Multi-Agent Systems

Veeralagan.J
Department of Computer Science
Alpha Arts and Science College
Chennai, India.

ABSTRACT

It has only been less than 20 years since Distributed Artificial Intelligence (DAI) emerged as a separate subject of AI. Systems with numerous independent entities interacting inside a domain are what DAI is concerned with. DAI has historically been broken down into two sub-disciplines: Distributed Problem Solving (DPS), Multiagent Systems (MAS). MAS deals with controlling the behavior of groups of various agents or independent entities. Agent-based applications are becoming commonplace thanks to the current technological advancements in a wide range of industries, including e-commerce, supply chain administration, logistics, telecommunications, medical care, and production. Multi-agent systems are also popular because they are considered as a technology and a tool that aid in the research and creation of new theories and models for large-scale distributed systems or human-centered systems.

Keywords- multi-agent systems; agent methodologies; single-agent system; MAS Learning

I. INTRODUCTION

The strength and complexity of AI techniques have grown significantly during the last few years. Using the ALVINN system, a car recently drove itself more than 95% of the way across the United States, to name just one of the numerous thrilling accomplishments [Pormerleau, 1993]. The topic of artificial intelligence called "multiagent systems" (MAS) is still in its infancy, but it tries to provide both concepts for building multiagent complex systems and mechanisms for coordinating the actions of autonomous agents[1]. Multi-agent systems are one type of distributed system, and what makes them unique is that each component is autonomous, self-interested, and focused on achieving its own goals. These systems also stand out because they lack a centralized design and are open systems [2]. The primary subject of this study, multi-agent systems (MAS), consists of autonomous creatures known as agents. Agents collaborate to complete tasks similarly to computing entities in DPS, but they provide additional flexibility because they have the innate capacity to learn and take independent decisions. In order to do the task assigned to them, agents make decisions and take actions on the environment [3]. To build MAS, a variety of complicated issues must be addressed, including agent coordination [4], learning, and security [5].

II. SINGLE-AGENT VS. MULTIAGENT SYSTEMS

We must first think about MAS' most obvious rival: centralized, single-agent systems, before we can analyze and classify MAS. In centralized systems, one agent is in charge of all decision-making, with the others serving as distant slaves. A "single-agent system" should be viewed for the purposes of this survey as a centralized system in a domain that also supports a multiagent approach.

Even with only one agent, a system may have various entities, such as numerous actuators or even robots. The central process is the lone agent, nevertheless, if every creature transmits its observations to it and receives its commands from it. Each entity is modeled by the central agent as a single "self." In this section, the single-agent and multi-agent strategies are contrasted.

A. Single-Agent System

In a single-agent system, the agent typically simulates the environment, itself, and the interactions between the three. However, for the purposes of this article, agents are also regarded to have extra-environmental components even though the agent itself is obviously a part of the environment. They are autonomous beings with their own objectives, courses of action, and knowledge. In a single-agent system, the agent does not recognize any further similar entities. Therefore, even if there are other agents in the world, they are not portrayed as having goals or anything else; instead, they are only seen as a component of the environment. Although agents are a component of their environment, it is important to note that they are explicitly described as having their own objectives, course of action, and domain knowledge (see figure 1).

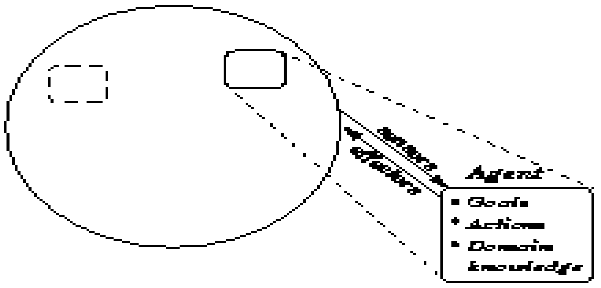

Figure 1: Single-Agent System

B. Multi-Agent System

The main way that multiagent systems differ from single-agent systems from the viewpoint of a single agent is that other agents can influence the environment's dynamics. Other agents purposefully alter the environment in unpredictable ways, in addition to any inherent uncertainty in the domain. Thus, it is possible to consider the environments of all multiagent systems as dynamic. Figure 2 demonstrates the idea that each agent is both modeled as a separate entity and a component of the ecosystem.

