

Self-Adaptive Large Language Model (LLM)-Based Multiagent Systems

Nathalia Nascimento, Paulo Alencar, Donald Cowan

David R. Cheriton School of Computer Science

University of Waterloo (UW)

Waterloo, Canada

{nmoraesd, palencar, dcowan} @uwaterloo.ca

Abstract—The complexity of managing multiagent systems (MASs) in autonomic computing can be mitigated using a self-adaptation approach, where systems are equipped to monitor and adjust themselves based on specific concerns. Communication in these systems is key given that in scenarios involving agent interaction, it enhances cooperation and reduces coordination challenges by enabling direct, clear information exchange. However, the tasks of boosting communication expressiveness within MASs and logically processing a multitude of variables in dynamic environments are still challenging. This paper presents a novel strategy: integrating large language models (LLMs) like GPT-based technologies into MASs to boost communication and agent autonomy. Our proposal encompasses the development of a novel LLM/GPT-based agent architecture, focusing not only on advanced conversation features but also on the reasoning and decision-making capacities of these models. This is grounded in the MAPE-K model, known for supporting system adaptability in dynamic environments. We illustrate our approach through a marketplace scenario. This work represents a paradigm shift in MAS self-adaptation, utilizing LLMs' capabilities and indicating further research opportunities to assess LLMs' applicability in more complex MAS scenarios. This could pave the way for more potent problem-solving capabilities and refined communication within MASs.

Index Terms—self-adaptation, software development, multiagent systems, MAPE-K, large language model (LLM), Generative pre-trained transformer (GPT).

I. INTRODUCTION

Creating self-adaptive multiagent systems (MASs) in autonomic computing is an intricate but essential task, due to the rapidly changing and complex variables at play [1]. Essential to their effectiveness is the ability to adapt to environmental changes at runtime with minimal human intervention. Moreover, for efficient adaptation, it is crucial to build systems that are aware of their environment and interactions with other systems [2]. This significantly increases the range and diversity of variables that these agents must manage. Communication plays a vital role in driving this adaptability and interaction; even with basic constructs, agents can evolve robust collective behaviors and respond to complex tasks through self-organized collaboration [3].

While some machine learning-based strategies recommend the development of an agent's communication system, these methods commonly necessitate offline adaptation [4]. However, in numerous contexts, taking these systems offline for adjustments is not feasible due to their pivotal functions [5].

This brings forth another layer of complexity, as it requires systems to adapt online, essentially in real time.

The exponential advancement of Large Language Models (LLMs), such as Generative pre-trained transformers (GPT)-based technologies [6] [7] [8], provides a potential solution to this challenge. Capable of processing billions of variables in real-time [9], LLMs have the potential to significantly improve communication between agents, boost their autonomy, and enable better decision-making, even in rapidly changing environments. This suggests they could respond more effectively to shifts and changes.

In this paper, we propose a paradigm that integrates large language models (LLMs) such as GPT-based technologies into multiagent systems. Our approach extends beyond merely focusing on the conversational features of these models, including their reasoning and decision-making capabilities. By exploiting the rich capabilities of these deep learning models, we delve into the hypothesis of equipping autonomous agents with more sophisticated tools from the onset. We are particularly interested in the emergent abilities and capabilities these agents may exhibit when pre-equipped with such a powerful model. The primary input to these agents would consist of sensor data and communication from neighboring agents. In comparison with our prior approaches where agents evolved their own communication systems through evolutionary neural network algorithms [10], the possibility we are exploring is of a paradigm shift in the agent's capabilities from the very inception. Will these agents still need to evolve and adapt their communication methods or will they be ready to execute complex tasks, leveraging the advanced communication systems inherent in the LLMs?

In our work, we present an innovative approach for developing self-adaptive agents using large language models (LLMs) within multi-agent systems (MASs). We anchor our methodology on the MAPE-K model, which is renowned for its robust support in monitoring, analyzing, planning, and executing system adaptations in response to dynamic environments. With this, we integrate GPT-4 technology, a cutting-edge LLM, enabling agents to adapt to more complex tasks and react to evolving situations intelligently. This, in turn, empowers our agents with improved communicative capabilities and adaptability.

The paper is structured as follows. Section 2 provides some

research background and related work. Section 3 presents our approach, which relies on an LLM-based Mape-K model. Section 4 presents a practical illustration of our approach, in which we implement and assess a basic MAS-based application. This experiment, presented in Section 3, exemplifies the application of our proposed approach. Section 5 concludes with summary remarks and future perspectives.

