

REIMAGINING AGENT-BASED MODELING WITH LARGE LANGUAGE MODEL AGENTS VIA SHACHI

So Kuroki¹, Yingtao Tian¹, Kou Misaki¹, Takashi Ikegami², Takuya Akiba¹, Yujin Tang¹

¹Sakana AI ²The University of Tokyo

ABSTRACT

The study of emergent behaviors in large language model (LLM)-driven multi-agent systems is a critical research challenge, yet progress is limited by a lack of principled methodologies for controlled experimentation. To address this, we introduce Shachi, a formal methodology and modular framework that decomposes an agent’s policy into core cognitive components: Configuration for intrinsic traits, Memory for contextual persistence, and Tools for expanded capabilities, all orchestrated by an LLM reasoning engine. This principled architecture moves beyond brittle, ad-hoc agent designs and enables the systematic analysis of how specific architectural choices influence collective behavior. We validate our methodology on a comprehensive 10-task benchmark and demonstrate its power through novel scientific inquiries. Critically, we establish the external validity of our approach by modeling a real-world U.S. tariff shock, showing that agent behaviors align with observed market reactions only when their cognitive architecture is appropriately configured with memory and tools. Our work provides a rigorous, open-source foundation for building and evaluating LLM agents, aimed at fostering more cumulative and scientifically grounded research. Code: <https://github.com/SakanaAI/shachi>

1 INTRODUCTION

Agent-based modeling (ABM) is a widely used methodology for simulating complex systems through the interactions of autonomous agents, and has been applied to fields such as economics, sociology, and political science (Gilbert & Terna, 2000; Gilbert, 2019; Davidsson, 2002). By enabling researchers to explore emergent phenomena and counterfactual scenarios, ABM offers a powerful tool for both theory-building and policy experiments. However, traditional ABMs often rely on handcrafted rules and heuristics, which can limit realism and interpretability.

Recently, large language models (LLMs) have demonstrated impressive capabilities across a wide range of reasoning, planning, and decision-making tasks (Guo et al., 2025; Kojima et al., 2022; Achiam et al., 2023), leading to a surge of interest in deploying them as agents (Park et al., 2023; Gao et al., 2024; Wang et al., 2024; Anthropic, 2024; Surapaneni et al., 2025). This momentum has naturally extended into ABM (Li et al., 2024; Yang et al., 2024; Wu et al., 2023; Manning et al., 2024), raising hopes that LLMs could alleviate the brittleness and manual effort of traditional agent behavioral designs. However, this rapid adoption has outpaced the development of rigorous methodology. Current approaches often rely on ad-hoc designs, creating a fragmented landscape where results are difficult to reproduce, compare, or build upon. This lack of a standardized framework hinders the systematic study of emergent behaviors and undermines the scientific potential of LLM-based ABM. Consequently, a principled and unified approach is essential for fostering cumulative and reliable research in this domain.

This methodological fragmentation creates three primary obstacles for the field. First, agent-environment interactions are defined by incompatible, bespoke interfaces, making it nearly impossible to transfer agents between studies or systematically compare their performance. Second, the internal architectures of agents are themselves scattered and inconsistent; capabilities like memory and tool use are implemented as one-off features rather than standardized, modular components, preventing principled analysis of their impact. Finally, this focus on isolated, synthetic tasks has limited the

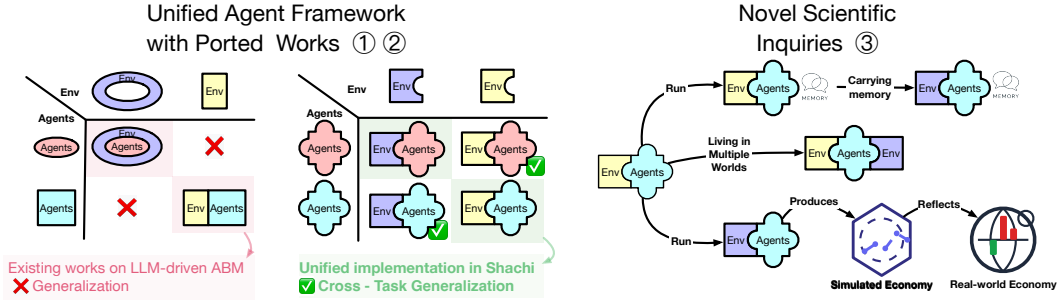


Figure 1: **Unifying LLM-based ABM Research with Shachi.** Shachi is a methodology and accompanying framework with a benchmark suite that accelerates social science research through LLM-based agents in ABM. Shachi facilitates research in this space by providing ① A unified agent architecture that standardizes core components (LLM, memory, tools, configuration) for modular and reproducible design; ② Cross-task generalization that allows extensive evaluation of different agent designs; and ③ Novel scientific inquiries previously infeasible, such as agents conducting memory transfer, living across multiple worlds, and demonstrating external validity through simulation of real-world economic events.

validation of LLM agents against complex real-world phenomena, casting doubt on their external validity and scientific utility.

To address these challenges, we propose Shachi¹, a formal methodology for LLM-based ABM, instantiated in a modular open-source framework. Shachi introduces a standardized agent architecture built around four key components: an LLM, a configuration module for shaping an agent’s intrinsic identity and behavioral policies (e.g., via system prompts or by adapting the model’s weights), tools that extend its capabilities, and memory for maintaining context and continuity. This modularity allows researchers to isolate and systematically investigate the architectural drivers of emergent behavior, moving beyond brittle, prompt-driven agent designs. As a result, Shachi enables rapid experimentation, encourages the reuse of existing components, and lowers the barrier for building complex, human-like agents in simulation. See Figure 1 for an overview of the proposed framework.

We validate our methodology on a suite of ten benchmark tasks, using them as a foundation for novel exploratory studies that demonstrate Shachi’s flexibility. By recomposing the framework’s core modules, we can go beyond simple replication. For instance, we investigate how agent biases evolve when **carrying memory to a new life** in a different environment, or how agents behave when **living in two worlds** by participating in both economic and social simulations simultaneously. Crucially, this modular approach allows us to establish the external validity of our framework by modeling a complex, real-world scenario. That is, we simulate nuanced market reactions to a **U.S. tariff shock**, where agent behavior aligns with observed economic events.

Our key contributions are summarized below:

- **A Structured Methodology for Agent-Based Modeling:** We introduce Shachi, a methodology that provides a structured decomposition of an agent’s policy into core cognitive components (Configs, Memory, Tools) and a reasoning engine (LLM). This principled architecture enables the systematic analysis of how design choices influence emergent behavior.
- **A Multi-Level Benchmark Suite for Validation:** We provide a 10-task benchmark suite, structured across three levels of social complexity (single-agent, non-communicative multi-agent, and communicative multi-agent). This suite serves as a standardized testbed for reproducing prior work and rigorously evaluating new agent designs.
- **Enabling Novel Scientific Inquiries:** Shachi enables exploratory studies previously infeasible with ad-hoc approaches. We demonstrate its power through novel experiments, including agents carrying memory to new environments and living in multiple worlds, and by establishing the external validity of LLM agents through a simulation of real-world economic events.

¹The name “Shachi” (鯨), meaning orca in Japanese, reflects the system’s goals: intelligent, social, and adaptive agents operating in complex environments, much like orcas navigating the ocean in coordinated pods.

2 RELATED WORKS

ABM is a computational approach to simulate interactions among autonomous agents within complex systems, enabling the study of emergent behaviors and social dynamics (Gilbert, 2019). From a computer science perspective, ABM integrates agent-based computing, social sciences, and computer simulation, fostering cross-disciplinary research (Davidsson, 2002). In the social sciences, ABM serves as a “third way” of research, complementing argumentation and formalization by enabling the modeling of complex processes and emergent phenomena (Gilbert & Terna, 2000). In economics, ABM has evolved into agent-based computational economics, modeling dynamic economic systems and revealing insights into market behaviors like strategic interactions and collective learning (Tesfatsion, 2006; Tesfatsion & Judd, 2006). Enhancements in agent design through behavioral economics and empirical data integration have made ABM simulations more realistic and applicable to complex social and economic systems (Steinbacher et al., 2021). With its wide applications, ABM remains a promising research area and motivates our work. Refer to Appendix E.1 for more works on ABM.

Recently, integrating LLMs into ABM has emerged as a promising direction to enhance the realism and adaptability of agents by improving environmental perception, human alignment, action generation, and evaluation (Gao et al., 2024; Nisioti et al., 2024). For example, PsychoBench (Huang et al., 2023) evaluates psychological traits; Generative agents (Park et al., 2023) simulate interactive social behaviors; and OASIS (Yang et al., 2024), Sotopia (Zhou et al., 2024), and EconAgent (Li et al., 2024) extend these ideas to large-scale simulations and economic and social reasoning. In the agent-based market domain, models like StockAgent (Zhang et al., 2024) and AuctionArena (Chen et al., 2023) test strategic and adaptive decision-making. See Appendix E.2 for more related works. Shachi’s contribution is distinct from both general-purpose agent toolkits and ML engineering frameworks. General frameworks like AutoGen (Wu et al., 2024), Concordia (Vezhnevets et al., 2023), and EDSSL (Expected Parrot, 2023) are designed for conversational task automation or game-master-led interactions, not the specific requirements of reproducible social simulation. In contrast, frameworks like MLE-Dojo (Qiang et al., 2025) focus on LLM training and engineering workflows, rather than the simulation and analysis of emergent social phenomena. Shachi bridges this gap with an architecture centered on a standardized agent-environment interface and environment-mediated communication, uniquely tailored for the reproducible and systematic study of these dynamics.

3 SHACHI: A METHODOLOGY FOR MODULAR AGENT-BASED MODELING

To enable principled experimentation, we introduce Shachi, a methodology that formalizes the design and evaluation of LLM-based agents in ABM. Shachi is built on the core principle of decoupling the agent’s internal architecture from the environment, which we achieve through a standardized interface and a modular component-based design for the agent itself. Figure 2 presents an overview.

3.1 DECOUPLING AGENT FROM ENVIRONMENT

The design of the agent-environment interface is central to Shachi. To ensure agents are portable and experimental results are reproducible, we introduce a principled abstraction layer that decouples an agent’s internal cognitive architecture from the external environment it inhabits. This design governs both agent-environment interaction and inter-agent communication.

