A Parallelized Framework for Simulating Large-Scale LLM Agents with Realistic Environments and Interactions

Jun Zhang[1] **Yuwei Yan**[2] **Junbo Yan**[1]
Zhiheng Zheng[1] **Jinghua Piao**[1] **Depeng Jin**[1] **Yong Li**[1*]
[1]Department of Electronic Engineering, BNRist, Tsinghua University
[2]Information Hub, The Hong Kong University of Science and Technology (Guangzhou)
zhangjun990222@gmail.com liyong07@tsinghua.edu.cn

Abstract

The development of large language models (LLMs) offers a feasible approach to simulating complex behavioral patterns of individuals, enabling the reconstruction of microscopic and realistic human societal dynamics. However, this approach demands a realistic environment to provide feedback for the evolving of agents, as well as a parallelized framework to support the massive and uncertain interactions among agents and environments. To address the gaps in existing works, which lack real-world environments and struggle with complex interactions, we design a scalable framework named **AgentSociety**, which integrates realistic societal environments and parallelized interactions to support simulations of large-scale agents. Experiments demonstrate that the framework can support simulations of 30,000 agents that are faster than the wall-clock time with 24 NVIDIA A800 GPUs and the performance grows linearly with the increase of LLM computational resources. We also show that the integration of realistic environments significantly enhances the authenticity of the agents' behaviors. Through the framework and experimental results, we are confident that deploying large-scale LLM Agents to simulate human societies becomes feasible. This will help practitioners in fields such as social sciences and management sciences to obtain new scientific discoveries via language generation technologies, and even improve planning and decision-making in the real world. The code is available at https://github.com/tsinghua-fib-lab/agentsociety/.

1 Introduction

In recent years, the rapid advancement of large language models (LLMs) has profoundly transformed the research paradigm of artificial intelligence and beyond (Zhao et al., 2023). One of the most important directions is the agent-based modeling (ABM) driven by LLMs (Gao et al., 2024a). Traditional ABM approaches, which rely on predefined rules and simplified environments, have achieved significant success in simulating macro-level social evolution phenomena, such as the phenomenon of segregation in society (Schelling, 1971) and polarization of opinion (Deffuant et al., 2000). This success is built upon researchers' comprehension of macroscopic principles governing human societies. Meanwhile, the powerful role-play capabilities of LLMs (Park et al., 2023; Jiang et al., 2024; Strachan et al., 2024; Li et al., 2024) empower researchers to re-examine ABM from a novel perspective: LLMs can be used to simulate complex behavioral patterns of individuals without the need for predefined rules, which can help us move beyond the traditional coarse-grained modeling paradigm and reconstruct microscopic and more realistic dynamics of human societies.

As the famous sociologist George Herbert Mead stated, ”The self is something which has a development; it is not initially there, at birth, but arises in the process of social experience and activity.” (Mead, 1934) LLM agents also learn and evolve through environmental feedback. However, most existing agent-based societal simulations predominantly adopt gaming environments (Park et al., 2023; AL et al., 2024) or simple rule settings (Gao et al., 2023; Tang et al., 2024), exhibiting insufficient attention to real-world human societal environments. This limitation inevitably constrains the authenticity of LLM agents' behaviors. Therefore, constructing **realistic environments** capable of providing feedback similar to human societies emerges as the primary challenge in leveraging LLM agents to simulate human societies.

Furthermore, in simulating such a complex system as human society, the scale serves as a prerequisite for the emergence of phenomena and the discovery of principles. Concurrently, societal simulations inherently involve massive and

[*] Yong Li is the Corresponding Author.

non-deterministic interactions between agents and environments, as well as among agents themselves. However, existing LLM agent programming frameworks are primarily designed for multi-agent collaboration scenarios and struggle to handle large-scale uncertain interactions in simulations. For example, CAMEL (Li et al., 2023) only implements the simulation of a Hackathon Judge Committee with fewer than 10 participants. AgentScope (Gao et al., 2024b), on the other hand, has only achieved a scale of tens of thousands of agents in extremely simple games such as the 2/3 number guessing game. Thus, there is an urgent need for **a framework with strong parallel execution and interaction processing capabilities** to accommodate the complex and non-deterministic interactions required for simulating human societies.

To address the aforementioned challenges, we design a scalable framework named **AgentSociety**, which integrates **realistic societal environments** capable of modeling mobility behaviors, social interactions, and economic activities, along with a **parallelized interaction engine** supporting the execution and interaction of large-scale LLM agents. Comprehensive performance experiments validate that the framework efficiently handles complex interactions while fully leveraging available LLM computational resources to simulate 30,000 LLM agents with 24 NVIDIA A800 GPUs that are faster than the wall-clock time. Meanwhile, the performance grows linearly with the supply of computational resources for LLMs. By deploying properly designed agents, the framework demonstrates its ability to provide agents with contextually appropriate environmental feedback, thereby enhancing the authenticity of agents' behaviors in a simulation scenario for urban resident behaviors in Beijing. Accordingly, we are confident in the feasibility of deploying large-scale agents to simulate human societies, which will help practitioners in social sciences, management sciences, and other fields to use language generation technologies to make new scientific discoveries and even improve real-world planning and decision-making.