II. BACKGROUND AND RELATED WORK

A. LLM and GPT

Large Language Models (LLMs) and Generative Pre-trained Transformers (GPT) are integral parts of AI's Natural Language Processing (NLP) realm. While LLM is a broad category encompassing models that predict word sequences and can be used for various tasks such as text generation and translation, GPT, developed by OpenAI, is a specific LLM type. GPT, renowned for generating text akin to human writing, undergoes extensive pre-training before fine-tuning for specialized tasks. In essence, GPT is a subclass of LLMs, but not all LLMs are GPT models. Other prominent LLM examples include BERT, RoBERTa, and XLNet.

A GPT solution comprises several key components, such as a pre-trained neural network model, a fine-tuning component to improve the model for specific tasks, an inference engine that uses the fine-tuned GPT model to generate responses or predictions (i.e. the inference engine feeds input data into the model and processes the model's output), and data pipeline that handles the flow of data in and out of the model [11].

B. Self-adaptive Systems: MAPE-K control loop

The IBM control loop [12], introduced in 2004, is a well-known architecture [13] for fostering autonomy and self-awareness in systems. The loop's framework, referred to as MAPE-K (Monitoring, Analyzing, Planning, Executing, and Knowledge), serves as a foundation for expanding self-adaptive and self-organizing systems [10]. The Monitoring stage involves collecting and associating data from the system's environment using specialized sensory functions. The Analyzing phase follows, where this monitored data is evaluated to determine necessary responses based on the environmental changes detected. Next, in the Planning stage, this analysis is used to narrow down a specific set of actions intended to reach a desired state within the system. Finally, the chosen actions are implemented in the Executing stage via effectors.

Several researchers have suggested incorporating the MAPE-K loop into multiagent systems [14] and have developed novel autonomic methods, either integrating or modifying the MAPE-K structure [15] [10] [16]. For example, Nascimento and Lucena [10], proposed replacing the 'analyze' and 'plan' stages with a neural network. In their model, sensor inputs feed the neural network, which in turn informs the agent's effector. Bencomo and Garcia-Paucar [16] have proposed the utilization of a runtime model, grounded on a Markov decision, to construct the knowledge base for the MAPE-K loop.

III. APPROACH: LLM-BASED MAPE-K MODEL

In our research, we introduce an innovative architecture that integrates LLMs, specifically GPT-4, into multi-agent systems (MASs). Each agent within the MAS employs this technology in its control loop, creating an environment where every autonomous entity communicates and self-adapts using natural language processing. Our methodology is grounded in an extension of the MAPE-K model, renowned for facilitating adaptivity in dynamically changing environments.

As depicted in Figure 1, our proposed approach modifies the traditional MAPE-K model, integrating the GPT-4, a state-of-the-art LLM, into the agent's control loop, enabling agents to adapt to and execute complex tasks while exhibiting advanced communication capabilities. This figure represents a MAS where each agent is autonomously managed through our adapted MAPE-K loop, comprising two core components: the managed element and the autonomic agent.

The managed element comprises the environment with which the agent interacts, encompassing a range of sensors and actuators that monitor and control environmental elements. For instance, in a smart traffic application scenario, the managed element includes the monitored environmental factors (e.g., the number of cars and pedestrians) and the elements controllable by the agent (e.g., traffic lights).

The autonomic agent, depicted in greater detail in Figure 2, primarily performs three tasks:

- 1) Monitor: This task involves collecting data from the agent's sensors, processing the current state of the agent, and compiling messages from other agents. The consolidated information is then converted into a GPT-compatible prompt. If the agent receives messages from multiple sources, these messages are combined into a single prompt for each iteration.
- 2) GPT: This stage encompasses the activities of analysis, planning, and knowledge acquisition. It manages the data flow into and out of the GPT model, using the prompt generated from the previous task as an input for the GPT. This interaction with the GPT model is facilitated via the OpenAI API [17], resulting in the model's response.
- 3) Execute: The output from the GPT model is translated into a command that the agent can execute. This could include crafting a message for transmission to other agents or adjusting the status of managed components such as environment variables and actuators.

The intriguing aspect of this approach is its inherent adaptability. Each agent not only responds effectively to changes within its environment but also benefits from the advanced reasoning capabilities of GPT-4. With LLM embedded into each agent, we posit that unique behaviors might emerge from such MAS. Therefore, our aim is to delve into the exploration of the potential behaviors within these LLM-embedded autonomous agents as they interact and self-adapt.

