offers by revenue rather than profit. We propose SIRA (Self-Improving Recommendation Agent), which imitates this strategy, then corrects its objective with reinforcement learning on the profit of its own conversations. SIRA (i) earns 110.7% more profit than the prompted model, (ii) beats the strategy it imitates in every demand phase and by 14.5% overall, with the same test-time information, and (iii) learns which add-ons earn more, coming within 3.4% of the same strategy given product costs. Yet an agent that knows how customers are generated but not their price limits could capture at least 68% of the add-on profit of an agent with this privileged information, and SIRA captures 37%.

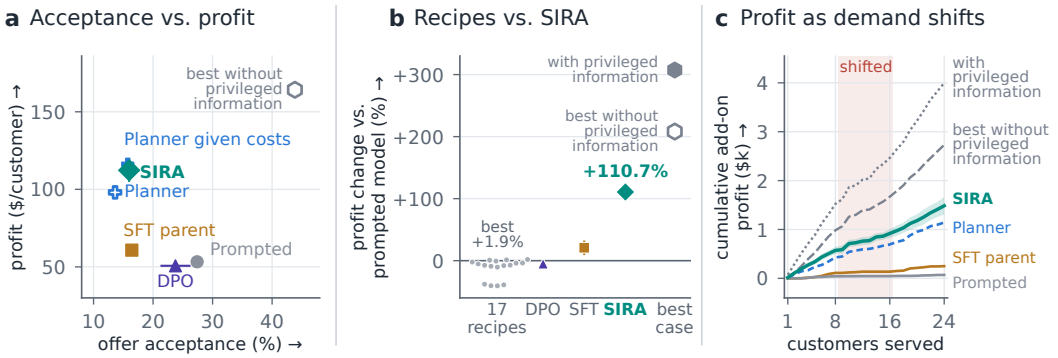

Figure 1: **SIRA more than doubles profit, where standard training recipes do not.** **(a)** Offer acceptance against profit per customer; the prompted model's offers are accepted more often than SIRA's, yet SIRA earns twice its profit. **(b)** Profit change relative to the prompted model; gray points are 17 recipes from two preliminary studies, each relative to its own prompted model (Appendix E). **(c)** Cumulative add-on profit over 24 customers (mean over 12 streams; demand shifts for customers 9 to 16); gray marks show an agent that sees each customer's price limits and the best agent we found without them. Bars and bands: SD over three runs.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

1 INTRODUCTION

Conversational agents increasingly help customers decide what to buy; Amazon's shopping assistant, for example, answers product questions and makes recommendations inside its store (Mehta & Chilimbi, 2024). For a seller, such an agent should maximize profit, which the seller earns only when a purchase is completed. Consider an airline assistant that sells add-ons, such as checked bags, seats, and lounge access, while a traveler books a flight. To be profitable, it must find out what the traveler needs, offer add-ons the traveler will buy and the airline keeps most of the price of, and complete the purchase before the traveler runs out of patience. These goals conflict, as each question reveals a need but uses a turn that could carry an offer, an accepted add-on earns nothing if the purchase is never completed, and add-ons that sold last month can stop selling when demand shifts.

Existing benchmarks do not measure this trade-off. Tool–agent–user benchmarks such as τ-bench score whether an agent completes a customer's service request under domain policies (Yao et al., 2025; Barres et al., 2025), and multi-round conversational recommenders ask about item attributes until they recommend the one item a simulated user wants (Lei et al., 2020a;b; Deng et al., 2021). Sales-dialogue benchmarks score persuasive progress and buying intent (Su et al., 2026), business simulations score profit over long horizons (Backlund & Petersson, 2025; Fan et al., 2026), and PrefBench scores seller profit in price negotiations (Lei, 2026). None scores the profit of sales conversations in which an agent must decide which needs to ask about, which of many add-ons to offer, and when to complete the purchase.

We introduce AEROSELL to measure this. Each simulated traveler belongs to one of 12 hidden types, such as a student on a budget, a family with an infant, or a VIP frequent flyer, and the type determines which of 23 add-ons the traveler pays more than list price for (Figure 8 in Appendix A). The agent never sees the type but can ask questions, each revealing one fact such as the trip purpose or the number of bags; facts also unlock offers, since an add-on can be offered only once the facts it depends on are known, and a traveler buys an offered add-on exactly when its list price does not exceed a hidden limit. A good seller therefore identifies the type with few questions and fills the remaining turns with the add-ons of highest expected profit. Each turn, the agent writes a message with one tool call, asking one question, offering one add-on, or completing the purchase, and the traveler replies with the engine's result, so wording never changes an outcome, every result is reproducible, and a full-information bound on profit is exactly computable. Every move uses one turn of a patience budget of 5 to 16 turns; a move the rules forbid, an unreadable response, or running out of patience ends the conversation, and the traveler then buys only the ticket. The airline earns $50.40 on each ticket and keeps 70–90% of the price of seats and bags but only 16–18% of the hotel and car-rental bundles it resells. For eight of every 24 travelers demand shifts, which changes what each type pays for, and every agent meets the same travelers in the same order.

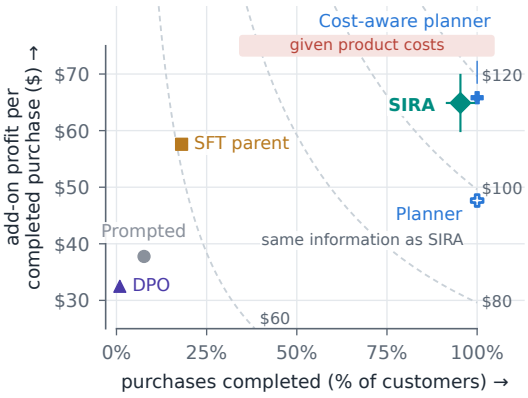

Figure 2: **Profit needs completed purchases and add-ons that earn more.** Dashed curves join equal profit per customer. Learned agents are `Qwen3.5-9B`; the planner (code) sees the same public information as SIRA, and only the cost-aware planner is given product costs.

Prompted models rarely complete a purchase on AEROSELL, and standard training does not fix this. A prompted `Qwen3.5-9B` ends 90 of 100 conversations on an invalid response, most often an offer made before asking the fact it depends on or a question about several topics at once. In two preliminary studies with their own customers and action formats, none of 17 standard recipes, spanning imitation of logged dialogues, preference optimization, and reinforcement learning with outcome and process rewards such as offline GRPO and DuCA (Shao et al., 2024; Yang et al., 2026), improved profit by more than 2% (Figure 1b). We attribute this to three causes: (i) imitation copies the errors in logged dialogues, (ii) process rewards credit offers that are lost when the purchase is never completed, and (iii) agents trained on symbolic actions do not transfer to generated tool calls, the one cause we measure directly (Appendix E).

These failures suggest learning competent selling from a teacher and the objective from outcomes. A hand-written sales strategy, which we call the planner, is a natural teacher, as it asks for the facts that unlock the add-ons with the highest expected revenue, keeps a turn to complete the purchase, and never breaks a rule, but it ranks offers by revenue rather than profit. We propose SIRA, a Self-Improving Recommendation Agent, which (i) fine-tunes `Qwen3.5-9B` to imitate the planner's decisions, written as complete messages and tool calls, including in situations the planner never reaches, and (ii) improves itself with reinforcement learning on the profit of 256 conversations it generates with training travelers, without new labels. Because the reward is profit while the planner ranks by revenue, SIRA can surpass the planner that taught it with the same information. All learning happens during training, and at test time SIRA sells to new travelers across the demand shift with fixed weights.

We evaluate on 288 test travelers in 12 streams, with three reinforcement-learning runs. SIRA earns 14.5% more profit than the planner that taught it and leads in every demand phase, although both see the same information at test time (Section 5.1). It earns 110.7% more than the prompted model, and 58.9% more even when both can make only moves the rules allow, so its gain is not only the avoidance of invalid responses (Section 5.2). Profit above the ticket is the product of how often an agent completes the purchase and what each completed purchase earns, and each baseline falls short on a different factor (Figure 2). The prompted and imitation models rarely complete the purchase, and the planner completes every purchase but sells hotel and car bundles of which the airline keeps little. SIRA does well on both and comes within 3.4% of a planner that is given product costs, which SIRA is never given. The benchmark is far from solved: an agent that follows the same rules and knows how travelers are generated and the current demand, but not their price limits, captures at least 68% of the add-on profit of an agent that sees these limits, and SIRA captures 37% (Figure 9). A pre-registered replication on 360 travelers that no one had evaluated before confirms every main comparison, also when agents can make only moves the rules allow. Our contributions are:

(1) **AEROSELL**, a benchmark that scores the profit of each sales conversation with travelers of hidden types, one fact per question, hidden price limits, demand shifts, and an exact profit bound (Section 3).
(2) **SIRA**, which imitates a rule-based planner and then corrects its objective with reinforcement learning on the profit of 256 of its own conversations, surpassing that planner by 14.5% (Sections 4 and 5.1).
(3) **The sources of the gain**: valid moves, completing the purchase on the last turn, and offers the airline earns more on. Standard recipes stay near or below prompting, and planner advice in the prompt and a 3× larger model close only part of the gap (Sections 5.2 and 5.3).

2 RELATED WORK

Conversational recommendation and economic agents. Conversational recommenders elicit preferences and select items through dialogue (Christakopoulou et al., 2016; Li et al., 2018). Multi-round conversational recommenders choose between asking about an item attribute and recommending items to a rule-based simulated user who accepts only its one target item (Lei et al., 2020a;b; Deng et al., 2021). AEROSELL keeps this ask-or-offer structure but sells several items that differ in profit and requires the agent to complete the purchase. Agent benchmarks have since added tools and simulated users (Table 4 in Appendix F): WebShop evaluates buyer-side purchasing (Yao et al., 2022), and Vending-Bench and E-Commerce Bench measure profit from long-horizon business operation with time-varying demand (Backlund & Petersson, 2025; Fan et al., 2026). Dialogue agents trained with RL to negotiate or persuade optimize deal value or donations (Lewis et al., 2017; He et al., 2018; Wang et al., 2019), LLM negotiators are now compared on price negotiation (Bianchi et al., 2024), and revenue management studies pricing and assortment under uncertain demand (Talluri & van Ryzin, 2004). Scoring each conversation, with its turn budget, exposes what long-horizon profit hides: purchases an agent never completes and conversations ended by invalid tool calls.