Our interface takes inspiration from standard reinforcement learning formalisms like OpenAI Gym (Brockman et al., 2016). The simulation proceeds in discrete time steps, guided by the environment’s STEP() and RESET() methods (see Appendix D.1). More formally, we model the simulation as a partially observable multi-agent decision process. At each time step t , each agent i possesses an internal state S_t^i (i.e., its memory). The environment, E , with a global state S_t^E , emits a tailored observation $O_t^i = f(S_t^E, i)$ to each agent. This observation contains all information required for decision-making, such as available tools and the expected response format. The agent’s cognitive architecture then computes an action A_t^i according to its policy π , which is conditioned by its intrinsic configuration C^i and internal state: $A_t^i \sim \pi(\cdot \mid O_t^i, S_t^i; C^i)$. The environment collects the set of all actions $\mathbf{A}_t = \{A_t^1, \dots, A_t^N\}$ and updates its state via a transition function, $S_{t+1}^E = T(S_t^E, \mathbf{A}_t)$. This formal separation of the agent’s policy π from the environment’s transition function T is what enables agents built in Shachi to be evaluated across diverse environments in a zero-shot manner.

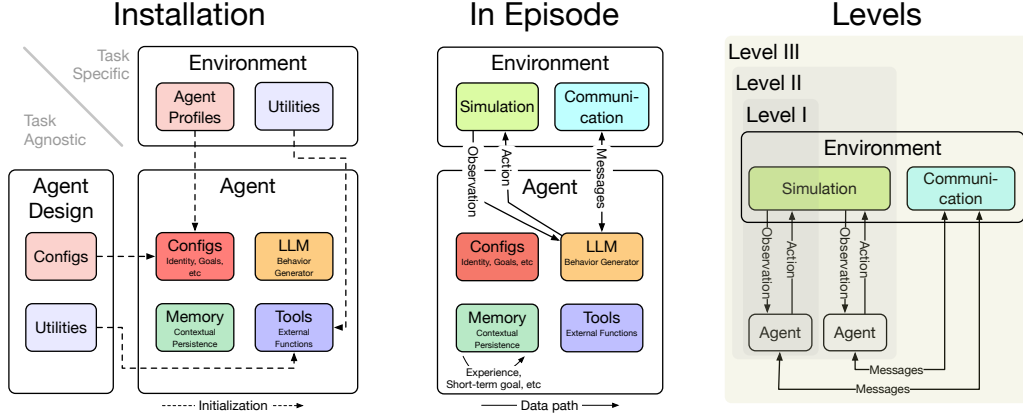


Figure 2: **Shachi Methodology Overview.** The figure illustrates the core principles of our methodology. **Left:** Agent instantiation decouples task-specific environment settings (e.g., agent profiles) from task-agnostic agent design. This ensures agent modularity and portability. **Middle:** The agent’s policy π is realized through a cognitive architecture of four components (Configs, Memory, Tools, and LLM). The policy π processes an observation O_t^i to generate an action A_t^i . The environment mediates both agent-environment interactions and inter-agent communications via structured messages and facilitates simulation. Agents receive immediate feedback via tool interfaces. **Right:** The methodology includes a structured three-level benchmark, enabling systematic analysis of agent behavior across contexts of increasing social complexity.

The distinction between actions and tool calls is a key methodological contribution. Unlike traditional RL environments with fixed action spaces, Shachi supports expressive, high-dimensional outputs. With this context, we define (1) an action to be an output from the policy π that is passed to the environment’s transition function T , thereby advancing the simulation’s state to $t + 1$; (2) a tool call to be an intra-step cognitive or information-gathering operation, which provides immediate feedback that informs the policy’s deliberation before an action is finalized, without advancing the global clock.

Inter-agent communication is also mediated through this robust interface. Instead of allowing direct function calls between agents, which would create complex dependencies, interactions are handled by the environment. For example, an environment may expose a function that allows one agent to send a message to another (this is the case in the OASIS task in our benchmark suite, where an agent talks to others via an environment-specific `COMMENT_TO()` function). These interactions are then embedded into the observation space and used to simulate realistic social dynamics such as broadcasting, targeted messaging, or asynchronous communication (see Appendix D.3). To handle the practical challenge of ensuring LLMs produce valid outputs, we leverage modern API features for structured data and employ robust parsing strategies (see Appendix D.4).

3.2 COMPOSING THE COGNITIVE ARCHITECTURE

In Shachi, an agent’s policy π is not a monolithic black box. Instead, we propose a modular cognitive architecture composed of four interacting components. This decomposition allows for the systematic study of how different cognitive faculties contribute to an agent’s behavior and the emergent system-level dynamics. These components are inspired by principles in cognitive science and are designed for extensibility and rigorous experimentation.

LLM serves as the core reasoning engine that powers the policy π . Just as human cognition is shaped by language-based reasoning and internal narration (Vygotsky, 2012), LLMs simulate this process by converting observations into natural language or structured responses. In Shachi, the agent constructs a prompt from its observation and forwards it to the LLM, which returns an action or message. We support flexible backend substitution, allowing calls to both proprietary APIs and open-source models. Each LLM call is invoked asynchronously, enabling efficient simulation through parallelism.

Configs component defines an agent’s identity, constraints, and tendencies. This is akin to an agent’s static identity. This module defines the conditioning variable C^i in the agent’s policy $\pi(\cdot \mid \cdot; C^i)$. Much like psychological traits or roles in human society (McCrae & Costa Jr, 1997), configurations

govern how an agent interprets tasks and responds to stimuli. In Shachi, this component can be implemented via prompting strategies or dynamic LoRA (Hu et al., 2022) module loading, which influence LLM decision thresholds or tool access policies. This abstraction allows researchers to simulate diverse agent archetypes or heterogeneous populations with varying roles or incentives.

Memory constitutes the agent’s dynamic internal state S_t^i , enabling longitudinal coherence and history-contingent behavior. Unlike the Configs above, but akin to human working and episodic memory, Shachi’s memory module allows agents to retrieve relevant past interactions and incorporate them into current decisions. This is critical for simulating agents with evolving goals, personalities, or bonds (Park et al., 2023). Memory implementations in Shachi are abstracted to support strategies from simple buffer-based recall to advanced retrieval-augmented or embedding-based approaches. Researchers can modify the memory capacity, retrieval method, or make memory learnable.

Tools component provides agents with access to external functions or services, reflecting how humans use tools and technologies to augment cognitive tasks (Norman, 2014). Cognitive science considers tool use a hallmark of intelligent behavior, allowing agents to transcend their intrinsic limitations. In Shachi, tools are provided through the observation O_t^i . They are provided by environments (task-specific) or from researchers’ toolbox (task-agnostic), and are defined with a name, a schema describing their parameters, and a callable function that executes the desired operation. Agents autonomously decide whether and when to use these tools, along with which arguments to supply, making tool use part of their decision-making process. This component enables researchers to introduce new tools to simulate domain-specific capabilities, context-sensitive environment interactions, or even social interactions.

3.3 ANALYZING SYSTEMATICALLY ACROSS SOCIAL COMPLEXITY

A core component of our methodology is a standardized testbed for evaluating agent architectures. To this end, we present a benchmark suite of ten tasks, adapted from prior work, and structured into three levels of increasing social complexity. This tiered structure is a deliberate design choice that allows researchers to isolate cognitive variables and systematically analyze agent behavior, from individual rationality to complex social dynamics. Details of each task and setup are in Appendices A and B.

Level I: Single-Agent Baselines. Tasks at this level feature a single agent in a controlled environment. They serve as the primary setting for calibrating and validating the core cognitive components of an agent’s architecture. Here, researchers can systematically probe how different Configs (e.g., personas), Memory implementations, or available Tools affect individual reasoning and behavior, free from the confounding variables of social interaction.

Level II: Non-Communicative Multi-Agent Dynamics. This level introduces multiple agents into a shared environment where they interact only indirectly through their impact on the environment state S_t^E . These settings are designed to test an agent’s ability to engage in strategic reasoning based purely on observations. They allow for the study of emergent phenomena like implicit coordination or competition, where agents must infer the strategies of others.

Level III: Communicative Multi-Agent Systems. The final level involves rich, direct communication, mediated through the environment. These tasks evaluate the entire socio-cognitive architecture, testing the interplay between language use, memory, and strategic action. They are essential for simulating sophisticated social phenomena such as negotiation and coalition formation.

Together, this structured benchmark provides a concrete pathway for conducting reproducible research. By leveraging the modularity of Shachi, researchers can now systematically compare how a specific architectural choice (e.g., adding a long-term memory module) impacts agent performance and behavior across all three levels of social complexity.

4 EXPERIMENTS

Our experiments demonstrate the utility of the Shachi methodology in two parts. First, we perform foundational validation, confirming the framework’s reproducibility by replicating prior work and its comparability through cross-task generalization studies. Having established this baseline, we then showcase how Shachi enables novel scientific inquiries, culminating in a simulation of a real-world U.S. tariff shock that establishes the methodology’s external validity.

4.1 FOUNDATIONAL VALIDATION

Experimental Setup For reproducibility, we reimplemented agents from eight benchmark tasks using our modular components to match their original implementation. For comparability, we conducted cross-task generalization studies, evaluating how these agents, each with a different cognitive architecture (e.g., with or without tools), perform in unseen environments. All detailed experimental settings are provided in Appendices C.2 and C.3.

Reproducibility Our reproducibility results show that Shachi faithfully replicates prior findings. As shown in Table 1, our implementation achieves significantly lower Mean Absolute Error (MAE) compared to baselines across all tasks, confirming that our components accurately capture the behavior of the original, bespoke models. We also qualitatively verify that our framework reproduces fine-grained system dynamics, such as temporal stock price evolution and auction bidding patterns (visualizations are in Appendix C.2).

Table 1: **Reproduction Results.** We report the mean absolute error (MAE) for the ported tasks. Shachi consistently achieves lower errors when compared with the baselines.

	PsychoBench	CoMPosT	CognitiveBiases	EmotionBench
Baseline	1.96	0.23	0.24	13.82
Shachi (Ours)	0.80	0.06	0.04	3.37
	EmergentAnalogies	StockAgent	AuctionArena	Sotopia
Baseline	0.64	9.07	10.49	3.17
Shachi (Ours)	0.05	2.63	2.22	0.95

Comparability The cross-task generalization results in Table 2 highlight the critical role of the cognitive architecture. While simpler tasks can be solved by minimal agents, performance in complex environments is highly dependent on the available components. For instance, an agent equipped with a full suite of components including Tools (e.g., StockAgent) generalizes effectively to other complex tasks. Conversely, agents lacking necessary components, such as Tools and Memory modules, fail when transferred to environments that require them. This demonstrates that Shachi’s modularity is crucial for systematically studying and building agents with robust, generalizable capabilities.

