Synergy and Commonality in Case-Based and Constraint-Based Reasoning

Lisa Purvis
Xerox Corporation
800 Phillips Road, 128-51E
Webster, NY 14580
E-Mail: lpurvis@wrc.xerox.com

Abstract

Although Case-Based Reasoning (CBR) is a natural formulation for many problems, our previous work on CBR as applied to design made it apparent that there were elements of the CBR paradigm that prevented it from being more widely applied. At the same time, we were evaluating Constraint Satisfaction techniques for design, and found a commonality in motivation between repair-based constraint satisfaction problems (CSP) and case adaptation. This led us to combine the two methodologies in order to gain the advantages of CSP for case-based reasoning, allowing CBR to be more widely and flexibly applied. In combining the two methodologies, we found some unexpected synergy and commonality between the approaches. This paper describes the synergy and commonality that emerged as we combined case-based and constraint-based reasoning, and gives a brief overview of our continuing and future work on exploiting the emergent synergy when combining these reasoning modes.

Introduction

Synergy is defined as: *The working together of two things to produce an effect that is greater than the sum of their individual effects* (Miller 1997).

It is precisely this interaction between two entities that motivated us to investigate combining constraint satisfaction and case-based reasoning (CBR) to achieve case adaptation in our previous work (Purvis 1995; Purvis & Pu 1995). Working in the domain of CBR in design, we noticed that although CBR was a natural formulation for the design problem, there were elements of the CBR paradigm that prevented the methodology from being more widely applied. In particular, the difficult problem of case adaptation had been largely ignored in CBR because it encompassed many problematic characteristics: convergence upon a solution was difficult to guarantee, widely varying case representations made it difficult to combine several cases, and domain-specific adaptation rules were often required, limiting the application of an adaptation methodology to one domain.

At the same time, we were evaluating constraint satisfaction techniques for design and found a certain commonality in approach between repair-based constraint satisfaction (in the form of the minimum conflicts repair algorithm (Minton, Johnston, & Laird 1992)), and case adaptation. This led us to combine the two paradigms in order to allow CBR to take advantage of the characteristics of the constraint satisfaction problem (CSP), which alleviated some of the difficulties of case adaptation, and has resulted in a CBR framework that can be more widely and flexibly applied.

In our system COMPOSER (Purvis 1995), which solved assembly sequence and configuration design problems with a combined CSP/CBR approach, we found both expected and unexpected benefits from integrating the two problem solving paradigms. We give a brief overview of this work on CSP for case adaptation and the synergy that we found in Section 1.

As we continue our work on problem solving via combined CBR and CSP techniques, our successes in integrating the two paradigms into a synergistic whole have also led us to explore integrating yet another problem solving paradigm, the Genetic Algorithm (GA), to take over where the combined CBR/CSP approach fails to satisfy our efficiency requirements. These new ideas are outlined in Section 2.

Our continuing work is also allowing us to identify additional potential commonalities between CSP and CBR that could be exploited to provide even more synergistic problem solving in the future. These commonalities are described in Section 3, and we conclude with a summary in Section 4.

Case Adaptation as a Constraint Satisfaction Problem

Adaptation is the component of a case-based reasoning system that adapts retrieved cases to fit new circumstances. It is often considered to be the most difficult problem in case-based reasoning (Leake 1995).

In order to enable an efficient and systematic adaptation process that combines several cases, we developed a case combination methodology in which each existing case is represented as a discrete CSP. The methodology then combines the cases that match the new problem situation into a valid solution for the new problem by applying the minimum conflicts repair algorithm (Minton, Johnston, & Laird 1992). The formalized adaptation methodology enables application of this adaptation process to any problem that can be formulated as a discrete CSP.

Benefits to CBR from using CSP formalism

Several benefits to case-based adaptation were realized as a result of formulating cases as constraint satisfaction problems and using a constraint satisfaction algorithm to do the adaptation.

Controlled Adaptation Controlling adaptation can typically be difficult, as described in (Hua & Faltings 1993), where it was observed that changing one feature during the adaptation process may result in non-convergent behavior for the adaptation. CSP provides a technique by which to ensure convergence on a solution, thereby controlling the adaptation process.

