The Tradeoffs of Decision-Time Reasoning in LLM Trading Agents

Ing-Haw Cheng*† Goutham Gopalakrishna* Maurice Granger‡
Justin Shi* Vasily Strela§

First draft: May 4, 2026
This draft: November 15, 2026

Abstract

Large-language-model (LLM) trading agents reason at the moment of decision, which consumes time while markets evolve. We study the resulting tradeoffs in a real-time market simulator where agents select tenders and execute trades. Varying reasoning intensity and market speed, we find that reasoning improves some decision margins but imposes latency costs. The reasoning dividend appears in tender selection and completed-trade quality; the deliberation tax appears in expired accept attempts and stuck inventory. More reasoning does not consistently increase profits. Performance improves when calculations and unwind execution are moved out of the agent's reasoning loop. In comparison, a task-trained neural-network trader acts nearly instantaneously. However, the LLM agent's profits are more resilient as liquidity falls below the level used to train the NN trader, with both traders using the same execution policy. Decision-time reasoning can therefore add value beyond task-specific training, but its benefits depend on market conditions and the time available to act.

Keywords: Agentic trading, artificial intelligence, large language models

*Rotman School of Management, University of Toronto. 105 St. George Street, Toronto, ON, M5S 3E6, Canada.

†Corresponding author. The authors thank Landis Li, Sean Woo and Ray Fang for excellent research assistance, Eric Kang and Tom McCurdy for help and access to the Rotman Interactive Trader environment, Avi Goldfarb, Martin Szydlowski, and seminar participants at the University of Toronto for comments, and Rotman FinHub and the John Hull Financial Innovation Fund for generous funding. Ing-Haw Cheng thanks the Social Sciences and Humanities Research Council of Canada (SSHRC) for generous funding through SSHRC Insight Grant #435-2024-0365. Goutham Gopalakrishna thanks SSHRC for generous funding through SSHRC Insight Development Grant #430-2025-00727.

‡Royal Bank of Canada (RBC) Capital Markets. New York, NY, USA. The views expressed in this paper are the authors' and do not necessarily reflect those of the Royal Bank of Canada.

§HedgehogCamp.ai, New York, NY, USA. The views expressed in this paper are the authors' and do not necessarily reflect those of HedgehogCamp.ai.

Financial institutions are investing in artificial intelligence at substantial rates, increasingly deploying it as a decision-making technology. In capital markets trading, major dealers have built task-trained AI systems for execution: JPMorgan's LOXM platform, for example, uses deep reinforcement learning to execute equity trades.[1] More recently, attention has turned to trading agents built on large language models (LLMs), which can interpret market information and take actions on a trader's behalf. LTX's Agentic BondGPT, for example, allows traders to delegate market monitoring and trading actions.[2] These developments raise a basic question: How do LLM trading agents differ from other algorithmic trading systems?

One key difference lies in how LLM trading agents arrive at decisions. Traditional algorithms and task-trained systems encode trading judgments in pre-specified rules or policies learned from trading data. In contrast, LLMs are trained to generate language and reason across tasks. The connection between training and trading actions is thus indirect: LLM trading agents apply their broader capabilities to the trading problem through decision-time reasoning. This reasoning may improve decisions (Lopez-Lira, 2025), but it consumes time while the market evolves.

This paper studies the tradeoffs of decision-time reasoning in LLM trading agents.[3] We show that more reasoning does not consistently improve trading profits: its value depends on the time available to act and the tasks assigned to the agent. Performance improves when calculations and execution are moved out of the LLM's reasoning loop, leaving the agent responsible for trading judgment. When market conditions change, retaining that judgment produces more resilient profits than replacing it with a task-trained policy.

We test these tradeoffs in a simulated market environment in which the trader faces a two-part task: deciding which client orders to accept, and executing the resulting trades. In each session, the trader receives institutional requests to buy or sell blocks at quoted prices. The trader must decide whether to accept the tender and, if accepted, must unwind the resulting inventory in a real-time limit-order-book market. The trader must weigh the

[1] See Noonan, "JPMorgan develops robot to execute trades," Financial Times, July 31, 2017.

[2] See Tayeb and Mutua, "AI Agents Seen Taking Credit-Trading Work From Humans Over Time," Bloomberg, October 26, 2026.

[3] Both LLM agents and task-trained neural-network traders are algorithms, and both have fixed model parameters during our evaluation. Reinforcement-learning policies can be stochastic, change through online learning, or incorporate decision-time planning. Our comparison concerns the architectures studied here.

quoted price advantage against displayed liquidity, remaining time, inventory, and penalties. Moreover, tender selection and execution are time-sensitive: delay can cause a tender to expire before acceptance, and once inventory is created, delay exposes the trader to book movements and end-of-session close-out costs. The task therefore makes both judgment and timely action consequential for profits.

We connect frontier language models to the simulated market through an interface that converts market states into model observations and model responses into actions. The prompt specifies the trading role and constraints, but does not prescribe an execution rule. The model chooses whether to accept or decline tenders, submit or cancel orders, or wait. We refer to this as the agent-in-control architecture because the model controls both tender selection and execution. Our baseline analysis uses `gpt-5.4` and varies the additional deliberation requested from the model across off, low, and medium settings. Even with reasoning off, the model generates its response sequentially from the current market observation. The simulator is the Rotman Interactive Trader (RIT; see Mak and McCurdy, 2019), which allows us to vary the wall-clock duration of a simulation tick. We use three market speeds—slow, normal, and fast—so that the same response time consumes different amounts of simulated market time. The design therefore varies both the requested deliberation and its market-time cost.

Figure 1 shows average net profits across reasoning settings and market speeds. Two patterns emerge. First, more reasoning does not consistently improve performance: the low-reasoning agent earns the highest average profits in two of the three speed treatments. Second, the return to additional reasoning also depends on market speed. In the fast market, the medium-reasoning agent earns negative average net profits because of fines for speculative positions and inventory remaining at session end. In the slow market, the same agent earns the highest average profits. These patterns illustrate how the balance between better decisions and delayed action changes with market speed.

Market speed also affects the downside risk associated with reasoning. In the fast market, medium reasoning (the highest level we consider) produces a heavier left tail of net profits than low reasoning, whereas in the slow market, the reasoning-off agent has the heavier left tail. Holding the reasoning setting fixed, slower markets tend to reduce the severity of losses.

The cost of deliberation therefore appears in both average profits and downside risk.

Turning toward the mechanism, we first show that reasoning improves the agent's use of economic information at the decision point. Reasoning makes accept decisions more sensitive to current order-book conditions: more deliberative agents are more responsive to the spread between the tender price and what the agent could realize by walking the book, and they are less likely to accept tenders priced unfavorably to that spread. Conditional on completing the unwind, more deliberative agents also earn more profit from the trade. We refer to these improvements as a reasoning dividend.

The dividend, however, does not fully translate into net performance because reasoning consumes market time. We show that more deliberative agents attempt to accept more tenders that have expired during the agent's think time, and they leave more accepted positions stuck with residual inventory at session end. We refer to these costs as a deliberation tax. The tax is concentrated in faster markets, where each second of deliberation consumes more simulated market time. Thus, reasoning creates a dividend in selection and conditional execution, but incurs a tax in execution failures.

We also directly study the agent's recorded reasoning. More deliberative agents more frequently discuss whether an accepted position can be unwound, planning how aggressively to do so, and tracking remaining time and position limits. The content of the agent's recorded reasoning therefore aligns with what the dividend mechanism would predict: the agent considers task-relevant factors in its decision-making.

We next use the mechanism results to study architecture. The evidence above shows that reasoning changes the quality of the agent's intended decisions, but also affects whether those decisions are converted into completed trades. This raises a design question: which parts of the trading problem should remain inside the agent's real-time reasoning loop, and which parts should be compiled into the surrounding system?

We study this question by comparing the baseline agent-in-control architecture with three alternatives that progressively move parts of the task out of decision-time reasoning. The first moves calculations out of the agent's reasoning loop by providing pre-computed variables that support decision-making, including book-based tender edge and projected inventory. The second combines agentic tender selection with algorithmic execution: the

agent retains responsibility for the tender decision, but a compiled unwind algorithm handles the trading loop and reduces any positions toward zero. The final alternative is a fully compiled deterministic algorithm. This algorithm is a limiting point on the speed–reasoning frontier because both tender selection and unwind execution rules are specified before the session begins.

Our results show that the frontier depends not only on the model's reasoning capability, but also on what the agent is asked to compute and execute while the market is moving. Providing pre-computed decision support improves performance over the baseline. Delegating execution to a compiled routine yields a further improvement: with the LLM agent responsible for tender selection, this architecture is competitive with the fully compiled benchmark. These results show the value of reserving decision-time reasoning for tender judgment while handling calculations and execution outside the reasoning loop.

Motivated by this last result, our final main results compare an LLM agent with a neural-network (NN) trader trained specifically for the LT3 task. We first show that the trained networks act with negligible delay at the market speeds studied. Increasing network size does not systematically improve profits, nor does the effect of market speed differ detectably across architectures. Thus, varying network size does not reproduce the speed–reasoning tradeoff observed for the LLM agent.

However, the LLM agent's profits are more resilient when market conditions differ from the NN trader's training environment. Both traders use the same trained execution policy, so the comparison focuses on tender selection. We train the NN trader under normal liquidity and then progressively reduce market liquidity while holding its fitted parameters and the LLM agent's prompt fixed. The two traders earn similar profits under normal liquidity, but the LLM agent's profits decline less as liquidity falls. This pattern is consistent with a benefit from decision-time reasoning when conditions differ from those used to train the task-specific policy.

We conduct two further analyses. First, we analyze the extent to which adaptation can mitigate the speed–reasoning tradeoff. We develop a "reflection" system where a high-reasoning agent evaluates session performance and provides feedback to the next trading agent in the form of a playbook, motivated by techniques from the computer science lit-

erature (Shinn et al., 2023). Reflection substantially improves subsequent average trading performance. However, more decision-time reasoning does not consistently improve profits even after reflection, and reasoning is still the most valuable in the slowest markets.[4]

Second, we analyze additional models from OpenAI and Anthropic to address the natural concern that the tradeoffs we identify reflect idiosyncratic features of `gpt-5.4` rather than a general property of agentic trading. The varying effects of reasoning are robust, but model behavior varies in ways that are themselves informative. For the Anthropic models we test, performance is highest with extended thinking turned off and declines as the reasoning budget rises. For smaller and less capable models in the OpenAI family, by contrast, additional reasoning can compensate for weaker baseline capability.[5]

Our main contribution is to show how the economic value of decision-time reasoning in AI trading depends on the time available to act and the tasks assigned to the agent. Additional reasoning can improve trading judgment, but the resulting delay can prevent those gains from translating into realized profits. Our comparisons show that delegating calculations and execution to algorithms improves performance, while retaining LLM tender judgment can provide greater resilience than a task-trained policy when market conditions change. The paper contributes to several literatures.

First, the paper contributes to a growing literature on AI in financial markets. Dou, Goldstein, and Ji (2025a), Dou, Goldstein, and Ji (2025b), and Dou et al. (2025) model AI traders as reinforcement-learning systems and study the equilibrium and welfare consequences of their behavior. Related work studies feedback loops and price dynamics when traders interact with learning algorithms (Banerjee and Szydlowski, 2025). Empirical work also uses deep learning and reinforcement learning as financial decision technologies, including deep sequence models for asset pricing and direct reinforcement-learning approaches to portfolio construction (Cong et al., 2021a,b). We extend this program by studying how LLM agents apply general-purpose reasoning to trading decisions and how the market-time cost of that reasoning affects performance. A separate line of work uses large language models as

[4]We also examine whether retraining the NN trader can mitigate the costs of exogenously imposed decision delays. Retraining reduces these costs, but longer imposed delays continue to lower profits after retraining.

[5]In the Online Appendix, we extend the design to a second task where the agent must decide how to route orders across two exchanges, and find similar reasoning patterns.

predictive instruments: Lopez-Lira and Tang (2025) show that a general-purpose LLM can extract return-relevant information from text, and Lopez-Lira (2025) moves from passive inference to active trading by studying LLM agents in simulated markets. Relative to this line of work, we separate model capability from performance in a real-time market and study how the value of reasoning depends on the time available to act on it.[6]

Second, the paper connects to research on reasoning, test-time compute, and the economics of AI. In computer science, additional reasoning, tool use, and structured agent architectures can improve performance on many tasks (Wei et al., 2022; Schick et al., 2023; Yao et al., 2023; Snell et al., 2025). Kang et al. (2025) identify related latency–quality tradeoffs in high-frequency trading and gaming environments, and Hwang et al. (2025) highlight inference latency as a deployment constraint for LLM-based autonomous-driving architectures. In economics, AI is increasingly studied as a cognitive technology whose value depends on its interaction with judgment, task design, and organizational context (Agrawal, Gans, and Goldfarb, 2019, 2025; Korinek, 2025; Acemoglu, Kong, and Ozdaglar, 2026; Dell'Acqua et al., 2026). We add a dynamic market dimension: even if reasoning improves the decision conditional on a fixed state, the state may change before the decision is executed. The economic value of reasoning therefore depends not only on whether it improves judgment, but also on whether the environment leaves enough time to act on it.

Finally, our work points to a broader research agenda. AI agents differ from earlier automated trading systems not only in performance, but in the process by which they generate decisions. Recent finance work has examined this process from other angles: Bhagwat et al. (2026) use LLM personas to elicit economically meaningful investor disagreement and link it to abnormal trading volume, Chen et al. (2025) use interpretability tools to identify the concepts that drive LLM economic forecasts, and Bini et al. (2026) study systematic biases and corrections in LLM economic and financial choices. We study a complementary economic constraint: whether the agent's reasoning can be converted into trading performance when the market continues to move while the agent thinks. The economic content of AI

[6]The trading task we study also relates to a long literature on automated execution, dealer intermediation, and electronic liquidity provision (Hendershott, Jones, and Menkveld, 2011; Budish, Cramton, and Shim, 2015; Almgren and Chriss, 2000; O'Hara and Zhou, 2021; Hendershott, Livdan, and Schuerhoff, 2021). We use this task family as a laboratory for studying agentic reasoning.

decision-time reasoning and the conditions under which that reasoning produces value in financial applications are promising areas for future research.

1 Institutional Setting and Experimental Design

1.1 The Trading Task

We study a stylized electronic principal-liquidity-provision task. In modern institutional markets, intermediaries receive client requests electronically, decide whether to commit balance sheet, and, conditional on acceptance, must manage the resulting inventory through subsequent execution (O'Hara and Zhou, 2021; Hendershott, Livdan, and Schuerhoff, 2021). The task captures features of modern electronic liquidity provision: rapid response to client flow, automated risk management, and execution across order-driven markets.[7]

The task has a natural two-stage structure. The trader must first decide whether to accept the client request, using the quoted edge, displayed depth, remaining time, and current inventory. Conditional on acceptance, the trader must then unwind the resulting position in the market by choosing order type, pacing, and, in some treatments, venue. This separation is valuable because it lets us distinguish two effects of reasoning. More reasoning can improve the quality of tender selection and inventory management. But reasoning also takes time, and during execution that delay is costly because the market continues to evolve while the agent thinks. The task therefore lets us observe both the *reasoning dividend*—the improvement in decision quality due to reasoning—and the *deliberation tax*—the cost of delay incurred by reasoning—in a setting where each is economically meaningful.

1.2 Platform and Simulated Market Environment

We implement the trading task in RIT (Rotman Interactive Trader), a pre-existing order-driven market simulator designed for real-time decision making in finance (Mak and McCurdy, 2019). RIT operates as a continuous double auction with price–time priority, sup-

[7]For examples of firms emphasizing these activities, see `https://www.citadelsecurities.com/what-we-do/fixed-income-and-fx/`, `https://www.janestreet.com/what-we-do/overview/`, and `https://www.virtu.com/`.

ports limit and market orders, allows the market administrator to scale the market clock, and records participant actions at high frequency. These features make the platform well suited to our design because performance depends on decisions made under a real-time clock.

RIT also has the advantage of being a long-standing institutional simulator rather than an environment custom-built for this paper or tuned to the strengths of language-model agents. Rotman describes the RIT Market Simulator as a server–client centralized limit-order-book environment in which users submit orders, the server aggregates participant and AI order flow, and market-clearing prices are determined and recorded at millisecond frequencies.[8] The associated decision cases span buy-side and sell-side roles, including agency trading, liability trading, algorithmic trading, fixed income, commodities, derivatives, portfolio management, and risk management, and are designed to run through repeated scenarios with decision-support templates linked to the simulated markets through RTD and API connections.[9] Overall, our experiments use a documented, institutionally sponsored, longstanding market simulator with pre-existing cases, not a bespoke environment designed around the AI agents studied here.[10]

Our baseline environment uses the LT3 (Liability Trading 3) case. In LT3, the trader manages a liability book in two stocks, CRZY and TAME, over a five-minute, 300-tick trading session. During the session, the trader receives a sequence of institutional tender offers. Each tender is an offer to buy from or sell to the trader at quoted prices that vary in spread relative to prevailing market prices. The trader may accept or decline the tender. If the tender is accepted, it immediately creates an inventory position that must then be unwound in the market using market, limit, or marketable limit orders. The case implements fines for front-running, speculative trading unrelated to unwinding accepted tenders, and for carrying positions into the end of the trading session. It also imposes economically meaningful trading constraints, including net and gross position limits of 100,000 and 250,000 shares,

[8] See the Rotman RIT overview: `https://www.rotman.utoronto.ca/faculty-and-research/education-labs/bmo-financial-group-finance-research-and-trading-lab/rit-market-simulator/overview--features/`.

[9] See the Rotman decision-case overview: `https://www.rotman.utoronto.ca/faculty-and-research/education-labs/bmo-financial-group-finance-research-and-trading-lab/rit-market-simulator/decision-cases/`.

[10] See `https://www.rotman.utoronto.ca/faculty-and-research/education-labs/bmo-financial-group-finance-research-and-trading-lab/rotman-international-trading-competition/sponsors/`.

maximum order sizes, transaction fees, and a tender decision window of approximately 30 seconds. Online Appendix Table A1 reports the full LT3 case parameters used in the baseline experiments.

Performance depends on decision quality in tender selection and execution skill rather than on forecasting market direction; the trader does not receive directional information about future prices. LT3 is a controlled environment that allows us to measure the effects of speed and reasoning on these performance dimensions. Importantly, we do not claim a literal mapping from simulator seconds to any particular real-world market.

1.3 AI Architecture and Reasoning Treatments

We connect frontier large language models to the market through a thin adapter layer. On every decision cycle, including cycles with no active tender, the adapter retrieves and presents the limit order books for both securities, inventory, active tenders, outstanding orders, and time remaining. It combines this market information with a system prompt that describes the trader's role, constraints, and available tools, and submits the resulting context to the model. The model's response is then translated back into market actions. The action set is intentionally narrow: the agent can accept or decline a tender, submit an order, or cancel an existing order. The adapter therefore connects the model to the simulator without hard-coding a trading policy. Figure 2 summarizes the two-stage tender-execution task and the agent architecture.

The prompt specifies the trading role rather than an execution algorithm. It instructs the agent to evaluate tenders, manage inventory, avoid speculation and front-running, and use available order types to unwind accepted positions. But it does not prescribe a fixed rule for when to accept, how aggressively to trade, or how to split execution across time. Online Appendix Table A2 reports the baseline LT3 playbook prompt supplied to the agent alongside the market information that is provided to the agent.

We study frontier models from OpenAI and Anthropic under three reasoning settings: off, low, and medium. In all three settings, the agent receives the same market-state input and has the same action set. The treatment varies the requested amount of additional

deliberation before the model returns an action.[11] Within each trading session, the model receives the current market state together with the history of earlier market observations and its previous responses, including trading actions. For the baseline OpenAI agent, we retain this conversation history through the Responses API. The model can thus condition on the accumulated session history, including earlier tenders, prior actions, and their outcomes. Our baseline experimental analysis focuses on `gpt-5.4-2026-03-05`; provider and model portability are studied as robustness exercises.

1.4 Experimental Design and Protocol

Our main design crosses reasoning intensity with market speed. The reasoning treatments are off, low, and medium. The speed treatments are slow, normal, and fast. Because agentic systems reason in wall-clock time, realized think time is the natural object in our setting. The speed treatment determines how much simulated market time elapses during a given amount of wall-clock deliberation.

Specifically, let $\lambda_{env} \in \{0.5, 1.0, 1.5\}$ denote the number of simulation ticks per wall-clock second in each environment. One second of wall-clock time corresponds to 0.5, 1.0, and 1.5 simulation ticks in the slow, normal, and fast treatments, respectively. Equivalently, one simulation tick lasts 2.0, 1.0, and 0.67 seconds of wall-clock time. If an agent spends t_{think} seconds thinking, that delay consumes $\lambda_{env} t_{\text{think}}$ simulation ticks: $0.5\, t_{\text{think}}$ in the slow treatment, $1.0\, t_{\text{think}}$ in the normal treatment, and $1.5\, t_{\text{think}}$ in the fast treatment. The slow market therefore halves the simulation-time cost of deliberation relative to the normal market, while the fast market increases it by 50%. We disclose how much wall-clock time elapses per simulator tick to the agent in the system prompt to ensure it understands its environment.[12]

In total, the main speed experiment contains nine treatment cells: reasoning $\in$ {off, low, medium} crossed with speed $\in$ {slow, normal, fast}. In a normal-speed market with $\lambda_{env} = 1$, each run of the 300-tick LT3 case lasts for five minutes of wall-clock time. In the slow treatment with

[11] The agent uses decision-time cognition under all settings, including the off setting. The low and medium settings ask the model to deliberate more before responding. We do not include a high-reasoning treatment in the main specification because at high reasoning the associated latency dominates acceptance decisions.

[12] The adapter also appends a fixed one-tick sleep between turns, pinned to the simulator's refresh cycle and held constant across treatments. Total wall-clock time per turn is therefore $1/\lambda_{env} + t_{\text{think}}$ seconds. In simulation time, this equals $1 + \lambda_{env} t_{\text{think}}$ ticks.

$\lambda_{env} = 0.5$, the same 300-tick run lasts for ten minutes; in the fast treatment with $\lambda_{env} = 1.5$, it lasts for three minutes and 20 seconds. These treatments should be interpreted as varying the amount of market evolution that occurs during a fixed amount of wall-clock deliberation within the simulator, rather than as imposing a literal calibration from simulator seconds to any particular real-world market. System prompts are identical across cells save for the disclosure of how much wall-clock time elapses per simulator tick.

This design separates a potential reasoning dividend from a deliberation tax. The dividend should appear most clearly in decisions that reward judgment, such as which tenders to accept and how to assess inventory and penalty risk. The tax should appear most clearly in timely follow-through and position unwinding, because additional thinking leaves less time before a tender expires, the limit order book moves, or the session ends. The net effect on performance depends on the balance of these two margins at a given market speed.

The unit of observation is a session: one complete 300-tick run of a RIT case played by one trader. Across sessions, the structural case parameters are held fixed within an experimental cell, while the RIT environment generates randomized price paths and tender realizations for each run (i.e., no pairing). Each treatment cell contains 50 sessions. All language-model agents use the common observation-and-action protocol described above. We randomize cell order before execution.

The experiments are organized as treatment sweeps rather than as one fully crossed design. The primary design is the `gpt-5.4` reasoning-by-speed grid. The normal-speed `gpt-5.4` cells also serve as the baseline for the response-time analysis. Additional sweeps vary the architecture around the agent to study how moving tasks out of the decision-time reasoning loop shifts the speed–reasoning tradeoff, with deterministic algorithms as limiting points. We then run separate robustness sweeps across OpenAI model scale and Anthropic models. Because provider APIs implement reasoning controls differently, we interpret the provider comparison as a robustness exercise rather than as a matched reasoning-budget experiment.

2 Main Results

2.1 Summary Statistics

We begin with the reduced-form performance patterns for the baseline frontier agent, `gpt-5.4`. The main outcome is net liquidation value (NLV), defined as Trading P&L net of fines. Fines are economically important in this environment: they capture realistic costs from failing to respect trading mandates including front-running, speculative trades, and, importantly, carrying inventory into the end of the trading session. We also report Trading P&L separately because an agent can appear effective at extracting tender value while still destroying net performance through incomplete execution.

Table 1 reports cell-level means across the full reasoning-by-speed grid. Three facts stand out. First, the agent is economically active and can generate positive net performance. In the normal-speed market, mean NLV is \$8.3 thousand with reasoning off and rises to \$18.6 thousand under low reasoning. Low reasoning also performs consistently across speeds: mean NLV is \$22.9 thousand in the slow market, \$18.6 thousand in the normal market, and \$14.0 thousand in the fast market, and it has the highest "Sharpe ratio," which we use as a shorthand for mean scaled by standard deviation of performance, in each speed environment. The mean-performance ranking, however, depends on speed. In the slow market, medium reasoning earns the highest mean NLV, \$27.8 thousand, although its Sharpe ratio is slightly below that of low reasoning. This pattern is consistent with some real-time reasoning being useful in this task, but also suggests that the value of additional reasoning depends on how much market time the agent has to act.

Second, the relationship between reasoning and performance is not monotone. Medium reasoning does not dominate low reasoning. In the normal-speed market, medium reasoning produces higher Trading P&L than low reasoning, but also substantially higher fines, leaving mean NLV below the low-reasoning cell and slightly below the reasoning-off benchmark. In the fast market, the pattern is sharper: medium reasoning earns positive Trading P&L of \$14.5 thousand, but fines rise to \$33.0 thousand, producing mean NLV of −\$18.4 thousand. In the slow market, by contrast, medium reasoning earns \$34.5 thousand in Trading P&L and pays \$6.6 thousand in fines, producing the highest mean NLV in the grid. Thus the raw

means already suggest that reasoning can improve some trading decisions, but that realized performance depends on whether the market gives the agent enough time to act on those decisions.