Table 2: **Cross-Task Agent Generalization.** Scores in each column are normalized against the one on the diagonal. Agents with all the components (i.e., StockAgent) maintain stable performance when transferred to other tasks.

	EmergentAnalogies	StockAgent	AuctionArena	Sotopia
EmergentAnalogies	1.00	1.08	0.62	1.01
StockAgent (config, mem, tool)	1.01	1.00	0.99	1.00
AuctionArena (config, mem)	1.00	0.93	1.00	0.99
Sotopia (mem)	1.00	0.93	0.92	1.00

4.2 SCIENTIFIC INQUIRIES

The true power of Shachi lies in the new questions it allows us to ask. In the following, we move beyond validation to demonstrate Shachi’s utility for scientific exploration, focusing on compositional behaviors and establishing the framework’s ability to model complex, real-world events.

4.2.1 CARRYING MEMORY TO THE NEXT LIFE

Experimental Setup In these studies, we used GPT-4o-mini as our LLMs and transferred agents from OASIS and EconAgent tasks without clearing their memories (i.e., their stream memories contain the observations and experiences from these tasks) to the CognitiveBiases task. The experiments were conducted three times to gauge statistical significance in our findings.

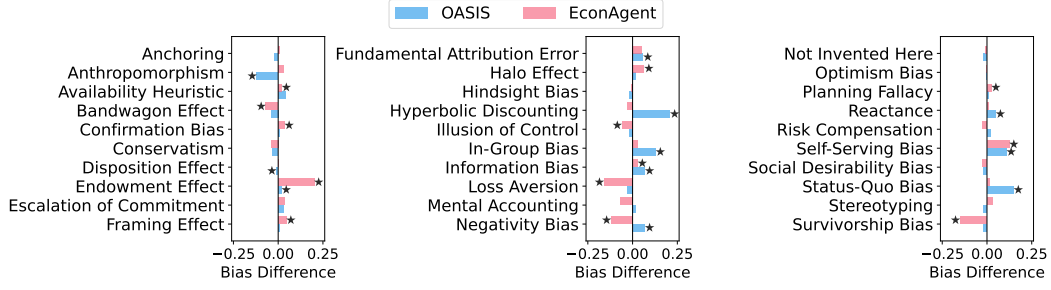


Figure 3: **Memory-transfer-induced Differences in the CognitiveBiases Task.** For each bias, the difference is calculated as the score with carry-over memory minus that with fresh memory. Statistically significant differences are indicated with star-shaped markers (paired t -test, $p < 0.01$).

Experimental Results Figure 3 shows that carry-over memories alter performance in the Cognitive-Biases task. Notably, memories from OASIS and EconAgent lead to distinct bias shifts. Among the largest changes, in OASIS, short-term reactions and community pressure could amplify *Hyperbolic Discounting* and *In-Group Bias*. Agents learned to heavily prioritize immediate feedback (raising hyperbolic discounting) and reinforce group identities (raising in-group bias). In EconAgent, repeated interactions and asset ownership may heighten the *Endowment Effect* while diminishing *Loss Aversion* and *Survivorship Bias*. Because agents regularly obtain experience with actual gains and losses, they become more attached to assets they already hold (raising endowment effect) and become less sensitive to losses and overly optimistic success rates (lowering loss aversion and survivorship bias).

4.2.2 LIVING IN MULTIPLE WORLDS

Experimental Setup We introduce a set of shared agents that operate across two distinct environments: StockAgent and OASIS. Unlike typical settings where agents act in only one environment, agents here cycle between both. In each cycle, agents first observe the StockAgent environment, make trading decisions, and apply their actions. Then, the group moves to OASIS, where agents observe the social media, respond, and act. This loop repeats, with each agent carrying its internal states across environments, allowing knowledge or strategies learned in one context to influence behavior in the other. StockAgent settings follow those in Section 4.1, with stock A representing a 10-year chemical stock and stock B representing a 3-year tech stock. The OASIS environment is described in Appendix B.10, where the main topic is Amazon’s newly established physical stores. We report results averaged over three independent trials.

Experimental Results We found that allowing agents to participate in multiple environments led to emergent behaviors that reflect cross-domain influence. Figure 4 shows that, with OASIS present, stock prices rise less than in the StockAgent-only setting. This is surprising to us, as we expected that introducing the Amazon topic would cause agents to buy more of the tech-related stock (stock B), leading to a surge in its price. On the other hand, Table 3 gives more detailed information under the hood: (1) Introducing social media increased market activity, as evidenced by the increased volume in both stocks; (2) The introduced Amazon topic caused agents to show more willingness to buy the tech stock and to become reluctant to sell, which is supported by the increase in #Buys and the decrease in #Sells for stock B; (3) In contrast, this topic increased both the willingness to buy and sell stock A. This information is more in line with our intuition than with the price movements.

In terms of agent behavioral changes in OASIS, we observed that some agents began tweeting about Amazon stock, blending financial observations with social commentary. Other agents responded with comments, echoing and endorsing the post, suggesting that economic actions in one environment can organically propagate into social discourse in another. The following text box 1 shows the agent logs discussed, where the agents’ posts and comments are labeled in blue and red.

Our results reveal that while agent-level behaviors align with intuitive financial logic, system-level outcomes do not necessarily follow, highlighting the importance of empirical simulation in social science research. At the same time, our exploratory study demonstrates Shachi’s capacity for rich, layered simulations by supporting agents that live across multiple interconnected worlds.

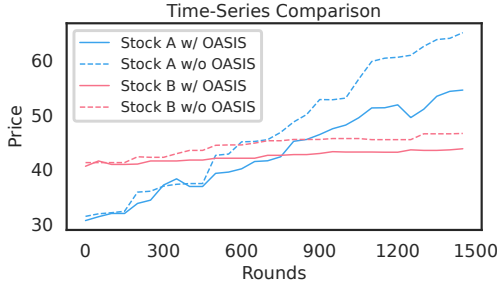


Figure 4: Comparison of Price Movements.

Table 3: **Change in Key Indicators.** Volume refers to the number of matched orders, while #Buys/#Sells are the numbers of submitted buy/sell orders. Changes are reported as percentage changes of treatment (w/ OASIS) relative to control (w/o OASIS).

Indicator	% Δ (Stock A)	% Δ (Stock B)
Volume	+10.0%	+20.0%
#Buys	+8.4%	+6.1%
#Sells	+6.6%	-8.5%

Text box 1: Extracted behaviors in OASIS when agents live in both StockAgent and OASIS

- **create_post**: user_id=14, current_time=3, action=create_post, info={'content': "Agent plan: I find it quite fascinating as a stock trader to see Amazon making a move into physical retail. This could potentially impact their stock, especially if they successfully blend the convenience of online shopping with the tactile experience of a physical store. It's also worth noting how this decision might influence other online retailers to consider similar ventures, possibly reshaping the retail landscape. I'll be keeping a close eye on how this development unfolds and the market's response to it. 🛒📈 #Amazon #RetailInnovation.", 'post_id': 16}
- **create_comment**: user_id=1, current_time=0, action=create_comment, info={'content': "Interesting to see Amazon branching out into physical retail spaces! It's a bold move considering their strong online presence. As a stock trader, I'm curious about how this strategy will affect Amazon's stock and their competition. What are your thoughts on this step?", 'comment_id': 1}

4.2.3 ESTABLISHING EXTERNAL VALIDITY: SIMULATING A U.S. TARIFF SHOCK

Experimental Setup We use StockAgent as our testbed, and simulate a 5-day trading period (April 1-5, 2025) surrounding a tariff shock. We conduct a cumulative ablation study across four settings:

1. Base: Agents from the standard StockAgent task with no extra information.
2. Base + Config: Agents are exposed to a pre-announcement news headline about imminent tariffs, added to their configuration prompt.
3. Base + Config + Memory: Agents are additionally equipped with memory containing a summary of academic research on how tariffs negatively impact markets (Amiti et al., 2021).
4. Base + Config + Memory + Tool: Agents are finally given access to a news-retrieval tool that provides daily updates on the escalating trade tensions.

This design isolates the impact of each component, from a simple awareness of the event (Config) to possessing deep knowledge (Memory) and receiving real-time information (Tools). For each setting, we run 5 trials and report the mean. Details of prompts and data sources are in Appendix C.4.

Simulation Results and Analysis We report the average ratio of buy to sell orders across days in the order book to gauge the agents' enthusiasm for the market. The results are summarized in Table 4. Comparing the 2nd and 3rd columns, we can

Table 4: **Buy-to-Sell Ratios (Cols. 2–3) and Changes (Cols. 4–5).**

Setting	Stock A	Stock B	Δ Stock A	Δ Stock B
#1	0.99	0.73	-	-
#2	0.51	0.45	-0.48 (w.r.t #1)	-0.28 (w.r.t #1)
#3	0.62	0.59	+0.11 (w.r.t #2)	+0.14 (w.r.t #2)
#4	0.44	0.55	-0.18 (w.r.t #3)	-0.04 (w.r.t #3)

see that the agents prefer Stock A to B in settings #1 to #3, but this preference flipped in setting #4. The last two columns reveal more insights: (i) The 2nd row shows that changing the agents'

configuration so that they become aware of the tariff policy caused the agents to become more likely to sell. (ii) On the other hand, the 3rd row indicates that when agents are given academic knowledge, they become less reactive to the raw news, softening their bearish behavior. (iii) The last row is the most interesting to us. It shows that having access to daily news altered the agents’ preference again, with Stock B showing a smaller drop than Stock A, creating a large difference between them.

Comparison with Real-World Data The outcome of our most complete simulation (#4) strongly aligns with real-world events. To establish the external validity of our method, we compare the simulated company performance with actual market data. To identify realistic counterparts for Stocks A and B, we first used ChatGPT to suggest companies matching their profiles (see Appendix C.4 for our prompt) and then manually verified the results. While these companies are not exact matches, they closely align with the stated profiles. The real-world results are supportive of our simulation results. The results are in Table 5, notice that since 4/5 is a weekend, 4/7 therefore corresponds to 4/5 in our simulation. Specifically, as shown in the return column of the real-world tables below, both Stock A and Stock B prices declined but Stock B experienced a smaller drop.