2 Realistic Societal Environments

2.1 Overall

Realistic societal environments, which serve as the foundation for simulating LLM agents as a human society, aim to provide agents with feedback and constraints similar to the real-world society, thereby fostering agent learning and the emergence of more realistic behavioral patterns. Given the complexity of diverse human behaviors, explicitly modeling the most fundamental behaviors facilitates this preliminary effort. Thus, we prioritize explicit modeling of mobility behaviors, social interactions, and basic economic activities—specifically employment and consumption—as representative components. By modeling these three categories of social behaviors, the environment enables the simulation of individuals' daily routines, such as commuting to work by car, collaborating with colleagues in workplaces, and engaging in post-work consumption activities, etc.

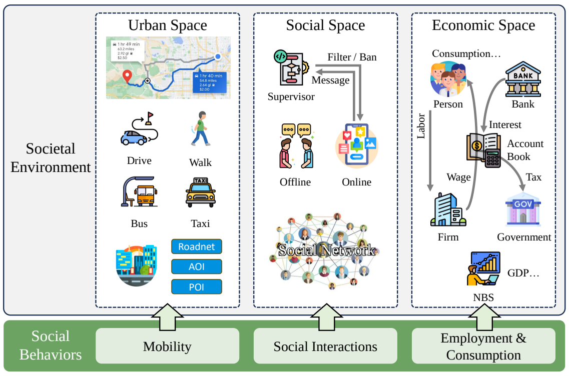

Figure 1: The relationship between the societal environments and agent behaviors.

To model these behaviors and provide realistic feedback, the built environments include urban space, social space, and economic space as illustrated in Figure 1. Their modeling and interactions will be discussed in the following sections.

2.2 Urban Space

The urban space is designed to address agents' mobility demands and their interactions with different places, capturing the processes of individual location changes driven by mobility behaviors.

Inspired by microscopic traffic simulation platforms (Behrisch et al., 2011; Zhang et al., 2019, 2024), we first build maps including road networks and functional zones, which are Areas of Interest (AOIs) and Points of Interest (POIs) by the MOSS toolkit (Zhang et al., 2024). The real-world data sources include OpenStreetMap[1] and SafeGraph[2]. Agents can retrieve accessible places from the map and obtain routes along with the estimated travel time for different transportation methods to help

[1] https://openstreetmap.org/
[2] https://www.safegraph.com/

them make better decisions about the destination and mode of travel. Furthermore, we implement a high-performance multi-modal mobility simulator in Golang[3], including driving, walking, taking buses, or riding in taxis, through a discrete time-stepping mechanism with 1-second step intervals. The simulator updates agents' states like positions at each step and allows agents to adjust travel plans through interactions with the environment while continuously accessing real-time states as feedback via gRPC[4].

2.3 Social Space

The social space, which models the social behaviors among agents, is also a fundamental component required for simulating human societies.

The most important element of the social space is social networks. Social networks store relationships and strengths between agents for social interaction target selection. During the simulation, agents can modify these relationships and strengths on their own to change the social network and future social behaviors. Social behaviors within the social space can be categorized into offline and online interactions. By enabling message exchange between any two agents, both offline and online social interactions are unified into a consistent implementation. Agents may select targets and send messages either through spatial proximity relationships or social networks, thereby accomplishing the two types of interactions, respectively. Agents can also receive messages and respond appropriately, such as replying to messages or changing their current actions.

Besides, to realistically simulate online social media platforms, we also implement the concept of the supervisor in the messaging system. The supervisor will identify content in online social messages, filter messages according to user-specified algorithms, and support the blocking of specific users or connections, thereby simulating the intervention process of social media platforms in information propagation.

2.4 Economic Space

The economic space includes the modeling of key elements in the macroeconomics (Wolf et al., 2013; Li et al., 2024) to simulate basic economic activities represented by employment and consumption.

In the economic space, agents serve as the most fundamental participants, obtaining wages through labor to cover consumption and fulfill their needs. Correspondingly, firms are modeled to provide job positions and distribute wages. Employment relationships can be dynamically adjusted by individuals or firms during the simulation process to model employee turnover behavior in the real world. Banks pay interest on deposits from individuals or firms, while the government levies taxes on income. Both interest rates and tax policies are adjustable during simulations. The National Bureau of Statistics (NBS) is implemented to compile macroeconomic indicators, such as GDP, average working hours per person, etc. Such designs, similar to real economic systems, require agents to carefully balance the relationship between work and consumption to avoid overspending, rather than engaging in unconstrained behaviors that are inconsistent with their predefined roles.

The aforementioned processes are implemented as an account-book-centered economic simulator in Golang, which provides all participants with the capability to adjust deposit increments and decrements. This simulator also facilitates the management of employment relationships, automated processing of interest and tax calculations, and automated computation of macroeconomic indicators. Additionally, it offers comprehensive query and modification interfaces for these functionalities.

3 Parallelized Interaction Engine

3.1 Overview

Facing the demand for executing large-scale LLM agents and processing massive and non-deterministic interactions in simulations, existing LLM agent programming frameworks are difficult to handle simultaneously due to their reliance on predefined standard operating procedures (SOP).