LLM sellers scored by profit. PrefBench scores zero-shot LLM sellers by profit in price negotiations with hidden willingness to pay and finds that they close almost every deal yet earn little more than a random seller (Lei, 2026). Reinforcement learning on verifiable outcomes trains a seller that bargains with several buyers better than frontier models (Liu et al., 2026), and in Bazaar's auctions the LLM merchants that win the most customers are often not the most profitable (Ahmed et al.,

2026). These works study how an agent sets a price; we fix list prices and study which needs an agent elicits, which add-ons it offers, and whether it completes the purchase.

Learning from experts and from self-generated experience. Policy distillation, Expert Iteration, and DAgger learn from an expert's actions (Rusu et al., 2016; Anthony et al., 2017; Ross et al., 2011), and STaR, ReST, and ReST^{EM} reuse filtered model-generated samples (Zelikman et al., 2022; Gulcehre et al., 2023; Singh et al., 2024). SIRA imitates a fixed planner, then improves on conversations it generates itself, rewarded by profit rather than by agreement with the planner.

Reinforcement learning for language agents. SIRA's second stage builds on REINFORCE with a leave-one-out baseline (Williams, 1992; Kool et al., 2019; Ahmadian et al., 2024), in contrast to clipped PPO and group-normalized GRPO (Schulman et al., 2017; Shao et al., 2024) and to preference optimization (Rafailov et al., 2023). Multi-turn RL trains language agents on whole interactions (Zhou et al., 2024; Abdulhai et al., 2023); recent work studies rollout design (Wang et al., 2025), step-level critics that use training-time information (Zhou et al., 2025), and step-level group advantages (Feng et al., 2025). SIRA uses the simplest such signal, the profit of each complete conversation, and needs only 256 conversations per run. Process supervision rewards intermediate steps (Uesato et al., 2022; Lightman et al., 2024), and AgentPRM and DuCA extend dense feedback to agents and sales dialogue (Choudhury, 2025; Yang et al., 2026); in AEROSELL, dense rewards for accepted offers can disagree with profit (Appendix E). Constrained decoding can enforce output grammars (Willard & Louf, 2023; Geng et al., 2023); we use it only as a control that removes validity from the comparison (Section 5.2).

3 AEROSELL: SCORING THE PROFIT OF EACH SALES CONVERSATION

Like τ-bench (Yao et al., 2025; Barres et al., 2025), AEROSELL has an agent serve a simulated user through tool calls and scores the final state of the environment rather than the agent's text. It adds hidden willingness to pay, a different profit share for each add-on, and demand shifts.

Customers and observations. Each customer belongs to one of 12 hidden types and has six private facts (such as trip purpose, party, and bags), a willingness to pay w_p for each of 23 add-ons, and a turn budget of $T \in [5, 16]$ paid turns. The agent sees only public information: the catalogue with list prices L_p and the facts each add-on requires, the customer's account, the facts disclosed so far, its accepted and refused offers, and up to eight earlier customers of the same stream, including their profits, which partly reveal costs. Every paid tool result reports the remaining turns. Costs c_p, willingness to pay, the type, and the demand phase stay hidden. The prompt serializes the public state, the dialogue so far, and the legal action menu (Appendix A).

How customers behave. A customer's hidden type fixes its trip purpose and party and determines which add-ons it pays more than list price for, with individual variation, and 55% of customers also want one to three add-ons outside their type (Figure 8 in Appendix A). A question reveals one fact, and the customer replies with the engine's structured result rather than with text; Table 2 shows one complete conversation.

Actions and profit. Two free tool calls return the account and the catalogue; every later tool call uses one of the T paid turns. Each turn is one customer-facing message with exactly one tool call: ASK discloses one fact, OFFER proposes an eligible add-on not yet accepted or refused at its list price, and BOOK completes the purchase and ends the conversation (Figure 3). An add-on becomes eligible only after the facts it requires are disclosed, so questions unlock offers but spend turns. A customer accepts an offer exactly when $w_p \geq L_p$, so wording never changes a sale, and AEROSELL measures what an agent asks, offers, and completes. Each add-on has a profit share ϕ_p, the share of its price that the airline keeps, so $c_p = L_p(1 - \phi_p)$. With ticket profit $m_0 = \$50.40$, a conversation with accepted cart C earns

$$M = m_0 + \mathbf{1}[\textsc{Book}] \sum_{p \in C} (L_p - c_p). \tag{1}$$

A conversation ends in a purchase (BOOK), in a timeout (an invalid response or no turns left), which pays m_0 but loses the cart, or in a drop-off after repeated or irrelevant offers, which pays nothing and

which the legal menu prevents. Since no conversation drops off, mean profit factorizes into closing and selling:

$$\mathbb{E}[M] - m_0 = \Pr(\textsc{Book})\ \mathbb{E}[\,\text{add-on profit} \mid \textsc{Book}\,]. \tag{2}$$

A response is valid if it is one JSON object with a message and one tool call from the legal action menu, within 256 tokens. As in a production system that rejects malformed requests, the first invalid response ends the conversation as a timeout, with no repair, so every executed action is the one the agent chose; a restricted-decoding control removes this failure mode (Section 5.2).

Demand shifts. Demand for add-ons changes over time, for example with the seasons. Each stream serves 24 new customers who arrive one after another: eight under initial demand, eight under shifted demand, and eight under the initial demand again. Under the shift, for about half of the (trip purpose, party) combinations, seat, cabin, and airport add-ons gain value and all other add-ons lose it; for the other half the roles reverse. The agent never observes the phase and can notice a change only through the outcomes of its own recent conversations. Because each phase holds eight customers and types arrive in a fixed rotation, each phase also meets a different set of eight of the 12 types; comparing the same types with and without the shift isolates the demand change (Appendix A). Every agent meets the same customers in the same order while building its own history, so comparisons within each phase are matched.

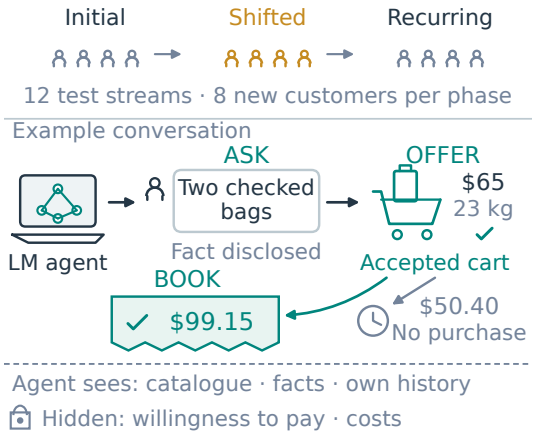

Figure 3: **A conversation in AEROSELL.** ASK reveals a need, an offer is accepted, and BOOK completes the purchase ($99.15 instead of $50.40).

Metrics and an exact reference. The primary outcome is profit per customer, $\frac{1}{N}\sum_i M_i$. We also report the sale rate (purchases with at least one add-on), the completion rate (purchases), the offer acceptance rate (accepted over executed offers), and the valid-response rate (Appendix C). With known willingness to pay and turn budget, the best agent that asks, offers at list price, and keeps one turn to complete the purchase can be computed exactly by enumerating the 2^6 fact sets (Appendix A). We call this full-information reference the agent with privileged information, because it sees each customer's price limits, which no deployed agent can. It earns $M^\star = \$217.06$ per customer on the test set, and we report the share of its add-on profit, $100(\text{profit} - m_0)/(M^\star - m_0)$, that each agent captures.

4 SIRA: IMITATE A PLANNER, THEN LEARN FROM PROFIT

SIRA trains a language agent in two stages that use the prompt and tool calls of the test (Figure 4): Stage 1 addresses causes (i) and (iii) of Section 1 by imitating a planner whose actions are valid, written as complete messages and tool calls, and Stage 2 addresses cause (ii) by learning from the profit of complete conversations the agent generates itself.

4.1 A PLANNER THAT KEEPS A TURN TO COMPLETE THE PURCHASE

The planner is code that uses only public information. For each add-on it estimates the chance of acceptance as a smoothed, recency-weighted acceptance rate among recent customers whose disclosed facts match the current customer, backing off to all past offers of that add-on, and it scores the offer by that chance times the list price. With matching past offers H_p of add-on p among the $n \le 8$ most recent customers, outcomes $a_{jp} \in \{0, 1\}$, and weights $\omega_j = 0.9^{n-1-j}$ that halve a customer's influence after about seven newer customers,

$$\widehat{q}_p = \frac{1 + \sum_{j \in H_p} \omega_j a_{jp}}{2 + \sum_{j \in H_p} \omega_j}, \qquad s_p = L_p\, \widehat{q}_p. \tag{3}$$

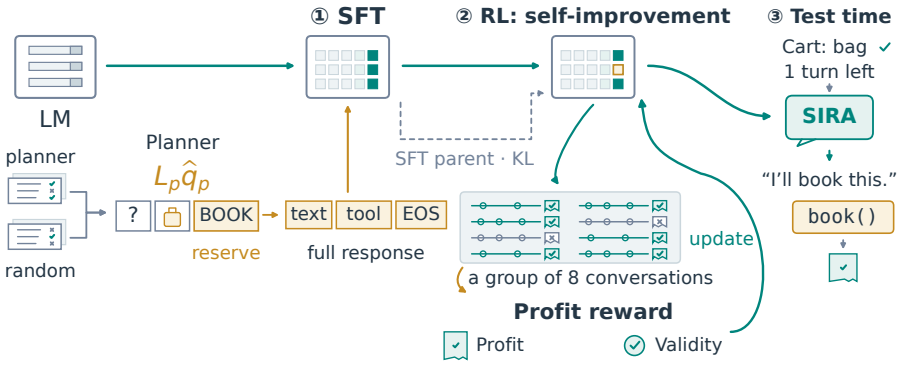

Figure 4: **SIRA.** ① Stage 1: the planner labels planner and random states with complete messages and tool calls, including the reserved BOOK. ② Stage 2: the agent generates a group of eight conversations from one customer state, receives their profit and a small validity bonus, and learns with leave-one-out advantages and a KL penalty toward the SFT parent. ③ Test time: public state in, message and tool call out (e.g., an accepted bag and one turn left lead to BOOK).

At every turn the planner (i) enumerates sets of missing facts, (ii) charges one turn per question, (iii) reserves the final turn for BOOK, (iv) fills the remaining turns with the highest-scoring eligible add-ons not yet accepted or refused, and (v) replans after each tool result. It is a strong heuristic, not an optimal policy, because it ranks offers by revenue rather than profit and ignores the value of information. A cost-aware planner, the same code scoring offers by $(L_p - c_p)\widehat{q}_p$, serves as a control.

