WHITEPAPER

SLX Multi-Objective Optimization (MOPT)

Simultaneous optimization to improve power and performance with SLX

SILEXICA

Table of Contents

1 Introduction

Technologies such as autonomous cars and 5G communication are seeing a rapid increase in the number of processing elements (PE) per platform. Where software professionals were used to programming one, two or a handful of cores, the game has now changed. Intel's Many Integrated Core Architecture [3] contains up to 78 cores, Nvidia Tegra XI[2] has up to 260 cores and Adapteva's Epiphany-V[1] has 1024 cores. Future systems will utilize platforms with thousands of cores and this poses many new challenges. Understanding your system has never been more important.

Mapping a parallel application on a multi-processor system-on-chip (MPSoC) platform refers to assigning different tasks of a parallel application onto PEs. Mapping is typically performed for a certain mapping objective, such as meeting deadlines, improving performance, reducing energy/power consumption and optimizing memory throughput. Objective-based mapping is considered to be an NP-complete problem[4]. This implies that it is unlikely for an approach to find an optimal mapping solution in polynomial time. Furthermore, the mapping search space (MSS) grows exponentially with MPSoC platform and/or parallel application complexity increase. Therefore, different mapping algorithms and heuristics have been developed to explore the MSS to find a procedure which can fulfil the required mapping objective criteria.

More and more applications require multiple objectives to be optimized simultaneously. This is referred to as multi-objective optimization. Typically, these objectives can be contradictory in nature and potentially lead to multiple solutions with varying results for the optimization goals. In this case, multi-objective optimization can produce a set of optimal solutions that represent a trade-off between the different objective values. The solution set is called Pareto optimal set or just Pareto solutions.

One approach to find the Pareto optimal solution is to analyze every possible mapping solution present in the MSS. For example, by using exhaustive search or brute force search techniques. This is usually not feasible because of long exploration times or resource requirements. Alternatively, multi-objective evolutionary algorithms (MOEA) are used. MOEAs are meta-heuristics that can provide a near optimal solution set in significantly reduced exploration time[6].

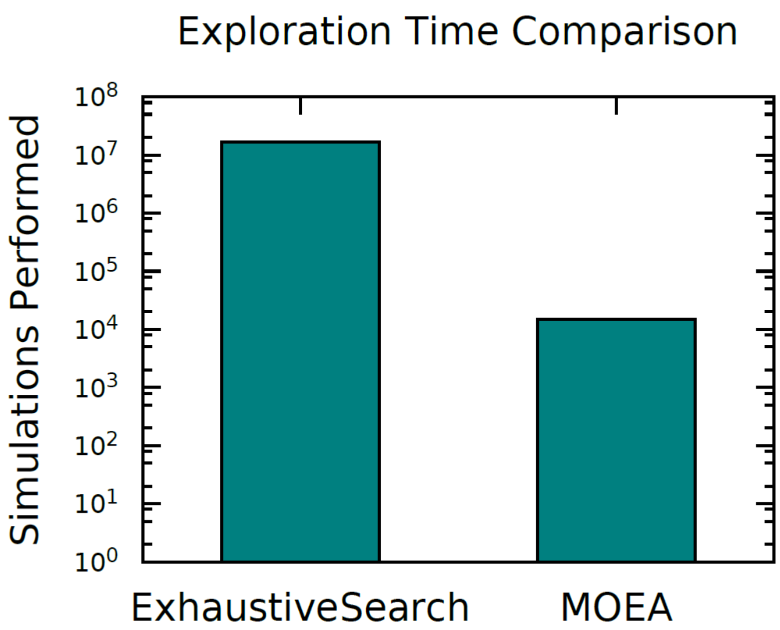


Figure 1.1 Number of simulations performed for MPSoC mapping exploration by an exhaustive search in contrast to a typical MOEA for an application with eight tasks and a platform with eight cores and ten different power configurations.

Figure 1.1 displays the number of mapping combinations evaluated by an exhaustive search and a typical MOEA to find a solution set. This is when mapping a parallel application with eight tasks onto a MPSoC with eight PEs and 10 platform power configurations. If the simulation time required for analyzing a mapping solution for this example is 2 seconds, then an exhaustive search approach would take approximate 5.30 years (sequential execution), while the MOEA would take about eight hours to complete the exploration. Correspondingly, for larger parallel applications and complex platforms, the exploration times increase exponentially.