Domain Independent Adaptation To achieve case adaptation, a case-based reasoner must know two key pieces of information: *what* from the case to adapt and *how* to adapt it. Rather than relying on domain-specific adaptation rules, a case-based reasoner that combines cases via the minimum conflicts algorithm can determine this information solely from the constraint information already contained in the problem representation.

For instance, the minimum conflicts algorithm tells the reasoner that the variables that are in conflict with the rest of the case features are the ones to adapt. So for example, in the problem shown in Figure 1A, where the variables are the nodes in the graph, variables 1, 2, and 4 are all in conflict, and therefore must be adapted/repaired.

Furthermore, the minimum conflicts algorithm tells the reasoner *how* to adapt by giving information about the number of constraint violations for each value. In the example shown in Figure 1, if we are adapting variable 1, then the adaptation to choose for it is the one which minimizes the number of conflicts. Thus in this example, the value *c* will be chosen as the adaptation for variable 1, since it causes only two conflicts. The reasoner requires no additional domain-specific information in order to come to this conclusion about what to adapt and how to adapt it.

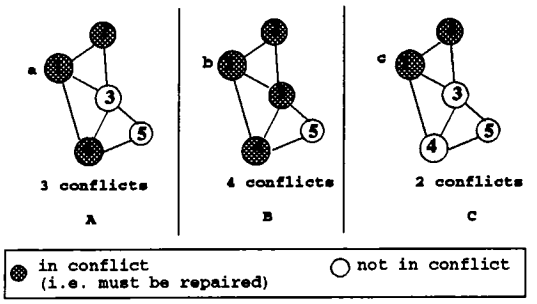

Figure 1: Domain Independent Adaptation

Note that our initial work performed such an adaptation based strictly on the number of conflicts, and did not consider information about the differing relative importances of the conflicts. Our follow-on work takes into account the importance of conflicts at the borders of cases in low adaptability situations. In this way, the adaptation process takes conflict weights into account without introducing domain-specific information.

Case Combination The systematic nature of CSP algorithms allows us to more efficiently combine *several* cases - a topic that has been problematic in CBR thus far. In CBR systems that use domain-dependent adaptation rules, it is difficult to combine cases efficiently because of the uncontrolled interactions between cases that can arise when they are combined. This has resulted in many CBR systems that adapt just one case. Using a CSP algorithm to do the adaptation has allowed us to increase the flexibility of our case-based system to allow the combination of several cases. In addition to the systematicity that CSP gives to the adaptation process, describing each case as a CSP provides a uniform knowledge representation, enabling many cases from varying environments to be combined more easily.

Unexpected Benefits of Combining CSP and CBR

The benefits to CBR outlined in the previous section were ones that we had anticipated, and in fact were our motivation for initially combining the two methodologies. As we proceeded with our experiments, we found that there were several unexpected benefits to formalizing case adaptation as a constraint satisfaction problem.

Increased Efficiency of Constraint Solving Our experiments showed that overall, starting with case solutions outperformed solving the CSP from random initial solutions (Purvis 1995). However, we also found

that in some situations, the case solutions did *not* provide better performance than from-scratch problem solving. This observation led us to another unexpected benefit of using CSP for case adaptation: the development of an adaptability criterion.

Adaptability Criterion The constraint satisfaction formalism enabled us to develop a measure of adaptability of the retrieved cases. Adaptability has been an elusive topic in CBR, since it is difficult to tell before solving the problem whether it will adapt easily or not. At the same time, it is a very useful piece of information to have before expending adaptation effort only to later find that it is fruitless.

Our experiments with assembly sequence generation and configuration design problems showed that the minimum conflicts algorithm could not perform an efficient case combination when there were a large number of highly constrained, initially inconsistent edge variables (i.e. variables at the boundary between two cases) (Purvis 1995). Our initial adaptability criterion is based on these two factors and is described in detail in (Purvis & Athalye 1997). By imposing a constraint satisfaction formalism on the adaptation process, we were able to define this domain independent adaptability criterion.

Synergy Between CSP and CBR

We found a strong synergy between CSP techniques and CBR techniques, whereby CSP helped to formalize the adaptation process, enabling systematicity and added flexibility for the case based reasoner, while CBR helped to increase the efficiency of constraint solving.

As we also mentioned, however, there were situations in which the combined CSP/CBR technique did not perform efficiently. It is for this reason that we have begun to investigate incorporating another reasoning paradigm into our problem solver, in order to better guide the solving in these difficult situations.