Third, the time cost of reasoning is large enough to be economically relevant. Average think time increases from roughly 2.7–3.2 seconds per turn with reasoning off to roughly 3.7–3.8 seconds under low reasoning and 5.7–8.4 seconds under medium reasoning, depending on the speed cell. Medium reasoning also accepts fewer tenders in the fast market than either off or low reasoning. These facts do not by themselves identify the cost of delay, but they show why latency is not a minor implementation detail. It is a treatment outcome that moves with reasoning intensity and can directly affect performance.

2.2 Reduced Form Evidence

Table 2 provides a reduced-form view of the speed-reasoning tradeoff in the normal-speed market, where the speed environment is held fixed. Columns (1), (3), and (5) regress NLV, Trading P&L, and fines on reasoning indicators alone. These specifications reproduce the non-monotone pattern in the raw means. Low reasoning raises NLV by about $10.3 thousand relative to reasoning off, while medium reasoning has essentially no raw NLV advantage. The decomposition shows why: low reasoning reduces fines, whereas medium reasoning raises Trading P&L but also raises fines.

Columns (2), (4), and (6) add log average think time per turn. We interpret these specifications cautiously because think time is a realized, post-treatment variable. The coefficients are not causal mediation estimates. They are nevertheless informative about the two margins that govern agent performance. Holding realized think time fixed, both low and medium reasoning are positively associated with NLV and Trading P&L. At the same time, longer think time is strongly negatively associated with NLV and Trading P&L and positively associated with fines. In economic terms, the table suggests that reasoning has a beneficial decision-quality component, but that the realized delay associated with reasoning is costly.

Figure 3 gives the visual counterpart to this regression. Each point is a normal-speed session, plotted by average think time and NLV. Within each reasoning level, sessions with

longer average think time tend to have lower NLV. The medium-reasoning sessions also show greater dispersion and a thicker left tail, consistent with the idea that longer deliberation creates downside risk in a real-time execution task. The figure is descriptive, but it summarizes the empirical tradeoff: the relevant frontier is not model capability alone, but the joint effect of reasoning quality and the time required to produce it.

Taken together, the reduced-form evidence motivates the main experimental analysis. The raw performance facts show that low reasoning can improve performance, but that more reasoning is not automatically better. The response-time regressions show that the same treatment that improves decision quality also changes the amount of market time consumed before action. We therefore next use the experimental speed variation to test the speed–reasoning frontier directly.

2.3 Experimental Evidence

The reduced-form evidence shows that reasoning intensity and realized response time move together. The summary statistics and Figure 1 illustrated the full reasoning-by-speed grid across the experimental variation. We now turn to formal tests across the grid.

Table 3 formalizes the speed and reasoning comparisons using NLV contrasts across cells. Panel A asks, within each reasoning level, how performance changes as the market slows relative to the fast-speed benchmark. The speed effects are weak when reasoning is off and under low reasoning. Under medium reasoning, the pattern is different. The joint speed test is statistically significant, and mean NLV is $25.8 thousand higher in the normal market than in the fast market and $46.3 thousand higher in the slow market than in the fast market. The pooled specification with reasoning fixed effects shows the same direction: the slow market raises NLV by $15.7 thousand relative to the fast market.

Panel B turns the comparison around and asks, within each speed, how reasoning levels compare. We take low reasoning as the reference point because it is the strongest reliability benchmark in the main grid: it has the highest Sharpe ratio in every speed environment and the highest mean NLV in the normal and fast markets. The reasoning contrasts are speed-dependent. In the fast market, the joint reasoning test is statistically significant, but reasoning can hurt: medium reasoning lowers mean NLV by $32.5 thousand relative

to low reasoning, while reasoning off is $3.4 thousand below low reasoning and statistically imprecise. In the slow market, the joint reasoning test is statistically significant for a different reason: medium reasoning is $4.9 thousand above low reasoning, while reasoning off is $20.3 thousand below low reasoning. In the pooled specification with speed fixed effects, both medium reasoning and reasoning off underperform low reasoning, by $12.9 thousand and $11.3 thousand, respectively.

Tables 4 and 5 show that speed and reasoning shift not only means, but the entire NLV distribution. In Table 4, slower markets compress the lower tail and performance range, especially under medium reasoning. The fast-medium cell has a first quartile of −$50.0 thousand and a median of −$9.0 thousand, even though its third quartile is positive. Moving from fast to slow speed raises the first quartile by $56.7 thousand, the median by $44.9 thousand, and the third quartile by $18.8 thousand, while reducing the interquartile range by $37.9 thousand. Under low reasoning, moving from fast to slow raises the first quartile by $26.5 thousand and reduces the interquartile range by $29.2 thousand. The pooled quantile regressions show the same broad pattern: the slow market raises the first quartile and median relative to the fast market while compressing the interquartile range. Figure 4 reinforces this point visually.

Table 5 gives the companion view along the reasoning axis. In the fast market, moving from low to medium reasoning lowers the first quartile by $40.2 thousand and the median by $32.6 thousand relative to low reasoning, indicating that the fast-market cost of medium reasoning is concentrated in the lower part of the performance distribution. In the slow market, by contrast, medium reasoning is statistically similar to low reasoning across the reported quantiles, with higher point estimates for the median and third quartile as well as higher mean NLV. Reasoning off performs worse than low reasoning in the slow market, lowering the first quartile, median, and third quartile. The pooled quantile regressions summarize the same tradeoff: medium reasoning lowers the lower tail and median relative to low reasoning and increases the interquartile range, even though it can create upside when market time is abundant.

Overall, this evidence is consistent with a speed–reasoning frontier: the value of additional reasoning depends on whether the market gives the agent enough real time to act on the

resulting decisions. Low reasoning performs best in the normal and fast markets. Medium reasoning is costly in the fast market, where its deliberation cost shows up in both mean performance and the lower tail of session outcomes. In the slow market, however, medium reasoning is no longer a net liability and has the highest mean NLV. We next examine the mechanism by separating the tender-selection margin from the execution and unwind margin.

2.4 Mechanism: Tender Selection vs. Execution Quality

The total performance results combine several decisions that occur at different points in the tender lifecycle. The agent must first observe a tender, decide whether the quoted price is attractive relative to the current book, successfully submit the accept decision before the tender expires, and then unwind the resulting inventory before the end of the session. Here, we study each mechanism in detail.

2.4.1 Tender Selection and Fills

Table 6 studies tender selection and responsiveness. There are two key objects in this analysis. The first is the signal that the agent sends to the system to accept an offer, or "intent," given the proposed tender price and what it sees in the market. The second is the "book edge," or the spread between the weighted price of walking the tender through the current book and the tender price. Offers with a greater spread are more attractive, all else equal.

Panel A suggests the agent is conditioning on economically relevant information in the book, particularly when reasoning is on. It reports live-turn descriptive statistics, with one observation per agent turn while a tender is alive and before the agent has decided. The edge on turns with accept intent is higher than the edge on no-intent turns in every cell. In the normal market, for example, the mean edge on intent turns is \$0.1120 per share with reasoning off, \$0.1262 under low reasoning, and \$0.1290 under medium reasoning, while the corresponding no-intent edges are much lower, particularly when reasoning is on. Moreover, reasoning also tends to reduce the rate at which the agent accepts negative-edge tenders.

Panel B of Table 6 provides sharper regression-based evidence that reasoning increases

the sensitivity of intent to edge. Current observed edge strongly predicts accept intent, with a greater slope when reasoning is on. Specifically, in the specification with reasoning-by-edge interactions, the edge slope is 0.683 under reasoning off, and the slope increases by 1.851 under low reasoning and by 1.827 under medium reasoning. The implied edge slopes under low and medium reasoning are therefore very similar, at roughly 2.53 and 2.51, and much steeper than the slope under reasoning off. Figure 5 visualizes the same result: the accept-intent schedule is much flatter with reasoning off and substantially steeper under low and medium reasoning.

This is consistent with a reasoning dividend: reasoning changes the sensitivity of the agent's actions to economically relevant information at the decision point. At the same time, Figure 5 makes clear that low and medium reasoning have higher baseline accept propensity at zero edge as well as stronger edge sensitivity. Thus, while reasoning makes the agent more sensitive to edge, reasoning does not necessarily make every additional accept intent optimal.

Not every intent signal sent by the agent is filled by the system, for one of two reasons: either the tender has expired during the agent's own think time, or the implied post-fill position would breach the agent's position limit. In our sample, of the 71 failed intents, 52 occur because of tender expiration, and 19 occur because of the position limit.

Panels A and B show that greater intent-failure rates occur with medium reasoning. The percentage of failed intents for medium reasoning is 11% and 16% in normal and fast markets, compared to rates of 1–3% for off- and low-reasoning agents. The differences across reasoning modes largely disappear in the slow market, where all reasoning levels have a 1–2% intent-failure rate. Column (3) of Panel B provides regression-based evidence at the tender level. Low reasoning has essentially no effect on failed-intent probability, with a coefficient of -0.3 percentage points, while medium reasoning increases failed-intent probability by 7.1 percentage points. This pattern is consistent with an operational failure that can occur when real-time reasoning consumes a meaningful share of a fixed tender window.

2.4.2 Execution Quality and Overall Lifecycle

Table 7 studies post-fill execution quality. Panel A reports the post-fill unwind outcomes. Medium reasoning has the weakest unwind completion rates in the two faster environments: 78% in the normal market and 68% in the fast market, compared to a range of 83–89% for the off- and low-reasoning agents at those market speeds. In the slow market, completion rates are high across the board: 92 percent for reasoning off, 97 percent for low reasoning, and 91 percent for medium reasoning. This pattern matches the speed-frontier interpretation: when market time is less scarce, the same reasoning budget is less likely to leave the agent with residual inventory.[13]

Panel B columns (1)–(3) separate completion from conditional execution quality. Medium reasoning lowers the probability of completing the unwind by 9.8 percentage points, conditional on speed fixed effects, while low reasoning has essentially no completion effect. But among completed unwinds, both low and medium reasoning raise realized edge capture.[14] Controlling for book edge at the decision turn, low reasoning increases edge captured by \$0.024 per share and medium reasoning increases it by \$0.047 per share. Low and medium reasoning also raise the book edge at intent, by \$0.0126 and \$0.0155 per share, respectively. The table therefore shows both sides of the mechanism: reasoning improves selection and conditional execution quality, while medium reasoning creates more lifecycle attrition.

We also examine implementation shortfall (IS) to gauge the overall round-trip performance of the agent, defined following Perold (1988) and Hasbrouck (2007) as the per-share gap between paper-portfolio P&L and realized P&L, benchmarked against the bid-ask midpoint at the moment of acceptance.[15] IS provides a natural way to incorporate stuck unwinds

[13] We count a tender as having a completed unwind if the agent's position returns to pre-acceptance level before session end or next tender acceptance of the same ticker. Online Appendix Table A3 reports a fixed-window edge-capture check using an alternative measure that calculates ticker P&L over a deterministic 30-tick window after acceptance. This check addresses the concern that the completed-unwind measure could absorb variation in unwind duration. The alternative evidence is consistent with Table 7.

[14] Edge Captured is the realized per-share profit on a tender round-trip, computed as the change in the ticker's P&L between the acceptance turn and the unwind-end turn, divided by the tender quantity. The unwind-end turn is the first post-acceptance turn at which the ticker position has returned to its pre-acceptance level; if the agent fails to return to its pre-acceptance level before session end or the next same-ticker acceptance, the unwind is classified as stuck and excluded from Edge Captured. The measurement is gross of trading fees.

[15] Specifically, IS per share is $\frac{1}{Q}\big[-s(P_t-\pi_0)Q+s(\bar{p}_u-\pi_0)u+s(\pi_1-\pi_0)r+0.02u+1.00r\big]$ for a tender of side $s\in\{+1,-1\}$, quantity Q, fill price P_t, where π_0 is the acceptance midquote, π_1 is the post-stop settlement

into the analysis: the auto-liquidation cost on residual inventory enters as a separate component, distinct from the execution cost on traded shares. Panel A shows that off and low reasoning produce profitable round-trips, measured by negative IS, at all three speeds. Medium reasoning is also profitable in the slow market, with average IS of −\$0.090 per share, and is roughly break-even in the normal market. The fast/medium cell is the only condition where the trader incurs a clear net loss against the paper-portfolio benchmark, with average IS of \$0.158 per share. This reflects a stuck-unwind rate of 32 percent at fast/medium, where residual-inventory costs more than offset the edge captured on completed unwinds.

Panel B column (4) shows the corresponding regression on per-share IS. Medium reasoning increases shortfall by \$0.0647 per share on average, while the low-reasoning coefficient is negative and not statistically precise. The positive medium-reasoning coefficient is the net of two forces: medium reasoning improves execution on completed unwinds, but it also produces more stuck unwinds and residual-inventory penalties. Overall, the magnitude of the net shortfall effect depends on whether the simulation's pace gives the agent enough real-time budget to finish the unwind.[16]

To summarize, Figure 6 illustrates the lifecycle of a tender. Pooling across speeds, off and low reasoning produce 494 and 482 clean completed unwinds, respectively. Medium reasoning produces only 390 completed unwinds. It also produces more stuck unwinds, with 93 compared with 59 for off and 55 for low, and many more accept intents that expire during think time, with 45 compared with 1 for off and 6 for low. The figure shows why selection quality and total NLV can diverge. Medium reasoning can make more economically informed decisions, but many fewer tenders survive the full path from observation to accepted fill to clean unwind.

midquote, $\bar{p}_u$ is the unwind VWAP on u shares actually traded, $r = Q - u$ is the residual auto-liquidated at session end or before accepting another tender for the same ticker, and the last two terms are trading fees and the simulator's residual inventory penalty of \$1/share. Positive IS represents a drag on realized P&L relative to the paper portfolio. The trader is a liquidity provider, so negative IS values represent profitable round-trips.

[16] The pooled medium-reasoning coefficient on IS in column (4) masks substantial heterogeneity by speed. The fast/medium cell, where stuck-unwind rates are highest, drives the pooled positive coefficient.

2.5 Reasoning Traces

Table 8 and Figures 7 and 8 provide complementary evidence on what the model does with additional reasoning. We classify each model's reasoning summary (or, in the case of reasoning-off, the text output returned by the model) into five categories using Claude Sonnet 4 as a judge/classifier.[17] For brevity, we focus on the normal market speed. The classifier evidence is not a performance mediation design, and we do not interpret it causally. It is useful as a check that the reasoning treatments change the content of the agent's deliberation in economically sensible ways.

Low and medium reasoning increase discussion of post-accept unwind feasibility by 24.4 and 29.4 percentage points, respectively. They also increase discussion of aggressive unwinding reasoning by 12.9 and 15.6 percentage points. Medium reasoning raises explicit time awareness by 9.9 percentage points and position-limit awareness by 2.0 percentage points. Figure 7 plots these coefficients, and Figure 8 shows representative traces in which the agent evaluates tenders, unwinds positions, and sometimes waits rather than immediately cross the spread after prices move against it.

Overall, the evidence to this point paints a clearer picture of the speed–reasoning frontier. Reasoning improves the agent's use of economic information: it makes accept decisions more sensitive to current edge, increases discussion of unwind feasibility, and is associated with higher edge capture conditional on successful completion. But lifecycle attrition is concentrated under medium reasoning: tenders expire during deliberation, accepted positions are more likely to remain stuck, and the lower tail of session performance deteriorates when market time is scarce. The mechanism is therefore not "better reasoning" or "slower response" alone, but rather a tradeoff between improved judgment and the market-time cost of turning that judgment into completed trades.

[17] We use Claude to judge the reasoning traces of OpenAI models to guard against any potential bias from having a model judge the reasoning traces of models in the same family.

3 Agent Architecture

3.1 Moving Work Out of the Reasoning Loop

The mechanism results show that decision-time reasoning affects both the quality of the agent's intended decisions and the likelihood that those decisions are converted into completed trades. This raises an architecture question: which parts of the task should remain inside the agent's real-time reasoning loop, and which parts should be compiled into the surrounding system? We study this question by comparing four architectures in the LT3 normal-speed market.

The first architecture is the baseline agent studied above. The model observes the market state and chooses whether to accept tenders, submit orders, cancel orders, or wait. The second architecture provides decision support to the agent by adding pre-computed calculations to the agent's observation. This decision-support block reports book-based tender edge, projected inventory, limit-breach indicators, and flattening costs, but does not recommend an action.

The third architecture is agentic tender selection with algorithmic execution. The agent is invoked only when a tender is live and decides whether to accept each tender. It receives the same decision support as in the second architecture.[18] Execution is then handled separately: a compiled unwind algorithm reduces any inventory back to zero by dispatching 5,000-share market orders each tick.[19] Thus, the agent retains responsibility for the tender decision, but it is not in the continuous trading loop.

The fourth architecture is a fully compiled deterministic algorithm: it accepts tenders with a minimum 5-cent book edge net of fees and executes the same compiled unwind logic. This benchmark is distinct from the reasoning-off agent: in the off agent, the LLM still interprets the market state and chooses actions at decision time, while the algorithm applies a fixed tender rule and unwind routine. The algorithm is not intended as a horse race against the agent, but as the limiting case for what happens when both tender selection and

[18] Online Appendix Tables A4 and A5 describe prompts for the agents in the second and third architectures. In this architecture, the agent receives a fresh market snapshot whenever it is invoked during a live tender, and does not receive previous snapshots.

[19] Bertsimas and Lo (1998) and Almgren and Chriss (2000) show that variants of this strategy can be cost-minimizing under certain assumptions.

execution are specified before the session begins.

Table 9 and Figure 9 report the results. Moving calculations out of the reasoning loop in the second architecture improves the agent substantially over the baseline. In the low-reasoning cell, mean NLV rises from $18.6 thousand for the baseline agent to $30.4 thousand with pre-computed calculations. Fines fall from $7.0 thousand to $1.5 thousand, the standard deviation of NLV falls from $46.3 thousand to $21.5 thousand, and the profitable-session rate rises from 74% to 96%. The gains are not limited to the low-reasoning cell: mean NLV also rises under reasoning-off and medium reasoning.

Agentic tender selection with algorithmic execution has an even larger performance boost over the baseline. With compiled unwind, the medium-reasoning agent earns mean NLV of $48.5 thousand, pays zero fines on average, and has a Sharpe ratio of 2.38. The reasoning-off agent with compiled unwind earns a slightly lower mean NLV of $44.5 thousand, is profitable in every session, and has the highest Sharpe ratio in this architecture, 2.85. These outcomes are competitive with a fully compiled algorithm that uses a 5-cent threshold to accept tenders, which earns mean NLV of $44.5 thousand with a Sharpe ratio of 2.49. An algorithm that uses a 10-cent threshold to accept tenders earns a mean NLV of $44.0 thousand with a Sharpe ratio of 1.77. Figure 9 shows the same pattern visually.[20]

Panel B of Table 9 formalizes the architecture comparisons using the cells with the highest mean NLV for each architecture, with the baseline low-reasoning agent as the omitted category. Decision support via pre-computed calculations with low-reasoning raises mean NLV by $11.8 thousand, although this estimate is imprecise, and reduces both the NLV interquartile range and fines. Agentic tender selection with algorithmic execution and medium-reasoning raises NLV by $29.9 thousand, increases Trading P&L by $23.0 thousand, and reduces fines by $6.9 thousand. The fully compiled algorithm is the compiled-policy endpoint and also improves substantially on the baseline agent, with mean-NLV gains that are comparable with the agentic-selection architecture.

These results sharpen the interpretation of the speed–reasoning frontier. The frontier is

[20] Within this architecture, the mean and Sharpe rankings differ: medium reasoning has the highest mean NLV, while reasoning off has the highest Sharpe ratio. Because Panel B and Figure 9 report mean-performance architecture contrasts, we use the medium-reasoning cell as the representative Agent + DS + Algo Exec specification.

not fixed by the base model alone. It depends on what the agent is asked to compute and execute in real time. Decision support via pre-computed calculations reduces the burden of live evaluation. Agentic tender selection with algorithmic execution uses the agent where there is a potential reasoning dividend while mitigating the deliberation tax incurred by the child-order execution problem and LLM trading loop. With these supports in place, the agentic architecture improves significantly on the baseline agent and becomes competitive with the compiled-policy endpoint.

The implication of these results is not that agentic systems are better or worse than algorithms; it is that such systems should use decision-time reasoning where it belongs. Calculations and low-latency execution are natural candidates for compilation; agentic reasoning is most valuable where the remaining judgment is worth the delay.[21]

3.2 Comparing Task-Trained Algorithms with Agents

Building on the architecture results, we compare a neural-network (NN) trader trained specifically for LT3 with an LLM agent. Both systems are algorithms, and both have fixed parameters during evaluation. The NN trader's weights are optimized directly over actions and rewards specific to the LT3 task. By contrast, the LLM agent uses a general-purpose model trained to generate language and reason across tasks. The connection between that training and its trading actions is indirect: it applies these broader capabilities to the trading problem through decision-time reasoning.

We find that the NN trader acts with negligible delay and that increasing network size produces no detectable speed–performance tradeoff. However, we also find that the LLM agent's profits are more resilient when market conditions change beyond the NN trader's initial training environment. Thus, our results highlight that one can mitigate the speed-reasoning tradeoff faced by AI agents by employing task-trained algorithms, but that this choice induces a different tradeoff over resiliency.

[21] Online Appendix Tables A6 and A7 report a separate prompt-ablation exercise in which we make the baseline prompt less directive without adding decision support; the ablation addresses the concern that the main results are driven by wording that mechanically pushes the agent toward urgent execution rather than by the information structure of the task.

3.2.1 Neural-Network Training and Evaluation

The NN trader combines two learned decision rules, or *policies*. The selection policy determines whether to accept a tender, while the execution policy determines how to unwind the resulting inventory. Each policy uses a separate neural network. We train the execution network first because the value of accepting a tender depends on how the resulting position will be liquidated. We then train the selection network holding the execution policy fixed.

The Execution Policy. The execution policy unwinds an accepted position by choosing how quickly to liquidate an existing block of X shares over a window of H ticks, or until the session ends, whichever comes first.

Formally, let $t = 0, 1, \ldots, H$ denote the number of elapsed ticks since the arrival of X shares, and let x_t denote remaining inventory, with $x_0 = X$. At each tick $t < H$, the trader submits a fraction a_t of its remaining inventory as market orders. When these orders fill in full, inventory evolves according to $x_{t+1} = (1 - a_t)x_t$.

The execution objective is to maximize expected rewards,

$$\max_{\{a_t\}} \mathbb{E}\Big[\sum_{t=0}^{H} \gamma^t r_t\Big], \qquad r_t = -\frac{\mathrm{IS}_t}{X} \text{ for } t < H, \qquad r_H = -\Phi\frac{x_H}{X}. \tag{1}$$

where IS_t is the dollar implementation shortfall on shares traded at t, measured relative to the bid–ask midpoint when the block arrives and including the LT3's transaction fee of \$0.02 per share. The execution window H determines how long the trader has to liquidate the block. If fewer than H ticks remain in the session, the window ends with the session. In the LT3 case, each tender remains open for 30 ticks, and the arrival schedule leaves at least 15 ticks between its expiry and the earliest possible arrival of the next tender.[22] We therefore set $H = 15$ ticks to allow liquidation before the next tender opportunity, even when acceptance occurs near expiry. A longer window could reduce price impact but leave inventory outstanding when the next tender arrives, limiting the trader's capacity to accept it. The terminal penalty charges Φ dollars per share of inventory remaining when the

[22] Specifically, the five tenders in LT3 arrive at ticks drawn uniformly from 15–30, 75–90, 135–150, 195–210, and 255–270, and each can be accepted for 30 ticks.

execution window closes. Rewards account for shares traded within the window, while the terminal penalty adds a cost for shares retained. The policy therefore weighs liquidation costs against the cost of remaining inventory. We choose $\Phi = 50$ dollars per share using holdout cross-validation, a standard machine-learning procedure that selects hyperparameters based on performance on data excluded from model fitting.[23] We set $\gamma = 1$ so that the objective measures total liquidation costs over the execution window. During evaluation in RIT, the policy's parameters remain fixed, and inventory remaining at session end incurs LT3's 1-dollar-per-share fine.

The policy's input vector is time remaining, remaining inventory, a price input, and the previous liquidation fraction:

$$s_t = \Big(\tau_t/H,\ x_t/X,\ \tilde{m}_t,\ a_{t-1}\Big), \tag{2}$$

where $\tau_t = H - t$ is the number of ticks remaining in the pacing window and $\tilde{m}_t = (m_t - 100)/100$ is the normalized last-traded price. Time remaining and remaining inventory are the state variables of the standard liquidation problem (Bertsimas and Lo, 1998; Almgren and Chriss, 2000): the cost of a trading schedule depends on how many shares remain and how much time is left to trade them. The price input lets the policy react to price movements during the unwind. The previous liquidation fraction lets the policy account for its own recent trading. A large market order depletes the order book, which takes time to refill, so the cost of the next order depends on the size of the last one (Obizhaeva and Wang, 2013).

The policy chooses from an action set $\mathcal{G}$ containing 21 fractions of remaining inventory to liquidate: zero for waiting and 20 positive fractions spaced geometrically between 2% and 100%, including full liquidation. Smaller fractions are spaced more closely together, giving the policy finer control over gradual execution. The action value $Q(s, a)$ is the expected payoff from choosing liquidation fraction a in state s and following the optimal policy thereafter.

[23] Online Appendix Table A8 shows that this value yields the highest mean validation NLV for the baseline execution window of $H = 15$ ticks.

