Declarative Local-Search Methods *

Gustav Björdal, Pierre Flener, and Justin Pearson

Uppsala University, Department of Information Technology, Uppsala, Sweden
{Gustav.Bjordal,Pierre.Flener,Justin.Pearson}@it.uu.se

Abstract The main focus of my PhD topic is combining MiniZinc, local search, and constraint programming into what we call declarative local-search. Towards achieving this, we have identified and addressed weaknesses in current local-search backends, introduced a declarative neighbourhood syntax to MiniZinc, and most recently shown how a constraint programming solver can utilise declarative neighbourhoods. This summary will briefly discuss each of these contributions.

Keywords: MiniZinc · Local search · Constraint programming · Declarative neighbourhoods

1 Overview

We have previous developed a constraint-based local search (CBLS) [9] backend to MiniZinc [5] called fzn-oscar-cbls [4], which is the starting point for my PhD topic. Given any MiniZinc model, which does not contain any local-search-specific information, fzn-oscar-cbls synthesizes a local-search strategy (including neighbourhood, heuristic, and meta-heuristic) from the MiniZinc model. The backend competes in the annual MiniZinc Challenge [8], where one can see that while fzn-oscar-cbls[1] is sometimes competitive (even winning on some problems) it ranks low overall.

The focus of my PhD topic is to understand the challenges of using local search to solve problems stated in a declarative and technology independent language, such as MiniZinc. Furthermore, I also aim at making local search more accessible via MiniZinc. Towards achieving this we have identified how variable structure in a model can negatively effect the performance of local search [1], extended the MiniZinc syntax and compiler such that one can prescribe neighbourhoods declaratively in MiniZinc [3], and most recently shown how constraint programming (CP) backends to MiniZinc can perform local search using our declarative neighbourhoods [2]. What follows is a brief overview of each of these contributions.

* This work is supported by the Swedish Research Council (VR) through Project Grant 2015-04910.

[1] Competing under the name OscaR/CBLS.

2 Generating Compound Moves by Hybridising with Constraint Programming

A local-search backend must automatically infer a combination of a neighbourhood, heuristic, and meta-heuristic from the structure of a declarative model, as these are normally not part of a declarative model. However, such a backend can perform poorly when the model has a structure that is inappropriate for local search, for example when the resulting neighbourhood considers moves modifying only variables that represent auxiliary information. This can be observed in the MiniZinc Challenge results from 2015 to 2018 where fzn-oscar-cbls and Yuck[2] consistently under-performs on most routing problems with time windows or capacity constraints, compared to what can be expected from specialised local-search algorithms.

Towards overcoming this particular inefficiency, which we show is due to the presence of auxiliary variables that are not functionally defined by constraints, we propose compound-move generation (CMG) [1]. This extension to local-search solvers uses a complete-search solver in order to augment moves. We suggest several refinements of CMG and show its positive impact on several third-party models. We also propose two methods for identifying such auxiliary variables in a model, so that CMG can be turned on by a local-search backend when needed.

3 Declarative Neighbourhoods in MiniZinc

The aim of technology-independent modelling is to create a model that is independent of a particular solver or technology for a problem. This avoids early commitment to a solving technology and allows easy comparison of technologies. MiniZinc is a technology-independent modelling language, supported by CP, MIP, SAT, SMT, and constraint-based local-search backends. However, some technologies, in particular CP and CBLS, require not only a model but also a search strategy in order to be effective. While backends for these technologies offer some default search strategies, it is often beneficial to include in a model a user-specified search strategy, especially if the strategy can encapsulate knowledge about the problem structure. MiniZinc has provided a CP-specific search language since it conception, but has not offered any way of specifying (parts of) a search strategy for local search. Doing so is complex since a local-search strategy (comprising a neighbourhood, a heuristic, and a meta-heuristic) is often tightly tied to the model. We therefore wanted to use the same language for specifying the model and the local-search strategy.

