An Evolutionary Approach for Pareto-optimal Configurations in SOC Platforms

Giuseppe Ascia, Vincenzo Catania, Maurizio Palesi
Dipartimento di Ingegneria Informatica e delle Telecomunicazioni, Università di Catania, Italy

Abstract: One of the most important problems in SOC platforms design is that of defining strategies for tuning the parameters of a parameterized system so as to obtain the Pareto-optimal set of configurations that provide multi-criteria optimisation. The paper proposes a methodology based on evolutionary techniques for exploration of the range of possible configurations of a parameterized system [1]. A highly parametric system-on-a-chip for digital camera applications will be taken as a case study and a multi-objective genetic algorithm will be used to search for the power-performance trade-off surface. The methodology proposed will be compared with that implemented in Platune [2] in terms of both accuracy and efficiency in relation to the number of simulations performed.

Key words: Parameterized systems, system-on-a-chip architectures, design space exploration, genetic algorithms, multi-objective optimization, Pareto-optimal configurations, power/performance-tradeoffs.

1. INTRODUCTION

A recent reduction in the time to market has led to the development of a new approach to IP-based design in which a highly parametric pre-designed system-on-a-chip (SOC) is configured according to the application it will have to execute. This new approach called *configure-and-execute* [3] is based on the presence of highly parametric IPs (Intellectual Properties) representing the basic components of a SOC. Once the architecture of a system has been designed, that is, it has been decided which IPs to use, it is necessary to find the optimal configuration for them according to the specific application (or set of applications) that have to be executed. The values

chosen for these parameters (bus sizes, coding techniques, cache parameters, arbitration schemes, etc.) are those that optimise a function which almost always depends on three main variables: area, power and performance.

The greatest problems in this area regard exploration of the range of possible system configurations in search of the optimal configuration for a given application. There are, in fact, a number of parameters involved (bus sizes, cache configurations, software algorithms, etc.), each of which has a great impact on design constraints such as area, power and performance. An exhaustive analysis of all possible configurations is thus computationally unfeasible.

The aim of this paper is to present a general methodology to search for the Pareto-set of configurations of a parameterized system that optimise the system in relation to various objectives. The methodology uses multi-objective optimisation techniques based on genetic algorithms. The results obtained in a case study (a highly parametric SOC for digital camera applications) show the efficiency of the approach in terms of both accuracy and the number of simulations required for the exploration.

The paper is structured as follows. In the next section, we review the related work which focuses on tuning methodologies for parameterised systems. In Section 3 the problem will be stated in formal terms. In Section 4 we will present our approach to design space exploration for parameterized systems. The approach will be validated in Section 5 on a highly parametric architecture for digital camera applications. Finally, Section 6 provides our conclusions and indications as to future developments.

2. RELATED WORK

Research in the field of parameterized system design has led to the definition of various approaches to explore the range of configurations. In [4] sensitivity analysis was used to search for the configuration that minimises the power-delay product for a cache memory. In [5] mono-objective genetic algorithms (GAs) were used to search for optimal configurations in terms of area, power and average access time for a memory hierarchy. In [6] a system comprising a CPU, caches and main memory and the interfaces between these cores was analysed to show the power-performance trade-off for various technologies. Of course, any technique used to explore a range of configurations requires tools to evaluate the configurations and thus system-level simulation and estimation techniques. These tools have to be capable of performing system-level simulations in as short a time as possible as well as estimating variables that are typical of a lower level of abstraction (e.g. clock cycles, power consumption) with an

adequate level of accuracy. In [7] the authors used power estimation data obtained from the gate-level for a cores representative input stimuli data, and propagated this data to a higher (object-oriented) system-level model, which is parameterizable and executable. They achieved simulation speedups of over 1000 with accuracies suitable for making reliable power-related system-level design decisions. In [8] the same authors describe a method for speeding up the evaluation further, through the use of instruction traces and trace simulators for every core, not just the microprocessor core.

3. STATEMENT OF THE PROBLEM

The problem dealt with in the paper is that of finding the best configurations (on the basis of certain indexes of optimality) for a parameterized system. These aims often clash, in that an improvement in one index may cause degradation in others. This means that in multi-objective optimisation problems reference is not made to an optimal solution (i.e. one that simultaneously optimises all the objectives), but rather to trade-off solutions (or configurations), that is, solutions that provide a trade-off between various objectives.

For example, *Figure 1* shows the normalised trends for area (A), total switching capacitance (P) and average access time (AAT) for a three-level memory hierarchy with various cache sizes. In each configuration the first-level caches are of the same size while the second-level cache is 4 times the size of the first-level ones. It is clearly impossible to find a configuration that optimises all the objectives at the same time.

