Modeling and Control of Distributed Processes on the Base of Multi-Agent Systems

Jolana Sebestyénová

Institute of Informatics, Slovak Academy of Sciences
Dúbravská cesta 9, 845 07 Bratislava, Slovakia
sebestyenova@savba.sk

Abstract: The paper describes decision support system for modeling, control, and simulation of continuous, as well as discrete event systems. Models, control methods, and tools are in database specified by their attributes. Each attribute's weight is initially estimated according to importance and classification power of a given feature. Automatic learning of attributes weights uses the answers of the users after simulation provided by the system to increase the quality of case-based reasoning. The proposed learning algorithm guarantees convergence of the attributes weights to relatively steady values yielding to CBR of best quality. If simulation of a new case was successful from the point of view of the user, the new case is added to case base.

Keywords: Decision support system, Case-based reasoning, Case-based learning, Learning of weights

1 Introduction

Decision support system is an interactive computer-based system intended to help decision makers use communications technologies, data, documents, knowledge and/or models to identify and solve problems, complete decision process tasks, and make decisions.

The concept of decision automation is deceptively simple and intriguingly complex. From a narrow perspective, a decision is a choice among defined alternative courses of action. From a broader perspective, a decision involves the complete process of gathering and evaluating information about a situation, identifying a need for a decision, identifying or in other ways defining relevant alternative courses of action, choosing the 'best', the 'most appropriate' or the 'optimum' action, and then applying the solution and choice in the situation. Automation refers to using technologies including computer processing to make decisions and implement programmed decision processes. Typically decision automation is considered most appropriate for well-structured, clearly defined, routine or programmed decision situations.

A decision support system [7] can be approached from two major disciplinary perspectives, those of information systems science and artificial intelligence. We present in this paper an extended ontology for a decision support system in control theory domain. The ontology explicates relevant constructs and presents a vocabulary for a decision support system, and emphasizes the need to cover environmental and contextual variables as an integral part of decision support system development and evaluation methodologies. These results help the system developers to take the system's context into account through the set of defined variables that are linked to the application domain. With these extensions the focus in decision support systems development shifts from task ontology towards domain ontology.

Most AI systems operate on a first-principles basis, using rules or axioms plus logical inference to do their work [5]. Those few reasoning systems that include analogy tend to treat it as a method of last resort, something to use only when other forms of inference have failed. The exceptions are case-based reasoning systems, which started out to provide computational mechanisms similar to those that people seem to use to solve everyday problems. Unfortunately, case-based reasoning systems generally have the opposite problem, tending to use only minimal first-principles reasoning.

Decision support system for modeling and control of continuous as well as discrete event systems developed in MARABU project [3] is used to illustrate reasoning. The database of the support system contains methods and tools for modeling of the systems, control synthesis, and simulation. Further, the database contains complete models of some systems.

2 Agent-based Decision Support System

In knowledge engineering, agents offer the flexibility to integrate many different categories of processing within a single system. Agent definitions range from descriptions based on a functional analysis of how agents are used in technology to far more ranging expositions based on different interpretations of the role and objectives of artificial intelligence and cognitive science. Artificial intelligence is a very diverse field and agents are used as metaphors for work in many areas.

Multi-agent systems are appropriate for domains that are naturally distributed and require automated reasoning [2]. Agents should perform the following capabilities to some degree:

- Planning or reacting to achieve goals,

- Modeling the environment to properly react to situations,
- Sensing and acting,
- Inter-agent coordination,
- Conflict resolution (coordination is a continuous process, conflict resolution is event-driven, triggered by conflict detection).

To design a multi-agent system for a given problem, the designer has to understand how should agent and AI techniques be applied to the domain, what competencies agents need, and which techniques implement those competencies. Thus, multi-agent system design consists of (1) dividing resources and domain responsibilities among agents, (2) determining which core competencies satisfy which domain responsibilities, and (3) selecting techniques to satisfy each core competency. According to distributed domain-specific responsibilities agent-based systems may be heterogeneous, with each agent responsible for a different set of goals or homogeneous, where agents share the same goals. Agents in the proposed system work according to simple workflow that is specified by user in terms of required support.

