Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches, as presented by Praveen Guddeti

Cory Lueninghoener

March 28, 2002

The discussion on case based reasoning began with a small discussion on how CBR is better than CSPs and some speculation about the gender of one of the authors, and then dove right into the meat of the topic.

The lecture began with a short history of CBR, including the roots of the field and a number of already-existing CBR systems. These systems include CYRUS, PROTOS, GREBE, HYPO, CABARET, CASEY, and MOLTKE. The list was appended by Dr. Choueiry to include CHEF(sp?), a system designed to create new Chinese-style recipes given a list of ingredients used in other recipes.

After the history of CBR we heard about the fundamentals of CBR. First we learned that the five main types of CBR methods are exemplar-based reasoning, instance-based reasoning, memory-based reasoning, case-based reasoning, and analogy-based reasoning and had a short discussion on each. At this point Dan asked how testing works, to which Praveen answered that tests can be run in the real world or in a computer simulation.

We then moved on to talk about the CBR cycle and its four processes: retrieve, reuse, revise, and retain. These four processes are shown in the decomposition diagrammed in Figure 2 in the paper. Dr. Choueiry wondered if one could jump around like a monkey from branch to branch in the tree shown in Figure 2, but it was concluded that you could only move linearly from node to node along connected paths.

We closed this section with a discussion about the three perspectives that a CBR system can be describe from (tasks, methods, and domain knowledge), and covered some problem areas in CBR. These problem areas had to do with the methods used for the four processes listed in the previous paragraph, as well as ways to represent knowledge.

Next we talked about how to represent cases. This is a very important topic, as a CBR system is very dependent on cases and their representation. This included a discussion on the importance of speed in case searching and case integration, as well as figuring out what to store and how to organize the case memory. We discussed two important case memory models: the dynamic memory model and the category and exemplar model.

The dynamic memory model is diagrammed in Figure 3 in the paper. Dr. Choueiry wondered why the model is called "dynamic," and Praveen answered that it is because the changing of the size of the tree is a dynamic process.

The category and exemplar model is diagrammed in Figure 4 in the paper and shows how features and cases are linked together to categories. At this point a couple people wondered why there are two links between Exemplar-1 and Exemplar-2, but nobody was able to come up with a good explanation.

Next was a discussion on case retrieval and the subtasks needed to retrieve a case: identification of features, the initial match, and the search and select. From this we learned that cases are retrieved based on syntactic and semantic similarities.

Lastly we talked about actual learning. There are three subtasks to this task: extraction, indexing, and integration. Extraction involves pulling information out of solved problem, whether that be by adding the present case to the knowledge base or by adding information to an already-existing case to make it cover the present case. With this we want to retain relevant problem descriptors, problem solutions, why the solution is a solution, the problem solving method, and failures. Here Dr. Choueiry and Amy wondered if a person has to come up wit the descriptors, to which Praveen answered that yes, the designer of the system is in charge of that.

Indexing involves figuring out what type of indices to use for future retrieval and how to structure the search space of indices, while integration involves actually putting the new knowledge into the knowledge base.

After touching on some current applications of case based reasoning and some future work in the area, the discussion moved on to a quick description of where CBR came from by Dr. Choueiry. She pointed out that first there were rule-based systems that just consisted of a bunch of `if...then...elsif...then...` rules. This idea was then changed such that all of the knowledge was compiled together into cases, which seemed to work better. Rob pointed out that the cases are not hard-edged, as there is some fuzziness in CBR (unlike rule-based systems).

At this point we moved on to some questions:

Tibor : This looks like a great use for a neural network. Why not use one of them?

Answer : You can't trace the reason with a neural net, and we want to be able to use that to increase case knowledge.

Xu Lin : If we keep adding cases to the database, doesn't it get really big eventually?

Answer : No, because eventually new cases are just subsets of cases already in the knowledge base, and thus they are "ignored."

Shabir : Does CBR use backjumping or something similar when searching?

Answer : Nope. It just matches features until a match is found.

Amy : CBR seems to be more general than CSP.

Answer : Yup. But they are different problems.

Rob : Go neural nets!

Unfortunately, that was all the time we had for questions during class. So, below are the rest of the questions/comments people had written.

Lin Xu : This is a very good paper. It opens a new window to how to solve problems by transferring it to a problem already solved. And I think this is more intelligent and more like a human mind.

If I have a new problem, the CBR will try to adapt it to an old problem or it can find this is a new problem then solve it.

Tibor : How often does the KB need to be updates? After ever case, or only on significant ones?

Page two only mentions storing solved cases, would a system benefit from storing unsolved ones as well?

Amy : CBR would be doomed in situations where every case has no similarity. Are there domains where this is the case?

Problems don't have to be identical to be retrieved from the KB - they can be retrieved on similarity

Do past cases determine more and less important features, or do they rely on human definition?

Rob : It's good to see rationalization of ideas based on classical psychology. I think that psychology is an important vector for finding new methods for approaching hard problems.

Would the storage and retrieval problem in CBR lend itself to the UB-Tree data indexing strategy? Database techniques in general would be useful, although I suppose the central problem is determining the relevant indices/features to use.

Cory : Ah, yes, decision trees. I like decision trees.

This sounds like a great application for a UB-Tree. Here we have a bunch of cases that can be described by their features and need to be searched through, which is exactly what a UB-Tree was designed for.