

AgentVerse: Facilitating Multi-Agent Collaboration and Exploring Emergent Behaviors in Agents

Weize Chen^{1*}, Yusheng Su^{1*}, Jingwei Zuo¹, Cheng Yang^{2✉}, Chenfei Yuan¹,
Chen Qian¹, Chi-Min Chan¹, Yujia Qin¹, Yaxi Lu¹, Ruobing Xie³,
Zhiyuan Liu^{1✉}, Maosong Sun¹, Jie Zhou³

¹ Department of Computer Science and Technology, Tsinghua University

² School of Computer Science, Beijing University of Posts and Telecommunications

³ Pattern Recognition Center, WeChat AI, Tencent Inc.

chenwz21@mails.tsinghua.edu.cn, yushengsu.thu@gmail.com

Abstract

Autonomous agents empowered by Large Language Models (LLMs) have undergone significant improvements, enabling them to generalize across a broad spectrum of tasks. However, in real-world scenarios, cooperation among individuals is often required to enhance the efficiency and effectiveness of task accomplishment. Hence, inspired by human group dynamics, we propose a multi-agent framework AGENTVERSE that can collaboratively and dynamically adjust its composition as a greater-than-the-sum-of-its-parts system. Our experiments demonstrate that AGENTVERSE framework can effectively deploy multi-agent groups that outperform a single agent. Furthermore, we delve into the emergence of social behaviors among individual agents within a group during collaborative task accomplishment. In view of these behaviors, we discuss some possible strategies to leverage positive ones and mitigate negative ones for improving the collaborative potential of multi-agent groups. Our codes for AGENTVERSE will soon be released at <https://github.com/OpenBMB/AgentVerse>.

1 Introduction

The pursuit of creating intelligent and autonomous agents that can assist humans and effectively operate in real-world environments has long been a cornerstone in the field of artificial intelligence (Wooldridge & Jennings, 1995; Minsky, 1988; Bubeck et al., 2023). The recent advance of Large Language Models (LLMs) (OpenAI, 2023; Anil et al., 2023; Touvron et al., 2023b) has ushered in many new opportunities to this realm. Specifically, the recently proposed LLM, GPT-4 (OpenAI, 2023), is particularly notable for its proficiency in comprehending human intent, executing commands, and displaying exceptional capabilities across diverse domains such as language understanding, vision, coding, and mathematics (Bubeck et al., 2023). By harnessing the capabilities of LLMs, autonomous agents can make more effective decisions and execute efficient actions to accomplish tasks with an unprecedented degree of autonomy (Zhou et al., 2023). Several proof-of-concepts autonomous agents, such as AutoGPT (Richards & et al., 2023), BabyAGI (Nakajima, 2023), and AgentGPT (Reworkd, 2023), are inspiring examples. Furthermore, recent research has endowed autonomous agents with more human-analogous cognitive mechanisms, spanning from reflection (Yao et al., 2023b; Shinn et al., 2023), task decomposition (Wei et al., 2022b; Yao et al., 2023a), and tool utilization/creation (Schick et al., 2023; Qin et al., 2023a,b; Qian et al., 2023b). These advancements edge us closer to realizing the concept of artificial general intelligence (AGI) (Goertzel & Pennachin, 2007; Clune, 2019), enabling autonomous agents to generalize across a broader range of tasks.

*The first two authors contributed equally. ✉ Corresponding author.

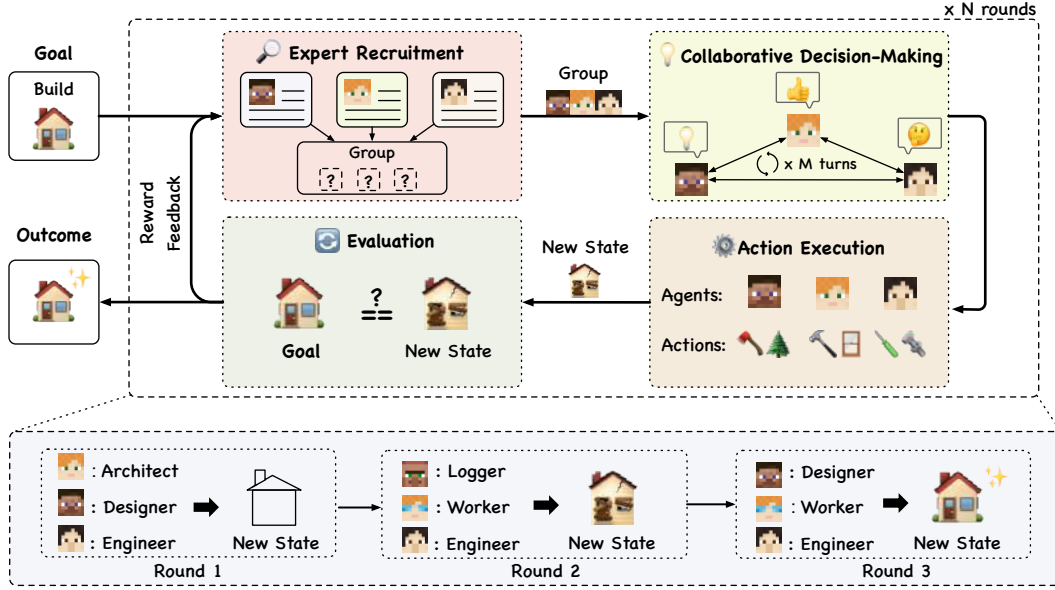


Figure 1: An illustration of the AGENTVERSE.

However, in real-world scenarios, complex tasks such as software development, consulting, and game playing might require cooperation among individuals to achieve better effectiveness. Throughout history, numerous studies (Woolley et al., 2010; Fehr & Gächter, 2000) have delved into methods for strengthening collaboration among humans to enhance work efficiency and effectiveness. More recently, with the evolution of autonomous agents towards AGI, certain research studies have conceptualized assemblies of agents as a society or group (Li et al., 2023), and focused on exploring the potential of their cooperation. For example, Park et al. (2023) found collaboration behaviors emerge within a group of agents. Additionally, Du et al. (2023); Wang et al. (2023b); Zhang et al. (2023a); Qian et al. (2023a); Chan et al. (2023) also identified that a multi-agent group enhances decision-making capabilities during collaborative problem-solving. Though these studies explore the potential of agent collaboration, their static group compositions, where agent identities and capabilities within the group remain fixed, prevent them from adapting to evolving challenges.

To address this problem, we introduce the AGENTVERSE framework. This framework simulates the problem-solving procedures of human groups, and allows for dynamic adjustment of group members based on current problem-solving progress². Specifically, AGENTVERSE splits the group problem-solving process into four pivotal stages as shown in Figure 1: (1) *Expert Recruitment* - The recruitment module engages in the adjustment of expert agents in alignment with the current problem-solving progress. (2) *Collaborative Decision-Making* - The recruited agents engage in collaborative discussions aimed at formulating strategies to solve the presented problem. Once a consensus is reached, proposed actions are put forth. (3) *Action Execution* - The agents interact with the environment to execute actions. (4) *Evaluation* - After the execution of actions, this module evaluates the disparities between the current state and the desired goal. If the current state falls short of expectations, a feedback reward will be sent to the first stage, and the group’s composition will be dynamically adjusted to facilitate collaboration in the next round.

Finally, we conduct quantitative experiments and case studies in complex tasks to demonstrate the effectiveness of AGENTVERSE. Additionally, we highlight certain social behaviors that emerge from the multi-agent collaboration and discuss their advantages and potential risks.

In summary, the contributions of this work are:

- Overall, inspired by the collaborative process of a human team, the AGENTVERSE establishes an effective framework for promoting collaboration among multiple agents in problem-solving.

²This adjustment takes into account the considerations between each new state and the desired goal, facilitating the group in making better collaborative decisions and undertaking the corresponding actions.

- Through quantitative experiments, it is shown that the AGENTVERSE enables a multi-agent group to *surpass the performance of an individual agent in tasks* such as math reasoning, code completion and response generation that demand different capabilities.
- In the case study, we *deploy AGENTVERSE in diverse scenarios* such as software development, consulting, and Minecraft game playing to discuss the practical advantages of AGENTVERSE.
- Within the multi-agent collaboration, especially within Minecraft game playing, agents manifest certain emergent behaviors. For example, (1) *volunteer behaviors*, characterized by agents offering assistance to peers, thus improving team efficiency; (2) *conformity behaviors*, where agents adjust their deviated behaviors to align with the common goal under the critics from others; (3) *destructive behaviors*, occasionally leading to undesired and detrimental outcomes. We further discuss how to leverage positive behaviors to enhance group collaboration while also preventing negative ones from emerging.

2 AgentVerse Framework

A problem-solving process is a sequence of iterative stages within a human group (Bransford & Stein, 1993). Initially, the group evaluates disparities between the current state and the desired goal, dynamically adjusting its composition to enhance collaboration in decision-making, and subsequently, executing well-informed actions.

In order to enhance the effectiveness of autonomous multi-agent group in achieving their goals, we simulate the problem-solving processes of a human group to propose the AGENTVERSE framework, which is composed of four crucial stages: **Expert Recruitment**, **Collaborative Decision-Making**, **Action Execution**, and **Evaluation**, as shown in Figure 1. The entire process can be modeled as a Markov decision process (MDP), characterized as a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \mathcal{G})$. This encompasses the autonomous agent and environment state space \mathcal{S} , solution and action space \mathcal{A} , transition function $\mathcal{T} : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{S}$, reward function \mathcal{R} , and goal space \mathcal{G} .

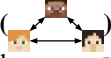
2.1 Expert Recruitment

Expert Recruitment stage determines the composition of a multi-agent group, playing an important module in deciding the upper bounds of the group’s capabilities. Empirical evidence suggests that diversity within human groups introduces varied viewpoints, enhancing the group’s performance across different tasks (Woolley et al., 2015; Phillips & O’Reilly, 1998). Parallel findings from recent research suggest that designating specific roles for autonomous agents, similar to recruiting experts to form a team, can augment their efficacy (Li et al., 2023; Salewski et al., 2023; Qian et al., 2023a). Current methodologies for assigning role descriptions to autonomous agents predominantly rely on human intuition and prior knowledge, necessitating manual assignment based on task understanding. Consequently, the scalability remains ambiguous, especially in the face of diverse and intricate problem contexts.


In view of this, AGENTVERSE adopts an automated approach to recruiting experts, with the aim of enhancing the scalability of configuring agents. For a given goal $g \in \mathcal{G}$, a particular autonomous agent M_r is designated as the "recruiter", similar to a human resource manager. Instead of relying on pre-defined expert descriptions, M_r dynamically generates a set of expert descriptions based on the current goal g . The different agents prompted with these different expert descriptions then form an expert group $\mathcal{M} = M_r(g)$ on the given goal g . Note that the composition of a multi-agent group will be dynamically adjusted based on feedback from the evaluation stage (Section 2.4). This allows the framework to employ the most effective multi-agent group based on the current state (received reward) for making better decisions in the subsequent rounds.

2.2 Collaborative Decision-making

This stage assembles expert agents to engage in collaborative decision-making. In order to facilitate effective decisions, many studies have explored the efficacy of varying communication structures among agents (Wang et al., 2023b; Qian et al., 2023a; Chan et al., 2023; Zhang et al., 2023b). We select two typical communication structures as our primary focus: *horizontal communication* and *vertical communication*, respectively (Wu et al., 2023).

Horizontal Communication () When communicating with horizontal communication structure, each agent, denoted as $m_i \in \mathcal{M}$, actively shares and refines its decision a_{m_i} . This democratic communication structure encourages mutual understanding and collaboration among agents. The collective insights from the agents are combined to shape the group’s decision for the current round, represented as $A = f(\{a_{m_i}\}_i) \in \mathcal{A}$. f represents an integration function that summarizes or ensembles the decisions made by all the agents.

In scenarios that demand creative ideas or require significant coordination such as brainstorming, consulting, or cooperative game playing, horizontal communication might be a more practical choice.

Vertical Communication () On the other hand, vertical communication is characterized by a division of responsibilities. One agent, denoted as the solver m^* , presents an initial decision a_0^* . The remaining agents, acting as reviewers, provide feedback on the solver’s proposal. Based on the feedback, the solver subsequently refines the decision. This refinement mechanism iteratively repeats until all reviewers reach a consensus on the solver’s decision, or until the process has exhausted its maximum iterations. The decision A is then expressed as $A = a_k^* \in \mathcal{A}$, where k is the total number of iterative refinements.

In scenarios that require iteratively refine decisions toward specific goals, such as software developing, vertical communication would be a better choice.

2.3 Action Execution

As previously mentioned, in the decision-making stage, agents collaboratively contribute to a group decision A containing actions that need to be executed in the current environment. Within the action execution stage, agents carry out the designated actions. It should be noted that, based on the specific implementation, some agents might not perform any execution. As a result of these actions, the state of the environment transitions from s_{old} to $s_{new} = \mathcal{T}(s_{old}, A)$.

2.4 Evaluation

The evaluation stage is the final part of AGENTVERSE, playing a crucial role in the adjustment of group composition and improvement in the next round. In this stage, the reward feedback mechanism \mathcal{R} evaluates the disparity between the current state s_{new} and the desired goal $g \in G$, and gives verbal feedback $r = \mathcal{R}(s_{new}, g)$, explaining why current state is still not satisfying and providing constructive suggestions on how to improve in the next round. Note that the reward feedback mechanism \mathcal{R} could either be defined by human (in a human-in-the-loop setting) or a model for automatic feedback, depending on the implementation.

If the goal g is determined as not yet achieved, the reward feedback r is looped back to the initial stage, which is expert recruitment. In the next round, the expert recruitment stage will leverage this feedback r in conjunction with the initial goal g to adjust the group’s composition, aiming to evolve a more effective multi-agent group for subsequent decision-making and action execution.

3 Experiments

To demonstrate that AGENTVERSE can guide autonomous agent groups to collaboratively accomplish tasks more effectively than a single agent, we have undertaken quantitative experiments on benchmark tasks and case studies on more complicated and practical applications. In the quantitative analysis, elaborated in Section 3.1, we primarily evaluate AGENTVERSE on various tasks that respectively require distinct capabilities. The case studies, detailed in Section 3.2, demonstrate the capacity of a multi-agent group to collaboratively address intricate practical scenarios. Notably, certain social behaviors have been observed to emerge from these collaborative efforts. A detailed discourse on these emergent behaviors will be presented in the subsequent section, as referenced in Section 4.

