Solving Mutilated Problems

Ramin Ramezani and Simon Colton
*Computational Creativity Group, Department of Computing, Imperial College, London, UK
raminr,sgc@doc.ic.ac.uk

1 Introduction and Motivation

Constraint solving, theorem proving and machine learning provide powerful techniques for solving AI problems. In all these approaches, information known as background knowledge needs to be provided, from which the system will infer new knowledge. Often, however, the background information may be obscure or incomplete, and is usually presented in a form suitable for only one type of problem solver, such as a first order theorem prover. In real world scenarios, there may not be enough background information for any single solver to solve the problem, and we are interested in cases where it may be possible to combine a machine learner, theorem prover and constraint solver in order to best use their incomplete background knowledge to solve the problem. We present here some preliminary experiments designed to test the feasibility of such an approach. We concentrate on the scenario of a police investigation of a murder case. In such a scenario, there may be previous solved cases which bear resemblance to the current case. Given that the previous cases were solved, one can imagine employing a machine learning system to learn a set of rules which can classify suspects in a case as either guilty or not guilty. The rule set could then be applied to the current case. If only one person was classified as guilty, this would solve the problem. While this reasoning may not be sound, it would at least help to identify a *prime suspect*. In addition, in the current case, there may be information describing the particulars of the case, arising from physical evidence, motives, alibis, general knowledge, etc. If so, it may be possible to define a set of constraints that the guilty suspect must satisfy, and then use a constraint solver to rule out suspects. If only one suspect satisfies all the constraints, again the problem is solved. Alternatively, the same information about the case may be used as axioms in a theorem proving setting. In such a setting, one could attempt to prove a set of conjectures, each one stating that a particular suspect is guilty/not guilty. If only one suspect is proved to be guilty (or alternatively, it is possible to prove that all but one suspects are not guilty), then the problem is once again solved.

2 The Mutilated Aunt Agatha Problem

To show the feasibility of using three different types of solvers to attack the same problem, we looked at the "Who Killed Aunt Agatha" problem from the TPTP library (i.e., problem `PUZ001` in (Sutcliffe and Suttner, 1998), originally from (Pelletier, 1986)). The background knowledge for this problem is stated in English as follows: Someone who lives in Dreadbury Mansion killed Aunt Agatha. Agatha, Butler and Charles live in Dreadbury Mansion and are the only people who live therein. A killer always hates his victim and is never richer than the victim. Charles hates no one that Aunt Agatha hates. Agatha hates everyone except the Butler. The Butler hates everyone not richer than Agatha. The Butler hates everyone Aunt Agatha hates. No one hates everyone and Agatha is not the butler. This problem is usually posed as a logic puzzle for theorem provers, where the aim is to prove that Aunt Agatha killed herself. However, in a more general setting, the answer wouldn't be given, i.e., we would be asked to find out who killed Aunt Agatha. With this tweak, we can make it amenable to the three different solving approaches as described above.

To show that – in principle – such problems are amenable to a machine learning approach, we firstly invented some data which embodies the axioms of the problem. In particular, we wrote down the details of five case studies with three people in, one of whom had been murdered. We specified who was richer than who, who hated who, who was killed and who the murderer had turned out to be. This was done in such a way that (a) there was a direct mapping from Agatha, Butler and Charles to one of the people in the case study, where the Agatha character was always killed and (b) all the axioms from the problem statement about who could possibly hate who, etc., were upheld. In the first instance, the data reflected the fact that the murderer and the victim were always the same person – the Agatha character. This data was produced in the syntax of the Progol machine learning system (Muggleton, 1995). We ran Progol and it hypothesised the rule that $killed(A, A)$. Given that Progol's output is generated in Prolog syntax, it was very easy to check that this profile applied to only Aunt Agatha in the current case. To make matters more interesting, in the second instance, we generated the data to still satisfy the axioms, but we varied the murderer/victim combination. In this instance, Progol hypothesised the following rule: $killed(A, B) \leftarrow hates(A, B), \neg richer(A, B)$. Again, when we applied this to the data about the current case, only Aunt Agatha fitted the profile.

