Volume : 3 | Issue : 7 | July 2014 • ISSN No 2277 - 8179

Research Paper

Engineering

KEYWORDS : Multi-agent system, Control application

Multi-Agent System: A Recent Trend in Control Engineering

Rakesh B. Patel

Assistant Professor, Instrumentation and Control Engineering Department
Sardar Vallabhbhai Patel Institute of Technology, Vasad, Dist. Anand. Gujarat, India.

ABSTRACT

Multi-agent control (MAC) is a new approach to control systems comprised of interacting autonomous agents. The MAC promises to have far-reaching effects on the way that businesses use computers to support decision-making and researchers use electronic laboratories to do research. Some have gone so far as to contend that MAC is a new way of designing control applications. Computational advances make possible a growing number of agent-based applications across many fields. This paper describes the foundations of MAC, identifies MAC toolkits and development methods illustrated through some applications review.

Introduction

The Multi-Agent based Control (MAC) is an interdisciplinary field based on the combination of Artificial Intelligence and Automation Control. Ideas and research results of all sects of Artificial Intelligence have been applied to control domain and then each branch of Intelligent Control was appeared, such as Expert System Control, Fuzzy Control, Neural Network Control and Simulating Human Intelligent Control, etc. Agent, especially Multi-agent systems (MAS), as a new technology of Distributed Artificial Intelligence (DAI), has become a new hotspot of Artificial Intelligence research, and lot of important production has been made. Agent technology has been extensively applied to many fields, from simple personal E-mail filter to complex systems such as aerial traffic control system and flexible manufacturing systems (FMA) (Li Zhang, 2004). Thus, a multi-agent approach would be sensible for problems that are inherently (physically or geographically) distributed where independent processes can be clearly distinguished. Such problems include, for example, distributed sensor networks, decision support systems, air traffic control, or other networked or distributed control systems. A distributed approach is not in itself enough, however, and there should also be requirements for intelligence or adaptively in the sub-processes that involve explicit reasoning about behavior, for example (Ruth, 1997).

What is an agent?

Although there is no universal agreement on the precise definition of the term "agent," definitions tend to agree on more points than they disagree. Some modelers consider any type of independent component (software, model, individual, etc.) to be an agent. An independent component's behavior can range from primitive reactive decision rules to complex adaptive artificial intelligence (AI). Others insist that a component's behavior must be adaptive in order for it to be considered an agent; the agent label is reserved for components that can in some sense learn from their environments and change their behaviors in response. Casti (1997) argues that agents should contain both base-level rules for behavior as well as a higher-level set of "rules to change the rules". The base level rules provide responses to the environment while the "rules to change the rules" provide adaptation. Jennings (2000) provides a computer science view of agency emphasizing the essential characteristic of autonomous behavior. The fundamental feature of an agent is the capability of the component to make independent decisions. This requires agents to be active rather than purely passive.

Agent characteristic

From a practical modeling standpoint, we consider agents to have certain characteristics (Figure 1) (Charles, 2006):

- An agent is identifiable, a discrete individual with a set of characteristics and rules governing its behaviors and decision-making capability. Agents are self-contained. The discreteness requirement implies that an agent has a boundary and one can easily determine whether something is part of an agent, is not part of an agent, or is a shared characteristic.
- An agent is situated, living in an environment with which it interacts along with other agents. Agents have protocols for interaction with other agents, such as for communication, and the capability to respond to the environment. Agents have the ability to recognize and distinguish the traits of other agents.
- An agent may be goal-directed, having goals to achieve (not necessarily objectives to maximize) with respect to its behaviors. This allows an agent to compare the outcome of its behavior relative to its goals.
- An agent is autonomous and self-directed. An agent can function independently in its environment and in its dealings with other agents, at least over a limited range of situations that are of interest.
- An agent is flexible, having the ability to learn and adapt its behaviors based on experience. This requires some form of memory. An agent may have rules that modify its rules of behavior.

