The Complexity of Constraint Satisfaction Revisited

Alan K. Mackworth

Department of Computer Science
University of British Columbia
Vancouver, B.C. V6T 1W5
Canada

Eugene C. Freuder

Department of Computer Science
University of New Hampshire
Durham, NH 03824
U. S. A.

Abstract

This paper is a retrospective account of some of the developments leading up to, and ensuing from, the analysis of the complexity of some polynomial network consistency algorithms for constraint satisfaction problems.

1 Historical Context

In 1970 one of us (AKM) worked on an implementation of Huffman-Clowes [1] labeling of line drawings. This exploited the consequences of a deceptively simple constraint on the visual world of planar objects: the three-dimensional interpretation of a line as an edge must be the same at both ends. Unfortunately, he observed that standard breadth-first and depth-first search techniques suffered from severe combinatorial explosions. About the same time the other one of us (ECF) shared a graduate student office in the M.I.T. AI Lab with David Waltz, who was also working on a program to interpret line drawings. Waltz designed a *filtering* process to remove inconsistent interpretations during the analysis of a scene [21], making the combinatorial explosion manageable. Waltz observed experimentally that the effort required for this filtering process was "roughly" linear in the size of the scene. A heuristic argument based on the semantics of his domain supported the plausibility of this behavior.

Since this technique appeared to have promise, AKM described a class of network consistency algorithms [10], abstracted away from the applications, which contains, amongst others, the algorithms described by Waltz [21] and Ugo Montanari [16]. Incidentally, one of the referees of [10] suggested further complexity analysis of the problems and the algorithms could be done. Bernard Meltzer, the founding editor of *Artificial Intelligence*, agreed but did not require it for publication. He suggested it as a topic for a sequel as, indeed, it became. John Gaschnig subsequently raised some doubt about the linear behavior of filtering [8]; however, he was careful not to draw any firm conclusions from the limited data, and the complexity of the process remained an open issue.

Both of us solved this problem, independently, in 1981. Raimund Seidel, a student in AKM's graduate course, had achieved a nice new algorithm [18]. In the course of discussion with ECF, Seidel realized we (AKM and ECF) each had the same result. We joined forces and eventually the paper appeared [11].

2 Complexity and Network Consistency

One outcome of our 1985 paper [11] was a resolution of the open issue. Heuristic intuition and experimental data could not, by their nature, hope to achieve a complete resolution of the question. We used formal analytical techniques to prove that the filtering process could be carried out in linear time for any application.

The proof relied on our analysis of an abstraction of the visual filtering process called *arc consistency*. Arc consistency is a basic tool in what has come to be called *constraint-based reasoning*. Constraint-based reasoning has been widely used in artificial intelligence: in vision, language, planning, diagnosis, scheduling, configuration, design, temporal reasoning, defeasible reasoning, truth maintenance, qualitative physics, logic programming and expert systems. The analysis of techniques like arc consistency can thus lead to tractability results in many areas of artificial intelligence.

A *constraint satisfaction problem* (CSP) involves finding values for a set of problem variables which simultaneously satisfy a set of restrictions (*constraints*) on which combinations of variables are acceptable (*consistent*). The Huffman-Clowes-Waltz scene labeling problem is a Finite CSP (FCSP) since the variable domains are discrete and finite. Our complexity results were for FCSPs.

One of the key insights of arc consistency for FCSPs can be found in Fikes' paper in the very first issue of *Artificial Intelligence* [6]; in particular, if a value, c, for one problem variable is inconsistent with all values for some other problem variable, then c will never participate in a complete solution to the problem and can be eliminated from all further consideration. The obvious algorithm for removing all such inconsistencies, AC-1, has an $O\left(n^3d^3\right)$ complexity bound, for an FCSP with n variables each with d possible values. AC-3, a simpler and more general version of the Waltz filtering algorithm AC-2, was shown in our paper to have an $O\left(n^2d^3\right)$ bound.

That bound can be expressed as $O\left(ed^3\right)$, where e is the number of constraints, or edges in a *constraint graph*, whose vertices correspond to variables and whose edges correspond to constraints between variables. (We will restrict our attention here to binary constraints, which involve only two variables; analogous methods are available for dealing with higher order constraints.) Since scene labeling problems have planar constraint graphs, and for planar graphs the number of edges is linear in the number of vertices, we were able to show that arc consistency for the scene labeling problem is linear in the number of problem variables. We also showed that path consistency, a generalization of arc consistency, could be achieved in time cubic in the number of variables.