The results from this experiment provide three key takeaways. First, it demonstrates Shachi’s flexibility, as each cognitive component (Config, Memory, and Tools) was used to systematically introduce new layers of information and influence agent behavior. Second, the resulting behaviors were insightful, evolving from a simple sell-off to a sophisticated market reaction that aligned with real-world events. Finally, this progression offers a powerful analogy for human cognition: setting #2 mirrors a person hearing the news without much economic literacy, following the panicked herd and exacerbating market meltdown; setting #3 resembles someone with in-depth economics knowledge, observing more before acting; setting #4 represents professionals who proactively seek more information and act swiftly after confirmation.

Table 5: **Real-World Stock Changes.** Prices and returns of stocks matching the profile of A/B are shown in the top/bottom.

Symbol	Price 4/1	Price 4/7	Return
DOW	34.61	27.52	-20.5%
EMN	88.08	73.65	-16.4%
LYB	70.04	56.60	-19.4%
PLTR	84.68	77.84	-8.1%
HOOD	42.16	35.41	-16.0%
PATH	10.50	9.79	-6.8%

5 CONCLUSION

Summary We introduced Shachi, a formal methodology aimed at establishing a principled foundation for LLM-based ABM. We move beyond ad-hoc agent design by proposing a cognitive architecture that decomposes an agent’s policy into four modular components: a core reasoning engine (LLM), intrinsic traits (Configs), contextual persistence (Memory), and expanded capabilities (Tools). This principled decomposition, combined with a standardized agent-environment interface, enables the systematic analysis of how specific architectural choices influence emergent behaviors. We validated this methodology by not only replicating prior work with high fidelity but also by using it to conduct novel scientific inquiries that were previously infeasible, culminating in a simulation of a real-world economic event that verifies the external validity of our approach.

Limitations Our methodology focuses on a principled decomposition of the agent’s cognitive architecture. While the four components provide a robust structure, the fidelity of any agent-based model is a product of both its agents and the world they inhabit. The design of the simulation’s underlying mechanics, for example, the market-clearing rules in stock trading, is another crucial factor that heavily influences emergent outcomes. Thus, while our work advances the design of the agents themselves, achieving comprehensive realism requires careful consideration of both the cognitive model and the environmental model.

Future Work A key avenue for future work is to enhance the agent’s cognitive autonomy. We propose introducing a persistent internal state, such as a learnable value system or motivational model. This would allow agents to develop and adapt goals over time, moving beyond the limitations of static prompting. Additionally, expanding Shachi to support multi-modal environments and interactions remains an important direction for creating more immersive simulations that capture the richness of real-world social behavior.

ACKNOWLEDGEMENTS

We thank Ryuichi Kanoh, Andrew Dai, Yutaro Yamada, Richard Yee, Kosuke Nakago for their valuable comments and suggestions that helped improve this paper.

AUTHORS CONTRIBUTIONS

So Kuroki co-led the project, ported OASIS, StockAgent, CognitiveBiases, EmotionBench and EmergentAnalogies, and conducted the corresponding experiments. Yingtao Tian ported EconAgent and CoMPosT, and conducted the corresponding experiments. Kou Misaki ported AuctionArena and PsychoBench, and conducted the corresponding experiments. Takashi Ikegami provided technical advice and directional suggestions. Takuya Akiba led the framework design, ported Sotopia, and conducted the corresponding experiments. Yujin Tang started and co-led the project, designed the experiments, and contributed significantly to the writing. All authors contributed to the framework design and paper writing.

REFERENCES

- Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.
- Mary Amity, Sang Hoon Kong, and David E Weinstein. Trade protection, stock-market returns, and welfare. Technical report, National Bureau of Economic Research, 2021.
- Philip W Anderson. *The economy as an evolving complex system*. CRC Press, 2018.
- Anthropic. Introducing the model context protocol. <https://www.anthropic.com/news/model-context-protocol>, 2024. Accessed: 2025-05-09.
- W Brian Arthur. Inductive reasoning and bounded rationality. *The American economic review*, 84(2): 406–411, 1994.
- Robert Axelrod. The dissemination of culture: A model with local convergence and global polarization. *Journal of conflict resolution*, 41(2):203–226, 1997.
- Robert Axelrod and William D Hamilton. The evolution of cooperation. *science*, 211(4489): 1390–1396, 1981.
- Robert L Axtell. Zipf distribution of us firm sizes. *science*, 293(5536):1818–1820, 2001.
- Eric Bonabeau. Agent-based modeling: Methods and techniques for simulating human systems. *Proceedings of the national academy of sciences*, 99(suppl_3):7280–7287, 2002.
- Nicolas Bonnotte. *Unidimensional and evolution methods for optimal transportation*. PhD thesis, Université Paris Sud-Paris XI; Scuola normale superiore (Pise, Italie), 2013.
- Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.
- Jiangjie Chen, Siyu Yuan, Rong Ye, Bodhisattwa Prasad Majumder, and Kyle Richardson. Put your money where your mouth is: Evaluating strategic planning and execution of llm agents in an auction arena. *arXiv preprint arXiv:2310.05746*, 2023.
- Myra Cheng, Tiziano Piccardi, and Diyi Yang. Compost: Characterizing and evaluating caricature in llm simulations. *Annual Meeting of the Association for Computational Linguistics*, 2023.
- Michael D Cohen, James G March, and Johan P Olsen. A garbage can model of organizational choice. *Administrative science quarterly*, pp. 1–25, 1972.
- Paul Davidsson. Agent based social simulation: A computer science view. *Journal of artificial societies and social simulation*, 5(1), 2002.

-
- Joshua M Epstein and Robert Axtell. *Growing artificial societies: social science from the bottom up*. Brookings Institution Press, 1996.
- Expected Parrot. EDSL: The Expected Document-Symbol Language. <https://github.com/expectedparrot/eds1>, 2023. Accessed: 2025-09-22.
- J Doyne Farmer and Duncan Foley. The economy needs agent-based modelling. *Nature*, 460(7256): 685–686, 2009.
- Chen Gao, Xiaochong Lan, Nian Li, Yuan Yuan, Jingtao Ding, Zhilun Zhou, Fengli Xu, and Yong Li. Large language models empowered agent-based modeling and simulation: A survey and perspectives. *Humanities and Social Sciences Communications*, 11(1):1–24, 2024.
- Nigel Gilbert. *Agent-based models*. Sage Publications, 2019.
- Nigel Gilbert and Pietro Terna. How to build and use agent-based models in social science. *Mind & Society*, 1:57–72, 2000.
- Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.
- Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022.
- Jen-tse Huang, Wenxuan Wang, Eric John Li, Man Ho Lam, Shujie Ren, Youliang Yuan, Wenxiang Jiao, Zhaopeng Tu, and Michael Lyu. On the humanity of conversational ai: Evaluating the psychological portrayal of llms. In *The Twelfth International Conference on Learning Representations*, 2023.
- Jen-tse Huang, Man Ho Lam, Eric John Li, Shujie Ren, Wenxuan Wang, Wenxiang Jiao, Zhaopeng Tu, and Michael R. Lyu. Apathetic or empathetic? evaluating LLMs’ emotional alignments with humans. In *Advances in Neural Information Processing Systems 37*, 2024.
- Alan Kirman. Ants, rationality, and recruitment. *The Quarterly Journal of Economics*, 108(1): 137–156, 1993.
- Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35: 22199–22213, 2022.
- Nian Li, Chen Gao, Mingyu Li, Yong Li, and Qingmin Liao. Econagent: large language model-empowered agents for simulating macroeconomic activities. *Annual Meeting of the Association for Computational Linguistics*, 2024.
- Simon Malberg, Roman Poletukhin, Carolin M Schuster, and Georg Groh. A comprehensive evaluation of cognitive biases in llms. *arXiv preprint arXiv:2410.15413*, 2024.
- Benjamin S Manning, Kehang Zhu, and John J Horton. Automated social science: Language models as scientist and subjects. Technical report, National Bureau of Economic Research, 2024.
- Robert R McCrae and Paul T Costa Jr. Personality trait structure as a human universal. *American psychologist*, 52(5):509, 1997.
- Eleni Nisioti, Claire Glanois, Elias Najarro, Andrew Dai, Elliot Meyerson, Joachim Winther Pedersen, Laetitia Teodorescu, Conor F Hayes, Shyam Sudhakaran, and Sebastian Risi. From text to life: On the reciprocal relationship between artificial life and large language models. In *Artificial Life Conference Proceedings 36*, volume 2024, pp. 39, 2024.
- Don Norman. *Things that make us smart: Defending human attributes in the age of the machine*. Diversion Books, 2014.
- Arthur M Okun. *Potential GNP: its measurement and significance*. Cowles Foundation for Research in Economics at Yale University, 1963.