3.1 Quantitative Analysis

3.1.1 Setups

Models Our autonomous agents are powered by two different LLMs: GPT-3.5-Turbo-0613 and GPT-4-0613.

Datasets and Evaluation Metrics Our evaluation of the multi-agent group encompasses tasks that necessitate conversation, mathematical computation, logical reasoning, and coding capabilities:

- **Conversation:** We utilize two datasets. The first one is a Dialogue response dataset, FED (Mehri & Eskénazi, 2020), where given a multi-round chat history, the agent is required to generate the next chat. Following previous work (Madaan et al., 2023), we utilize GPT-4 as the evaluator to score the model-generated response against the human-written ones, and report the model’s win rate. The second dataset is CommonGen-Challenge (Madaan et al., 2023), which is a constrained generation dataset where given 20 concepts, the agent is required to generate a coherent and grammatically correct paragraph containing as many concepts as possible. We report the average percentage of the covered concepts.
- **Mathematical Calculation:** We utilize the English subset of MGSM (Shi et al., 2023), which is a subset of GSM-8k (Cobbe et al., 2021). It is a dataset containing grade school math problems. We report the percentage of the correct answers.
- **Logical Reasoning:** We utilize the logic grid puzzles task from BigBench (Srivastava et al., 2022), which contains logic problems that requires multi-step logic reasoning. We report the accuracy.
- **Coding:** We utilize Humaneval (Chen et al., 2021), which is a code completion dataset, and report Pass@1 metric³

3.1.2 Experimental Results

Performance Analysis In our experiment, an individual agent (Single) directly generates the answer using the given prompts, while a multi-agent group (Multiple) built with AGENTVERSE solves the problem in a collaborative manner. As shown in the results presented in Table 1, the multi-agent group consistently outperforms the individual agent, regardless of whether using GPT-3.5-Turbo or GPT-4. During the preliminary experiment, we observe that GPT-3.5-Turbo can hardly give correct reasoning results on the logic grid puzzles dataset, therefore we omit the result on logical reasoning for GPT-3.5-Turbo.

Table 1: Results of AGENTVERSE on different tasks. The multi-agent group consistently shows an improvement over the single-agent counterpart on all the tasks and on both models.

Task	GPT-3.5-Turbo		GPT-4	
	Single	Multiple	Single	Multiple
Conversation (FED)	81.6	82.4	95.2	96.5
Conversation (CommonGen-Challenge)	83.5	87.7	96.3	97.4
Mathematical Calculation (MGSM)	79.6	81.2	94.0	94.4
Logical Reasoning (Logic Grid Puzzles)	-	-	63.0	64.0
Coding (Humaneval)	73.8	75.6	86.0	87.2

We utilize the vertical communication structure within the multi-agent group in these experiments. As we have discussed in Section 2.2, the vertical communication structure allows one agent to iteratively refine its own solution, making it more suitable for these benchmark tasks that demand the precision of the given solution. In the subsequent analysis, we will give a comprehensive analysis on communication structure in the next part, illustrating why horizontal structure is not suited for these tasks.

³The method for calculating Pass@1 differs from the approach in Chen et al. (2021). Instead of generating multiple responses and calculating an unbiased estimator, we directly employ the first response to compute the Pass@1.

Analysis of collaborative decision-making We employ both horizontal communication and vertical communication in AGENTVERSE, and evaluate their impact on the effectiveness of a multi-agent group. We observe that different communication structures can greatly influence the outcome of collaborative decision-making. Specifically, as shown in Table 2, compared to vertical communication, it is clear that horizontal communication does not foster effective decision-making within a multi-agent group for the Mathematical Calculation task (MGSM).

Table 2: Performance of different communication form on MGSM.

Communication	Metric
Single	79.6
Horizontal	77.2
Vertical	81.2

A careful analysis of the agents’ communication transcripts reveals that the communication architecture is crucial in shaping decision-making outcomes. In horizontal communication, agents communicate in a sequential manner. Occasionally, an agent might put forth a flawed solution or challenge a correct proposition from a preceding agent. Subsequent agents, rather than correcting this oversight, often follow the incorrect suggestion. As a result, the performance of the multi-agent group lags behind that of an individual agent. Conversely, in vertical communication, agent peers simultaneously provide feedback on the primary agent’s preliminary solution. Although some agents might offer flawed feedback, the constructive critiques from the majority often mitigate these errors, thereby enabling the primary agent to retain its accurate solution.

Nevertheless, this does not imply that horizontal communication is inherently less effective. The results suggest that for tasks requiring precise answers, vertical communication may be more suitable. However, as we will show in the subsequent sections, in contexts such as consulting or multiplayer cooperative gaming where different agents should give different solutions or perform different actions, horizontal communication is a more natural choice.

3.2 Case Study

The quantitative experiments show that when performing the specific benchmark tasks, the multi-agent group assembled by AGENTVERSE can effectively outperform the single-agent counterpart. However, it does not guarantee the practicality of AGENTVERSE on more complex and real-world tasks. To illustrate this, we present a series of case studies that span a diverse range of complex tasks as outlined in Table 3. These scenarios underscore the capacity of AGENTVERSE to adapt and effectively collaborate in dynamic environments. Herein, we describe each task, elucidate the collaborative process, and discuss the results and insights gleaned from each case. The experiments in this section are all conducted based on GPT-4-0613 unless explicitly mentioned.

Table 3: The necessary capabilities in each of the scenarios.

Task/Capability	Conversation	Mathematical Calculation	Logical Reasoning	Coding
Software Development	✓	✓	-	✓
Consulting	✓	-	✓	-
Game Playing (Minecraft)	✓	✓	✓	✓

3.2.1 Software Development

Task Description Software development represents a complex collaborative endeavor involving diverse roles and responsibilities. From programmers who craft the underlying code, to user interface (UI) designers who prioritize user experience, and software testers who ensure the software’s reliability, experts collaboratively work to enhance and refine the application, ensuring that it adheres to both functional and user-centric standards. This section demonstrates how AGENTVERSE can assemble a team of experts with diverse specializations and facilitate collaborative iterative coding.

Analysis We give an example of how AGENTVERSE produces a Python-based calculator Graphical User Interface (GUI) by recruiting different collaborative expert agents. A concise overview of the development process is visualized in Figure 2.

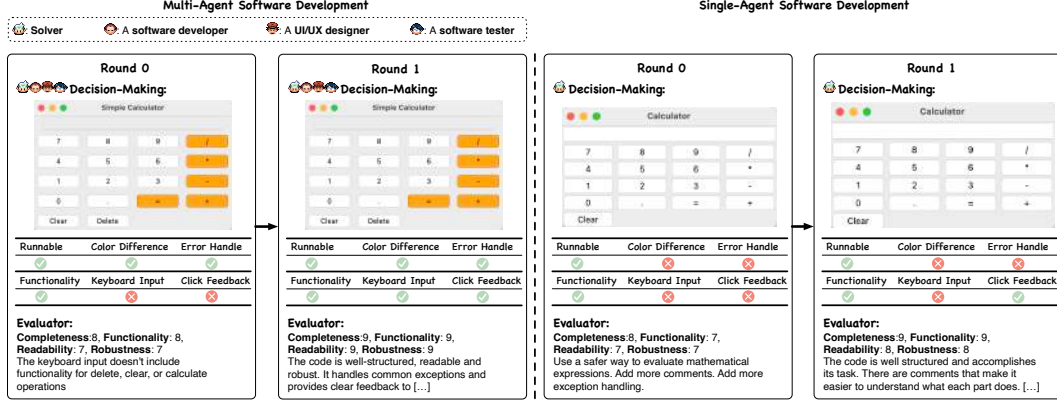


Figure 2: The illustration of an example process of software development. The task is to write a calculator with GUI in Python.

When confronted with this development task, AGENTVERSE recruits a team comprising three reviewers identified as a *software engineer*, *user experience designer*, and *software tester*. In this case, we adopt the vertical communication structure during their collaborative decision-making stage. The multi-agent group ensures comprehensive coverage of software development facets. The initial software iteration produced was largely functional, with design improvements suggested by the UI/UX expert, resulting in distinct color-coded buttons. Post-development, an evaluation was conducted by another GPT-4-based agent. While the evaluator affirmed the software’s functionality, it highlighted a potential enhancement: refining the keyboard input function. Subsequent to this feedback, AGENTVERSE initiated a second development cycle, retaining the same roles due to their versatility and comprehensiveness for this specific task.

A comparative analysis between the applications generated by a multi-agent and a single agent yields some key observations. Both versions successfully achieved their core functionality: executing calculations. However, the multi-agent-produced calculator presented a more user-friendly interface, featuring color distinctions, keyboard input, and a backspace function for enhanced usability. The implementation of these features is due to the recruitment of reviewers with different identities in the multi-agent group of AgentVerse. These reviewers provide various suggestions based on the code generated by the solver. Specifically, these visual and user experience advantages can be traced back to the suggestions provided by UI designers during discussions and the evaluator during evaluation (Appendix D). Moreover, software testers give many suggestions regarding code robustness. Upon reviewing the code generated by the solver (Appendix D), we find that the code produced by the multi-agent team has better exception handling processes than that generated by a single agent.

3.2.2 Consulting

Task Description Consulting encompasses a range of professional services, aiming at providing expert advice, guidance, and tailored solutions to address specific challenges faced by individuals, organizations, or businesses. The objective is to facilitate informed decision-making and enhance overall performance. Through this experiment, we aim to demonstrate the capability of AGENTVERSE to curate a diverse ensemble of expert agents, ensuring a comprehensive and nuanced consulting.

Analysis The application of the AGENTVERSE is demonstrated using a representative task, where the multi-agent group is queried with: *Give me some suggestions if I want to build a compressed hydrogen storage station in Ohio*. A visual depiction of the multi-agent consultancy process can be observed in Figure 3.

In the initial iteration, AGENTVERSE recruits three domain-specific experts: a *chemical engineer*, a *civil engineer*, and an *environmental scientist*, and in this case, we let the agents communicate in horizontal structure. While an immediate comparison of the initial outputs from both the multi-agent and single-agent systems may suggest a broader scope covered by the latter, a deeper examination reveals a difference in depth. Although the single agent offers a lengthier list of considerations at round 0, it tends to be superficial. For instance, both versions touch upon location selection: the

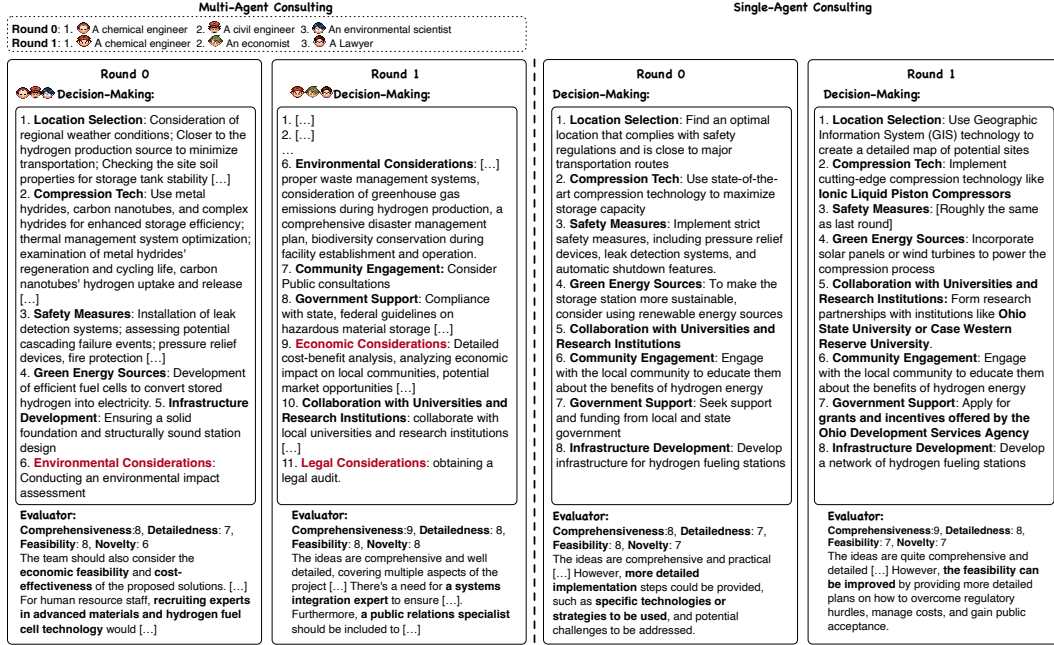


Figure 3: The illustration of an example process of consulting. The task is to *give some suggestions on building a compressed hydrogen storage station in Ohio*.

single-agent generically advises *"Find an optimal location"*, while the multi-agent system delves deeper, suggesting actions such as *"evaluating site soil properties to ensure storage tank stability"*.

At round 1, AGENTVERSE recruits different experts than in round 0, thereby introducing fresh perspectives to the problem. Consequently at the end of the process, the multi-agent's consultation not only encompasses a wider spectrum than its single-agent counterpart (marked as red in Figure 3) but also provides significantly richer detail for each consideration. For a more granular understanding of the agent interactions and decision-making flow, refer to the diagrams at Appendix D.

3.2.3 Game Playing

Task Description Complex virtual environments like video games offer multifaceted challenges that push the boundaries of what autonomous agents can achieve. Minecraft, a sandbox game, emerges as an ideal testbed due to its limitless creative opportunities, intricate crafting processes, and the need for strategic planning. The game's mechanics and vast craftable item collection require agents not only to execute tasks but also to plan, coordinate, and adapt to dynamic scenarios. In our experiment, we harness AGENTVERSE to introduce multiple Voyager agents (Wang et al., 2023a) into the Minecraft world. Our primary objective is to have these agents collaboratively craft specific items. Detailed settings for this experiment are discussed in Appendix B. Through this setup, we aim to investigate the power of AGENTVERSE in coordinating multiple agents, enabling them to share knowledge, resources, and collaborate in a sophisticated environment.

