CSPLib: Twenty Years On[⋆]

Ian P. Gent[1] and Toby Walsh[2]

[1] University of St Andrews
Scotland
Ian.Gent@st-andrews.ac.uk
[2] TU Berlin and UNSW Sydney
Germany and Australia
tw@cse.unsw.edu.au

Abstract. In 1999, we introduced CSPLib, a benchmark library for the constraints community. Our CP-1999 poster paper about CSPLib [8] discussed the advantages and disadvantages of building such a library. Unlike some other domains such as theorem proving, or machine learning, representation was then and remains today a major issue in the success or failure to solve problems. Benchmarks in CSPLib are therefore specified in natural language as this allows users to find good representations for themselves. The community responded positively and CSPLib has become a valuable resource but, as we discuss here, we cannot rest.

1 Introduction

Thirty years ago, new constraint satisfaction algorithms were often benchmarked on a small range of problems like the zebra problem (which is unrepresentative of many problems as it has an unique solution) or the n-queens problem (which is also unrepresentative of many problems as it has many solutions for large n), Over the next decade, experimental practice improved a little once it became the norm to benchmark on hard, random problems (see, for example, [19, 11, 17, 12, 10, 14, 1, 18, 9]. But random problems (whilst they can be hard to solve if we generate them at some satisfiability phase boundary) still don't represent many real world problems actually met in practice. There are structures common in real world problems that aren't met with any great frequency in simple random ensembles (see, for instance, [13, 21, 5, 22]).

We are not that surprised our paper about CSPLib in the CP-1999 proceedings [15] attracted a lot of citations subsequently. It has over citations 300 on Google Scholar today, putting it in the top three scientific papers either of us have ever written. This is not because of any great scientific merit or insights it contains. It simply reflects that our intuitions two decades ago were correct. The constraint programming community needed to benchmark on a larger and more representative class of problems. And hundreds of researchers have taken advantage of this benchmarking resource subsequently.

[⋆] An extended version of the original CP-1999 paper about CSPLib is available without pay-wall at https://tinyurl.com/CSPLibAt20

2 A Very Brief History of CSPLib

It seemed very natural to propose a benchmark library for constraints, following other areas of research such as Theorem Proving's TPTP [20] and Operations Research's OR-Lib [3]. Seeing the good and bad points of other libraries also helped inform our opinions of what a good library should look like. Indeed, one of us (Ian) had engaged in an argument in the Journal of Heuristics on the pitfalls of benchmark libraries [6, 4, 7]. Toby started leading discussions in 1998 on how best to set up CSPLib for example giving presentations on what we were thinking. The value of this can be seen by a change of mind following a comment from Barbara Smith in one of those sessions. We had previously thought we would come up with a formal language to express problems in, but Barbara argued that problems should be described in English. While this inevitably makes it harder to run the problems, modelling is such a critical part of constraint solving that any formal expression, no matter how high-level, may compromise finding the best model. Barbara persuaded us, except that we changed 'English' to 'Natural Language', to allow for submissions in other languages.

After deciding on the details we set up a simple website and included some initial problems. The next important phase was to publicise the library and also solicit contributions of problems. The site went public in 1999 and as part of this we decided to write a paper and submit it to CP 1999. As well as the paper submission we put the full version online as a technical report [15][3]. Strange as it may seem for what is now a citation classic, our paper was in fact *not* accepted in full. We saw the logic that our paper was perhaps not a significant research contribution. Instead, we were asked (and of course agreed) to produce a two page version for the proceedings and a poster for the conference itself. History does seem to have confirmed the editor's words that poster papers were intended 'to showcase yet more interesting directions' [16].

An important part of the history of CSPLib has been that the editors pass on the baton of the website to avoid stagnation. As well as new ideas, new editors of the website can have more enthusiasm and perhaps better contacts with new generations of researchers to encourage submissions from. Since foundation the leadership of CSPLib has changed twice. First, we handed over to Ian Miguel and Brahim Hnich, and later they in turn passed it to Chris Jefferson and Özgür Akgün (with some coding and maintenance help by Bilal Syed Hussain).

3 Design considerations

There are a number of desiderata for any benchmark library.

Easy to find: Just go to CSPLib.org. It's hard to imagine a simpler or more apt URL.

Large and diverse: the library needs to be diverse, so we don't over-fit. It needs to grow continuously, again to prevent over-fitting. And it needs to contain a mix of problems of varying difficulty: hard and easy problems, open and solved problems, and problems of a wide variety that .

[3] The technical report version is available at https://tinyurl.com/CSPLib-html

Easy to use: if there is one area in which CSPLib fails, it is this. You can't simply read in problems with a provided parser as you can with, say, SATLib or MIPLib. You have to understand the natural language description, work out a good model, and implement this yourself. But given that we didn't want to commit users to a particular representation, it's hard to imagine how we could have avoided this?

Ever growing: Benchmark libraries like CSPLib should always be growing. Unfortunately, the last problem submitted to CSPLib was about a year ago. Perhaps you can fix this?