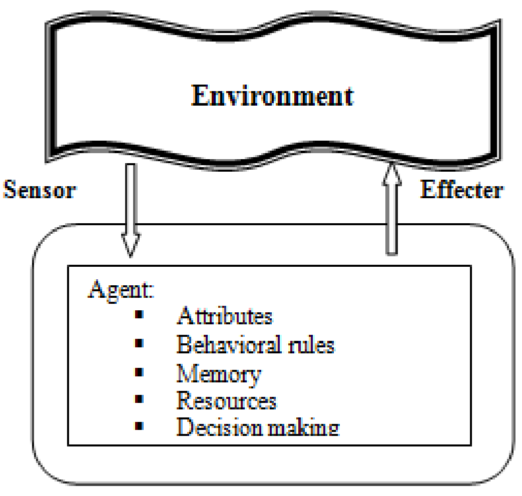

Figure: 1 an agent

Multi-agent system

The group of agents is called as multi agent system. It may be considered as intelligent tool for the solution of such problems as planning, scheduling, decision-making and control in the framework of production processes. The MAS approach seems to be the most feasible. It respects the complicated characteristics of the goal that we aim to achieve. There are some significant reasons that motivate us to choose the MAS approach to the solution of decision-making, such as (Jennings, 2000):

Modularity:

Each agent is an autonomous module and can work without interventions of the external world. Each agent can have different capabilities or functionalities and through cooperation the agents are able to achieve a variety of goal.

Parallelism:
The MAS approach enables to work in parallel. A complicated problem could be solved in an acceptable time by using a number of agents, e.g., gathering information from various resources allocated in different places.

Flexibility:
The MAS approach is able to react in a flexible manner to each change in the environment. Through cooperation the agents can assist each other to compensate the lack of capability or knowledge. They can share information or own capacity to resolve a newly appeared situation, if one agent is not able to do so. Beside that, each intelligent agent can do reasoning about whom and when it has to cooperate with, in order to achieve the effective performance.

Of course, there are also some difficult questions associated with the MAS approach, e.g., which types of agents are needed, how many agents are optimal, what is a functionality of each agent, cooperation between agents, etc.

Multi agent based control system
In a control context, typically a system is supposed to behave in a certain way. It should accomplish some task, which may involve reaching a certain number of goals. The task has to be accomplished while making sure that any possible constraints are not violated. Tasks may be provided by a human or some artificial entity, or they may follow from some behavioral characteristics or reasoning of the system. Goals can be short-term goals, e.g., to bring the system in a certain state, or long-term, e.g., to maximize the long-term performance or to minimize the long-term operation costs. Note that tasks need not have one single goal. They may have multiple, possibly conflicting, goals. In that case they are referred to as so-called multi-objective tasks. Actions that can be performed on the system have to be chosen in such a way that the task of the system is achieved, keeping in mind the dynamics of the system, and possible constraints on the actions. Finding the actions that achieve the goal is called the Dynamic Control Problem (DCP) (Negenborn, 2004). A typical DCP setting is shown in Figure 2.

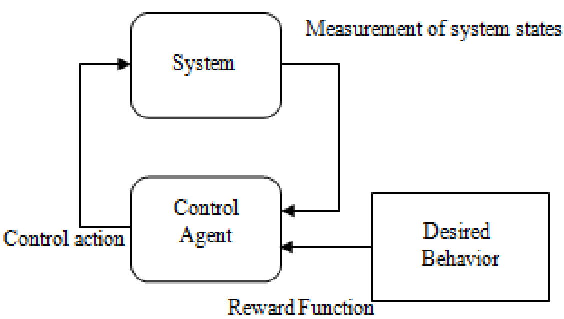

Figure: 2 Dynamic Control Problem

The general DCP can be formulated as:
Find the actions such that the goal is achieved optimally subject to

a model of the dynamic system including constraints on the actions and states

This problem can be seen as an optimization problem, since the actions have to be chosen such that they achieve the goal in the best possible way. Note that the goal is independent of the model of the dynamic system. This section defines the elements that play a role in controlling a general system. As mentioned, controlling a system comes forth from having the desire to have the system achieve a certain goal. In order to obtain the goal we will assume that there is a system model of the system under consideration. Such a model can be used to define more precisely what the goal is and to predict how the system will behave given certain actions. A control problem model defines what the exact control problem is, often based on the system model. A control problem model is used by agents that solve the problem. The agents are organized in agent architecture and follow some communication protocol.

Solving DCPs is done through the use of controllers, or agents. In general, agents are problem solvers that have abilities to act, sense, reason, learn, and communicate with each other in order to solve a given problem. Agents have information set containing their knowledge (including information from sensing and communicating), and an action set containing their skills. Agents may be organized in architectures, e.g., through communication links. We can again distinguish three agent architectures:

- A centralized agent architecture, in which there is only one single agent,
- A decentralized agent architecture, in which there are numerous agents that do not have any interaction among one another,
- A hierarchical agent architecture, in which there are different, layers of agents. Higher layers may supervise and receive information from lower layers. Lower layers may follow instructions from and provide information to higher layers. Agents on the same layer may be allowed to communicate directly with one another, or through the higher layers. See Figure 3 for an example of hierarchically structured agent architecture.

