“Lucian Blaga” University of Sibiu
Engineering Faculty, Computer Science and Electrical Engineering Department

Developing Effective Multi-Objective Optimization Methods for Complex Computing Systems

- Summary -

PhD Thesis

Author:
Radu Chiş, M.Sc.

PhD Supervisor:
Professor Lucian N. Vinţan, PhD, Full-member of the Academy of Technical Sciences of Romania

SIBIU, September 2017

Contents

"There is no elevator to success. You have to take the stairs."

Zig Ziglar

Sumamry

During the last decade, computing systems have advanced at a much faster pace than software developments, becoming increasingly complex [1], [2], [3], [4], [5]. Systems designers and architects must cope with the huge design space of millions of billions of different configurations, created by the many different parameters that can be varied. Before a new microprocessor configuration can be released, a software simulator that accurately models the CPU is implemented, where changes of the input parameters are easily operated. The simulations of the configurations must pass the time-consuming evaluations of different objectives like: performance, temperature, power consumption, integration area, costs, etc. The benchmarks on which the different configurations are evaluated can require several days to be run, so that an exhaustive search of the entire search space is not feasible.

There are a number of automatic design space exploration (ADSE) frameworks available, that could help the designers find the best configurations in respect to the desired output objectives. These frameworks use heuristic search and optimization algorithms to solve NP-hard problems. With the increase in complexity and number of cores, the problem of finding the best solutions is becoming increasingly difficult for the ADSE frameworks. Improvements and optimizations of these frameworks are needed for the meta-heuristic search algorithms, but also for the simulators running them.

The **scope** of this PhD thesis is to use our FADSE tool [6] (Framework for Automatic Design Space Exploration), developed and implemented in Professor Lucian Vinţan's research group, to develop effective multi-objective optimization methods for complex computing systems. For this we have to fulfill the following **objectives**:

- analyse the state-of-the-art design space exploration methods, including multi-objective meta-heuristics and performance metrics;
- analyse the state-of-the-art automatic design space exploration frameworks and compare them to our FADSE tool;
- extend and improve FADSE so that it can be used with more simulators (microprocessor or other types, like complex magnetic actuators) by incorporating some new checkpointing mechanisms, output constraints and floating-point parameters;
- integrate some other optimization meta-heuristics and compare them to the already implemented algorithms;
- integrate some new metrics that could better evaluate the quality of the generated solutions;
- integrate a state-of-the-art multi-core x86 simulator with performance, area, power and temperature outputs, to run a complex real-life simulation process;

- improve the search and optimization algorithms in FADSE by running two or more algorithms in parallel (meta-optimization), but in an amount of time equal to the one needed for the run of one single algorithm;
- improve the simulation time in FADSE by using either faster simulators (not always possible) or encapsulating machine learning techniques that would decrease the simulation time or bypass the simulations entirely;
- integrate some other simulators in order to show the versatility of the FADSE tool.

Chapter 2 presents some well-known design space exploration methods that we used in the search and optimization processes in the next chapters. We focused on multi-objective meta-heuristics, Pareto efficiency and multi-objective performance evaluation metrics.

In Chapter 3 we describe the Framework for Automatic Design Space Exploration (FADSE), its main features and our implemented improvements which will be used in our next optimization processes and include: a specific checkpointing mechanism for SMPSO bio-inspired optimization algorithm, output constraints and floating-point parameters. We also present our implementation of some meta-heuristics, CNSGA-II and MOCHC, and a new performance evaluation metric: the two set hypervolume difference. Finally, we implemented a dynamic distribution of the simulations to the clients, with important benefits related to the optimization time.