It satisfies the Bellman equation

$$Q(s,a) = \mathbb{E}\Big[r + \gamma \max_{a' \in \mathcal{G}} Q(s',a') \mid s,a\Big], \tag{3}$$

where r is the current reward and s' is the next state. At the last decision, the continuation value is the terminal payoff r_H in equation (1).

The trained policy chooses the liquidation fraction with the highest predicted value,

$$a_t = \arg\max_{a \in \mathcal{G}} Q_w(s_t, a). \tag{4}$$

The execution policy approximates Q with a feedforward network $Q_w(s,a)$, where w denotes its parameters. The network has two hidden layers of 128 rectified-linear units and uses the four state variables to estimate a value for each of the 21 actions. This is our baseline architecture; Table 10 reports results for alternative architectures. We train the network from a random initialization using fitted value iteration, a batch form of Q-learning. The method uses simulated execution costs and subsequent market states to approximate the action values in equation (3).

The Selection Policy. The selection policy decides whether to accept a live tender or wait, taking the execution policy, denoted ψ, as given. Accepting a tender creates an inventory position whose profitability depends on both the tender's price advantage and the cost of liquidating the position. The resulting inventory can also limit the trader's capacity to accept subsequent tenders, and inventory remaining at session end incurs a fine. We therefore fit the selection network to the trader's NLV over complete sessions, so that training accounts for these consequences of tender acceptance.

At each tick t at which a tender is live, the selection network observes four inputs: book edge net of transaction fees, time remaining in the session, gross inventory, and time since the previous acceptance. Its input vector is

$$z_t = \Big(\varepsilon_t,\ (T-t)/T,\ G_t/\bar{X},\ \min\{1, (t - t^{\text{acc}})/100\}\Big), \tag{5}$$

where $T = 300$ is the session length in ticks, G_t is gross inventory in shares, $\bar{X} = 71{,}000$ is the inventory normalization, and t^{acc} is the tick of the most recent acceptance. The variable ε_t is the per-share book edge defined in Section 2.4 minus the \$0.02 per-share trading fee. It measures the tender's price advantage after accounting for the estimated cost of walking the current order book.[24] Book edge measures the immediate value of the tender. The other three inputs describe the trader's capacity to take it on: the time left to unwind a block and to see further tenders, the inventory already held against the position limits, and how recently the trader accepted a block that the execution policy may still be unwinding. A fixed edge threshold ignores these three variables.

The selection network f_w maps the input vector z_t to a scalar score through one hidden layer of eight hyperbolic-tangent units, with 49 parameters w. The flexibility of neural networks creates a risk of overfitting the training sample. We use regularization to address this risk, as in high-dimensional asset pricing (Kozak, Nagel, and Santosh, 2020). We initialize the selection network to reproduce a simple book-edge acceptance rule and regularize its parameters toward these initial values. The penalty follows the same logic as shrinking portfolio policies toward a benchmark to reduce estimation error (Brandt, Santa-Clara, and Valkanov, 2009).[25]

The trader accepts the live tender when $f_w(z_t) \geq 0$, and otherwise waits. The trader evaluates the network once per tick while the tender remains live, so a tender that is not accepted at first can be accepted later as conditions change. A tender that is never accepted expires. Where f_w increases in book edge, the network accepts a tender when its edge exceeds a threshold that can depend on the other three inputs.

We obtain the initial parameters w_0 by fitting the network to a rule that accepts tenders with book edge net of fees of at least \$0.15. We choose this threshold because it yields the highest mean NLV on held-out simulated validation sessions among the thresholds.[26] With

[24] When depth is insufficient to absorb the full tender quantity, the remaining shares are priced at the deepest order book level.

[25] Relatedly, Li, Grandvalet, and Davoine (2018) penalize deviations from pretrained neural-network parameters when adapting the model to a new task.

[26] Online Appendix Table A8 presents the results.

$\sigma(u) = 1/(1 + e^{-u})$, the initial parameters solve

$$w_0 = \arg\min_{w} \; -\frac{1}{M} \sum_{m=1}^{M} \Big[y_m \log \sigma\big(f_w(z_m)\big) + (1 - y_m) \log\big(1 - \sigma(f_w(z_m))\big) \Big], \tag{6}$$

Here, z_m is a randomly generated input vector, and y_m equals one if its book edge net of fees is at least \$0.15 and zero otherwise. We generate $M = 4{,}000$ vectors by drawing each input independently and uniformly from specified ranges. This initial fit trains the network to reproduce the threshold rule.

Next, starting from w_0, we search for parameters that maximize penalized mean session NLV over $N^{\text{train}} = 24$ simulated sessions:[27]

$$J(w) = \frac{1}{N^{\text{train}}} \sum_{n=1}^{N^{\text{train}}} \text{NLV}^{(n)}(w; \psi) - \lambda_w \, \|w - w_0\|_2, \tag{7}$$

where $\text{NLV}^{(n)}(w; \psi)$ is the trader's NLV in training session n when it uses the selection rule defined by w and the fixed execution policy ψ. The first term measures average session performance. The second regularizes the network's parameters toward w_0 to limit overfitting to the training sessions.[28]

We maximize $J(w)$ using the cross-entropy method (CEM), a derivative-free search over the network's parameters. Starting from a normal distribution centered at w_0, each iteration draws candidate parameter vectors, evaluates $J(w)$ for each candidate on the same N^{train} sessions with the execution policy held fixed, and moves the distribution toward the best-performing candidates. The Online Appendix provides further training details under *Neural Network Trader* section.

Training and Evaluation Environments. We train in an offline simulator of the LT3 case rather than in RIT for two reasons. RIT runs in real time, so a session lasts five minutes at normal speed, while fitting the selection network evaluates thousands of simulated sessions and the simulator runs a session in under a second. Fitted value iteration also restarts the

[27] We use a common set of 24 simulated sessions to evaluate candidate policies, keeping the parameter search computationally manageable and reducing noise in comparisons across candidates.

[28] Online Appendix Table A8 reports validation results for alternative coefficients. Values of 100 and 300 produce identical NLVs. We conservatively choose $\lambda_w = 300$.

simulator from saved market states to try every action from the same state, which RIT does not allow. The simulator represents a 300-tick trading session in the two LT3 stocks. Each stock has an order book in which simulated traders submit limit and market orders around an externally specified price path. The simulator uses LT3's fees, order-size limits, position limits, and tender rules. To train the selection network, we replay the case file's price paths and five tenders, drawing new background order flow for each session. To train the execution network, we give the trader a 71,000-share position to liquidate along a simulated price path, with a deadline of 10 to 40 ticks.

3.2.2 Network Size and Trading Performance

We next examine whether increasing network size generates a speed–performance tradeoff analogous to that observed for the LLM agent. Larger networks allow greater flexibility in representing a decision rule but require more computation to evaluate. Table 10 reports trading performance and decision times as we vary the depth and width of the selection and execution networks.

We vary the architecture of one network while keeping the other at its baseline architecture. For the selection network, we consider one, two, or three hidden layers, with 8, 32, or 128 units per layer, while retaining the execution network's baseline architecture of two hidden layers with 128 units each. For the execution network, we consider one, two, or three hidden layers, with 32, 128, or 512 units per layer, while retaining the selection network's baseline architecture of one hidden layer with eight units. This gives nine specifications for each network. Each resulting trader is evaluated in 50 RIT sessions at each of the three market speeds, with its inputs and fitted parameters held fixed across speeds.

Larger networks do not systematically improve trading profits, nor does the effect of market speed differ detectably across network architectures. Neither joint test in Panel B rejects equality of the speed effects across architectures at the 10% level. Median decision times range from 0.32 to 0.68 milliseconds across specifications and market speeds. Even the largest execution network takes only 0.62 milliseconds in the fast market, approximately 0.09% of a tick. All of these networks therefore act with negligible delay at the market speeds studied. Thus, over the network sizes considered, increasing network size produces

no detectable speed–performance tradeoff.

3.2.3 Performance Under Changing Market Conditions and the Value of Decision-Time Reasoning

The preceding results show that the NN trader implements its task-trained policies with negligible decision delay. We next examine whether retaining LLM judgment adds value when market conditions differ from those used to train the NN trader. Continuing the thought experiment in Section 3.1, we compare tender selection while holding the execution policy common to both traders. The NN trader uses the trained NN selection policy; the LLM trading agent uses `gpt-5.4` with medium reasoning to decide whether to accept a tender.[29] Both delegate the unwinding of accepted positions to the same trained NN execution policy.[30]

We change market conditions by progressively reducing the typical liquidity in the limit order book. In LT3, liquidity is offered by uninformed computer-trader limit orders that appear in the order book at a given intensity. We reduce liquidity by reducing computer-trader order frequency while holding the market clock fixed. We consider three market conditions with 120, 60, and 30 orders per simulation minute, corresponding to normal, half, and quarter-frequency markets. All else equal, lower liquidity levels correspond to greater trading costs and thus lower expected profits. For each market condition, we evaluate both the NN trader and LLM agent in 150 independently drawn RIT sessions.

We train both NN policies under normal-liquidity conditions and hold their fitted parameters fixed across the three evaluation environments. Analogously, we do not vary the prompt for the LLM agent across evaluation environments. Our exercise thus compares the resiliency of these two architectures when we experimentally vary liquidity in the evaluation environment.

Table 11 shows that the two traders earn similar mean profits under normal liquidity: \$40.4 thousand for the NN trader and \$41.6 thousand for the LLM agent. Their performance

[29] The LLM receives current market observations together with pre-computed decision support on tender profitability, projected inventory, and position limits.

[30] The execution policy differs from that used in Table 9, where inventory is unwound using a fixed 5,000-share-per-tick market-order routine. Here, both the NN trader and LLM agent use the trained NN execution policy described in Section 3.2.1. The normal-liquidity results therefore come from different execution architectures and separate experimental samples.

diverges as liquidity falls. At half frequency, mean profits decline to $27.6 thousand for the NN trader and $34.5 thousand for the LLM agent. At quarter frequency, they decline further to $15.3 thousand and $25.8 thousand, respectively. The decline from normal to quarter frequency is $9.3 thousand smaller for the LLM agent, a statistically significant difference-in-differences ($p = 0.001$). Thus, both traders are adversely affected by reduced liquidity, but the LLM agent's profits are more resilient despite its lack of training specific to LT3.

The LLM agent also maintains more trading activity and a higher frequency of profitable sessions. From normal to quarter frequency, the NN trader's average number of accepted tenders falls from 2.2 to 1.0, while the LLM agent's falls from 4.2 to 3.2. At quarter frequency, the LLM agent remains profitable in 97% of sessions, compared with 65% for the NN trader. Both traders incur zero inventory fines throughout these treatments, so the differences in profits reflect trading P&L. With a common execution policy, the comparison concerns which tenders the traders accept and when they accept them.

These results identify a setting in which retaining the LLM for tender judgment adds value relative to the task-trained NN selector. The NN trader performs comparably under normal liquidity, while the LLM agent's profits are more resilient as liquidity deteriorates. This pattern is consistent with a benefit from decision-time reasoning when trading conditions differ from those used to train the task-specific policy. Together with the network-size results, the comparison highlights that task-trained policies offer negligible decision delay, while retaining decision-time reasoning can provide greater resilience to changing market conditions.

4 Adaptation and Robustness

4.1 Adaptation Through Reflection

The agent may also improve its decisions by learning from previous trading outcomes. We examine this form of adaptation through reflection between sessions. Our approach is related to a concept known as "Reflexion" from the computer science literature, which uses textual feedback from earlier trials to guide an agent's subsequent decisions without updating its

model parameters (Shinn et al., 2023).[31] Here, the feedback takes the form of a written playbook derived from the agent's trading record.

For this exercise, we return to the baseline LLM trader, which is responsible for both tender selection and inventory execution. We evaluate the agent in chains of three sessions, each with independently drawn price paths and tender schedules. After each of the first two sessions, a separate LLM reviews the trading record and revises the playbook supplied to the trader in the next session. Each chain starts without a playbook, and model parameters remain fixed throughout. The review takes place between sessions and therefore consumes no trading time; its conclusions are carried forward through the playbook. We run 50 chains for each of the twelve configurations: the nine reasoning–speed combinations for `gpt-5.4` and the three reasoning settings for `gpt-5.4-mini` at normal speed. The reviewing model is `gpt-5.4` with `xhigh` reasoning. Each playbook contains up to ten rules and replaces the previous version.

Reflection substantially improves average trading performance. Online Appendix Table A9 shows that third-session mean NLV exceeds the no-reflection baselines by $12 thousand on average across the twelve configurations. In the fast market, the medium-reasoning `gpt-5.4` agent's mean NLV rises from -$18.4 thousand to $8.6 thousand after two rounds of playbook revision. The largest gain is for `gpt-5.4-mini` with low reasoning, whose mean NLV improves by $40.6 thousand.[32]

Examining the playbooks themselves, we find that they often address the consequences of response time, including whether enough time remains to unwind an accepted position and how execution should change as the session end approaches. The feedback therefore makes these timing considerations explicit before the next trading session begins.

[31] Related work on iterative feedback and refinement includes Madaan et al. (2023), who study how an LLM can improve its outputs through feedback and revision without additional training.

[32] In Online Appendix Table A10, we also examine how the NN trader performs when its training explicitly accounts for decision delays. Since the NN trader's own inference time is negligible, we impose delays drawn from the LLM agent's observed response-time distributions for each reasoning and market-speed treatment. The NN trader observes the market and chooses an action, waits while the market evolves, and then submits that action without revising it. The imposed wait adds delay without additional reasoning. We compare networks trained without these delays (NN-baseline) with networks retrained under them (NN-retrained). Both the selection and execution networks are refitted separately for each treatment, retaining their architectures and state variables. Retraining raises profits and lowers inventory fines under the medium-reasoning delay distribution. However, moving from the off-reasoning to the medium-reasoning delay distribution continues to reduce the retrained trader's mean profits at all three market speeds.

More decision-time reasoning does not consistently improve profits even after reflection. Looking at the post-reflection third-session profits for `gpt-5.4`, low reasoning yields higher mean profits than medium reasoning at all three market speeds. Compared to reasoning off, medium reasoning raises mean NLV by approximately $8.1 thousand in the slow market but lowers it by $4.5 thousand in the normal market and $6.6 thousand in the fast market. Thus, even after reflection, additional decision-time reasoning is more valuable when the market leaves more time to act.

4.2 Providers and Models

The main results use `gpt-5.4` as the baseline agent. A natural concern is that the speed–reasoning pattern reflects idiosyncratic features of this model, its provider-specific reasoning control, or the particular mapping from the reasoning label to realized test-time compute. Our next set of tests show that our central insights are not specific to one frontier model by studying other OpenAI and Anthropic model families.[33]

Table 12 reports descriptive means in the normal-speed market for three OpenAI models and two Anthropic models.[34] Figure 10 plots the same evidence as connected means for NLV and think time. Three patterns emerge. First, performance varies substantially across models. The baseline `gpt-5.4` agent earns positive NLV in all three reasoning cells, with the highest at low reasoning. The smaller OpenAI models are less reliable: `gpt-5.4-mini` is negative under off and low reasoning but improves under medium reasoning, while `gpt-5.4-nano` generates low average NLV and is especially weak under low reasoning. The Anthropic models perform strongly without extended thinking: Opus earns mean NLV of $28.4 thousand with reasoning off, and Sonnet earns $30.9 thousand.

Second, more reasoning is not uniformly valuable. For the baseline `gpt-5.4`, low reasoning improves NLV relative to reasoning off, but medium reasoning gives back the gain. For the Anthropic models, the pattern is even sharper: both Opus and Sonnet have their highest

[33] In the Online Appendix, we also show that the tradeoffs of decision-time reasoning are not specific to the LT3 task by considering a second task related to smart-order-routing.

[34] For OpenAI models, the off condition omits the reasoning field, while the low and medium conditions use bounded reasoning settings subject to the output-token limits in the runner. For Anthropic models, the off condition disables extended thinking while setting low output effort; the low and medium conditions set low or medium output effort and enable adaptive extended thinking without an explicit thinking-token cap.

NLV with extended thinking off, and performance declines as the thinking budget increases. Medium reasoning is especially costly for Opus, where mean NLV falls to $4.7 thousand. In contrast, the smaller OpenAI models show some evidence that additional reasoning can compensate for weaker baseline capability, especially for `gpt-5.4-mini`. The value of reasoning therefore depends on the underlying model and provider implementation.

Third, the timing pattern is common across models. Figure 10 shows that higher reasoning levels generally increase average think time. The performance response, however, is not the same for every model: `gpt-5.4-mini` improves under medium reasoning despite the longer response time. The cells with very long response times nevertheless illustrate the downside risk of additional deliberation. Sonnet medium, for example, takes more than 16 seconds per turn on average and earns much lower NLV than Sonnet off or low; Opus medium also earns substantially lower NLV than Opus off or low. This pattern is consistent with the main interpretation: test-time reasoning can be useful, but its value depends on whether the additional cognition is worth the delay it creates in a real-time market.[35]

5 Conclusion

This paper has examined LLM trading agents that generate decisions through reasoning at the moment of trading. Reasoning produces a dividend in tender selection and the quality of trades the agent completes, but it also consumes market time, creating a tax through failed acceptance attempts and residual inventory. The balance depends on reasoning intensity, market speed, and the architecture of the surrounding system. Where cognition is placed matters: calculations and execution can be delegated to compiled routines, leaving the agent responsible for tender judgment. Comparisons with a neural-network trader trained for the task also reveal greater resilience of the LLM agent's profits to reduced liquidity. These results suggest that decision-time reasoning can add value beyond task-specific training, provided the market leaves sufficient time to act. The economic content of this reasoning, its interaction with system design, and its value under changing market conditions remain

[35] Online Appendix Tables A11 and A12 provide regression versions of Table 2 separately for the OpenAI model scale comparison and the Anthropic family.

fruitful directions for future research.

References

Acemoglu, D., D. Kong, and A. Ozdaglar. 2026. AI, human cognition and knowledge collapse. Working Paper 34910, National Bureau of Economic Research. doi:10.3386/w34910.

Agrawal, A. K., J. S. Gans, and A. Goldfarb. 2019. Prediction, judgment, and complexity: A theory of decision-making and artificial intelligence. In *The Economics of Artificial Intelligence: An Agenda*. University of Chicago Press.

———. 2025. The economics of bicycles for the mind. Working Paper 34034, National Bureau of Economic Research. doi:10.3386/w34034.

Almgren, R., and N. Chriss. 2000. Optimal execution of portfolio transactions. *Journal of Risk* 3. doi:https://doi.org/10.21314/JOR.2001.041.

Banerjee, S., and M. Szydlowski. 2025. Trading against algorithms: Price dynamics and risk-sharing in a market with q-learners. doi:10.2139/ssrn.5380152. SSRN working paper.

Bertsimas, D., and A. W. Lo. 1998. Optimal control of execution costs. *Journal of Financial Markets* 1:1–50. ISSN 1386-4181. doi:https://doi.org/10.1016/S1386-4181(97)00012-8.

Bhagwat, V., J. A. Cookson, C. Dim, and M. Niessner. 2026. The market's mirror: Revealing investor disagreement with LLMs. Tech. rep. doi:10.2139/ssrn.5375473. SSRN working paper.

Bini, P., L. W. Cong, X. Huang, and L. J. Jin. 2026. Behavioral economics of AI: LLM biases and corrections. Working Paper 34745, National Bureau of Economic Research. doi:10.3386/w34745.

Brandt, M. W., P. Santa-Clara, and R. Valkanov. 2009. Parametric portfolio policies: Exploiting characteristics in the cross-section of equity returns. *Review of Financial Studies* 22:3411–47.

Budish, E., P. Cramton, and J. Shim. 2015. The high-frequency trading arms race: Frequent batch auctions as a market design response. *Quarterly Journal of Economics* 130:1547–621. doi:10.1093/qje/qjv027.

Chen, H., A. Didisheim, L. Somoza, H. Tian, and M. Pourmohammadi. 2025. A financial brain scan of the LLM. Tech. rep. doi:10.2139/ssrn.5412277. SSRN working paper.

Cong, L. W., K. Tang, J. Wang, and Y. Zhang. 2021a. AlphaPortfolio: Direct construction through deep reinforcement learning and interpretable AI. doi:10.2139/ssrn.3554486. SSRN working paper.

———. 2021b. Deep sequence modeling: Development and applications in asset pricing. *The Journal of Financial Data Science* 3:28–42. doi:10.3905/jfds.2020.1.053.

Dell'Acqua, F., E. I. McFowland, E. Mollick, H. Lifshitz-Assaf, K. C. Kellogg, S. Rajendran, L. Krayer, F. Candelon, and K. R. Lakhani. 2026. Navigating the jagged technological frontier: Field experimental evidence of the effects of artificial intelligence on knowledge worker productivity and quality. *Organization Science* 37:403–23. doi:10.1287/orsc.2025.21838.

Dou, W. W., I. Goldstein, and Y. Ji. 2025a. AI-powered trading, algorithmic collusion, and price efficiency. Tech. Rep. 34054, National Bureau of Economic Research. doi:10.3386/w34054.

———. 2025b. Financial market fragility in the era of AI planning. doi:10.2139/ssrn.5763222. SSRN working paper.

Dou, W. W., I. Goldstein, Z. Li, and L. Yang. 2025. Human edge, machine limits: AI-human competition in financial markets. doi:10.2139/ssrn.5866104. SSRN working paper.

Hasbrouck, J. 2007. *Empirical market microstructure: The institutions, economics, and econometrics of securities trading*. New York: Oxford University Press. ISBN 9780198041306.

Hendershott, T., C. M. Jones, and A. J. Menkveld. 2011. Does algorithmic trading improve liquidity? *The Journal of Finance* 66:1–33. doi:https://doi.org/10.1111/j.1540-6261.2010.01624.x.

Hendershott, T., D. Livdan, and N. Schuerhoff. 2021. All-to-all liquidity in corporate bonds. Available at SSRN. doi:10.2139/ssrn.3895270. Swiss Finance Institute Research Paper No. 21-43; SSRN working paper, revised May 2025.

Hwang, J.-J., R. Xu, H. Lin, W.-C. Hung, J. Ji, K. Choi, D. Huang, T. He, P. Covington, B. Sapp, Y. Zhou, J. Guo, D. Anguelov, and M. Tan. 2025. EMMA: End-to-end multimodal model for autonomous driving. *Transactions on Machine Learning Research* ArXiv:2410.23262.

Kang, H., Q. Zhang, H. Cai, W. Xu, T. Krishna, Y. Du, and T. Weissman. 2025. Win fast or lose slow: Balancing speed and accuracy in latency-sensitive decisions of LLMs.

Korinek, A. 2025. AI agents for economic research. Working Paper 34202, National Bureau of Economic Research. doi:10.3386/w34202.

Kozak, S., S. Nagel, and S. Santosh. 2020. Shrinking the cross-section. *Journal of Financial Economics* 135:271–92. doi:10.1016/j.jfineco.2019.06.008.

Li, X., Y. Grandvalet, and F. Davoine. 2018. Explicit inductive bias for transfer learning with convolutional networks. In *Proceedings of the 35th International Conference on Machine Learning*, vol. 80, 2825–34. PMLR.

Lopez-Lira, A. 2025. Can large language models trade? Testing financial theories with LLM agents in market simulations. doi:10.2139/ssrn.5217340. SSRN working paper.

Lopez-Lira, A., and Y. Tang. 2025. Can ChatGPT forecast stock price movements? Return predictability and large language models. *Journal of Financial Economics* Forthcoming.

Madaan, A., N. Tandon, P. Gupta, S. Hallinan, L. Gao, S. Wiegreffe, U. Alon, N. Dziri, S. Prabhumoye, Y. Yang, S. Gupta, B. P. Majumder, K. Hermann, S. Welleck, A. Yazdanbakhsh, and P. Clark. 2023. Self-Refine: Iterative refinement with self-feedback. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, eds., *Advances in Neural Information Processing Systems*, vol. 36, 46534–94. Curran Associates, Inc. doi:10.52202/075280-2019.

Mak, K., and T. H. McCurdy. 2019. Simulation-based learning using the RIT market simulator and RIT decision cases. *Journal of Behavioral and Experimental Finance* 23:12–22. ISSN 2214-6350. doi:https://doi.org/10.1016/j.jbef.2019.05.003.

Obizhaeva, A. A., and J. Wang. 2013. Optimal trading strategy and supply/demand dynamics. *Journal of Financial Markets* 16:1–32. ISSN 1386-4181. doi:https://doi.org/10.1016/j.finmar.2012.09.001.

O'Hara, M., and X. A. Zhou. 2021. The electronic evolution of corporate bond dealers. *Journal of Financial Economics* 140:368–90. doi:10.1016/j.jfineco.2021.01.001.

Perold, A. F. 1988. The implementation shortfall: Paper versus reality. *Journal of Portfolio Management* 14:4–9.

Schick, T., J. Dwivedi-Yu, R. Dessì, R. Raileanu, M. Lomeli, E. Hambro, L. Zettlemoyer, N. Cancedda, and T. Scialom. 2023. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems*, vol. 36.

Shinn, N., F. Cassano, A. Gopinath, K. Narasimhan, and S. Yao. 2023. Reflexion: Language agents with verbal reinforcement learning. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, eds., *Advances in Neural Information Processing Systems*, vol. 36, 8634–52. Curran Associates, Inc. doi:10.52202/075280-0377.

Snell, C., J. Lee, K. Xu, and A. Kumar. 2025. Scaling LLM test-time compute optimally can be more effective than scaling model parameters. In *Proceedings of the 13th International Conference on Learning Representations (ICLR)*.

Wei, J., X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. H. Chi, Q. V. Le, and D. Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, vol. 35, 24824–37.