-
- Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, pp. 1–22, 2023.
- Edmund S Phelps. Phillips curves, expectations of inflation and optimal unemployment over time. *Economica*, pp. 254–281, 1967.
- Rushi Qiang, Yuchen Zhuang, Yinghao Li, Dingu Sagar V K, Rongzhi Zhang, Changhao Li, Ian Shu-Hei Wong, Sherry Yang, Percy Liang, Chao Zhang, and Bo Dai. Mle-dojo: Interactive environments for empowering llm agents in machine learning engineering, 2025. URL <https://arxiv.org/abs/2505.07782>.
- James M Sakoda. The checkerboard model of social interaction. *The Journal of Mathematical Sociology*, 1(1):119–132, 1971.
- Thomas C Schelling. Dynamic models of segregation. *Journal of mathematical sociology*, 1(2): 143–186, 1971.
- Mitja Steinbacher, Matthias Raddant, Fariba Karimi, Eva Camacho Cuena, Simone Alfarano, Giulia Iori, and Thomas Lux. Advances in the agent-based modeling of economic and social behavior. *SN Business & Economics*, 1(7):99, 2021.
- Rao Surapaneni, Miku Jha, Michael Vakoc, and Todd Segal. Announcing the agent2agent protocol (a2a). <https://developers.googleblog.com/en/a2a-a-new-era-of-agent-interopability/>, April 2025. Accessed: 2025-05-09.
- Ryosuke Takata, Atsushi Masumori, and Takashi Ikegami. Spontaneous emergence of agent individuality through social interactions in large language model-based communities. *Entropy*, 26(12): 1092, 2024.
- Leigh Tesfatsion. Agent-based computational economics: A constructive approach to economic theory. *Handbook of computational economics*, 2:831–880, 2006.
- Leigh Tesfatsion and Kenneth L Judd. *Handbook of computational economics: agent-based computational economics*, volume 2. Elsevier, 2006.
- Alexander Sasha Vezhnevets, John P Agapiou, Avia Aharon, Ron Ziv, Jayd Matyas, Edgar A Duéñez-Guzmán, William A Cunningham, Simon Osindero, Danny Karmon, and Joel Z Leibo. Generative agent-based modeling with actions grounded in physical, social, or digital space using concordia. *arXiv preprint arXiv:2312.03664*, 2023.
- Lev S Vygotsky. *Thought and language*, volume 29. MIT press, 2012.
- Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, et al. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6):186345, 2024.
- Taylor Webb, Keith J Holyoak, and Hongjing Lu. Emergent analogical reasoning in large language models. *Nature Human Behaviour*, 7(9):1526–1541, 2023.
- Patrick Y Wu, Jonathan Nagler, Joshua A Tucker, and Solomon Messing. Large language models can be used to estimate the latent positions of politicians. *arXiv preprint arXiv:2303.12057*, 2023.
- Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, et al. Autogen: Enabling next-gen llm applications via multi-agent conversations. In *First Conference on Language Modeling*, 2024.
- Ziyi Yang, Zaibin Zhang, Zirui Zheng, Yuxian Jiang, Ziyue Gan, Zhiyu Wang, Zijian Ling, Jinsong Chen, Martz Ma, Bowen Dong, et al. Oasis: Open agents social interaction simulations on one million agents. *arXiv preprint arXiv:2411.11581*, 2024.

-
- Chong Zhang, Xinyi Liu, Zhongmou Zhang, Mingyu Jin, Lingyao Li, Zhenting Wang, Wenyue Hua, Dong Shu, Suiyuan Zhu, Xiaobo Jin, et al. When ai meets finance (stockagent): Large language model-based stock trading in simulated real-world environments. *arXiv preprint arXiv:2407.18957*, 2024.
- Stephan Zheng, Alexander Trott, Sunil Srinivasa, David C Parkes, and Richard Socher. The ai economist: Taxation policy design via two-level deep multiagent reinforcement learning. *Science advances*, 8(18):eabk2607, 2022.
- Xuhui Zhou, Hao Zhu, Leena Mathur, Ruohong Zhang, Haofei Yu, Zhengyang Qi, Louis-Philippe Morency, Yonatan Bisk, Daniel Fried, Graham Neubig, et al. Sotopia: Interactive evaluation for social intelligence in language agents. *International Conference on Learning Representations*, 2024.

A TASKS IN SHACHI

Table 6: **Overview of Tasks and Their Levels.** Detailed task information is provided in Section B.

#	Lvl	Task	Description
1	I	PsychoBench (Huang et al., 2023)	Evaluates psychological traits of LLMs via 13 psychometric scales across personality, interpersonal, motivational, and emotional domains
2	I	CoMPosT (Cheng et al., 2023)	Measures LLM simulations’ susceptibility to caricature through several dimensions
3	I	CognitiveBiases (Malberg et al., 2024)	Evaluates 30 classic cognitive biases with paired control–treatment prompts
4	I	EmotionBench (Huang et al., 2024)	Measures shifts in eight core emotions triggered by situational prompts
5	I	EmergentAnalogies (Webb et al., 2023)	Probes zero-shot analogical reasoning across matrix, string, verbal, and story tasks
6	II	EconAgent (Li et al., 2024)	LLM-powered multi-agent system for macroeconomic simulation with human-like behaviors
7	II	StockAgent (Zhang et al., 2024)	LLM-based multi-agent system that simulates real-world stock trading under dynamic market conditions.
8	II	AuctionArena (Chen et al., 2023)	Evaluates strategic planning and adaptive reasoning of LLM agents in simulated dynamic auctions
9	III	OASIS (Yang et al., 2024)	Large-scale multi-agent simulation benchmark designed for studying social media phenomena
10	III	Sotopia (Zhou et al., 2024)	Open-ended role-play environment to simulate complex social interactions and measure agents’ social intelligence

B DETAILED TASK IMPLEMENTATION

B.1 PSYCHOBENCH (LEVEL I)

Description PsychoBench (Huang et al., 2023) evaluates the psychological portrayal of LLMs, drawing from psychometric research to examine their human-like psychological traits.

Method It systematically measures thirteen psychological dimensions categorized into personality traits (e.g., Big Five Inventory, Dark Triad), interpersonal relationships (e.g., Bem’s Sex Role Inventory), motivational tendencies (e.g., General Self-Efficacy), and emotional abilities (e.g., Emotional Intelligence Scale). The methodology involves administering psychometric scales directly via prompts. Crucial experimental parameters include detailed instructions for Likert-scale responses, randomized question order to ensure robustness, and strict control of model inference temperature (set to zero or near-zero).

Experimental Settings For the reproduction study reported in Table 1, we follow the original setup and use Llama-2-13b-chat-hf as the LLM component without a memory component. The task includes neither config nor tool components. We fix the LLM temperature to 0, test 10 random seeds for question ordering, and average the results. We compute the MAE between the scores obtained with Shachi (Ours) and those presented in the original paper across all psychological subscales. We add a naive baseline that answers each prompt by randomly selecting from the available choices. It takes a few minutes for the LLM to complete all psychometric scales.

B.2 COMPOST (LEVEL I)

Description CoMPosT (Cheng et al., 2023) investigates how susceptible large language models (LLMs) are to caricature.

Method To quantify this effect, the framework decomposes caricature into four orthogonal dimensions—*context*, *model*, *persona*, and *topic*—which specify the simulated scenario, the LLM configuration, the target opinion, and the domain of discourse, respectively. Two metrics are introduced: the *individuation score*, which tests whether the simulated persona is distinguishable from the default persona, and the *exaggeration score*, which measures the degree to which the simulation amplifies persona–topic features.

Experimental Settings In our reproduction study (Table 1), we use GPT-3.5-turbo (gpt-3.5-turbo-0125) as the LLM component without a memory component. The task includes neither config nor tool components. We compare the scores produced by our implementation (and a baseline) with those reported in the original paper. Although we replicate the experimental setup as faithfully as possible, minor differences arise owing to updates in the underlying LLMs. Each method outputs a distribution of scores. We assess similarity between two such distributions via one-dimensional optimal transport with the absolute difference as the cost function. In this setting, the transport cost simplifies to MAE between the paired, sorted samples of the distributions, following Bonnotte (2013). The evaluation takes approximately 20 minutes.

B.3 COGNITIVEBIASES (LEVEL I)

Description CognitiveBiases (Malberg et al., 2024) evaluates how LLMs exhibit 30 well-known cognitive biases, motivated by the increasing use of LLMs in high-stakes decision-making.

Method It specifically measures biases such as anchoring, framing, and 28 others commonly identified in psychology and behavioral economics. The core methodology employs a systematic framework that generates and administers 30,000 bias-specific test cases across 200 distinct decision-making scenarios, comparing model responses under control vs. treatment conditions. Crucial parameters include the explicit control/treatment designs for each bias, two standardized answer scales (7-point Likert or 11-point percentage), and reversed option orders to account for position bias, ensuring reproducibility and comprehensive coverage.

Experimental Settings For the reproduction study in Table 1, we follow the original setup and use GPT-4o-mini (`gpt-4o-mini-2024-07-18`) as the LLM component without a memory component. The task includes neither config nor tool components. The LLM temperature is fixed at 0. We run the evaluation under three random seeds for question ordering and report the average. We measure MAE between the bias scores obtained with Shachi (ours) and those in the original paper across 30 biases. We include a naive baseline that responds to each prompt by randomly sampling from the available choices. Evaluating all 30 biases requires roughly one hour.

B.4 EMOTIONBENCH (LEVEL I)

Description EmotionBench (Huang et al., 2024) evaluates how LLMs respond emotionally to various real-life situations, drawing from emotion appraisal theory to examine their alignment with human-like emotional reactions.

Method It measures eight key positive and negative emotions (anger, anxiety, depression, frustration, jealousy, guilt, fear, embarrassment) and tracks how situational contexts raise or lower these emotions compared to a default baseline. It uses self-report scales (e.g., PANAS), first measuring a model’s default emotional state, then presenting situational prompts, and finally re-measuring changes in emotional scores.

Experimental Settings For the reproduction study reported in Table 1, we follow the original setup and use GPT-3.5-turbo (`gpt-3.5-turbo-0125`) as the LLM component without a memory component. The task includes neither config nor tool components. We compute MAE between the eight key positive and negative emotion scores obtained with Shachi (Ours) and those reproduced with their original code on the PANAS scale. We add a naive baseline that answers each prompt by randomly selecting from the available choices. Evaluating the LLM’s affective state takes roughly one minute.

B.5 EMERGENTANALOGIES (LEVEL I)

Description EmergentAnalogies (Webb et al., 2023) evaluates zero-shot analogical reasoning in LLMs, highlighting analogy’s key role in fluid intelligence.

Method The benchmark tests a range of domains for abstract pattern induction and relational reasoning, featuring four core tasks—matrix reasoning, letter-string analogies, four-term verbal analogies, and story analogies. We specifically target free-response accuracy on the matrix reasoning.

Experimental Settings For our reproduction study in Table 1, we use GPT-4 (`gpt-4-0613`) as the LLM component without a memory component. The task includes neither config nor tool components. We set the LLM temperature to 0 and randomly sampled problems from each category. We use three different seeds for sampling and averaged the results. We compute MAE between the category-wise averages obtained with Shachi (Ours) and those reproduced with their source code. As a naive baseline, we include a model that simply generates a random matrix for each prompt. For our cross-task agent study in Table 2, we utilize GPT-4o (`gpt-4o-2024-08-06`). We compute an overall average across all categories. It takes approximately one minute to assess matrix reasoning.

B.6 ECONAGENT (LEVEL II)

Description Econagent (Li et al., 2024) is a LLM-powered multi-agent system for macroeconomic simulation with human-like behaviors.