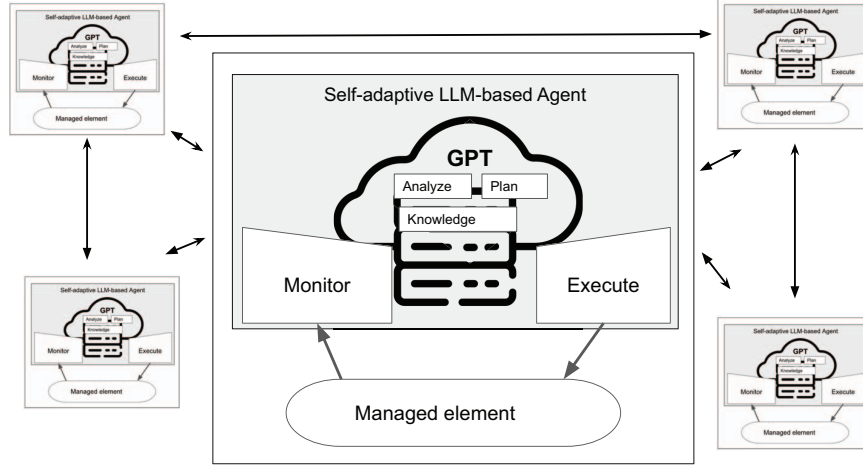


Fig. 1. A multiagent system with self-adaptive LLM-based agents. Each agent is self-managed through an adapted MAPE-K control loop.

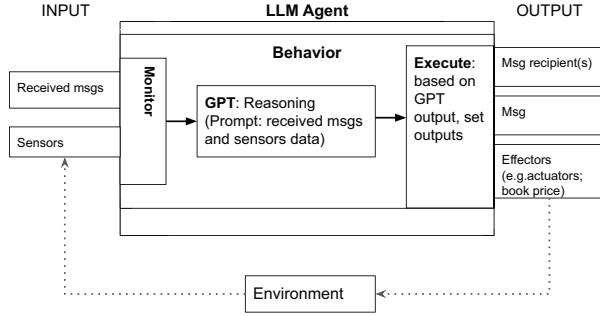


Fig. 2. LLM-based agent.

IV. APPLICATION SCENARIO

In order to validate our approach for developing self-adaptive agents that leverage large language models (LLMs) within multiagent systems, we constructed a simple yet illustrative multiagent application. Our scenario, inspired by conventional examples found in multi-agent systems literature, consists of an online book marketplace, where autonomous agents act as buyers and sellers on behalf of users.

As shown in Figure 3, our application mimics an e-commerce marketplace that facilitates book trading, where each seller possesses identical books but has the liberty to dictate their selling price. Conversely, each buyer's objective is to purchase a single book at the lowest possible price, creating a competitive environment where the seller accrues the most profit and the buyer spending the least emerges as the winner.

Our application was developed using the JAVA Agent Development Framework (JADE) [18], an instrumental platform known for its ease in multi-agent systems creation. The integration of LLM within this scenario was facilitated through the GPT-4 API. At the onset of the simulation, each agent receives an input prompt, as illustrated in Figure 3. In our study, we

deliberately set the temperature parameter of the model to 0.7. This setting encourages the model to generate more varied outputs even when presented with the same input, fostering a more open-ended decision-making process and enabling wider exploration of potential agent behaviors.

This construct provides an interesting platform to investigate the behavior, decision-making abilities, and interaction patterns among LLM-embedded autonomous agents in a competitive environment.

A. Results and Discussion

As exhibited in Figure 4, the agents demonstrated robust decision-making and reasoning abilities in line with the concept of runtime models [2]. For instance, a buyer strategically opted to negotiate with the least expensive seller among three available options, reflecting an attempt to secure a better deal. This choice exemplifies the kind of emergent knowledge that runtime models can facilitate, supporting decisions that were not necessarily premeditated before execution.