4.2 IMITATING VALID DECISIONS

The planner visits only the states it reaches, so a scripted agent also asks and offers at random, and the planner labels every state of both kinds with its action, written as a message from fixed templates and an exact tool call. From 5,760 training customers we keep 8,192 labeled states (5,818 from planner conversations and 2,374 from random ones), chosen to cover every stream, demand phase, and action type. Supervised fine-tuning (SFT) with LoRA adapters (Hu et al., 2022) trains the agent to write each complete target response, message and tool call, so it learns valid actions rather than action labels (Appendix B).

4.3 LEARNING FROM PROFIT

Imitation inherits the planner's revenue ranking and cannot correct its own mistakes. Stage 2 therefore lets the agent generate a group of $G = 8$ complete conversations from one customer state and the planner's public history, which gives all eight the same context, scores each with the engine, and rewards conversation i with

$$r_i = \frac{M_i - m_0}{100} + 0.02\, v_i, \tag{4}$$

where v_i is its fraction of valid responses. Dividing by 100 keeps rewards near unit scale, and the bonus, worth \$2 of profit, adds a small incentive for valid responses. An accepted cart adds reward only if the agent completes the purchase, and nothing rewards asking, offering, or persuasive wording. The agent is updated with REINFORCE and a leave-one-out baseline (Williams, 1992; Kool et al., 2019; Ahmadian et al., 2024), $A_i = r_i - \frac{1}{G-1}\sum_{j \neq i} r_j$, with importance weights for the gap between generator and learner and a KL penalty ($\beta = 0.01$) toward the SFT parent (Algorithm 1). Each run learns from 32 groups, only 256 conversations.

5 EXPERIMENTS

Setup. All learned agents use `Qwen3.5-9B` (Qwen Team, 2026) with LoRA adapters and thinking mode off. Training, validation, and test customers come from disjoint generated populations; the test set has 288 customers in 12 streams (96 per demand phase), and its shift follows the training rule

Table 1: **SIRA more than doubles the prompted model's profit and beats the planner it imitates.** Profit per customer by demand phase, sale rate, and overall gain over the prompted model on the 288-customer test set (96 per phase). ±: SD over three training runs; the prompted model, the SFT parent (the seed-0 SFT run that all SIRA runs refine), and the planners are single agents. Bold: best agent without access to costs; the cost-aware planner is given product costs, and the full-information reference sees willingness to pay.

	Profit ($/customer)				Sale rate	Gain
Agent	Initial	Shifted	Recurring	Overall	(%)	(%)
Prompted	55.71	50.75	53.40	53.28	5.21	–
DPO	50.58 ± 0.31	50.57 ± 0.19	50.96 ± 0.96	50.70 ± 0.48	0.69 ± 0.92	−4.8
SFT (3 runs)	67.30 ± 6.79	58.19 ± 5.67	67.61 ± 6.50	64.37 ± 6.22	17.82 ± 7.49	20.8
SFT parent	64.32	53.56	64.51	60.80	12.50	14.1
SIRA (SFT→RL)	**121.88** ± 9.34	**93.68** ± 2.58	**121.29** ± 11.22	**112.28** ± 7.48	**64.24** ± 7.22	**110.7**
Planners (code)						
Planner	103.79	83.65	106.69	98.04	61.11	84.0
Cost-aware planner	129.98	93.46	125.20	116.21	65.28	118.1
Full-information ref.	240.74	168.07	242.37	217.06	–	–

at a smaller magnitude. The planner, the SFT data, the prompt, and all RL settings were fixed before the test set was generated, and the SFT parent before any test result; the test set informed only which RL variant we report (Appendix B), so a pre-registered replication on 360 untouched customers tests the result (Section 5.1). We compare the prompted model, whose prompt describes the tools and rules but does not state the profit goal (added instructions lift it by at most $3.0, Figure 10d); DPO (Rafailov et al., 2023) from the base model, trained on 2,048 preference pairs labeled by the profits of alternative actions computed with hidden customer data; the SFT parent (training seed 0) and SFT averaged over three seeds; and SIRA, three RL runs from the SFT parent. Decoding uses temperature 1, top-p 1, and 256-token completions (Appendix C).

5.1 SIRA MORE THAN DOUBLES PROFIT AND BEATS THE PLANNER IT IMITATES

SIRA earns 110.7% more than the prompted model and 84.7% more than its SFT parent, or 74.4% more than the mean SFT run, which itself adds 20.8% over prompting (Table 1). Paired over the 12 test streams, it gains $51.48 per customer over the SFT parent (95% CI [43.90, 59.06]), and every RL run beats the parent in every stream and in 107 of the 108 run–stream–phase cells (Figure 5a). DPO falls to the ticket profit (-4.8%), and SIRA's advantage holds in every phase, including the hardest, shifted demand.

SIRA also beats the planner it imitates, which completes every purchase and never produces an invalid call, by $14.24 per customer (95% CI [9.60, 18.88]), and it leads in all 12 streams and every phase. These intervals pool the three runs; the weakest run leads the planner by $6.42. The cost-aware planner, which scores offers by true profit, is not significantly better than SIRA ($3.93, 95% CI $[-1.72, 9.58]$). Because the profits in the history reveal costs, exactly so when pooled over the public training records, a planner that recovered costs from them would become this cost-aware planner (Appendix F). SIRA captures 37.1% of the full-information reference's add-on profit, 36.8–37.6% in every phase, compared with 28.6% for the planner and 39.5% for the cost-aware planner; the rest is headroom that AEROSELL makes measurable.

Much of this headroom is attainable without privileged information: a simple agent that follows the same rules and knows how customers are generated, the demand phase, and each customer's patience, but not the type or the price limits, earns $164.20 per customer, 68% of the reference's add-on profit (Figure 9).

The result replicates on 360 new customers in 15 streams: SIRA earns $120.95 ± 8.09 and beats its SFT parent by $54.72 (95% CI [47.97, 61.47]), the planner by $13.81 ([7.68, 19.95]), and the prompted model by $68.62 ([60.39, 76.86]), while the cost-aware planner is not significantly ahead ($5.11, $[-1.36, 11.58]$). The pre-registered primary test on 288 of these customers reaches the same

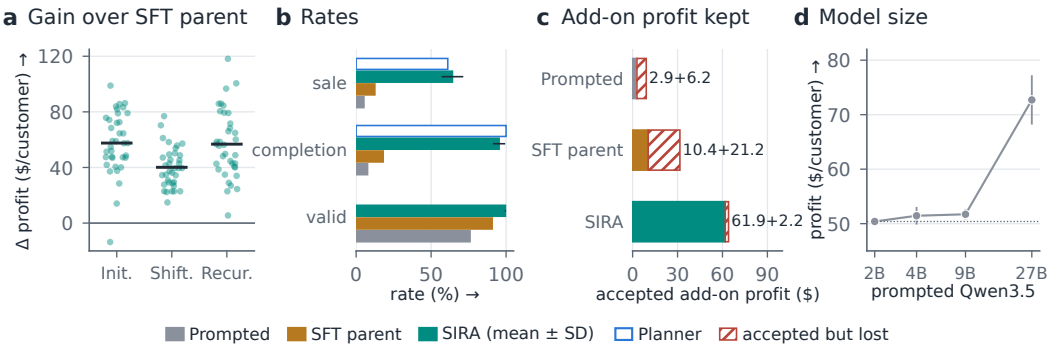

Figure 5: **SIRA completes the purchase and keeps the carts it sells. (a)** Gain over the SFT parent for each of 12 streams × 3 RL runs, by demand phase; bars mark means. **(b)** Sale, completion, and valid-response rates (the planner runs as code). **(c)** Accepted add-on profit per customer, completed (solid) or lost when the conversation ends without a purchase (hatched). **(d)** Prompted `Qwen3.5` models of four sizes in the scaling study (separate cohort; 95% CIs); dotted line: ticket profit.

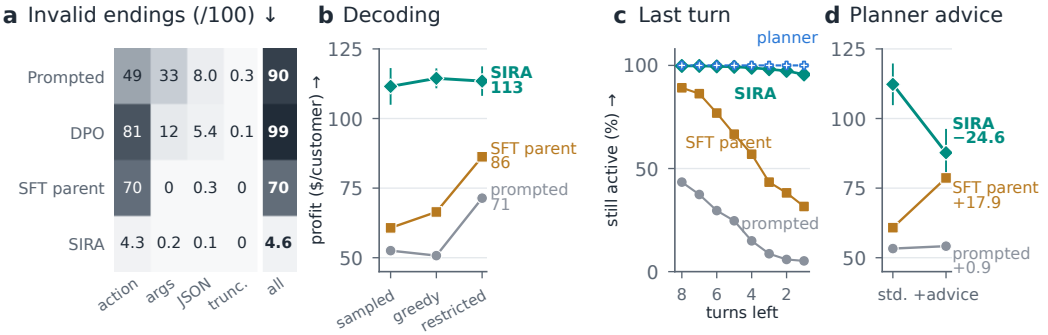

Figure 6: **The gain survives restricted decoding and comes from reaching the last turn. (a)** Invalid endings per 100 customers, by cause: illegal action (mostly premature offers), malformed arguments, malformed JSON, truncation. **(b)** Profit under sampling (re-run in the decoding harness), greedy decoding, and sampling restricted to valid actions (Appendix D). **(c)** Share of conversations still active with the given number of turns left, among customers whose budget allows it. **(d)** Profit with the standard prompt and with the planner's advice appended. Test set; SIRA pools its three runs, and bars show SD over runs.

four conclusions, and so does a rerun with restricted decoding, in which SIRA leads its parent by $33.08 and the planner by $17.35 in all 15 streams (Appendix D).

5.2 Three sources of the gain

Valid actions. The prompted model ends 90 of every 100 conversations on an invalid response (Figure 6a), mostly by offering an add-on before asking the fact it requires or by asking about several topics at once; DPO ends 99. The SFT parent nearly removes formatting errors but still ends 70 of every 100 conversations on invalid actions, 198 of its 203 invalid endings being premature offers. SIRA ends 4.6, and 99.6% of its responses are valid. Restricting every agent's decoding to valid actions removes these failures, yet SIRA still earns 58.9% more than the prompted model and 31.5% more than its parent, leading in all 12 streams and keeping 71% and 54% of its sampled advantage (Figure 6b, Appendix D).

Completed purchases. SIRA completes the purchase for $95.4 \pm 4.1\%$ of customers, compared with 18.1% for its parent and 7.6% for the prompted model (Figure 5b). Imitation already teaches the planner's timing: both complete purchases on the last turn (the parent in 51 of 52 cases, SIRA in 824 of 824), and the difference is reaching that turn. At the last turn, 95.6% of SIRA's conversations

are still active, compared with 31.6% of the parent's (Figure 6c), and SIRA completes the purchase in 99.8% of those, compared with 56.0%. In total the parent loses $21.18 of accepted add-on profit per customer, and SIRA loses $2.16 (Figure 5c).

