Distributed Case-Based Learning

M V Nagendra Prasad

Center for Strategic Technology Research
Andersen Consulting
3773 Willow Road
Northbrook, IL 60062
nagendra@cstar.ac.com

Abstract

Multi-agent systems exploiting case based reasoning techniques have to deal with the problem of retrieving episodes that are themselves distributed across a set of agents. From a Gestalt perspective, a good overall case may not be the one derived from the summation of best subcases. In this paper we deal with issues involved in learning and exploiting the learned knowledge in multi-agent case-based systems.

Introduction

Case Based Reasoning (CBR) has been attracting much attention recently as a paradigm with a wide variety of applications [Kolodner 93]. In this paper, we discuss issues pertaining to cooperative retrieval and composition of a case in which subcases are distributed across different agents in a multi-agent system.

A multi-agent system comprises a group of intelligent agents working towards a set of common global goals or separate individual goals that may interact. In such a system, each of the agents may not be individually capable of achieving the global goal and/or their goals may have interactions - leading to a need for coordination among the set of agents. Due to its partial view of the problem-solving situation, an agent may have access only to a part of the environment, and communication bandwidth limitations and heterogeneity of representations may limit its view of other agents' states. An agent may have to communicate and negotiate with other agents to resolve any uncertainties (arising out of the partial or imperfect views of the global problem-solving context) to the extent that it can make positive contributions to the ongoing problem solving process [Lesser 91]. More specifically, in a distributed case-based reasoning system (DCBR), each agent's partial view may result in the best local cases, which when assembled, may not result in the best overall case in terms of global measures. This gives rise to a need for the agents to cooperatively access their case bases to retrieve and assemble a good composite case [Nagendra Prasad, Lesser and Lander 1996].

We will discuss these concepts using a multi-agent transportation-planning domain. Two countries C_1 and C_2 share a border with a number of check-posts. Agent A_1 transports a load from a source in C_1 to one of the border check-posts and agent A_2 needs to carry the load from that check-post to a destination in C_2. Given a source in country C_1 and a destination in country C_2, the agents need to plan a route -- including the common check-post -- from the source to the destination. Each of the agents can rely on its local case base of paths to plan its route to a check-post. However, the agents need to "talk to each other" to reach an agreement on a common check-post so that one agent can takeover the load from the other agent.

The rest of the paper is organized as follows. Section 2 discusses how distributed case bases arise. Section 3 introduces an efficient retrieval algorithm from distributed case bases. In Section 4, we present some early experimental results and then conclude with a discussion of the implications of the proposed mechanism and future work in the final section.

Distributed Case Bases

In a multi-agent system, a problem-solving episode may not be situated at a single physical location and may be distributed across a set of agents. How do distributed case bases arise in these systems? A system that performs rote learning by storing successful problem-solving episodes, where each agent stores its own local case in its case-base, could give rise to such a Distributed Case Base (DCB). However, this may not be the only way. If the agents are a set of "open" agents[1] [Lander 94], then each of them could have acquired its own independent problem-solving experiences by participating in different teams of agents. A *Case-Knowledge Engineer* could design each of the case bases by giving them episodes from his collection of cases.

[1] An agent is "open" if it is designed to be flexible enough to participate in a number of agent systems.

Another scenario one could envisage in the near future is the existence of case bases spread across a communication network. Certain queries may not be satisfied by any one case base and may need a composite case derived from different case bases.

Barletta and Mark [Barletta and Mark 88] and Redmond [Redmond 90] deal with distributed cases in single agent systems. Each case is divided into subcases or snippets and a snippet is indexed using both global goals of the entire case and the local context of that snippet within the case. This kind of elaborate engineering in the form of indexing the case pieces using both global and local problem solving contexts may not be feasible for multi-agent CBR systems. The agents may only have a partial view of the global problem solving context and the internal context of a case piece. Case bases for individual agents may be built independently, without the knowledge of the kinds of problem solving systems in which they are going to participate. Veloso and Carbonell [Veloso and Carbonell 93] use multiple cases to guide the adaptation process to derive a plan. Each of the cases covers a subset of the goals and provides a plan for achieving those goals. As in the previously discussed work, due to its single-agent nature, the need for dynamically exchanging information to resolve conflicts is not an explicit part of this work. Purvis and Pu [Purvis and Pu 95] use constraint satisfaction techniques to adapt a case whose features are derived from features of more than one past case. The constraints are used to adapt the new case by changing its feature values to satisfy them. Unlike the work presented in this paper, the system is a single agent system. We use distributed constraint optimization techniques to assemble cases from pieces distributed across multiple databases. The constraint optimization in this paper is not for adaptation but for deciding which cases to put together into a coherent overall case; an issue that is unique and important to DCBs.