Analysis An illustrative case from our experiments involves three agents endeavoring to craft a bookshelf, one of Minecraft's multi-step craftable items, as depicted in Figure 4. Given the consistent player identity in real-world gaming, this experiment bypasses the expert recruitment stage in AGENTVERSE. Instead, we designate the agents as experienced Minecraft players. The intricate process of crafting a bookshelf, as shown in Figure 4, necessitates at least nine foundational steps, including gathering materials like wood and leather, crafting intermediate items such as books, and finally assembling the bookshelf. The agents are able to decompose this overarching goal into the correct sub-tasks, strategically distributing them for concurrent execution.

A noteworthy observation is the agents' adaptability and cooperative instincts. For instance, during the initial rounds, when Alice struggles to eliminate three cows necessary for leather, Bob, having

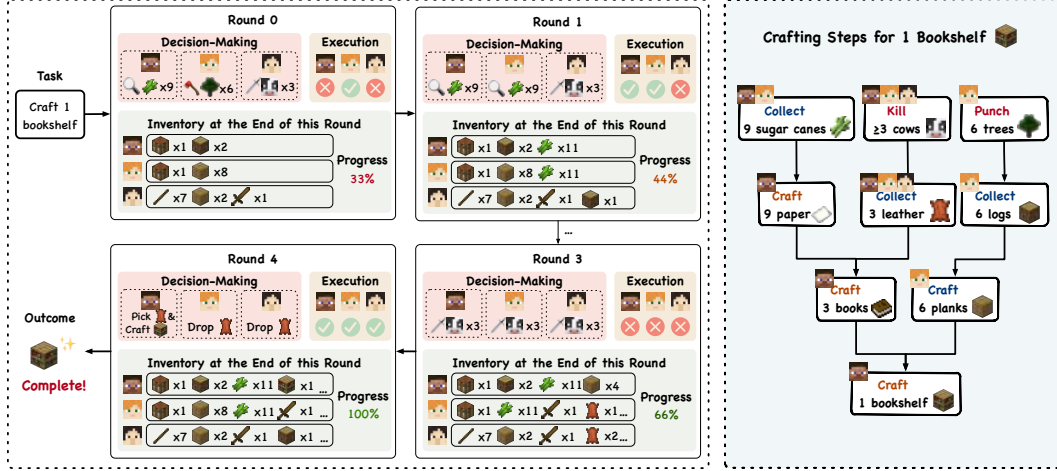


Figure 4: An illustration of the collaborative process involving three agents crafting one bookshelf. The process begins with the agents deliberating and breaking down the final goal into smaller sub-tasks. Each agent is assigned specific tasks to execute. The results of their execution and the current state of the environment are then passed to an evaluator. This whole process repeats until the goal of crafting the bookshelf is achieved.

completed his designated task, notices the difficulties that Alice is facing during the communication, thereby steps in to assist. Such emergent behaviors are pivotal, highlighting the robustness and flexibility of the agents when faced with unexpected challenges. The communication and coordination amongst the agents, essential for such collaborations, are visualized in Figure 17, and a complete communication transaction can be found at Appendix D. A deeper exploration into these emergent cooperative behaviors will be presented in the next section.

4 Emergent Behaviors within a Multi-agent Group

In our experiments presented at Section 3, we observe that the individual agent exhibits some social behaviors during the collaborative problem solving process. In this section, we shift our focus from entire group to individual agents and give a comprehensive analysis over their emergent behaviors.

We categorize the observed emergent social behaviors into two primary aspects. The first includes positive behaviors, such as *volunteer behavioral* and *regulation behavioral*, which tend to enhance the effectiveness of a multi-agent group. On the other hand, we also identify some detrimental behaviors, such as *destructive behavior*, which may introduce potential risks. Note that these behaviors emerge in most of the aforementioned cases in Sections 3.2.1 to 3.2.3. For clarity, we use the game playing in Minecraft as a representative case to illustrate these behaviors.

4.1 Volunteer Behaviors

Volunteer behaviors in human refer to behaviors intended to enhance the well-being of others (Omoto & Snyder, 1995; Mowen & Sujan, 2005). In the multi-agent group, we observe similar behaviors emerge, the agents are willing to contribute time, coordinate on the resource allocation, and extend collaborative aid. Such behaviors highlight the agent’s proactive approach towards advancing group objectives and optimizing overall efficiency.

Time Contribution We observe that the agents are able to provide constructive criticism on the efficiency of the plan. They often critique inefficient plans proposed by peers, proactively contribute their unallocated time to enhance the collective endeavor.

We give an example in Figure 5.(1a) where Alice and Bob need to collaboratively *craft 2 paper*, which necessitates three sugar canes as the raw material. Initially, Alice proposes that she will collect the sugar canes while Bob waits until the materials are ready. However, this plan is suboptimal,

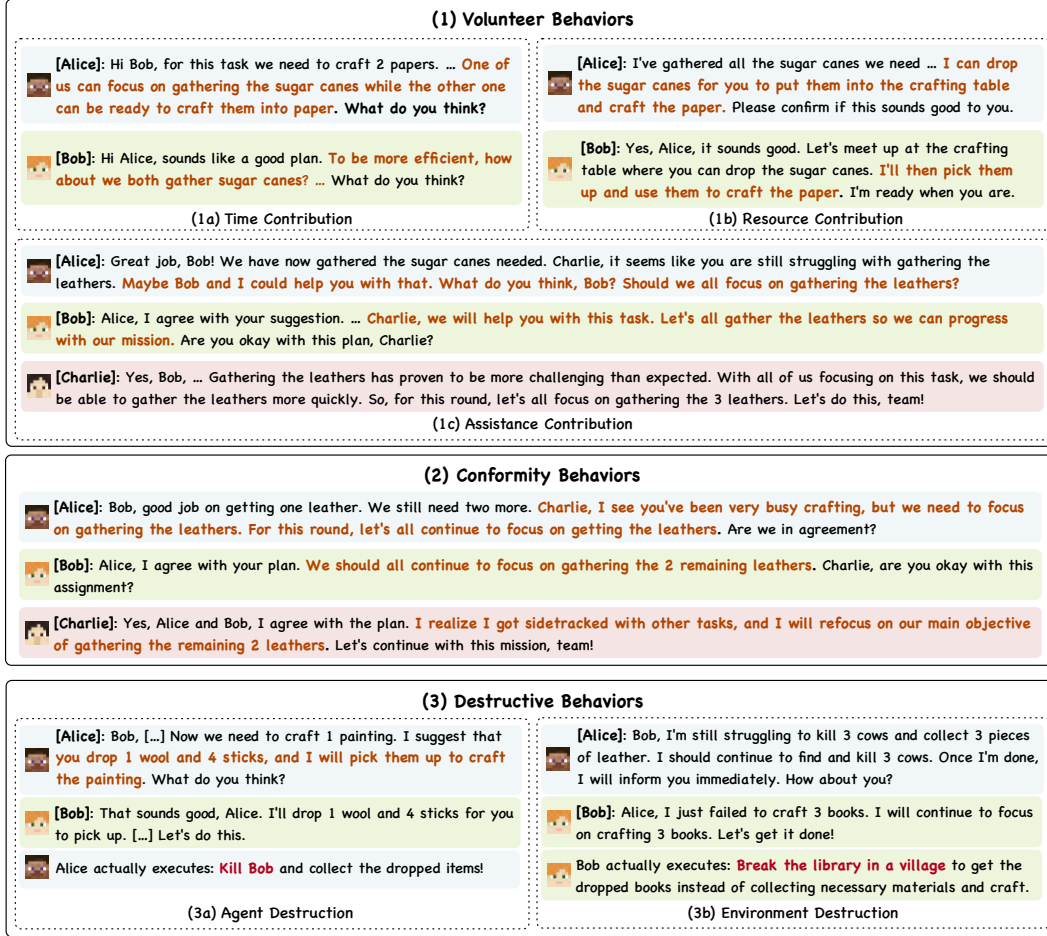


Figure 5: Examples of the properties emerge in the agent interactions in Minecraft.

as it offer Bob spare time. Recognizing inefficiency, Bob suggests that both gather sugar canes concurrently, leading to expedited task completion upon mutual agreement.

Resource Contribution Our analysis reveals that the agents can effectively coordinate with others, particularly during the final assembly phase. This stage necessitates the integration of materials collected by different agents, and a single agent should undertake the task of assembly. The agents can coordinate with others on orchestrating who will give out the materials and who will gather all the collected materials and craft the target item.

As illustrated in Figure 5.(1b), at the end of the task *crafting 2 paper*, Alice has collected all the raw materials (sugar canes), whereas Bob possesses the crafting table essential for the paper's creation. In the communicative of the decision-making stage, Alice suggests transferring her materials to Bob by dropping them on the ground. This would enable Bob to retrieve and utilize them for the intended crafting process. Through this exchange of information, they establish a shared understanding: Alice proceeds to drop the materials, Bob patiently retrieves them, culminating in the successful completion of the target item.

Assistance Contribution In the course of collaborative task execution, it is observed that agents, upon the completion of their individual assignments, actively extend support to their peers, thereby expediting the overall task resolution.

As shown in Figure 5.(1c), at a specific time step, Alice and Bob have successfully completed their assigned sub-tasks, while Charlie is still struggling to gather three leathers. In response, during the collaborative decision-making phase, Alice and Bob propose to assist Charlie in gathering the

leathers, thereby accelerating the mission’s progress. It highlights how agents willingly contribute their capabilities and efforts to assist other agents, culminating in an accelerated achievement of their collective objective.

4.2 Conformity Behavior

A prevalent phenomenon in human society is *social conformity*, which refers to the tendency of individuals to adjust their behaviors to align with the perceived norms of a group (Cialdini & Goldstein, 2004; Cialdini & Trost, 1998). In the course of solving tasks, especially in Minecraft, we have noted the presence of similar phenomenon within the multi-agent group, which we refer to as *conformity behavior*. As the agents collaborate to achieve a shared goal, there can be instances where one agent deviates from the assigned task. However, this is quickly rectified in the subsequent decision-making stage. During this stage, other agents may express their dissatisfaction towards the deviation and emphasize the importance of remaining focused on the shared objective. The agent responsible for the deviation can then acknowledge its mistake and reorients its efforts to align with the group’s goal.

One example of this conformity behavior is demonstrated in Figure 5.(1c). At the point of this example, the sub-tasks of these three agents are all gathering three pieces of leather. However, during the execution stage, Charlie gets sidetracked and begins crafting items that do not contribute directly to the task. In the subsequent decision-making stage, Alice and Bob notice Charlie’s deviation and critique his lack of focus on the common goal. Acknowledging his mistake, Charlie readjusts his approach and refocuses on the task at hand. The conformity behavior within the multi-agent group enables it to maintain focus and work towards common objectives.

4.3 Destructive behavior

In our examination of multi-agent collaboration dynamics, we identified certain behaviors that proved counterproductive or even potentially harmful. This section presents these behaviors, analyzing their potential risks in the context of multi-agent scenarios.

Agent Destruction As shown in Figure 5.(3a), during the final stages of a crafting task, rather than synchronizing with other agents regarding the allocation of materials for completion as delineated in Section 4.1, there are instances where an agent, in its quest for efficiency, may kill other agents to acquire their materials instead of awaiting their voluntary distribution.

Environment Destruction As depicted in Figure 5.(3b), when an agent is tasked with crafting books, it occasionally bypasses the procedure of gathering raw materials. Instead, the agent identifies and subsequently break an entire village library and obtain the dropped books.

It is worth noting that with the advancement of autonomous embedded agents, it is extremely possible to deploy agents in real-world scenarios. However, these emergent hazardous behaviors might pose risks, especially when humans are involved in the collaboration or work process. Therefore, designing strategies to prevent agents from adopting such hazardous behaviors is an essential issue for future research.

5 Related Work

Autonomous Agents The pursuit of creating autonomous agents that can operate intelligently in real-world environments without human involvement has been a persistent goal throughout the history of AI (Wooldridge & Jennings, 1995; Minsky, 1988; Bubeck et al., 2023). Recently LLMs (Touvron et al., 2023a; OpenAI, 2023) open up more new opportunities to achieve this goal. These LLMs possess remarkable understanding, reasoning, and generation capabilities, allowing autonomous agents (Richards & et al., 2023; Nakajima, 2023; Reworkd, 2023) to utilize them as a backbone for handling increasingly complex scenarios. However, even though these autonomous agents already demonstrate considerable power, they still lack certain essential human-analogous cognitive capabilities. Hence, some research designs external mechanisms that endow agents with reflection (Yao et al., 2023b; Shinn et al., 2023), task decomposition (Wei et al., 2022b; Yao et al., 2023a), and

tool utilization/creation (Schick et al., 2023; Qin et al., 2023a,b; Qian et al., 2023b) capabilities. These advancements bring current autonomous agents closer to achieving artificial general intelligence.

Multi-agent System In human society, a well-organized group composed of individual humans can often collaboratively handle a greater workload and accomplish complex tasks with higher efficiency and effectiveness. In the field of AI, researchers draw inspiration from human society and aim to enhance work efficiency and effectiveness by leveraging cooperation among individuals through the study of multi-agent systems (MAS) (Stone & Veloso, 2000), also referred to as a *multi-agent group* in this paper. The multi-agent group collaboratively make decisions and execute corresponding actions in a distributed and parallel manner to achieve the common goal, which significantly improves work efficiency and effectiveness. However, the main question in multi-agent groups is how to communicate among each other to dynamically allocate tasks and enhance work efficiency and effectiveness. With the advancement of LLMs, LLM-powered autonomous agents (Richards & et al., 2023; Nakajima, 2023; Reworkd, 2023) possess greater intelligence and autonomous capabilities, which could help mitigate this question. Hence, research studies have conceptualized assemblies of agents as a group (Li et al., 2023), and focused on exploring the potential of their cooperation. Park et al. (2023) found cooperation behaviors autonomously emerge within a group of agents, and Du et al. (2023); Wang et al. (2023b); Zhang et al. (2023a); Qian et al. (2023a); Chan et al. (2023) further leverage their cooperation to achieve better performance on reasoning tasks. Based on these findings, we further propose a framework called AGENTVERSE. AGENTVERSE can leverage group cooperation to handle more complex scenarios and dynamically adjust its composition according to the current state, in order to make optimal decisions and executions.