To address the overcome, we redesign the parallelized interaction engine by drawing inspiration from real-world societal structures. In the real world, individuals make decisions through independent reasoning and collaborate via linguistic communication. Consequently, in our design, each agent operates as an independent execution unit while influencing others through message passing in the social space. Guided by this principle, we implement parallelized agent execution using the Ray framework (Moritz et al., 2018), construct a high-performance **agent messaging system** leveraging

[3] https://go.dev/

[4] https://grpc.io/

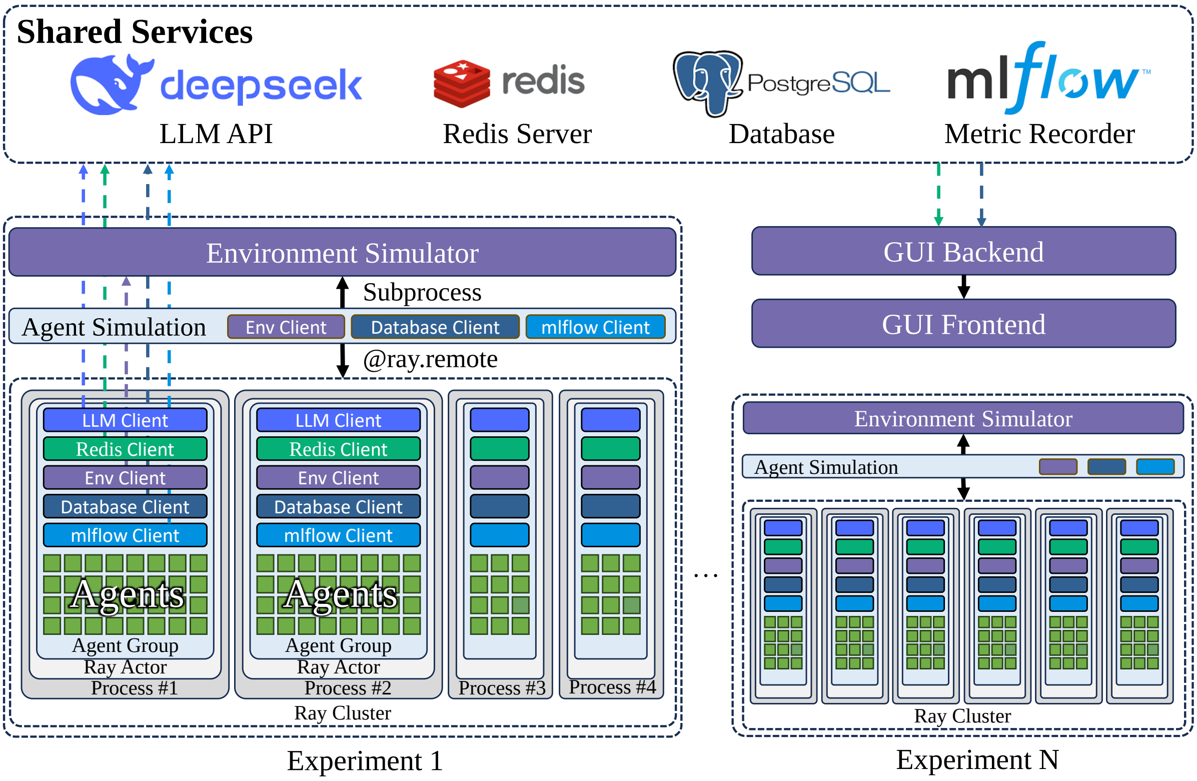

Figure 2: The architecture of the parallelized interaction engine.

Redis' publish/subscribe capabilities for message exchange, and integrate the realistic societal environment simulations as remote function calls.

However, preliminary attempts at functional integration revealed critical failures. Excessive Ray actors and network service clients rapidly exhaust machine memory and port resources, while environment access through function calls causes inconsistent perceived time progression in simulations due to variable LLM inference latencies per simulation step. To resolve these issues, we further develop **group-based parallel execution** to optimize resource utilization and adopt **time alignment mechanism** from Mirage (Zhang et al., 2022) to ensure fixed-duration environmental progression per simulation step. Finally, we provide comprehensive utilities to enhance user experiences, such as simulation logging using PostgreSQL[5] and metric recording using mlflow (Zaharia et al., 2018).

The overview of the final system architecture is shown in Figure 2. The critical components of the design will be discussed in subsequent sections.

3.2 Group-based Parallel Execution

Since each Ray actor corresponds to a worker process with independent TCP connections to various services, scaling the number of agents to tens of thousands will exceed system TCP port limits, causing program errors that prevent new connections from being established. Concurrently, the massive number of processes also induces severe memory insufficiency issues.

To address these issues, we adopt the group-based distributed execution strategy. We first evenly partition agents into multiple agent groups and make each group correspond to a Ray actor. Agents within the same group share a set of service clients and leverage asyncio's asynchronous capabilities to perform parallel network requests with connection reuse to optimize resource utilization.

Since LLM agent execution is essentially an IO-intensive processing task, this approach successfully maintains efficient parallel execution while significantly reducing port occupation and additional memory consumption caused by multi-process overhead.

