Proceedings of
National Conference on Innovative Trends in Computer Science Engineering (ITCSE-2015)
held at BRCMCET , Bahal on 4th April 2015

Decision making process using MAS for Case based Reasoning

[1]Keshav Jindal , [2]Dr. S. Srinivasan
[1]Research Scholar, [2]Professor
[1]Suresh Gyan Vihar University Jaipur,
[2]PDM Engineering College , Bhadurgarh, Haryana,
[1]Jindal.keshav43@gmail.com , [2]dss_dce@yahoo.com

Abstract :This paper presents a structure for culture in a dispersed information situation with decentralized decision making. We have based our structure in multi agent System (MAS) in order to contain decentralized decision making, and in Case based Reasoning (CBR), since the lethargic knowledge nature of CBR is appropriate for lively multi agent Systems. Furthermore, we are concerned in independent agents so as to collaboratively work as ensembles. An ensemble of agents solves struggle in the subsequent technique: each agent solves the predicament at offer separately and makes its entity forecast, and followed by all those predictions are aggregated to beginning a inclusive prophecy. Consequently, in this work we are concerned in embryonic all together Case based reasoning strategies for multi agent System. Specifically, we will present the multi agent Case based Reasoning framework, a multi agent approach to CBR. Each individual agent in a multi agent Case based Reasoning system is capable of separately learns and solves problems using CBR with an individual case base. In addition each case base is owned and managed by an entity agent and some information is disclosed otherwise mutual simply if the agent decides so. Therefore, this structure preserves the isolation of data, and the independence to make known data.

Keywords: MAS, MH, CBR, AI

1. INTRODUCTION

Recently, the interest in decentralized approaches for the resolution of complex real world problems, like Scheduling, is gaining much attention due to its wide applications. Several of these approaches belong to Distributed Systems research area, where a number of entities work together to solve problems in a cooperative way. In this area, it is possible to emphasize Multi-Agent Systems (MAS), concerning the coordination of agent's behaviours in order to share knowledge, abilities, and objectives, in the resolution of complex problems. Due to the exponential growing of system's complexity, it is important that MAS become more autonomous to deal with dynamism,
overloads and failures recovery. Multi-Agent Systems typically operate in open, complex, dynamic, and unpredictable environments. Therefore, learning becomes crucial. Learning is a relevant area from Artificial Intelligence (AI) as from human intelligence. Plaza et al. [3] defined learning as "the process of improving individual performance, precision (or quality) of solutions, efficiency of finding solutions and scope of solvable problems". Although this definition is very useful, it is a severe and constricted view of learning. In a more general way, it is possible to define learning as the acquisition of new knowledge or updating existing knowledge. As per Alonso et al. [1], intelligence implies a certain degree of autonomy, which requires the capacity of taking decisions autonomously. Thus, agents should have the appropriate tools to take such decisions. In dynamic domains it is not possible to predict every situation that an agent can find, so it is necessary that agents have the ability to adapt to new situations. This is especially true in MAS, where in many cases the global behaviour emerges instead of being pre-defined. Consequently, learning is a crucial component of autonomy and pro-activeness, which must be a study target of agents and MAS [1].

The adaptation of ideas from different research areas, inspired from nature, led to the development of Meta-Heuristics (MH), which are techniques aiming to solve complex generic problems of combinatorial optimization, in which the scheduling problem is included. Meta-heuristics are very useful to achieve good solutions in reasonable execution times. Sometimes they even obtain optimal solutions. However, to achieve near optimal solutions, it is required the appropriate tuning of parameters. Parameter tuning of MH has a great influence in the effectiveness and efficiency of the search process. The definition of the parameters is not obvious because they depend on the problem and the time that the user has to solve the problem [2]. Therefore, this paper proposes the use of a learning mechanism in order to perform the MH parameter tuning in the resolution of the scheduling problem. Case-based Reasoning (CBR) was chosen since it assumes that similar problems may require similar solutions. As a MAS is used to model a dynamic scheduling system, with agents representing both tasks and resources, it is proposed that each resource agent have their own CBR module, allowing a multi-apprentice learning. With this type of learning, agents learn how to perform their own single-machine scheduling problem.

II.CASE BASED REASONING

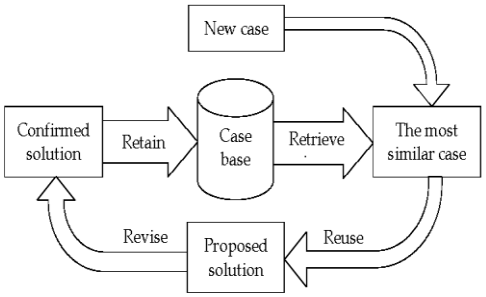

Fig 1: The Case Based Reasoning Cycle

Finally, in the Retain stage, the system decides whether to incorporate the new solved case into the case base or not.