There are other important issues to note in DCBs. It may in fact be true that even though the cases for individual agents may be derived from past problem-solving experiences, there could be combinations of these cases that may be assembled into an episode that the system as a whole has never seen before. As discussed later, this has some important implications in DCBs. In addition, case integration does not require that the overall episode be completely represented at any one node; in some situations, the distributed episode components are integrated only by their mutual consistency.

Retrieval in Distributed Cases Bases

We now introduce our multi-agent case-based learning system called MILTON. Each agent in MILTON is associated with a local case base from which it contributes a local case to the overall case. The overall case is the composition of a set of compatible local cases, one from each of the agents in the system. Each agent is also associated with a set of constraints representing the requirements of the present problem instance and the contexts in which a local case from that agent can participate in the overall case. A constraint could be defined on local and/or non-local variables representing features derivable from partial cases. A local case is also associated with some credibility or cost measure representing the "goodness" of a case. For example, in the transportation-planning domain, each agent has delivery paths to the check-posts from source or destination as local cases and a constraint that the check-post of a local path has to match with that of the non-local path. The credibilty measure for a local path is its distance. Retrieval from distributed case bases can be cast as a distributed constraint optimization problem where the overall case should be compatible with the local constraints of each of the participant agents and it should be the best such case (in terms of credibility measures). Local case consistency constraints arise from the knowledge that an agent has about the generic requirements of the context in which its local subcase can usefully participate.

Searching for Optimal Cases

Central to the approach to distributed case-based systems in MILTON is the Opportunistic A^* or OA^* search algorithm for retrieving optimal overall cases from distributed case bases. OA^* is a distributed version of A^* search with modifications for merging partial search paths, in addition to expanding them. The proof of optimality of solutions in this search is a straightforward extension of the A^* search algorithm and can be found in [Nagendra Prasad98]. Each agent executes the algorithm shown in Figure 1.

Each agent executes *Do_Local_Retrieval* to retrieve all those local subcases that satisfy the constraints on local features (features available from subcases retrieved locally). From these locally valid subcases, the set of agents needs to assemble an overall case that is globally acceptable i.e., it satisfies all the non-local constraints in each of the agents. In addition, it should be the best such case in terms of credibility. In the transportation-planning domain, the agents have to not only agree on a case with a common check-point, but they have to find one with the minimum distance. Agents perform a distributed A^* search over the space of partial cases to accomplish this. Initially the agents announce their worst credibility local case from the set of locally valid cases. These credibility measures are used for forming the heuristic estimate for the credibility of a partial case. Let the sum of the credibilities of component local cases of a partial case be g. Let the sum of the worst credibilities of subcases from agents that have not yet contributed component subcases to this partial

case be the admissible estimate h'. Then an estimate of the credibility of a partial case f' is:

$$f' = g + h'$$

During a single iteration in the loop of the OA* algorithm, each agent announces its best credibility case. The agents collectively choose the best partial case (in a distributed manner) and try to merge it with each of their partial cases. The merging process at an agent involves checking if the case components from the two partial cases satisfy the constraints at that agent on all the features of the two partial cases. If a constraint is violated the merge does not succeed. The search does not terminate with the first solution found, unlike in A* search. It proceeds until all the partial cases are expanded to the extent where their credibility measures are lower than the best complete case found. This is needed to guarantee the optimality of the assembled case. Note that OA* is completely distributed and performs all these functions in a distributed manner.

```
agent_partial_cases = Do_Local_Retrieval (agent);
Announce_Worst_Credibility_Local_Case(agent);
Receive_Worst_Costs(agent);
best_complete_case = nil;
Loop forever
   best_local_partial_case = Choose_Best_Local_Partial_Case(agent_partial_cases)
   if Credibility(best_complete_case) > Credibility(best_local_partial_case) exit;
   Announce_Partial_Case (best_local_partial_case)
   potential_partial_cases_to_merge = Receive_Partial_Cases(agent)
   push (best_local_partial_case, potential_partial_cases_to_merge)
   next_partial_case_to_merge = Choose_Best_Partial_Case(potential_partial_cases_to_merge)
  if (best_local_partial_case == next_partial_case_to_merge)
    remove_case(best_local_partial_case,agent_partial_cases)
  else
    merged_partial_cases = merge(next_partial_case_to_merge,agent_partial_cases)
    if (complete_cases = Get_Complete_Cases(merged_partial_cases))
       Announce_Complete_Cases (complete_cases)
    agent_partial_cases = append (agent_partial_cases,merged_partial_cases)
    if (null agent_partial_cases) exit;
  endif
End Loop
```