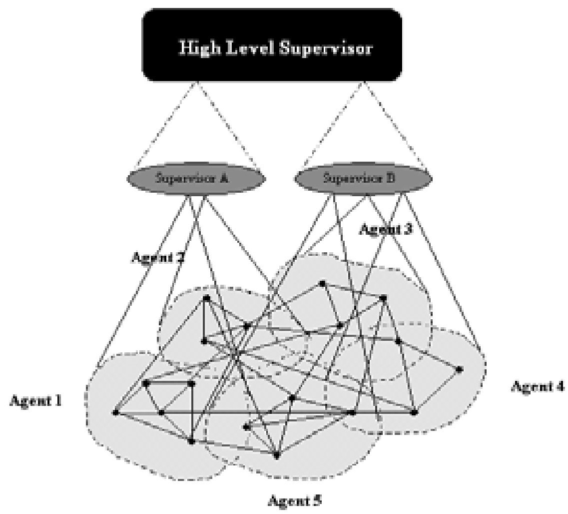

Figure 2: Hierarchically structured agent architecture.

Note that communication between two agents on the same layer can be replaced by a virtual communication agent one layer higher in order to satisfy a no-communication on-a-layer assumption. Note that when considering hierarchical architectures, it is not only important to determine which information is communicable, but also in which order information is accessible to agents. That is, there needs to be a communication protocol (Schutter, 2006).

Applications
A multi-agent architecture has been proposed to cope with the complexity and diverse expertise required for developing intelligent controller for control applications. The rationale for such architecture is discussed and different kinds of agents are designed. This architecture can make the automated system more robust and reliable, and also make it possible to easily add new agents to enhance the analysis quality without affecting the operability of other agents, thereby increasing the scalability and flexibility of the system (Chunhua Zhao). The different multi-agent based controller applications are discussed shown in appendix: A.

Multi-agent Tools
Up to now a various and wide range of environments have been

developed for design, modelling and simulation of agents. Some examples are as follows:

- Agent0, an old agent language and interpreter built to run on top of Common Lisp (Shoham, 1991)
- April, a process oriented language which Act the processes communicate with each other in a multi-agent system (Jennings, 1995).
- JADE (Java Agent Development Environment), a suitable environment for modelling agents using Java as programming Language and supporting FIPA standards for message passing (Nikolaos, 2006)
- Xraptor, a simulator for continuous multi-agent systems that is written in C++ (Mossinger, 1995)
- IUITmac, which is proper for design and implementation of controllers for Mechatronic systems.
- LALO is a language which uses the Agent Oriented Programming (AOP) paradigm (Yoav Shoam, 1996)
- Agent Building Environment -- a toolkit for software developers that makes it easy to build an application based on agents, or to add them to an existing application based on C++ and JAVA(IBM, 1996)
- Oz Programming System -a concurrent constraint programming language designed for applications that require complex symbolic computations, organization into multiple agents, and soft real-time control
- Agents Kernel Language (AKL) - concurrent constraint programming language developed at the Swedish Institute of Computer Science (SICS).
- Fresco - CORBA-based language implemented in C++ which provides support for object distribution, a standard high-level notation for object definition, Tcl-based scripting and multi-threading.
- MatLab - particularly important for research purposes, because it provides almost unlimited flexibility for agent programming and utilization of direct commands (Hupkens, 2004).

Some special purpose environments are also developed for agent-based design, modelling and simulation in various fields. In the domain of control engineering, modelling and simulation packages such as simulink and 2Osim are available. The recent year object oriented technology are strongly preferred in agent based environments because of their popularity of UTMAC, an object oriented C^{++} library for implementation and simulation of multi agent controller systems which provides several classes for implementation controller-agents with a simple. Now while it simulates the designed controller in software or hardware in loop environments another issue of this work is utilizing C^{++} was preferred to java, Jade and other frameworks because of its higher ability in interfacing with hardware dynamics, faster memory intelligent in systems with online monitoring and more compatibility with real time simulation which are considerable factors in control systems.

8. Conclusion

In this paper, we illustrate how multi-agent architecture can be applied for different control applications and different toolkits for multi-agent programming. In the architecture, the whole control problem is distributed in partial control problem, each partial control problem can be solved using optimal agent controller. We get the overall solution using integration of all agent solution. This method is too much useful for complex control problem where the problem can not be solved using single optimum control algorithm for whole system.