Decision support systems are used by people who are skilled in their jobs and who need to be supported rather than replaced by a computer system. The broadest definition states that decision support system is an interactive computer-based system or subsystem intended to help decision makers use communications technologies, data, documents, knowledge and/or models to identify and solve problems, complete decision process tasks, and make decisions. Five specific decision support system types include [7]:

- Communications-driven DSS,
- Data-driven DSS,
- Document-driven DSS,
- Knowledge-driven DSS,
- Model-driven DSS.

2.1 Knowledge Representation

There are two kinds of conceptual knowledge: concept and set. A concept is defined by the essence of the objects it subsumes and not by their state. Such a definition allows us to focus on the essence of the concepts and not on their state. An essence is invariant, which is not the case of state. On the other hand, a set makes it possible to put together objects whose state shares some common properties. For instance, if ‘Human Being’ refers to a concept, ‘Teenagers’ refers to a set composed of human beings whose age is in given constraints.

Differences are elementary units from which the meaning of terms is built. This means they have no meaning in themselves. A difference belongs to the essence of objects. Unlike an attribute it cannot be removed from the definition of an object without changing its nature; nor can it be valued. For example, for human beings ‘mortal’ is a difference whereas ‘age’ is an attribute. A difference is a unit that builds meanings and divides concepts. Classification of concepts used in such a large area is not a trivial task [8].

In database design, it is important to properly arrange and index the attributes to achieve effective reasoning. The proposed database consists of three parts: DB of methods and tools that are available, case-base of concrete examples, and knowledge base of control theory domain. They are arranged in 37 tables of relational database. For any model, we need to know which system is the model of, what modeling method is used, which tool was used to create the model, and what modeling requirements were given. Similar specifications are used for control, too.

A straight comparison between a DB and ontology needs to take the nature of the data into account. The advent of object orientated databases, improved logics and faster inference is making the distinction between DBs and ontologies more fuzzy [4]. For a domain of control theory, following terms have to be defined in the ontology: system, model, control, system description method, control method, and tool. Each term is characterized by attributes. Concept hierarchy plus attributes gives ontology. Relations that are used in the used ontology are: part of, attribute of, value of, is a (subclass of), instance of.

Knowledge base of described agent-based decision support system contains ontology-based representation of data relevant to control theory domain and further data needed for system functionality [8]:

- Persistent data stored in database,
- Temporary information stored in system variables, e.g. type of required support is stored invariables: *model, control, simulation, similar_cases.*

2.2 Web Portal of Decision Support System

The web-based portals prove to be very suitable for knowledge management. Knowledge portals are flexible and easy to use and may provide almost any kind of content or functionality. To structure the architecture of a knowledge portal, the following three-layer model is being used: (1) user interface and navigation, (2) functions (personalization, active process support, coordination of agents, document management), and (3) knowledge base. Knowledge portals provide a flexible knowledge environment to a potentially large number of users. The mission of a knowledge portal is not only to provide a library-like pool of information, but to actively support the user in his or her decision processes.

Agent-based decision support system MARABU [9] consists of: (1) database of modeling and control methods and tools, (2) case base of models created and simulated in past, (3) knowledge base of the control theory domain, (4) web-portal enabling to specify user requirements, to display results of reasoning, and to connect a provider of a selected tool. Web portal of proposed support system provides basic information about the system, besides obvious functions, such as registration and login of authorized users. Web portal further enables authorized users to:

Specification of a required support: User can choose any combination from four given options: creation of a model of a specified system, control synthesis, simulation, or list of accomplished similar cases.

Basic system characteristics: User has to choose one of the three basic system types: continuous systems, discrete event systems, distributed systems. Basic system characteristic determines content and sequence of questionnaires for specification of attributes and requirements, as well as context of reasoning.

Questionnaires: Actual workflow of the system is determined by requirements fulfilled in the questionnaires. User needs to specify only required values of those attributes that are important from his point of view. All possible values of not specified attributes are taken by reasoning algorithm as applicable. Usage of modeling and control methods and tools provided by the decision support system is described in help files.