Method Building on the virtual economic framework of Zheng et al. (2022), it employs an economic environment where each agents are placed into a shared, quasi-realistic market with an endowment of specific skills and wealth. Agents decide how much to work and consume, and their decisions collectively produce macroeconomic dynamics. A rule-based environment acts as both a central government (collecting taxes) and a central bank (adjusting interest rates), forming a macroeconomic loop. The original work demonstrates that LLM-powered agents make realistic decisions individually and, collectively, produce coherent macro-level dynamics.

Experimental Settings In our experiments in Section C.1, we use several models as the LLM component without a memory component. The task includes neither config nor tool components. We follow the original paper of simulating 100 agents for 240 months, making 20 annual tax-and-monetary economic cycles. The simulation takes approximately one hour to execute with our Shachi-based implementation. Unlike the original paper, which employs a single LLM model, our replication evaluates a widely used model suite. For the carrying memory experiment in Section 4.2.1, we add a buffer memory to record each agent’s behaviors and transfer an agent to the CognitiveBiases task.

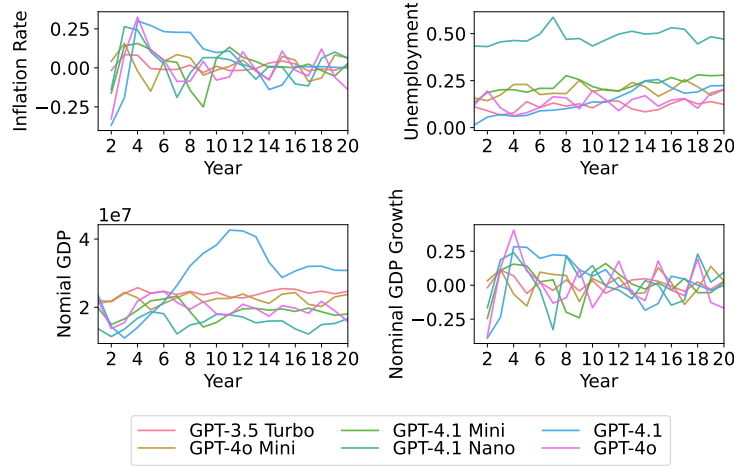


Figure 5: **All Marcoeconomic Indicators.** These are extra results accompanying those in Section C.1.

Extra Results In Figure 5 we share all macro-economic indicators. All LLMs show a change in indicators, yet emerging behaviors differ in detail, showing characteristics of different LLMs.

B.7 STOCKAGENT (LEVEL II)

Description StockAgent (Zhang et al., 2024) is a large language model-based multi-agent system that simulates real-world stock trading under dynamic market conditions.

Method Specifically, it runs event-driven simulations where LLM-driven agents sequentially make loans, buy, sell, predict, and post and check forum decisions, while the market data and stock prices evolve daily. The framework models two distinct stocks: Stock A, a 10-year chemical stock, and Stock B, a 3-year tech stock, dynamically simulating their price fluctuations. Notable parameters include initial agent capital allocations, loan-to-value ratios, interest rates, and real-world-like events (e.g., financial reports).

Experimental Settings In the reproduction study reported in Table 1 and Figure 7, we follow the original setup and use GPT-3.5-turbo (gpt-3.5-turbo-0125) as the LLM component with buffer memory component (length 3). The task contains both config and tool components. Our Shachi implementation exposes a forum API as the tool: each agent autonomously decides whether to read comments posted the previous day about the stock market. For config, the environment assigns each agent one of four investment styles—*Conservative*, *Aggressive*, *Balanced*, or *Growth-Oriented*. We fixed random seeds and event sequences to match the original study. The simulation runs for 10 days with three sessions per day, involving 50 agents and 1,500 rounds. We average session-level price changes over three seeds and compute MAE between the session-by-session price dynamics of stocks A and B produced by Shachi (ours) and those reproduced with the authors’ source code. As a baseline, we remove the tool and memory modules from the agents to isolate their impact on performance. For our cross-task agent study in Table 2, we deploy 50 agents—25 powered by GPT-4o (gpt-4o-2024-08-06) and 25 by GPT-3.5-turbo (gpt-3.5-turbo-0125). To quantify volatility, we calculate the price change rate for Stocks A and B between the first and final sessions and average the two rates. Simulating 1,500 rounds required several hours.

B.8 AUCTIONARENA (LEVEL II)

Description AuctionArena (Chen et al., 2023) evaluates the strategic planning and execution capabilities of LLM agents within a dynamic auction environment, motivated by the need for realistic benchmarks of sequential decision-making in competitive scenarios.

Method The environment specifically assesses skills such as resource allocation, risk management, and adaptive strategic reasoning. The methodology employs a simulation of open ascending-price auctions where agents act as bidders, making decisions based on the Belief-Desire-Intention (BDI) framework. Crucial parameters include item valuation (distinguishing between cheap and expensive items), intentional overestimation of item value to simulate “winner’s curse”, and explicit prioritization strategies that agents dynamically adjust after each round.

Experimental Settings For the reproduction study reported in Table 1, we follow “Standard Competition” setting, where the evaluated agent (GPT-4-turbo, gpt-4-1106-preview) competes directly against GPT-3.5-turbo (gpt-3.5-turbo-1106) and GPT-4-turbo (gpt-4-1106-preview) agents. All agents use a chat-history memory (window 20, token 10,000). The environment supplies a config component but no tools. As for the config, each agent is assigned one of two strategies: profit-first — maximize final profit, or item-first — win the most items. We fix the temperature at 0, randomly assign the item order, and use the TrueSkill system to quantify agent performance across 10 auctions with different item sequences. As a baseline, we repeat this comparison while removing the agent’s memory module to isolate its impact on performance. As for Figure 7, we use the result for GPT-4 (gpt-4-0613) agent in the competition against GPT-3.5-turbo (gpt-3.5-turbo-0125) and GPT-4 (gpt-4-0613), with items ordered by descending price. For our cross-task agent study in Table 2, we use a TrueSkill score of GPT-4o (gpt-4o-2024-08-06) in “Standard Competition” setting. A three-agent, ten-item auction with a \$20,000 budget takes about 20 minutes to run.

B.9 SOTOPIA (LEVEL III)

Description Sotopia (Zhou et al., 2024) introduces an open-ended role-play environment with a multidimensional evaluation framework to simulate complex social interactions and systematically measure LLM agents’ social intelligence.

Method In the original Sotopia implementation, at every turn, it concatenates the entire dialogue history from all agents into a single prompt. In Shachi, by contrast, memory management is an agent-side responsibility, so the environment supplies only the most recent message. The evaluation result consists of seven metrics (SOC, SEC, FIN, REL, KNO, GOAL, and BEL).

Experimental Settings For the Table 1 experiments, we follow the original setup and use GPT-4 (gpt-4-0613) as the LLM component with a buffer memory component (all history with a 16,000 token limit). The task includes neither config nor tool components. To faithfully reproduce Sotopia inside Shachi, our reproduced agent (the agent reported as “Shachi (Ours)” in Table 1 and the “Sotopia” line in Table 2) restructures the full conversation history into one prompt in the same format. We set the LLM temperature to 1. We consider the MAE between the results reported in the original paper and our experiments. For our cross-task agent study in Table 2, we utilize GPT-4o (gpt-4o-2024-08-06). We apply min-max normalization using the maximum and minimum values defined for each metric and then calculate the overall average across these metrics. The evaluation takes approximately 20 minutes.

B.10 OASIS (LEVEL III)

Description OASIS (Yang et al., 2024) is a large-scale multi-agent simulation benchmark for studying how up to one million LLM-based agents interact on social media platforms, focusing on information propagation, group polarization, and herd effects.

Method OASIS simulates large-scale social media environments by combining an environment server, a recommendation system, and a time engine. Each user is modeled as an LLM-based agent

with a 21-type action space (e.g., posting, commenting, following), whose behavior and memory evolve in real time. By supporting up to one million agents, OASIS facilitates the study of complex emergent phenomena, such as information spreading, group polarization, and herd effects, in both X and Reddit-like settings. In our experiment, we utilize an X-like setting.

Experimental Settings In our experiments, OASIS is employed in both Sections 4.2.1 and 4.2.2 under the same setup, following one of the original settings. Specifically, we use one influential agent who posts about Amazon (“report: amazon plans to open its first physical store in new york URL”), and its followers respond, with limited connectivity among the followers themselves. We use several models as the LLM component based on experiments with a chat-history memory (window 5, token limit 100,000). The environment supplies a config component but no tools. As for the config, each agent is assigned a distinct profile, such as “@ohiostate alumni. High tech marketer, salon entrepreneur, & web design enthusiast. Fashion & food are my passions. Views are my own.” In Section 4.2.1, we use GPT-4o-mini (gpt-4o-mini-2024-07-18), while Section 4.2.2 employs GPT-3.5-turbo (gpt-3.5-turbo-0125) and GPT-4o (gpt-4o-2024-08-06). Ten iterations with 111 agents complete in a few minutes.

C EXTRA SETUPS AND RESULTS

C.1 IMPACT OF BACKEND LLMs

Experimental Setup For EconAgent, we replaced the single model in the original study with a set of widely used models. All other settings remain unchanged: an economy of 100 simulated agents evolves over 240 months, and we track both standard macro-indicator trajectories and the emergence of two canonical macroeconomic regularities: the Phillips Curve (Phelps, 1967) and Okun’s Law (Okun, 1963).

Experimental Results Figure 6 visualizes how the choice of backend LLM affects macroeconomic patterns and indicators. All LLM-based agents collectively show behaviors in accordance with both Phillips Curve and Okun’s Law. Nonetheless, the curves differ in detail: intercept shifts in the Phillips Curve point to varying baseline unemployment rates, while slope changes in Okun’s Law indicate differing GDP-unemployment trade-offs. Moreover, although most LLM backends produce similar variations of indicators, GPT-4.1 Nano yields consistently higher unemployment, and GPT-4.1 delivers markedly stronger GDP growth. Our Shachi framework’s modular separation of concerns allows easy backend change with a single configuration line, unlike ad-hoc setups, where replacing the LLM typically means re-implementing or realigning several parts of the agent pipeline. This streamlined flexibility enables cleaner comparisons and faster iteration, like what we find here.