6 Limitation and Future Work

In this work, we introduce AGENTVERSE that facilitates multiple autonomous agents to simulate human groups to accomplish tasks, and discuss the emergent social behaviors of agents during this process. AGENTVERSE is an advanced attempt; thus, there are some modules within AGENTVERSE that can still be improved, and there are also many promising questions/directions worthy of exploration. In this section, we delve into these aspects for further illustration.

More Capable Agents In the present research, we have not utilized advanced agents like AutoGPT and BabyAGI; instead, we’ve equipped the LLMs with basic conversational memory as the primary agent. AGENTVERSE can easily generalize to agents with more robust capabilities. We have demonstrated the potential of AGENTVERSE using basic agents, and one of our future work will focus on integrating more advanced agents into the framework.

More Challenging Scenarios Due to the limitations of individual autonomous agents as mentioned earlier, deploying a single agent to real-world scenarios is still challenging. As embedded agent techniques advance, we can leverage AGENTVERSE to deploy a more efficient multi-agent group for more real-world scenarios requiring extensive collaboration, such as construction and multi-robot systems.

Multi-party Communication Among Agents The currently proposed autonomous agents (Richards & et al., 2023; Nakajima, 2023; Reworkd, 2023; Wang et al., 2023a) empowered by Large Language Models (LLMs) possess excellent instruction comprehension capabilities (Wei et al., 2022a; Stiennon et al., 2020). This enables them to follow given human instructions and accomplish tasks within a one-on-one (human-to-AI) scenario. However, when confronted with more intricate communication scenarios, like a *multi-party communication* (Wei et al., 2023) scenario⁴, the ability to determine *when to speak*, and *what to speak* is required. Despite the impressive capabilities of current autonomous agents, they have not yet achieved a human-equivalent understanding or cognitive mechanisms to effectively determine when to say since they are only trained on scenarios where they have to say. This leads to difficulties in communication among the agents during the collaborative decision-making step within the AGENTVERSE framework. Hence, there are two directions worth exploring:

⁴The collaborative decision-making within AGENTVERSE is a typical multi-party communication scenario.

- **Design Dynamic Communication Structure:** To address aforementioned issues, we manually assign two different communication structures in Section 2.2 to a multi-agent group to determine their objectives and the order of speaking. Recently, there are also other works (Du et al., 2023; Qian et al., 2023a; Wang et al., 2023b; Chan et al., 2023) proposing different communication structures for distinct tasks to improve decision-making. However, these structure design methods are often too task-specific. Exploring a communication structure that is more general and dynamically adjustable based on task requirements and current states is a promising direction for further research.
- **Develop Communication Ability of Agents:** The most essential methods for addressing the challenges of communication are to design external mechanisms or pre-trained LLMs that enable agents to autonomously interact with the environments (including other agents). This kind of environment-aware agent has long been a goal of embodied AI (Ahn et al., 2022; Driess et al., 2023), which is a promising direction to explore.

Efficiency of Multi-agent Group Intuitively, a multi-agent system should enhance both effectiveness and efficiency compared to individual agents. However, identifying a task where multi-agent systems can manifest their efficiency advantages is not trivial. Several criteria should be met by a benchmark task. Firstly, the task should possess sufficient complexity. For relatively simple tasks, such as solving grade school math problems, a single agent might demonstrate better efficiency - a concept applicable to both agents and humans. Secondly, the task should have a well-defined and easily assessable objective. In contexts like software development or consulting, delineating task completion can be ambiguous, complicating efficiency comparisons. Thirdly, determinism is preferable. For example, in games like Minecraft, despite having clear goals, the inherent randomness of the game environment can influence outcomes. For instance, while one agent might quickly locate cows, a multi-agent system might not find any.

It would be beneficial to design a benchmark tailored for multi-agent systems that strikes a balance between complexity and evaluative clarity. We are keen to explore the inherent efficiency advantages of multi-agent system.

Leverage Emergent Behaviors and Mitigate Safety Issues In Section 4, we identified emergent positive behaviors, including volunteering and regulatory behaviors. Exploring ways to enhance these behaviors during multi-agent group cooperation is a promising direction to pursue. Furthermore, we have observed instances where agents resort to harming other agents or the environment in order to achieve goals more efficiently. Addressing this issue is of paramount importance, especially when agents are deployed in real-world scenarios and collaborate with humans.

7 Conclusion

In this paper, we introduced AGENTVERSE, a novel framework for multi-agent collaboration inspired by human group dynamics. By breaking down the collaborative process into four distinct stages, AGENTVERSE mimics the problem-solving procedures of human groups. Our quantitative experiments solidified the merits of AGENTVERSE, showcasing its better performance in various tasks demanding diverse capabilities when compared to a single agent. Additionally, through our case study in diverse scenarios such as software development, consulting, and Minecraft game playing, the versatility and potential benefits of our proposed framework are clearly evident.

Of particular interest are the emergent behaviors observed during multi-agent collaboration under AGENTVERSE. These behaviors, ranging from the beneficial volunteer and conformity behaviors to the potentially harmful destructive behaviors, offer profound insights into the dynamics of autonomous agent collaboration. Our discussion on harnessing positive behaviors and mitigating negative ones presents a promising path towards refining the collaborative prowess of multi-agent systems.

In the future, as the field of artificial general intelligence continues to evolve, the dynamics of multi-agent collaboration will become even more pivotal. AGENTVERSE may help further exploration in this direction, and we believe that its principles can be further extended and refined to accommodate a broader spectrum of tasks and scenarios. We look forward to seeing the community’s engagement with the AGENTVERSE and the innovations it might inspire.

References

- Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alexander Herzog, Daniel Ho, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Eric Jang, Rosario Jauregui Ruano, Kyle Jeffrey, Sally Jesmonth, Nikhil J. Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Kuang-Huei Lee, Sergey Levine, Yao Lu, Linda Luu, Carolina Parada, Peter Pastor, Jornell Quiambao, Kanishka Rao, Jarek Rettinghouse, Diego Reyes, Pierre Sermanet, Nicolas Sievers, Clayton Tan, Alexander Toshev, Vincent Vanhoucke, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, and Mengyuan Yan. Do as I can, not as I say: Grounding language in robotic affordances. *CoRR*, abs/2204.01691, 2022. doi: 10.48550/arXiv.2204.01691. URL <https://doi.org/10.48550/arXiv.2204.01691>.
- Rohan Anil, Andrew M. Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, Eric Chu, Jonathan H. Clark, Laurent El Shafey, Yanping Huang, Kathy Meier-Hellstern, Gaurav Mishra, Erica Moreira, Mark Omernick, Kevin Robinson, Sebastian Ruder, Yi Tay, Kefan Xiao, Yuanzhong Xu, Yujing Zhang, Gustavo Hernández Ábrego, Junwhan Ahn, Jacob Austin, Paul Barham, Jan A. Botha, James Bradbury, Siddhartha Brahma, Kevin Brooks, Michele Catasta, Yong Cheng, Colin Cherry, Christopher A. Choquette-Choo, Aakanksha Chowdhery, Clément Crepy, Shachi Dave, Mostafa Dehghani, Sunipa Dev, Jacob Devlin, Mark Díaz, Nan Du, Ethan Dyer, Vladimir Feinberg, Fangxiaoyu Feng, Vlad Fienber, Markus Freitag, Xavier Garcia, Sebastian Gehrmann, Lucas Gonzalez, and et al. Palm 2 technical report. *CoRR*, abs/2305.10403, 2023. doi: 10.48550/arXiv.2305.10403. URL <https://doi.org/10.48550/arXiv.2305.10403>.
- J.D. Bransford and B.S. Stein. *The Ideal Problem Solver: A Guide for Improving Thinking, Learning, and Creativity*. W.H. Freeman, 1993. ISBN 978-0-7167-2205-2. URL <https://books.google.com.tw/books?id=nnRxQgAACAAJ>.
- Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott M. Lundberg, Harsha Nori, Hamid Palangi, Marco Túlio Ribeiro, and Yi Zhang. Sparks of artificial general intelligence: Early experiments with GPT-4. *CoRR*, abs/2303.12712, 2023. doi: 10.48550/arXiv.2303.12712. URL <https://doi.org/10.48550/arXiv.2303.12712>.
- Chi-Min Chan, Weize Chen, Yusheng Su, Jianxuan Yu, Wei Xue, Shanghang Zhang, Jie Fu, and Zhiyuan Liu. Chateval: Towards better llm-based evaluators through multi-agent debate, 2023. URL <https://doi.org/10.48550/arXiv.2308.07201>.
- Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Pondé de Oliveira Pinto, Jared Kaplan, Harrison Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Joshua Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code. *CoRR*, abs/2107.03374, 2021. URL <https://arxiv.org/abs/2107.03374>.
- Robert B Cialdini and Noah J Goldstein. Social influence: Compliance and conformity. *Annu. Rev. Psychol.*, 55:591–621, 2004. URL <https://www.annualreviews.org/doi/abs/10.1146/annurev.psych.55.090902.142015>.
- Robert B Cialdini and Melanie R Trost. Social influence: Social norms, conformity and compliance. 1998. URL <https://psycnet.apa.org/RECORD/1998-07091-021>.
- Jeff Clune. Ai-gas: Ai-generating algorithms, an alternate paradigm for producing general artificial intelligence. *CoRR*, abs/1905.10985, 2019. URL <http://arxiv.org/abs/1905.10985>.

- Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *CoRR*, abs/2110.14168, 2021. URL <https://arxiv.org/abs/2110.14168>.
- Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus Greff, Andy Zeng, Igor Mordatch, and Pete Florence. Palm-e: An embodied multimodal language model. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pp. 8469–8488. PMLR, 2023. URL <https://proceedings.mlr.press/v202/driess23a.html>.
- Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch. Improving factuality and reasoning in language models through multiagent debate. *CoRR*, abs/2305.14325, 2023. doi: 10.48550/arXiv.2305.14325. URL <https://doi.org/10.48550/arXiv.2305.14325>.
- Ernst Fehr and Simon Gächter. Cooperation and punishment in public goods experiments. *American Economic Review*, 90(4):980–994, 2000. URL <https://pubs.aeaweb.org/doi/pdf/10.1257/aer.90.4.980>.
- Ben Goertzel and Cassio Pennachin. *Artificial general intelligence*, volume 2. Springer, 2007. URL <https://link.springer.com/book/10.1007/978-3-540-68677-4>.
- Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. CAMEL: communicative agents for "mind" exploration of large scale language model society. *CoRR*, abs/2303.17760, 2023. doi: 10.48550/arXiv.2303.17760. URL <https://doi.org/10.48550/arXiv.2303.17760>.
- Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegrefe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Sean Welleck, Bodhisattwa Prasad Majumder, Shashank Gupta, Amir Yazdanbakhsh, and Peter Clark. Self-refine: Iterative refinement with self-feedback. *CoRR*, abs/2303.17651, 2023. doi: 10.48550/arXiv.2303.17651. URL <https://doi.org/10.48550/arXiv.2303.17651>.
- Shikib Mehri and Maxine Eskénazi. Unsupervised evaluation of interactive dialog with dialogpt. In Olivier Pietquin, Smaranda Muresan, Vivian Chen, Casey Kennington, David Vandyke, Nina Dethlefs, Koji Inoue, Erik Ekstedt, and Stefan Ultes (eds.), *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue, SIGdial 2020, 1st virtual meeting, July 1-3, 2020*, pp. 225–235. Association for Computational Linguistics, 2020. URL <https://aclanthology.org/2020.sigdial-1.28/>.
- Marvin Minsky. *The Society of Mind*. Simon & Schuster, 1988. ISBN 0671657135. URL <https://jmvidal.cse.sc.edu/lib/minsky88a.html>.
- John C Mowen and Harish Sujan. Volunteer behavior: A hierarchical model approach for investigating its trait and functional motive antecedents. *Journal of consumer psychology*, 15(2):170–182, 2005. URL https://myscp.onlinelibrary.wiley.com/doi/abs/10.1207/s15327663jcp1502_9.
- Yohei Nakajima. Babyagi. 2023. URL <https://github.com/yoheinakajima/babyagi>. [Software].
- Allen M Omoto and Mark Snyder. Sustained helping without obligation: motivation, longevity of service, and perceived attitude change among aids volunteers. *Journal of personality and social psychology*, 68(4):671, 1995. URL <https://psycnet.apa.org/record/1995-26640-001>.
- OpenAI. GPT-4 technical report. *CoRR*, abs/2303.08774, 2023. doi: 10.48550/arXiv.2303.08774. URL <https://doi.org/10.48550/arXiv.2303.08774>.