3.3 Agent Messaging System

Based on the design of the social space and the parallelized interaction engine, the agent messaging system should support message exchange between any pair of agents. Such design can also enable external programs (e.g., GUIs) to send messages to

[5] https://www.postgresql.org/

specific agents for dialogues or interviews, which could significantly expand the framework's application potential.

In practice, we utilize the time-tested Pub/Sub functionality of the Redis database to build a high-performance message exchange mechanism. During simulations, each agent adopts the `PSUBSCRIBE` method to subscribe to the channel pattern `exps:<exp_id>:agents:<agent_id>:*` via a shared Redis client, enabling them to receive and process messages. The wildcard `*` is replaced by specific patterns (e.g., `agent-chat` for inter-agent messaging or `user-chat` for user-agent interactions) on the publisher's side when calling the `PUBLISH` method. This design ensures that the agent messaging system can readily support various future extensions, enriching agents' interaction capabilities.

3.4 Time Alignment Mechanism

Since the execution time of LLM agents is constrained by the response speed of LLM APIs, which fluctuates significantly due to server load, the duration required for completing one agent iteration becomes uncontrollable. Concurrently, the clock speed within the environment also varies with operational efficiency. The mismatch between these two factors will result in uncertainty regarding the elapsed time between consecutive agent iterations, thereby compromising the reproducibility of simulation outcomes.

Following Mirage (Zhang et al., 2022), we implement a clock manager and embed it into the environment simulator. Each round of agent iteration is required to take time alignment with the environment simulator to synchronize their operational speeds. The default setting maps one round of agent iteration to 300 steps (equivalent to 300 seconds) in the environment simulator, balancing behavioral authenticity with execution efficiency.

3.5 Utilities

In addition, we also provide a rich set of utilities to facilitate the usage of the framework including LLM API adapters, a JSON parser, a retry mechanism, a metric recorder based on mlflow, simulation result logging using both the local file storage with the AVRO format[6] and PostgreSQL databases. A GUI program has been developed to create and manage simulations, and visualize results stored in the PostgreSQL database, significantly enhancing usability and making the system more accessible to general users.

4 Experiments

The experiments in the section focus on the following research questions:

- RQ1: What is the performance of the framework for different agent sizes, agent group sizes, and LLM computational resources?
- RQ2: Can the realistic societal environments enhance the authenticity of agent behavior?

All experiments were conducted on a Huawei Cloud c7.16xlarge.4 cloud server to ensure comparability of results. The LLMs operate on multiple servers with 8 NVIDIA A800 cards using vLLM v0.8.1 (Kwon et al., 2023) and the Qwen2.5-7B-Instruct model (Yang et al., 2024). The details of the deployment can be found in Appendix A.

4.1 Framework Performance

To evaluate the performance of the proposed framework AgentSociety in practical deployments, we conducted a series of experiments with the agent design above to capture various metrics during system operation under different configurations of agent numbers, group numbers, and LLM computational resource provisioning.

First, we evaluated the results of {1000, 3000, 10000, 30000} agents under {4, 8, 16, 32} groups, reporting the results in Table 1. Collected metrics include runtime statistics and time costs. Besides, we counted the average input tokens and output tokens requested by LLMs. The results are very close in all cases, being 347.97 ± 0.80 and 62.30 ± 0.42 respectively. The results show that the framework achieves faster than real-time simulation at the scale of 30,000 agents, demonstrating the parallel performance of the framework. Additionally, it can be observed from the results that the simulation efficiency mainly depends on the efficiency of LLM calls. Moreover, an increase in the number of groups, on one hand, enhances the efficiency of environment calls, while on the other hand, it may lead to exceeding the load capacity of LLM services, thereby increasing unnecessary retry time. This highlights the importance of reasonably setting the degree of parallelism according to the supply of LLM services.

Second, we evaluated the performance of sim-

[6] https://avro.apache.org/

Table 1: Performance metrics for different configurations using the 24-GPU vLLM cluster as LLM providers. All values are the means of 10 rounds of iterations, standard deviations are not provided due to their distribution not being normal. **#LC** represents the number of successful LLM calls. **LCSR** stands for LLM Call Success Rate and is used to record the percentage of all LLM requests attempted to be called that are returned correctly. **#EC** and **#MC** denotes the number of environment simulator call and agent message system call, respectively. In the time cost part, **All** denotes the average time taken by all the agents to iterate a round. **LLM**, **Env**, and **Msg** represents the time spent for each LLM call, environment simulator call, and agent message system call, respectively. The dash (-) indicates experimental failure due to excessive failed LLM requests.