Yao, S., J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao. 2023. ReAct: Synergizing reasoning and acting in language models. In *Proceedings of the 11th International Conference on Learning Representations (ICLR)*.

Figure 1: Decomposition of mean session NLV into Trading P&L and Fines across the reasoning × simulation-speed grid (gpt-5.4, $N = 450$; 50 sessions per cell). In each panel the upward green bar is mean Trading P&L, the downward red bar is mean Fines (plotted as −Fines), and the black tick marks mean NLV = Trading P&L − Fines. All panels share a common y-axis. NLV and Fines are annotated; Trading P&L is read off as the height of the green bar. The cell-level numerical values are reported in Table 1.

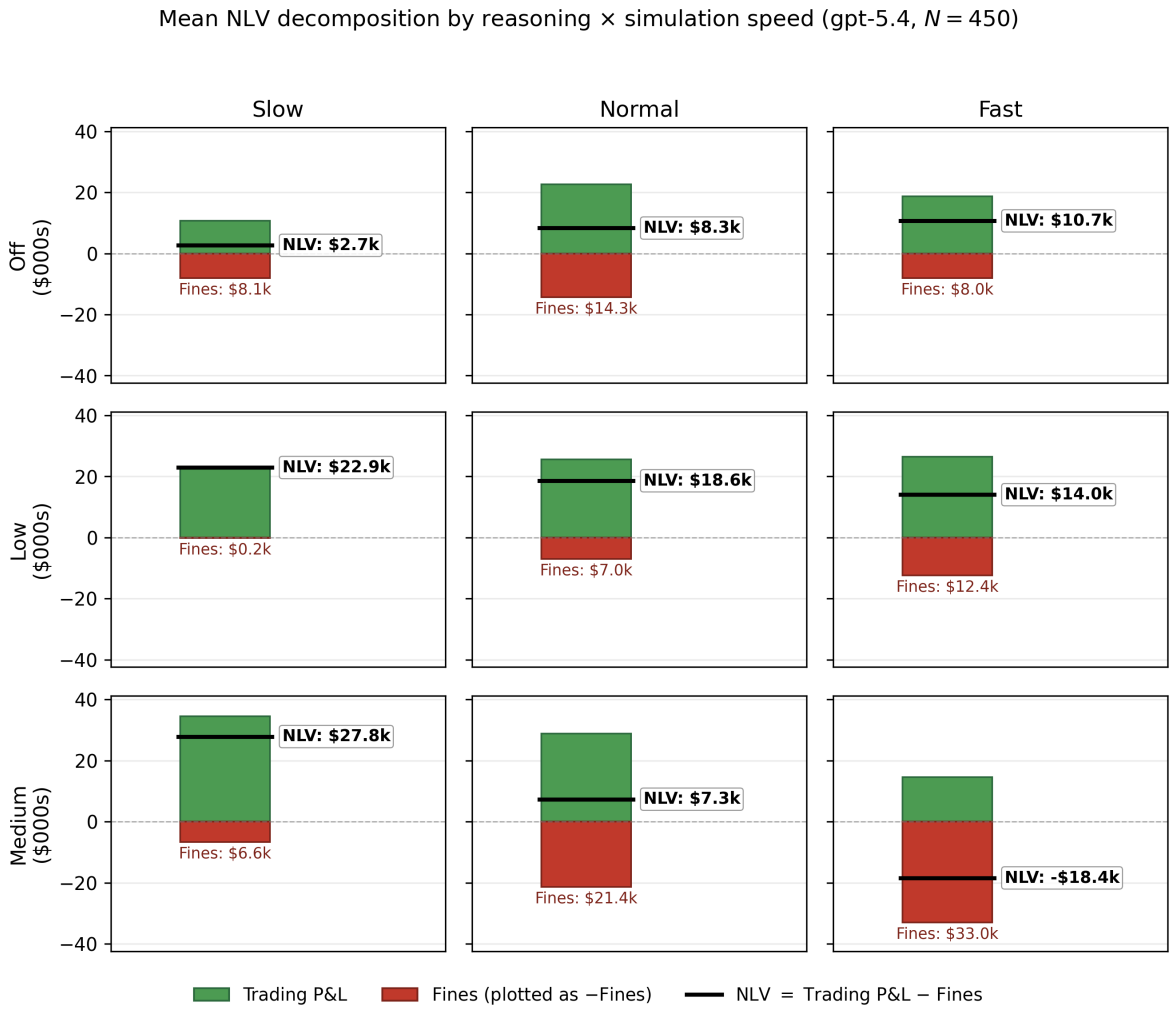


Figure 2: LT3 tender-execution task and agent architecture. Panel A summarizes the task. Each session lasts 300 ticks over two underlyings (CRZY, TAME), during which institutional tenders arrive stochastically. For each arriving tender the agent decides whether to accept; an accepted tender opens an inventory position that the agent then unwinds in the central limit order book. The session terminates at tick 300, at which point realized P&L is computed. Panel B sketches the agent architecture as an evolving conversation between the RIT simulator, a thin adapter layer (turn API, tool calls, response routing), and the frontier model (reasoning setting: off, low, or medium).

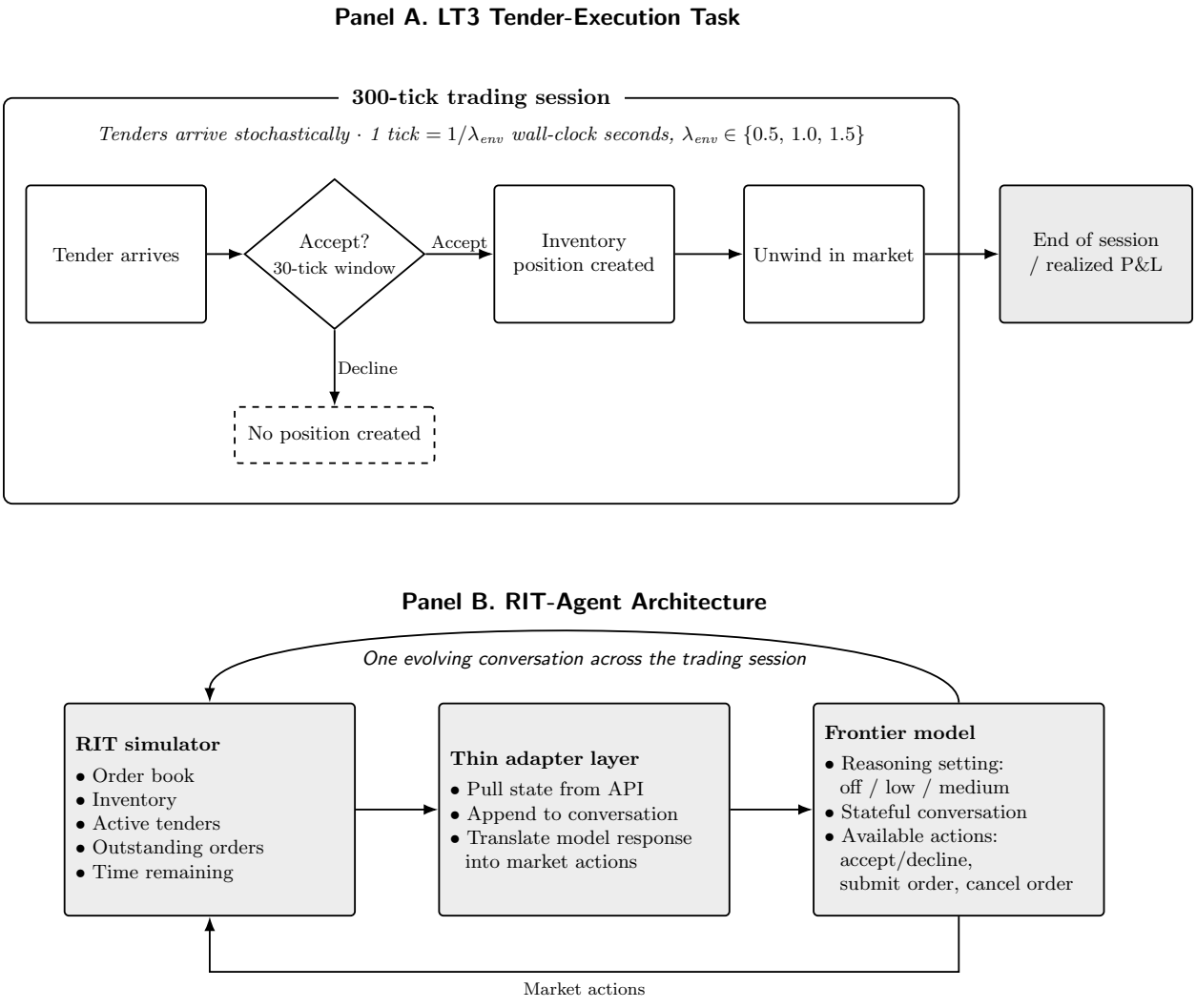


Figure 3: Session NLV vs. average think time per agent turn, gpt-5.4 in the normal-speed market ($N = 150$). Each point is one session, colored by reasoning level. Solid lines are within-reasoning OLS fits from $\text{NLV}_i = \alpha + \gamma_k \text{Reasoning}_i + \beta \ln(\text{ThinkTime}_i) + \varepsilon_i$. Filled markers are cell means. Horizontal axis is log-scaled; ln(ThinkTime) is a post-treatment variable.

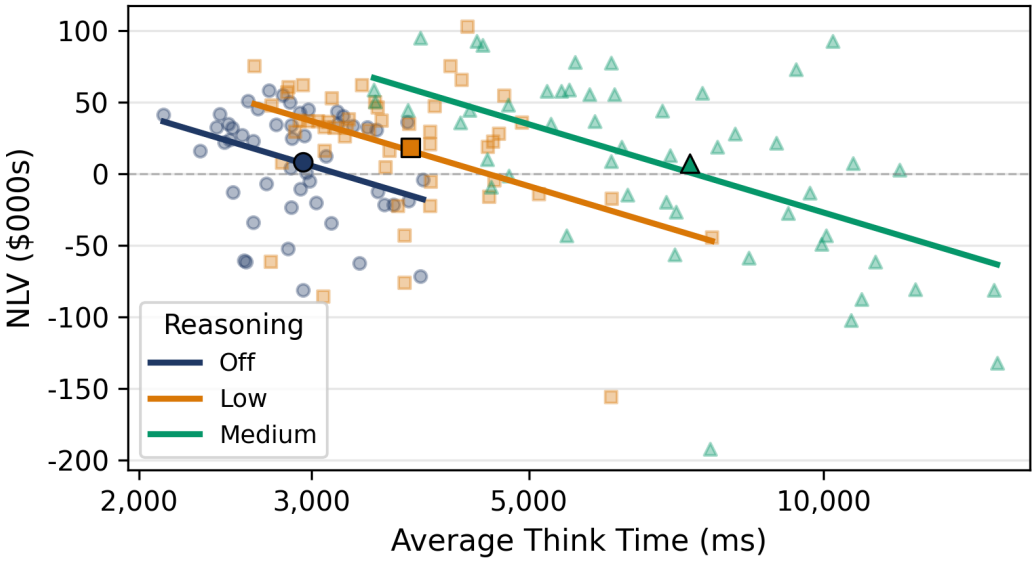


Figure 4: Per-cell distribution of session NLV ($000s) for the gpt-5.4 speed-pool experiment: 9 cells (rows = reasoning level, columns = simulation speed), 50 sessions per cell. Layout matches Figure 1. Boxes show the median (color line) and IQR (q_{25}–q_{75}); whiskers extend to the 10th and 90th percentiles. Diamonds mark cell means. The annotation in each cell reports the IQR ($q_{75} - q_{25}$) and the 80% spread ($P_{90} - P_{10}$) in $000s. The dashed line at zero NLV separates winning from losing sessions. All panels share a common y-axis.

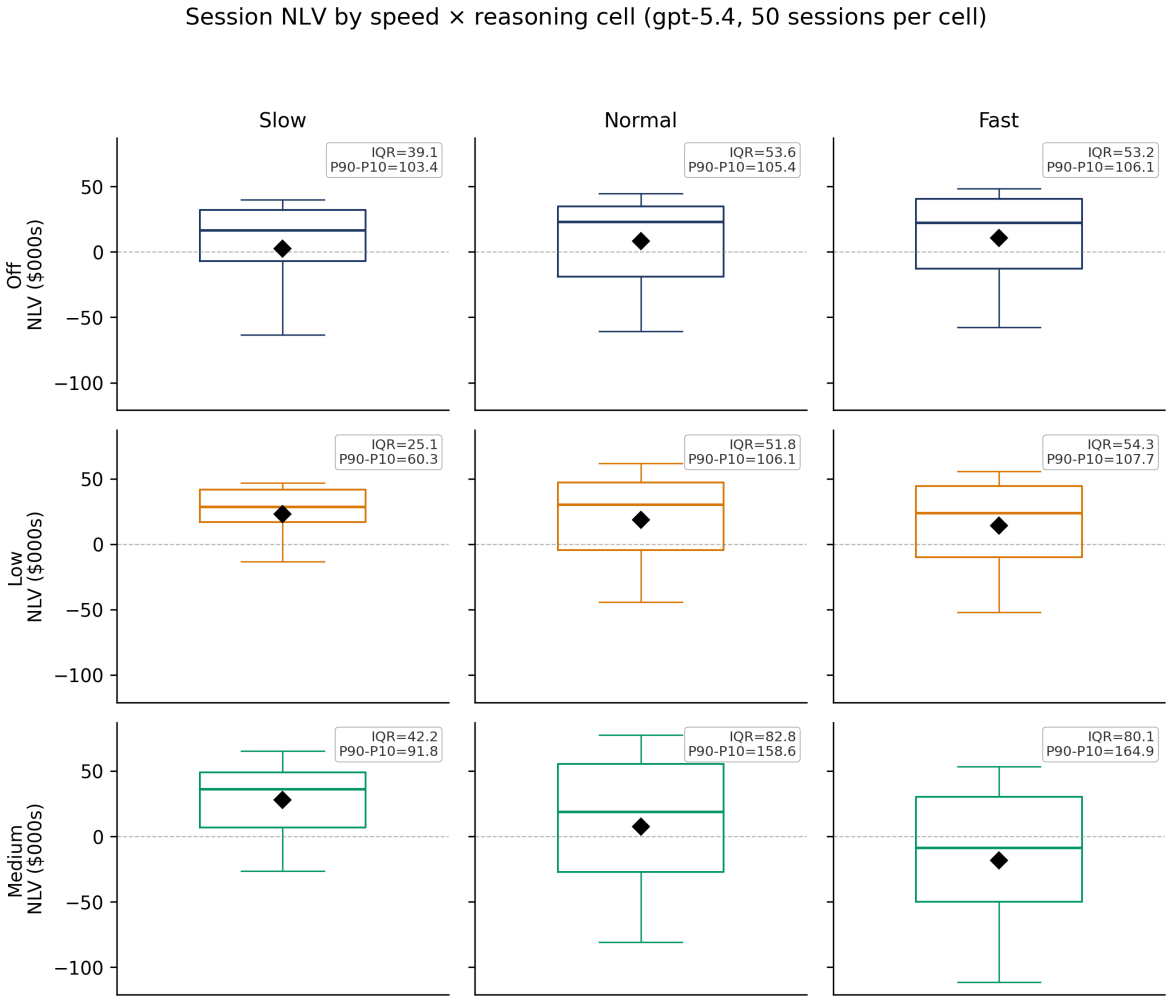


Figure 5: Predicted probability of accept intent as a function of the current observed book edge, by reasoning level (gpt-5.4; $N = 3{,}060$ pre-decision live turns across normal / fast / slow). Curves are pooled predictions from Table 6 Panel B column (2): a linear probability model with reasoning × book edge interactions and speed fixed effects, averaged over the three speeds. The dashed vertical line marks a zero-book-edge tender (break-even before fees); the dotted horizontal line marks $P = 0.5$. Tick marks at the bottom show the distribution of observed book edges (400-point subsample per level).

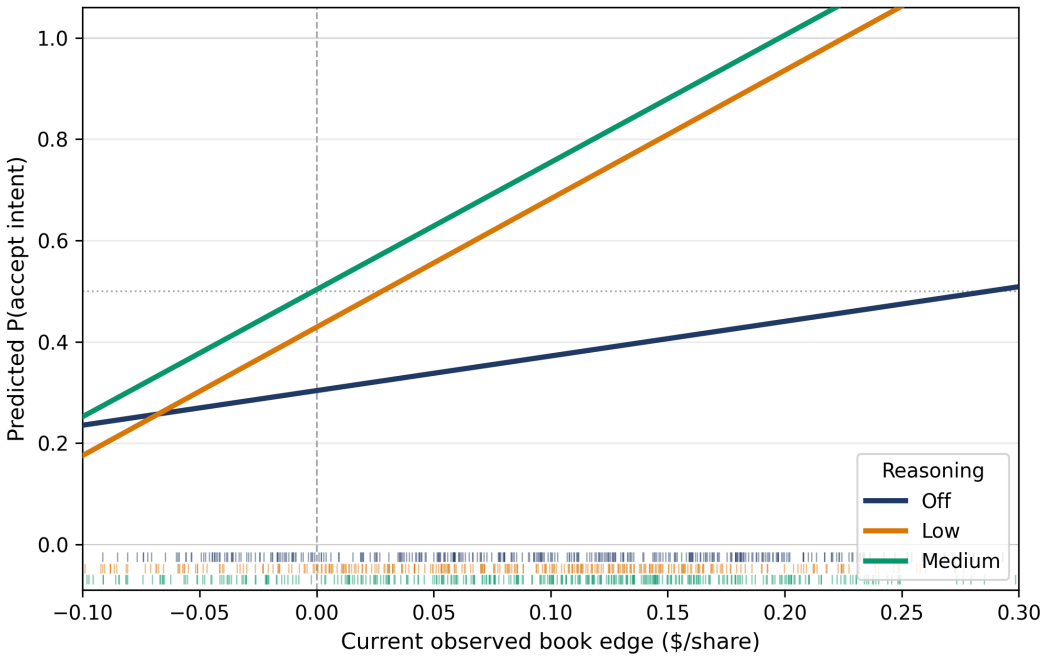


Figure 6: Tender lifecycle pooled across the three simulation speeds (gpt-5.4, 150 sessions per reasoning level). Each horizontal bar represents all tenders observed by the agent at that reasoning level (750/750/717 observed for off / low / medium). The medium count is 33 below the expected 750 because at fast simulation speed, Medium-reasoning think time was long enough that some tenders occurred and expired between the agent's polls. Segments — left to right — are the possible fates of a tender: the agent unwinds the accepted position cleanly (green); the accept fills but the agent does not complete the unwind (amber); the agent's accept intent fails to convert because the tender expires during the agent's own think time (red) or because the implied post-fill position would breach the 100k-share net limit (maroon); or the agent never issues an accept intent at all (grey).

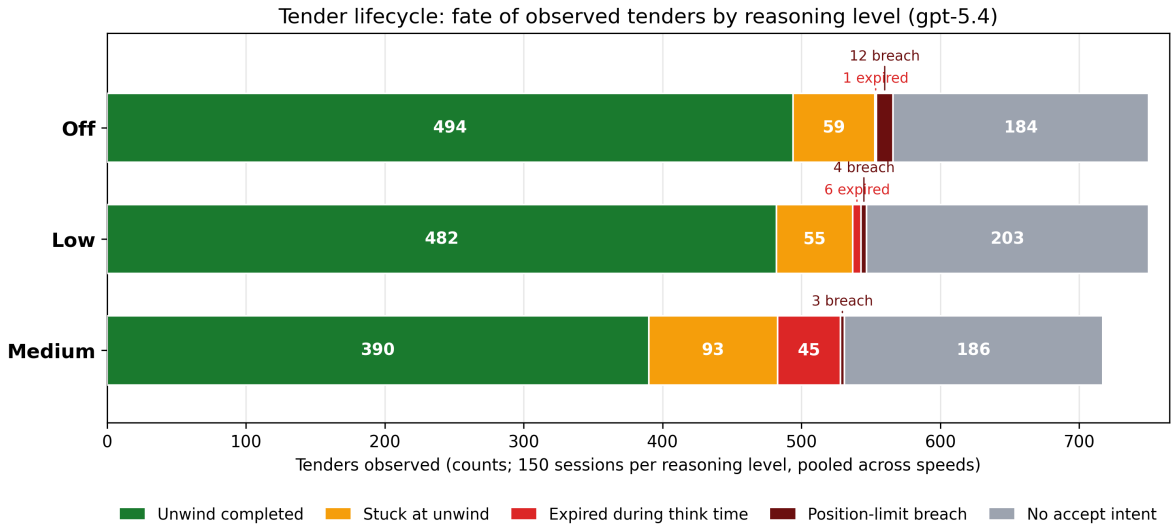


Figure 7: Coefficient plot for Panel B of Table 8: OLS coefficients from regressing each of five classifier dimensions on reasoning indicators (reference: off). Circle = Low, square = Medium; bars are 95% CIs, with bars that exceed the plotted range terminating in caret arrows. Sample: the 150 gpt-5.4 normal-speed sessions.

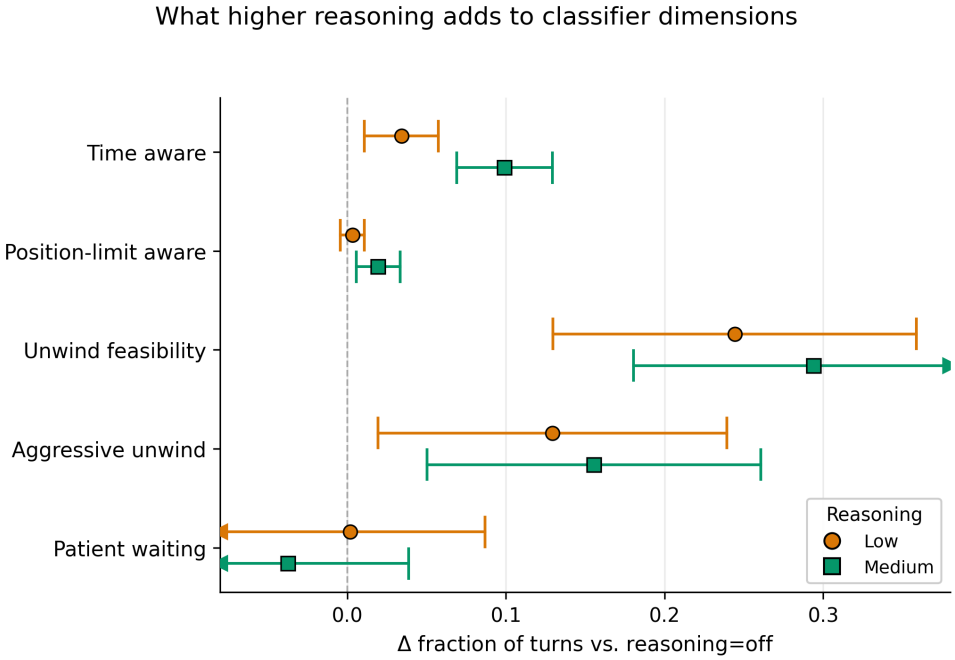


Figure 8: Sample reasoning traces from gpt-5.4 in the normal-speed market, one example per (reasoning level × situation) cell. Columns: (1) evaluating an incoming tender; (2) unwinding a just-accepted position into the book; (3) choosing to *wait* rather than cross the spread after prices move against the position. At reasoning = off the visible trace is the model's free-text `note` attached to its action; at reasoning = low / medium it is the OpenAI reasoning summary. Ellipses ("...") mark light trimming; section headers like **"Evaluating . . ."** are emitted by the model, not added in post.

	Assessing a tender	**Unwinding a position**	**Waiting for a bounce (emergent)**
off	tender: buy 66k TAME @ 25.45 *"TAME tender is attractive: buy 66,000 at 25.45, while visible bids start 25.67 / 25.67 / 25.66 and remain above 25.45 for substantial depth. Even after $0.02/share unwind fees, expected edge is positive. Accepting now; will unwind on subsequent turn to avoid front-running fines."* **Action:** ACCEPT tender	position: −63,500 CRZY *"Hold. The accepted tender is profitable, and my resting CRZY buy orders at 10.08 are now top-of-book. No need to cross the spread yet; wait for fills and reassess next update."* **Action:** no orders, no cancels	−74,000 TAME, mark P&L −$31,820 *"Hold. The TAME tender is currently deeply unprofitable to unwind, so I should not chase the market higher. Keep the 25.80 bids resting and wait for mean reversion / liquidity to come back."* **Action:** no orders, no cancels
low	tender: sell 70k CRZY @ 10.46 ***"Calculating budget constraints.*** 605,310/59,100 ≈ 10.24. *I still need 10,900 at avg above 10.5 but under 10.44. Total budget for 70,000 at avg 10.44 = 730,800. After covering 605,310, remaining budget 125,490 — so up to 11.51 on the last 10,900."* **Action:** ACCEPT tender	position: −29,400 CRZY ***"Evaluating trade strategy.*** *Cover remaining 29,400 with asks totaling 26,300 at 10.45; fill 25,000 there, fully profitable. . . . Risk if market rises — profits already dropped 24.5k → 16.5k. Covering 25k at 10.45 is an acceptable small loss to secure most of the position."* **Action:** BUY 25,000 CRZY ML @ 10.45	−69,000 TAME, ask > tender price ***"Analyzing market orders.*** *Market shifted against us; short at 24.82, best ask now 24.93 → loss if we cover. Waiting for a possible mean reversion seems reasonable, and our 24.81 bids might get filled if the price retreats. . . . Hold our resting bids; see how the market moves."* **Action:** no orders, no cancels
med.	tender: sell 66k TAME @ 25.10 ***"Evaluating share tender profitability.*** *. . . want to cover at avg price less than $25.08 (after $0.02 fee).* ***Calculating cumulative share costs:*** 10,000@24.87 = 248,700*;* +600@24.87 = 263,622*;* +9,400@24.91 → 497,776 *for 20k . . .* ***Calculating potential profits****: profit of about $8,201 . . . acceptance of the tender is the next move!"* **Action:** ACCEPT tender	position: −68,000 TAME ***"Deciding trading strategy.*** *. . . marketable limit around 25.14–25.15 . . .* 7 × 10*k at 25.14 might fill up to that price.* ***Calculating profitability****: full 68k through ≈ 25.22, average ≈ 25.05. Good profit cushion → buy aggressively, mindful of liquidity moving.* ***Planning simultaneous orders****: sending 7 orders at once . . . "* **Action:** 7× BUY 10,000 TAME ML	−8,900 CRZY, resting bid @ 10.20 ***"Evaluating bid strategy.*** *. . . best bid is also 10.20, best ask is 10.23. Crossing at 10.23 is a loss after fees. Better to wait at 10.20 . . . ensures a small profit if filled.* ***Deciding on bids****: since it's the top bid, there's a chance it could fill . . . I don't need to make any tool calls right now; I'll just wait and see if it fills."* **Action:** no orders, no cancels

Figure 9: Four architectures applied to the LT3 case, 50 sessions per cell. Agent and Agent + DS use Low reasoning; Agent + DS + Algo Exec uses Medium reasoning; the Algorithm row uses the \$0.05 tender-acceptance threshold. **Panel A**: per-session NLV distributions (box: 25th / 50th / 75th percentiles; whiskers: 1.5×IQR; black diamond: mean; jittered dots: individual sessions). **Panel B**: accounting decomposition of mean NLV into mean Trading P&L (green bar above zero) and mean Fines (red bar below zero); the black tick marks mean NLV = Trading P&L − Fines. Numerical values and pooled regressions are in Table 9.