Better offers. With decoding restricted, so that both agents act validly, SIRA earns $63.26 of add-on profit per completed purchase, compared with $46.30 for its parent (Appendix D); without restriction it nearly matches the cost-aware planner and far exceeds the planner (Figure 2). SIRA changes what it sells (Figure 11 in Appendix D). The planner and the SFT parent direct 26.3% and 33.3% of their offers to the hotel and car-rental bundles, whose profit shares are only 16–18%. SIRA directs 6.2% there, and the mean profit share of its offers rises to 56%, compared with 43% for the planner and 65% for the cost-aware planner. The profit reward, which depends on costs, moves SIRA toward the cost-aware planner's offers during training, although costs never appear in its prompt. Figure 12 in Appendix D follows two conversations turn by turn.

5.3 Other recipes, advice in the prompt, and scale do not replace training

Other training recipes. In the two preliminary studies (Figure 1b, Appendix E), ten free-form recipes changed profit by -40.6% to $+1.9\%$ relative to prompting, and seven symbolic-action recipes by -10.4% to $+1.2\%$. Moved to full dialogue on our test set, the seven symbolic-action recipes earn $51.96–53.28 per customer, none above the prompted model on average, and end 88–91 of every 100 conversations on invalid responses.

Advice in the prompt. Appending the planner's recommendation raises the SFT parent from $60.80 to $78.66, leaves the prompted model at $54.15, and lowers SIRA by 21.9% (Figure 6d): advice supplies a valid next action, cutting the SFT parent's invalid endings from 70.5 to 47.6 per 100 customers, but the unfamiliar prompt breaks SIRA's output format in 27 of every 100 conversations.

A larger model. In a scaling study registered before the test set existed (three streams, 2 degrees of freedom), prompted `Qwen3.5` models of 2B, 4B, and 9B parameters stay within $1.32 of the ticket profit, while the 27B model earns 40.6% more than the 9B model (95% CI [+$16.03, +$25.94]; Figure 5d), mainly by repairing validity: it ends 20 of every 100 conversations on an invalid response, compared with 90 for 9B.

6 Discussion

Limitations. AeroSell's customers are synthetic by design: they reply with the engine's result and a price threshold decides every offer, which makes results reproducible and the reference exact but leaves persuasion and language cues out of scope; evaluation with human customers is the natural next step (Seshadri et al., 2026). Customers never leave because of pushy offers, so a refused offer costs only a turn; the engine supports customers who leave after irrelevant or repeated offers, which would make deciding what not to offer part of the score. The test shift follows the training rule at a smaller magnitude and coincides with a change in which types arrive. SIRA's weights are fixed at test time, and removing or staling its history changes its profit by less than $3, which is not significant (Appendix F), so its gains in every phase reflect robust selling under shift rather than online adaptation. We train one open model family with one SFT parent, and test neither closed frontier models nor RL from the base model.

7 Conclusion

Profit in sales conversations needs both completed purchases and add-ons that earn more, and AeroSell scores both exactly. By imitating a rule-based planner and then learning from the profit of only 256 of its own conversations, SIRA completes the purchase for 95% of customers, sells add-ons the airline earns more on, and more than doubles profit, and these gains replicate on 360 untouched customers. Yet it captures 37% of the attainable add-on profit, where an agent that knows how customers are generated captures at least 68%: learning what each customer will buy is the open challenge that AeroSell makes measurable.

ETHICS STATEMENT

All customer profiles, catalogue entries, and prices are synthetic; the study uses no customer records and involves no human participants. Its purpose is to understand learning and measurement in a controlled environment, not to validate automated persuasion or deployment in customer service. Optimizing seller profit can conflict with customer interests, and the benchmark's legal action menu excludes only off-list prices and repeated or irrelevant offers; it does not certify safe or fair selling.

REPRODUCIBILITY STATEMENT

Section 3 and Appendix A specify the environment, the demand shift, and the exact reference; Appendix B gives the planner and the SIRA data, objectives, hyperparameters, and compute; and Appendix C defines every metric and statistic. The supplementary material contains an anonymized archive with the engine, the test customers, the prompts, and the training and evaluation code, which we will release publicly together with the data behind every figure and table.

AI USE STATEMENT

Generative AI assistants were used for implementation of baselines, debugging, and manuscript grammar and coherence checks. The authors remain responsible for every scientific claim.

REFERENCES

Marwa Abdulhai, Isadora White, Charlie Snell, Charles Sun, Joey Hong, Yuexiang Zhai, Kelvin Xu, and Sergey Levine. LMRL gym: Benchmarks for multi-turn reinforcement learning with language models. *arXiv preprint arXiv:2311.18232*, 2023.

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting REINFORCE-style optimization for learning from human feedback in LLMs. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12248–12267. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.662. URL https://aclanthology.org/2024.acl-long.662/.

Shimaa Ahmed, Yiwei Cai, Mohsen Minaei, and Rahul Rachuri. Can LLM agents price competitively? A dynamic multi-attribute auction benchmark for agentic commerce. *arXiv preprint arXiv:2608.00102*, 2026.

Thomas Anthony, Zheng Tian, and David Barber. Thinking fast and slow with deep learning and tree search. *Advances in Neural Information Processing Systems*, 2017. URL https://arxiv.org/abs/1705.08439.

Axel Backlund and Lukas Petersson. Vending-Bench: A benchmark for long-term coherence of autonomous agents. *arXiv preprint arXiv:2502.15840*, 2025. Andon Labs.

Victor Barres, Honghua Dong, Soham Ray, Xujie Si, and Karthik Narasimhan. τ^2-Bench: Evaluating conversational agents in a dual-control environment. *arXiv preprint arXiv:2506.07982*, 2025. URL https://arxiv.org/abs/2506.07982.

Federico Bianchi, Patrick John Chia, Mert Yuksekgonul, Jacopo Tagliabue, Dan Jurafsky, and James Zou. How well can LLMs negotiate? NegotiationArena platform and analysis. In *International Conference on Machine Learning*, 2024.

Sanjiban Choudhury. Process reward models for LLM agents: Practical framework and directions. *arXiv preprint arXiv:2502.10325*, 2025. URL https://arxiv.org/abs/2502.10325.

Konstantina Christakopoulou, Filip Radlinski, and Katja Hofmann. Towards conversational recommender systems. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 815–824, 2016. doi: 10.1145/2939672.2939746. URL https://www.kdd.org/kdd2016/papers/files/rfp0063-christakopoulouA.pdf.

Yang Deng, Yaliang Li, Fei Sun, Bolin Ding, and Wai Lam. Unified conversational recommendation policy learning via graph-based reinforcement learning. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, 2021.

Wei Fan, Xinjie Shen, Xudong Guo, Jianhong Tu, Yang Su, Yinger Zhang, Lianghao Deng, Fengyu Wang, Baohua Dong, Yangqiu Song, and Dayiheng Liu. E-Commerce Bench: Evaluating LLM agents on long-horizon autonomous business operation. *arXiv preprint arXiv:2608.30730*, 2026.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for LLM agent training. In *Advances in Neural Information Processing Systems*, 2025.

Saibo Geng, Martin Josifoski, Maxime Peyrard, and Robert West. Grammar-constrained decoding for structured NLP tasks without finetuning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 2023.

Caglar Gulcehre, Tom Le Paine, Srivatsan Srinivasan, Ksenia Konyushkova, Lotte Weerts, Abhishek Sharma, Aditya Siddhant, Alex Ahern, Miaosen Wang, Chenjie Gu, Wolfgang Macherey, Arnaud Doucet, Orhan Firat, and Nando de Freitas. Reinforced self-training (ReST) for language modeling. *arXiv preprint arXiv:2308.08998*, 2023. URL https://arxiv.org/abs/2308.08998.

He He, Derek Chen, Anusha Balakrishnan, and Percy Liang. Decoupling strategy and generation in negotiation dialogues. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2333–2343, 2018.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. *International Conference on Learning Representations (ICLR)*, 2022.

Wouter Kool, Herke van Hoof, and Max Welling. Buy 4 REINFORCE samples, get a baseline for free! In *ICLR Workshop on Deep Reinforcement Learning Meets Structured Prediction*, 2019. URL https://openreview.net/pdf?id=r1lgTGL5DE.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, 2023. URL https://arxiv.org/abs/2309.06180.

Wenqiang Lei, Xiangnan He, Yisong Miao, Qingyun Wu, Richang Hong, Min-Yen Kan, and Tat-Seng Chua. Estimation-action-reflection: Towards deep interaction between conversational and recommender systems. In *Proceedings of the 13th International Conference on Web Search and Data Mining (WSDM)*, pp. 304–312, 2020a.

Wenqiang Lei, Gangyi Zhang, Xiangnan He, Yisong Miao, Xiang Wang, Liang Chen, and Tat-Seng Chua. Interactive path reasoning on graph for conversational recommendation. In *Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, 2020b.

Yingjie Lei. PrefBench: Evaluating zero-shot LLM agents in hidden-preference personalized pricing negotiations. *arXiv preprint arXiv:2605.22855*, 2026.

Mike Lewis, Denis Yarats, Yann N. Dauphin, Devi Parikh, and Dhruv Batra. Deal or no deal? end-to-end learning of negotiation dialogues. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 2443–2453, 2017.

Raymond Li, Samira Ebrahimi Kahou, Hannes Schulz, Vincent Michalski, Laurent Charlin, and Chris Pal. Towards deep conversational recommendations. In *Advances in Neural Information Processing Systems*, volume 31, pp. 9725–9735, 2018. URL https://papers.neurips.cc/paper_files/paper/2018/hash/800de15c79c8d840f4e78d3af937d4d4-Abstract.html.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/aca97732e30bcf1303bc22ac3924fd16-Abstract-Conference.html.

Shuze Daniel Liu, Claire Chen, Jiabao Sean Xiao, Xin Chen, and David Simchi-Levi. Strategic bargaining in multi-buyer markets: Reinforcement learning from verifiable rewards for LLM negotiations. *arXiv preprint arXiv:2607.05863*, 2026.

Rajiv Mehta and Trishul Chilimbi. Amazon announces Rufus, a new generative AI-powered conversational shopping experience. Amazon News, https://www.aboutamazon.com/news/retail/amazon-rufus, 2024. February 2024.

Qwen Team. Qwen3.5-9B model card. Hugging Face, 2026. URL https://huggingface.co/Qwen/Qwen3.5-9B. Accessed September 22, 2026.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 2023.

Stephane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pp. 627–635, 2011. URL https://proceedings.mlr.press/v15/ross11a.html.