We show how to extend both the MiniZinc syntax and compiler so that one can attach a fully declarative neighbourhood specification to a model, while maintaining the technology-independence of the language [3]. We explain how to support these declarative neighbourhood in a CBLS backend, fzn-oscar-cbls in particular, by integrating their support with the automatic synthesizes of

[2] https://github.com/informarte/yuck

the heuristic and meta-heuristic. We therefore now allow users of MiniZinc and fzn-oscar-cbls to solver MiniZinc models with either no local-search strategy specified or with a specified declarative neighbourhood. Specifying the heuristic and meta-heuristic in a declarative manner in MiniZinc remains future work.

4 Exploring Declarative Neighbourhoods with Constraint Programming

Using constraint programming to explore a local-search neighbourhood was first tried in [6] and improved in [7]. The advantage of doing so is that constraint propagation can quickly rule out uninteresting neighbours, sometimes greatly reducing the number of actually probed neighbours. However, in both [6] and [7], a CP model of the neighbourhood has to be handcrafted from the model of the problem: this can be difficult and tedious. Perhaps for this reason, that research direction appears abandoned as large-neighbourhood search and constraint-based local search arose as alternatives that seem easier to use. In order for our declarative neighbourhoods, discussed in Section 3, to be technology independent and consistent with the output language of the MiniZinc compiler, the declarative neighbourhoods are essentially compiled into constraint satisfaction problems that describe the neighbourhood. This allow the declarative neighbourhood to be used by CBLS backends to MiniZinc.

We demonstrate in [2] that, by using the old ideas of [6], declarative neighbourhoods can also be used by CP backends to perform local search. Specifically, we introduce a new global constraint, Writes, and show how to encode automatically (rather than by hand as in [6] and [7]) a declarative neighbourhood into a CP model of the neighbourhood. This confirms that our declarative neighbourhood are technology independent. Our prototype is competitive with CP, CBLS, and LNS backends to MiniZinc. Interestingly, it turns out that local search performed in this way, by a CP solver, does not suffer from the presence of auxiliary variables (discussed in Section 2) in the same way that the fzn-oscar-cbls and Yuck backends do.

References

1. Björdal, G., Flener, P., Pearson, J.: Generating compound moves in local search by hybridisation with complete search. In: Rousseau, L.M., Stergiou, K. (eds.) CP-AI-OR 2019. LNCS, vol. 11494, pp. 95–111. Springer (2019)
2. Björdal, G., Flener, P., Pearson, J., Stuckey, P.J.: Exploring declarative local-search neighbourhoods with constraint programming. In: CP 2019. LNCS
3. Björdal, G., Flener, P., Pearson, J., Stuckey, P.J., Tack, G.: Declarative local-search neighbourhoods in MiniZinc. In: Alamaniotis, M., Lagniez, J.M., Lallouet, A. (eds.) ICTAI 2018. pp. 98–105. IEEE Computer Society (2018)
4. Björdal, G., Monette, J.N., Flener, P., Pearson, J.: A constraint-based local search backend for MiniZinc. Constraints **20**(3), 325–345 (July 2015), the fzn-oscar-cbls backend is available at http://optimisation.research.it.uu.se/software

5. Nethercote, N., Stuckey, P.J., Becket, R., Brand, S., Duck, G.J., Tack, G.: MiniZinc: Towards a standard CP modelling language. In: Bessière, C. (ed.) CP 2007. LNCS, vol. 4741, pp. 529–543. Springer (2007), the MiniZinc toolchain is available at https://www.minizinc.org
6. Pesant, G., Gendreau, M.: A constraint programming framework for local search methods. Journal of Heuristics **5**(3), 255–279 (October 1999), extends a preliminary version at CP 1996, LNCS, vol. 1118, pp. 353–366, Springer (1996)
7. Shaw, P., De Backer, B., Furnon, V.: Improved local search for CP toolkits. Annals of Operations Research **115**(1–4), 31–50 (September 2002)
8. Stuckey, P.J., Feydy, T., Schutt, A., Tack, G., Fischer, J.: The MiniZinc Challenge 2008–2013. AI Magazine **35**(2), 55–60 (summer 2014), see https://www.minizinc.org/challenge.html
9. Van Hentenryck, P., Michel, L.: Constraint-Based Local Search. The MIT Press (2005)