What is Multi-Agent Collaboration? | IBM

The evolution from large language models (LLMs) to artificial intelligent agent (AI agent) integration changed the landscape of artificial intelligence (AI). Now, mutli-agent systems (MAS) are ushering in a whole new wave of AI-native products and software development services.

Traditional LLM applications powered by generative AI (gen AI) were mostly focused on boosting productivity, answering questions or summarizing information. The introduction of agents and the capacity for AI agent communication provide the power to create autonomous workflows that significantly cut down on the manual work involved in research, support, analysis and operations. Now, multi-agent systems handle complex real-world tasks such as customer service triage, financial analysis, technical troubleshooting and compliance monitoring and have become scalable, autonomous and continuously improvable.

What is multi-agent collaboration?

The coordinated actions of several independent agents in a distributed system, each having local knowledge and decision-making capacities, are referred to as **multi-agent collaboration**.

In multi-agent collaboration, the agents cooperate by using established communication protocols to exchange state information, assign responsibilities and coordinate actions. The cooperation usually includes methods for work decomposition, resource distribution, conflict resolution and cooperative planning. It can be explicit through message passing or implicit through modifications to the shared environment. These systems prioritize scalability, fault tolerance and emergent cooperative behavior in their design to operate without centralized control. Let us consider an analogy: Suppose that a fleet of drones is searching a disaster site for survivors or information. Each drone takes its own path, avoids other drones, reports what it finds and changes direction in the case of an unexpected event. Think of this scenario as multi-agent collaboration: Every drone operates alone as well as collectively, in a sense like an assistant. Without a single leader managing them, they work together, coordinate with each other and share what they see. This approach is how an autonomous fleet of agents works collaboratively, intelligently and quickly to solve complex problems.

This collaborative architecture is redefining product architecture, generating various use cases that run almost anytime, adapt to growing demands and continuously learn and optimize without manual intervention. The process of agentic automation is enabled by specialized agents with adaptive capabilities designed to handle specific tasks with precision and autonomy. Specialized AI agents work together in real-time to provide intelligent, customized and end-to-end services in chatbots (by using rag framework), a new type of multi-agent application.[1]

Why do agents need to collaborate?

Cooperation among multiple agents is an important requirement when designing and deploying an intelligent system, especially in environments that are highly complex, distributed and have privacy constraints. Multi-agent collaboration provides numerous architectural, computational and operational benefits in contrast with other agentic architecture types, specifically a single-agent system. This dynamic is true in complex, distributed, real-time systems where multiple distinct levels of privacy are inherent. Multi-agent systems (MAS) enable decentralized, autonomous agents to work together to achieve collective or interdependent goals, helping to overcome some of the structural limitations of constrained single-agent systems. For example, monolithic, single agent systems that only scale up to a limited degree or have limits on latency and functional generality. Each agent maintains a level of autonomy, completing local computations, cooperating with other agents by using communication protocols to share partial knowledge about their environment, collaborate on decision making and coordinate a distributed control strategy. The ability to maintain modular scalability permits seamless integration of new agents or subsystems while providing adaptive behavior in dynamic environments in real-time. For example, in a smart healthcare system, either a subset of, or all agents might have domain-specific assignments; such as monitoring physiological signals, identifying anomalies, recommending therapy and managing patient identifiable data in accordance with policy. Their cooperation also enables continuity, accuracy and fault tolerance throughout the entire process. The ability to normalize computations across agents increases computational efficiency by sharing the parameterization across agents and obviates reliance on centralized computations.[2]

How do multi-agents collaborate?

To comprehend how multi-agent systems work, let's dissect the cooperative process into a sequence of well-coordinated steps, each of which emphasizes how independent individuals interact, assign and work together to accomplish challenging tasks.

Agents **collaborate and coordinate** through structured channels where each agent is an intelligent component with five key elements.

a. **The foundation model (*m*):** This element is the agent's main reasoning engine, allowing for the generation and comprehension of natural language.

b. **Objective (o):** Agent's goal or task that they are focused on doing is defined by the objective (*o*).

c. **Environment (*e*):** This element indicates the situation in which the agent functions. This could involve other agents, tools, shared memory or application programming interfaces (APIs).

d. The information an agent receives from its surroundings or from other agents is known as input **perception (*x*).**

e. **Output or Action (*y*):** The agent's conduct or response in light of its current objective and line of reasoning.

Collaboration occurs when several AI agents cooperate as a team to accomplish a task. During the collaboration phase, the system receives a **task** from the user or environment. The system decides **which agents** are needed and what **roles** they'll play.

The system divides the complex problems into manageable pieces. This is achieved either by a planner or the language model with reasoning capability. Communication happens either through shared memory or intermediate outputs. Assigned tasks are carried out by agents either concurrently, sequentially or in dynamic.

The outcomes of various agents are compiled to create a significant response.
The orchestrator or final agent initiates an action or gives the user the complete response.[3]

Various collaboration strategies

Agents collaborate with other agents by using various strategies that determine how they will interact, coordinate and contribute to shared objectives. Various collaboration strategies include:

- **Rule-based collaboration**:

In this collaboration type, the agents interactions with one another is tightly controlled by a specific set of rules or guidelines. These rules dictate how agents act, communicate and make choices in a predictable way. The scope of learning or adapting is limited as agents stick to a set policy based on certain conditions or inputs. This method is often carried out using if-then statements, state machines or logic-based frameworks. This collaboration works best for tasks that are highly structured or predictable, where maintaining consistency is key.