The complexity of arc consistency has since been refined further. Mohr and Henderson [15] found an arc consistency algorithm, AC-4, which has a theoretically optimal $O\left(ed^2\right)$ bound. (In retrospect, we regret that this did not fall out in our paper; optimality was within our grasp — only a factor of d away!) This brought the complexity of scene labeling filtering down to

$O(nd^2)$. However, better bounds have been found for arc consistency for restricted classes of problems. In particular, Perlin [17] has identified a class of problems that includes scene labeling for which arc consistency can be obtained in time linear in d. Thus arc consistency can, in fact, be obtained for scene labeling in time that is linear in both the number of variables and the number of values per variable. There are even cases where it can be obtained in $O(e \log d)$ [12]. This may be the end of that story, but there are other stories to tell, too many for this short note.

3 Tractable Problem Classes

It is important to realize that the varying forms of consistency algorithms can be seen as *approximation* algorithms, in that they impose *necessary* but not always *sufficient* conditions for the existence of a solution on a CSP. Each of them can be thought of as a low-order polynomial algorithm for exactly solving a relaxed version of an FCSP whose solution set contains the set of solutions to the FCSP. The more effort one puts into finding the approximation the smaller the discrepancy between the approximating solution set and the exact solution set.

Since FCSPs are so hard (NP-complete) as a general class, it became important to identify specific classes of problems which admit tractable solution techniques. Tradeoffs can be made between representational and computational complexity, trading representational complexity to remain within the comfortable computational confines of a tractable problem class. These tractable classes can also be used to assist in the solution of more general problems. One way to identify these classes is to look for restricted FCSP classes where the approximation algorithms are *exact*, namely, where the consistency conditions are necessary *and* sufficient. These classes can be characterized by restrictions on the topology of the constraint graph, on the size of the domains or on the nature of the constraints. We pointed out this possibility, giving one concrete example and leaving it as an open issue to identify others.

FCSPs with tree-structured constraint graphs were the first such tractable class to be identified, and provide a good illustration of these issues. Our paper provided an $O(nd^3)$ bound on the complexity of tree-structured problems (improved to an optimal $O(nd^2)$ in [3]). Tambe and Rosenbloom used these results to bound the complexity of production rule pattern matching by restricting to tree structures [19]. Dechter, Pearl and Meiri have demonstrated how tree-structured substructure or superstructure can assist in the solution of non tree-structured problems [3,4,2,14]. Complexity bounds have been obtained for "higher-level" tree structures, where each level trades increased representational power for increased complexity [7].

One of the practical consequences of our results was that the designers and implementers of constraint-based programming languages could feel comfortable including consistency algorithms as primitives in the language [6,10]. Ideally, a language primitive should require constant time; but, failing that, it is comforting to know that it will terminate in linear time. The constraint logic programming language CHIP [20] was the first to exploit this potential fully by providing an arc consistency based inference engine.

Progress continues to be made on finding efficient ways to solve important classes of problems, e.g. Deville and Van Hentenryck's $O(ed)$ algorithm for a successor to CHIP [5], and on identifying the trade-offs between representational adequacy and computational complexity, e.g. Meiri's clarification of the effort required to answer consistency questions for classes of temporal reasoning problems [13].

Another interesting follow-on result was that although arc consistency is achievable in linear sequential time there is apparently no polylogarithmic time parallel algorithm in the general case: it is log-space complete for P [9] and, hence, unlikely to be in NC. (There are, though, well-behaved parallel and distributed algorithms for some special cases [22].) This negative result struck some as counter-intuitive. Algorithm AC-1, which has poor sequential complexity, has a high degree of intrinsic parallelism (but potential serial data dependencies); whereas each AC-p (p>1) has been optimized for a single processor. In fact, various generalizations of AC-1 have been proposed for neural networks. But the gloomy theoretical result has not deterred the designers of AC VLSI chips or other intrepid experimentalists.

4 Conclusion

The development of constraint satisfaction algorithms was originally motivated by concerns for efficiency. The subsequent analysis of the complexity of both the problems and the algorithms further stimulated the development of practical tools and the identification of significant tractable problem classes. So the history of the topic is a tale of intimate interaction amongst theory, implementation, experiment and application characteristic of artificial intelligence research.