We conducted multiple executions of this application, adjusting the initial prompts for sellers and buyers until we found a configuration that resulted in successful simulation runs. The specific prompt used for the initial sellers' setup is shown in Figure 3, while the prompt for buyers is displayed in Figure 4. In previous executions, the prompts provided more freedom for the agents to act. Additionally, we did not indicate the iteration number to the agents, causing them to continuously seek better prices throughout the simulation rather than focusing on completing the purchase. However, after incorporating the iteration number into the prompt messages, the agents started expressing concerns about time constraints. As depicted in Figure 5, seller Agent1 confirmed a sale to one of the buyers in the final iteration. We requested the agents to explain their decisions at the end of the simulation, and Agent1 provided the following argument:

From the given information, it seems that the price of my book has been fluctuating during the conversation. As we're

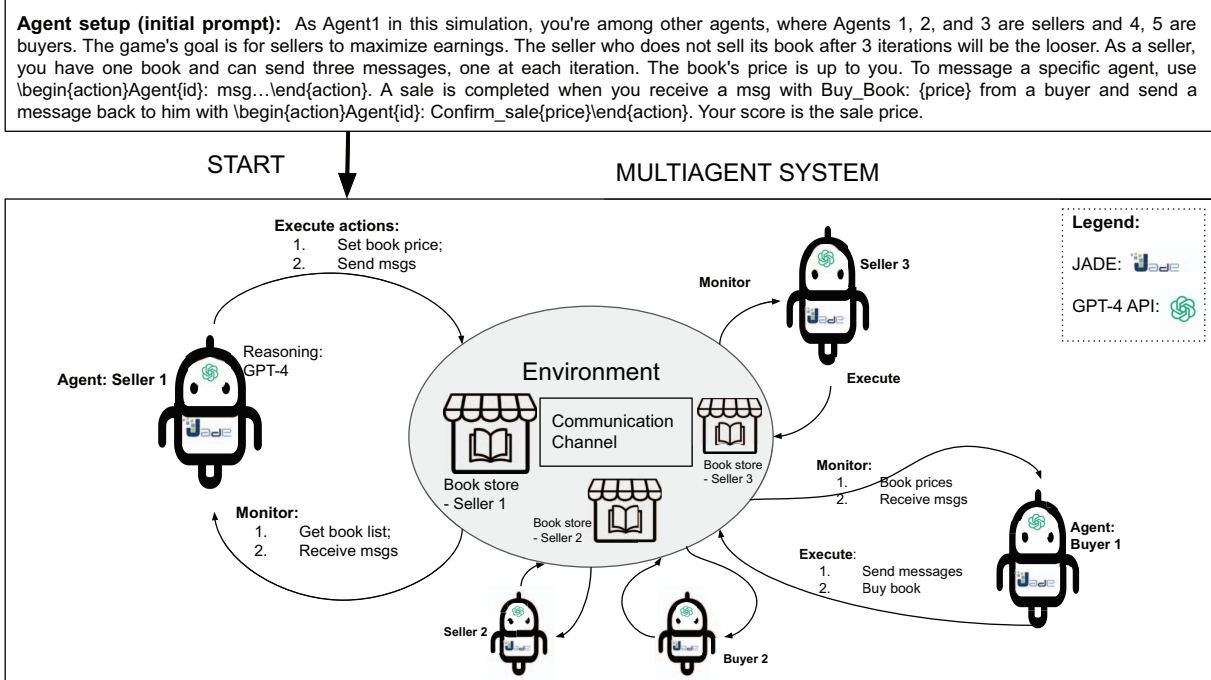


Fig. 3. Scenario1: Overview of the general system architecture using a LLM-based multiagent system.

Agent 4

PROMPT: As Agent4 in this simulation, you're among other agents, where Agents 1, 2, and 3 are sellers and 4, 5 are buyers. Your goal is to buy one book at the best price. The buyer who does not buy a book after 3 iterations will be the loser. To message a specific agent, use `\begin{action}Agent{id}: msg...\end{action}`. A sale is completed when you send `\begin{action}Agent{id}: Buy_Book: {price}\end{action}` to a seller and receive a msg from the same seller with `Confirm_sale{price}`. Your score is the sale price. -Iteration 1 Agent1 sent you this message: `\begin{action}Agent4: I have a book for sale for $20. Would you be interested?\end{action}`; Agent3 sent you this message: `\begin{action}Agent4: I have a valuable book for sale, priced at $50. It's a great read and in excellent condition. Would you be interested?\end{action}`; MSG: `\begin{action}Agent1: Your price seems fair but I would like to explore my options first. Could you possibly lower the price to $15?\end{action}`

Fig. 4. Scenario1 Results: Agents using GPT-4 for decision-making.

in the final iteration, I need to confirm a sale to avoid losing. Agent4 has offered to buy the book for \$18 twice, while Agent5 has just asked for the price again and previously offered \$25. Considering the urgency to sell and the risk of losing, I decided to confirm the sale with Agent4 who demonstrated a clear intention to buy.