Appendix A: Comparison of different methods & application

Title	Application	Agent Distribution	Agent Technique	Software	Results
Development of agent-based autonomous robotic wheelchair control systems	Wheelchair (Biomedical)	Wall-following, goal-seeking, and obstacle avoidance	Fuzzy logic	Microsoft Visual C++ [7] and SQL server to develop the application software.	In this paper, a robotic wheelchair controller is developed based on the multi-agent control architecture and fuzzy algorithms.
An Agent-based Controller for Vehicular Automation	Vehicular Automation (Traffic Control)	Default agent and active agent	Default agent- Fuzzy logic Active agent- Neural Network	Not shown	In this approach, the driving task is conducted by a group of control agent each of which focuses on only a few specific operating conditions.
An Agent-based Framework for Distributed Intelligent Control Systems	Digester process (Chemical)	Organization Level Agent, Coordination Level Agent, Execution Level Agent	Fuzzy Logic, Expert system, Neural Network	C++ and CORBA technique	A case study shows that the coordination optimal strategy of MAS is successful and the hybrid optimization algorithm approach combined intelligence and multi-objective optimization can get better Optimal results.
An intelligent control hydropower system based on multi-agent theory	Power Plant	Management agent, Communication agent, Function agent	Optimal algorithm	Not Shown	Combination multi-agent theory and optimization to create a more economical hydropower station operation control system will greatly improve efficiency and reduce the operating costs.
Design and implementation of a room thermostat using an agent-based approach	Temperature control (Process Control)	Tracking Agent, Regulator agent, Bang-bang up agent, Bang-bang down agent,	Neural Network	C++ software library	In conclusion, the agent-based approach has shown to support the divide and conquer approach and gives a designer a structured method to deal with complex Control problems.
Automatic control system of water conservancy project model based on multi-agent	Water Conservancy Project (Process Engineering)	System Agent, Flow control agent, Water control Agent, Velocity control agent	Different Control algorithms	Not shown	Through the experimental data, we can find that the control precision and stability of flow and water level meet the request of water conservancy project model test.

REFERENCE

Bart De Schutter, and Hans Hellendoorn, Rudy R. Negenborn, "Multi-Agent Model Predictive Control of Transportation Networks", IEEE | | Casti, J. 1997 "Would-be worlds: how simulation is changing the world of science", New York: Wiley. | | Charles M. Macal, Michael J. North, 2006, "tutorial on agent based modeling and Simulation part 2: how to model with Agents?" Winter Simulation conference, IEEE | | Chunhua Zhao, Mani Bhushan, Venkat venkatasubramanian, "A multi-agent architecture for automated process safety analysis" | Hupkens, Thaens, 2004 "A Multi Agent System for MS Windows Using Matlab-enabled Agents" 16th Belgian-Dutch Conference on Artificial Intelligence (BNAIC) | | Jennings, N. R.2000 "On agent-based software engineering". Artificial Intelligence, 117:277-296. | | Jeinnings & Wooldridge, 1995, "April: Agent Process Interaction Language", Intelligent Agents. | | Lei Ming, Yang, 1998, "CORBA-based agent-driven design for distributed intelligent manufacturing systems" Journal of Intelligent Manufacturing, Vol 9 | | Li Zhang, Weiren Shi, Jianqiu Cao, Yinhua Linz 2004 "An Agent-based Framework for Distributed Intelligent Control Systems", SICE Annual Conference in Sapporo. | | Mossinger P., Polani D., Spall R., and T, 1995, "Xraptor-A synthetic Multi-Agent Environment for Evaluation of Adaptive Control Mechanisms", Proc. of the EUROSIM Conference. | | Negenborn R. R., De Schutter, J. Hellendoorn 2004, "Multi-Agent Model Predictive Control: A Survey", Technical report 04-010, Delft Center for Systems and Control. | | Nikolaos S., Moraitis P., 2006, "The Gaia2JADE Process for Multi-Agent Systems Development", Applied Artificial Intelligence Journal, Taylor & Francis, Vol. 20, No. 2-4, pp. 251-273. | | Ruth Aylett, Frances Brazier, Nick Jennings 1997, "agent a systems and applications" Second UK workshop on foundations of Multi-agent systems, FoMAS. | | Shoham Y., 1991 "Agent0: A Simple Agent Language and its interpreter", In Ninth National Conference on Artificial intelligence. | |