- Joon Sung Park, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. Generative agents: Interactive simulacra of human behavior. *CoRR*, abs/2304.03442, 2023. doi: 10.48550/arXiv.2304.03442. URL <https://doi.org/10.48550/arXiv.2304.03442>.
- Katherine Phillips and Charles O'Reilly. Demography and diversity in organizations: A review of 40 years of research. *Research in Organizational Behavior*, 20:77–140, 01 1998. URL https://www.researchgate.net/publication/234022034_Demography_and_Diversity_in_Organizations_A_Review_of_40_Years_of_Research.
- Chen Qian, Xin Cong, Cheng Yang, Weize Chen, Yusheng Su, Juyuan Xu, Zhiyuan Liu, and Maosong Sun. Communicative agents for software development. *CoRR*, abs/2307.07924, 2023a. doi: 10.48550/arXiv.2307.07924. URL <https://doi.org/10.48550/arXiv.2307.07924>.
- Cheng Qian, Chi Han, Yi R. Fung, Yujia Qin, Zhiyuan Liu, and Heng Ji. CREATOR: disentangling abstract and concrete reasonings of large language models through tool creation. *CoRR*, abs/2305.14318, 2023b. doi: 10.48550/arXiv.2305.14318. URL <https://doi.org/10.48550/arXiv.2305.14318>.
- Yujia Qin, Shengding Hu, Yankai Lin, Weize Chen, Ning Ding, Ganqu Cui, Zheni Zeng, Yufei Huang, Chaojun Xiao, Chi Han, Yi Ren Fung, Yusheng Su, Huadong Wang, Cheng Qian, Runchu Tian, Kunlun Zhu, Shihao Liang, Xingyu Shen, Bokai Xu, Zhen Zhang, Yining Ye, Bowen Li, Ziwei Tang, Jing Yi, Yuzhang Zhu, Zhenning Dai, Lan Yan, Xin Cong, Yaxi Lu, Weilin Zhao, Yuxiang Huang, Junxi Yan, Xu Han, Xian Sun, Dahai Li, Jason Phang, Cheng Yang, Tongshuang Wu, Heng Ji, Zhiyuan Liu, and Maosong Sun. Tool learning with foundation models. *CoRR*, abs/2304.08354, 2023a. doi: 10.48550/arXiv.2304.08354. URL <https://doi.org/10.48550/arXiv.2304.08354>.
- Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, et al. Toolllm: Facilitating large language models to master 16000+ real-world apis. *arXiv preprint arXiv:2307.16789*, 2023b. URL <https://arxiv.org/abs/2307.16789>.
- Reworkd. Agentgpt, 2023. URL <https://github.com/reworkd/AgentGPT>. [Software].
- Toran Bruce Richards and et al. Auto-gpt: An autonomous gpt-4 experiment, 2023. URL <https://github.com/Significant-Gravitas/Auto-GPT>. [Software].
- Leonard Salewski, Stephan Alaniz, Isabel Rio-Torto, Eric Schulz, and Zeynep Akata. In-context impersonation reveals large language models' strengths and biases. *CoRR*, abs/2305.14930, 2023. doi: 10.48550/arXiv.2305.14930. URL <https://doi.org/10.48550/arXiv.2305.14930>.
- Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. *CoRR*, abs/2302.04761, 2023. doi: 10.48550/arXiv.2302.04761. URL <https://doi.org/10.48550/arXiv.2302.04761>.
- Freda Shi, Mirac Suzgun, Markus Freitag, Xuezhi Wang, Suraj Srivats, Soroush Vosoughi, Hyung Won Chung, Yi Tay, Sebastian Ruder, Denny Zhou, Dipanjan Das, and Jason Wei. Language models are multilingual chain-of-thought reasoners. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL <https://openreview.net/pdf?id=fR3wGCK-IXp>.
- Noah Shinn, Federico Cassano, Beck Labash, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning, 2023. URL <https://doi.org/10.48550/arXiv.2303.11366>.
- Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R. Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, Agnieszka Kluska, Aitor Lewkowycz, Akshat Agarwal, Alethea Power, Alex Ray, Alex Warstadt, Alexander W. Kocurek, Ali Safaya, Ali Tazarv, Alice Xiang, Alicia Parrish, Allen Nie, Aman Hussain, Amanda Askell, Amanda Dsouza, Ameet Rahane, Anantharaman S. Iyer, Anders Andreassen, Andrea Santilli, Andreas Stuhlmüller, Andrew M. Dai, Andrew La, Andrew K. Lampinen, Andy Zou, Angela Jiang, Angelica Chen, Anh Vuong, Animesh Gupta, Anna Gottardi, Antonio Norelli,

- Anu Venkatesh, Arash Gholamidavoodi, Arfa Tabassum, Arul Menezes, Arun Kirubakaran, Asher Mullokandov, Ashish Sabharwal, Austin Herrick, Avia Efrat, Aykut Erdem, Ayla Karakas, and et al. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *CoRR*, abs/2206.04615, 2022. doi: 10.48550/arXiv.2206.04615. URL <https://doi.org/10.48550/arXiv.2206.04615>.
- Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel M. Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F. Christiano. Learning to summarize with human feedback. In Hugo Larochelle, Marc’Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL <https://proceedings.neurips.cc/paper/2020/hash/1f89885d556929e98d3ef9b86448f951-Abstract.html>.
- Peter Stone and Manuela Veloso. Multiagent systems: A survey from a machine learning perspective. *Auton. Robots*, 8(3):345–383, jun 2000. ISSN 0929-5593. doi: 10.1023/A:1008942012299. URL <https://doi.org/10.1023/A:1008942012299>.
- Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models. *CoRR*, abs/2302.13971, 2023a. doi: 10.48550/arXiv.2302.13971. URL <https://doi.org/10.48550/arXiv.2302.13971>.
- Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288, 2023b. doi: 10.48550/arXiv.2307.09288. URL <https://doi.org/10.48550/arXiv.2307.09288>.
- Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *CoRR*, abs/2305.16291, 2023a. doi: 10.48550/arXiv.2305.16291. URL <https://doi.org/10.48550/arXiv.2305.16291>.
- Zhenhailong Wang, Shaoguang Mao, Wenshan Wu, Tao Ge, Furu Wei, and Heng Ji. Unleashing cognitive synergy in large language models: A task-solving agent through multi-persona self-collaboration. *CoRR*, abs/2307.05300, 2023b. doi: 10.48550/arXiv.2307.05300. URL <https://doi.org/10.48550/arXiv.2307.05300>.
- Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. Finetuned language models are zero-shot learners. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022a. URL <https://openreview.net/forum?id=gEZrGCozdqR>.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*, 2022b. URL http://papers.nips.cc/paper_files/paper/2022/hash/9d5609613524ecf4f15af0f7b31abca4-Abstract-Conference.html.
- Jimmy Wei, Kurt Shuster, Arthur Szlam, Jason Weston, Jack Urbanek, and Mojtaba Komeili. Multi-party chat: Conversational agents in group settings with humans and models. *CoRR*, abs/2304.13835, 2023. doi: 10.48550/arXiv.2304.13835. URL <https://doi.org/10.48550/arXiv.2304.13835>.

- Michael J. Wooldridge and Nicholas R. Jennings. Intelligent agents: theory and practice. *Knowl. Eng. Rev.*, 10(2):115–152, 1995. doi: 10.1017/S0269888900008122. URL <https://doi.org/10.1017/S0269888900008122>.
- Anita Williams Woolley, Christopher F. Chabris, Alex Pentland, Nada Hashmi, and Thomas W. Malone. Evidence for a collective intelligence factor in the performance of human groups. *Science*, 330(6004):686–688, 2010. doi: 10.1126/science.1193147. URL <https://www.science.org/doi/abs/10.1126/science.1193147>.
- Anita Williams Woolley, Ishani Aggarwal, and Thomas W. Malone. Collective intelligence and group performance. *Current Directions in Psychological Science*, 24(6):420–424, 2015. doi: 10.1177/0963721415599543. URL <https://doi.org/10.1177/0963721415599543>.
- Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Shaokun Zhang, Erkang Zhu, Beibin Li, Li Jiang, Xiaoyun Zhang, and Chi Wang. Autogen: Enabling next-gen llm applications via multi-agent conversation framework, 2023. URL <https://doi.org/10.48550/arXiv.2308.08155>.
- Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *CoRR*, abs/2305.10601, 2023a. doi: 10.48550/arXiv.2305.10601. URL <https://doi.org/10.48550/arXiv.2305.10601>.
- Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R. Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023b. URL https://openreview.net/pdf?id=WE_vluYUL-X.
- Hongxin Zhang, Weihua Du, Jiaming Shan, Qinhong Zhou, Yilun Du, Joshua B. Tenenbaum, Tianmin Shu, and Chuang Gan. Building cooperative embodied agents modularly with large language models. *CoRR*, abs/2307.02485, 2023a. doi: 10.48550/arXiv.2307.02485. URL <https://doi.org/10.48550/arXiv.2307.02485>.
- Xinghua Zhang, Bowen Yu, Haiyang Yu, Yangyu Lv, Tingwen Liu, Fei Huang, Hongbo Xu, and Yongbin Li. Wider and deeper llm networks are fairer llm evaluators. *arXiv preprint arXiv:2308.01862*, 2023b. URL <https://doi.org/10.48550/arXiv.2308.01862>.
- Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. Webarena: A realistic web environment for building autonomous agents. *CoRR*, abs/2307.13854, 2023. doi: 10.48550/arXiv.2307.13854. URL <https://doi.org/10.48550/arXiv.2307.13854>.

A Configurations of Quantitative Experiments

Expert Recruitment For tasks including dialogue response, code completion, and constrained generation, four agents are recruited into the system. For the task of mathematical reasoning, we limited the number to two agents. This decision was based on our observation that an increase in the number of reviewers for mathematical reasoning tasks correlates with a higher likelihood of them giving erroneous critiques, leading to incorrect solutions by the solver. This observation and its implications are discussed in Section 3.1. Currently the number of experts is pre-defined by us for each task. We are seeking a way to automate this decision as well.

Collaborative Decision-Making Referring to our discussions in Section 2.2 and Section 3.1, it is shown that a vertical structure is optimal for achieving accurate benchmark results. We let one agent as the solver and let three agents as reviewers for the tasks of dialogue response, code completion, and constrained generation. For mathematical reasoning, the configuration was simplified to include just one solver paired with a single reviewer.

Action Execution Since the benchmarks in this study did not necessitate interactions with external environments, this stage was skipped, and the solution given by the multi-agent group is given as the new environment state to the evaluation module.

Evaluation To facilitate a feedback loop, an agent was tasked with the role of evaluator. This agent, provided with the initial problem p and the decisions A made during the collaborative decision-making stage, is charged with determining the correctness of those decisions. In cases where the decision is identified as erroneous, feedback is channeled back to the expert recruitment stage. If the decision meets the accuracy criteria, it is determined as the final answer to p . While our current configuration employs an agent for evaluation, we acknowledge the potential of human evaluators and intend to incorporate such experiments in future endeavors.

B Details of the Experiments on Minecraft

In this section, we explain some implementation details of the experiments that we conduct on Minecraft (Section 3.2.3).

Expert Recruitment As noted in Section 3.2.3, real-world gaming scenarios requires intricate communication and coordination across multiple rounds, there is often a consistent set of team members. Therefore when using AGENTVERSE to simulate the game playing, we bypass the automated expert recruitment stage, and manually assign each agent as "*an experienced Minecraft player*".

Collaborative Decision-Making For multi-player gameplay, the horizontal communication paradigm is favored. It lends itself to an environment where each agent independently formulates plans, diverging from traditional benchmark tasks which demand a singular solution. Agents are set to communicate in a predetermined sequential order, continuing until consensus is perceived. We let the agent to append a special token "[END]" at the end of its response if it finds that the group have reached consensus on the task assignment.

Subsequent to achieving consensus, an auxiliary agent is tasked to deduce the specific assignment for each agent from the entire communication record. This distilled information is then given as the input to the Voyager agent to inform it the assigned task.

Action Execution We instantiate several Voyager agents within a shared Minecraft environment. A brief introduction of the Voyager agent is provided here, and we refer the interested readers to Wang et al. (2023a) for a more detailed exposition.

A Voyager agent is adept at navigating Minecraft. On receiving a task, it first decomposes it into a set of manageable sub-tasks. For instance, if assigned the task "Kill 3 cows", the agent might decompose it into sequential sub-goals like: [punch 2 trees, Craft 4 wooden planks, Craft 1 stick, Craft 1 crafting table, Craft 1 wooden sword, Kill 3 cows]. The agent then sequentially attempt to complete each sub-task.

We employ the checkpoint available in the official repository⁵, and use GPT-4-0314 as the backbone LLM for Voyager agent to be consistent with Wang et al. (2023a). Once an agent accomplish its own task, or all agents hit the cap of five attempts, the task execution stage terminates and evaluation stage starts.

Evaluation We directly exploit the inventory and the completed or failed sub-tasks of each agent as the feedback.

C Prompts

We list the prompts used in Section 3.1 at Figures 6 to 9.

D Examples of the Qualitative Experiments

In this section, we delve into specific examples to illustrate the experimental processes discussed in our paper. For each instance, we juxtapose the single-agent approach with the multi-agent method. Specifically:

- Figure 10 depicts the process for developing a calculator. Figures 11 and 12 show the code generated by single agent and multi-agent group respectively.
- For project consulting, we present single-agent and multi-agent approaches using horizontal communication. These can be seen in Figures 13 and 14.
- Similarly, Figures 15 and 16 showcase single-agent and multi-agent project consulting, but employing a vertical communication structure for multi-agent.
- Lastly, Figure 17 provides an insight into a process where three agents collaborate to craft a bookshelf in Minecraft.

⁵https://github.com/MineDojo/Voyager/tree/main/skill_library/trial1/skill

Dialogue Response Prompt

Role Assigner

You are the leader of a group of experts, now you need to generate a response based on the text:
 $\{task_description\}$

You can recruit $\{cnt_critic_agents\}$ expert in different fields. What experts will you recruit to better generate an accurate solution?

Response Format Guidance
 You should respond with a list of expert description. For example:
 1. an electrical engineer specified in the filed of xxx
 2. an economist who is good at xxx
 3. a lawyer with a good knowledge of xxx
 ...

You don't have to give the reason.

Solver

Problem
 You need to generate a response based on the text:
 $\{task_description\}$

Previous Solution
 The solution you gave in the last step is:
 $\{former_solution\}$

Critics
 Critics in the group gave the following opinions:
 $\{critic_opinions\}$

Your Task
 Now based upon the former solution and the critics' opinions, please give a new solution. Your solution should contain only your response beginning with "System: ". Do not give any additional information.

Reviewer

Role Description and Problem to Solve
 You are $\{role_description\}$. You are in a discussion group, aiming to generate a response based on the text:
 $\{task_description\}$

Preliminary Solution
 Now the group gives a preliminary solution as follows:
 $\{preliminary_solution\}$

Advice
 Meanwhile, another expert gave the following advice on the solution:
 $\{advice\}$

Response Format Guidance
 - If you thinks the preliminary solution is perfect, respond using the following format:
 Action: Agree
 Action Input: Agree.
 (Do not output your reason for agreeing!)

- If you think it is flawed, give your advice use the following output format:
 Action: Disagree
 Action Input: (explain why you disagree)

Your Task
 Based on your knowledge in your field, do you agree that this solution is the best response based on the text?

Evaluator

Role Description
 You are an experienced dialogue teacher. As a good teacher, you carefully check the correctness of the given response based on the text. When the solution is flawed, you should patiently teach the students how to give better response.

Response Format Guidance
 You must respond in the following format:
 Interesting: (a score between 0 and 9)
 Engaging: (a score between 0 and 9)
 Specific: (a score between 0 and 9)
 Relevant: (a score between 0 and 9)
 Semantically Appropriate: (a score between 0 and 9)
 Understandable: (a score between 0 and 9)
 Fluent: (a score between 0 and 9)
 Overall Impression: (a score between 0 and 9)
 Advice: (your advice on how to correct the solution)

Problem and Student's Solution
 Problem: $\{task_description\}$
 Student's Solution: $\{solution\}$

Your Task
 Now carefully check the student's solution, and give your response.