2 SLX Multi-Objective Mapping Heuristic

Although, MOEAs are faster than exhaustive searching, exploration times can still be significant. Especially when the search of optimal configuration parameters of an MOEA is also considered. Therefore, Silexica has developed **SLX** MOPT - a multi-objective optimization heuristic that is available in the latest desktop release of **SLX**. It is an intermediate MSS exploration approach, that provides a good Pareto optimal set approximation in a far shorter time in comparison to MOEAs and exhaustive searches.

The **SLX** MOPT heuristic is an iterative approach to find a Pareto solution set for the mapping objectives power and execution time. It uses application profiling information, target platform information and performance estimates to automatically classify based on power and performance. The classification information significantly decreases the overall mapping search space by pruning certain target platform configurations that might yield sub-optimal solutions. The **SLX** MOPT heuristic iteratively configures the target platform to one of the optimal power configurations determined by the qualifier, as shown in **Figure 2.1**. Then it maps it automatically using a specifically selected mapping algorithm from one of those available in the **SLX** Dataflow Solution.

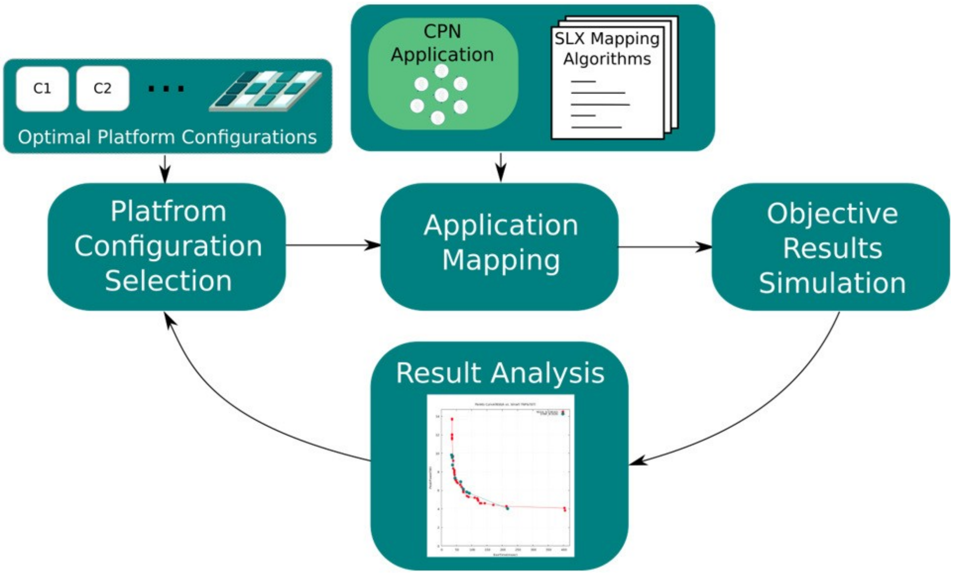


Figure 2.1 Reduction of the MSS by the SLX MOPT heuristic qualifier

The configuration of the **SLX** MOPT requires only a single tuneable parameter i.e. number of iterations to perform. This allows the user to determine the trade-off between accuracy and exploration time. This unique **SLX** feature is in contrast to other algorithms that require a detailed insight of the application, target platform and internal working of the algorithm to optimally configure for convergence.

Additionally, in the March 2018 release of **SLX**, MOPT heuristics have been improved to further speed-up its exploration, which significantly reduces the total exploration time required.

3 SLX MOPT vs. MOEA

To present a greater insight into the capabilities of the **SLX** MOPT heuristic, a comparison is provided to an implementation of a well-known MOEA, called “Non-dominated sorting genetic algorithm II" (NSGA-II)[5]. The comparison criteria is coverage of the objective space by the generated Pareto solution set and the exploration time needed to find this set. Furthermore, other features like the quality of the generated Pareto curve are also compared. The details of the features considered for comparing the **SLX** MOPT heuristic with the NSAGA-II are listed below.

3.1 Exploration Time

The exploration time required to find the Pareto set is of great importance, as shown in Figure 1.1. Although MOEAs provide near optimum solutions, they typically require very long exploration times, which significantly affect the delivery time of a product. Therefore, a favorable feature of an exploration approach would be to yield a Pareto solution faster than with MOEAs and other brute force techniques. By using **SLX** for a rapid and comprehensive overview of a system during the design phase, software professionals can make informed decisions in hours rather than months - which can only be beneficial in a project.

3.2 Coverage

The coverage determines the optimality of the resulting Pareto solution set generated by an exploration technique. It is computed by using an approach that calculates the area under the Pareto curve called objective space non-domination (OSN). As shown in **Figure 3.1** the OSN metric evaluates the range of the objective space not dominated by the solution set. A large OSN implies that only a small area of the objective space is dominated by the solution set. The exploration technique that yields a solution set with a smaller OSN is considered better and closer to the optimal Pareto solution set.