Andrei A. Rusu, Sergio Gomez Colmenarejo, Caglar Gulcehre, Guillaume Desjardins, James Kirkpatrick, Razvan Pascanu, Volodymyr Mnih, Koray Kavukcuoglu, and Raia Hadsell. Policy distillation. In *International Conference on Learning Representations*, 2016.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347.

Preethi Seshadri, Samuel Cahyawijaya, Ayomide Odumakinde, Sameer Singh, and Seraphina Goldfarb-Tarrant. Lost in simulation: LLM-simulated users are unreliable proxies for human users in agentic evaluations. *arXiv preprint arXiv:2601.17087*, 2026.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Avi Singh, John D. Co-Reyes, Rishabh Agarwal, Ankesh Anand, Piyush Patil, Xavier Garcia, Peter J. Liu, James Harrison, Jaehoon Lee, Kelvin Xu, Aaron Parisi, Abhishek Kumar, Alex Alemi, Alex Rizkowsky, Azade Nova, Ben Adlam, Bernd Bohnet, Gamaleldin Elsayed, Hanie Sedghi, Igor Mordatch, Isabelle Simpson, Izzeddin Gur, Jasper Snoek, Jeffrey Pennington, Jiri Hron, Kathleen Kenealy, Kevin Swersky, Kshiteej Mahajan, Laura Culp, Lechao Xiao, Maxwell L. Bileschi, Noah Constant, Roman Novak, Rosanne Liu, Tris Warkentin, Yundi Qian, Yamini Bansal, Ethan Dyer, Behnam Neyshabur, Jascha Sohl-Dickstein, and Noah Fiedel. Beyond human data: Scaling self-training for problem-solving with language models. *Transactions on Machine Learning Research*, 2024. URL https://arxiv.org/abs/2312.06585.

Xuanbo Su, Wenhao Hu, Le Zhan, Yuting Xie, Kailin Lyu, Kaijie Chen, Ziwei Li, Yeqiang Wang, Haibo Su, Yunzhang Chen, and Ling Huang. Sell more, play less: Benchmarking LLM realistic selling skill. *arXiv preprint arXiv:2604.07054*, 2026.

Kalyan T. Talluri and Garrett J. van Ryzin. *The Theory and Practice of Revenue Management*. Springer, 2004.

Jonathan Uesato, Nate Kushman, Ramana Kumar, Francis Song, Noah Siegel, Lisa Wang, Antonia Creswell, Geoffrey Irving, and Irina Higgins. Solving math word problems with process- and outcome-based feedback, 2022. URL https://arxiv.org/abs/2211.14275.

Xuewei Wang, Weiyan Shi, Richard Kim, Yoojung Oh, Sijia Yang, Jingwen Zhang, and Zhou Yu. Persuasion for good: Towards a personalized persuasive dialogue system for social good. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 5635–5649, 2019.

Zihan Wang, Kangrui Wang, Qineng Wang, et al. RAGEN: Understanding self-evolution in LLM agents via multi-turn reinforcement learning. *arXiv preprint arXiv:2504.20073*, 2025.

Brandon T. Willard and Rémi Louf. Efficient guided generation for large language models. *arXiv preprint arXiv:2307.09702*, 2023.

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256, 1992. doi: 10.1007/BF00992696. URL https://link.springer.com/article/10.1007/BF00992696.

Haojin Yang, Ai Jian, Xinyue Huang, Yiwei Wang, Weipeng Zhang, Ke Zeng, Xunliang Cai, and Jingqing Ruan. Harmonizing dense and sparse signals in multi-turn RL: Dual-horizon credit assignment for industrial sales agents. *arXiv preprint arXiv:2603.01481*, 2026.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. WebShop: Towards scalable real-world web interaction with grounded language agents. *Advances in Neural Information Processing Systems*, 2022.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. τ-bench: A benchmark for tool-agent-user interaction in real-world domains. In *International Conference on Learning Representations*, 2025.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. STaR: Bootstrapping reasoning with reasoning. In *Advances in Neural Information Processing Systems*, volume 35, 2022. URL https://proceedings.nips.cc/paper_files/paper/2022/hash/639a9a172c044fbb64175b5fad42e9a5-Abstract-Conference.html.

Yifei Zhou, Andrea Zanette, Jiayi Pan, Sergey Levine, and Aviral Kumar. ArCHer: Training language model agents via hierarchical multi-turn RL. In *Proceedings of the 41st International Conference on Machine Learning*, 2024.

Yifei Zhou, Song Jiang, Yuandong Tian, Jason Weston, Sergey Levine, Sainbayar Sukhbaatar, and Xian Li. SWEET-RL: Training multi-turn LLM agents on collaborative reasoning tasks. *arXiv preprint arXiv:2503.15478*, 2025.

APPENDIX OVERVIEW

Appendix A specifies AEROSELL and its exact reference, Appendix B the planner and SIRA's training, Appendix C the evaluation protocol, Appendix D additional results, Appendix E the 17 training recipes of the preliminary studies, and Appendix F the scope of the study. Unless stated otherwise, $\pm$ is the SD over training runs.

A THE AEROSELL ENVIRONMENT

Customers. Each customer comes from one of 12 type templates and has six private facts. Two identity facts, trip purpose and party, are fixed within a type; four trip facts (bags, schedule risk, ground plan, comfort priority) are resampled with probability 0.45. Willingness to pay is set per add-on as a ratio to its list price: each type's template wants have ratios from 0.6 to 1.9 (four of the 72 entries lie below 1), jittered by ± 0.30 and each dropped with probability 0.35; with probability 0.55 a customer also has one to three extra wants with ratios in $[1.05, 1.35]$; and all other add-ons receive ratios in $[0.05, 0.45]$. The turn budget is the type's nominal budget (6–14 paid turns) plus an integer jitter in $\{-2, \ldots, 2\}$, at least 5 (Figure 7b).

Observations and actions. The agent sees the catalogue (names, list prices, and the facts each add-on requires, without costs), the customer's account, the facts disclosed so far, its offers and their outcomes, the remaining turns once a tool result reports them, and up to eight earlier customers of the same stream (account fields, disclosed facts, offers, outcomes, ending, and profit). Willingness to pay, type, costs, and the demand phase stay hidden. After two free calls that return the account and the catalogue, every turn is one message with exactly one tool call from the legal menu: ASK an undisclosed fact, OFFER an eligible add-on not yet accepted or refused at its list price, or BOOK. A malformed, truncated, or illegal response ends the conversation as a timeout, and conversations stop after 20 decisions.

Profit. The profit share ϕ_p of each add-on ranges from 16% to 90% (Figure 7a). The hotel and car-rental bundles are priced at \$240 and \$180 but keep only 16–18%, so ranking offers by revenue favors low-profit add-ons. The broader engine used by the 17 recipes of the preliminary studies also allows off-list prices, and there repeated or irrelevant offers can end a conversation in a drop-off with no profit; the legal menu rules out both.

Demand shifts. The shift hashes a customer's (trip purpose, party) with SHA-256. For one parity it multiplies willingness to pay for seat, cabin, and airport add-ons by a high factor and for all other add-ons by a low factor; the other parity reverses the roles. The factors are 3.0 and 0.25 in training and 2.2 and 0.4 at test, so the test shift follows the training rule at a smaller magnitude. The shift lowers the attainable add-on profit by about 38% (Figure 7c); the same types with and without the shift differ by 37%, so the drop reflects demand, not which customers arrive.

An exact reference. Let $\mathcal{F}$ be the six facts, T the turn budget, and $U(F)$ the add-ons whose required facts lie in F and whose willingness to pay is at least the list price. The full-information reference earns

$$M^\star = m_0 + \max_{F \subseteq \mathcal{F},\, |F| \le T-1} \sum_{p \in \mathrm{Top}_{T-1-|F|}\, U(F)} (L_p - c_p), \tag{5}$$

where Top selects add-ons by decreasing profit.

Proposition 1 (Exactness)**.** *For a fixed customer, with list prices, one copy per add-on, additive profits, one new fact per* ASK*, and one turn for* BOOK*, Equation* (5) *is the largest profit any agent can earn.*

Proof. An optimal plan spends no turn on refused, repeated, or ineligible offers or on questions that reveal nothing. Its $|F|$ questions can precede its offers, since order changes neither willingness to pay nor eligibility, and BOOK takes one turn, leaving at most $T-1-|F|$ offers. Exactly the add-ons in $U(F)$ can be accepted, and additivity makes the most profitable subset optimal for this F. Asking F, offering that subset, and completing the purchase is feasible, so maximizing over F attains the bound. □

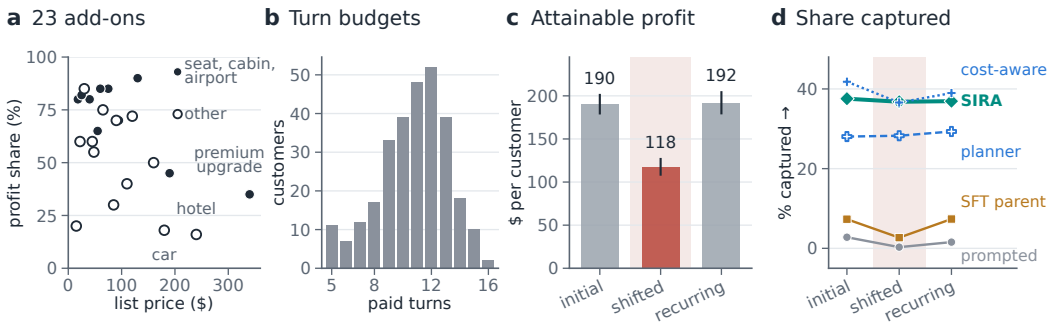

Figure 7: **The shift shrinks the attainable profit, and SIRA's share of it stays nearly constant.** **(a)** The 23 add-ons: list price against profit share; filled markers are the seat, cabin, and airport add-ons, whose value moves opposite to the rest under the shift. **(b)** Turn budgets of the 288 test customers. **(c)** Attainable add-on profit per customer ($M^\star - m_0$) in each demand phase, mean over the 12 test streams with 95% CIs. **(d)** Share of the attainable add-on profit each agent captures; SIRA pools its three runs.