Pros and cons: This approach delivers great efficiency and fairness, but it struggles with adaptability and scalability, especially in fast-changing or complex situations.

- **Role-based collaboration:**

In this approach, agents are given specific roles or responsibilities that align with a clear organizational or communication framework. Each role comes with its own set of functions, permissions and objectives that are often linked to various parts of the overall system goal. While agents work semi-independently within their designated roles, they also play a part in the bigger picture by coordinating and sharing information with one another. This concept draws inspiration from human team dynamics, where individuals take on different roles such as leader, observer or executor. It's particularly beneficial for breaking down tasks, designing modular systems and allowing agents with diverse expertise to collaborate effectively.

Pros and cons: It allows for modular, expert-driven collaboration, but it might face challenges with flexibility and its reliance on agent integration.

- **Model-based collaboration:**

In this type of collaboration, agents create internal models to understand their own state, the environment around them, other agents and the common goal they're all working toward. These models are typically probabilistic or learned, which helps agents plan their actions even when things are uncertain. Their interactions rely on updating beliefs, making inferences and predicting outcomes, which allows their strategies to be flexible and aware of the context. Some common methods that they use include Bayesian reasoning, Markov decision processes (MDPs) and various machine learning models. This approach is particularly useful in situations where agents need to think about unknown factors, adapt to changes or work together without having complete visibility.

Pros and cons: This approach offers great flexibility and solid decision-making capabilities, but it does come with a significant level of complexity and a hefty computational cost.[4]

Frameworks

Several well-known frameworks are being developed, each using its own distinct methods to help agents work together effectively in real-world applications. Let's explore the commonly used frameworks:

1. IBM Bee Agent framework: It is an open-source application that facilitates the development and administration of multi-agent, scalable processes. It establishes the foundation for applications in which multiple AI agents collaborate to accomplish challenging tasks by using massive LLMs such as IBM® Granite®, gpt-4 and Llama 3. With ready-to-use components for agents, tools, memory management and monitoring, the framework boasts a modular design. Serializing agent states is one of its most notable characteristics. This ability enables complex procedures to be stopped and resumed without erasing any data. Its emphasis on production-level control, extensibility and modularity allows for the creation of sophisticated multi-agent systems for a wide range of applications, with plans for further advancements in multi-agent orchestration.

2. LangChain agents: LangChain is a robust framework for building language model-driven applications that emphasize a strong agent-based architecture. This option means that the agents can perceive its environment and use many tools available to gather information, interpret and act. Within LangChain itself, developers have access to many tools and integrations to make it easier to engineer agents to perform complex reasoning, dynamic decision-making and task accomplishment. LangChain allows the developer to harness the highest capabilities of large language models (LLMs) in developing intelligent systems to accomplish sophisticated tasks such as contextual question answering, multistep workflows and natural language generation.

3. **OpenAI Swarm framework:** This structure presents a new way of coordinating multiple agents in terms of routines and handoffs. Instead of one agent acting independently, each agent can be viewed as a specialized unit working with custom tools and customized directions. The transfer of an existing task or conversation from one agent to another allows a smooth user experience where each agent is specialized for a specific role. This approach ultimately increases the overall efficiency, modularity and responsiveness of the system overall. The term Swarm emphasizes lightweight coordination and effective carry out of a task, which enables it to be deployed on a larger scale in real-world tasks.[5]

Enterprise solution

Watsonx Orchestrate

Watsonx Orchestrate® makes it easy to enable multi-agent collaboration by using a collection of interconnected components that work together to orchestrate AI-enabled workflows. Skills are independent agents that execute specific tasks, like sending emails or querying data; they are described and registered in a Skill Registry that outlines their capabilities and metadata. When a user submits a request, an Intent Parser uses natural language processing (NLP) to read the user's input and relate it to the skills.

The Flow Orchestrator provides the execution logic and flow, including task sequencing, branching, errors and retries to help ensure agents are executed in the required order and failed steps can be retried. The flow orchestrator allows agents to be executed simultaneously when necessary. The Shared Context and Memory Store provides a common space to store data, intermediate outputs and decisions in one space, allowing agents to be aware of each other and keep continuity during their workflow. The LLM assistant uses large language models to help with reasoning, navigation of a changing context and fill knowledge gaps while collaborating.

The Human Interface allows the user to see the flow and manage the agentic workflow if they want to get involved. The components can support multi-agent collaboration to help ensure that watsonx Orchestrate can independently manage complex, multi-agents workflows while allowing a human in the loop.[6]

Future predictions

Emergent collective intelligence: As autonomous agents work together through a well-defined **collaboration framework** with **guardrails** to help ensure alignment, safety and task relevance, intelligent behaviors begin to emerge—exceeding the individual capabilities of any single agent. Accuracy, relevance, efficiency, explainability and overall system coherence are some of the multifaceted metrics that can be used to continuously evaluate and improve the efficacy of these systems.

The collective intelligence gives these systems the ability to solve complex and multidimensional problems by using distributed reasoning and decomposition of tasks resulting in **automation**, **decision making** and **orchestration** of multi-step workflows.