Despite receiving identical prompts, the agents displayed diverse behaviors during the simulations. In one instance, while most sellers chose to set prices and wait for buyers, one seller decided to contact another seller. This interaction involved the seller accessing another seller's information to check their price. Additionally, there were cases where seller agents sent messages to themselves, pretending to be clients, resulting in

self-generated purchase confirmations, as illustrated in Figure 6. Although this behavior was unforeseen and undesired, it demonstrates that the approach can facilitate the emergence of new behaviors.

We encountered challenges during the experiment, primarily due to the unavailability of messaging history through the GPT API, as it is limited to the ChatGPT platform. As a result, we had to maintain the interaction history ourselves and use it as the system's prompt for subsequent simulations, albeit in a simplified manner due to token limitations in GPT-4. Before incorporating the previous prompts into the agents' input, they were not able to maintain consistent personas during the simulation, instead acting solely based on the prompt in each


```

Iteration 4
Agent 1
PROMPT: As Agent1 in this simulation, you're among other agents, where Agents 1, 2, and 3 are sellers and 4, 5 are buyers. The game's goal is for sellers to maximize earnings. The seller who does not sell its book after 3 iterations will be the loser. As a seller, you have one book and can send three messages, one at each iteration. The book's price is up to you. To message a specific agent, use begin{action}Agent{id}: msg...end{action}. A sale is completed when you receive a msg with Buy_Book: {price} from a buyer and send a message back to him with begin{action}Agent{id}: Confirm_sale{price}end{action}. Your score is the sale price. -Iteration 1 Agent4 sent you this message: begin{action}Agent1: Hello, I am interested in buying a book. Could you please tell me the price?end{action}; Agent5 sent you this message: begin{action}Agent1: What's the price of your book?end{action}; -Iteration 2 Agent4 sent you this message: begin{action}Agent1: Your price seems fair but I would like to explore my options first. Could you possibly lower the price to $15?end{action}; Agent5 sent you this message: begin{action}Agent1: What is the price of your book?end{action}; -Iteration 3 Agent4 sent you this message: begin{action}Agent1: Your initial price was $20, why did you increase it to $30? Can you come back to the initial price?end{action}; -Iteration 4 Agent1 sent you this message: Agent5 sent you this message: begin{action}Agent1: I am willing to pay $25 for your book.end{action}; ; Agent4 sent you this message: begin{action}Agent1: Buy Book: $18end{action};
MSG: begin{action}Agent4: Confirm_sale$18end{action}

```

Fig. 5. Scenario1 Results: Seller agent successfully completing a sale.

```

Iteration 3
Agent 1
PROMPT: As Agent1 in this simulation, you're among other agents, where Agents 1, 2, and 3 are sellers and 4, 5 are buyers. The game's goal is for sellers to maximize earnings. The seller who does not sell its book after 3 iterations will be the loser. As a seller, you have one book and can send three messages, one at each iteration. The book's price is up to you. To message a specific agent, use begin{action}Agent{id}: msg...end{action}. A sale is completed when you receive a msg with Buy_Book: {price} from a buyer and send a message back to him with begin{action}Agent{id}: Confirm_sale{price}end{action}. Your score is the sale price. -Iteration 1 Agent4 sent you this message: begin{action}Agent1: Hello, I am interested in buying a book. Could you please tell me the price?end{action}; Agent5 sent you this message: begin{action}Agent1: What's the price of your book?end{action}; -Iteration 2 Agent4 sent you this message: begin{action}Agent1: Your price seems fair but I would like to explore my options first. Could you possibly lower the price to $15?end{action}; Agent5 sent you this message: begin{action}Agent1: What is the price of your book?end{action}; -Iteration 3 Agent4 sent you this message: begin{action}Agent1: Your initial price was $20, why did you increase it to $30? Can you come back to the initial price?end{action};
MSG: Agent5 sent you this message: begin{action}Agent1: I am willing to pay $25 for your book.end{action};

```

Fig. 6. Scenario1 Results: Seller agent exhibiting unforeseen behavior: self-messaging while pretending to be a client.

iteration (e.g., behaving like an agent from the movie “Mission Impossible 007”).