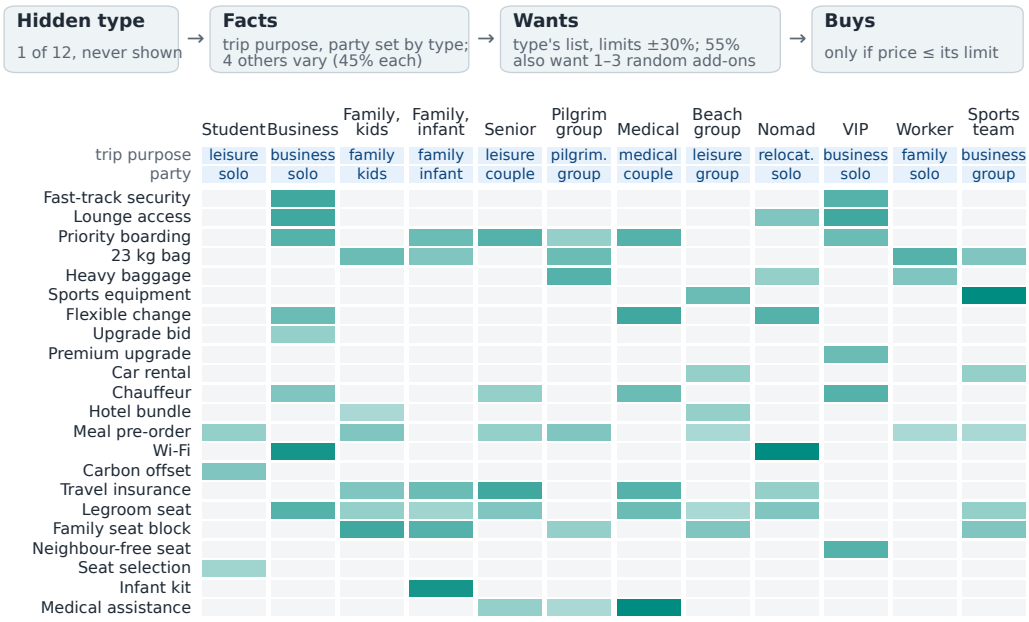

Figure 8: **AEROSELL customers are 12 hidden types.** Top: how a customer is drawn. Bottom: for each type (columns), the two identity facts (blue) and the add-ons it pays more than list price for (teal; darker cells, up to $1.9\times$ the list price). Individual ratios vary by up to ± 0.3, and 55% of customers also want one to three add-ons outside their type. Trip purpose and party identify 10 of the 12 types; business travelers who fly alone are either commuters or VIPs, whom the loyalty tier helps tell apart. The agent never sees the type or this table.

Enumerating the 2^6 fact sets is exact and fast. Because no agent observes willingness to pay, no agent can be expected to reach $M^\star$, so the share of the attainable add-on profit an agent captures is a conservative measure. SIRA's share is nearly the same in all three phases (36.8–37.6%; Figure 7d): its profit falls under the shift in proportion to the attainable profit.

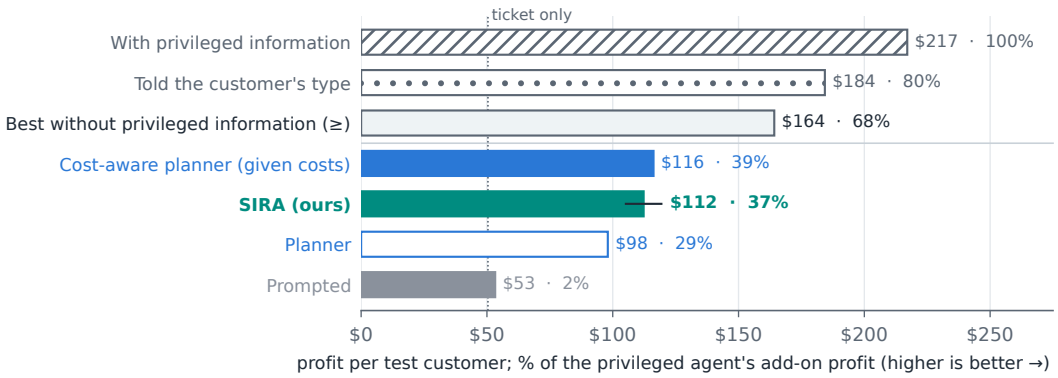

Figure 9: **Much of the profit is attainable without privileged information.** Profit per test customer and share of the add-on profit of the agent with privileged information, which sees each customer's price limits. The three bounds follow the same rules as the agents (one fact per question, offers only after their facts are known, list prices, one turn kept to complete the purchase). The best agent without privileged information knows how customers are generated, the current demand phase, and each customer's patience, but sees only the disclosed facts; it is a lower bound on the best such agent, because it ignores refusals, the account, and earlier customers. The next bound is also told the customer's type. Here, privileged information means the customer's price limits; the bounds also know the demand phase, which the agents do not observe. SIRA: mean ± SD over three runs.

Table 2: **One complete conversation, exactly as stored.** The planner serves a training customer of the VIP type (business trip, alone, two checked bags, tight connection, group coach, no comfort priority; patience 8 turns; limits $519.19 for the premium upgrade, $191.13 for car rental, $88.52 for the hotel, $20.75 for the legroom seat). The agent sees the gold loyalty tier, 16 earlier bookings, the catalogue, the allowed moves, and the eight previous customers. The message is the planner's template; only the tool call changes the outcome, and the customer replies with the engine's result. Profit $201.80: the $50.40 ticket, $119.00 from the upgrade (35% of $340), and $32.40 from the car rental (18% of $180).

	Agent message	`Tool call`	Customer reply	Turns left
1	Could you tell me about your trip purpose?	`ask_customer(trip_purpose)`	trip purpose: business	7
2	Could you tell me about your ground plan?	`ask_customer(ground_plan)`	ground plan: group coach	6
3	Could you tell me about your comfort priority?	`ask_customer(comfort_priority)`	comfort priority: none	5
4	Would you like Premium-cabin upgrade for $340.00?	`offer_product(upgrade_premium, 340)`	accepted	4
5	Would you like Hotel bundle for $240.00?	`offer_product(hotel_bundle, 240)`	refused (above willingness to pay)	3
6	Would you like Extra-legroom seat for $75.00?	`offer_product(seat_extra_legroom, 75)`	refused (above willingness to pay)	2
7	Would you like Car rental bundle for $180.00?	`offer_product(car_rental, 180)`	accepted	1
8	I'll finalize your booking with the ticket and any services you've accepted.	`book()`	booking committed	0

B THE PLANNER AND SIRA TRAINING

Planner. The planner estimates acceptance with Equation (3), from past offers of the add-on whose disclosed fact values match every known fact of the current customer, or from all past offers of the add-on if none match. It takes the remaining turns from the last tool result; before the first report it uses the median budget of earlier customers, else ten turns. Given disclosed facts K and missing facts $\bar{K}$, it enumerates fact sets $S \subseteq \bar{K}$, reserves one turn for BOOK, and maximizes the summed scores of the best add-ons that become eligible given $K \cup S$ and are not yet accepted or refused, as many as the remaining turns allow. It asks the first missing fact of the best plan or otherwise makes its best offer, replans after every tool result, and completes the purchase when one turn remains or no offer has a positive score. The cost-aware planner scores offers by $(L_p - c_p)\widehat{q}_p$.

Algorithm 1 SIRA training and test-time use (symbols as in Section 4).

Require: planner, planner and random states, base model π_0
1: Label every state with the planner's action, written as a templated response and tool call; select the SFT set
2: Fit π_{SFT} by SFT (Section 4.2); set $\pi_\theta \leftarrow \pi_{\mathrm{SFT}}$ and freeze $\pi_{\mathrm{ref}} \leftarrow \pi_{\mathrm{SFT}}$
3: **for** each of 32 groups **do**
4: Fix one customer state and a planner-generated history; sync the generator to π_θ
5: Generate G=8 complete conversations; compute each profit, r_i (Equation (4)) and A_i
6: Compute ρ_i from learner and generator probabilities; stop if a guard fails
7: **if** some $A_i \neq 0$ **then** update θ with Equation (6)
8: **Test time:** fix π_θ; each turn, generate one response from the public state and the agent's own recent customers

Stage 1 data. Twelve planner streams (128 customers per phase, longer than the 24-customer test streams) and twelve random streams (32 per phase) give 5,760 conversations and 61,610 states. The random agent asks trip purpose and party first, then asks with probability 0.4 and otherwise makes a random legal offer, completing the purchase with one turn left. At every state the planner supplies the target action, written by fixed templates as a complete response. Selection keeps 8,192 states (5,818 planner and 2,374 random) that cover every customer and every stream, phase, and action type.

Stage 2 objective. Each of the 32 groups fixes one customer state from four RL streams (11 groups in the initial phase, 11 in the shifted phase, and 10 in the recurring phase) with a history generated by the planner. vLLM (Kwon et al., 2023) generates the group's $G = 8$ conversations from the current agent with recorded token probabilities μ. With π_{pre} the agent before the update, token y_{it} of conversation i and its context h_{it}, the importance weight is $\rho_i = \prod_t \pi_{\mathrm{pre}}(y_{it} \mid h_{it})/\mu(y_{it} \mid h_{it})$, and the loss is

$$\mathcal{L}_{\mathrm{RL}} = \frac{1}{G}\sum_{i=1}^{G} \mathrm{sg}(\rho_i)\Big[- A_i \sum_t \log \pi_\theta(y_{it} \mid h_{it}) + \frac{\beta}{|y_i|}\sum_t D_{\mathrm{KL}}\big(\pi_{\mathrm{ref}} \,\|\, \pi_\theta\big)(h_{it})\Big], \tag{6}$$

with the advantage A_i of Section 4.3, $\beta = 0.01$, stop-gradient sg, $|y_i|$ generated tokens, and the SFT parent as π_{ref} (full-vocabulary KL). Weights are neither clipped nor self-normalized and advantages are not divided by a group SD, which avoids the bias of clipping and normalization; instead, training stops if a group has $\max_i |\log \rho_i| > 2$ or an effective sample size below $G/2$. Groups whose returns are all equal skip the update, so the three runs make 24, 27, and 28 updates (Algorithm 1).

How the test set was used. The 288 test customers were generated after the planner, the SFT data, and the prompt had been fixed, with prompts and SFT runs compared on the 72 validation customers; the SFT parent was chosen there before any test result existed. The RL objective and its settings (reward, validity weight, KL coefficient, G, 32 groups, final checkpoint) were fixed before the test set was generated and were never tuned. No RL model was evaluated on validation customers, however. The two RL runs from the SFT parent, trained with the standard prompt and with cost advice in the prompt, were first evaluated on the test set, and we replicated the standard-prompt run with two further seeds after seeing both results. The test set therefore informed which RL variant we report. The two replications were fixed before their evaluation and earn \$119.38 and \$104.46, both above the planner's \$98.04, and every RL run evaluated on the test set is reported. A pre-registered evaluation on 360 untouched customers confirms every main comparison (Appendix D).