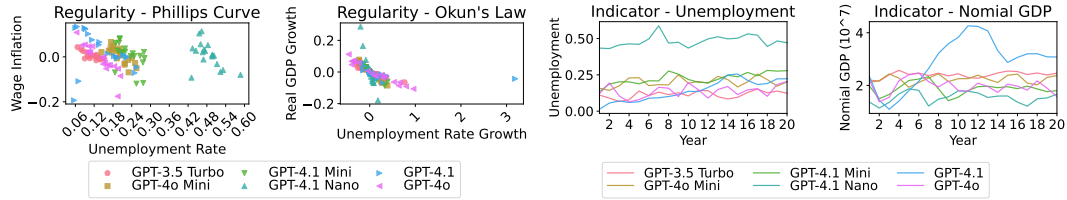


Figure 6: **Impact of Different LLMs.** Macroeconomic regularities and indicators in EconAgent Task. All LLM-based agents collectively show correct behaviors in macroeconomic regularities and similar indicators, yet emerging behaviours differ in details, showing characteristics of different LLMs.

C.2 DETAILED RESULTS FROM REPRODUCING PRIOR WORKS

Experimental Setup We selected eight out of the ten tasks in our benchmark suite for reproduction. These tasks were chosen primarily because they report concrete, quantitative metrics that make it feasible to assess reproduction fidelity in a principled way. All agents were implemented within Shachi using modular combinations of LLM, memory, tools and configuration components, preserving as much of the original behavioral logic as possible. The baselines are variations of the original agents (e.g., removing all the components and making it a thin wrapper of LLM) or agents with randomized behaviors.

Experimental Results Table 1 shows the reproduction error for all selected tasks, measured by mean absolute error (MAE) between the original and the reproduced results. Across all tasks, Shachi achieves consistently lower error than the baseline, often by a large margin. This indicates that Shachi accurately preserves the original agents’ quantitative outputs, validating its reliability.

While Table 1 provides an aggregated view of reproduction fidelity, we also present “zoomed-in” results in Figure 7 to illustrate how well Shachi captures the dynamics of the original simulations. On the left, we visualize the price trajectories of two stocks in the StockAgent task, showing a close match between the original and Shachi-reproduced trends over time. On the right, we show that Shachi also replicates fine-grained behavior patterns during the sequential auction of ten items, as shown in the heatmap of averaged priority scores for the remaining items across bidding rounds in AuctionArena. These qualitative visualizations reinforce that Shachi not only matches the original quantitative outputs but also captures the underlying temporal and structural patterns.

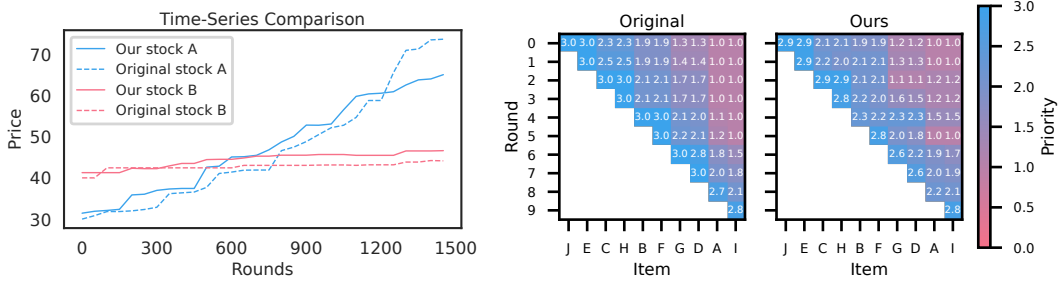


Figure 7: **Reproducing System Dynamics.** Left: Evolution of the two stock prices in StockAgent, comparing the original study with our Shachi reproduction; Right: Reported and reproduced heatmaps of priority scores and their changes before each bidding round in AuctionArena (descending). Together, these results illustrate how well Shachi captures the dynamics of the original simulations.

C.3 DETAILED RESULTS FROM GENERALIZATION

Experimental Setup In this setting, we fixed the underlying LLM across all experiments to GPT-4o (gpt-4o-2024-08-06) (Achiam et al., 2023) to ensure consistency, and applied agents originally developed for one task to other tasks. In particular, we study how the presence or absence of configuration, memory, and tool modules affects an agent’s ability to generalize.

We include four agents/tasks in this experiment. Note that the evaluation metrics are task-dependent (see above), but all scores are normalized relative to each agent’s in-domain performance. Agents differ in composition depending on the original task for which they were designed. For example, the agent in EmergentAnalogies only consists of an LLM component without any additional structure, while StockAgent includes all components. By evaluating how well each agent performs when deployed in a new task, we aim to understand whether components help or hinder generalization.

Experimental Results We report the normalized performance of each agent-task pairing in Table 2. Each column corresponds to a target task, and each row to an agent, with its components listed in the parentheses. Scores are normalized such that the in-domain performance (diagonal entries) is always one, allowing for easier interpretation of relative effectiveness across tasks.

Two immediate observations emerge from the table: (1) The EmergentAnalogies task requires only a minimal agent consisting solely of an LLM without additional components. As a result, all agents’ performance is similar on this task. This suggests that when the target task is simple and does not require auxiliary capabilities like tool use or configuration adaptation, minimalist agents can generalize sufficiently well. (2) Sotopia represents a complex multi-agent communication setting that originally uses memory. We expected the performance from an EmergentAnalogies to be different from the other three agents on this task due to the lack of a memory module, but to our surprise, all transferred agents achieve values close to 1. One possible explanation is that memory was not effectively leveraged even in the original setup, or that the LLM’s intrinsic context window suffices for short-term coherence.

More nuanced patterns appear in the mid-table entries. For example, the agent from StockAgent includes all four components and transfers reasonably well to all other tasks (2nd row). On the other hand, the agent from AuctionArena demonstrated similar performance on tasks except for StockAgent (3rd row), which requires tool usage, a component that the AuctionArena agent does not have. These results suggest that the presence or absence of the tool component leads to the behavioral differences.

C.4 DETAILED SETTINGS FROM REAL-WORLD VALIDATION (U.S. TARIFF SHOCK)

Experimental Setup We use the StockAgent task as our testbed, and simulate a 5-day trading period (April 1-5, 2025) surrounding a tariff shock. We conduct a cumulative ablation study across four settings: (1) Base: Agents from the standard StockAgent task with no extra information. (2) Base + Config: Agents are exposed to a pre-announcement news headline about imminent tariffs, added to their configuration prompt. (3) Base + Prompt + Memory: Agents are additionally equipped with memory containing a summary of academic research on how tariffs negatively impact markets (Amity

et al., 2021). (4) Base + Prompt + Memory + Tool: Agents are finally given access to a news-retrieval tool that provides daily updates on the escalating trade tensions.

Details about the information used in the agent settings are as follows:

- Config in setting #2: In prompt, we incorporate the news title “*Trump’s ‘Liberation Day’ Tariffs Loom; Treasury Yields Fall President Trump has set a deadline of Wednesday to announce sweeping tariffs.*” from WSJ². This news was released before the massive sell-off in April.
- Memory in setting #3: In augmented memory, we use a summary of the paper “*Trade protection, stock-market returns, and welfare*” (Amiti et al., 2021) produced by gpt-4o (see text box 2). The last revision date of this paper was also before the April sell-off.
- Tool in setting #4: We collected WSJ news^{3,4,5} before the April sell-off to store in a news database, the news-fetching tool grabs one news on each day and adds that to the agents’ input.

For each setting, we run 5 trials and report the mean. Each trial is run for 5 simulation days (April 1–5, 2025). More concretely for setting #4, the agents use the news-fetching tool to see the news (footnote 3) on April 2nd, the news (footnote 4) on April 3rd, and the news (footnote 5) on April 4th.

Text box 2: Paper summary by gpt-4o

NBER Working Paper 28758, titled “Trade Protection, Stock-Market Returns, and Welfare” by Mary Amiti, Matthieu Gomez, Sang Hoon Kong, and David Weinstein, studies the impact of tariff announcements during the U.S.-China trade war on financial markets and welfare. Key findings:

- Tariff announcements significantly reduced U.S. stock prices (-11.5% cumulatively across 11 key events).
- They also triggered a drop in nominal and real government bond yields (flight to safety).
- Firms exposed to China saw worse stock returns and future business outcomes (profits, employment, productivity).
- The authors develop a dynamic specific-factors model showing that welfare loss stems not just from price distortions but also from expected future declines in productivity (TFP).
- Estimated U.S. welfare loss from the trade war is 3.0%, with 1.1 percentage points attributed to expected TFP decline.

The paper uses a novel approach of mapping financial market responses to welfare analysis, bridging asset pricing models with trade theory.

We use the following prompt (text box 3) to find stocks matching the profiles of Stock A and B.

Text box 3: Prompt used to find the stocks matching Stocks A and B

Find me 3 US listed companies each, matching the profile of stocks A and B below.

Stock A: Established chemical company with 10-year listing history, experiencing revenue decline but stable operations under new proactive CEO leadership.

Stock B: Recently listed 3-year tech company with high growth potential but questionable data reliability and past IPO disclosure issues.

²Trump’s ‘Liberation Day’ Tariffs Loom; Treasury Yields Fall. WSJ, April 1st 2025.

³Trump Unveils Sweeping Levies in Stark Shift in Trade Policy. WSJ, April 2nd 2025.

⁴Tariffs Send Dow to 1600-Point Decline, Dollar Slumps. WSJ, April 3rd 2025.

⁵Dow Tumbles 2,200 Points, Bonds Rally After China Retaliates Against Trump Tariffs. WSJ, April 4th 2025.

D CODE EXAMPLES

D.1 GYM-STYLE INTERFACE

Listing 1 is a example showing how agents can be organized alongside an environment that follows a Gym-style interface in Shachi. Each agent receives observations from the environment and produces responses (actions), while the environment tracks episodes, rewards, and termination conditions.

```
1 N_EPISODES = 10
2
3 async def main() -> None:
4     agents = [
5         AgentExample1(model="openai/gpt-4o", temperature=0),
6         AgentExample2(model="openai/gpt-3.5-turbo", temperature=0),
7         AgentExample3(model="openai/gpt-4", temperature=0),
8     ]
9     env = EnvironmentExample()
10
11     total_rewards = {agent_id: 0.0 for agent_id in range(len(agents))}
12     for _ in range(N_EPISODES):
13         observations = await env.reset()
14         while not env.done():
15             futures = {
16                 agent_id: agents[agent_id].step(observation)
17                 for agent_id, observation in observations.items()
18             }
19             responses = dict(zip(futures.keys(), await asyncio.gather(*
20                                 futures.values())))
21             observations = await env.step(responses)
22             for agent_id, observation in observations.items():
23                 if observation.reward is not None:
24                     total_rewards[agent_id] += observation.reward
25
26 if __name__ == "__main__":
27     asyncio.run(main())
```

Listing 1: Example code for the gym-like agent loop used in our framework.