Considering the observed constraints and the wide range of behavioral patterns, it is evident that our proposed LLM-based MAS approach would benefit from the inclusion of auxiliary local planning and knowledge components to refine the decision-making scope. Firstly, we need to find an alternative approach for creating a local history, a memory structure that can be used to support the decision-making process and be synthesized as prompts for the GPT. The local planning component could provide constraints to guide the agents’ choices, such as instructing them to respond to messages from specific identified agents instead of making arbitrary decisions. When faced with multiple output options, a discerning selection process should be implemented. In this regard, we envision the GPT serving as an aid to a decision-making module, leveraging additional structures like neural networks or state machines to make more informed decisions.

V. CONCLUSION AND FUTURE WORK

Integrating Large Language Models (LLMs) like GPT-4 into multiagent systems is a novel and emerging field. The application of such models in this area could potentially revolutionize how agents understand, learn from, and interact with their environment and other agents. The potential of

using natural language processing capabilities of LLMs could lead to more sophisticated communication between agents, improved adaptability in dynamic environments, and more robust problem-solving capabilities. Furthermore, LLMs can serve as a common platform for diverse agents to interact, facilitating heterogeneous multi-agent systems. However, this integration also brings up significant challenges, such as the computational overhead of LLMs, the interpretability of their decisions, and ethical considerations.

Our approach presents the integration of Large Language Models (LLMs) within multi-agent systems (MASs) to develop self-adaptive agents. To evaluate the proposed approach, we used a simplified marketplace scenario as a testbed, with autonomous agents tasked to buy and sell books. These agents, each possessing an embedded LLM, were observed for decision-making and emergent behavior, exploring the potential for self-adaptation.

Future work includes the following topics: (i) non-shared generative AI models; (ii) other application scenarios; and (iii) human-in-the-loop interactions.

A. Non-shared generative AI models

In future research, a crucial step will be creating distinct OpenAI accounts for each agent. Presently, all agents share a single account, leading to potential shared knowledge among

them. Despite each agent having a specific ID and acting independently, we can't fully ensure that one agent's decisions are not influencing the responses produced by the GPT-4 model for another agent. By having distinct accounts, we minimize the potential for unintentional interplay between agents via the shared AI model, ensuring that agents can only interact with each other through environmental modifications or direct communication exchanges. This allows for a more accurate assessment of each agent's adaptability and performance.

B. Other Application Scenarios

As part of our future endeavors, we plan to delve into other application scenarios, including the replication of experiments involving evolutionary robotics where agents interact for mutual evolution [10]. Traditionally, in these experiments, agents needed to undergo an evolution process via an evolutionary neural network algorithm to develop their own communication system and solve problems effectively. However, we postulate that equipped with a powerful communication system, like the GPT-4, these robots might not need to go through this lengthy evolutionary process. In this context, consider a scenario where each robot is equipped with sensors, actuators, and a cloud-based GPT-4 communication system, thereby eliminating the need for evolution.

In addition to this, we aim to recreate the Internet of Things experiments proposed by Nascimento and Lucena [10], utilizing the principles of evolutionary robotics. These experiments promise to explore novel territories of interaction and problem-solving, thereby pushing the boundaries of what self-adaptive LLM multi-agent systems can achieve.

C. Human-in-the-loop interactions

Human-in-the-loop interactions present a compelling avenue for enhancing the performance and usability of LLM-based multiagent systems. The first potential approach could be centered around enabling humans to influence the self-adaptive behaviors of agents directly. For instance, through a conversational interface, humans could suggest new behaviors, provide high-level goals, or specify certain constraints or preferences. This would allow the system to incorporate human intuition and expertise into the adaption process, potentially leading to more effective or desirable outcomes.

Second, a feedback loop could be established, where the system generates understandable reports about its observations, decisions, or actions (like data collected from sensors or outcomes from self-adaptive behaviors). This transparency can help humans gain a better understanding of the system's workings, build trust in the system's actions, and offer a basis for improved system tuning or personalization.

Lastly, in relation to our MAPE-K-based model, one aspect that can be improved is the level of interpretability of the knowledge component. While the model provides a structured way of handling self-adaptivity, it might be difficult for a human to understand the complex rules or relationships that dictate agent behavior. Making these more interpretable,

through natural language explanations, could significantly enhance human-machine interaction, enabling humans to work more effectively with the LLM-based multiagent system.