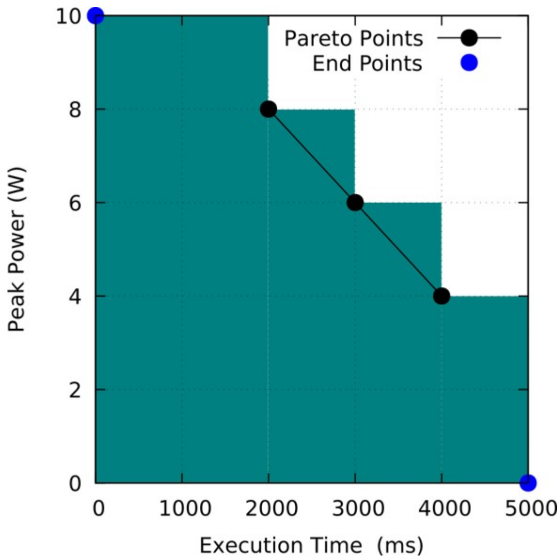


Figure 3.1 Three pareto solution points for the objectives: Peak Power and Execution time that exhibit an OSN value of 0.76 for a normalized MSS

3.3 Evaluation

The multi-objective optimization techniques were tested for a set of KPN specific applications. In the **SLX** Dataflow framework, C *for process network* (CPN) is used to implement KPN applications in ANSI-C. The framework contains multiple benchmarks along with multiple models of MPSoC platforms that are used for comparing the **SLX** MOPT heuristics with the NSGA-II algorithm.

The NSGA-II was configured with two termination criteria: 1) generate a maximum of 200 generations 2) an OSN based criterion comparing results between successive generations. Furthermore, a parallel version of NSGA-II was implemented that could perform multiple parallel selection, crossover and mutation jobs within creation of a generation.

The **SLX** MOPT was configured to run for a fixed number of iterations (30), whereas the parallel NSGA-II implementation ran until one of its termination criteria was met. Furthermore, the NSGA-II and the **SLX** MOPT heuristics were configured to optimize for two objectives simultaneously: Peak Power vs. Execution Time and Average Power vs. Execution Time.

3.3.1 Host System Setup

Architecture	**x86_64**
Byte Order	**Little Endian**
CPU(s)	**6 cores with 2 threads each**
Model name	**Intel(R) Core(TM) i7-6800K CPU @ 3.40GHz**
L1d cache	**32K**
L1i cache	**32K**
L2 cache	**256K**
L3 cache	**15360K**
Memory	**64 GB**
OS	**Ubuntu 1604 amd64**

Table 3.1 Details of the host system, where the SLX MOPT and NSGA-II were evaluated

Both approaches are tested on an x86-64-based host (see **Table 3.1**). A sequential implementation of the **SLX** MOPT heuristics is performed within context of this white paper, it utilizes a single thread for its exploration, whereas the NSGA-II implementation has been extended to utilize all possible available threads to perform its explorations.

3.3.2 Results

Table 3.2 and **3.3** list the minimum, maximum and the average of all the exploration results produced with both approaches for all benchmarks and platforms.

Figure 3.2 summarizes these results using separate box plots for the individual comparison criteria. **Figure 3.2a** and **3.2b** illustrate the overall result for OSN and the exploration time, respectively. The NSGA-II dominates on average 7.60 % more of the objective space than **SLX** MOPT. However, the **SLX** MOPT performs its exploration 11x faster than the NSGA-II.

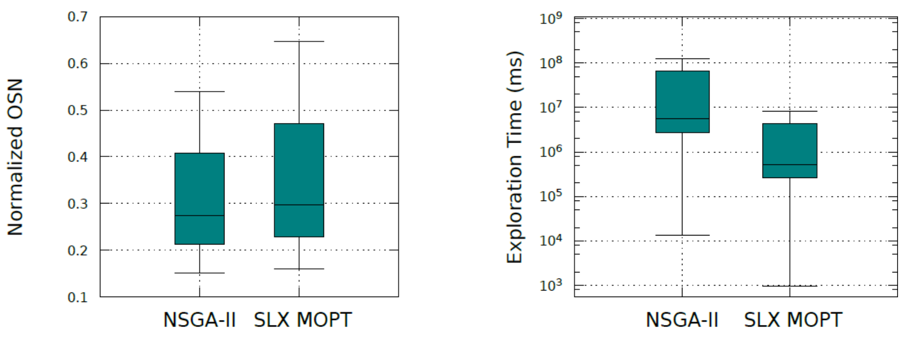
Similarly, the **Figure 3.3** summarizes the results for the objective: average power vs. execution time. **Figure 3.3a** and **3.3b** illustrate the overall result for OSN and the exploration time. Similar to the OSN result in **Figure 3.2a**, the greatest difference between the maximum and minimum nominal values is 0.05, which indicates that the disparity between OSN results is not high. The **SLX** MOPT performs the exploration 8.7x faster than NSGA-II, whereas NSGA-II dominates 8.70 % more of the objective space.