D.2 TOOL USAGE

In this framework, the environment provides a tool component that the agent can dynamically call through an LLM. The agent sends observations to the LLM, which may respond with tool call instructions. The agent executes these tools, returns their outputs, and continues the conversation until no more tool calls are needed. The final LLM response thus incorporates results from any tools. Listing 2 shows a minimal implementation of this process.

```
1 async def step(self, observation: kujira.base.Observation) -> str:
2     prompt = observation.format_as_prompt_text()
3     available_tools = observation.tools
4
5     # Convert tool information for LLM
6     tools_for_llm = [
7         {
8             "type": "function",
9             "function": {
10                 "name": tool.name,
11                 "description": tool.description,
12                 "parameters": tool.parameters_type.model_json_schema(),
13             },
14         }
15         for tool in available_tools
```



```

16     ]
17
18     # Chat history with proper typing
19     messages: list[dict[str, Any]] = [{"role": "user", "content": prompt
20                                         }]
21
22     # Allow up to 5 tool calls
23     for _ in range(5):
24         completion = await litellm.acompletion(
25             messages=messages,
26             model=self.model,
27             tools=tools_for_llm,
28             tool_choice="auto",
29         )
30         assistant_message = completion.choices[0].message
31
32         # Add message to chat history with proper typing
33         message_entry: dict[str, Any] = {
34             "role": "assistant",
35             "content": assistant_message.content
36             if assistant_message.content is not None
37             else "",
38         }
39         if hasattr(assistant_message, "tool_calls") and assistant_message
40             .tool_calls:
41             message_entry["tool_calls"] = assistant_message.tool_calls
42             messages.append(message_entry)
43
44         # If no tool calls, exit
45         if not hasattr(assistant_message, "tool_calls") or not
46             assistant_message.tool_calls:
47             return assistant_message.content if assistant_message.content
48                 is not None else ""
49
50     # Process tool calls
51     for tool_call in assistant_message.tool_calls:
52         function_name = tool_call.function.name
53         function_args = tool_call.function.arguments
54
55         # Find corresponding tool
56         matching_tool = None
57         for tool in available_tools:
58             if tool.name == function_name:
59                 matching_tool = tool
60                 break
61
62         if matching_tool:
63             # Execute tool
64             try:
65                 parameters = matching_tool.parameters_type.
66                     model_validate_json(
67                         function_args
68                     )
69                 tool_response = matching_tool.fun(parameters)
70                 response_text = tool_response.format_as_prompt_text()
71
72                 # Add tool response to chat history with proper
73                 typing
74                 tool_message: dict[str, Any] = {
75                     "role": "tool",
76                     "tool_call_id": tool_call.id,
77                     "name": function_name,
78                     "content": response_text,
79                 }
80                 messages.append(tool_message)

```

```

75         print(f"Tool call: {function_name}")
76         print(f"Arguments: {function_args}")
77         print(f"Response: {response_text}")
78     except Exception as e:
79         print(f"Error occurred during tool execution: {str(e)}")
80
81     # Get final response
82     final_completion = await litellm.acompletion(
83         messages=messages,
84         model=self.model,
85     )
86     return (
87         final_completion.choices[0].message.content
88         if final_completion.choices[0].message.content is not None
89         else ""
90     )
91

```

Listing 2: Example code for the tool usage in our framework.

D.3 INTER-AGENT INTERACTION DESIGN

Shachi can specify both static and dynamic communication topologies. In Shachi, messages are structured with explicit sender id (`src_agent_id`) and receiver id (`dst_agent_id`). Listing 3 shows a minimal implementation of this process.

```

1 class Message(pydantic.BaseModel, abc.ABC):
2     time: int
3     src_agent_id: int | None # sender id
4     dst_agent_id: int | None # receiver id

```

Listing 3: Messaging design for inter-agent communication in Shachi.

1. Dynamic Communication Graphs: Agents can autonomously determine their communication partners by setting the receiver id (`dst_agent_id`) at runtime based on their observations, internal state, or task requirements. This allows for flexible communication patterns that evolve during simulation.
2. Static Communication Graphs: The environment can enforce predetermined communication topologies by restricting which receiver id (`dst_agent_id`) values are valid for each agent, effectively implementing static network structures.

Upon receiving messages from agents, the environment functions as a message router that processes sender-receiver pairs to implement various communication patterns. It also includes broadcasting to all agents when receiver id (`dst_agent_id`) is set to `None`.

D.4 TWO-STAGE PARSING

Listing 4 presents a minimal example of how agents employ two-stage parsing in Shachi. To generate outputs in the required format without harming LLM performance, we leverage API features such as structured outputs and function calls, together with the two-stage parsing strategy.

```

1 async def call_llm(
2     messages: list[dict[str, str]],
3     model: str,
4     temperature: float,
5     parsing_mode: PARSING_MODE,
6     parsing_model: str | None = "gpt-4.1-mini-2025-04-14",
7     response_type: TResponseType | None = None,
8 ) -> str | TResponseType:
9
10     # First stage: generate in a plain text
11     completion1 = await litellm.acompletion(
12         messages=messages,

```

```

13         model=model,
14         temperature=temperature,
15         max_retries=MAX_RETRIES,
16     )
17     response_text_1: str = completion1.choices[0].message.content
18
19     # Second stage: parse the plain text into a structured output
20     completion2 = await litellm.acompletion(
21         messages=[
22             {
23                 "role": "user",
24                 "content": f"""
25 Based on the text provided below, output JSON. If the input is plain text
26 ',
27 extract the necessary information while preserving the original wording
28 as much as possible. If the input is JSON, output it unchanged, except
29 fix any formatting errors you find.
30 '''
31 {response_text_1}
32 '''
33 The JSON should follow the schema below:
34 '''
35 {response_type.model_json_schema()}
36 '''
37 """.strip(),
38         ],
39         model=parsing_model,
40         temperature=temperature,
41         response_format=response_type,
42         max_retries=MAX_RETRIES,
43     )
44     response_text: str = completion2.choices[0].message.content
45     response_obj = response_type.model_validate_json(response_text)
46     return response_obj

```

Listing 4: Example code for the two-stage parsing used in our framework.

E MORE RELATED WORKS

E.1 ABM WITHOUT LLMs

During the 1970s and 1980s, ABM emerged as a powerful way to study complex social processes. In these formative years, Schelling (Schelling, 1971) and Sakoda (Sakoda, 1971) revealed how minimal individual preferences can produce starkly segregated or patterned neighborhoods, while the Garbage Can Model (Cohen et al., 1972) captured the unpredictable intersections of agents, problems, solutions, and participation opportunities in organizations. Meanwhile, through computer tournaments of the iterated Prisoner’s Dilemma (Axelrod & Hamilton, 1981), Axelrod showed that ongoing reciprocal encounters can make cooperation a self-interested strategy. Around the same time, a study illustrated how heterogeneous, boundedly rational agents on evolving networks can sustain perpetual adaptation and out-of-equilibrium dynamics, setting the stage for viewing large-scale patterns as emergent from local interactions (Anderson, 2018).

In the 1990s, researchers built on these foundational concepts by applying ABM more intensively to economic and social phenomena. For example, El Farol Bar problem (Arthur, 1994) highlighted how learning and adaptation among heterogeneous agents may produce oscillatory behavior, whereas Kirman’s study of ants (Kirman, 1993) demonstrated that tiny random shifts in decision-making can trigger large-scale herding. Broadening ABM’s scope, Epstein and Axtell (Epstein & Axtell, 1996) introduced the model to show how wealth, culture, and disease dynamics emerge from simple local rules. Axelrod (Axelrod, 1997) further explored cultural dissemination, revealing how social interaction fosters both convergence and enduring diversity.

Since the early 2000s, ABM has steadily expanded into large-scale empirical contexts, including macroeconomics and public policy. Axtell (Axtell, 2001) demonstrated how firm-size distributions—specifically, the Zipf distribution—could be replicated from bottom-up processes driven by micro-level agent interactions. In parallel, Bonabeau (Bonabeau, 2002) surveyed ABM’s exceptional capacity to capture emergent phenomena, emphasizing how complex global patterns can arise unpredictably from simpler, localized rules. After the financial crisis of 2008, Farmer and Foley (Farmer & Foley, 2009) further highlighted ABM’s potential for illuminating financial instabilities and guiding policy interventions, thereby reinforcing the method’s predictive and explanatory power in volatile economic environments.

E.2 ABM WITH LLMs

Below, we elaborate on representative efforts that investigate human-like behavior in LLM agents across psychological, social, economic, and financial domains.

One effort in this direction is PsychoBench (Huang et al., 2023), which assesses the psychological portrayal of LLM agents by examining aspects like personality traits and social intelligence, shedding light on the challenges of modeling human-like behavior in LLM-based agents. In the realm of social simulations, generative agents (Park et al., 2023) utilize LLMs to create believable human-like behaviors, including planning, interaction, and reflection, as demonstrated in interactive environments where agents autonomously coordinate social activities like organizing events. In a similar vein, spontaneously emergent behaviors—such as personality differentiation, social norm formation, and collective hallucinations—have been observed through repeated interactions in a simulated community (Takata et al., 2024). OASIS (Yang et al., 2024) extends this concept by enabling large-scale social media simulations with up to one million agents, facilitating the study of complex social phenomena such as information spreading, group polarization, and herd behavior. Sotopia (Zhou et al., 2024) further examines social intelligence by evaluating LLM agents’ coordination, collaboration, and strategic reasoning in diverse scenarios, identifying challenges in achieving human-like social behavior. In the economic domain, EconAgent (Li et al., 2024) leverages LLMs to simulate macroeconomic activities by incorporating human-like decision-making processes, such as work and consumption behaviors, and dynamically adapting to market trends through memory mechanisms. This approach enables more realistic economic dynamics compared to traditional rule-based or learning-based models. In the financial domain, StockAgent (Zhang et al., 2024) models investor behaviors influenced by external factors, providing insights into trading dynamics, while AuctionArena (Chen et al., 2023) focuses on strategic decision-making in competitive bidding environments, testing LLMs’ ability to manage resources and adapt strategies.

Together, these works provide a foundation for understanding the capabilities and limitations of LLMs in modeling complex human behaviors across diverse environments.