Parameters		**Runtime Statistics**				**Time Costs**			
#Agents	**#Groups**	**#LCs (/round)**	**LCSR(%)**	**#ECs (/round)**	**#MCs (/round)**	**All (s/round)**	**LLM (s/call)**	**Env (ms/call)**	**Msg (ms/call)**
1,000	4	995.5	100.0	7,547.0	2.0	**13.19**	4.60	88.01	2.25
1,000	8	992.5	100.0	7,547.4	1.9	13.70	4.59	44.61	1.16
1,000	16	987.0	100.0	7,529.9	2.3	13.21	4.77	21.26	0.79
1,000	32	988.5	100.0	7,513.2	2.3	13.96	4.70	10.87	0.81
3,000	4	2,963.9	100.0	22,567.9	7.0	31.87	13.80	219.12	1.98
3,000	8	2,977.7	100.0	22,644.2	7.2	**28.98**	13.15	103.57	1.30
3,000	16	2,975.7	85.2	22,594.4	5.6	33.64	14.32	55.20	1.43
3,000	32	2,978.4	86.3	22,601.5	6.9	34.70	14.95	28.63	1.14
10,000	4	9,905.1	100.0	75,335.1	21.9	93.10	44.96	943.61	8.35
10,000	8	9,885.8	100.0	75,291.7	22.9	**81.45**	42.75	349.49	3.59
10,000	16	9,897.0	97.1	75,343.9	22.2	98.08	41.01	208.75	3.48
10,000	32	-	-	-	-	-	-	-	-
30,000	4	30,686.3	100.0	230,309.9	83.0	327.39	130.58	4,102.45	21.82
30,000	8	29,869.7	100.0	226,915.8	70.7	**251.85**	123.22	1,682.88	14.48
30,000	16	-	-	-	-	-	-	-	-

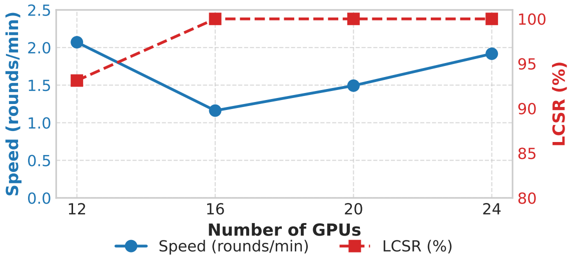

Figure 3: Performance with different LLM computational resources.

ulating 3,000 agents (with #Groups set to 8) with the same experimental setup as before, under different deployments of LLMs on {4, 8, 12, 16, 20, 24} GPUs. The experiments failed when the number of GPUs was less than or equal to 8. The remaining results are shown in Figure 3. The results indicate that, when LLM calls are always successful, the framework's performance increases linearly with computational resource supply. Instead, when some calls fail the performance is higher, possibly because appropriate failures and retries facilitate the reallocation of LLM computational demands over time, thereby enhancing overall throughput. Thus, designing appropriate LLM request scheduling is an important future work of AgentSociety.

4.2 Environment Impact

To evaluate the impact of the realistic societal environments on agent performance, we constructed a social agent and an experimental scenario. The agent is designed to simulate urban residents' behaviors, comprising a guiding module based on the Needs Model (Maslow, 1943) and Planned-Behavior Model (Ajzen, 1991), along with multi-dimensional action modules (cognition, mobility, economy, and social interactions), interconnected via stream memory and function calling. The experimental scenario integrates mobility and cognitive scenarios, constructed using mobility trajectories collected from 169 urban residents in Beijing, each accompanied by associated intention data (Shao et al., 2024).

Table 2 presents a comparative analysis of agent performance under conditions with environment support (W-Env) and without environment support (WO-Env). W-Env was conducted using the proposed environment simulator, whereas WO-Env relied on an LLM-based textual simulator whose detailed prompt implementations can be found in Appendix B.1. We also compared the results with classical generative models including TimeGeo (Jiang et al., 2016), Movesim (Feng et al., 2020), Volunteer (Long et al., 2023), DiffTraj (Zhu et al., 2023), and Act2Loc (Liu et al., 2024).

The results highlight the critical importance of the realistic societal environment, particularly reflected by data support for feasible destinations, inter-location distances, and travel durations. Performance significantly declines in mobility-related metrics (e.g., radius and dayloc) under WO-Env

Table 2: Authenticity comparison among LLM Agent simulations with/without the realistic societal environments and classical generative models. Refer to Appendix B.2 and B.3 for more details about the metrics and the distributions, respectively.

Method	Radius	Dayloc	itdNum	itdError	itdDur
TimeGeo	0.254	0.258	0.297	0.536	0.155
Movesim	0.233	0.051	0.154	0.904	0.178
Volunteer	0.455	0.049	0.318	0.804	0.162
DiffTraj	0.027	0.647	0.695	0.597	0.080
Act2Loc	0.024	0.042	0.131	0.391	0.040
WO-Env	0.427	0.129	0.158	0.241	0.091
W-Env	**0.023**	**0.038**	**0.073**	**0.094**	**0.027**

conditions. Cognitive metrics (itdNum, itdError, and itdType) also show noticeable degradation. This indicates that the absence of environmental context severely restricts agents' capacity to accurately replicate realistic human behaviors. Besides, under W-Env conditions, agents in our proposed framework demonstrate excellent performance and outperform all baseline methods, effectively capturing authentic behavior patterns.

5 Related Works

5.1 LLM Agent-driven Simulation

Existing studies have validated the feasibility of LLM agent-driven simulations across multiple dimensions. Works such as Smallville (Park et al., 2023) and Project Sid (AL et al., 2024), through agent simulations within gaming environments, have demonstrated that LLMs can exhibit anthropomorphic behaviors and generate emergent social phenomena. Meanwhile, other studies employing rule-driven environments have further validated the similarities between LLM agents and real humans in aspects such as economic behaviors (Li et al., 2024) and social interactions (Gao et al., 2023; Tang et al., 2024).