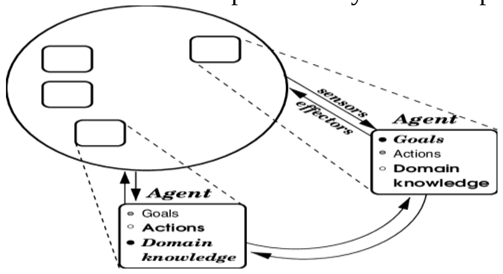

Figure 2: Multi-Agent System

III. MAS AND METHODOLOGIES

Various publications have emerged in recent years attempting to suggest novel procedures and methods for the creation of multi-agent systems. Software engineering techniques to organize the development process, artificial intelligence techniques to give systems the ability to handle unforeseen circumstances and make decisions, concurrent programming to address task coordination carried out on various machines with various scheduling policies—all of these technologies are integrated into the construction of MAS [6]. Three pieces of work pertaining to agent-oriented software engineering have been included in this Paper.

A. Accountability

In human organizations, accountability is a well-known vital resource. Additionally, it is suggested that agent systems be created with accountability as a quality that is by design assured. To show how to create agents and the organization to which they belong, being mutually accountable, one might utilize the interaction protocol known as ADOPT, JaCaMo[7].

B. Development Methodology

SemanticWeb-Enabled Agent Modeling Language (SEA_ML), a domain-specific modeling language, can be used to create MAS that operate in semantic web contexts.

C. Agent development framework

The framework's support for linked data is the primary contribution. The capacity to supply the agent's beliefs from the connected data environment and to apply those beliefs during the planning process is referred to as linked data support.

IV. MAS LEARNING

Agents may have to learn since they don't fully understand their environment or are unaware of how other agents behave. The context in which the learning takes place might be either cooperative, where we also want the agents to share what they have learned, or competitive, where we want them to outperform one another. We propose analysis and learning algorithms for these distinct environments' learning agents.

A. Cooperative Learning

Several agents work together in cooperative multi-agent systems to try and solve problems or maximize value through their interactions. Multi-agent problem complexity can increase quickly with agent count or behavioral sophistication due to interactions among the agents.

B. Competitive Learning

Two or more agents are pitted against one another in competitive learning, hopefully starting an advantageous "arms race" between them. For instance, it could be preferable for two agents to learn to play chess at roughly the same rate, resulting in a gradual increase in difficulty.

REFERENCES

[1]. Peter Stone, Manuela Veloso "Multiagent Systems: A Survey from a Machine Learning Perspective" Autonomous Robotics volume 8, number 3. July, 2000.
[2]. Wooldridge, M. An Introduction to MultiAgent Systems, 2nd ed.;Wiley Publishing: Hoboken, NJ, USA, 2009.
[3]. ALI DORRI, SALIL S. KANHERE, "Multi-Agent Systems: A survey", 10.1109/ACCESS.2017.DOI.
[4]. A.-M. Zou, K. D. Kumar, and Z.-G. Hou, "Distributed consensus control for multi-agent systems using terminal sliding mode and chebyshev neural networks," International Journal of Robust and Nonlinear Control, vol. 23, no. 3, pp. 334–357, 2013.
[5].M. H. Bowling, "Convergence and no-regret in multiagent learning." In NIPS, 2004, pp. 209–216.
[6]. Kravari, K.; Bassiliades, N. A Survey of Agent Platforms. J. Artif. Soc. Soc. Simul. 2015, 18, 11. [CrossRef].
[7]. Boissier, O.; Bordini, R.H.; Hübner, J.F.; Ricci, A.; Santi, A. Multi-agent Oriented Programming with JaCaMo. Sci. Comput. Program. 2013, 78, 747–761. [CrossRef]
[8]. Challenger, M.; Demirkol, S.; Getir, S.; Mernik, M.; Kardas, G.; Kosar, T. On the Use of a Domain-specific Modeling Language in the Development of Multiagent Systems. Eng. Appl. Artif. Intell. 2014, 28, 111–141. [CrossRef]