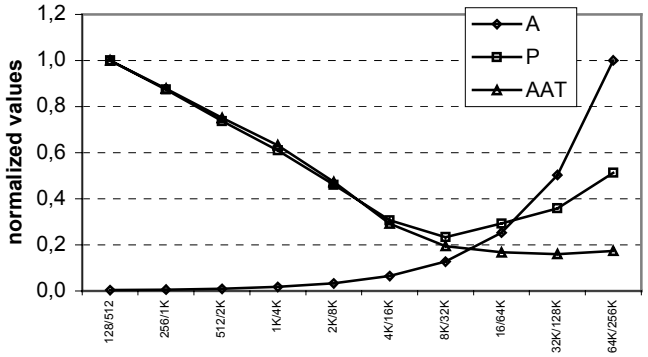

Figure 1. Normalised trends for area, total switching capacitance and average access time for various cache configurations.

Exploration of a range of configurations for a parameterized system can be defined as a set of techniques and strategies to be used to determine *mutually non-dominated* configurations (the concept of dominance will be explained below). The solution to these problems falls into the multi-objective optimisation strategy class. Multiobjective optimisation (also called multicriteria optimisation, multiperformance or vector evaluation) can be defined as the problem of finding [9]: *a vector of decision variables which satisfies constraints and optimises a vector function whose elements represent the objective functions. These functions form a mathematical description of performance criteria which are usually in conflict with each other. Hence, the term "optimise" means finding a solution which would give values for all the objective functions such as to be acceptable to the designer*.

In formal terms we can define the problem in this way: find the vector $\underline{x}^*=[x_1^*, x_2^*,\ldots, x_n^*]^T$ which will satisfy the *m* inequality constraints:

$$g_i(\underline{x}) \geq 0 \quad i = 1,2,\ldots,m \tag{1}$$

the *p* equality constraints

$$h_i(\underline{x}) = 0 \quad i = 1,2,\ldots,p \tag{2}$$

and optimizes the vector function

$$\underline{f}(\underline{x}) = [f_1(\underline{x}), f_2(\underline{x}),\ldots, f_k(\underline{x})]^T \tag{3}$$

where $\underline{x}=[x_1, x_2,\ldots, x_n]^T$ is the vector of decision variables (i.e. a vector representative of a configuration of a parameterized system).

Let be $\Im$ the set of vector $\underline{x}$ that will satisfy (1) and (2). We say that a point $\underline{x}^* \in \Im$ is Pareto-optimal if for all $\underline{x} \in \Im$ either

$$f_i(\underline{x}) = f_i(\underline{x}^*) \quad i = 1,2,\ldots,k$$

or there is at least one $j \in \{1,2,\ldots,k\}$ such that

$$f_j(\underline{x}) > f_j(\underline{x}^*)$$

This definition states that $\underline{x}^*$ is Pareto-optimal if there exist no vectors $\underline{x} \in \Im$ that decrease the value of any component of the cost function (assuming that the objective function is a cost function to be minimised) without

increasing the value of another component of the cost function. Unfortunately, the Pareto optimum is not a single one but a set of solutions called *non-inferior* or *non-dominated* solutions.

4. METHODOLOGY PROPOSED

In this section we will present an approach to design space exploration (DSE) for a parameterised system based on multi-objective genetic algorithms (GAs) to determine the Pareto-optimal configurations.

4.1 Problem Formulation

Current SOC platforms integrate a number of parameterised cores that make it possible to cover a wide range of applications. The space of possible configurations that can be mapped onto a SOC platform is so vast that an exhaustive search for the Pareto-optimal configurations is computationally unfeasible. Evaluation of a configuration requires simulation of the system once configured. Even if a high-level model of the system is available, thus allowing for rapid simulation of a configuration and estimation with a reasonable degree of accuracy of the variables to be optimised, it would be unthinkable to evaluate the whole range of configurations in order to find the Pareto optimal-set.

One possible approach to exploring the range of configurations uses heuristic techniques to limit the range [4][10]. The main disadvantage of this approach is that it requires accurate analysis of the architecture to identify and discard any Pareto-dominated configurations and thus avoid the need to simulate them. This can be solved by using evolutionary techniques and thus treating the exploration in terms of a problem of global optimisation [5].