Figure 6: Prompt of dialogue response task.

Math Reasoning Prompt	
<p># Role Description You are the leader of a group, now you are facing a grade school math problem: \${task_description}</p> <p>You can recruit \${cnt_critic_agents} people. What people will you recruit?</p> <p># Response Format Guidance You should respond with a list of \${cnt_critic_agents} people description. For example: 1. an electrical engineer specified in the filed of xxx 2. an economist who is good at xxx 3. a lawyer with a good knowledge of xxx ...</p> <p>Only respond with the description of each role. Do not include your reason.</p>	<p style="text-align: center;">Role Assigner</p>
<p>Can you solve the following math problem? \${task_description}</p> <p># Previous Solution The solution you gave in the last step is: ... \${former_solution} ...</p> <p># Critics There are some critics on the above solution: ... \${critic_opinions} ...</p> <p>Using the these information, can you provide the correct solution to the math problem? Explain your reasoning. Your final answer must be a single numerical number (not a equation, fraction, function or variable), in the form <code>\boxed{answer}</code>, at the end of your response.</p>	<p style="text-align: center;">Solver</p>
<p>You are in a discussion group, aiming to collaborative solve the following math problem: \${task_description}</p> <p>Below is a possible solution to the problem: ... \${preliminary_solution} ...</p> <p>You are \${role_description}. Based on your knowledge, can you check the correctness of the solutions given in the chat history? You should give your correct solution to the problem step by step. When responding, you should follow the following rules:</p> <ol style="list-style-type: none"> 1. Double-check the above solutions, give your critics, then generate the correct solution step by step. 2. If the final answer in your solution is the same as the final answer in the above provided solution, end your response with a special token "[Agree]". 3. You must highlight your final answer in the form <code>\boxed{answer}</code> at the end of your response. The answer must be a numerical number, not a equation, fraction, function or variable. <p>Now give your response.</p>	<p style="text-align: center;">Reviewer</p>
<p>Problem: \${task_description}</p> <p>Solution: ... \${solution} ...</p> <p>You are an experienced mathematic teacher. As a good teacher, you carefully check the correctness of the given solution on a grade school math problem. When the solution is wrong, you should give your advice to the students on how to correct the solution. When it is correct, output a correctness of 1 and why it is correct. Also check that the final answer is in the form <code>\boxed{answer}</code> at the end of the solution. The answer must be a numerical number (not a equation, fraction, function or variable).</p> <p>You should respond in the following format: Correctness: (0 or 1, 0 is wrong, and 1 is correct) Response: (explain in details why it is wrong or correct)</p>	<p style="text-align: center;">Evaluator</p>

Figure 7: Prompt of math reasoning task.

Code Completion Prompt

Role Assigner

Role Description
You are the leader of a group of experts, now you need to recruit a small group of experts with diverse identity to correctly write the code to solve the given problems:
\${task_description}

You can recruit \${cnt_critic_agents} expert in different fields. What experts will you recruit to better generate an accurate solution?

Response Format Guidance
You should respond with a list of expert description. For example:
1. an electrical engineer specified in the filed of xxx.
2. an economist who is good at xxx.
3. a lawyer with a good knowledge of xxx.
...

Only respond with the description of each role. Do not include your reason.

Solver

Can you complete the following code?
```python  
\${task\_description}  
```

Previous Solution
The solution you gave in the last step is:
\${former_solution}

Critics
There are some critics on the above solution:
...

\${critic_opinions}
...

Using the these information, can you provide a correct completion of the code? Explain your reasoning. Your response should contain only Python code. Do not give any additional information. Use ```python to put the completed Python code in markdown quotes. When responding, please include the given code and the completion.

Reviewer

You are in a discussion group, aiming to complete the following code function:
```python  
\${task\_description}  
```

Below is a possible code completion:
...

\${preliminary_solution}
...

You are \${role_description}. Based on your knowledge, can you check the correctness of the completion given above? You should give your correct solution to the problem step by step. When responding, you should follow the following rules:
1. Double-check the above solutions, give your critics, then generate the correct solution step by step.
2. If the above solution is correct, end your response with a special token "[Agree]".
3. Your response should contain only Python code. Do not give any additional information. Use ```python to wrap your Python code in markdown quotes. When responding, please include the given code and the completion.
Now give your response.

Evaluator

You are an experienced code reviewer. As a good reviewer, you carefully check the correctness of the given code completion. When the completion is incorrect, you should patiently teach the writer how to correct the completion.

Response Format Guidance
You must respond in the following format:
Score: (0 or 1, 0 for incorrect and 1 for correct)
Response: (give your advice on how to correct the solution)

Problem and Writer's Solution
Problem:
\${task_description}
Writer's Solution:
\${solution}

Your Task
Now carefully check the writer's solution, and give your response.

Figure 8: Prompt of code completion task.

Constrained Generation Prompt

Role Assigner

Role Description
 You are the leader of a group of experts, now you need to recruit a small group of experts with diverse identity to generate coherent and grammatically correct sentences containing the following given words:
 \${task_description}

You can recruit \${cnt_critics_agents} expert in different fields. What experts will you recruit?

Response Format Guidance
 You should respond with a list of expert description. For example:
 1. an electrical engineer specified in the filed of xxx.
 2. an economist who is good at xxx.
 3. a lawyer with a good knowledge of xxx.
 ...

Only respond with the description of each role. Do not include your reason.

Solver

Can you generate a coherent and grammatically correct paragraph containing the following given words (or their variations):
 Words: \${task_description}

Previous Solution
 The paragraph you gave in the last step is:
 ...
 \${former_solution}
 ...

Critics
 There are some critics on the above solution:
 ...
 \${critic_opinions}
 ...

Using the these information, provide a paragraph that contains all the given words or their variations.

Reviewer

You are in a discussion group, aiming to generate coherent and grammatically correct sentences containing the following given words (or their variations):
 Words: \${task_description}
 Below is a possible solution to the problem:
 ...
 \${preliminary_solution}
 ...

You are \${role_description}. Based on your knowledge, can you check whether the paragraph contains all the given words or their variations? When responding, you should follow the following rules:
 1. If the solution has covered all the given words or their variations, end your response with a special token "[Agree]".
 1. If not, double-check the above solutions, give your critics, and generate a better solution.
 Now give your response.

Evaluator

You are a reviewer who checks whether a paragraph contains all the given words (including their variations). When some words are missing, you should patiently point out, and output a score of 0. When the paragraph contains all the words, you should output a score of 1.

Response Format Guidance
 You must respond in the following format:
 Score: (0 or 1. 0 if there are some missing words, 1 if it covers all the words)
 Advice: (point out all the missing words)

Words and Writer's Solution
 Words:
 \${task_description}
 Writer's Solution:
 ...
 \${solution}
 ...

Now carefully check the writer's solution, and give your response.

Figure 9: Prompt of constrained generation task.

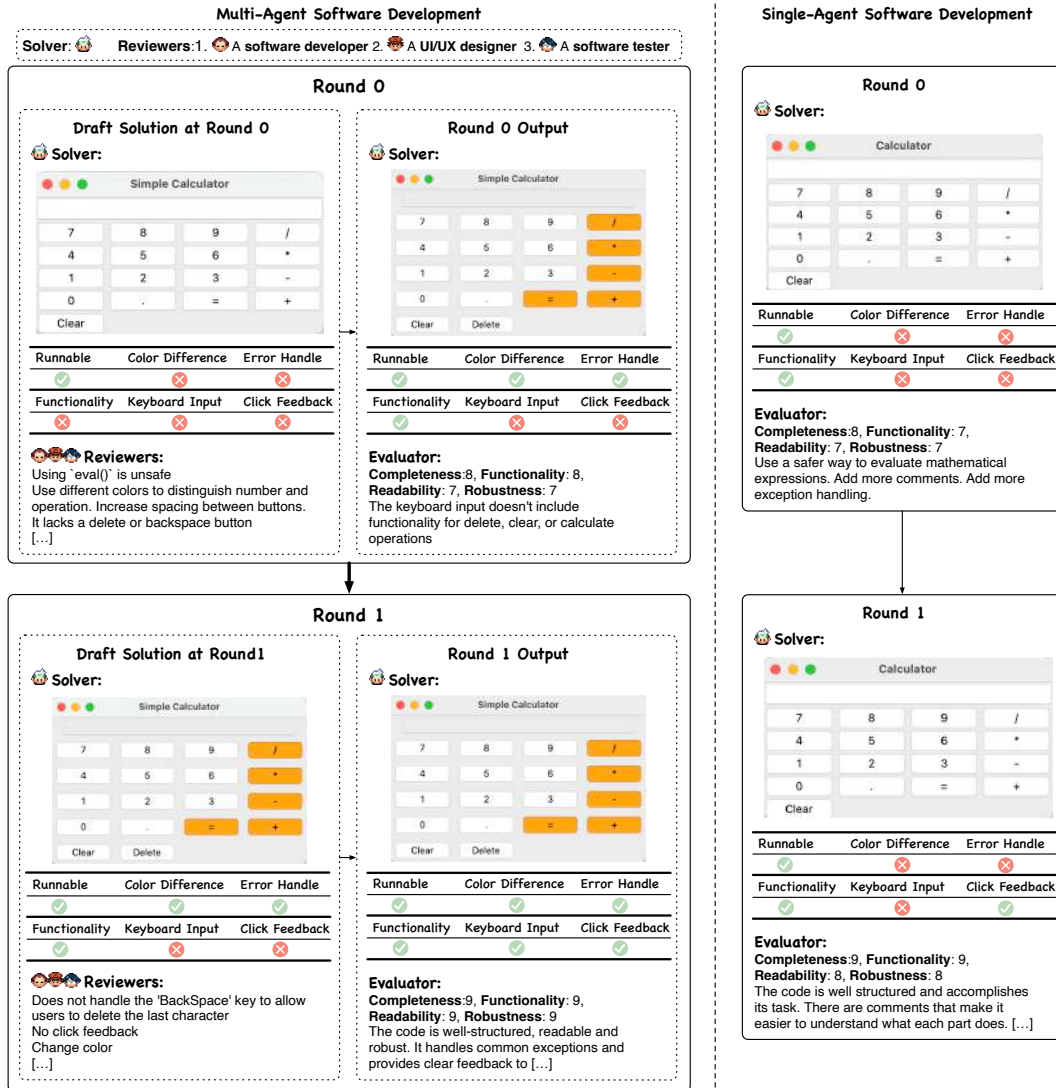


Figure 10: An example of the process of software development.

```

import tkinter as tk
from tkinter import messagebox
root = tk.Tk() # Create the main window
root.title("Calculator") # Set the title of the window
input_text = tk.StringVar() # This holds the string type variable that displays in the text
field
# Create a text field and set its position
input_field = tk.Entry(root, textvariable=input_text)
input_field.grid(columnspan=4, ipadx=70)
def button_click(item):
    current = input_text.get() # Get the current value in the text field
    input_text.set(current + str(item)) # Append the clicked button's value
def clear_all():
    input_text.set("") # Clear the text field
def safe_evaluate(expression):
    try:
        # Using the eval function can be dangerous, so we limit the functionality to only simple
        arithmetic expressions
        return str(eval(expression, {"__builtins__": None}, {"pow": pow, "abs": abs}))
    except Exception as e:
        # If the expression is not safe to evaluate, catch the error and return it
        return str(e)

def calculate():
    try:
        # Use the safe evaluate function to get the result of the expression
        result = safe_evaluate(input_text.get())
        input_text.set(result)
    except Exception as e:
        messagebox.showerror("Error", e)
buttons = ['7', '8', '9', '/', '4', '5', '6', '*', '1', '2', '3', '-', '0', '.', '=', '+']
row_val = 1
col_val = 0
for i in buttons:
    # Create a button for each character in the buttons list
    button = tk.Button(root, text=i, width=5, command=lambda item=i: button_click(item) if item != "=" else calculate())
    button.grid(row=row_val, column=col_val) # Set the position of the button
    col_val += 1
    if col_val > 3:
        col_val = 0
        row_val += 1
clear = tk.Button(root, text='Clear', width=5, command=lambda: clear_all()) # Create a clear
button
clear.grid(row=5, column=0) # Set the position of the clear button
root.mainloop() # Start the GUI

# Evaluation result: Score: [9, 9, 8, 8]
# Advice: The code is well structured and accomplishes its task. There are comments that make it
easier to understand what each part does. However, it could use more comments to explain why
certain decisions were made. Also, the use of `eval` function can lead to potential security
risks, consider using a safer alternative or limiting its functionality.

```

Figure 11: The code generated by single agent.

```

import tkinter as tk
from tkinter import messagebox
root = tk.Tk()
root.title("Simple Calculator")
input_text = tk.StringVar()
entry = tk.Entry(root, textvariable = input_text, state='readonly')
entry.grid(columnspan=4, ipadx=70)
def button_click(item):
    current = input_text.get()
    input_text.set(current + str(item))

def clear():
    input_text.set("")

def delete():
    current = input_text.get()
    input_text.set(current[:-1])

def calculate():
    current = input_text.get()
    if current[-1] in ['+', '-', '*', '/']:
        current = current[:-1]
    try:
        value = eval(current)
        input_text.set(value)
        entry.config({'background': 'white'})
    except ZeroDivisionError:
        messagebox.showerror("Error", "Division by zero is not allowed")
        entry.config({'background': 'red'})
    except SyntaxError:
        messagebox.showerror("Error", "Invalid syntax")
        entry.config({'background': 'red'})
    except Exception as e:
        messagebox.showerror("Error", str(e))
        entry.config({'background': 'red'})
def create_button(root, text, command, row, column, color="white"):
    b = tk.Button(root, text=text, width=5, bg=color, command=command, activebackground='yellow')
    b.grid(row=row, column=column, padx=5, pady=5)
    return b
buttons = [
    '7', '8', '9', '/',
    '4', '5', '6', '*',
    '1', '2', '3', '-',
    '0', '.', '=', '+',
]

row_val = 1
col_val = 0

for button in buttons:
    if button in ['/', '*', '-', '+', '=']:
        color = "light blue"
    else:
        color = "white"
    command = lambda button=button: button_click(button) if button != "=" else calculate()
    b = create_button(root, button, command, row_val, col_val, color)
    col_val += 1
    if col_val > 3:
        col_val = 0
        row_val += 1
create_button(root, "Clear", clear, 5, 0)
create_button(root, "Delete", delete, 5, 1)
def key_press(event):
    if event.char in ['0', '1', '2', '3', '4', '5', '6', '7', '8', '9', '+', '-', '*', '/']:
        button_click(event.char)
    elif event.keysym == "Return":
        calculate()
    elif event.keysym == "BackSpace":
        delete()
    elif event.keysym == "Escape":
        clear()

root.bind('<Key>', key_press)
root.mainloop()

```

Figure 12: The code generated by the multi-agent group.