Figure 1: Opportunistic A* Algorithm

Learning

We discuss distributed case-based learning in MILTON in the context of the transportation-planning domain previously introduced. Agents can either use a distributed search-based planner or a distributed case-based planner.

The search-based planning algorithm works as follows:

1. Each agent uses A* to find the best paths from source (for A_1) or destination (for A_2) to each checkpoint.
2. The agents together search for an optimal composition of the paths using the OA* algorithm.

The distributed case-based planner works as follows:

1. Each agent retrieves paths (or past cases) with source (for A_1) or destination (for A_2) that is close to the present source (for A_1) or destination (for A_2).
2. Each agent modifies each local case to connect the gap between the present source/destination with the source/destination of the local case.
3. The agents together search for an optimal composition of the paths using the OA* algorithm.

Learning in this system is straightforward. Every time a planner is invoked, the local cases of the optimal path are stored at the respective agents. A planner is invoked by an agent if it does not find similar past cases.

Experiments

The transportation domain has been implemented in a grid world. For the experiments below, we used a 50 x 100 grid where each of the countries occupied 50 x 50 grid with

some obstacles. There are two check-posts between the countries. The OA* algorithm has been implemented as a general distributed constraint optimization search mechanism and integrated into the transportation domain.

We trained MILTON on 100 transportation-planning instances and ran this system and the search-based system on each of the test instances. Figures 2 and 3 show the results of running the system on 20 test instances. Despite the preliminary nature of our experiments, they do reveal the effectiveness of the DCBR system. Average path length for MILTON is 108.9 units whereas the search-based planner produces paths with an average length of 104.6. MILTON takes 0.48 seconds of CPU time to produce a path on average whereas the search-based planner needs 0.87 seconds. With a marginal sacrifice in the optimality of the planning paths, MILTON can boost the efficiency of finding the paths by as much as 81%. The grids used in these experiments are very small and larger grids can be expected to produce more dramatic results.

Conclusion and Future Work

We see the contributions of this work as many fold.

We identified some of the unique aspects of case-based reasoning in multi-agent systems. Further studies along this line can help make available a vast array of tools developed for dealing with the centralized, monolithic cases to bear on multi-agent systems.

The OA* algorithm itself is independent of how the local paths are generated. Hence, it is possible to integrate local paths from agents using different planning mechanisms to form these local paths. For example, one of the agents can use cases where as the other agent could use A* search.

The OA* algorithm is a general and efficient distributed constraint-optimization search mechanism that can deal with both procedurally and declaratively expressed constraints [Nagendra Prasad 98].

In addition, there are a number of other intuitions about multi-agent learning derived in the process of our work with this system [Nagendra Prasad 98]. These need to be validated by further statistical studies and this is our primary focus for the near future.

References

R. Barletta and W. Mark, 1988, Breaking Cases into Pieces, Proceedings of Case-Based Reasoning Workshop, St. Paul, MN, 12-17.

Janet L. Kolodner, 1993, *Case-Based Reasoning*, Morgan Kaufmann Publishers, San Mateo, CA.

Susan E. Lander, 1994, *Distributed Search in Heterogeneous and Reusable Multi-Agent Systems*, Ph.D. Thesis, University of Massachusetts.

Victor R. Lesser, 1991, A Retrospective View of FA/C Distributed Problem Solving, *IEEE Systems, Man and Cybernetics 21(6)*, 1347 – 1362.

M. V. Nagendra Prasad, Victor R. Lesser & Susan Lander, 1996, Reasoning and Retrieval in Distributed Case Bases, *Journal of Visual Communication & Image Representation, Special Issue on Digital Libraries*, 7(1), 74—87.

M V Nagendra Prasad, 1998, Learning in Distributed Case Bases, Forthcoming Tech Report, Center for Strategic Technology Research, Andersen Consulting, Northbrook, IL.