However, these works have yet to incorporate realistic environments to provide feedback similar to human societies, thereby making it difficult to conduct LLM agent-driven simulations of them.

5.2 LLM Agent Programming Frameworks

Existing LLM agent programming frameworks are predominantly oriented toward multi-agent collaboration to enhance task-specific performance. These frameworks (Hong et al., 2024; Qian et al., 2024; Gao et al., 2024b; Li et al., 2023) typically require users to design SOPs based on message dependencies among agents and orchestrate parallel execution via directed acyclic graphs (DAGs), while treating environmental interactions as external function calls for LLMs. Such designs are difficult to handle the complex and non-deterministic interactions among agents and environments. Moreover, they face significant challenges in scaling effectively under conditions of complex interactions.

Additionally, Concordia (Vezhnevets et al., 2023) has attempted to design simulation-oriented LLM agent programming architectures. However, the LLM-driven Game Master introduces a bottleneck during large-scale simulations, severely limiting their scalability and practical applicability.

Therefore, there remains an urgent demand for LLM agent programming frameworks explicitly tailored for large-scale LLM agent simulation scenarios, capable of supporting massive, dynamic, and non-deterministic interactions.

6 Conclusion

In conclusion, the proposed AgentSociety provides a scalable framework for the simulation of LLM agents by integrating realistic societal environments and parallelized interactions, supporting large-scale human society simulation with highly realistic agents' behaviors. It successfully achieved a simulation of 30,000 agents faster than real-time clock speed with 24 NVIDIA A800 GPUs. We hope that AgentSociety will come to the attention of practitioners in social sciences, management sciences, and other fields so that simulations based on LLM agents become a new driving force behind new scientific discoveries and better real-world planning and decision-making.

7 Limitation

Although AgentSociety has achieved preliminary success in supporting the simulation of human societies using LLM Agents, we believe significant work remains to achieve a comprehensive social simulation. In terms of environmental modeling, there remains a substantial gap between current economic system representations and real-world ones, such as the lack of simulations for market mechanisms and firm decision-making processes. Regarding system architecture, improving agent execution efficiency through prompt engineering or other enhancements to enable large-scale simulations remains an open challenge.

Acknowledgments

This work is supported in part by the National Key Research and Development Program of China under 2024YFC3307603 and in part by Tsinghua-Toyota Joint Research Center.

References

Icek Ajzen. 1991. The theory of planned behavior. *Organizational Behavior and Human Decision Processes*.

Altera AL, Andrew Ahn, Nic Becker, Stephanie Carroll, Nico Christie, Manuel Cortes, Arda Demirci, Melissa Du, Frankie Li, Shuying Luo, et al. 2024. Project sid: Many-agent simulations toward ai civilization. *arXiv preprint arXiv:2411.00114*.

Michael Behrisch, Laura Bieker, Jakob Erdmann, and Daniel Krajzewicz. 2011. Sumo–simulation of urban mobility: an overview. In *Proceedings of SIMUL 2011, The Third International Conference on Advances in System Simulation*. ThinkMind.

Guillaume Deffuant, David Neau, Frederic Amblard, and Gérard Weisbuch. 2000. Mixing beliefs among interacting agents. *Advances in Complex Systems*, 3(01n04):87–98.

Jie Feng, Zeyu Yang, Fengli Xu, Haisu Yu, Mudan Wang, and Yong Li. 2020. Learning to simulate human mobility. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 3426–3433.

Chen Gao, Xiaochong Lan, Nian Li, Yuan Yuan, Jingtao Ding, Zhilun Zhou, Fengli Xu, and Yong Li. 2024a. Large language models empowered agent-based modeling and simulation: A survey and perspectives. *Humanities and Social Sciences Communications*, 11(1):1–24.

Chen Gao, Xiaochong Lan, Zhihong Lu, Jinzhu Mao, Jinghua Piao, Huandong Wang, Depeng Jin, and Yong Li. 2023. S3: Social-network simulation system with large language model-empowered agents. *arXiv preprint arXiv:2307.14984*.

Dawei Gao, Zitao Li, Xuchen Pan, Weirui Kuang, Zhijian Ma, Bingchen Qian, Fei Wei, Wenhao Zhang, Yuexiang Xie, Daoyuan Chen, et al. 2024b. Agentscope: A flexible yet robust multi-agent platform. *arXiv preprint arXiv:2402.14034*.

Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. 2024. MetaGPT: Meta programming for a multi-agent collaborative framework. In *The Twelfth International Conference on Learning Representations*.

Hang Jiang, Xiajie Zhang, Xubo Cao, Cynthia Breazeal, Deb Roy, and Jad Kabbara. 2024. Personallm: Investigating the ability of large language models to express personality traits. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 3605–3627.

Shan Jiang, Yingxiang Yang, Siddharth Gupta, Daniele Veneziano, Shounak Athavale, and Marta C González. 2016. The timegeo modeling framework for urban mobility without travel surveys. *Proceedings of the National Academy of Sciences*, 113(37):E5370–E5378.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pages 611–626.