Specifically, in this section we are going to focus in the Revise and Retain stages. For the Revise stage, we are interested on techniques that allow a CBR agent to build an explanation (to be presented to an human expert or to another CBR agent) that endorses the solution of the new case. For the Retain stage, we are interested in case retention strategies that permit a CBR agent to select which cases to store in its local case base in order to obtain a compact competent case base (i.e. a small case base that allows the agent to solve the largest range of problems possible). Related to case retention is the machine learning subfield of active learning. Active learning focuses on learning systems that have some kind of control over which examples to learn from.

III.MULTI-AGENT LEARNING

In AI, machine learning is a research area concerning the development of algorithms and techniques in order to provide computers with learning faculties. Commonly accepted in the literature, machine learning algorithms and techniques can be classified in three categories:

• Supervised learning (where data have labels or classes);
• Unsupervised learning (data have no labels);
• Reinforcement learning (where the objective is to maximize a reward).

Some authors refer another category, placed between Supervised and Unsupervised learning, named Semi-Supervised learning, that uses both labelled and not labelled data. It is also very common the reference to another category, known by Instance based learning [6] or Non-Parametric Methods [7], where CBR can be included, described in the next section.

It is possible to apply machine learning concepts to many research areas, including natural language processing, pattern recognition, market analysis, DNA sequences classification, speech and handwriting recognition, object recognition in computer vision, game playing and robot locomotion. Panait and Luke [8] have focused machine learning application to problems related with MAS. They use machine learning in order to explore ways to automate the inductive process, e.g., put a machine agent to find by itself how to solve a task or minimize error. They have referred that machine learning is a popular approach for the resolution of MAS problems because the complexity intrinsic to many of those problems can make solutions prohibitively hard to obtain.

In the next subsections, it will be described four learning techniques used in MAS, namely Reactive learning, Social learning, Team learning and Concurrent learning.

A) Reactive learning

In reactive systems, the cooperative behaviour emerges from the interaction between agents. Instead of implementing coordination protocols or providing complex recognition models, it is assumed that agents work with value-based information (e.g. the distance they should keep from neighbours) which produces the social behaviour. Once internal processing is avoided, these techniques allow MAS reacting to changes in an efficient way [1].

As a collateral effect, agents do not know the domain, which is crucial to take decisions in complex and dynamic scenarios. In this view, it is not possible to simulate complex social interactions and, in order to have high-level behaviours agents need to summarize experiences in concepts. An entity that can conceptualize can also transform experience in knowledge and guide the vital resources until necessary [1].

B)Social Learning

Social learning is composed by learning mechanisms arising from AI and Biology. In persistent MAS, where new agents enter a world already populated with experienced agents, a new agent starts with a blank state and has not had yet the opportunity to learn about the environment. However, a new agent does not need to discover everything about the environment since it can benefit from the accumulated learning from the experienced population of agents [1].

An important difference between artificial agents and animals is that, in the first, it is possible to simulate a completely cooperative scenario, where exists a common utility function. Even though cooperation occurs in many animal species, the possibility of conflicts emerging is always present, due to the competition in genes' self replication of evolutionary process [1].

There are several different ways to an agent learn from other agents behaviors. Despite the existence of imitation (direct copy from other agents behaviors), this has proved to be complex since it involves not only the behaviors' understanding and reproducing but also the understanding of the changes in the environment caused by these behaviors [1].

C)Team Learning

In Team Learning it only exists an apprentice. However, it has the objective to discover a subset of behaviors for a team of agents, instead for a unique agent. It is a simple approach to Multi-Agent learning because the apprentice can use machine learning techniques, which avoid the difficulties emerging from the co-adaptation of multiple agents in Concurrent Learning approaches. Another advantage in the using of a unique apprentice agent is that it only cares about the team performance, and not with itself. For this reason, Team learning approaches can ignore the inter-agent credit assignment that is usually hard to determine [8].

However, Panait and Luke [8] also pointed some disadvantages in the use of Team learning. The main problem refers to the large state space for the learning process, which can be devastating for learning methods that explore the utility state space (such as Reinforcement learning) but cannot affect so drastically techniques that explore the behaviours space (such as Evolutionary computing). A second disadvantage refers to the learning algorithm centralization problem: every resource need to be available in the same place where the program will be executed. This can be uncomfortable for domains where data are inherently distributed. Team learning can be divided in homogeneous and heterogeneous. Homogeneous apprentices develop an unique identical behaviour for each agent, even if agents are different. Heterogeneous apprentices must deal with a large search space, but with the guarantee to get better solutions

through agents' specialization. The choice between approaches depends if experts are necessary in the team.

