Design Space Exploration with Answer Set Programming Modulo Theories

Philipp Wanko

Philipp Wanko, University of Potsdam, wanko@cs.uni-potsdam.de

With increasing demands for functionality, performance, and energy consumption in both industrial and private environments, the development of corresponding embedded processing systems is becoming more and more intricate. Also, desired properties are conflicting and compromises have to be found from a vast number of options to decide the most viable design alternatives. Hence, effective Design Space Exploration (DSE; [8]) is imperative to create modern embedded systems with desirable properties; it aims to find a representative set of optimal valid solutions to a design problem helping the designer to identify the best possible options.

Until now, mostly, meta-heuristic algorithms were used to perform DSE, but they do not guarantee optimality and are ineffective for finding feasible designs in highly constrained environments. ASP-based solutions alleviate this problem and have shown themselves to be for system synthesis [2]. Also, recent developments in ASP solving allow for a tight integration of background theories covering all (numeric) constraints occurring in DSE. This enables partial solution checking to quickly identify infeasible or suboptimal areas of the design space.

While DSE can be done at various abstraction levels, the overall goal is to identify optimal implementations given a set of applications and a hardware platform. Our work targets streaming applications (such as video decoders) and heterogeneous hardware platforms organized as networks on chip (such as many-core SoCs) described at the electronic system level. Here, applications are defined as task-level descriptions and hardware platforms comprise networks of processing and memory elements. The DSE problem is twofold: first, evaluate a single feasible implementation, called a *design point*, and second, cover multiple (optimal) design points of the design space during exploration.

Obtaining a feasible implementation given a hardware platform and a set of applications is typically divided into three steps: *binding*, *routing*, and *scheduling*. Binding describes the process of allocating a resource for a specific task, routing ensures that messages of communicating tasks are correctly delivered through the network, and scheduling assigns starting points for all tasks and communications so that no conflicts occur while executing applications.

By assigning worst-case execution times to tasks, as well as energy consumption and costs to resources, we are able to evaluate several quality criteria of a design point. We mainly focused on *latency*, *energy consumption* and *hardware cost*. These quality criteria are aggregated via a *Pareto* preference, i.e., a design point is better if it is at least as good in all criteria and strictly better in at least one compared to another design point. Note that this preference might lead to a vast amount of optimal solutions since design points may be incomparable.

Until now, the focus of our work lay on developing exact and flexible methods using ASP technology for finding design points for complex system models, obtaining optimal design points, and enumerating and storing optimal design points. In detail, in [6], we propose a novel ASPmT system synthesis approach. It supports more sophisticated system models, and makes use of tightly integrated background theories and partial solution checking. We present a comprehensive ASPmT encoding of all aspects of system synthesis, i.e., binding, routing, scheduling. As underlying technology, we use the ASP system *clingo* [3] whose grounding and solving components allow for incorporating application- or theory-specific reasoning into ASP. Furthermore, in [4], we instantiate the theory framework of *clingo* with linear constraints over reals and integers. The resulting systems *clingo[DL]*, *clingo[LP]* and *clingcon* handle the background theories of difference logic, integer linear programming and constraint programming, respectively. In [7], we present a novel approach to a holistic system level DSE based on ASPmT. DSE including feasibility check and optimization is performed directly within the solving process. To achieve that, we include additional background theories that concurrently guarantee compliance with hard constraints and perform the simultaneous optimization of several design objectives. Binding, routing, scheduling and design objectives are represented in a declarative fashion in one encoding. Finally, we address comparability of different DSE techniques in [5] by proposing a methodology for test case generation and presenting a versatile and easily expendable benchmark generator based on ASP that is able to produce hard synthesis problem instances.

References

1. Carro, M., King, A. (eds.): Technical Communications of the Thirty-second International Conference on Logic Programming (ICLP'16), vol. 52. OASIcs (2016)
2. Biewer, A., Andres, B., Gladigau, J., Schaub, T., Haubelt, C.: A symbolic system synthesis approach for hard real-time systems based on coordinated SMT-solving. In: Proceedings of Design, Automation and Test in Europe (DATE'15) (2015).
3. Gebser, M., Kaminski, R., Kaufmann, B., Ostrowski, M., Schaub, T., Wanko, P.: Theory solving made easy with clingo 5. In: [1], pp. 2:1–2:15
4. Janhunen, T., Kaminski, R., Ostrowski, M., Schaub, T., Schellhorn, S., Wanko, P.: Clingo goes linear constraints over reals and integers. TPLP **17**(5-6), 872–888 (2017)
5. Neubauer, K., Haubelt, C., Wanko, P., Schaub, T.: Systematic test case instance generation for the assessment of system-level design space exploration approaches. In: Proceedings of MBMV'18 (2018).
6. Neubauer, K., Wanko, P., Schaub, T., Haubelt, C.: Enhancing symbolic system synthesis through ASPmT with partial assignment evaluation. In: Proceedings of the 20th Conference on Design, Automation and Test in Europe (DATE'17) (2017)
7. Neubauer, K., Wanko, P., Schaub, T., Haubelt, C.: Exact multi-objective design space exploration using ASPmT. In: Proceedings of Design, Automation and Test in Europe (DATE'18) (2018).
8. Pimentel, A.: Exploring exploration: A tutorial introduction to embedded systems design space exploration. IEEE Design & Test **34**(1), 77–90 (2017)
9. Romero, J., Schaub, T., Wanko, P.: Computing diverse optimal stable models. In: [1], pp. 3:1–3:14