4.2 GA-based Approaches to Multi-objective Optimisation

Application of evolutionary algorithms (EAs) in multiobjective optimisation has attracted the attention of researchers from different backgrounds [11]. GA-based approaches to multiobjective optimisation are divided into two classes: those not based on the notion of Pareto optimum and Pareto based ones. The first class includes approaches that use aggregating functions to transform the problem of multiobjective optimisation into one of scalar optimisation [12][13]. This approach was used in [5] to search for the configurations that minimised a cost function defined as the aggregate of the area, power and average access time

parameters in a memory hierarchy. The main disadvantage of approaches based on aggregation functions is that they do not generate proper Pareto optimal solutions in the presence of non-convex search spaces, which is a serious drawback in most real-world applications. This can be solved by using Pareto-based approaches in which the idea is to find the individuals that are Pareto non-dominated by the rest of the population. These individuals are then assigned the highest rank and eliminated from further contention. Another set of Pareto non-dominated individuals are determined from the remaining population and are assigned the next highest rank. The procedure is repeated until the whole population is suitably ranked.

4.3 Our Proposal

In this paper we have considered multiobjective optimisation techniques that use Pareto-based GAs. More specifically, we chose the *Strength Pareto EA* (SPEA)[14][15] approach, which is very effective in sampling from along the entire Pareto-optimal front and distributing the solutions generated over the trade-off surface. The flow proposed is shown in *Figure 2*.

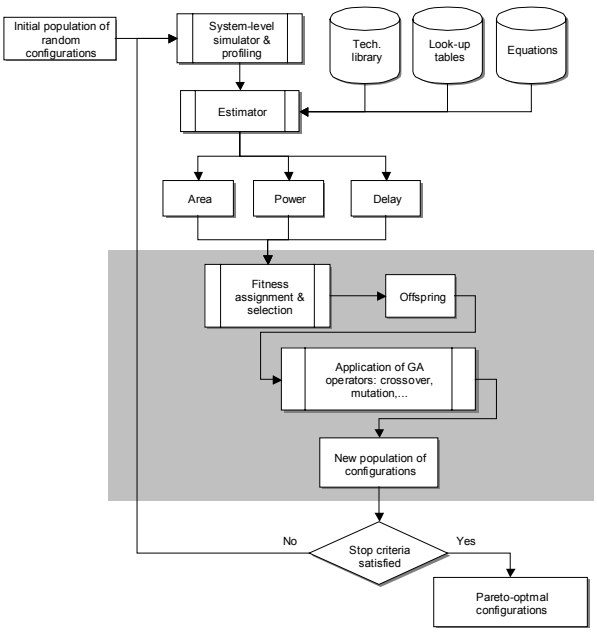

Figure 2. Design flow.

Initially a population of random configurations is generated. Each configuration is mapped onto the SOC platform and the specific application

is executed. The information collected (cache misses, memory accesses, transitions on the buses, etc.) is used by an estimation process which, by means of mathematical equations, look-up tables and technological parameters, gives a fairly accurate estimate of the variables to be optimised (e.g. area, power, performance etc.). Each configuration is assigned a fitness value: the higher this value, the "better" the configuration is in relation to the variables to be optimised (objectives to be reached). The configurations with the highest fitness values are selected and genetic operators (crossover and mutation) are applied with a probability proportional to their fitness value. The population thus generated is the input for a new iteration (or generation). This cycle is repeated for a sufficiently large number of generations.

The algorithm was implemented using Galib [16] (a C++ library of genetic algorithm components). A configuration is represented by an individual of the population whose genome defines its parameters. Each gene represents a system parameter (defined by means of an allele) that only codes values defined within the range that is admissible for the parameter involved. Impossible configurations were excluded by using the approach classified in [17] as rejection of unfeasible individuals.

5. APPLICATION OF THE PROPOSED METHODOLOGY

In this section the approach described in the previous one will be applied to a highly parametric system for digital camera applications, to obtain the Pareto-optimal configurations that optimise the dissipated power and execution time for certain specific applications.

5.1 Reference Architecture

The methodology was applied to the architecture shown in *Figure 3*. It is a highly parametric SOC for digital camera applications developed under the Dalton Project at the University of California at Riverside [18]. The project is an open source one and comprises a parameterized simulation model of a system-on-a-chip composed of an MIPS R3000 processor core, instruction cache (I$), data cache (D$), memory, MIPS to instruction cache bus, MIPS to data cache bus, instruction/data cache to memory bus, bus bridge, peripheral bus, uart and codec.