The proposed support system enables devices to be virtually shared, managed, and accessed across a consortium or workgroup. Although the physical resources may reside in multiple locations, users have seamless and uninterrupted access to these resources.

3 Reasoning

There are two main directions in DB reasoning: forward chaining and backward chaining algorithms. Forward chaining is an example of the general concept of data-driven reasoning - that is, reasoning in which the focus of attention starts with the known data. It can be used within an agent to derive conclusions from incoming percepts, often without a specific query in mind. New facts can be added to the agenda to initiate new inferences.

The backward-chaining algorithm, as its name suggests, works backwards from the query. If the query q is known to be true, then no work is needed. Otherwise, the algorithm finds those implications in the knowledge base that conclude q. Backward chaining is a form of goal-directed reasoning. It is useful for answering specific questions such as ‘What shall I do now?’ Often, the cost of backward chaining is much less than linear in the size of the knowledge base, because the process touches only relevant facts. In MAS, an agent should share the work between forward and backward reasoning, limiting forward reasoning to the generation of facts that are likely to be relevant to queries that will be solved by backward chaining.

One of the bottlenecks in the creation of AI systems is the difficulty of creating large knowledge bases. There have been a number of systems that capture some aspects of reasoning by analogy. No previous analogy systems have been successfully used with multiple, large general-purpose knowledge bases created by other research groups. While the majority of today's CBR systems have moved to feature-vector representations, there are a number of systems that still use relational information.

3.1 Case-based Reasoning

There is mounting psychological evidence that human cognition centrally involves similarity computations over structured representations, in tasks ranging from high-level visual perception to problem solving, learning, and conceptual change. Understanding how to integrate analogical processing into AI systems seems crucial to creating more human-like reasoning systems [6]. Yet similarity plays at best a minor role in many AI systems. Most AI systems operate on a first-principles basis, using rules or axioms plus logical inference to do their work. Those few reasoning systems that include analogy tend to treat it as a method of last resort, something to use only when other forms of inference have failed.

The exceptions are case-based reasoning systems, which started out to provide computational mechanisms similar to those that people seem to use to solve everyday problems. Unfortunately, CBR systems generally have the opposite problem, tending to use only minimal first-principles reasoning. Moreover, most of today's CBR systems also tend to rely on feature-based descriptions that cannot match the expressive power of predicate calculus. Those relatively few CBR systems that rely on more expressive representations tend to use domain-specific and task-specific similarity metrics. This can be fine for a specific application, but being able to exploit similarity computations that are more like what people do could make such systems even more useful, since they will be more understandable to their human partners. While many useful application systems can be built with purely first-principles reasoning [5] and with today's CBR technologies, integrating analogical processing with first-principles reasoning will bring us closer to the flexibility and power of human reasoning.

CBR enables to use in computer program such kind of problem solving that is usually used by people, i.e. a new task is solved by adapting previously accomplished solution.

In most case-based reasoning systems, cases are stored as named collections of facts in a memory. They are designed for a specific range of problems. Each case is a set of features, or attribute-value pairs, that encode the context in which the ambiguity was encountered. The case retrieval algorithm is mostly a simple k-nearest neighbors algorithm. The basic case-based learning algorithm performs poorly when cases contain many irrelevant attributes. Unfortunately, deciding which features are important for a particular learning task is difficult.

At the highest level of generality, a general CBR cycle may be described [1] by the following four processes:

- **Retrieve** the most similar case or cases,
- **Reuse** the information and knowledge in that case to solve the problem,
- **Revise** the proposed solution,
- **Retain** the parts of this experience to be useful for future problem solving.

