If it worked last time, it should work again: justifying and validating case-based plans

Kathryn E. Sanders
Department of Computer Science
University of Massachusetts
Amherst, MA 01003
sanders@freya.cs.umass.edu[1]

Throughout artificial intelligence, more attention is being given to validation and evaluation of programs. This is especially an issue in case-based reasoning, since the domains for which CBR is best suited are those with a weak domain theory, such as law, medical diagnosis, cooking, and military strategy. In such domains, validation must take on a different meaning. We cannot "prove" that the output of our programs is correct, but we can test the programs in various ways and accumulate convincing evidence that they work.

There are various methods for accumulating such evidence. For planners, we can try the plan out (in simulation or in reality) and see how it works. In adversarial domains such as chess, we can test the system against stronger and stronger opponents. We can input the facts of an actual case not in the case base and compare the system's results with those in the actual case. Similarly, we can use the facts and results of examples from some authoritative source. We can ask experts in the field to evaluate the system's results.

In this paper, I focus on an additional method: presenting the cases on which the system's solution was based and explaining how they support that solution. Understanding the way in which the system operates increases the user's confidence in the results. In rule-based expert systems, this is called an explanation facility. Typically, it involves displaying the rules applied in arriving at a conclusion. CBR systems have the advantage that by using cases, they can provide an explanation that is generally much easier to understand and more compelling than a chain of rules.

Perhaps because citations are essential in any legal writing, case-based legal analysis systems have generally included in their output a discussion of the cases used and how they relate to the current situation; in this paper, I describe the implementation of a similar technique within the context of a case-based planner called CHIRON in the legal domain. CHIRON constructs plans for simple individual income tax problems, annotated with HYPO-style arguments for and against the success of each plan, based on the plan's similarity or dissimilarity to previous cases.

This technique could be incorporated in any system that uses cases to construct plans or analyze situations, and in fact, it is generally implicit in such programs, since an execution trace will show which cases were chosen and how they were used. An explicit explanation facility is particularly appropriate in the domain of tax planning, because in this domain, certain other methods of validation such as experiment and simulation are not available, but it could be used in any domain. It is not a sufficient means of validation on its own, but is a useful supplement to whatever other methods are chosen.

[1]This paper is based on work done at Brown University, supported by the Advanced Research Projects Agency of the Department of Defense monitored by the Air Force under Contract No. F30602-91-C-0041.