Chapter 4 presents an optimization process on the Grid ALU Processor (GAP) [7], where we compared our newly implemented meta-heuristics from Chapter 3 using some well-known performance metrics. In this chapter, we ran multiple meta-heuristics from the same initial random population to examine which one generates the best solutions. For this we ran some genetic multi-objective algorithms: NSGA-II and SPEA2, and our implementations of CNSGA-II (in multiple versions) and MOCHC, but also a bio-inspired meta-heuristic called SMPSO. We compared the results using some well-known quality metrics: coverage, hypervolume and our implementation of the two set hypervolume difference. We found that at the end of the optimization process SMPSO produces the best quality individuals, NSGA-II is better than SPEA2 from the individuals' quality point of view, and that MOCHC algorithm has the fastest convergence speed. In the end we also tried to use some feature selection methods to decrease the search space from around 1 million configurations to about 17.000 configurations. This has led to a decrease in simulation time but also in solutions' quality.

Chapter 5 introduces a new state-of-the-art multi-core/many-core simulator, Sniper, where we ran a hardware and software co-optimization with three objectives: performance, area and energy. The hardware complexity improvements made possible by the decrease in transistor sizes should be supplemented by improvements in the software, but this is not always the case. In this research, we wanted to see if the overhead in optimizing also the software parameters pays off. We started with a manual DSE process to evaluate the impact of the software parameters (gcc optimization flags, thread number, scheduler) on performance and energy. We found that some parameters have an important impact on the objectives. After this, we ran the automatic optimization process using FADSE for two runs: one with the variation of only hardware parameters and one where we changed both hardware and software parameters (HW-SW optimization approach). Finally, we compared the two runs and concluded that the HW-

SW optimization run yields better results and the co-optimization should be the design choice when creating new advanced microprocessor configurations.

Chapter 6 further improves the DSE process on the Sniper multi-core simulator by adding a fourth objective: temperature. This represents nowadays the hard limit that CPUs cannot cross: at around 100 degrees Celsius the life of the CPU shortens and above 115 degrees physical damage occurs. We improved the Sniper simulator by adding some scripts that provide HotSpot, the thermal modeling software, with power traces and functional unit areas in order to compute temperatures for each unit. One challenge we overcame was the automatic creation of the CPU floorplan in order to accurately compute the areas and positions of the CPU components. Once we overcame this problem, we were able to run an automatic DSE process with 4 objectives (4-dimensions, 4-D). We are, as far as we know, the first to run an automatic 4-D optimization that includes the temperature using the Sniper multi-core simulator. We compared our results to a run that would compute the temperature objective only afterwards, and concluded that the 4-D run provides better results.

Chapter 7 introduces a new simulator from a totally different domain: a MATLAB with COMSOL model of a magnetic actuator provided by the Continental Automotive Systems Branch in Sibiu. This was a new type of software simulator for which we had to write a new connector and use the output constraints and floating-point parameters presented in Chapter 3. This optimization run posed some new challenges because it had 8 output objectives that had to be optimized, some of them with constraints. FADSE tool coped with the large number of outputs and provided good results in about a week. We then used some machine learning techniques to try to reduce the number of outputs, removing dependent outputs and also use Response Surface Methodology (RSM) techniques to decrease the simulation time. We managed to train some artificial neural networks (ANNs) and get normalized root mean square errors (NRMSE) of less than 1%. We then ran a new DSE process where, instead of simulating the actuator configurations, we predicted the outputs using the ANN and got NRMSE of around 2%. We concluded that in this case the decrease in simulation time was worth the 2% NRMSE.

Chapter 8 presents a major original improvement to the FADSE tool: a meta-optimization layer, that can run multiple meta-heuristic algorithms in parallel, in the time needed to run a single algorithm. We ran the proof-of-concept DSE search of our meta-optimization on the GAP simulator because the evaluations on one configuration take some minutes, not hours. We first implemented a naïve approach, which incorporated a random selection of the next generation. The second approach is more elaborated and each algorithm dynamically generates a percentage of offsprings. The percentage is determined by the solutions quality of the individuals generated in the previous generation by each algorithm. This way algorithms that generate better individuals will have a higher percentage of offsprings created. We analyzed the results and detected a synergy inside our new meta-optimization approach, because it yielded better results than a super-positioned run of two meta-heuristics with two times more individuals and two times the simulation time.

The theory presented in this work (classifications, definitions, etc.) is strictly used in the scope and objectives of this PhD thesis and does not claim a general valid text-book rigor and comprehensiveness.