D)Concurrent Learning

The most common alternative to Team learning is Concurrent learning, where multiple apprentices try to improve parts from the team. Typically, each agent has its own learning process to modify the behaviors [8]. The main difficulty subjacent to Concurrent learning is to know in which domains it achieves better results when compared with Team learning. Jansen and Wiegand [9] argue that Concurrent learning can perform better in domains where decomposition is possible and helpful (such as Scheduling), and when it is useful to focus each sub-problem regardless others. This happens because Concurrent learning separates the search space into smaller ones. If the problem can be decomposed, such that agents' individual behaviors are relatively disjoint, it can result in a significant reduction of the search space and computational complexity. Another advantage is that decomposing the learning process into smaller pieces allows a greater flexibility using computational resources in each process learning, since they can, at least partially, be learned regardless others. The main challenge of Concurrent learning consists in the adaption of each apprentice behaviours to the context of others, which its cannot control. In single agent scenarios, an apprentice explores his environment and improves his behaviour. But things are quite different when using multiple apprentices: while agents learn, they change the behaviours, which can ruin the learned behaviours by other agents, making outdated assumptions. A simple approach to deal with this co-adaptation

is to treat other apprentices as part of the dynamic environment for which each apprentice must adapt [11].
In this research, we propose a concurrent learning approach, in which several agents learn about their internal behaviours and environment.

IV.MULTI-AGENT SCHEDULING SYSTEM

The developed MAS for the resolution of scheduling problem consist in a hybrid autonomous architecture [12]. As illustrated in Fig. 2, there are three kinds of agents. The proposed MAS have agents representing jobs/tasks and agents representing resources/ machines. The system is able to find optimal or near optimal solutions through the use of MH, dealing with dynamism (arriving of new jobs, cancelled jobs, changing jobs attributes, etc.), change/adapt the parameters of the algorithm according to the current situation, switch from one MH to another, and perform a

coordination between agents through cooperation or negotiation mechanisms. Job agents process the necessary information about the respective job. They are responsible for the generation of the earliest and latest processing times on the respective job and automatically separate each job's operation for the respective Resource Agent.
Resource agents are responsible for scheduling the operations that require processing in the machine supervised by the agent. These agents implement MH in order to find the best possible single-machine schedules/plans of operations and communicate those solutions to the Agent user interface for later feasibility check. Since it is impossible to predict each problem to treat, the system should be capable of learning about its experience during lifetime, as humans do. To perform this learning mechanism, it is proposed the use of CBR within Resource agents.

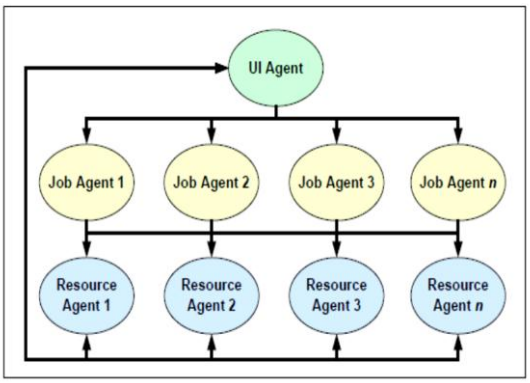

Fig. 2. Multi-agent Scheduling System

V.PROPOSED SYSTEM

The proposed CBR approach consists in retrieving the most similar case or cases to the new problem, regardless the Meta-Heuristics to be used, as well as its parameters. It is important for the system to decide which technique and respective parameters may be used, because not every Meta-Heuristics is suitable to all types of problems. The main objective of CBR

module is to choose a Meta-Heuristics to be used by the respective Resource Agent in which the CBR is included. The secondary objective is to perform the parameter tuning of Meta-Heuristics, according to the problem to solve. Based on past experience, each case contains the Meta-Heuristics and the respective parameters. If the parameters were effective and efficient in the resolution of a similar case, then they have a great probability to be effective and efficient in the resolution of the new problem. It is possible to describe our CBR module as a hyper-heuristic approach but since it performs a self-parameterization of Meta-Heuristics it is more appropriate to see it as a parameter tuning approach.
It is important to notice that, like previously described in Fig. 1, every new problem or perturbations occurred leads to a new case in the system, with the previous most similar cases being retrieved from the casebase. After that, the better case is reused, becoming a suggested solution. After the solution revision, the case is executed in the MAS. This revision is performed to allow escaping from local optimal solutions and Meta-Heuristics stagnation, since it is used some disturbance in the parameters of the proposed solution. After the conclusion of the MAS execution, the case is confirmed as a good solution, being retained on the database as a new learned case, for future use.
Fig. 3 illustrates the inclusion of CBR in the system. Each Resource Agent has its own CBR module. With this