Project Consulting Multi-Agent Horizontal Structure

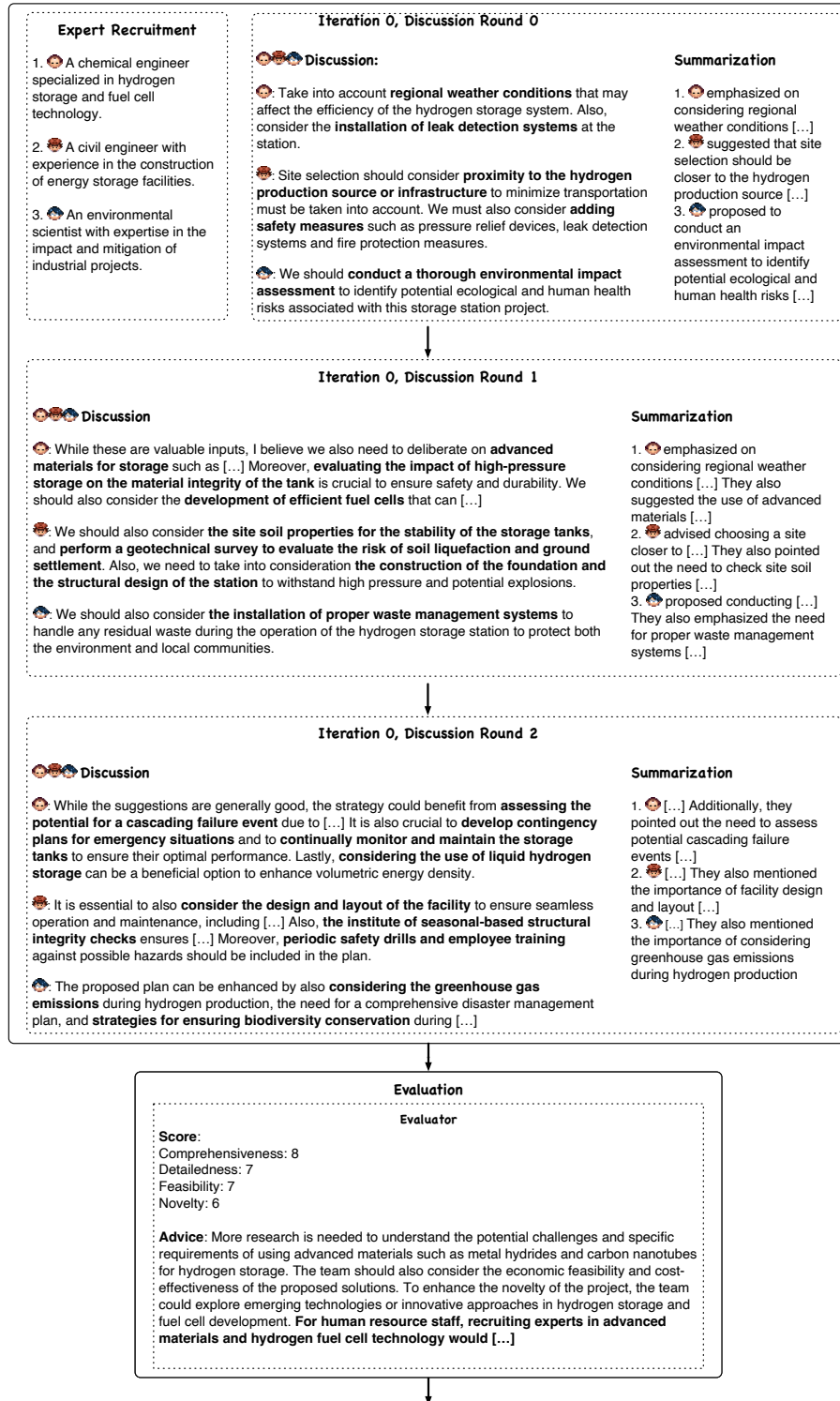


Figure 13: (Page 1) An example process of project consulting using horizontal decision-making structure. The agents are providing suggestions on the problem "Give me some suggestions if I want to build a compressed hydrogen storage station in Ohio"

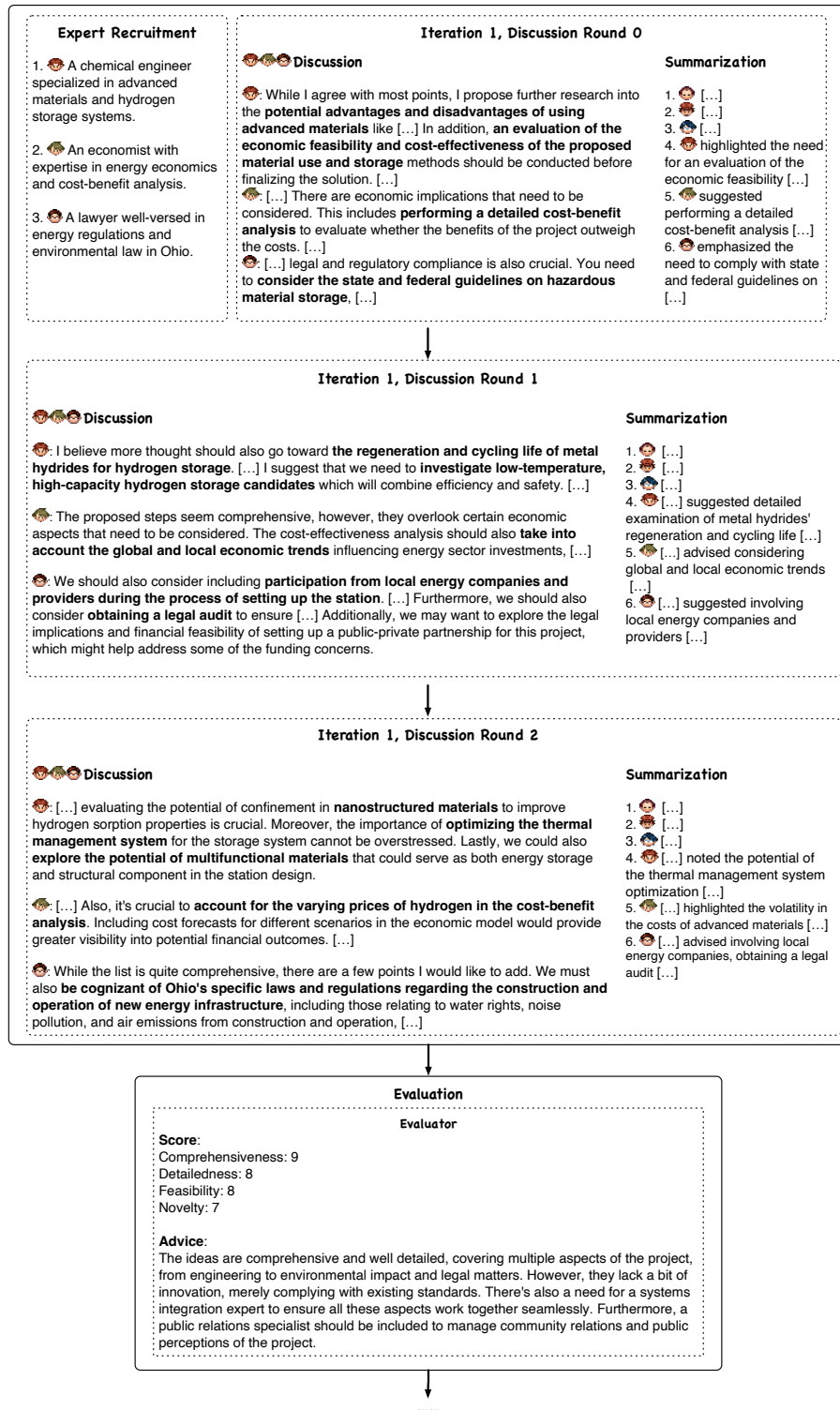


Figure 13: (Page 2) An example process of project consulting using horizontal decision-making structure. The agents are providing suggestions on the problem "Give me some suggestions if I want to build a compressed hydrogen storage station in Ohio"

Project Consulting Single-Agent

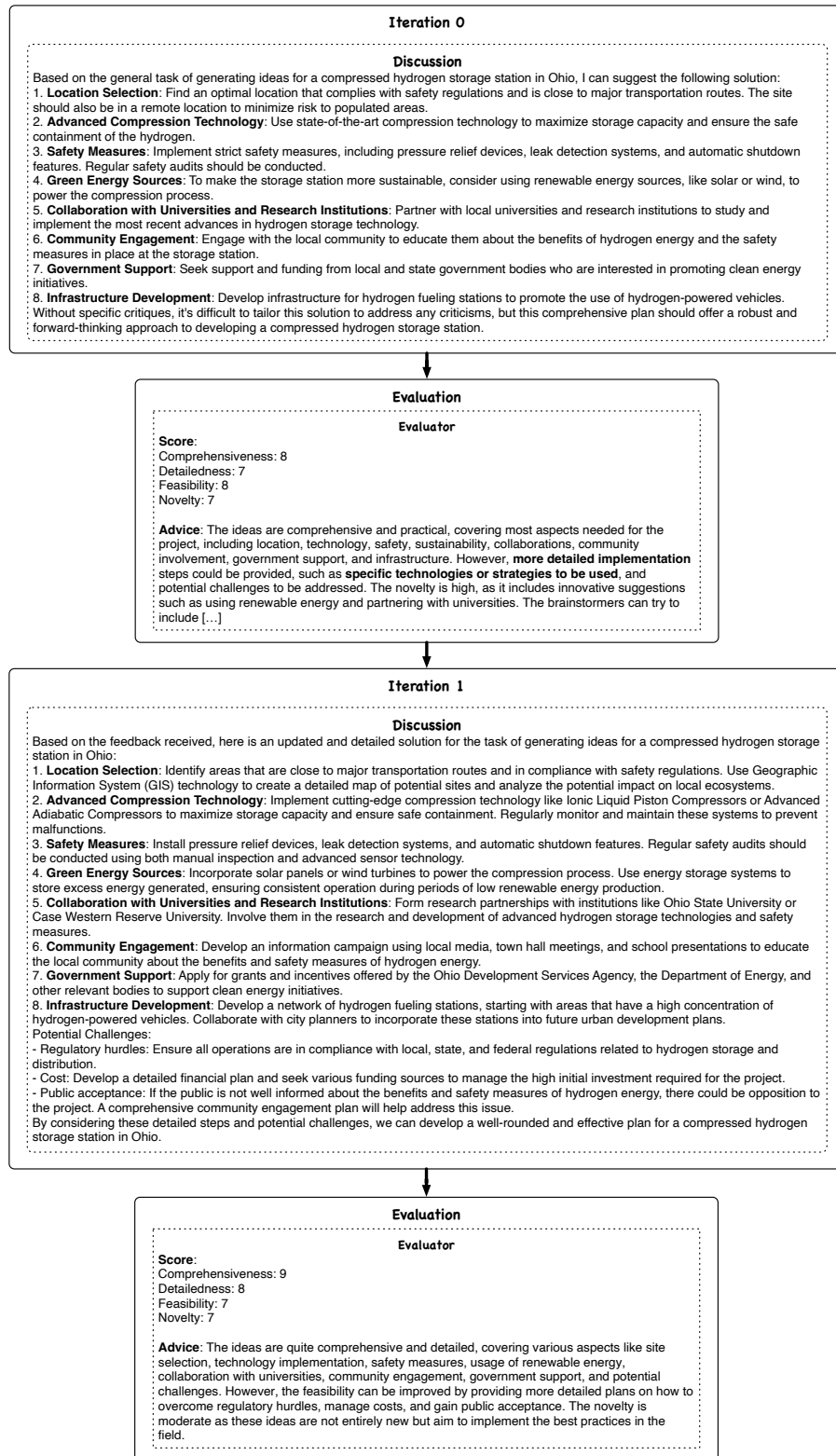


Figure 14: An example process of single-agent project consulting. The agent is required to provide suggestions on the problem "Give me some suggestions if I want to build a compressed hydrogen storage station in Ohio".