To show that such problems are amenable to a constraint solving approach, we wrote a constraint satisfaction problem (CSP) in the syntax of the Sicstus Prolog CLPFD module (Carlsson et al., 1997). In brief, the CSP had one variable which could take one of three values representing Agatha, Butler and Charles respectively, and was constrained as per the axioms of the problem. Sicstus solved the problem in such a way that it ruled out Butler and Charles, but could not rule out Agatha, hence solving the problem of who killed Aunt Agatha. Finally, we specified six conjectures to the Otter theorem prover (McCune, 1994). The axioms in the conjectures represented the information from the problem statement, and the conjectures were respectively: Agatha killed/didn't kill Agatha; Butler killed/didn't kill Agatha; Charles killed/didn't kill Agatha. Otter successfully proved that Agatha killed Agatha, and that Butler and Charles didn't kill Agatha. If failed to prove any of the other conjectures. This shows that such *whodunnit* problems are amenable to solution by theorem provers.

The Aunt Agatha problem becomes more interesting if we remove information from each of the three problem statements in such a way that neither Progol, Sicstus nor Otter can solve the problem. We can then investigate methods for combining these reasoning systems in such a way that a solution can still be found. Our experiments are still preliminary, and we plan in future to investigate many different ways to mutilate the problem, yet still solve it via a combination of systems. So far, we have only investigated one opportunity for combining different reasoning systems. In particular, from the theorem proving and CSP problems, we removed the axiom that "no-one hates everyone". This is crucial to solving the problem, because without it, Sicstus cannot rule out Butler as the killer, and Otter can similarly prove that both Butler and Agatha killed Agatha. We investigated whether the data from the machine learning approach could be used to recover the missing axioms. In particular, we employed the HR automated theory formation (Colton, 2002) to form a theory about the previous case studies. Details are omitted, but using HR's `forall`, `exists`, `negate` and `compose` production rules, HR made the conjecture that in all case studies: $\nexists\, x$ s.t. $person(x) \wedge (\forall\, y,\ (person(y) \rightarrow hates(x, y)))$. This states that, in all cases, there is no person who hates everyone. Hence we see that HR has recovered the missing axiom, which could be used by the constraint solver or prover to solve the problem.

3 Future Work

We are building a system which is able to take a general problem statement, such as a *whodunnit* problem and translate it to the syntax of various solvers such as Sicstus, Otter and Progol. Moreover, in situations where none of the solvers are initially successful, the system will be able take the partial solutions from each solver and see whether these can be used together to fully solve the problem. In addition, the system will employ a theory formation program such as HR to discover potential axioms exhibited by the data which enable a solution to be found. This will give us a platform to investigate more exotic combinations of reasoning systems which are able to solve ill-formed problems. For instance, the axioms provided to a theorem prover could be used to generate artificial data to supplement the given data in a machine learning problem, hopefully enabling the learner to solve the problem. We believe that such combined AI systems will enable more powerful, more flexible solvers to be built and employed.

Acknowledgements

This work is funded by EPSRC grant EP/F036647. We would like to thank John Charnley and Murray Shanahan for their input into the mutilated Aunt Agatha problem.

References

M Carlsson, G Ottosson, and B Carlson. An open-ended finite domain constraint solver. In *Proc. Programming Languages: Implementations, Logics, and Programs*, 1997.

S Colton. *Automated Theory Formation in Pure Mathematics*. Springer-Verlag, 2002.

W McCune. Otter 3.0 Reference Manual and Guide. Technical Report ANL-94/6, Argonne National Laboratory, Argonne, USA, 1994.

S Muggleton. Inverse entailment and Progol. *New Generation Computing*, 13:245–286, 1995.

F Pelletier. Seventy-five Problems for Testing Automatic Theorem Provers. *Journal of Automated Reasoning*, 2(2):191–216, 1986.

G Sutcliffe and C Suttner. The TPTP problem library: CNF release v1.2.1. *Journal of Automated Reasoning*, 21(2): 177–203, 1998.