4 An ongoing concern

The challenge that CSPLib always had (which we suspect is probably true of almost every other benchmark library too) is that, whilst people often use the benchmarks, few contribute to it. In part, this is structural. What credit do you get for going to the effort of submitting problems? This is not an easy problem to fix. Economists might perhaps suggest we need to design a market in which people had an incentive to submit. For example, the current maintainers introduced an incentive that might help. Each problem page in CSPLib provides bibtex to give credit to the proposer of each problem. For example, the largest numbered problem at time of writing is Problem 133, the Knapsack problem, which we hereby cite [2]. Another activity has been CSPLib hackathons at CP conferences, encouraging people to sit in a room together to write new entries.

Acknowledgments

We thank all those who helped us set up CSPLib, especially Barbara Smith and Bart Selman. We thank all subsequent editors of CSPLib, Ian Miguel, Brahim Hnich, Chris Jefferson and Özgür Akgün, and also Bilal Syed Hussain for his technical work on the site. Most importantly, we thank all the contributors to CSPLib over the past twenty years. Toby Walsh is supported by the European Research Council under the Horizon 2020 Programme via AMPLify 670077.

References

1. D. Achlioptas, L.M. Kirousis, E. Kranakis, D. Krizanc, M.S.O. Molloy, and Y.C. Stamatiou. Random constraint satisfaction: A more accurate picture. In G. Smolka, editor, *Proceedings of Third International Conference on Principles and Practice of Constraint Programming (CP97)*, pages 107–120. Springer, 1997.
2. Özgür Akgün. CSPLib problem 133: Knapsack problem. http://www.csplib.org/Problems/prob133.
3. John E Beasley. Or-library: distributing test problems by electronic mail. *Journal of the operational research society*, 41(11):1069–1072, 1990.
4. Emanuel Falkenauer. On method overfitting. *Journal of Heuristics*, 4(3):281–287, Sep 1998.
5. Ian Gent, Ewan MacIntyre, Patrick Prosser, Barbara Smith, and Toby Walsh. Random constraint satisfaction: Flaws and structure. *Constraints*, 6(4):345–372, 2001.

6. Ian P. Gent. Heuristic solution of open bin packing problems. *Journal of Heuristics*, 3(4):299–304, Mar 1998.
7. Ian P. Gent. A response to “on method overfitting”. *Journal of Heuristics*, 5(1):109–111, Apr 1999.
8. Ian P. Gent and Toby Walsh. Csp$_{\text{lib}}$: A benchmark library for constraints. In Jaffar [16], pages 480–481.
9. I.P. Gent, H. Hoos, P. Prosser, and T. Walsh. Morphing: Combining structure and randomness. In *Proceedings of the 16th National Conference on AI*. Association for Advancement of Artificial Intelligence, 1999.
10. I.P. Gent, E. MacIntyre, P. Prosser, and T. Walsh. Scaling effects in the CSP phase transition. In *1st International Conference on Principles and Practices of Constraint Programming (CP-95)*, pages 70–87. Springer-Verlag, 1995.
11. I.P. Gent and T. Walsh. The SAT phase transition. In A G Cohn, editor, *Proceedings of 11th ECAI*, pages 105–109. John Wiley & Sons, 1994.
12. I.P. Gent and T. Walsh. The SAT phase transition. In *Proc. of the 11th European Conference on Artificial Intelligence (ECAI-94)*, pages 105–109, 1994.
13. I.P. Gent and T. Walsh. Phase transitions from real computational problems. In *Proceedings of the 8th International Symposium on Artificial Intelligence*, pages 356–364, 1995.
14. I.P. Gent and T. Walsh. The satisfiability constraint gap. *Artificial Intelligence*, 81(1–2), 1996.
15. I.P. Gent and T. Walsh. CSPLib: a benchmark library for constraints. Technical report, Technical report APES-09-1999, 1999. A shorter version appears in the Proceedings of the 5th International Conference on Principles and Practices of Constraint Programming (CP-99).
16. Joxan Jaffar, editor. *Principles and Practice of Constraint Programming - CP’99, 5th International Conference, Alexandria, Virginia, USA, October 11-14, 1999, Proceedings*, volume 1713 of *Lecture Notes in Computer Science*. Springer, 1999.
17. S. Kirkpatrick and B. Selman. Criticial behaviour in the satisfiability of random Boolean expressions. *Science*, 264:1297–1301, 1994.
18. E. MacIntyre, P. Prosser, B.M. Smith, and T. Walsh. Random constraint satisfaction: Theory meets practice. In *4th International Conference on Principles and Practices of Constraint Programming (CP-98)*, pages 325–339. Springer, 1998.
19. D. Mitchell, B. Selman, and H. Levesque. Hard and Easy Distributions of SAT Problems. In *Proceedings of the 10th National Conference on AI*, pages 459–465. Association for Advancement of Artificial Intelligence, 1992.
20. G. Sutcliffe. The TPTP World - Infrastructure for Automated Reasoning. In E. Clarke and A. Voronkov, editors, *Proceedings of the 16th International Conference on Logic for Programming Artificial Intelligence and Reasoning*, number 6355 in Lecture Notes in Artificial Intelligence, pages 1–12. Springer-Verlag, 2010.
21. T. Walsh. Search in a small world. In *Proceedings of 16th IJCAI*. International Joint Conference on Artificial Intelligence, 1999.
22. T. Walsh. Search on high degree graphs. In *Proceedings of 17th IJCAI*. International Joint Conference on Artificial Intelligence, 2001.