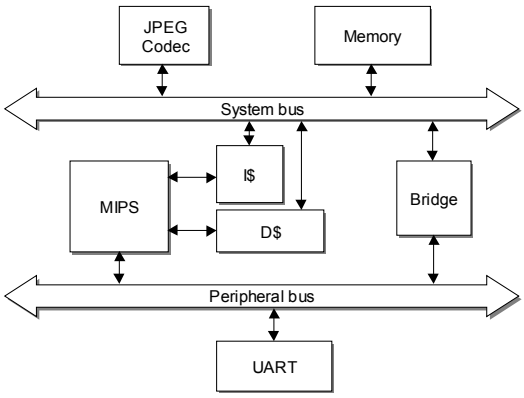

Figure 3. Reference architecture.

Each core is parametric and *Table 1* gives the free parameters and the set of admissible values. For each bus (data bus or address bus) it is possible to configure the number of lines and the encoding scheme to minimise the switching activity. The caches can be configured in size, line size and associativity. For the UART it is possible to define the transmission and reception buffer sizes, and for the JPEG Codec the pixel width can be varied. In all there are 26 separate parameters, giving a total of 9.7×10^{15} possible configurations.

Table 1. Free parameters and the set of admissible values for each core.

Core	Parameter	Parameter space	Config. space
I & D Cache	Size	128B, 256B, 512B, ..., 64KB	10×2
	Line	4B, 8B, 16B, ..., 128B	6×2
	associativity	1, 2, ..., 16B	5×2
I$ & D$ → CPU	dbus/abus width	4, 8, ..., 32	$4 \times 2 \times 2$
	dbus/abus encoding	bin, gray, inv	$3 \times 2 \times 2$
$ → MEM	dbus/abus width	4, 8,... , 32	4×2
	dbus/abus encoding	bin, gray, inv	3×2
Peripheral bus	dbus/abus width	4, 8, 16, 32	4×2
	dbus/abus encoding	bin, gray, inv	3×2
UART	TX/RX buf size	1,2,4,8,16	5×2
Codec	pixel width	10,12	2
Global	volt vs. freq	(1.5,33), (2.6,57), (3.3,72) , (4.0,88), (5.0,110)	5

There are two versions of the system: both a synthesisable VHDL version and a high-level model written in C++. With this model it is possible to perform rapid simulations of the system when it is executing an applications, as well as estimating the execution time and power consumption by using the estimation model described in [7].

5.2 Experiments

The methodology was tested on three benchmarks typical of embedded applications. *Image* rotates an image by 90 degrees and converts it into a grayscale. The *Matrix* application performs a matrix invert operation on a large matrix. *Key* implements a complex chroma-key algorithm.

The GA-based approach proposed was compared with Platune using two evaluation indexes. The first measures the accuracy (or quality) of the solutions obtained by the two techniques/approaches. The second measures the efficiency of our approach by counting the number of configurations explored (i.e. the number of simulations required) to obtain the Pareto-optimal set.

In all cases a crossover probability of 0.9 and a mutation probability of 0.01 were set. Various tests were carried out for each of the three benchmarks, varying the internal population and the number of generations. *Figure 4* compares the results obtained using Platune and those given by our approach, for various numbers of generations. As can be seen, after only 30 generations the solutions found dominate those found by Platune. The results given in the figure refer to the *Image* application, but from a qualitative point of view the same conclusions were reached with the other applications.

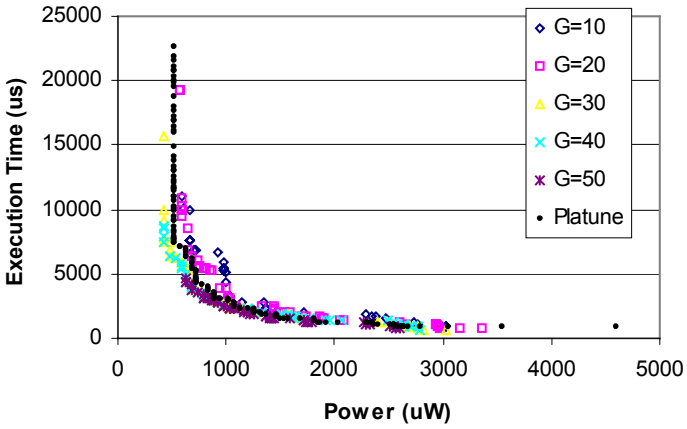

Figure 4. Comparison between the results obtained using Platune and those given by our approach, for various numbers of generations.

The improvements in terms of the quality of the solutions found do not vary significantly when the size of the internal population changes (see *Figure 5*). When, in fact, the population goes from 10 to 50 individuals after 50 generations the results obtained are almost equivalent. As an increase in

the number of individuals making up the population means increasing the number of configurations visited, it is advisable to use small internal populations so as to enhance efficiency without an appreciable deterioration in the results.