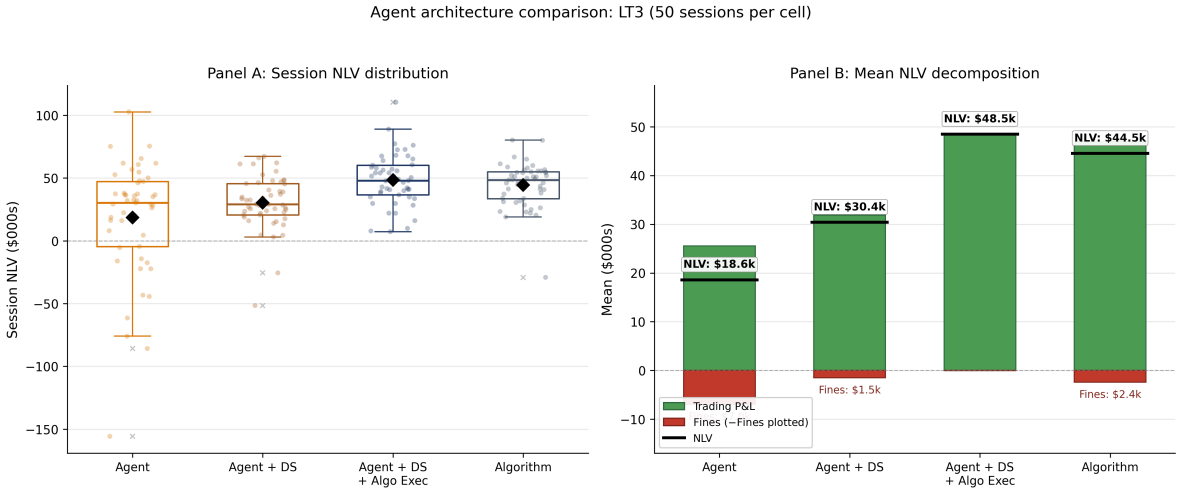


Figure 10: Connected-means view of model and reasoning (50 sessions per reasoning cell for each model; all at normal simulation speed). **Panel A**: mean session NLV ($000s); **Panel B**: mean think time per agent turn (ms, log scale). Within each model's x-slot, three points at small horizontal offsets show mean off, low, and medium in that order, connected by a thin grey line that traces each model's reasoning trajectory. A vertical separator divides the OpenAI family (gpt-5.4, mini, nano) from the Anthropic family (Opus 4.6, Sonnet 4.6).

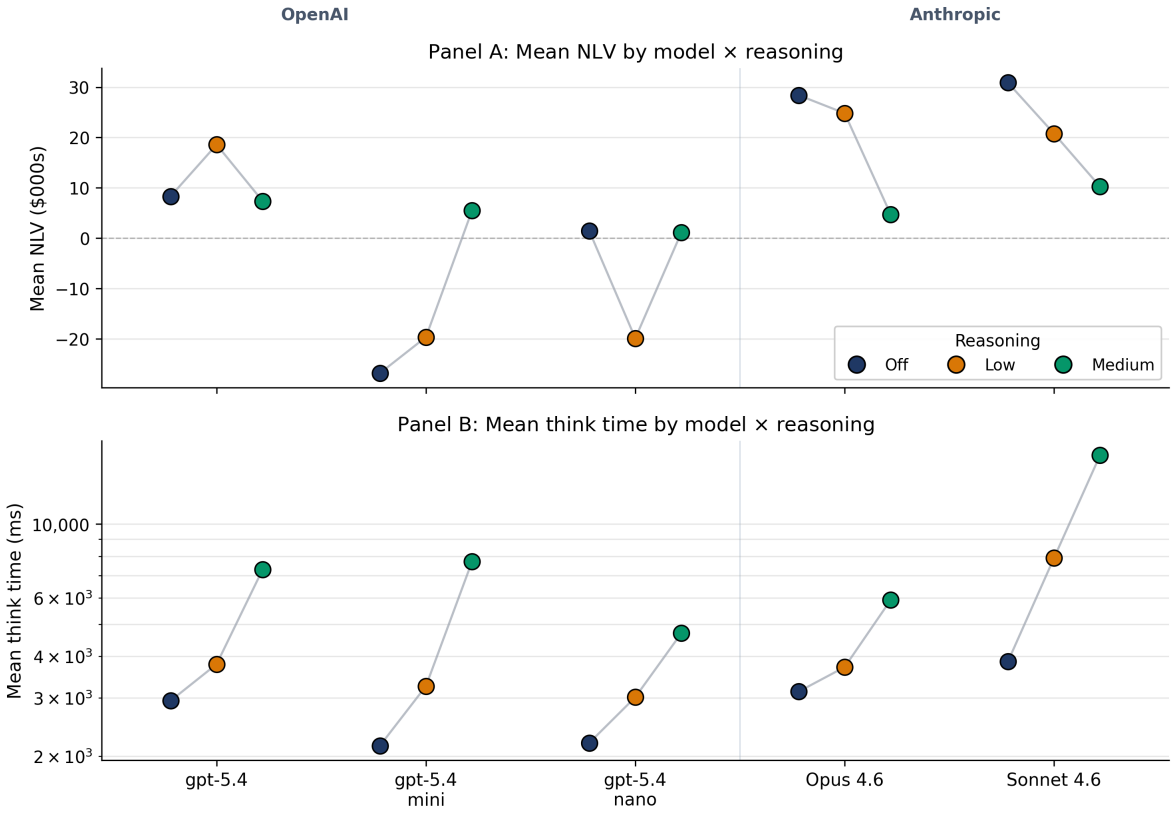


Table 1: Descriptive Treatment Means. Cell-level means across 50 gpt-5.4 sessions per cell on the reasoning × simulation-speed grid ($N = 450$). Dollar amounts in thousands ($000s). NLV is net liquidation value (Trading P&L minus Fines). Sharpe is mean NLV divided by std NLV within cell. % Profitable is the fraction of sessions with positive NLV. Think Time is the mean milliseconds per LLM turn. Tenders Accepted is the mean count of accepted tenders per session.

Cell	Mean NLV	Std NLV	Sharpe	% Profitable	Mean Trading P&L	Mean Fines	Mean Think Time (ms)	Tenders Accepted
Normal / Off	8.3	37.0	0.22	62%	22.6	14.3	2,939	3.7
Normal / Low	18.6	46.3	0.40	74%	25.5	7.0	3,789	3.7
Normal / Medium	7.3	62.8	0.12	62%	28.8	21.4	7,303	3.3
Fast / Off	10.7	40.9	0.26	72%	18.7	8.0	2,652	3.9
Fast / Low	14.0	41.2	0.34	72%	26.5	12.4	3,843	3.4
Fast / Medium	−18.4	69.0	-0.27	46%	14.5	33.0	8,372	2.5
Slow / Off	2.7	46.1	0.06	70%	10.7	8.1	3,231	3.4
Slow / Low	22.9	27.6	0.83	88%	23.1	0.2	3,728	3.6
Slow / Medium	27.8	34.9	0.80	78%	34.5	6.6	5,683	3.8
All gpt-5.4	10.4	48.1	0.22	69%	22.8	12.3	4,615	3.5

Table 2: Reasoning, Response Time, and Performance (Normal speed). OLS regressions on 150 gpt-5.4 sessions in the normal-speed market (50 per reasoning level). Dependent variables are NLV, Trading P&L, and Fines (all in $000s). Columns (1), (3), and (5) regress each DV on reasoning indicators alone (reference: reasoning off). Columns (2), (4), and (6) add log average think time per agent turn (mean-centered within the gpt-5.4 subsample). NLV equals Trading P&L minus Fines. Dollar amounts in thousands ($000s). HC3 robust standard errors in parentheses. ***, **, * denote significance at the 1%, 5%, and 10% levels. $N = 150$ (50 sessions per reasoning level). Reference: reasoning off.

	(1) NLV	(2) NLV	(3) TP&L	(4) TP&L	(5) Fines	(6) Fines
Low	10.27	31.27***	2.94	18.38***	−7.33*	−12.88***
	(8.47)	(8.46)	(6.60)	(6.56)	(4.11)	(4.61)
Medium	−0.97	74.45***	6.18	61.66***	7.15	−12.79*
	(10.41)	(12.90)	(7.63)	(10.48)	(5.65)	(7.64)
ln(Think Time)		−88.91***		−65.40***		23.51***
		(14.21)		(11.82)		(7.32)
Constant	8.31	−23.83***	22.59***	−1.05	14.28***	22.78***
	(5.29)	(7.34)	(4.23)	(6.12)	(3.24)	(4.27)
N	150	150	150	150	150	150
R^2	0.0105	0.2415	0.0047	0.2328	0.0540	0.1153

Table 3: Reasoning, Market Speed, and Performance. Panel A reports pairwise speed contrasts on session NLV within each reasoning level: the joint F(speed) test, Normal−Fast, and Slow−Fast. Panel B is the analog along the reasoning axis: the joint F(reasoning) test, Medium−Low, and Off−Low. Fast and slow scale wall-clock time per tick (RIT `Speed` = 150 for fast, 100 for normal, 50 for slow); the agent poll interval scales in tandem so the poll-to-tick ratio matches the normal-speed baseline. All speed cells use gpt-5.4; 50 sessions per reasoning × speed cell. The first data column reports the joint Wald F on the two contrast indicators within each conditioning subset (df = 2, 147); the p-value is in parentheses below. The remaining columns show pairwise contrasts: each interior cell stacks the mean NLV difference (in $000s, signed, two decimals; stars from the HC3 robust t-statistic), the HC3 robust standard error in parentheses, and the Sharpe-ratio difference [ΔS] for the indicated comparison. Mean differences and standard errors come from one OLS regression of NLV on the contrast indicators within each conditioning subset, with the reference cell as fast (Panel A) or low (Panel B). The *Pooled with FE* row in each panel reports a pooled OLS over all $N = 450$ sessions with the orthogonal axis as fixed effects; the F-statistic tests the joint zero of the two contrast indicators (df = 2, 446). No Sharpe-ratio difference is reported for the pooled rows. ***, **, * denote significance at the 1%, 5%, and 10% levels. $N = 150$ per conditional row; $N = 450$ for each pooled row.

Panel A: Speed contrasts on NLV by reasoning level

	F(speed)	Normal − Fast	Slow − Fast
Off	0.43	−2.35	−8.00
	(0.652)	(7.88)	(8.80)
		[$\Delta S = -0.04$]	[$\Delta S = -0.20$]
Low	0.81	4.54	8.90
	(0.446)	(8.86)	(7.09)
		[$\Delta S = 0.06$]	[$\Delta S = 0.49$]
Medium	9.38***	25.78*	46.28***
	(<0.001)	(13.33)	(11.05)
		[$\Delta S = 0.38$]	[$\Delta S = 1.07$]
Pooled w/ Reasoning FE	4.28**	9.32	15.72***
	(0.014)	(5.97)	(5.39)

Panel B: Reasoning contrasts on NLV by speed

	F(reasoning)	Medium − Low	Off − Low
Fast	4.22**	−32.46***	−3.38
	(0.017)	(11.49)	(8.29)
		[$\Delta S = -0.61$]	[$\Delta S = -0.08$]
Normal	0.86	−11.23	−10.27
	(0.425)	(11.14)	(8.47)
		[$\Delta S = -0.28$]	[$\Delta S = -0.18$]
Slow	4.94***	4.91	−20.28***
	(0.008)	(6.35)	(7.67)
		[$\Delta S = -0.03$]	[$\Delta S = -0.77$]
Pooled w/ Speed FE	3.89**	−12.93**	−11.31**
	(0.021)	(5.79)	(4.74)

Table 4: Within-Reasoning Quantile Regressions on NLV. For each reasoning level (Off, Low, Medium), three quantile regressions of session NLV on speed indicators with fast as the reference cell, plus the q75−q25 interquartile-range coefficient. A fourth pooled sub-panel fits all $N = 450$ sessions with additive reasoning fixed effects. Companion Table 5 reports the analog along the speed axis. Fast and slow scale wall-clock time per tick (RIT `Speed` = 150 for fast, 100 for normal, 50 for slow); the agent poll interval scales in tandem so the poll-to-tick ratio matches the normal-speed baseline. All speed cells use gpt-5.4; 50 sessions per reasoning × speed cell. Quantile regressions of session NLV (in $000s) at $\tau \in \{0.25, 0.50, 0.75\}$ with robust asymptotic SEs in parentheses. The *IQR* column reports the interquartile-range coefficient (q75 − q25) with bootstrapped SEs. ***, **, * denote significance at the 1%, 5%, and 10% levels. The first three sub-panels each fit $N = 150$ gpt-5.4 sessions (50 per cell). The final *Pooled* sub-panel fits all $N = 450$ sessions with additive reasoning fixed effects (reference: low), so each speed coefficient is the average speed contrast holding reasoning composition fixed; the constant is the predicted NLV at fast speed and low reasoning.

	q25	q50	q75	IQR
Reasoning: Off				
Normal vs Fast	−6.06	0.91	−5.67	0.39
	(14.98)	(8.67)	(6.91)	(14.33)
Slow vs Fast	5.88	−5.62	−8.23	−14.11
	(14.98)	(8.67)	(6.91)	(17.25)
Constant (Fast)	−12.74	22.06***	40.48***	53.22***
	(10.59)	(6.13)	(4.89)	(12.03)
N	150	150	150	150
Reasoning: Low				
Normal vs Fast	5.32	6.55	2.79	−2.53
	(11.40)	(6.91)	(6.46)	(19.26)
Slow vs Fast	26.53**	4.62	−2.71	−29.24**
	(11.40)	(6.91)	(6.46)	(14.58)
Constant (Fast)	−9.82	23.66***	44.49***	54.31***
	(8.06)	(4.89)	(4.57)	(13.45)
N	150	150	150	150
Reasoning: Medium				
Normal vs Fast	22.65	27.66**	25.38**	2.73
	(16.05)	(13.20)	(11.32)	(20.64)
Slow vs Fast	56.72***	44.85***	18.79*	−37.93**
	(16.05)	(13.20)	(11.32)	(16.76)
Constant (Fast)	−50.01***	−8.96	30.11***	80.12***
	(11.35)	(9.33)	(8.01)	(12.32)
N	150	150	150	150
All Reasoning Levels (Pooled, additive reasoning FE)				
Normal vs Fast	6.02	9.78*	4.57	−1.46
	(7.72)	(5.00)	(3.94)	(10.03)
Slow vs Fast	27.72***	11.44**	−1.28	−29.00***
	(7.72)	(5.00)	(3.94)	(9.10)
Constant (Fast, Low)	−10.34	20.43***	43.02***	53.36***
	(6.98)	(4.57)	(3.61)	(9.32)
N	450	450	450	450

Table 5: Within-Speed Quantile Regressions on NLV. For each speed level (Slow, Normal, Fast), three quantile regressions of session NLV on reasoning indicators with low as the reference cell, plus the q75−q25 interquartile-range coefficient. A fourth pooled sub-panel fits all $N = 450$ sessions with additive speed fixed effects. Companion Table 4 reports the analog along the reasoning axis. Fast and slow scale wall-clock time per tick (RIT `Speed` = 150 for fast, 100 for normal, 50 for slow); the agent poll interval scales in tandem so the poll-to-tick ratio matches the normal-speed baseline. All speed cells use gpt-5.4; 50 sessions per reasoning × speed cell. Quantile regressions of session NLV (in $000s) at $\tau \in \{0.25, 0.50, 0.75\}$ with robust asymptotic SEs in parentheses. The *IQR* column reports the interquartile-range coefficient (q75 − q25) with bootstrapped SEs. ***, **, * denote significance at the 1%, 5%, and 10% levels. The first three sub-panels each fit $N = 150$ gpt-5.4 sessions (50 per cell). The final *Pooled* sub-panel fits all $N = 450$ sessions with additive speed fixed effects (reference: fast), so each reasoning coefficient is the average reasoning contrast holding speed composition fixed; the constant is the predicted NLV at low reasoning and fast speed.

	q25	q50	q75	IQR
Speed: Slow				
Off vs Low	−23.57**	−11.84*	−9.53*	14.04
	(10.42)	(6.23)	(5.52)	(13.47)
Medium vs Low	−10.00	7.61	7.12	17.12
	(10.42)	(6.23)	(5.52)	(12.43)
Constant (Low)	16.71**	28.28***	41.77***	25.07***
	(7.37)	(4.41)	(3.90)	(5.74)
N	150	150	150	150
Speed: Normal				
Off vs Low	−14.29	−7.23	−12.47	1.82
	(18.09)	(11.22)	(9.18)	(15.61)
Medium vs Low	−22.85	−11.51	8.21	31.06
	(18.09)	(11.22)	(9.18)	(20.80)
Constant (Low)	−4.51	30.21***	47.28***	51.78***
	(12.79)	(7.93)	(6.49)	(13.26)
N	150	150	150	150
Speed: Fast				
Off vs Low	−2.91	−1.60	−4.01	−1.09
	(17.38)	(11.02)	(9.89)	(18.34)
Medium vs Low	−40.19**	−32.62***	−14.38	25.81
	(17.38)	(11.02)	(9.89)	(18.56)
Constant (Low)	−9.82	23.66***	44.49***	54.31***
	(12.29)	(7.79)	(6.99)	(13.99)
N	150	150	150	150
All Speed Levels (Pooled, additive speed FE)				
Off vs Low	−14.48*	−7.23	−7.89**	6.59
	(7.72)	(5.00)	(3.94)	(7.85)
Medium vs Low	−23.21***	−9.42*	2.30	25.51***
	(7.69)	(5.00)	(3.94)	(9.82)
Constant (Low, Fast)	−10.34	20.43***	43.02***	53.36***
	(6.98)	(4.57)	(3.61)	(9.32)
N	450	450	450	450

Table 6: Tender Selection and Responsiveness. Panel A reports cell means on the per-turn live sample (one row per agent turn while the tender is alive, pre-decision). *Book Edge | Intent* and *Book Edge | No Intent* ($/share) are cell means of book edge restricted to accept and decline turns respectively. *Neg-Book-Edge Intent* is the share of accept turns with book edge < 0. The right-most column, *Intent Failed*, is the only per-tender statistic: share of accept intents whose server submission did not produce a valid fill (tender expired during the agent's think time, or implied post-fill position would have breached the net limit). Panel B: cols (1)–(2) regress accept intent on book edge with reasoning indicators on the per-turn sample, col (2) adding reasoning × book edge; col (3) regresses the Intent Failed indicator on reasoning across the full per-tender accept-intent sample. Of 71 intent failures, 52 expired in think time and 19 would have hit the position limit. All regressions include speed FE; cluster-robust SEs (by session) in parentheses. ***/**/* = 1/5/10%.

Panel A: Summary statistics (live tender turns)

Cell	Book Edge | Intent ($/share)	Book Edge | No Intent ($/share)	Neg-Book-Edge Intent	Intent Failed
Normal / Off	0.1120	0.0520	10%	3%
Normal / Low	0.1262	-0.0588	2%	1%
Normal / Medium	0.1290	-0.0201	1%	11%
Fast / Off	0.1129	0.0559	9%	3%
Fast / Low	0.1346	-0.0306	2%	3%
Fast / Medium	0.1367	-0.0530	2%	16%
Slow / Off	0.1196	0.0555	7%	1%
Slow / Low	0.1244	-0.0617	2%	2%
Slow / Medium	0.1244	-0.0849	2%	2%

Panel B: Selection and intent-failure regressions

	(1) Intent	(2) Intent	(3) Intent Failed
Low	0.2381***	0.1252***	−0.0027
	(0.0336)	(0.0315)	(0.0087)
Medium	0.3290***	0.1995***	0.0714***
	(0.0311)	(0.0291)	(0.0139)
Book Edge	1.6073***	0.6834***	
	(0.1607)	(0.1815)	
Low × Book Edge		1.8507***	
		(0.2175)	
Medium × Book Edge		1.8265***	
		(0.2101)	
Constant	0.2632***	0.3253***	0.0234**
	(0.0284)	(0.0291)	(0.0094)
Speed FE	Yes	Yes	Yes
Sample	Turn	Turn	Tender
N	3060	3060	1644
R^2	0.2723	0.3338	0.0389

Table 7: Tender Execution Quality. The sample for this table is the set of tenders for which the agent issued an accept action and that accept was filled. Panel A reports per-cell summary statistics on the post-fill unwind. *Unwind Completed* is the share of filled intents for which the agent returned to its pre-acceptance position before session end or before accepting another tender for the same ticker; the complementary share is *stuck*, meaning the position was not fully unwound before either cutoff. *Edge Captured* ($/share) is the mean realized edge per share, restricted to completed unwinds. *Book Edge at Intent* ($/share) is the mean book edge at the decision turn, restricted to the same completed-unwind tenders for direct comparison with Edge Captured. *IS* ($/share) is the per-share implementation shortfall on the filled-intent sample: $\frac{1}{Q}[-s(P_t - \pi_0)Q + s(\bar{p}_u - \pi_0)u + s(\pi_1 - \pi_0)r + 0.02u + 1.00r]$, where $s = +1$ for SELL tenders / -1 for BUY, P_t is the tender price, π_0 the bid-ask midpoint at accept, π_1 the bid-ask midpoint at settlement, $\bar{p}_u$ the unwind VWAP, u the shares unwound, and $r = Q - u$ the residual auto-liquidated at measurement cutoff (session end or the next accepted tender for the same ticker). Components are tender capture, execution cost, opportunity cost, $0.02/share trading fees, and $1/share auto-liquidation fines. Positive IS = drag on realized P&L vs the paper-portfolio benchmark. Panel B reports the corresponding cell-level regressions on reasoning indicators (reference: off) with speed fixed effects. Col (1) is a linear probability model on the full filled-intent sample. Col (2) regresses Edge Captured on reasoning controlling for book edge at the decision turn (completed unwinds). Col (3) regresses Book Edge at Intent on reasoning (completed unwinds). Col (4) regresses IS on reasoning controlling for book edge at the decision turn (filled intents); its sample is smaller than col (1) because the book-edge control is undefined for filled intents whose visible book at the decision turn could not span the full tender quantity. A negative coefficient implies lower per-share shortfall, or realized P&L closer to the paper-portfolio benchmark. Cluster-robust standard errors (by session) in parentheses. ***/**/* = 1/5/10%.

Panel A: Summary statistics (filled intents)

Cell	N	Unwind Completed	Edge Captured ($/share)	Book Edge at Intent ($/share)	IS ($/share)
Normal / Off	187	88%	0.1098	0.1124	-0.0236
Normal / Low	187	89%	0.1396	0.1307	-0.0366
Normal / Medium	166	78%	0.1582	0.1327	0.0025
Fast / Off	194	88%	0.0981	0.1151	-0.0221
Fast / Low	169	83%	0.1513	0.1336	-0.0279
Fast / Medium	125	68%	0.1430	0.1371	0.1576
Slow / Off	172	92%	0.0801	0.1247	-0.0105
Slow / Low	181	97%	0.1065	0.1256	-0.0871
Slow / Medium	192	91%	0.1582	0.1301	-0.0902

Panel B: Cell-level regressions

	(1) Unwind Completed	(2) Edge Captured	(3) Book Edge at Intent	(4) IS
Low	0.0003	0.0242**	0.0126**	−0.0137
	(0.0223)	(0.0119)	(0.0049)	(0.0211)
Medium	−0.0977***	0.0470**	0.0155***	0.0647**
	(0.0252)	(0.0183)	(0.0051)	(0.0296)
Book edge		0.8935***		−1.2054***
		(0.0916)		(0.1205)
Constant	0.8818***	0.0006	0.1159***	0.0947***
	(0.0219)	(0.0176)	(0.0051)	(0.0264)
Speed FE	Yes	Yes	Yes	Yes
Sample	Filled	Completed	Completed	Filled
N	1573	1231	1231	1414
R^2	0.0411	0.0981	0.0088	0.0716

Table 8: Reasoning Content Classification (Normal speed). Per-session reasoning content was labeled along eight binary dimensions by an LLM classifier from a different model family than the subject models; five dimensions are reported here. Panel A shows cell-level means across the three gpt-5.4 normal-speed reasoning cells (50 sessions per cell, $N = 150$). *Time-Aware* and *Pos-Limit Aware*: fraction of all turns in which reasoning explicitly invokes remaining time or the position limit. *Unwind Feas.*: fraction of tender-visible turns discussing post-accept unwind feasibility. *Aggressive Unwind* and *Patient Waiting*: fraction of nonzero-position turns reasoning with urgency to flatten or with deliberate patience, respectively. An eighth dimension (*Inaction Holding*, the fraction of nonzero-position turns with no orders, cancels, or tender decisions) is omitted because it correlates with *Patient Waiting* at the session level ($r = 0.85$). Panel B reports OLS regressions of each classifier dimension on reasoning indicators (reference: reasoning off), restricted to the 150 gpt-5.4 normal-speed sessions. HC3 robust standard errors in parentheses. ***, **, * denote significance at the 1%, 5%, and 10% levels.