Incorporating a Genetic Algorithm into the Problem Solver

In order to improve case adaptability, we have begun to investigate the incorporation of a genetic algorithm into the adaptation process (Purvis & Athalye 1997). The reason that the minimum conflicts CSP algorithm does not perform well in some case combinations is that it vacillates around a local minimum when the adaptability of the cases is low. An example of this is shown in Figure 2, where two cases are combined.

Each case is individually consistent, but the two are inconsistent with one another, because of conflicts at the borders of the case solutions. To achieve a global

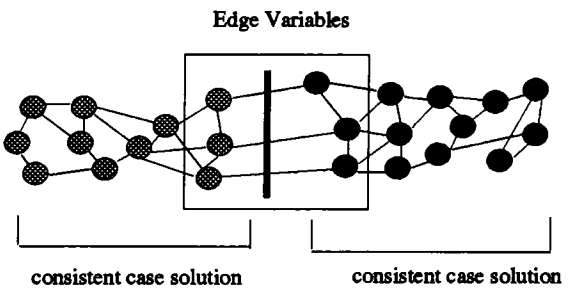

Figure 2: Low Adaptability Example

solution in such a low adaptability case, the boundary between consistent solutions must be pushed completely to the right or left during repair. However, pushing the boundary to the right or left causes new case conflicts, which in turn increases the total number of conflicts. This causes the minimum conflicts algorithm to vacillate around the initial case boundary.

We have chosen a GA to alleviate the difficulties in low adaptability situations because of its ability to operate both with a progression towards an improved solution, and to examine multiple regions of the search space via crossover and mutation operators, thereby enabling a quick escape from initial local minimum. We have introduced a fitness function for the GA that will perturb a case solution so that when it is subsequently re-combined with the other cases, its adaptability has been improved enough to allow a more efficient combination (Purvis & Athalye 1997).

The fitness function we have developed is based on the principle that it is more important to satisfy those constraints that affect a larger number of variables. In these types of critical constraints, the effect of changing the value of one variable would be reflected in its other constraints (we will call this the variable's *propagation potential*). Our propagation potential measurement considers *inter-problem* constraint violations to be more important than *intra-problem* constraint violations. An inter-problem constraint is one that crosses a case boundary, and an intra-problem constraint is one that connects only variables in the same case, as shown in Figure 3. In effect, an inter-problem constraint is one that connects edge variables.

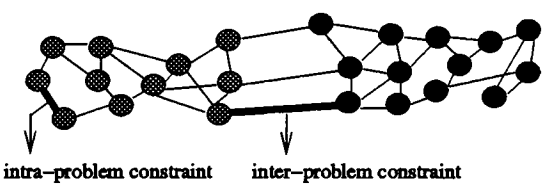

Figure 3: Inter/Intra-problem constraint example

In evaluating a solution, we want to ensure that the edge variables are correct, and also that the intra-problem variables that are highly connected are also correct. These two criteria help us to favor those solutions that will not cause propagation into the neighboring case, and also to favor those that promise to have little intra-problem repair. This propagation potential measurement and its associated fitness function are described in detail in (Purvis & Athalye 1997).

Further Exploitable Commonalities Between CSP and CBR

In continuing our work on combining CSP and CBR to achieve efficient and flexible case combination, we have noticed a similarity in the motivation and challenges faced by case adaptation and dynamic constraint satisfaction problems (DCSP).

Dynamic CSP and its Similarities to CBR

The traditional CSP framework is a static framework, where the set of variables and constraints is completely known and fixed at the beginning of problem solving. However, many real life problem domains involve an environment where the variables and constraints in the problem evolve as time goes on (e.g. reactive scheduling, design). For these reasons, the dynamic CSP formalism has been developed, which relaxes the assumption that the problem variables and constraints remain fixed. The DCSP is a sequence of CSPs, where each one differs from the previous one by the addition or removal of some constraints (Dechter & Dechter 1988).

Both DCSP and CBR recognize that solving from scratch each time a new problem is encountered is inefficient. Significant effort is being placed on discovering ways by which to reuse solutions in DCSPs (Bellicha 1994; Hentenryck 1990; Verfaillie & Schiex 1994). Similarly, effectively reusing old solutions to solve new problems is the adaptation problem in CBR.