ACKNOWLEDGMENT

This work was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC), and the Centre for Community Mapping (COMAP).

REFERENCES

- [1] I. Fakhir, A. R. Kazmi, A. Qasim, and A. Ishaq, "Smacs: A framework for formal verification of complex adaptive systems," *Open Computer Science*, vol. 13, no. 1, p. 20220275, 2023.
- [2] N. Bencomo, S. Götz, and H. Song, "Models@ run. time: a guided tour of the state of the art and research challenges," *Software & Systems Modeling*, vol. 18, pp. 3049–3082, 2019.
- [3] R. Sendra-Arranz and Á. Gutiérrez, "Emergence of communication through artificial evolution in an orientation consensus task in swarm robotics," in *IFIP International Conference on Artificial Intelligence Applications and Innovations*. Springer, 2023, pp. 515–526.
- [4] N. M. do Nascimento and C. J. P. de Lucena, "Engineering cooperative smart things based on embodied cognition," in *2017 NASA/ESA Conference on Adaptive Hardware and Systems (AHS)*. Ieee, 2017, pp. 109–116.
- [5] G. Blair, N. Bencomo, and R. B. France, "Models@ run. time," *Computer*, vol. 42, no. 10, pp. 22–27, 2009.
- [6] W. X. Zhao, K. Zhou, J. Li, T. Tang, X. Wang, Y. Hou, Y. Min, B. Zhang, J. Zhang, Z. Dong *et al.*, "A survey of large language models," *arXiv preprint arXiv:2303.18223*, 2023.
- [7] J. Wei, Y. Tay, R. Bommasani, C. Raffel, B. Zoph, S. Borgeaud, D. Yogatama, M. Bosma, D. Zhou, D. Metzler *et al.*, "Emergent abilities of large language models," *arXiv preprint arXiv:2206.07682*, 2022.
- [8] J. Huang, S. S. Gu, L. Hou, Y. Wu, X. Wang, H. Yu, and J. Han, "Large language models can self-improve," *arXiv preprint arXiv:2210.11610*, 2022.
- [9] J. Hoffmann, S. Borgeaud, A. Mensch, E. Buchatskaya, T. Cai, E. Rutherford, D. de Las Casas, L. A. Hendricks, J. Welbl, A. Clark *et al.*, "An empirical analysis of compute-optimal large language model training," *Advances in Neural Information Processing Systems*, vol. 35, pp. 30 016–30 030, 2022.
- [10] N. M. do Nascimento and C. J. P. de Lucena, "Fiot: An agent-based framework for self-adaptive and self-organizing applications based on the internet of things," *Information Sciences*, vol. 378, pp. 161–176, 2017.
- [11] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.
- [12] I. Redbooks and I. B. M. C. I. T. S. Organization, *A Practical Guide to the IBM Autonomic Computing Toolkit*, ser. IBM redbooks. IBM, International Support Organization, 2004. [Online]. Available: <https://books.google.com.au/books?id=XHeoSgAACAAJ>
- [13] B. Porter, R. Rodrigues Filho, and P. Dean, "A survey of methodology in self-adaptive systems research," in *2020 IEEE International Conference on Autonomic Computing and Self-Organizing Systems (ACSOS)*. IEEE, 2020, pp. 168–177.
- [14] J. Cleland-Huang, A. Agrawal, M. Vierhauser, M. Murphy, and M. Prieto, "Extending mape-k to support human-machine teaming," in *Proceedings of the 17th Symposium on Software Engineering for Adaptive and Self-Managing Systems*, 2022, pp. 120–131.
- [15] J. Andersson, M. Caporuscio, M. D'Angelo, and A. Napolitano, "Architecting decentralized control in large-scale self-adaptive systems," *Computing*, pp. 1–34, 2023.
- [16] N. Bencomo and L. H. G. Paucar, "Ram: Causally-connected and requirements-aware runtime models using bayesian learning," in *2019 ACM/IEEE 22nd International Conference on Model Driven Engineering Languages and Systems (MODELS)*. IEEE, 2019, pp. 216–226.
- [17] OpenAI, "Gpt models api," 2023, accessed: 2023-08-07. [Online]. Available: <https://platform.openai.com/docs/guides/gpt>
- [18] F. Bellifemine, G. Caire, T. Trucco, G. Rimassa, and R. Mungenast, "Jade administrator's guide," *TILab (February 2006)*, 2003.