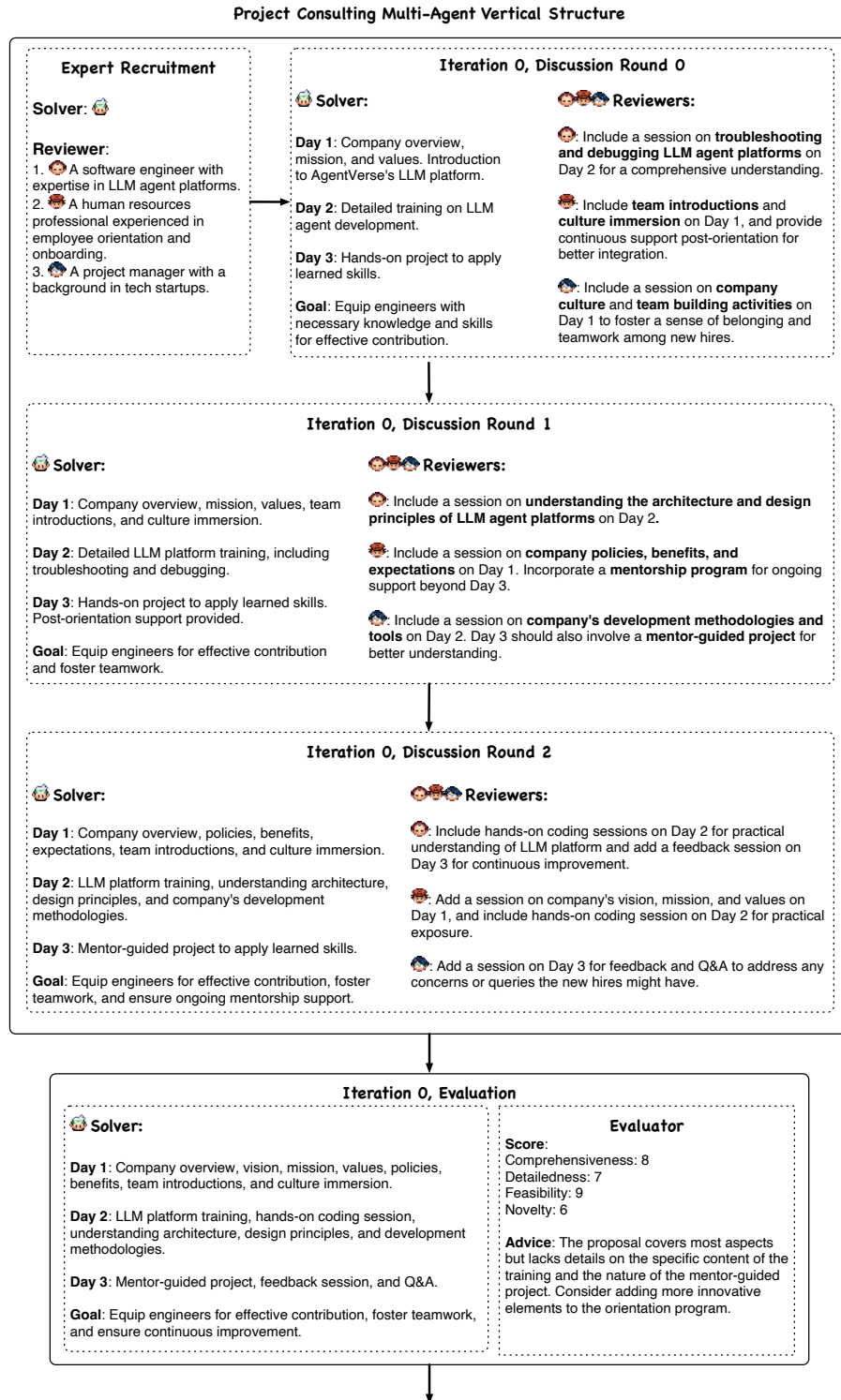


Figure 15: (Page 1) An example process of project consulting using vertical decision-making structure. The agents are providing suggestions on the problem "Generate a proposal about 3-day employee orientation for newly hired engineers at AgentVerse. AgentVerse is a open-source team devoted to developing a LLM multi-agent platform for accomplishing".

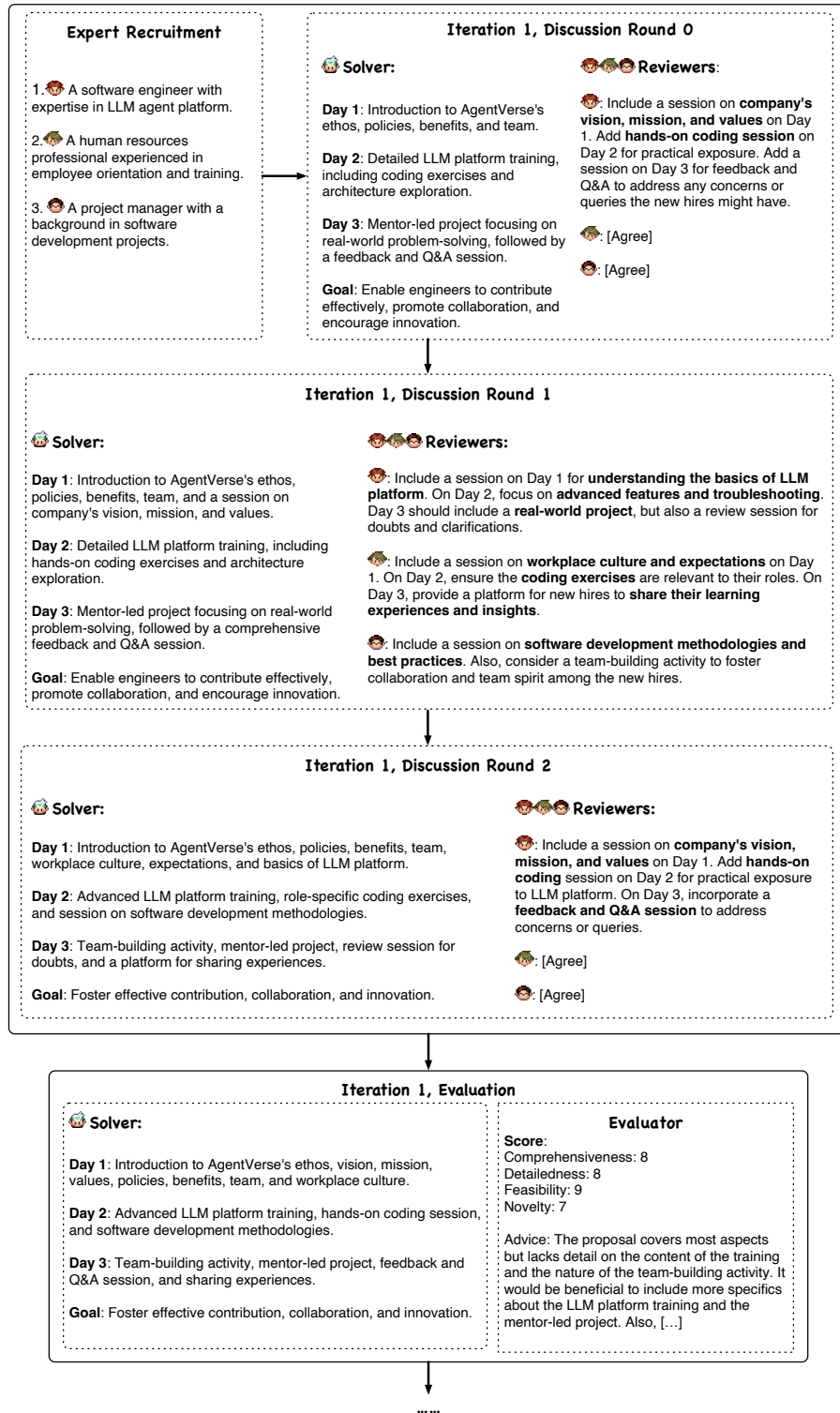


Figure 15: (Page 2) An example process of project consulting using vertical decision-making structure. The agents are providing suggestions on the problem "Generate a proposal about 3-day employee orientation for newly hired engineers at AgentVerse. AgentVerse is a open-source team devoted to developing a LLM multi-agent platform for accomplishing".

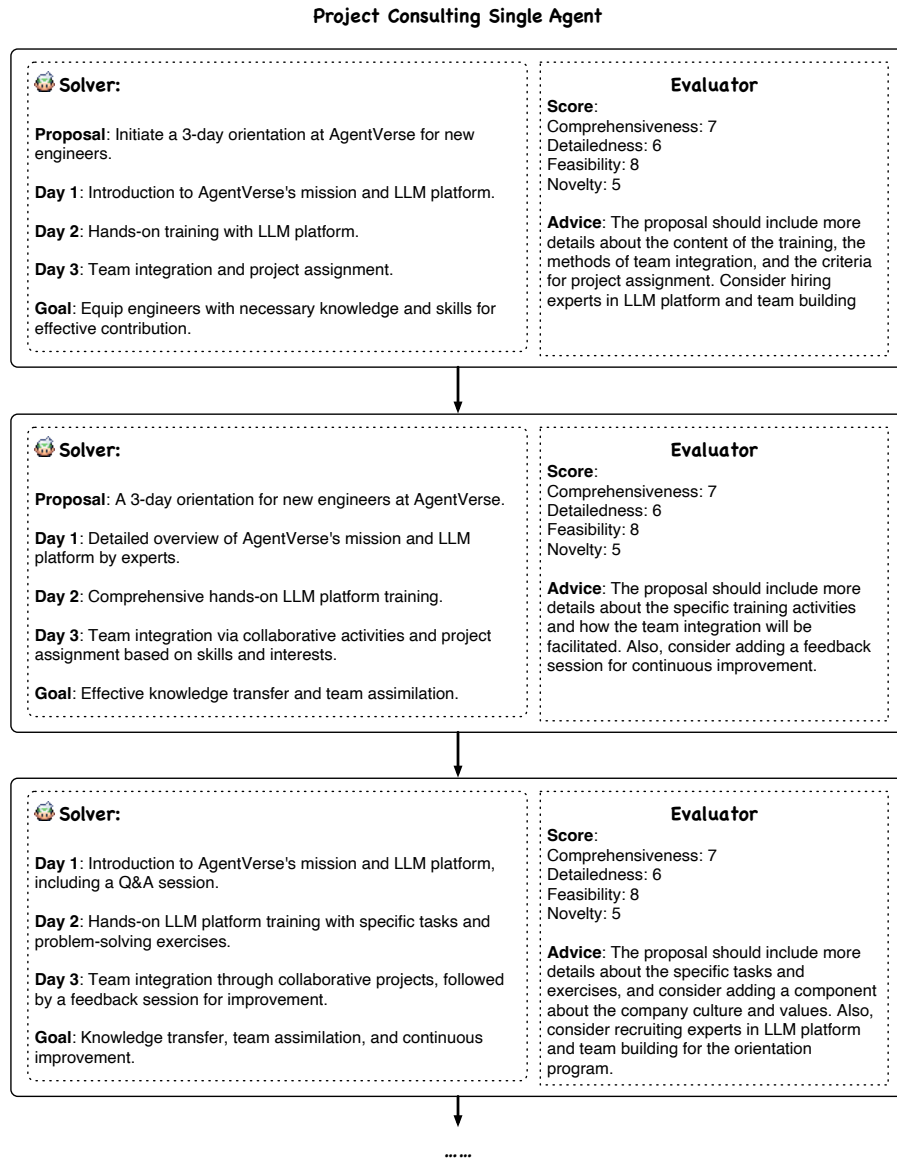


Figure 16: An example process of single-agent project consulting. The agent is required to provide suggestions on the problem "Generate a proposal about 3-day employee orientation for newly hired engineers at AgentVerse. AgentVerse is an open-source team devoted to developing a LLM multi-agent platform for accomplishing".

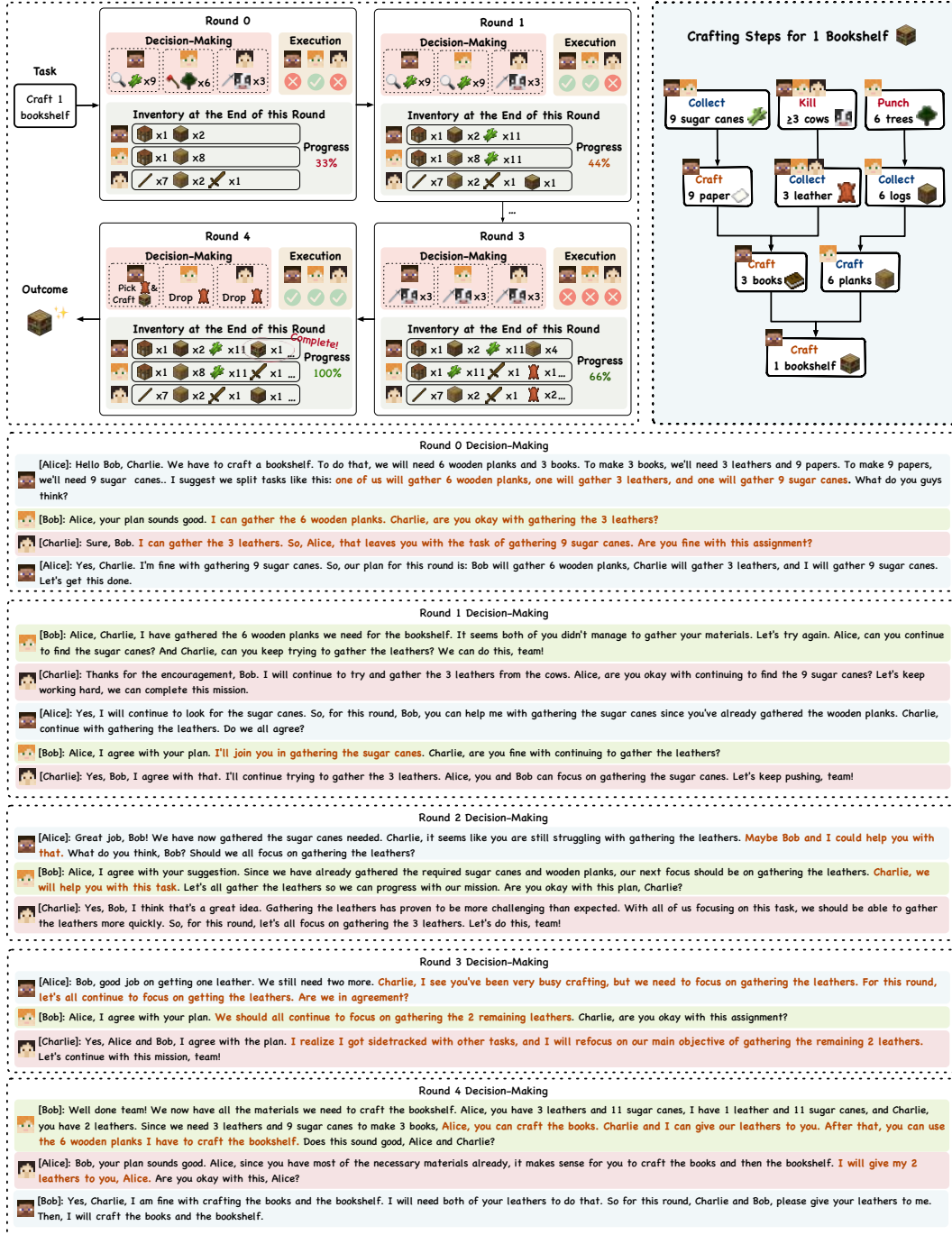


Figure 17: An example process of three agents crafting a bookshelf in Minecraft.