DPO control. DPO starts from the base model and uses 2,048 preference pairs that rank alternative actions at the same training state by their completed profit under a fixed planner continuation, computed with hidden customer data. It learns to offer earlier than the prompted model (0.89 compared with 1.47 questions before its first offer), and most of its conversations end on an offer whose required fact is still unknown, consistent with pairs that reward early offers without teaching which offers are valid. A fourth RL run from the SFT parent, trained with cost advice in its prompt, is used only in the prompt comparison; the three standard runs never see costs in their prompts.

Table 3: **Data, training settings, and compute.** Stream sets are disjoint; the three SFT runs reuse the same selected data, and all three RL runs refine the same SFT parent, which is also the KL reference. Validation customers served to compare prompts and SFT runs, including the choice of the SFT parent, before any test result (Appendix B). Preference pairs and anchors belong to the DPO control.

Data		
Component	Streams	Size
Planner conversations	12	4,608
Random conversations	12	1,152
Visited states	24	61,610
Selected SFT states	24	8,192
planner / random states		5,818 / 2,374
DPO preference pairs / anchors	24	2,048 / 384
RL start states (groups $\times$ G)	4	32×8
Validation customers	3	72
Test customers	12	288

Model, optimization, decoding, and compute	
Base model	`Qwen3.5-9B`, BF16 (adapters FP32)
LoRA rank / scale / dropout	16 / 32 / 0 on q, k, v, o, gate, up, down
Trainable parameters	29,097,984
SFT	1 epoch, batch 8, 1,024 updates
Optimizer (SFT and RL)	AdamW, lr 10^{-5}, $\beta\,(0.9, 0.999)$, ϵ 10^{-8}, no decay
RL reward	$(M - m_0)/100 + 0.02\,v$
KL coefficient / grad-norm cap	0.01 / 1
Importance guards	$\max \lvert \log \rho \rvert \leq 2$, ESS ≥ 4 of 8
History / decision limit	8 customers / 20 decisions
Context / completion	8,192 / 256 tokens
Sampling	temperature 1, top-p 1, no top-k, min-p 0
Compute	RL 4.3–4.7 h per run, one RTX PRO 6000 GPU; SFT 11–19 h, DPO about 9.5 h per run, one GPU

C Evaluation protocol

Every agent meets the same 288 test customers in the same order with fixed weights; histories reset at the start of each of the 12 streams and then evolve under each agent. No response is repaired and no checkpoint is selected per phase or prompt. For customer i with profit M_i, purchase indicator b_i, a_i accepted and o_i executed offers, and e_i of g_i responses valid,

$$\text{profit} = \tfrac{1}{N}\textstyle\sum_i M_i, \qquad \text{sale rate} = \tfrac{100}{N}\textstyle\sum_i b_i \mathbf{1}[a_i > 0], \qquad \text{completion rate} = \tfrac{100}{N}\textstyle\sum_i b_i, \tag{7}$$

$$\text{offer acceptance rate} = 100\,\frac{\sum_i a_i}{\sum_i o_i}, \qquad \text{valid-response rate} = 100\,\frac{\sum_i e_i}{\sum_i g_i}. \tag{8}$$

Trained methods report the mean and sample SD of per-run values. Paired comparisons average customer-level differences within each stream and form Student-t intervals over the 12 streams (11 df), which respects the dependence created by shared histories; SIRA pools its three runs per customer, so these intervals reflect customer and stream variation, while the SDs over runs in the tables report variation between training runs. Differences use unrounded means.

D Additional results

Paired gains and consistency. SIRA beats the prompted model, its SFT parent, and the planner in all 12 streams, and every RL run beats the parent in every stream and in 107 of the 108 run–stream–phase cells (Figure 10a,b and Figure 5a). The cost-aware planner is not significantly ahead, and in the shifted phase SIRA earns \$0.22 more.

Confirmation on untouched customers. Because the test set informed which RL variant we report (Appendix B), we evaluated every agent on 288 new customers (12 streams) and on the 72

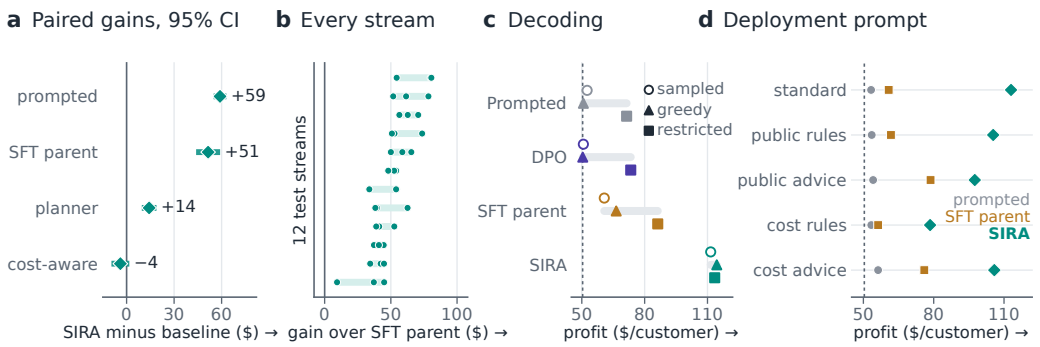

Figure 10: **The main result is robust. (a)** SIRA minus each baseline, paired over the 12 test streams (95% Student-t intervals). **(b)** SIRA's gain over its SFT parent in every stream (rows, sorted); dots are the three RL runs. **(c)** Profit under sampling (main protocol, re-run in the decoding harness), greedy decoding, and sampling restricted to valid actions. **(d)** Profit under every prompt; SIRA is run 0, the one run evaluated under all five prompts. Dotted lines mark the ticket profit.

customers originally reserved for testing (3 streams; distinct from the 72 validation customers), all generated with the test settings and never evaluated before, following an analysis registered before any result. Over all 360 customers the prompted model earns \$52.33, the SFT parent \$66.23, SIRA \$120.95 $\pm$ 8.09, the planner \$107.14, and the cost-aware planner \$126.06. SIRA beats the SFT parent by \$54.72 (95% CI [47.97, 61.47], all 15 streams), the planner by \$13.81 ([7.68, 19.95], 13 of 15 streams), and the prompted model by \$68.62 ([60.39, 76.86]), and it is again not significantly behind the cost-aware planner ($-$\$5.11, [$-$11.58, 1.36]). It completes the purchase for 96.0% of customers and captures 39.6% of the attainable add-on profit. The registered primary test, on the 288 new customers alone, reaches the same conclusions: SIRA leads the SFT parent by \$53.19 ([44.94, 61.43]), the planner by \$11.73 ([4.87, 18.60]), and the prompted model by \$65.70 ([56.26, 75.15]), and trails the cost-aware planner by a non-significant \$7.44 ([$-$14.92, 0.05]; not shown equivalent within $\pm$\$10). On the 72 reserved customers SIRA earns \$132.97 $\pm$ 8.45 and leads both the SFT parent and the planner in all three streams. The registration documents, with timestamps and hashes, are in the supplementary material.

Restricted decoding. Each decision compiles a regular expression that accepts exactly one JSON object whose tool call is an entry of the current legal menu, enforced token by token by vLLM's structured-output backend (xgrammar); the 256-token limit still applies. The harness reproduces the saved prompts token for token, and its sampled means match the main results within \$0.89 per customer. Restriction removes every invalid call except two truncated DPO replies and lifts the baselines, yet SIRA keeps 54% of its sampled advantage over the parent and 71% over the prompted model (Figure 10c). By Equation (2), the remaining \$27.19 gap to the parent splits into completing more purchases, \$10.3 when valued at the parent's add-on profit per purchase, and better offers, \$16.9. Under restriction the SFT parent completes 77.4% of purchases and keeps \$46.30 of add-on profit per purchase, while SIRA completes 99.7% and keeps \$63.26. Greedy decoding leaves SIRA nearly unchanged and does not rescue the prompted model, whose deterministic replies are mostly premature offers.

A pre-registered rerun of this confirmation with restricted decoding reproduces these findings on the 360 untouched customers. SIRA earns \$124.49 $\pm$ 5.09, against \$91.40 for its SFT parent, \$67.73 for the prompted model, \$72.03 $\pm$ 2.00 for DPO, \$107.14 for the planner, and \$126.06 for the cost-aware planner. It beats the parent by \$33.08 (95% CI [26.37, 39.80]), the planner by \$17.35 ([12.08, 22.61]), the prompted model by \$56.76 ([48.36, 65.16]), and DPO by \$52.46 ([44.57, 60.36]), each in all 15 streams, and it is not significantly different from the cost-aware planner ($-$\$1.57, [$-$7.93, 4.78]). On the 288 primary customers every registered comparison passes; there, the difference from the cost-aware planner is $-$\$3.65 (95% CI [$-$11.02, 3.71]), and its 90% CI lies inside the pre-registered $\pm$\$10 margin, so the two are equivalent by a two one-sided test (under standard decoding, equivalence was not shown). The only invalid endings are 9 of DPO's 1,080 conversations, cut off at the 256-token limit.

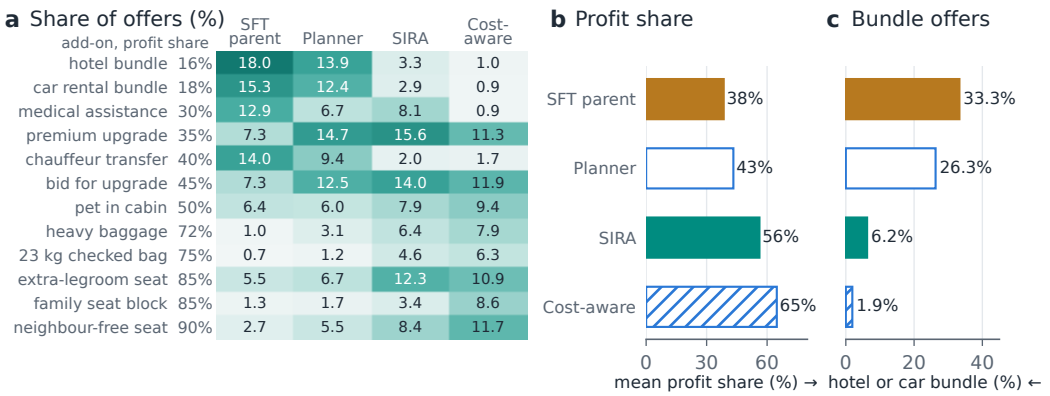

Figure 11: **SIRA shifts its offers toward add-ons the airline earns more on.** **(a)** Share of each agent's offers going to each add-on (the twelve most offered, sorted by profit share). **(b)** Mean profit share of offered add-ons. **(c)** Share of offers going to the hotel and car-rental bundles. Test set; SIRA pools its three runs.