Guohao Li, Hasan Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. 2023. Camel: Communicative agents for" mind" exploration of large language model society. *Advances in Neural Information Processing Systems*, 36:51991–52008.

Nian Li, Chen Gao, Mingyu Li, Yong Li, and Qingmin Liao. 2024. Econagent: large language model-empowered agents for simulating macroeconomic activities. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15523–15536.

Kang Liu, Xin Jin, Shifen Cheng, Song Gao, Ling Yin, and Feng Lu. 2024. Act2loc: a synthetic trajectory generation method by combining machine learning and mechanistic models. *International Journal of Geographical Information Science*, 38(3):407–431.

Qingyue Long, Huandong Wang, Tong Li, Lisi Huang, Kun Wang, Qiong Wu, Guangyu Li, Yanping Liang, Li Yu, and Yong Li. 2023. Practical synthetic human trajectories generation based on variational point processes. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 4561–4571.

AH Maslow. 1943. A theory of human motivation. *Psychological Review*, 2:21–28.

George Herbert Mead. 1934. *Mind, self, and society from the standpoint of a social behaviorist.* Chicago.

Philipp Moritz, Robert Nishihara, Stephanie Wang, Alexey Tumanov, Richard Liaw, Eric Liang, Melih Elibol, Zongheng Yang, William Paul, Michael I Jordan, et al. 2018. Ray: A distributed framework for emerging {AI} applications. In *13th USENIX symposium on operating systems design and implementation (OSDI 18)*, pages 561–577.

Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. 2023. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, pages 1–22.

Chen Qian, Wei Liu, Hongzhang Liu, Nuo Chen, Yufan Dang, Jiahao Li, Cheng Yang, Weize Chen, Yusheng Su, Xin Cong, et al. 2024. Chatdev: Communicative agents for software development. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15174–15186.

Thomas C Schelling. 1971. Dynamic models of segregation. *Journal of mathematical sociology*, 1(2):143–186.

Chenyang Shao, Fengli Xu, Bingbing Fan, Jingtao Ding, Yuan Yuan, Meng Wang, and Yong Li. 2024. Chain-of-planned-behaviour workflow elicits few-shot mobility generation in llms. *arXiv preprint arXiv:2402.09836*.

James WA Strachan, Dalila Albergo, Giulia Borghini, Oriana Pansardi, Eugenio Scaliti, Saurabh Gupta, Krati Saxena, Alessandro Rufo, Stefano Panzeri, Guido Manzi, et al. 2024. Testing theory of mind in large language models and humans. *Nature Human Behaviour*, pages 1–11.

Jiakai Tang, Heyang Gao, Xuchen Pan, Lei Wang, Haoran Tan, Dawei Gao, Yushuo Chen, Xu Chen, Yankai Lin, Yaliang Li, et al. 2024. Gensim: A general social simulation platform with large language model based agents. *arXiv preprint arXiv:2410.04360*.

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. 2023. Generative agent-based modeling with actions grounded in physical, social, or digital space using concordia. *arXiv preprint arXiv:2312.03664*.

Sarah Wolf, Steffen Fürst, Antoine Mandel, Wiebke Lass, Daniel Lincke, Federico Pablo-Marti, and Carlo Jaeger. 2013. A multi-agent model of several economic regions. *Environmental modelling & software*, 44:25–43.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. 2024. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.

Matei Zaharia, Andrew Chen, Aaron Davidson, Ali Ghodsi, Sue Ann Hong, Andy Konwinski, Siddharth Murching, Tomas Nykodym, Paul Ogilvie, Mani Parkhe, et al. 2018. Accelerating the machine learning lifecycle with mlflow. *IEEE Data Eng. Bull.*, 41(4):39–45.

Huichu Zhang, Siyuan Feng, Chang Liu, Yaoyao Ding, Yichen Zhu, Zihan Zhou, Weinan Zhang, Yong Yu, Haiming Jin, and Zhenhui Li. 2019. Cityflow: A multi-agent reinforcement learning environment for large scale city traffic scenario. In *The world wide web conference*, pages 3620–3624.

Jun Zhang, Wenxuan Ao, Junbo Yan, Can Rong, Depeng Jin, Wei Wu, and Yong Li. 2024. Moss: A large-scale open microscopic traffic simulation system. *arXiv preprint arXiv:2405.12520*.

Jun Zhang, Depeng Jin, and Yong Li. 2022. Mirage: an efficient and extensible city simulation framework (systems paper). In *Proceedings of the 30th International Conference on Advances in Geographic Information Systems*, pages 1–4.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 1(2).

Yuanshao Zhu, Yongchao Ye, Shiyao Zhang, Xiangyu Zhao, and James Yu. 2023. Difftraj: Generating gps trajectory with diffusion probabilistic model. *Advances in Neural Information Processing Systems*, 36:65168–65188.

A vLLM Deployment

We deploy a vLLM cluster across 3 servers, each equipped with 8 NVIDIA A800 40GB GPUs, 128-thread processors, and 1024GB RAM. The deployed model is Qwen2.5-7B-Instruct, with automatic tool selection and chunked prefill enabled, configured with `max-num-batched-tokens` set to 8192 (without extensive tuning). The guided decoding backend uses outlines. We do not enable tensor parallelism. Instead, we independently run a vLLM instance on each GPU and construct a reverse proxy supporting round-robin load balancing through Caddyserver[7] as the access endpoint. Our program accesses this endpoint to invoke the LLM computation services provided by vLLM.