3.2 Reasoning in Decision Support System

A described agent-based decision support system MARABU [9] consists of: (1) database of modeling and control methods and tools, (2) case base of models created and simulated in past, (3) knowledge base of the control theory domain, (4) web-portal enabling to specify user requirements, to display results of reasoning, and to connect a provider of a selected tool. In the system, there are three possibilities of reasoning:

- Classical database querying: Models, control methods and tools are searched in database according to user specified requirements and given context.
- Case-based reasoning: Whenever no model or control method matches exactly the user requirements, the model forms and control methods are reasoned from similar cases.
- List of similar cases to the user specifications and requirements: If the user requires support in a form of accomplished similar case, similar cases with references to a tool where simulation can be done are provided (useful for e-learning purposes).

Attributes used in questionnaires to systems specification are weighted by real number 0 - 1. Each attribute's weight is initially estimated according to importance and classification power of a given feature.

Case-based reasoning algorithm proposed for system MARABU works in following steps:

1) An accomplished (history) case from case-base is searched that am best fulfils user specifications and requirements. Similarity rate of the history cases to the actual user specified task is counted using attributes weights.

2) Model and/or control method used in the history case with maximal similarity rate are offered to user to solution of actually specified task.

3) After reasoning, the user agent sends information to the selected tool provider's agent. Using the tool, simulation of a created model can be provided to the user. After simulation, the user can answer to the decision support system by filling in a form, whether his/her requirements were fulfilled.

4a) Automatic learning of weights uses this answer to increase the quality of case-based reasoning. The proposed learning algorithm guarantees convergence of the attributes weights to relatively steady values yielding to CBR of best quality.

4b) Some of new cases after a validation step are added to the case-base.

4 Learning Algorithm

Important feature of case-based reasoning is its coupling to learning [10]. The notion of case-based reasoning does not only denote a particular reasoning method, it also denotes a machine learning paradigm that enables sustained learning by updating the case base after a problem has been solved. Learning in CBR occurs as a natural by-product of problem solving. When a problem is successfully solved, the experience is retained in order to solve similar problems in the future.

Case retainment is the process of incorporating what is useful to retain from the new problem-solving episode into the existing knowledge. The learning from success or failure of the proposed solution is triggered by the outcome of the evaluation and possible repair. It involves selecting which information from the case to retain, in what form to retain it, how to index the case for later retrieval from similar problems, and how to integrate the new case in the memory structure. The tuning of existing indexes is an important part of CBR learning. Index strengths or importance for a particular case or solution are adjusted due to the success or failure of using the case to solve the input problem. For features that have been judged relevant for retrieving a successful case, the association with the case is strengthened, while it is weakened for features that lead to unsuccessful cases being retrieved. The weights are updated, based on feedback of success or failure.

One issue in similarity assessment [6] is how to determine the right features to compare. Decisions about which features are important are often based on explanations of feature relevance, but those explanations may be imperfect, leading to a need for robust similarity metrics that take the difficulties in specifying important features into account. Defining adequate similarity measures is one of the most difficult tasks when developing CBR applications. Unfortunately, only a limited number of techniques for supporting this task by using machine learning techniques have been developed up to now.

In described decision support system, attributes used to systems specification are weighted by real number 0 - 1 (i.e. 0% - 100%). Each attribute's weight is initially estimated according to importance and classification power of a given feature.

After reasoning, the user agent sends information to the selected tool provider's agent. Using the tool, simulation of a created model can be provided to the user. After simulation, the user can answer to the decision support system by filling in a form, whether his/her requirements were fulfilled or not (i.e. reasoning was successful or it failed).

Automatic learning of weights uses the answers of the users to increase the quality of case-based reasoning. The proposed learning algorithm guarantees convergence of the attributes weights to relatively steady values yielding to CBR of best quality. The attributes weights are modified in following way:

$$w_i = \begin{cases} w_i + K_{1,j} \text{ or } \quad w_{i,max} & \text{if success and } u_i == a_i \\ w_i - K_{2,j} \text{ or } \quad w_{i,min} & \text{if fail and } u_i == a_i \\ & \\ w_i - K_{1,j} \text{ or } \quad w_{i,min} & \text{if success and } u_i \neq a_i \\ w_i + K_{2,j} \text{ or } \quad w_{i,max} & \text{if fail and } u_i \neq a_i \end{cases}$$