approach, different Meta-Heuristics may be chosen in the resolution of the same Job-Shop problem. This can be considered as an advantage because the Resource Agents can have different number of operations to schedule. Some Meta-Heuristics are more suitable to schedule problems with large number of operations than others.

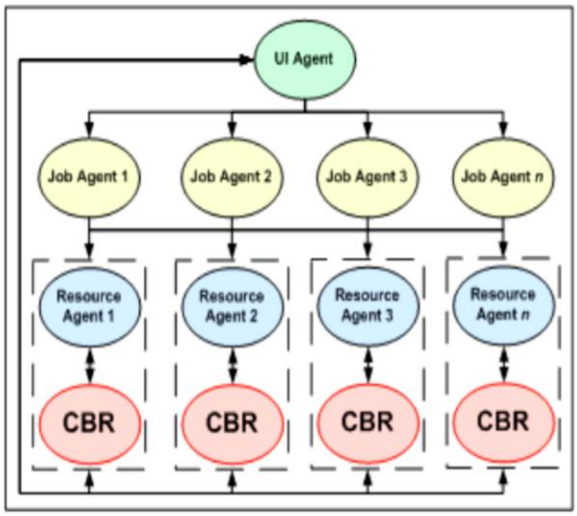

Fig. 3. CBR module within Resource agents

The most important part of a CBR module is its similarity measure because it decides how much two cases are similar between each other. The similarity measure of the proposed CBR module is very simple and is defined in equation (1).

Similarity= Number Operation $Case_A$ /
Number Operation $Case_B$ Equ. 1.

As previously mentioned, each Resource Agent has a number of operations to schedule. This number of operations can be different, depending on the problem to treat, and is enough to define a problem. The MH and the respective parameters may be chosen according to the dimension of the problem to treat. So, with this similarity measure it is possible to have a ratio between two cases. The similarity is a value in the interval [0, 1], who's limits correspond to non similar and completely similar cases, respectively. If there are more than one case very similar to the problem to be solved, the most effective and efficient case is reused.

If some perturbations occur in the problem, the MH and the parameters may change, because a different problem may be solved. For example, if new jobs arrive or if some jobs are cancelled, the problem's dimension is different and so other MH and/or other parameters may be used. This decision is autonomously performed by the CBR module in run time.

VI.CONCLUSION

The main objective of this paper is to analyze the integration of CBR in an effective and efficient way, comparing the system's performance with CBR included versus the system's performance before the integration of CBR. Another objective is to obtain some conclusions about the usage of MH in the resolution of Job-Shop instances, after the integration of CBR. The presented scheduling system consists in MAS with different agents representing both jobs and resources. The proposed CBR module is included in resource agents with the objective to chose the best MH and perform the respective parameter tuning.

REFERENCE

[1] E. Alonso, D. Kudenko, M. Luch, and J. Noble, Learning in Multi-agent Systems, The Knowledge Engineering Review, volume 16 (3), pp.277-84, 2001.
[2] El-Ghazali Talbi, Metaheuristics - From Design to Implementation, Wiley, 2009.
[3] E. Plaza, J. Arcos and F. Martin, Cooperative Case-Based Reasoning, in Weiss, G., ed. Distributed Artificial Intelligence Meets Machine Learning, Lecture Notes in Artificial Intelligence., Springer, 1996.
[4] V. Cerny, "A thermo dynamical approach to the travelling salesman problem: An efficient simulation algorithm", pp.41-51, 1985.
[5] Agnar Aamodt and Enric Plaza. Case-based reasoning: Foundational issues, methodological variations, and system approaches. Artificial Intelligence Communications, 7(1):39–59, 1994.
[6] T. Mitchell, Machine Learning. McGraw-Hill Education, ISE Editions, 1997.
[7] E. Alpaydin, Introduction to Machine Learning, Adaptive Computation and Machine Learning, The MIT Press, 2004.
[8] L. Panait and S. Luke, Cooperative Multi-Agent Learning: The State of the Art, Autonomous Agents and Multi-Agent Systems, pp.387-434, 2005.