B Supplementary Materials Regarding the Environment Impact Experiments

B.1 LLM-based Textual Simulator Prompts

As an alternative to the realistic simulation environment, we designed the following prompts to leverage the existing knowledge of LLMs to achieve functions including text-based location type selection, destination selection, and travel time estimation to support the agent's mobility behavior simulation.

Place Type Selection: This prompt assists the agent in determining the appropriate type of location to visit, based on its current needs and internal states.

```
    You are an intelligent assistant specializing in understanding user needs and suggesting appropriate location types. Based on the user's intention, provide the most suitable location type.

- User's intention: {intention}

Please output in JSON format without any other text:

{
"type": "string", location type
}

Example Output:
{
"type": "Grocery Store"
}
```

Destination Selection: This prompt guides the agent in selecting a specific destination, given its current location and desired location type. It also includes information regarding the distance between these two locations.

```
    You are an intelligent assistant specializing in suggesting specific destinations based on location types. Provide a suitable location name and estimate its distance from the current position.

- Current location: {current location}
- Target location type: {place type}

Please output in JSON format without any other text:
{
"name": "string", locations' name
"distance": "integer", in meter
}

Example Output:
{
"name": "Supermarket",
"distance": 1500
}
```

Travel Time Estimation: This prompt estimates the time required for the agent to reach the selected destination, considering both the current environmental conditions and the agent's status.

```
    You are an intelligent assistant specializing in travel time estimation. Based on the provided distance, calculate the estimated time required to reach the destination, assuming typical traffic conditions.

- User's profile: {agent profile}
- Weather: {weather}
- distance: {distance} m

Please output in JSON format without any other text:
{
"time": "integer", in minutes
}

Example Output:
{
"time": 10
}
```

[7] https://caddyserver.com/

B.2 Metrics

The specific meanings of the five metrics used in the experiments are as follows:

- Radius: radius of gyration, representing the spatial dispersion of an agent's movements;
- Dayloc: daily visited locations, indicating the number of unique locations visited each day;
- itdNum: the number of intentions per day, measuring daily intention frequency;
- itdError: the similarity between intention sequences, reflecting consistency in agent behaviors;
- itdType: time proportion of intentions, denoting the temporal distribution of different intentions.

B.3 Distribution Details

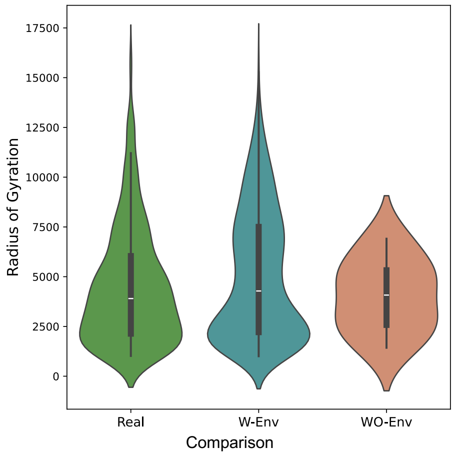

Figure 4: Distribution of Radius of Gyration.

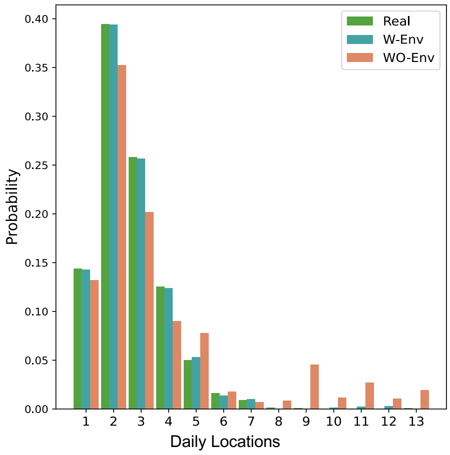

Figure 5: Distribution of daily locations.

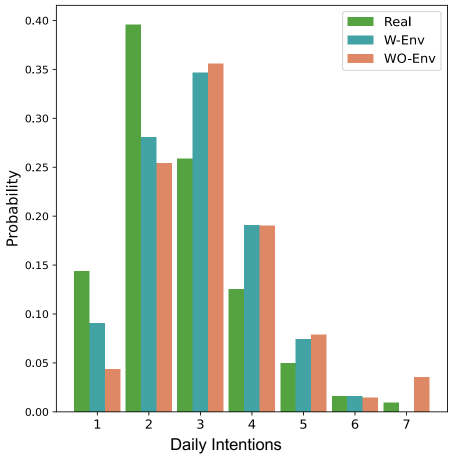

Figure 6: Distribution of daily intentions.

This section provides the distribution details for the experiments in Section 4.2. From these results, we can see that the information and constraints introduced by the realistic societal environments significantly improve the movement behavior patterns of the agents, making them highly approximate to the real data (Figure 4 and 5) And bring about a certain improvement in the distribution of intentions (Figure 6).