However, in both DCSP and in CBR, starting from certain existing solutions does not always result in reduced solving complexity. As we found in our experiments, using existing cases did not always result in fewer backtracks to a solution. Similarly in DCSP, even if a small change is made in the constraint network, this small change can make re-solving very difficult. Thus in both CBR and DCSP, the best matching cases do not always constitute easily adaptable problems.

Synergy Between DCSP and CBR

Our current work proposes that using a CBR framework for DCSP enables more efficient and flexible dynamic constraint solving.

The case-based reasoning strategy of combining solutions can be especially important for CSP, where time to solve can increase exponentially with the size of the problem. Using a CBR system that allows case combination enables the solutions of several smaller CSPs to be stored and re-used to efficiently solve larger problems. Thus, the experience of previous solutions can be re-used even in the first iteration of a dynamic CSP. Those sub-cases that best match the larger problem can be retrieved and combined via the constraint-based adaptation described in our previous work (Purvis 1995).

Second, a *similarity measure* can be defined that will search the case base full of previously solved problems for those that are 'most similar' to the new problem. By using a CBR framework for solving a DCSP, the CBR similarity measure can be used to free the DCSP from always having to begin with the solution found during the previous iteration of the DCSP. Rather, there is a rich case base of starting points to choose from, making it more likely that the retrieved solutions will be 'closer' to the new solution, requiring less computation to solve. Our future work in this area involves comparing existing approaches for reusing solutions in DCSP with the CBR approach.

Conclusion

We outlined our previous work on case adaptation using constraint satisfaction techniques, and showed the synergy that we found between the two reasoning paradigms. CSP helped CBR by providing a systematic methodology for case adaptation that enabled multi-case adaptation, convergence upon a solution, a domain independent adaptation methodology, and definition of an adaptability criterion. CBR helped CSP by providing an increased efficiency of constraint solving by using cases from the case base.

Our work on combining the two paradigms has also led us to discover possibilities for extending the problem solving framework with a Genetic Algorithm for those situations where case adaptability is low, and for utilizing case adaptation techniques to increase the efficiency of dynamic constraint solving.

In these ways, the combination of the two reasoning modes provided us with results that would not have been possible by using one reasoning paradigm exclusively. This in turn gives evidence that multi-modal reasoning has many real and practical benefits.

References

Bellicha, A. 1994. Maintenance of solution in a dynamic constraint satisfaction problem. In *Artificial Intelligence in Engineering.*

Dechter, R., and Dechter, A. 1988. Belief maintenance in dynamic constraint networks. In *Proceedings of the Sixth National Conference on Artificial Intelligence*, 37 – 42.

Hentenryck, P. V. 1990. Incremental constraint satisfaction in logic programming. In *Seventh International Conference on Logic Programming*, 189 – 202.

Hua, K., and Faltings, B. 1993. Exploring case-based building design - cadre. In *Artificial Intelligence in Engineering Design, Analysis, and Manufacturing.*

Leake, D. 1995. Cbr in context: The present and future. In *Case Based Reasoning: Experiences, Lessons, and Future Directions.*

Miller, G. A. 1997. *WordNet 1.5 Database (online dictionary, at http://www.cogsci.princeton.edu/~wn).* Princeton University, Cognitive Science Laboratory.

Minton, S.; Johnston, M.; and Laird, P. 1992. Minimizing conflicts: A heuristic repair method for constraint satisfaction and scheduling problems. *Artificial Intelligence* 161–205.

Purvis, L., and Athalye, S. 1997. Towards improving case adaptability with a genetic algorithm. In *Proceedings of the Second International Conference on Case-Based Reasoning*, 403–412.

Purvis, L., and Pu, P. 1995. Adaptation using constraint satisfaction techniques. In *Proceedings of the First International Conference on Case-Based Reasoning, Sesimbra, Portugal*, 289–300.

Purvis, L. 1995. *Intelligent Design Problem Solving Using Case-Based and Constraint-Based Techniques.* Ph.D. Dissertation, The University of Connecticut.

Purvis, L. 1997. Dynamic constraint satisfaction using case-based reasoning techniques. In *Proceedings of the CP'97 Workshop on Dynamic Constraint Satisfaction.*

Verfaillie, G., and Schiex, T. 1994. Solution reuse in dynamic constraint satisfaction problems. In *Proceedings of the 12th National Conference on Artificial Intelligence*, 307–312.