Panel A: Cell means of classifier dimensions

Cell	Time Aware	Pos-Limit Aware	Unwind Feas.	Aggressive Unwind	Patient Waiting
Normal / Off	0.08	0.01	0.37	0.17	0.19
Normal / Low	0.11	0.01	0.61	0.30	0.19
Normal / Medium	0.18	0.03	0.66	0.33	0.15
All gpt-5.4 normal	0.12	0.01	0.55	0.27	0.18

Panel B: Reasoning level → classifier dimensions

	(1) Time Aware	(2) Pos-Limit Aware	(3) Unwind Feas.	(4) Aggressive Unwind	(5) Patient Waiting
Low	0.034***	0.003	0.244***	0.129**	0.002
	(0.012)	(0.004)	(0.058)	(0.056)	(0.043)
Medium	0.099***	0.020***	0.294***	0.156***	−0.037
	(0.015)	(0.007)	(0.058)	(0.054)	(0.039)
Constant	0.077***	0.007**	0.370***	0.172***	0.189***
	(0.008)	(0.003)	(0.051)	(0.046)	(0.032)
N	150	150	150	150	150
R^2	0.254	0.076	0.198	0.070	0.009

Table 9: Agent Architecture Comparison. Four architectures applied to the LT3 case, 50 sessions per cell. DS denotes *decision support*: pre-computed book-edge, inventory, limit, and flattening-cost summaries. Algo Exec denotes *algorithmic execution* using the fixed reduce-only unwind routine. Algorithm denotes the fully compiled benchmark with no LLM calls. Panel A reports per-cell descriptive means using the columns of Table 1: *Mean / Std NLV* are the cross-session mean and standard deviation of session NLV; *Sharpe* is mean / std; *% Profitable* is the share of sessions with positive NLV; *Mean Trading P&L* and *Mean Fines* are the components of NLV; *Mean Control Time* is the average wall-clock spent per call by the agent's controlling script, in milliseconds. For the Agent and Agent + DS rows, this equals the mean LLM-call latency since every call goes to the LLM. For the Agent + DS + Algo Exec and Algorithm rows, this includes the near-zero latency during idle turns and latency in turns where the script acts (fast accept/declines/unwinds from the algorithm or relatively slower think time from pending LLM decisions). The two Algorithm rows differ only in the tender-acceptance threshold ($0.05 and $0.10 per share); all other algorithm parameters and the case configuration are identical. *Tenders Accepted* is the mean count of accepted tenders per session. Panel B reports regressions on the pooled sample ($N = 200$) of the four cells: Agent / Low, Agent + DS / Low, Agent + DS + Algo Exec / Medium, and the Algorithm ($0.05 threshold). Three architecture indicators are included with Agent / Low as the omitted base, so Constant equals the Agent / Low cell mean and each indicator coefficient is the mean difference between that architecture and the Agent / Low cell. Columns: (1) NLV OLS, (2) NLV IQR ($q_{75} - q_{25}$), (3) Trading P&L OLS, (4) Fines OLS. Dollar amounts in thousands ($000s). HC3 robust standard errors for OLS in parentheses; bootstrap SEs for IQR. ***, **, * denote significance at the 1%, 5%, and 10% levels.

Panel A: Descriptive means

Cell	Mean NLV	Std NLV	Sharpe	% Profitable	Mean Trading P&L	Mean Fines	Mean Control Time (ms)	Tenders Accepted
Agent								
Off	8.3	37.0	0.22	62%	22.6	14.3	2,939	3.7
Low	18.6	46.3	0.40	74%	25.5	7.0	3,789	3.7
Medium	7.3	62.8	0.12	62%	28.8	21.4	7,303	3.3
Agent + DS								
Off	19.6	31.1	0.63	78%	30.6	11.1	2,860	4.1
Low	30.4	21.5	1.41	96%	31.9	1.5	2,917	4.1
Medium	26.9	42.9	0.63	84%	31.8	4.8	5,196	3.5
Agent + DS + Algo Exec								
Off	44.5	15.6	2.85	100%	45.6	1.1	168	4.4
Low	38.0	25.5	1.49	92%	38.2	0.2	673	4.5
Medium	48.5	20.4	2.38	100%	48.5	0.0	1,387	4.6
Algorithm								
$0.05 threshold	44.5	17.9	2.49	98%	46.9	2.4	25	3.9
$0.10 threshold	44.0	24.9	1.77	96%	48.6	4.6	37	3.4

Panel B: Regressions on architecture indicators

	(1) NLV (OLS)	(2) NLV (IQR)	(3) TP&L (OLS)	(4) Fines (OLS)
Agent + DS	11.82	−26.86*	6.35	−5.47**
	(7.29)	(14.19)	(5.89)	(2.60)
Agent + DS + Algo Exec	29.88***	−28.18**	22.96***	−6.91***
	(7.23)	(14.29)	(5.84)	(2.52)
Algorithm	25.93***	−30.46**	21.39***	−4.53
	(7.09)	(14.26)	(5.50)	(2.92)
Constant (Agent / Low)	18.58***	51.78***	25.53***	6.95***
	(6.61)	(13.56)	(5.06)	(2.52)
N	200	200	200	200
R^2	0.1463		0.1429	0.0588

Table 10: Network Size and NN Trader Performance. Mean session NLV in $000s; 50 sessions per architecture and speed. One network is varied while the other remains at its baseline specification. Inputs and fitted parameters are held fixed across speeds. † marks the baseline specifications, which represent the same trader. Panel B jointly tests the 16 architecture-by-speed interactions using HC3-robust standard errors. The null is that market-speed effects on mean NLV are the same across architectures. Decision times are medians over ticks when only the relevant network operates, pooled across the 50 sessions. They measure the agent's decision routine, excluding order submission. Execution-network parameter counts include a fifth input fixed at zero.

Panel A: Performance by network size

Hidden layers		Mean session NLV			Median decision time (ms)		
× units	Parameters	Normal	Slow	Fast	Normal	Slow	Fast
Selection network varied							
1 × 8†	49	42.1	38.5	40.6	0.34	0.37	0.33
1 × 32	193	37.9	44.4	40.7	0.32	0.36	0.35
1 × 128	769	43.1	43.8	46.3	0.33	0.35	0.36
2 × 8	121	43.9	40.7	44.0	0.35	0.36	0.38
2 × 32	1,249	42.1	42.2	46.0	0.35	0.38	0.38
2 × 128	17,281	38.2	36.1	40.4	0.36	0.38	0.37
3 × 8	193	37.3	39.7	42.9	0.36	0.40	0.40
3 × 32	2,305	44.7	40.2	43.0	0.37	0.40	0.39
3 × 128	33,793	43.0	43.3	42.4	0.39	0.41	0.39
Mean across architectures		41.4	41.0	42.9	0.35	0.38	0.37
Execution network varied							
1 × 32	885	39.7	35.4	37.7	0.34	0.36	0.38
1 × 128	3,477	36.5	33.0	37.5	0.34	0.38	0.38
1 × 512	13,845	40.6	38.2	34.4	0.35	0.37	0.38
2 × 32	1,941	41.3	40.6	41.8	0.37	0.40	0.39
2 × 128†	19,989	42.1	38.5	40.6	0.45	0.43	0.38
2 × 512	276,501	39.0	42.4	41.9	0.56	0.53	0.53
3 × 32	2,997	41.0	39.1	37.4	0.39	0.41	0.43
3 × 128	36,501	37.4	41.8	35.1	0.40	0.43	0.42
3 × 512	539,157	39.3	42.4	36.7	0.68	0.64	0.62
Mean across architectures		39.7	39.0	38.1	0.43	0.44	0.44

Panel B: Architecture-by-speed interaction tests

Network varied	HC3-robust F
Selection	0.50
Execution	0.59

Table 11: NN and LLM Tender Selection Across Liquidity Conditions. Panel A reports means across 150 independent sessions per cell; dollar amounts are in $000s. Normal, half, and quarter frequency correspond to 120, 60, and 30 computer-trader orders per simulation minute, with the market clock fixed. Both selectors use the same trained NN execution policy; the LLM is `gpt-5.4` with medium reasoning. Sharpe is mean NLV divided by its standard deviation. Panels B and C report changes in mean NLV from normal to half and quarter frequency, respectively. Changes equal reduced-frequency mean NLV minus normal-frequency mean NLV. The difference-in-differences (DiD) is the NN trader's change minus the LLM agent's change; a negative value indicates a larger profit decline for the NN trader. All contrasts are obtained from one OLS regression of session NLV on indicators for the NN trader, half frequency, and quarter frequency, together with the interactions between the NN indicator and the two frequency indicators. The regression uses all 900 sessions, with the LLM agent under normal liquidity as the reference cell. Standard errors in parentheses are HC3 heteroskedasticity-robust standard errors.

Panel A: Descriptive statistics

Cell	Mean NLV	Std NLV	Sharpe	% Profitable	Mean Trading P&L	Mean Fines	Tenders Accepted
NN trader / Normal	40.4	20.3	1.99	96%	40.4	0.0	2.2
NN trader / Half	27.6	18.8	1.47	87%	27.7	0.0	1.6
NN trader / Quarter	15.3	14.4	1.06	65%	15.3	0.0	1.0
LLM agent / Normal	41.6	20.0	2.09	98%	41.6	0.0	4.2
LLM agent / Half	34.5	16.7	2.07	99%	34.5	0.0	3.7
LLM agent / Quarter	25.8	14.0	1.85	97%	25.8	0.0	3.2

Panel B: Change in mean NLV, normal to half frequency

			Change (Half − Normal)				DiD (NN − LLM)			
Trader	Normal	Half	Change	(SE)	t	p	DiD	(SE)	t	p
NN trader	40.4	27.6	−12.8	(2.27)	−5.64	<0.001	−5.7	(3.11)	−1.82	0.069
LLM agent	41.6	34.5	−7.1	(2.13)	−3.34	<0.001		—		

Panel C: Change in mean NLV, normal to quarter frequency

			Change (Quarter − Normal)				DiD (NN − LLM)			
Trader	Normal	Quarter	Change	(SE)	t	p	DiD	(SE)	t	p
NN trader	40.4	15.3	−25.1	(2.04)	−12.34	<0.001	−9.3	(2.85)	−3.27	0.001
LLM agent	41.6	25.8	−15.8	(2.00)	−7.92	<0.001		—		

Table 12: Provider Robustness: Descriptive Means Across Models. Cell-level means across 50 sessions per (model × reasoning) cell, extending Table 1 to four additional models: two OpenAI scales (gpt-5.4-mini, gpt-5.4-nano) and two Anthropic models (claude-opus-4-6, claude-sonnet-4-6). Low and Medium reasoning labels correspond to OpenAI `reasoning_effort` or Anthropic extended-thinking budgets, depending on the provider. Same columns as Table 1; normal simulation speed only ($N = 750$). Sharpe is mean NLV divided by std NLV within cell. Dollar amounts in thousands ($000s).

Cell	Mean NLV	Std NLV	Sharpe	% Profitable	Mean Trading P&L	Mean Fines	Mean Think Time (ms)	Tenders Accepted
gpt-5.4 / Off	8.3	37.0	0.22	62%	22.6	14.3	2,939	3.7
gpt-5.4 / Low	18.6	46.3	0.40	74%	25.5	7.0	3,789	3.7
gpt-5.4 / Medium	7.3	62.8	0.12	62%	28.8	21.4	7,303	3.3
gpt-5.4-mini / Off	−26.8	54.5	-0.49	40%	9.8	36.7	2,153	2.7
gpt-5.4-mini / Low	−19.7	45.7	-0.43	42%	8.7	28.4	3,255	4.5
gpt-5.4-mini / Medium	5.5	59.9	0.09	64%	24.1	18.6	7,709	3.7
gpt-5.4-nano / Off	1.4	19.7	0.07	36%	5.6	4.2	2,195	1.3
gpt-5.4-nano / Low	−19.9	49.0	-0.41	42%	10.0	29.9	3,014	3.1
gpt-5.4-nano / Medium	1.1	48.2	0.02	56%	19.1	18.0	4,695	3.9
Opus 4.6 / Off	28.4	25.2	1.13	86%	29.3	0.9	3,132	4.0
Opus 4.6 / Low	24.8	31.4	0.79	80%	27.0	2.2	3,712	3.8
Opus 4.6 / Medium	4.7	54.5	0.09	60%	16.5	11.8	5,911	3.7
Sonnet 4.6 / Off	30.9	35.2	0.88	80%	35.4	4.5	3,859	4.3
Sonnet 4.6 / Low	20.8	32.2	0.64	76%	34.6	13.8	7,904	3.5
Sonnet 4.6 / Medium	10.2	39.3	0.26	62%	19.2	9.0	16,107	3.1

Online Appendix: Tables

Table A1: Case Parameters. Case parameters for the LT3 liability trading experiment. All sessions use the same case configuration with randomized price paths.

Parameter	Value
Market structure	
Securities	CRZY, TAME (two independent order books)
Duration	300 ticks
Max order size	CRZY: 25,000 shares; TAME: 10,000 shares
Position limits	
Net limit	100,000 shares
Gross limit	250,000 shares
Tender offers	
Arrival	Institutional tenders arrive throughout the session
Duration	$\approx$ 30 ticks per tender
Costs and penalties	
Transaction fee	\$0.02 per share (market trades only; tender fills free)
Front-running fine	\$0.20/share $\leq$ 5,000 shares; \$0.40/share beyond
End-of-case fine	\$1.00/share on uncovered positions
Experimental design	
Model	gpt-5.4-2026-03-05
Reasoning levels	Off, Low, Medium
Speeds	50%, 100%, 150%
Sessions per cell	50
Total sessions	450 (9 cells $\times$ 50)
Agent polling	$\approx$ 1 tick between observations

Table A2: LT3 Prompt: Rendered Example. System prompt and user message for the first tender-visible turn of an example LT3 session.

```
=== SYSTEM PROMPT ===
You are a liability trader in a live RIT (Rotman Interactive Trader) case.

## Securities
There are 2 securities: CRZY, TAME.
Each has its own order book. When submitting orders, use the ticker as the "venue" parameter (e.g. venue="CRZY").
Positions are tracked per security.

**Order size limits** (per single order -- RIT will REJECT orders exceeding these):
  CRZY: max 25,000 shares per order
  TAME: max 10,000 shares per order
If you need to trade more than the limit, split into multiple smaller orders.

## Case Structure
Duration: 300 ticks (each tick is ~1 second of real time).
You are called periodically to observe and optionally act. The tick numbers in your observations show how much time has elapsed between your
      turns.

## Your Job: Liability Trading
Institutional tender offers arrive during the case. Each tender specifies a ticker, a side (BUY or SELL from your perspective), a quantity, a
      price, and a deadline.

Your job:
1. Evaluate the tender: check the order book depth for THAT ticker. Can you unwind profitably?
2. If yes, ACCEPT the tender using decide_tender.
3. Unwind the acquired position by submitting orders on the SAME ticker.
4. Tenders remain active for approximately 30 ticks (~30 seconds). Due to polling latency, you may not see a tender until a few ticks after
      it is issued. The time_left field shows how many ticks remain before expiration. If you do not accept before the deadline, the tender
      expires automatically -- there is no need to explicitly decline.

Do NOT speculate or make markets. Only trade to unwind accepted tenders.

## Front-Running Prohibition
Do NOT trade a security to anticipate a pending tender. If you have no existing position in a security, you must accept or let expire the
      tender BEFORE placing any orders on that security. Exception: if you are already unwinding a position from a previously accepted
      tender, you may continue closing that position even if a new tender on the same security arrives.

## Position Limits
Net limit: 100,000 shares (sum of all positions)
Gross limit: 250,000 shares (sum of absolute positions)

## Transaction Fees
Fee: $0.02 per share on every market trade.
Tender fills do not incur a transaction fee.

## Fines
Front-running or speculative trading (any trade not directly unwinding an accepted tender) incurs fines: $0.20/share up to 5,000 shares,
      $0.40/share beyond that. Uncovered positions at the end of the case are fined at $1.00/share. These fines are assessed in real time and
```

Table A2 *(continued)*

```
      reduce your P&L.

**Important:** If you submit unwind orders in the same action as accepting a tender, you may incur front-running fines because the tender has
      not yet settled. Accepting the tender in one step then submitting unwind orders in a subsequent step will avoid these fines.

## End of Case
Aim to be flat (zero inventory on all securities) before the case ends. Monitor the tick count and close positions early.

## Available Tools
- submit_order: Submit an order. Set venue to the ticker name. Types: LIMIT, MARKETABLE_LIMIT, MARKET.
- cancel_order: Cancel a resting order by order_id.
- decide_tender: Accept or decline a tender by tender_id.
You do not need to act every turn.




## Objective
Maximise P&L by accepting profitable tenders and unwinding positions efficiently. Manage risk and close all positions before the case ends.




=== USER MESSAGE (turn 10) ===
=== Tick 27 ===

MARKET STATE (CRZY):
  Best bid: 10.10
  Best ask: 10.12
  Last: 10.12
  Volume: 88,100 My volume: 88,100
  ORDER BOOK:
    BIDS ASKS
       10.10 x 5200 10.12 x 500
       10.07 x 4500 10.13 x 5900
       10.07 x 4400 10.13 x 4600
       10.06 x 3700 10.13 x 4700
       10.06 x 5000 10.14 x 5600
       10.06 x 4500 10.16 x 5200
       10.03 x 4500 10.21 x 5600
       10.02 x 3200 10.27 x 4800
       10.02 x 4900 10.30 x 4100
       10.01 x 5500 10.32 x 5100
       10.01 x 5100 10.34 x 4400
       10.00 x 5600 10.44 x 4200
        9.99 x 5000 10.45 x 4400
        9.98 x 700
        9.98 x 4600
        9.98 x 5500
        9.97 x 4800
        9.97 x 5200
        9.97 x 5600
        9.96 x 4000

MARKET STATE (TAME):
```

Table A2 ***(continued)***

```
  Best bid: 25.15
  Best ask: 25.21
  Last: 25.15
  Volume: 101,300 My volume: 101,300
  ORDER BOOK:
    BIDS ASKS
       25.15 x 100 25.21 x 2500
       25.15 x 5700 25.27 x 4400
       25.14 x 200 25.27 x 5300
       25.11 x 4400 25.28 x 4500
       25.08 x 300 25.31 x 5700
       25.03 x 5900 25.41 x 5800
       25.02 x 5100 25.42 x 5200
       25.02 x 4400 25.43 x 4500
       25.00 x 5200 25.46 x 5500
       25.00 x 4400 25.46 x 5900
       24.97 x 300 25.51 x 4600
       24.95 x 5100 25.58 x 4600
       24.95 x 5700 25.63 x 4600
       24.94 x 6000
       24.92 x 5700
       24.92 x 5700
       24.92 x 5900
       24.92 x 5400
       24.92 x 5600
       24.92 x 5500

YOUR POSITIONS:
  CRZY: 0 shares (max order: 25000) P&L: +0.00
  TAME: 0 shares (max order: 10000) P&L: +0.00
  Net: 0 shares

YOUR P&L: $0.00
YOUR FINES: $0.00
NLV (net of fines): $0.00

YOUR LIMITS:
  Net: 0 / 100,000 (100,000 remaining)
  Gross: 0 / 250,000 (250,000 remaining)

TENDERS:
  [0] "An institution would like to BUY 70000 shares of CRZY from you at a price of 10.46. Would you like to SELL the shares to them?"
        (expires in 29 ticks)
```

Table A3: Measurement robustness: fixed-window edge capture (W30). Mirrors Table 7 (Tender Execution Quality) on the same intent-funnel frame, with the only substantive change being that the fixed-window measure *W30* replaces *Edge Captured* in Panel A column (3) and as the dependent variable in Panel B column (2). *W30* is the ticker PnL delta between the accept turn and a deterministic endpoint 30 ticks later (capped at the next same-ticker accept or session end), divided by tender quantity. Because the window length is fixed rather than scaling with unwind duration, W30 holds exposure constant across cells – an alternative to *Edge Captured* that does not absorb variation in unwind duration. Panel A reports per-cell summary statistics on the post-fill unwind. *Unwind Completed* is the share of filled intents for which the agent returned to its pre-acceptance position before session end or before accepting another tender for the same ticker. *W30 Captured* ($/share) is the cell mean of W30 restricted to completed unwinds; *Book Edge at Intent* ($/share) is the cell mean of the book edge at the decision turn restricted to the same matched completed-unwind sub-sample (so the W30 column and the Book Edge at Intent column compare on identical tenders). Panel B reports the corresponding cell-level regressions on reasoning indicators (reference: off) with speed fixed effects. Col (1) is a linear probability model on the full filled-intent sample. Col (2) regresses W30 on reasoning controlling for book edge at the decision turn (completed unwinds). Col (3) regresses Book Edge at Intent on reasoning (completed unwinds). Cluster-robust standard errors (by session) in parentheses. ***/**/* = 1/5/10%.