NSGA-II	Coverage	Exploration Time(ms)
Min	0.15	13 693
Mean	0.27	5 535 587
Max	0.54	125 837 738

Table 3.2 Min, max and average values of all comparison criteria for the objectives:
peak power vs. execution time.

SLX MOPT	Coverage	Exploration Time(ms)
Min	0.16	960
Mean	0.30	518 455
Max	0.65	8 260 443

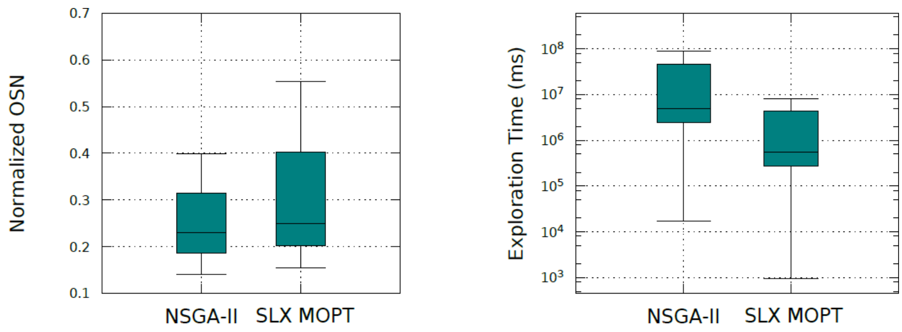
Table 3.3 Min, max and average values of all comparison criteria for the objectives:
average power vs. execution time.



(a) Objective Space Non-Domination

(b) Execution Time

Figure 3.2 Summary of all results for the optimization objectives:
peak power vs. execution time for the comparison criteria



(a) Objective Space Non-Domination

(b) Execution Time

Figure 3.3 Summary of all results for the optimization objectives:
peak power vs. execution time for the comparison criteria

4 Summary

The headline figures from this white paper are that the **SLX** MOPT performs the exploration 8.7x faster than NSGA-II, whereas NSGA-II dominates 8.70 % more of the objective space. Even though in some cases **SLX** MOPT does not determine better results than the MOEA, it can be argued that its superior exploration time makes it a valid alternative to perform design space exploration. Given the new heterogeneous multicore era, this can be a difference for software professionals of between months and minutes. This is especially important, when considering that the OSN is only worse by 8 %.

By using **SLX** for fast exploration, the benefits throughout a project lifecycle can prove invaluable. MOPT is now available in the C/C++, FPGA and Automotive solutions. For further information, for a free trial of **SLX** or for a product demonstration, please contact Silexica at info@silexica.com or visit our website for more multicore heterogeneous programming solutions: www.silexica.com

Bibliography

[1] Epiphany-v: A 1024-core 64-bit risc processor. https://www.parallella.org/ 2016/10/05/epiphany-v-a-1024-core-64-bit-risc-processor/. Accessed: 2017-08-14.

[2] NVIDIA TEGRA X1. http://www.nvidia.com/object/tegra-x1-processor.html. Accessed: 2017-08-11.

[3] A. Duran and M. Klemm. The Intel® many integrated core architecture. In *High Performance Computing and Simulation (HPCS), 2012 International Conference on*, pages 365–366. IEEE, 2012.

[4] W. Quan and A. D. Pimentel. An iterative multi-application mapping algorithm for heterogeneous MPSoCs. In *The 11th IEEE Symposium on Embedded Systems for Real-time Multimedia*, pages 115–124, Oct 2013.

[5] N. Srinivas and K. Deb. Muiltiobjective Optimization Using Nondominated Sorting in Genetic Algorithms. *Evolutionary Computation*, 2(3):221–248, Sept 1994.

[6] L. Thiele, I. Bacivarov, W. Haid, and K. Huang. Mapping Applications to Tiled Multi-processor Embedded Systems. In *Seventh International Conference on Application of Concurrency to System Design (ACSD 2007)*, pages 29–40, July 2007.