where $i = 1, \dots, n$

n … number of attributes used to specification of the system and user requirements

w_i … weight of i-th attribute, minimal and maximal values $w_{i,min} = 0.$, $w_{i,max} = 1.$

u_i … valuation of i-th attribute in user's specification of the system and user requirements

a_i … valuation of corresponding i-th attribute in a given case

learning cycles $j = 0, \dots, \infty$

initial and minimal values of coefficients $K_{1,0} = 0.2$, $K_{2,0} = 0.1$, $K_{1,min} = 0.01$, $K_{2,min} = 0.01$

$$K_{1,j+1} = \begin{cases} K_{1,j} * c_1 \\ \\ K_{1,min} \end{cases} \qquad K_{2,j+1} = \begin{cases} K_{2,j} * c_2 \\ \\ K_{2,min} \end{cases}$$

In described decision support system following values were used: $c_1 = c_2 = 0.9$

If simulation of a new case was successful from the point of view of the user, the new case with the attributes stored in database (session information) is added to case base of the system before deletion of given session information.

Conclusions

Decision support system for control theory domain has been described. The database of the system contains methods and tools for modeling and control synthesis as well as a set of complete models of systems specified by their attributes. Each attribute's weight is initially estimated according to importance and classification power of a given feature. Automatic learning of attributes weights uses the answers of the users to increase the quality of case-based reasoning. The proposed learning algorithm guarantees convergence of the attributes weights to relatively steady values yielding to CBR of best quality.

Acknowledgement

The author is grateful to the Slovak Grant Agency for Science (grant No. 2/4148/25) for partial support of this work.

References

[1] Aamodt A., E. Plaza: Case-based Reasoning: Foundational Issues, Methodological Variations, and System Approaches, in Artificial Intelligence Communications, 1994, IOS Press, Vol. 7: 1, pp. 39-59, http://www.iiia.csic.es/

[2] Davis D. N.: Synthetic Agents: Synthetic Minds? Frontiers of Cognitive Agents, in IEEE Symposium on Systems, Man and Cybernetics, San Diego, 1998, IEEE Press

[3] Frankovič B., I. Budinská, J. Sebestyénová, Dang T. Tung, V. Oravec: MARABU - Multiagentový podporný systém pre modelovanie, riadenie a simuláciu dynamických systémov, in AT&P Journal 4, 2005, ISSN 1335-2237, pp. 57-59, in Slovak

[4] Heijst G.: The Role of Ontologies in Knowledge Engineering, PhD. Thesis, University of Amsterdam, Netherlands, 1995

[5] Kenneth D. Forbus, T. Mostek, R. Ferguson: An Analogy Ontology for Integrating Analogical Processing and First-Principles Reasoning, American Association for Artificial Intelligence, 2002, www.aaai.org

[6] Leake D. B.: Case-based Reasoning: Experiences, Lessons, and Future Directions, AAAI Press/MIT Press, 1996, www.cs.indiana.edu/~leake/papers/a-96-book.html

[7] Power D. J.: Decision Support Systems Hyperbook. Cedar Falls, IA: DSSResources.COM, HTML version, Fall 2000, http://dssresources.com/dssbook/

[8] Sebestyénová J.: Ontology-based Databases for Decision Support Systems, in WSEAS Trans. on Information Science and Applications, Issue 12, Vol. 2, December 2005, ISSN: 1790-0832, pp. 2184-2191

[9] Sebestyénová J.: Decision Support System for Modelling of Systems and Control Systems Design, in Proc. IEEE Int. Conf. on Computational Intelligence for Modelling Control and Automation CIMCA'2005, Ed. M. Mohammadian, Vol. 1, Nov. 28-30, 2005, Vienna, ISBN-13: 978-0-7695-2504-4, ISBN-10: 0-7695-2504-0, pp. 70-75

[10] Wettschereck D., D. W. Aha: Weighting Features. Proc. of the First Int. Conference on Case-based Reasoning, 1995, pp. 347-358, http://citeseer.ifi.unizh.ch/article/wettschereck95weighting.html