Pearl Pu and Lisa Purvis, 1994, Formalizing Case Adaptation in a Case-based Design System, Proceedings of the Third International Conference on Artificial Intelligence in Design (AID94).

M. A. Redmond, 1990, Distributed Cases for Case-based Reasoning: Facilitating use of multiple cases, Proceedings of the Eighth National Conference on Artificial Intelligence, Cambridge, MA, 304 – 309.

Maneula Veloso and Jaime Carbonell, 1993, Derivational Analogy in PRODIGY: Automating Case Acquisition, Storage, and Utilization, *Machine Learning 10*, 249 – 278.

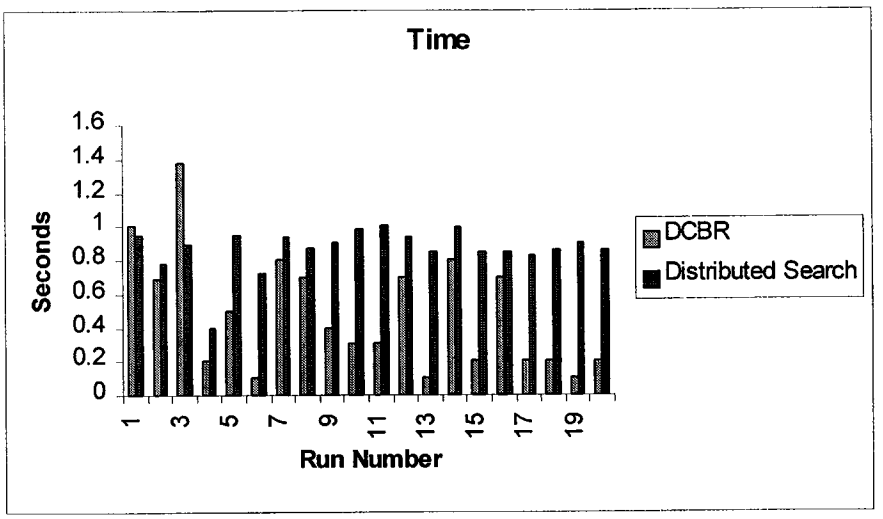

Figure 2: Time taken for a path planning run

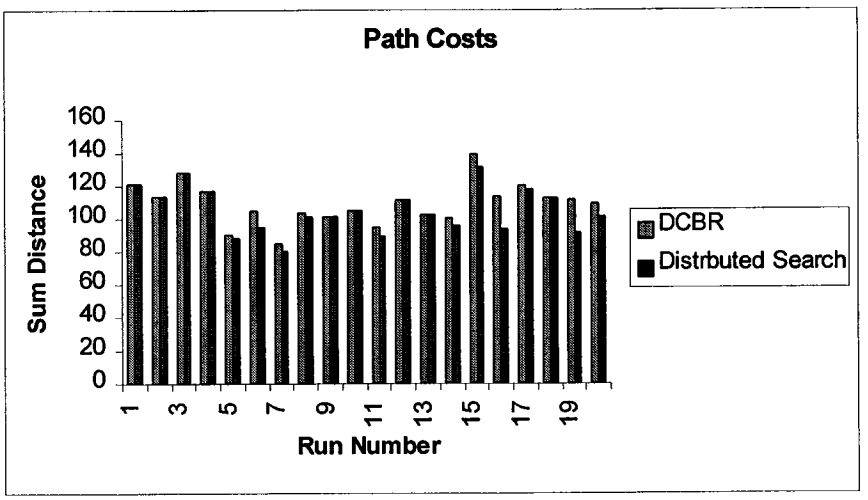

Figure 3: Path Costs for planning instances

Appendix

1. **Integration name/category:** MILTON integrates case snippets distributed across multiple agents.

2. **Performance Task:** Retrieval and composition of distributed case pieces in a distributed travel-planning domain.

3. **Integration Objective:** Essential in the case of multi-agent case-based systems with distributed cases.

4. **Reasoning Components:** Multiple agents that jointly perform a distributed A* search.

5. **Control Architecture:** Each agent is a CBR reasoner that works with peer agents to form a good overall case.

6. **CBR Cycle Step(s) Supported:** Retrieval and adaptation

7. **Representations:** Cases in all agents. All agents also do a cooperative search during case assembly

8. **Additional Components:** ?

9. **Integration Status:** Empirically evaluated (partially).

10. **Priority future work:** Further evaluation and exploration, especially for tougher tasks and domains.