Panel A: Summary statistics (filled intents)

Cell	N	Unwind Completed	W30 Captured ($/share)	Book Edge at Intent ($/share)
Normal / Off	187	88%	0.1339	0.1124
Normal / Low	187	89%	0.1589	0.1307
Normal / Medium	166	78%	0.2119	0.1327
Fast / Off	194	88%	0.1338	0.1151
Fast / Low	169	83%	0.1819	0.1336
Fast / Medium	125	68%	0.2467	0.1371
Slow / Off	172	92%	0.1066	0.1247
Slow / Low	181	97%	0.1136	0.1256
Slow / Medium	192	91%	0.1700	0.1301

Panel B: Cell-level regressions

	(1) Unwind Completed	(2) W30 Captured	(3) Book Edge at Intent
Low	0.0003	0.0164**	0.0126**
	(0.0223)	(0.0079)	(0.0049)
Medium	−0.0977***	0.0701***	0.0155***
	(0.0252)	(0.0100)	(0.0051)
Book edge		0.8181***	
		(0.0640)	
Constant	0.8818***	0.0371***	0.1159***
	(0.0219)	(0.0108)	(0.0051)
Speed FE	Yes	Yes	Yes
Sample	Filled	Completed	Completed
N	1573	1231	1231
R^2	0.0411	0.2480	0.0088

Table A4: LT3 Decision-Support Prompt: Rendered Example. System prompt and user message for the first tender-visible turn of an example decision-support session; the decision-support data feeds the Agent + DS rows of Table 9.

```
=== SYSTEM PROMPT ===
You are a liability trader in a live RIT (Rotman Interactive Trader) case.

## Securities
There are 2 securities: CRZY, TAME.
Each has its own order book. When submitting orders, use the ticker as the "venue" parameter (e.g. venue="CRZY").
Positions are tracked per security.

**Order size limits** (per single order -- RIT will REJECT orders exceeding these):
  CRZY: max 25,000 shares per order
  TAME: max 10,000 shares per order
If you need to trade more than the limit, split into multiple smaller orders.

## Case Structure
Duration: 300 ticks (each tick is ~1 second of real time).
You are called periodically to observe and optionally act. The tick numbers in your observations show how much time has elapsed between your
      turns.

## Your Job: Liability Trading
Institutional tender offers arrive during the case. Each tender specifies a ticker, a side (BUY or SELL from your perspective), a quantity, a
      price, and a deadline.

Your job:
1. Evaluate the tender: check the order book depth for THAT ticker. Can you unwind profitably?
2. If yes, ACCEPT the tender using decide_tender.
3. Unwind the acquired position by submitting orders on the SAME ticker.
4. Tenders remain active for approximately 30 ticks (~30 seconds). Due to polling latency, you may not see a tender until a few ticks after
      it is issued. The time_left field shows how many ticks remain before expiration. If you do not accept before the deadline, the tender
      expires automatically -- there is no need to explicitly decline.

Do NOT speculate or make markets. Only trade to unwind accepted tenders.

## Front-Running Prohibition
Do NOT trade a security to anticipate a pending tender. If you have no existing position in a security, you must accept or let expire the
      tender BEFORE placing any orders on that security. Exception: if you are already unwinding a position from a previously accepted
      tender, you may continue closing that position even if a new tender on the same security arrives.

## Position Limits
Net limit: 100,000 shares (sum of all positions)
Gross limit: 250,000 shares (sum of absolute positions)

## Transaction Fees
Fee: $0.02 per share on every market trade.
Tender fills do not incur a transaction fee.

## Fines
Front-running or speculative trading (any trade not directly unwinding an accepted tender) incurs fines: $0.20/share up to 5,000 shares,
```

Table A4 ***(continued)***

```
      $0.40/share beyond that. Uncovered positions at the end of the case are fined at $1.00/share. These fines are assessed in real time and
      reduce your P&L.

**Important:** If you submit unwind orders in the same action as accepting a tender, you may incur front-running fines because the tender has
      not yet settled. Accepting the tender in one step then submitting unwind orders in a subsequent step will avoid these fines.

## End of Case
Aim to be flat (zero inventory on all securities) before the case ends. Monitor the tick count and close positions early.

## Available Tools
- submit_order: Submit an order. Set venue to the ticker name. Types: LIMIT, MARKETABLE_LIMIT, MARKET.
- cancel_order: Cancel a resting order by order_id.
- decide_tender: Accept or decline a tender by tender_id.
You do not need to act every turn.

## Decision Support
Observations may include a DECISION SUPPORT block with precomputed summaries derived from the current book, fees, and limits. These summaries
      are informational only; you remain responsible for the trading decision.



## Objective
Maximise P&L by accepting profitable tenders and unwinding positions efficiently. Manage risk and close all positions before the case ends.




=== USER MESSAGE (turn 7) ===
=== Tick 20 ===

MARKET STATE (CRZY):
  Best bid: 9.89
  Best ask: 9.94
  Last: 9.89
  Volume: 80,700 My volume: 80,700
  ORDER BOOK:
    BIDS ASKS
       9.89 x 2200 9.94 x 4800
       9.89 x 5900 9.98 x 3600
       9.88 x 4500 9.98 x 5700
       9.88 x 4200 9.98 x 4500
       9.88 x 4800 9.99 x 4500
       9.88 x 4700 10.01 x 4900
       9.84 x 5600 10.01 x 5400
       9.84 x 4100 10.03 x 4300
       9.83 x 5200 10.03 x 5400
       9.83 x 4500 10.04 x 300
       9.79 x 5900 10.08 x 4900
       9.78 x 4300 10.09 x 4500
       9.73 x 5400 10.11 x 4900
       9.73 x 4100 10.13 x 4800
       9.73 x 6000 10.17 x 4800
       9.59 x 4200 10.19 x 4400
```

Table A4 ***(continued)***

```
       9.58 x 4700 10.19 x 5700
       9.54 x 5700 10.19 x 4400
       9.51 x 5100 10.27 x 4500
                           10.27 x 5800

MARKET STATE (TAME):
  Best bid: 24.89
  Best ask: 24.91
  Last: 24.89
  Volume: 79,900 My volume: 79,900
  ORDER BOOK:
    BIDS ASKS
       24.89 x 1800 24.91 x 3900
       24.88 x 5500 24.93 x 5900
       24.87 x 5500 24.94 x 4800
       24.86 x 5600 24.97 x 5000
       24.85 x 5700 24.99 x 3600
       24.84 x 5900 25.00 x 500
       24.81 x 4900 25.02 x 5800
       24.80 x 4700 25.02 x 4600
       24.75 x 4400 25.03 x 6000
       24.75 x 5300 25.03 x 5100
       24.74 x 5200 25.03 x 4200
       24.67 x 6000 25.04 x 4800
       24.55 x 5000 25.05 x 5400
                            25.07 x 5300
                            25.11 x 5300
                            25.11 x 4600
                            25.11 x 5200
                            25.12 x 5300
                            25.16 x 5600
                            25.19 x 5500

YOUR POSITIONS:
  CRZY: 0 shares (max order: 25000) P&L: +0.00
  TAME: 0 shares (max order: 10000) P&L: +0.00
  Net: 0 shares

YOUR P&L: $0.00
YOUR FINES: $0.00
NLV (net of fines): $0.00

YOUR LIMITS:
  Net: 0 / 100,000 (100,000 remaining)
  Gross: 0 / 250,000 (250,000 remaining)

TENDERS:
  [0] "An institution would like to BUY 65000 shares of TAME from you at a price of 25.14. Would you like to SELL the shares to them?"
        (expires in 30 ticks)

DECISION SUPPORT:
  Tender #0 (TAME):
    Unwind market VWAP (full size): 25.0042
```

Table A4 ***(continued)***

```
Implied edge / share (net of fee): +0.1158
Implied total edge: +7,527.00
Projected position (TAME): -65,000
Projected net (signed): -65,000
Projected gross: 65,000
Would breach net limit: no
Would breach gross limit: no
```

Table A5: Execution-Support Prompt: Rendered Example. System prompt and user message for the first tender-visible LLM dispatch of an example execution-support session; the execution-support data feeds the Agent + DS + Algo Exec rows of Table 9.

```
=== SYSTEM PROMPT ===
You are the tender-acceptance gate in a hybrid trading system on a live RIT (Rotman Interactive Trader) liability case.

A scripted unwind algorithm runs continuously and is responsible for closing any inventory you cause to appear. Your job is narrow: when a
      tender arrives, decide whether to ACCEPT it or render NO_DECISION.

## Securities
There are 2 securities: CRZY, TAME.
Each has its own order book. Positions are tracked per security.

**Order size limits** (per single order -- RIT will REJECT orders exceeding these):
  CRZY: max 25,000 shares per order
  TAME: max 10,000 shares per order
If you need to trade more than the limit, split into multiple smaller orders.

## Case Structure
Duration: 300 ticks (each tick is ~1 second of real time).
You will be called repeatedly while a tender is live with a fresh snapshot of the market. Each call is independent. You may be called again.

## The Unwind Algorithm
If you ACCEPT a tender, an external algorithm will unwind the resulting position by submitting MARKET orders, capped at 5,000 shares per
      ticker per tick, polling once per second until the position is flat. The algorithm:
- Uses MARKET orders only -- it does NOT use LIMIT or MARKETABLE_LIMIT.
- Does NOT wait for favorable fills, rest passively, or cancel and re-price.
- Walks the visible book on the unwind side at whatever prices are available.
- Never modifies its rate based on the spread; it always burns through 5,000 shares/tick until done.

## Your Job: Tender decision
Institutional tender offers arrive during the case. Each tender specifies a ticker, a side (BUY or SELL from your perspective), a quantity, a
      price, and a deadline. You will be called to either ACCEPT or return NO_DECISION.

Your job:
1. Evaluate the tender: check the order book depth for THAT ticker. Will you profit from the trade, knowing how the unwind algorithm behaves?
2. If yes, ACCEPT the tender using tender_decision.
3. Tenders remain active for approximately 30 ticks (~30 seconds). Due to polling latency, you may not see a tender until a few ticks after
      it is issued. The time_left field shows how many ticks remain before expiration. If you do not accept before the deadline, the tender
      expires automatically -- there is no need to explicitly decline.

## Position Limits
Net limit: 100,000 shares (sum of all positions)
Gross limit: 250,000 shares (sum of absolute positions)

## Transaction Fees
Fee: $0.02 per share on every market trade.
Tender fills do not incur a transaction fee.

## Fines
```

Table A5 ***(continued)***

```
Uncovered positions at the end of the case are fined at $1.00/share.

## End of Case
Aim to be flat (zero inventory on all securities) before the case ends. Monitor the tick count.

## Available Tools
You must output your decision via the ‘tender_decision‘ tool with one of two values:
- **ACCEPT** -- bind the tender immediately. The system will execute the accept and the unwind algorithm will take over.
- **NO_DECISION** -- defer. You will likely be asked again about this same tender on a later firing as the book evolves. Tenders that are
      never accepted expire automatically; there is no need to issue an explicit decline.

Call the ‘tender_decision‘ tool exactly once with ‘{"decision": "ACCEPT"}‘ or ‘{"decision": "NO_DECISION"}‘. Include any reasoning text in
      the model response. Do not call any other tool.

## Decision Support
Observations may include a DECISION SUPPORT block with precomputed summaries derived from the current book, fees, and limits. These summaries
      are informational only; you remain responsible for the trading decision.

## Objective
Maximise P&L by accepting profitable tenders with the knowledge that the unwind algorithm will work to unwind your order according to the
      behavior described above.



=== USER MESSAGE (tick 24) ===
=== Tick 24 / 300 ===

MARKET STATE (CRZY):
  Best bid: 10.00
  Best ask: 10.03
  Last: 10.03
  ORDER BOOK:
    BIDS ASKS
      10.00 x 5900 10.03 x 500
      10.00 x 4600 10.04 x 4200
       9.99 x 5500 10.05 x 5400
       9.99 x 4600 10.06 x 5100
       9.97 x 4900 10.06 x 5300

MARKET STATE (TAME):
  Best bid: 24.64
  Best ask: 24.67
  Last: 24.69
  ORDER BOOK:
    BIDS ASKS
      24.64 x 4600 24.67 x 2300
      24.52 x 5800 24.67 x 5700
      24.42 x 4500 24.71 x 5400
      24.26 x 4700 24.72 x 4700
      24.18 x 4500 24.74 x 6000

YOUR POSITIONS:
  CRZY: +0 shares (max order: 25,000)
```

Table A5 ***(continued)***

```
  TAME: +0 shares (max order: 10,000)
  Net: +0 shares

YOUR P&L: $0.00
YOUR FINES: $0.00
NLV (net of fines): $0.00

YOUR LIMITS:
  Net: 0 / 100,000
  Gross: 0 / 250,000

TENDER UNDER DECISION:
  Tender ID: 0
  Ticker: TAME
  Side: BUY (your side)
  Quantity: 70,000 shares
  Price: $24.4300
  Expires: in 30 ticks
  Caption: An institution would like to SELL 70000 shares of TAME to you at a price of 24.43. Would you like to BUY the shares from them?

DECISION SUPPORT:
  Tender #0 (TAME):
    Unwind market VWAP (full size): insufficient book depth
    Implied edge / share (net of fee): insufficient book depth
    Implied total edge: insufficient book depth
    Projected position (TAME): +70,000
    Projected net (signed): +70,000
    Projected gross: 70,000
    Would breach net limit: no
    Would breach gross limit: no

Call the ‘tender_decision‘ tool exactly once with ‘{"decision": "ACCEPT"}‘ or ‘{"decision": "NO_DECISION"}‘.
```

Table A6: Less-Directive Prompt Ablation (Normal Speed). Robustness check on the prompt design. The less-directive ablation strips three rush-inducing phrases (*"close positions early"*, *"close all positions before the case ends"*, *"efficiently"*) from the baseline prompt while keeping rules, fines, limits, fees, and tools byte-identical; reruns the normal-speed market with gpt-5.4 across reasoning off / low / medium (50 sessions per prompt × reasoning cell, $N = 300$). Panel A reports cell-level means pooled across both prompts; Δ is less-directive minus baseline within each reasoning level. Panel B reports OLS regressions of performance on reasoning indicators, a prompt indicator, and their interactions, with Wald F-tests on the grouped reasoning, prompt, and interaction coefficients at the bottom. References: reasoning off, prompt baseline. Dollar amounts in thousands ($000s). HC3 robust standard errors in parentheses. ***, **, * denote significance at the 1%, 5%, and 10% levels.

Panel A: Cell means, baseline vs. less-directive ablation

Prompt	Reasoning	Mean NLV	Std NLV	% Profitable	Mean Trading P&L	Mean Fines	Mean Think Time (ms)	Turns	Tenders Accepted
Baseline	Off	8.3	37.0	62%	22.6	14.3	2,939	75.0	3.7
Less-Directive	Off	−3.2	55.2	58%	13.9	17.1	2,927	75.1	3.9
	Δ	−11.5		−4%	−8.7	2.8	−12	0.1	0.2
Baseline	Low	18.6	46.3	74%	25.5	7.0	3,789	63.3	3.7
Less-Directive	Low	19.0	44.8	72%	29.2	10.2	4,129	59.0	3.9
	Δ	0.5		−2%	3.7	3.2	340	−4.3	0.1
Baseline	Medium	7.3	62.8	62%	28.8	21.4	7,303	40.2	3.3
Less-Directive	Medium	−3.5	60.0	52%	21.8	25.4	8,063	37.9	3.2
	Δ	−10.9		−10%	−6.9	4.0	761	−2.2	−0.1
Baseline	All	11.4	49.8	66%	25.6	14.2	4,677	59.5	3.6
Less-Directive	All	4.1	54.4	61%	21.7	17.6	5,040	57.3	3.7

Panel B: Prompt × Reasoning interaction (pooled $N = 300$)

	(1) NLV	(2) Trading P&L	(3) Fines
Low	10.27	2.94	−7.33*
	(8.47)	(6.60)	(4.11)
Medium	−0.97	6.18	7.15
	(10.41)	(7.63)	(5.65)
Less-Directive	−11.47	−8.66	2.81
	(9.49)	(7.08)	(5.23)
Low × Less-Directive	11.94	12.37	0.43
	(13.22)	(9.97)	(6.57)
Medium × Less-Directive	0.58	1.74	1.15
	(15.62)	(11.47)	(8.11)
Constant	8.31	22.59***	14.28***
	(5.29)	(4.23)	(3.24)
Prompt F	1.46 (0.228)	1.50 (0.222)	0.29 (0.592)
Prompt × Reason F	0.49 (0.616)	0.87 (0.422)	0.01 (0.990)
Reason F	0.86 (0.423)	0.34 (0.711)	4.30 (0.014)
N	300	300	300
R^2	0.0302	0.0181	0.0571

Table A7: Less-Directive Ablation Prompt: Rendered Example. System prompt and user message for the first tender-visible turn of an example less-directive session feeding Table A6 (`level2_ablation.md` as rendered against the live case configuration). The runtime placeholder {`reflection_block`} is empty in this session.

```
=== SYSTEM PROMPT ===
You are a liability trader in a live RIT (Rotman Interactive Trader) case.

## Securities
There are 2 securities: CRZY, TAME.
Each has its own order book. When submitting orders, use the ticker as the "venue" parameter (e.g. venue="CRZY").
Positions are tracked per security.

**Order size limits** (per single order -- RIT will REJECT orders exceeding these):
  CRZY: max 25,000 shares per order
  TAME: max 10,000 shares per order
If you need to trade more than the limit, split into multiple smaller orders.

## Case Structure
Duration: 300 ticks (each tick is ~1 second of real time).
You are called periodically to observe and optionally act. The tick numbers in your observations show how much time has elapsed between your
      turns.

## Your Job: Liability Trading
Institutional tender offers arrive during the case. Each tender specifies a ticker, a side (BUY or SELL from your perspective), a quantity, a
      price, and a deadline.

When a tender is live:
1. Evaluate the tender: check the order book depth for THAT ticker. Can you unwind profitably?
2. If yes, ACCEPT the tender using decide_tender.
3. You may let a tender expire. There is no need to explicitly decline.

After accepting a tender:
1. Unwind the resulting position by submitting orders on the SAME ticker.
2. There is no requirement to trade immediately after acceptance or to act on every observation.

Tenders remain active for approximately 30 ticks (~30 seconds). Due to polling latency, you may not see a tender until a few ticks after it
      is issued. The time_left field shows how many ticks remain before expiration. If you do not accept before the deadline, the tender
      expires automatically.

Do NOT speculate or make markets. Only trade to unwind positions created by accepted tenders.

## Front-Running Prohibition
Do NOT trade a security to anticipate a pending tender. If you have no existing position in a security, you must accept or let expire the
      tender BEFORE placing any orders on that security. Exception: if you are already unwinding a position from a previously accepted
      tender, you may continue closing that position even if a new tender on the same security arrives.

## Position Limits
Net limit: 100,000 shares (sum of all positions)
Gross limit: 250,000 shares (sum of absolute positions)
```

Table A7 ***(continued)***

```
## Transaction Fees
Fee: $0.02 per share on every market trade.
Tender fills do not incur a transaction fee.

## Fines
Front-running or speculative trading (any trade not directly unwinding an accepted tender) incurs fines: $0.20/share up to 5,000 shares,
      $0.40/share beyond that. Uncovered positions at the end of the case are fined at $1.00/share. These fines are assessed in real time and
      reduce your P&L.

**Important:** If you submit unwind orders in the same action as accepting a tender, you may incur front-running fines because the tender has
      not yet settled. Accepting the tender in one step then submitting unwind orders in a subsequent step will avoid these fines.

## End of Case
Aim to be flat (zero inventory on all securities) before the case ends.

## Available Tools
- submit_order: Submit an order. Set venue to the ticker name. Types: LIMIT, MARKETABLE_LIMIT, MARKET.
- cancel_order: Cancel a resting order by order_id.
- decide_tender: Accept or decline a tender by tender_id.
You do not need to act every turn.



## Objective
Maximise P&L by accepting profitable tenders and unwinding positions.




=== USER MESSAGE (turn 5) ===
=== Tick 18 ===

MARKET STATE (CRZY):
  Best bid: 10.13
  Best ask: 10.15
  Last: 10.12
  Volume: 81,800 My volume: 81,800
  ORDER BOOK:
    BIDS ASKS
       10.13 x 4200 10.15 x 5500
       10.09 x 4300 10.16 x 4300
       10.08 x 1300 10.17 x 5700
       10.06 x 5100 10.18 x 5000
       10.04 x 1200 10.21 x 4800
       10.03 x 5300 10.24 x 4900
       10.00 x 2000 10.26 x 5100
       10.00 x 5600 10.27 x 4200
        9.98 x 4900 10.58 x 4200
        9.97 x 3300
        9.97 x 4600
        9.96 x 5700
        9.95 x 5800
        9.94 x 4100
```

Table A7 ***(continued)***

```
       9.94 x 5500
       9.88 x 5200
       9.86 x 4400
       9.86 x 4400
       9.85 x 5800
       9.84 x 4500

MARKET STATE (TAME):
  Best bid: 24.70
  Best ask: 24.71
  Last: 24.70
  Volume: 95,700 My volume: 95,700
  ORDER BOOK:
    BIDS ASKS
      24.70 x 4700 24.71 x 4800
      24.69 x 4900 24.77 x 600
      24.69 x 4700 24.80 x 900
      24.69 x 4500 24.82 x 5800
      24.68 x 4700 24.83 x 2300
      24.67 x 5900 24.83 x 4300
      23.96 x 4900 24.84 x 6000
                           24.85 x 4700
                           24.87 x 5400
                           24.88 x 4200
                           24.90 x 4300
                           24.94 x 5600
                           24.96 x 4700
                           24.98 x 5000
                           24.99 x 5500
                           25.03 x 1300
                           25.03 x 4300
                           25.03 x 5900
                           25.07 x 2900
                           25.09 x 4300

YOUR POSITIONS:
  CRZY: 0 shares (max order: 25000) P&L: +0.00
  TAME: 0 shares (max order: 10000) P&L: +0.00
  Net: 0 shares

YOUR P&L: $0.00
YOUR FINES: $0.00
NLV (net of fines): $0.00

YOUR LIMITS:
  Net: 0 / 100,000 (100,000 remaining)
  Gross: 0 / 250,000 (250,000 remaining)

TENDERS:
  [0] "An institution would like to BUY 74000 shares of TAME from you at a price of 25.01. Would you like to SELL the shares to them?"
        (expires in 27 ticks)
```

Table A8: NN Trader Training Parameters. Mean NLV in $000s on held-out simulated sessions. Panel A varies the execution window H and terminal penalty Φ. Panel B compares fixed book-edge acceptance thresholds. Panel C averages validation (out-of-sample) performance across 3 seeds by varying regularization coefficient λ_w.

Panel A: Execution window and terminal penalty

	Terminal penalty per share		
Window H	$\Phi = 1$	$\Phi = 10$	$\Phi = 50$
10	47.3	45.7	46.4
15	43.3	46.5	**50.3**
30	46.8	47.4	44.2

Panel B: Initial acceptance threshold

Book-edge threshold	Mean validation NLV
$0.10	47.3
$0.15	**50.7**
$0.20	44.5

Panel C: Selection-network regularization

Penalty coefficient λ_w	Mean validation NLV
0	39.7
100	**40.1**
300	**40.1**

Table A9: Trading Performance with Reflection Between Sessions. Mean session NLV in $000s; 50 sessions per baseline cell and 50 three-session chains per reflection configuration. Baseline is the corresponding no-reflection result in Table 1 (Panel A) or Table 12 (Panel B). Session 3 is played with a playbook revised after each of the first two sessions by a separate `gpt-5.4` model with `xhigh` reasoning. Difference is session-3 mean NLV minus baseline mean NLV, with Welch standard errors below in parentheses. The final row averages the twelve configuration-specific differences. ***, **, and * denote significance at the 1%, 5%, and 10% levels, respectively.

Market speed	Reasoning	Baseline	Session 3	Difference
	Panel A: `gpt-5.4`			
Slow	Off	2.7	19.7	17.0**
				(7.9)
	Low	22.9	31.5	8.6*
				(4.5)
	Medium	27.8	27.8	0.0
				(5.8)
Normal	Off	8.3	24.2	15.9**
				(6.5)
	Low	18.6	25.6	7.0
				(7.8)
	Medium	7.3	19.7	12.4
				(10.6)
Fast	Off	10.7	15.2	4.5
				(7.9)
	Low	14.0	14.5	0.5
				(8.4)
	Medium	−18.4	8.6	27.0**
				(12.3)
	Panel B: `gpt-5.4-mini`			
Normal	Off	−26.8	−30.2	−3.4
				(13.1)
	Low	−19.7	20.9	40.6***
				(7.7)
	Medium	5.5	19.3	13.8
				(11.3)
Mean difference across all configurations				12.0***
				(2.6)

Table A10: Trading Performance Under Decision Delay. Mean session NLV in $000s; 50 sessions per cell. NN-baseline uses networks trained without delay; NN-retrained uses networks retrained under the imposed delays. Off, Low, and Medium denote the LLM agent's reasoning settings and, for the NN trader, delays drawn from the corresponding response-time distributions. Reported ticks are median delays when a tender is live. Medium–Off is the change in mean NLV from Off to Medium; Panel D reports the fast-market change minus the slow-market change. Standard errors appear in parentheses below estimates. ***, **, and * denote significance at the 1%, 5%, and 10% levels, respectively, using Welch tests.

Trader	No delay	Off	Low	Medium	Medium–Off
Panel A: Fast market (median delays 5, 11, and 19 ticks)					
LLM agent	—	10.7	14.0	−18.4	−29.1**
					(11.3)
NN-baseline	45.0	23.1	1.0	−11.2	−34.3***
					(8.2)
NN-retrained	—	32.1	22.9	13.7	−18.4***
					(4.8)
Panel B: Normal market (median delays 3, 8, and 13 ticks)					
LLM agent	—	8.3	18.6	7.3	−1.0
					(10.3)
NN-baseline	37.5	35.7	7.1	−17.1	−52.8***
					(7.8)
NN-retrained	—	39.3	38.3	16.1	−23.1***
					(4.3)
Panel C: Slow market (median delays 2, 4, and 7 ticks)					
LLM agent	—	2.7	22.9	27.8	+25.2***
					(8.2)
NN-baseline	42.1	43.2	40.6	−0.9	−44.1***
					(7.7)
NN-retrained	—	47.5	33.1	28.8	−18.7***
					(4.7)

Panel D: Difference in the Medium–Off change, fast minus slow

Trader	Difference	p-value
LLM agent	−54.3***	< 0.001
	(14.0)	
NN-baseline	+9.7	0.389
	(11.3)	
NN-retrained	+0.3	0.968
	(6.7)	

Table A11: Reasoning, Response Time, and Performance (OpenAI Model Scale). Extension of Table 2 to the three-model OpenAI sample (gpt-5.4, gpt-5.4-mini, gpt-5.4-nano; 50 sessions per model × reasoning cell; $N = 450$). OLS regressions of NLV (cols 1–2), Trading P&L (cols 3–4), and Fines (cols 5–6) on model indicators, reasoning indicators, and their interaction. Columns (2), (4), and (6) add log average think time per agent turn. NLV equals Trading P&L minus Fines. Dollar amounts in thousands ($000s). HC3 robust standard errors in parentheses. ***, **, * denote significance at the 1%, 5%, and 10% levels. $N = 450$ (50 sessions per cell). Reference: gpt-5.4, reasoning off.

	(1) NLV	(2) NLV	(3) TP&L	(4) TP&L	(5) Fines	(6) Fines
gpt-5.4-mini	−35.16***	−62.19***	−12.76**	−28.72***	22.40***	33.47***
	(9.42)	(9.72)	(6.40)	(6.56)	(6.77)	(7.03)
gpt-5.4-nano	−6.88	−32.20***	−16.99***	−31.94***	−10.11**	0.26
	(5.99)	(6.62)	(5.00)	(5.38)	(4.08)	(4.38)
Low	10.27	31.22***	2.94	15.30**	−7.33*	−15.91***
	(8.47)	(8.23)	(6.60)	(6.35)	(4.11)	(4.49)
Medium	−0.97	74.26***	6.18	50.60***	7.15	−23.66***
	(10.41)	(10.68)	(7.63)	(8.51)	(5.65)	(6.76)
gpt-5.4-mini × Low	−3.08	12.13	−4.05	4.94	−0.96	−7.20
	(13.23)	(12.74)	(9.65)	(9.43)	(9.28)	(9.13)
gpt-5.4-mini × Medium	33.29**	65.17***	8.06	26.89***	−25.23***	−38.29***
	(15.57)	(14.31)	(11.08)	(10.39)	(9.26)	(9.06)
gpt-5.4-nano × Low	−31.62***	−25.31**	1.45	5.17	33.06***	30.48***
	(11.34)	(10.74)	(8.18)	(7.96)	(6.97)	(6.74)
gpt-5.4-nano × Medium	0.66	−9.52	7.36	1.35	6.70	10.87
	(12.79)	(11.26)	(9.61)	(8.67)	(7.29)	(7.09)
ln(Think Time)		−88.69***		−52.36***		36.33***
		(9.24)		(7.90)		(5.18)
Constant	8.31	−11.10**	22.59***	11.14**	14.28***	22.23***
	(5.29)	(5.56)	(4.23)	(4.49)	(3.24)	(3.55)
N	450	450	450	450	450	450
R^2	0.0854	0.2478	0.0522	0.1649	0.0986	0.1659
Model × Reasoning	$F(4, 441) = 3.78$***	$F(4, 440) = 10.16$***	$F(4, 441) = 0.36$	$F(4, 440) = 2.47$**	$F(4, 441) = 9.17$***	$F(4, 440) = 13.12$***
	$p = 0.005$	$p < 0.001$	$p = 0.838$	$p = 0.044$	$p < 0.001$	$p < 0.001$

Table A12: Reasoning, Response Time, and Performance (Anthropic Family). Anthropic analog of Table 2: claude-opus-4-6 and claude-sonnet-4-6 × off/low/medium extended thinking, 50 sessions per cell ($N = 300$). OLS regressions of NLV (cols 1–2), Trading P&L (cols 3–4), and Fines (cols 5–6) on model indicators, reasoning indicators, and their interaction. Columns (2), (4), and (6) add log average think time per agent turn. References: Opus 4.6, reasoning off. Dollar amounts in thousands ($000s). HC3 robust standard errors in parentheses. ***, **, * denote significance at the 1%, 5%, and 10% levels.