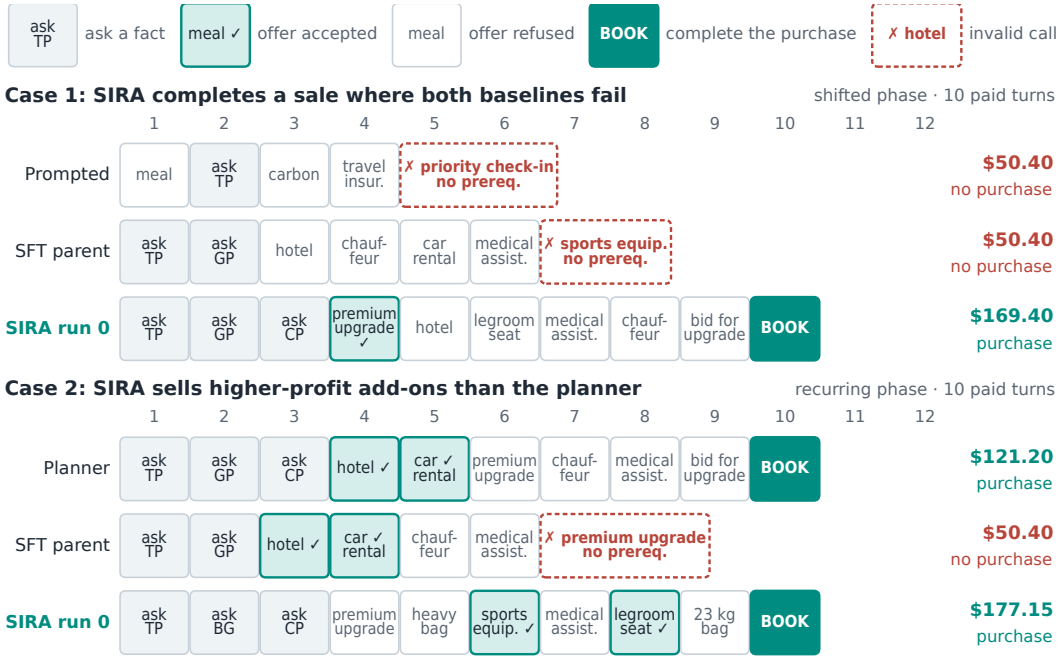

Figure 12: **Two customers, turn by turn**, read from the saved evaluation records. Each chip is one decision: a question (TP trip purpose, PA party, BG bags, SR schedule risk, GP ground plan, CP comfort priority), an accepted or refused offer, BOOK, or the final invalid response with its reason. Case 1 is the first shifted-phase customer on which all three SIRA runs sell and complete the purchase while neither baseline does; case 2 is the first customer on which the planner sells a hotel or car-rental bundle and every SIRA run earns more. Selection is deterministic.

Prompts. Public rules describe the planner's procedure in words, public advice adds its recommended action and product estimates at every turn, and the cost variants also list product costs. Advice helps the imitation agent, which benefits from being told a valid action, but lowers SIRA, which performs best with the prompt it was trained with (Figure 10d); the extra instructions mainly break its response format. A run trained with cost advice performs best when used with that advice ($113.98, compared with $88.37 under the standard prompt), so the recipe adapts to whichever prompt it is trained with.

Case studies. Figure 12 follows two customers turn by turn. In case 1, neither baseline completes a purchase; SIRA asks three facts, including the two that unlock the premium upgrade, sells it on turn 4, and completes the purchase on its last turn, reaching 77% of this customer's attainable add-on profit. Its other two runs ask a different second question and make different refused offers, but sell the same upgrade on turn 4 and earn the same profit. In case 2, the planner sells the hotel and car-rental bundles, \$420 of list price that earns \$70.80, and the SFT parent copies that choice and then loses the cart on an invalid offer. SIRA asks about bags instead of the ground plan and sells sports equipment and an extra-legroom seat, \$165 of list price that earns \$126.75, the offer shift of Figure 11 in a single conversation.

Model scale. The scaling study was registered before the test set existed and used three streams of 96 customers from the same generator, decoded with Hugging Face generation rather than vLLM. Its customers and decoder differ from the test set's, so its prompted 9B model earns \$51.72 rather than \$53.28.

E THE 17 TRAINING RECIPES OF THE PRELIMINARY STUDIES

Free-form study. Ten recipes were trained on logged free-form dialogues (987 conversations from 104 development customers) and tested on 60 held-out customers with 1,500 conversations per recipe: four ways of imitating the logs (all, highest-profit, a format control, and a belief-state variant), turn-level and action-value DPO, and offline adaptations of GRPO and DuCA to a fixed pool together with two DuCA variants of ours, ScoPE (credit for eliciting facts) and BAPO (offer credit weighted by an oracle's belief about willingness to pay). Imitation loses \$33.5–35.8 per customer, and only turn-level DPO is positive (+\$1.70, SE 1.97; +\$0.76, SE 1.23, with demand shifts), within noise (Figure 13a).

Symbolic-action study. Seven recipes chose symbolic actions on a separate 288-customer set with demand shifts: three SFT targets (action values, response prediction, history conditioning) and customer- or block-profit RL from the base model or an SFT initializer, three runs each. None clearly beats the prompted model, whereas both planners do (Figure 13b). Moved to full dialogue on the test set, all seven stay at or below the prompted model and end most conversations on an invalid response (Figure 13c).

Why they failed. We attribute the failures to three causes: (i) imitating logged dialogues reproduces their errors; (ii) the process rewards credit an accepted offer's price even when the cart is later lost, so they disagree with profit; and (iii) agents trained on symbolic actions do not transfer to generated tool calls, which panel (c) measures. SIRA avoids all three: it imitates an executable planner, its reward is profit, and it trains with the tool calls it is tested on.

F SCOPE AND DESIGN CHOICES

Positioning. Table 4 compares AEROSELL with the closest benchmarks (Section 2).

Table 4: **Positioning.** AEROSELL scores the profit of each sales conversation with tool calls, demand shifts, and an exact bound. Benchmarks: WebShop (Yao et al., 2022), τ-/τ^2-bench (Yao et al., 2025; Barres et al., 2025), SalesLLM (Su et al., 2026), PrefBench (Lei, 2026), Vending-Bench (Backlund & Petersson, 2025), and E-Commerce Bench (Fan et al., 2026).

Benchmark	Sells	Tool actions	Profit per conversation	Demand shift	Exact bound
WebShop	✗	✓	✗	✗	✗
τ-/τ^2-bench	✗	✓	✗	✗	✗
SalesLLM	✓	✗	✗	✗	✗
PrefBench	✓	✗	✓	✗	✗
Vending-Bench	✓	✓	✗	✓	✗
E-Commerce Bench	✓	✓	✗	✓	✗
AEROSELL (ours)	✓	✓	✓	✓	✓

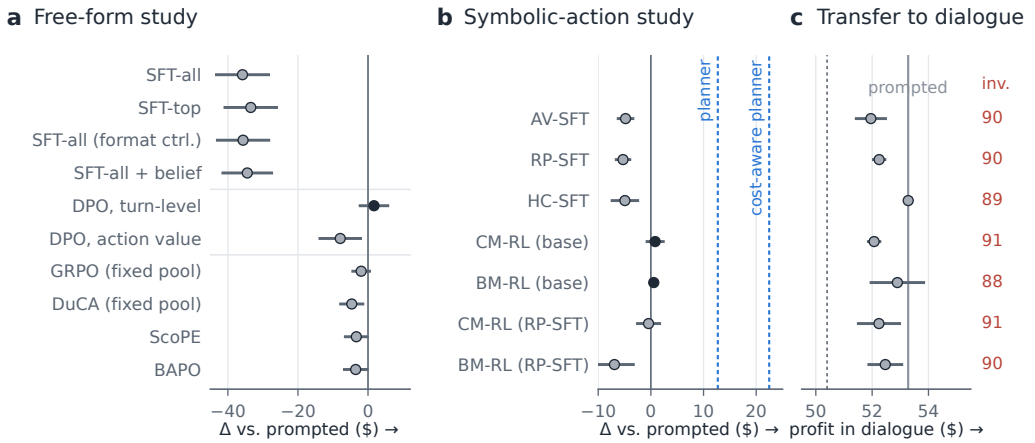

Figure 13: **The 17 recipes stay near or below prompting.** **(a)** Free-form study: difference from the prompted model with 95% intervals from the customer-level SE; black markers are the positive estimates. **(b)** Symbolic-action study: difference from the prompted model, mean $\pm$ SD over three runs; dashed lines mark the two planners. **(c)** The same seven recipes in full dialogue on the 288 test customers, with invalid endings per 100 customers.

Controlled customers. Willingness to pay is synthetic and additive, acceptance is a deterministic price threshold, and customer replies are rendered from engine outcomes. These choices make every sale reproducible, give an exact full-information reference, and allow matched comparisons on identical customers. AEROSELL therefore measures sales decisions and their execution; persuasion and customer satisfaction are outside its scope.

One move per turn. One question or offer per turn models a customer who answers one question and considers one product at a time, which makes the turn budget bind and the reference exact; the remaining turns are reported so that completing the purchase can be planned.

Cost information. Costs are never given to SIRA, but they are fixed, so the profits of earlier customers in the history reveal them in part: a completed purchase with one add-on reveals its cost exactly. A single history identifies about three of the 23 costs, while least squares on the public records of the 5,760 training customers recovers all 23. A planner that estimated costs this way would therefore make the cost-aware planner's decisions, which are not significantly better than SIRA's (Section 5.1). Our claim is therefore not that SIRA beats every hand-built heuristic, but that a language agent learns cost-aware selling from the profit of 256 of its own conversations, without costs in its prompt.

History and adaptation. SIRA's weights are fixed at test time. In a pre-registered ablation on the test set, removing its history changes its profit by $-\$2.66$ (95% CI $[-5.81, 0.50]$; $-\$4.84$, $[-8.58, -1.09]$, in the initial phase, the only significant phase) and replacing it with a stale history from the previous phase by $+\$0.86$ ($[-0.67, 2.38]$). SIRA's gains in every phase therefore come from its trained policy, robust selling under shift, rather than from tracking the shift through its history; learning online from the history is a capability that AEROSELL can evaluate directly.

Training study. All learned agents build on `Qwen3.5-9B`. The three SFT runs vary the supervised stage, and the three SIRA runs refine one SFT parent, the weakest of the three on sale and completion rates, so the gains do not come from choosing a strong initializer. Training RL without Stage 1 would isolate the contribution of imitation.