Acknowledgment

This material is based, in part, upon work supported by the National Science Foundation under Grant No. IRI-8913040 to Eugene Freuder. The United States Government has certain rights in part of this material. Alan Mackworth is supported by the Shell Canada Fellowship of the Canadian Institute for Advanced Research, by the Institute for Robotics and Intelligent Systems Network of Centres of Excellence and by the Natural Sciences and Engineering Research Council of Canada.

References

[1] Clowes, M. B. On seeing things. *Artificial Intelligence 2* (1971), 79–116.

[2] Dechter, R. Enhancement schemes for constraint processing: backjumping, learning and cutset decomposition. *Artificial Intelligence 41* (1990), 273–312.

[3] Dechter, R., and Pearl, J. Network-based heuristics for constraint satisfaction problems. *Artificial Intelligence 34* (1988), 1–38.

[4] Dechter, R., and Pearl, J. Tree clustering for constraint networks. *Artificial Intelligence 38* (1989), 353–366.

[5] Deville, Y., and Van Hentenryck, P. An efficient arc consistency algorithm for a class of CSP problems. In *Proc. 12th International Joint Conf. on Artificial Intelligence* (1991), pp. 325–330.

[6] Fikes, R. E. REF-ARF: a system for solving problems stated as procedures. *Artificial Intelligence 1* (1970), 27–120.

[7] Freuder, E. C. Complexity of k-tree structured constraint satisfaction problems. In *Proc. 8th National Conference on Artificial Intelligence* (Cambridge, Massachusetts, July 1990), pp. 4–9.

[8] Gaschnig, J. *Performance measurement and analysis of certain search algorithms*. Thesis CMU-CS-79–124, Department of Computer Science, Carnegie-Mellon University, Pittsburgh, Penn., 1979.

[9] Kasif, S. On the parallel complexity of discrete relaxation in constraint satisfaction networks. *Artificial Intelligence 45* (1990), 275–286.

[10] Mackworth, A. K. Consistency in networks of relations. *Artificial Intelligence 8* (1977), 99–118.

[11] Mackworth, A. K., and Freuder, E. C. The complexity of some polynomial network consistency algorithms for constraint satisfaction problems. *Artificial Intelligence 25* (1985), 65–74.

[12] Mackworth, A. K., Mulder, J. A., and Havens, W. S. Hierarchical arc consistency: exploiting structured domains in constraint satisfaction problems. *Computational Intelligence 1* (1985), 118–126.

[13] Meiri, I. Combining qualitative and quantitative constraints in temporal reasoning. In *Proc. 9th National Conference on Artificial Intelligence* (1991), pp. 260–267.

[14] Meiri, I., Dechter, R., and Pearl, J. Tree decomposition with applications to constraint processing. In *Proc. 8th National Conference on Artificial Intelligence* (1990), pp. 10–16.

[15] Mohr, R., and Henderson, T. C. Arc and path consistency revisited. *Artificial Intelligence 25* (1986), 65–74.

[16] Montanari, U. Networks of constraints: fundamental properties and applications to picture processing. *Information Sciences 7* (1974), 95–132.

[17] Perlin, M. Arc consistency for factorable relations. *Artificial Intelligence* (to appear).

[18] Seidel, R. A new method for solving constraint satisfaction problems. In *Proc. 7th International Joint Conf. on Artificial Intelligence* (Vancouver, BC, 1981), pp. 338–342.

[19] Tambe, M., and Rosenbloom, P. A framework for investigating production system formulations with polynomially bounded match. In *Proc. 8th National Conference on Artificial Intelligence* (1990), pp. 693–700.

[20] Van Hentenryck, P. *Constraint Satisfaction in Logic Programming*. MIT Press, Cambridge, Massachusetts, 1989.

[21] Waltz, D. Understanding line drawings of scenes with shadows. In *The Psychology of Computer Vision*, P. Winston, Ed. McGraw-Hill, New York, 1975, pp. 19–91.

[22] Zhang, Y., and Mackworth, A. K. Parallel and distributed algorithms for finite constraint satisfaction problems. In *Proc. 3rd IEEE Symposium on Parallel and Distributed Processing* (Dallas, TX, Dec. 1991), pp. 394–397.