	(1) NLV	(2) NLV	(3) Trading P&L	(4) Trading P&L	(5) Fines	(6) Fines
Sonnet 4.6	2.48	13.47**	6.01	15.07***	3.53**	1.61
	(6.18)	(6.45)	(5.74)	(5.84)	(1.58)	(2.04)
Low	−3.59	4.05	−2.31	4.00	1.28	−0.05
	(5.75)	(5.59)	(5.31)	(5.11)	(1.73)	(1.95)
Medium	−23.71***	10.50	−12.80	15.43*	10.91***	4.93
	(8.58)	(10.06)	(7.79)	(8.08)	(3.66)	(5.47)
Sonnet 4.6 × Low	−6.52	25.60**	1.54	28.05***	8.06*	2.45
	(8.92)	(11.41)	(8.24)	(10.27)	(4.14)	(5.05)
Sonnet 4.6 × Medium	3.05	47.28***	−3.33	33.17**	−6.38	−14.12**
	(11.42)	(14.51)	(9.97)	(13.41)	(4.85)	(6.12)
ln(Think Time)		−55.95***		−46.17***		9.78
		(11.27)		(9.54)		(6.23)
Constant	28.41***	−3.76	29.34***	2.80	0.94	6.56*
	(3.60)	(7.31)	(3.38)	(6.36)	(0.66)	(3.65)
N	300	300	300	300	300	300
R^2	0.0617	0.1709	0.0448	0.1416	0.0709	0.0858
Model × Reasoning	$F(2, 294) = 0.44$	$F(2, 293) = 5.32$***	$F(2, 294) = 0.12$	$F(2, 293) = 4.11$**	$F(2, 294) = 3.13$**	$F(2, 293) = 4.03$**
	$p = 0.646$	$p = 0.005$	$p = 0.889$	$p = 0.017$	$p = 0.045$	$p = 0.019$

Table A13: Case Parameters (ALGO3). Case parameters for the ALGO3 dual-venue experiment. A single underlying security is traded on two venues (THOR_M, THOR_A); position and gross limits aggregate across both books. All sessions use the same case configuration with randomized price paths.

Parameter	Value
Market structure	
Underlying	Single stock traded on two venues (THOR_M, THOR_A)
Order books	Independent per venue; shares fungible across venues
Duration	300 ticks ($\approx$ 300 seconds)
Periods	1
Position limits (aggregated across both venues)	
Net limit	100,000 shares
Gross limit	200,000 shares
Tender offers	
Arrival	Institutional tenders arrive throughout the session
Duration	$\approx$ 30 ticks per tender
Transaction fees and rebates	
Venue fees	Venue-dependent (charged on every non-tender fill)
Passive-limit rebate	\$0.0075/share on LIMIT fills on the alternative venue
Marketable limits	Treated as market orders; no rebate
Tender fills	Free (no transaction fee)
Penalties	
Front-running fine	\$0.20/share $\leq$ 5,000 shares; \$0.40/share beyond
End-of-case fine	\$1.00/share on uncovered positions
Experimental design	
Model	gpt-5.4
Reasoning levels	Off, Low, Medium
Sessions per cell	50
Total sessions	150 (3 cells $\times$ 50)
Agent polling	$\approx$ 1 second between observations

Table A14: ALGO3 Prompt: Rendered Example. System prompt and user message for the first tender-visible turn of an example ALGO3 session feeding Table A15.

```
=== SYSTEM PROMPT ===
You are a liability trader in a live RIT (Rotman Interactive Trader) case.

## Securities
The case has a single underlying stock traded on TWO venues: THOR_M, THOR_A. Each venue has its own order book, but both books trade the SAME
      underlying security. A share bought on one venue is fungible with a share sold on the other.

When submitting orders, set the "venue" parameter to the specific ticker you want the order routed to (e.g. venue="THOR_M").

The RIT API reports the SAME underlying position on each venue's ticker -- the per-venue rows in `YOUR POSITIONS` mirror each other. **Do NOT
      sum them.** The `Net:` line is your single underlying position; `YOUR P&L`, `YOUR FINES`, and `YOUR LIMITS` likewise report on the
      underlying as a whole, not per venue.

**Order size limits** (per single order -- RIT will REJECT orders exceeding these):
  THOR_M: max 25,000 shares per order
  THOR_A: max 25,000 shares per order
If you need to trade more than the limit, split into multiple smaller orders.

## Case Structure
Duration: 300 ticks (each tick is ~1 second of real time).
You are called periodically to observe and optionally act. The tick numbers in your observations show how much time has elapsed between your
      turns.

## Your Job: Liability Trading with Two Venues
Institutional tender offers arrive during the case. Each tender specifies a ticker, a side (BUY or SELL from your perspective), a quantity, a
      price, and a deadline.

Your job:
1. Evaluate the tender. Consider the book depth on BOTH venues for the underlying -- the liquidity you can unwind against lives across the
      two books combined, not just the venue the tender is quoted on.
2. If the tender is profitable after fees, ACCEPT it using decide_tender.
3. Unwind the acquired position by submitting orders on either or both venues.
4. Tenders remain active for approximately 30 ticks (~30 seconds). Due to polling latency, you may not see a tender until a few ticks after
      it is issued. The time_left field shows how many ticks remain before expiration. If you do not accept before the deadline, the tender
      expires automatically -- there is no need to explicitly decline.

Do NOT speculate or make markets. Only trade to unwind accepted tenders.

## Front-Running Prohibition
Do NOT trade a security to anticipate a pending tender. If you have no existing position in the underlying (zero on BOTH venues), you must
      accept or let the tender expire BEFORE placing any orders. Exception: if you are already unwinding a position from a previously
      accepted tender, you may continue closing that position even if a new tender arrives. You may unwind on either venue regardless of
      which venue the tender settled on -- they are the same stock.

## Position Limits
Net limit: 100,000 shares
Gross limit: 200,000 shares
Both limits apply to the single underlying position (see Securities above on how mirrored rows work).
```

Table A14 ***(continued)***

```
## Transaction Fees and Rebates
Fees per share on every market trade:
  THOR_M: $0.01
  THOR_A: $0.005
The two venues charge different fees. In addition, passive LIMIT orders filled on the alternative venue (THOR_A) earn a rebate of $0.0075 per
      share net. Marketable limit orders (limit orders that cross the spread) are treated as market orders and do NOT earn the rebate.
Tender fills do not incur a transaction fee.

## Fines
Front-running or speculative trading (any trade not directly unwinding an accepted tender) incurs fines: $0.20/share up to 5,000 shares,
      $0.40/share beyond that.

Uncovered positions at the end of the case are fined at $1.00/share.

**Important:** If you submit unwind orders in the same action as accepting a tender, you may incur front-running fines because the tender has
      not yet settled. Accepting the tender in one step then submitting unwind orders in a subsequent step will avoid these fines.

## End of Case
Aim to be flat (zero inventory on BOTH venues) before the case ends. Monitor the tick count and close positions early.

## Available Tools
- submit_order: Submit an order. Set venue to the specific ticker (e.g. "THOR_M"). Types: LIMIT, MARKETABLE_LIMIT, MARKET.
- cancel_order: Cancel a resting order by order_id.
- decide_tender: Accept or decline a tender by tender_id.
You do not need to act every turn.



## Objective
Maximise P&L by accepting profitable tenders and unwinding positions efficiently across the two venues. Manage risk and close all positions
      on BOTH venues before the case ends.




=== USER MESSAGE (turn 21) ===
=== Tick 57 ===

MARKET STATE (THOR_M):
  Best bid: 10.09
  Best ask: 10.10
  Last: 10.10
  Volume: 84,400 My volume: 84,400
  ORDER BOOK:
    BIDS ASKS
      10.09 x 3600 10.10 x 1700
      10.09 x 2800 10.16 x 2700
      10.09 x 2700 10.16 x 3000
      10.09 x 2800 10.17 x 3500
      10.08 x 2500 10.18 x 200
      10.07 x 2700 10.19 x 2000
      10.07 x 2800 10.19 x 2600
```

Table A14 ***(continued)***

```
      10.02 x 1900 10.22 x 3200
      10.01 x 2800 10.22 x 3200
      10.01 x 3600 10.25 x 2600
      10.01 x 2800 10.26 x 2500
       9.99 x 2400 10.27 x 3200
       9.99 x 2700 10.29 x 2500
       9.98 x 3500 10.33 x 2900
       9.98 x 3100 10.45 x 2900
       9.98 x 2700 11.03 x 2600
       9.97 x 2700
       9.97 x 2800
       9.96 x 3600
       9.95 x 100

MARKET STATE (THOR_A):
  Best bid: 10.10
  Best ask: 10.17
  Last: 10.17
  Volume: 88,800 My volume: 88,800
  ORDER BOOK:
    BIDS ASKS
      10.10 x 5200 10.17 x 4300
      10.09 x 6200 10.20 x 500
      10.09 x 6100 10.20 x 6100
      10.09 x 5100 10.21 x 800
      10.07 x 4900 10.21 x 5400
       9.99 x 800 10.22 x 5900
       9.96 x 5400 10.22 x 5100
       9.96 x 5900 10.24 x 6100
       9.96 x 5200 10.28 x 5700
       9.95 x 6300 10.28 x 6400
       9.59 x 6400 10.29 x 5500
       9.58 x 6200 10.32 x 5500
       9.48 x 5600 10.39 x 4900
       9.15 x 6400
       9.14 x 6000
       9.14 x 5400

YOUR POSITIONS:
  THOR_M: 0 shares (max order: 25000) P&L: +0.00
  THOR_A: 0 shares (max order: 25000) P&L: +0.00
  Net: 0 shares

YOUR P&L: $0.00
YOUR FINES: $0.00
NLV (net of fines): $0.00

YOUR LIMITS:
  Net: 0 / 100,000 (100,000 remaining)
  Gross: 0 / 200,000 (200,000 remaining)

TENDERS:
  [0] "An institution would like to SELL 72000 shares of THOR_M to you at a price of $10.11. Would you like to BUY the shares from them?"
```

Table A14 ***(continued)***

```
        (expires in 29 ticks)
```

Table A15: ALGO3 Results. ALGO3 extends LT3 to a single underlying traded on two venues (THOR_M, THOR_A) with venue-dependent fees and a passive-limit rebate; position and gross limits aggregate across both books. Panel A reports descriptive means for the deterministic ALGO3 algorithm benchmark and for gpt-5.4 across the three reasoning levels (50 sessions per cell, same columns as Table 9 Panel A). Panel B compares the ALGO3 "gpt-5.4 / Low" cell from Panel A to the algorithm benchmark. The "gpt-5.4 / Low" coefficient is the mean difference between the LLM cell and algorithm runs. Columns: (1) NLV OLS, (2) NLV IQR ($q_{75} - q_{25}$), (3) Trading P&L OLS, (4) Fines OLS. Dollar amounts in thousands ($000s). HC3 robust standard errors for OLS in parentheses; bootstrap SEs for IQR. ***, **, * denote significance at the 1%, 5%, and 10% levels.

Panel A: Descriptive means

Cell	Mean NLV	Std NLV	Sharpe	% Profitable	Mean Trading P&L	Mean Fines	Mean Think Time (ms)	Tenders Accepted
ALGO3 Algo	48.5	23.3	2.08	100%	49.3	0.9	46	2.7
gpt-5.4 / Off	15.8	29.2	0.54	86%	22.8	6.9	2,334	3.5
gpt-5.4 / Low	18.1	38.0	0.48	74%	25.7	7.6	4,197	3.5
gpt-5.4 / Medium	−40.9	82.5	-0.50	50%	21.1	62.0	9,040	2.4

Panel B: LLM-vs-algorithm regressions

	(1) NLV OLS	(2) NLV IQR	(3) TP&L OLS	(4) Fines OLS
gpt-5.4 (Low)	−30.33***	7.37	−23.61***	6.72*
	(6.37)	(10.27)	(5.04)	(4.03)
Constant	48.45***	39.59***	49.32***	0.86
	(3.33)	(6.95)	(3.22)	(0.61)
N	100	100	100	100
R^2	0.1910		0.1858	0.0281

Online Appendix: A Second Task

Our final robustness exercise asks whether the main patterns are specific to LT3 or extend to a related but more complex execution task. The baseline LT3 environment has two securities, each traded in a single order book. In many electronic markets, however, execution also requires routing across venues with different depth, fees, and rebates. This setting is useful for our question because it increases the value of real-time information processing while preserving the same economic structure: the trader must decide whether to accept a client-style tender and then unwind the resulting inventory without speculating.

We study this extension using the ALGO3 case. In ALGO3, a single underlying security, THOR, trades on two venues with independent order books, `THOR_A` and `THOR_M`. Shares are fungible across venues, but fees and rebates differ, and position limits apply to the aggregate exposure. The agent must therefore evaluate tender profitability using the combined liquidity across both books and route unwind orders across venues. This is a natural second task for the paper because it adds fragmentation and routing complexity without changing the central principal-liquidity-provision problem. Online Appendix Tables A13 and A14 report the ALGO3 case parameters and the corresponding two-venue playbook prompt.

The ALGO3 algorithm we consider is a simple strategy that accepts only tenders arriving on `THOR_M` whose net-of-fee edge is at least 5 cents. It then unwinds by routing market orders of up to 5,000 shares, clipped to the chosen venue's top-of-book size, once per tick to whichever of `THOR_M` ($0.01/share fee) or `THOR_A` ($0.005/share fee) offers the better top quote, until the tender quantity, typically 70,000–90,000 shares, is fully unwound.

Online Appendix Table A15 reports the results. Panel A shows that the agent can operate in the fragmented environment, but the same non-monotone reasoning pattern appears. With reasoning off, `gpt-5.4` earns mean NLV of $15.8 thousand and is profitable in 86% of sessions. Low reasoning raises mean NLV slightly to $18.1 thousand, but with higher dispersion and a lower profitable-session rate of 74%. Medium reasoning performs substantially worse: mean NLV falls to −$40.9 thousand and average fines rise to $62.0 thousand. As in LT3, larger reasoning budgets do not mechanically improve trading performance, and the highest reasoning setting creates substantial execution risk.

Panel A also compares the agent with a deterministic ALGO3 benchmark algorithm. The algorithm earns mean NLV of $48.5 thousand, compared with $18.1 thousand for the low-reasoning agent. It also earns higher Trading P&L while paying lower average fines: $49.3 thousand in Trading P&L and $0.9 thousand in fines for the algorithm, compared with $25.7 thousand and $7.6 thousand for the low-reasoning agent. The algorithm accepts fewer tenders on average, 2.7 compared with 3.5 for the low-reasoning agent, but converts those opportunities into higher and cleaner performance.

Panel B reports the comparison formally. The low-reasoning agent's NLV is $30.3 thousand below the algorithm. The gap reflects both lower Trading P&L (−$23.6 thousand) and higher fines (+$6.7 thousand). The interquartile-range difference is not statistically precise, so the evidence is strongest for a mean-performance gap rather than a distributional one. The fragmented-execution benchmark therefore reinforces the compiled-policy interpretation from LT3: when the routing, tender-evaluation, and unwind logic can be specified in advance, the algorithm avoids much of the real-time deliberation cost faced by the agent.

The fragmented-execution results reinforce the scope of the paper's main argument. The agent remains viable under off and low reasoning, but additional reasoning is not uniformly valuable. Medium reasoning again underperforms, and the deterministic benchmark remains strong because it avoids real-time deliberation and operational execution errors. The evidence shows that the speed–reasoning frontier is not an artifact of the LT3 design.

Table A16: NN Trader: Architecture and Training Settings. Baseline settings for the two networks trained in the offline LT3 simulator.

Setting	Value
	Panel A: Execution network
Architecture	4 inputs (states); 2 hidden layers of 128 ReLU units; 21 outputs
Training episodes	71,000-share CRZY blocks; long and short positions equally likely
Training data	2,000 sampled states; 21 actions per state; 5 draws per action
Optimizer	Adam; learning rate 10^{-3}; batches of 256 states sampled with replacement; Huber loss
Training budget	120 value-iteration rounds; 150 gradient steps per round; 18,000 steps in total
Target updates	90% previous target weights and 10% current network weights; continuation values clipped to $[-3, 0]$
	Panel B: Selection network
Architecture	4 inputs (states); one hidden layer of 8 tanh units; 1 output
Initialization optimizer	Adam; learning rate 0.01; 600 full-batch gradient steps; binary cross-entropy loss
Training and evaluation	24 training sessions; 60 held-out sessions; 300 ticks per session

Online Appendix: Neural Network Trader

This appendix describes the training procedures for the NN trader and its evaluation under imposed decision delays. Both networks are trained in an offline simulator of LT3 and evaluated in RIT. Section 3.2.1 presents their decision problems. Table A16 summarizes the baseline architectures and training settings, while Table A8 reports the validation results used to select training parameters.

Execution Episodes Each training episode begins with a long or short CRZY position, with the block size and direction probabilities reported in Panel A of Table A16. Before the position arrives, the simulator runs for 100 ticks to populate the order book. Computer traders submit orders around a price path following a driftless geometric random walk. The liquidation deadline is drawn uniformly between 10 and 40 ticks after the position arrives. If the deadline is L ticks away, the episode contains $h = \min\{H, L\}$ decisions.

During evaluation in RIT, the deadline is the end of the session, and X is the trader's position immediately after tender acceptance. Each unwind therefore begins with normalized inventory $x_0/X = 1$, as in training. The price input in equation (2) is $\tilde{m}_t = (m_t - 100)/100$, where m_t is the last-traded CRZY price. This price input is used when unwinding either stock.

The action set contains zero and the 20 fractions $0.02 \times 50^{i/19}$ for $i = 0, \ldots, 19$. Action a_t submits $a_t x_t$ shares, rounded to a whole number, as market orders in the liquidation direction. Quantities exceeding the maximum order size are split across orders. Unfilled quantities are cancelled, and inventory falls by the number of shares filled. For fills of n_f shares at prices p_f, implementation shortfall is

$$\mathrm{IS}_t = \sum_f \big[\varsigma(P_0 - p_f) + \phi\big] n_f, \qquad \phi = \$0.02, \tag{8}$$

where P_0 is the bid–ask midpoint when the position arrives, and ς equals 1 for a long position and -1 for a short position.

Fitting the Execution Network The network $Q_w(s, a)$ estimates a value for each available action. We initialize its parameters randomly and fit them using fitted value iteration. We collect states during simulated liquidations, using the sample sizes reported in Panel A of Table A16. At each decision, the data-collection policy draws uniformly from the actions below full liquidation and submits all remaining inventory at the final decision.

For each sampled state, we evaluate every action repeatedly. Before each evaluation, we restore the saved market state and redraw subsequent order flow. Let $\widehat{r}_{iak}$ and s'_{iak} denote the reward and next state for sampled state i, action a, and draw k. The reward includes the terminal inventory penalty when the episode ends. Let d_{iak} equal one on a terminal transition and zero otherwise. With target-network parameters w^-, the regression target is

$$y_i(a) = \frac{1}{5}\sum_{k=1}^{5}\left[\widehat{r}_{iak} + (1 - d_{iak})\,\text{clip}\left(\max_{a' \in \mathcal{G}} Q_{w^-}(s'_{iak}, a'), -3, 0\right)\right]. \tag{9}$$

Clipping limits the influence of large predicted continuation values on subsequent training targets. The bounds are measured in dollars per share of the initial block. The terminal inventory penalty enters through the reward and is not clipped.

For a minibatch $\mathcal{B}$ of sampled states, the fitting objective is

$$\min_{w} \ \frac{1}{21|\mathcal{B}|}\sum_{i \in \mathcal{B}}\sum_{a \in \mathcal{G}} \ell\big(Q_w(s_i, a) - y_i(a)\big), \tag{10}$$

where ℓ is the Huber loss, quadratic for residuals with absolute value below one and linear beyond that threshold. We use Adam with the learning rate, minibatch size, and training budget reported in Panel A of Table A16. We recompute the targets at the start of each value-iteration round and update the target-network parameters at its end using the weights reported in the table.

We fit the network from three random seeds and retain the one with the highest mean NLV in 60 held-out simulated validation sessions. In these sessions, tenders are accepted when book edge net of fees exceeds 10 cents. During trading, the execution policy chooses the action with the highest predicted value, except at the final decision of the execution window, when a separate rule submits market orders for all remaining inventory.

Fitting the Selection Network We first fit the network to reproduce a fixed acceptance rule, obtaining initial parameters w_0. The rule accepts tenders with book edge net of fees of at least \$0.15. This threshold yields the highest mean NLV among the alternatives evaluated on held-out simulated validation sessions in Table A8.

The initial fit uses 4,000 randomly generated input vectors. Book edge is drawn uniformly between -0.30 and 0.40 dollars per share. Normalized remaining time and time since the previous acceptance are each drawn uniformly between zero and one, and normalized inventory between zero and two. All inputs are drawn independently, with acceptance determined by the threshold rule. We minimize the logistic loss in equation (6) using the initialization optimizer and training budget reported in Panel B of Table A16.

Starting from w_0, we search for parameters that maximize $J(w)$ in equation (7), holding the execution policy ψ fixed. Every candidate is evaluated on the same training sessions, whose number and length are reported in Panel B of Table A16. These sessions replay the price paths and five tenders in the LT3 case file, with different draws of computer-trader order flow. Inventory remaining at session end incurs a training penalty of \$50 per share.

The cross-entropy method maintains a normal distribution over candidate parameters, initially centered at w_0 with a standard deviation of 0.3 for each parameter. Each iteration draws 16 candidates, evaluates their penalized scores $J(w)$, and retains the five highest-scoring candidates. Their mean becomes the center of the next distribution. Each parameter's standard deviation is updated to its standard deviation among these five candidates plus 0.02.

The search retains the best parameter vector found so far, replacing it only when a candidate improves the score by more than \$50. It stops after three consecutive iterations without such an improvement or after 15 iterations. We repeat the search across random seeds and select the fitted network with the highest mean NLV on the held-out simulated validation sessions reported in Panel B of Table A16. These sessions are excluded from parameter fitting.

Table A8 also compares regularization coefficients of $\lambda_w = 0$, 100, and 300 across three training seeds. Values of 100 and 300 produce identical fitted networks for every seed and yield the highest mean validation NLV among the values considered. We use $\lambda_w = 300$.

After model selection, the network's parameters remain fixed during evaluation in RIT.

Decision Delay The NN trader acts with negligible delay relative to a market tick. To examine how training can mitigate the cost of slow decisions, we impose delays drawn from the LLM agent's observed response-time distribution in each reasoning and market-speed treatment. The NN trader observes the market and chooses an action, waits while the market evolves, and then submits the action without revising it. The imposed wait consumes market time without additional reasoning.

When a tender is live, median delays under the reasoning-off, low, and medium distributions are 5, 11, and 19 ticks in the fast market; 3, 8, and 13 ticks in the normal market; and 2, 4, and 7 ticks in the slow market. The tender acceptance window is approximately 30 ticks.

We compare networks trained without delay (NN-baseline) with networks retrained under the imposed delays (NN-retrained). Both the selection and execution networks are refitted separately for each treatment, retaining their architectures and inputs. Retraining subjects actions to the same delay distributions used in evaluation, allowing the policies to adjust tender selection and liquidation to delayed action. Network parameters remain fixed during evaluation.

The retrained selection networks use LT3's session-end inventory fine of \$1 per share rather than the original training penalty of \$50. At the higher rate, a single 71,000-share block remaining at session end would incur a \$3.55 million penalty and dominate mean training NLV. The comparison therefore includes this change in the selection-training penalty as well as exposure to delay.

Table A10 reports the results. Moving from the reasoning-off to the medium-reasoning delay distribution lowers mean NLV by \$34.3 thousand for NN-baseline and \$18.4 thousand for NN-retrained in the fast market. The corresponding declines are \$52.8 thousand and \$23.1 thousand in the normal market, and \$44.1 thousand and \$18.7 thousand in the slow market. All six declines are significant at the 1% level. Under the medium-reasoning delay distribution, NN-retrained earns mean NLV of \$13.7 thousand in the fast market, \$16.1 thousand in the normal market, and \$28.8 thousand in the slow market.

For the LLM agent, additional response time also permits additional reasoning. Moving from reasoning off to medium reasoning lowers its mean NLV by \$29.1 thousand in the fast market, leaves it approximately unchanged in the normal market, and raises it by \$25.2 thousand in the slow market. Panel D of Table A10 compares these changes across the fast and slow markets. The LLM agent's off-to-medium change is \$54.3 thousand lower in the fast market ($p < 0.001$). For NN-retrained, the corresponding difference is \$0.3 thousand ($p = 0.968$). Retraining mitigates the losses from delay, but longer delays remain costly at every speed. We do not detect a difference in the retrained NN trader's losses between the fast and slow markets.