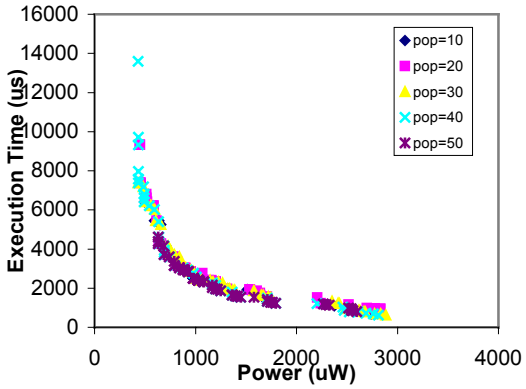

Figure 5. The improvements in terms of the quality of the solutions found do not vary significantly when the size of the internal population changes.

Table 2 summarises the results obtained for the three benchmarks, giving the number of simulations performed by Platune and the genetic approach for varying numbers of generations and internal population sizes. In general, the genetic approach allows an 85 to 99% saving in simulations as compared with Platune for all the benchmarks analysed.

Table 2. Results obtained for the three benchmarks, giving the number of simulations performed by Platune and the genetic approach for varying numbers of generations and internal population sizes.

Platune		GA					
			Internal population size				
		generations	10	20	30	40	50
Image	36090	10	155	365	488	601	741
		20	448	878	1195	1331	1731
		30	906	1705	1934	2445	2665
		40	1540	2436	2904	3186	3710
		50	2294	3104	3535	4336	4581
			Internal population size				
		generations	10	20	30	40	50
Key	41410	10	214	310	436	585	707
		20	515	997	1305	1271	1456
		30	1242	1736	1851	2514	2925
		40	1741	2268	2601	3422	3751
		50	2130	3219	3684	4278	4587

Platune		GA					
			Internal population size				
		generations	10	20	30	40	50
Matrix	34690	10	211	358	471	575	703
		20	575	710	993	1344	1454
		30	901	1392	1969	2390	2564
		40	1213	2234	2638	3076	3650
		50	1969	3007	3788	4090	4683

6. CONCLUSIONS

In this paper we have proposed a GA-based methodology for multiobjective exploration of the range of configurations for a parameterized system. The methodology was applied to a highly parametric SOC for digital camera applications. The approach was evaluated in terms of both accuracy and the CPU time required to explore the range of configurations and compared with the approach used in Platune [2]. The results obtained show that unlike others this approach solves the problem by successive refinement: the more the algorithm is made to evolve, the closer the Pareto-set found is to the Pareto-optimal set. This would appear to be a very useful feature, as the user can choose the accuracy/CPU time trade-off. In terms of CPU time to search for the trade-off front, the approach requires on average 90% fewer simulations than the Platune approach.

Future developments will address definition of a mixed genetic/heuristic approach using evolutionary techniques to achieve global optimisation together with efficient local optimisation techniques.

REFERENCES

[1]T. D. Givargis and F. Vahid. Parametrized system design. In *8th International Workshop on Hardware/Software Codesign*, 2000.

[2] The UCR Dalton Project IP-Based Embedded System Design. http://www.cs.ucr.edu/~dalton/.

[3]F. Vahid and T. Givargis. The case for a configure-and-execute paradigm. In *International Workshop on Hardware/Software Codesign (CODES)*, pages 59--63, May 1999.

[4]W. Fornaciari, D. Sciuto, C. Silvano, and V. Zaccaria. A design framework to efficiently explore energy-delay tradeoffs. *9th. International Symposium on Hardware/Software Co-Design*, pages 260--265, Copenhagen, Denmark, Apr. 25--27 2001.

[5]G. Ascia, V. Catania, and M. Palesi. Parameterized system design based on genetic algorithms. *9th. International Symposium on Hardware/Software Co-Design*, pages 177--182, Copenhagen, Denmark, Apr. 25--27 2001.

[6]T. D. Givargis, J. Henkel, and F. Vahid. Interface and cache power exploration for core-based embedded system design. In *International Conference on Computer-Aided Design (ICCAD)*, pages 270--273, Nov. 1999.

[7] T. D. Givargis, F. Vahid, and J. Henkel. A hybrid approach for core-based system-level power modeling. In *Asia and South Pacific Design Automation Conference*, 2000.

[8] T. D. Givargis, F. Vahid, and J. Henkel. Trace-driven system-level power evaluation of system-on-a-chip peripheral cores. In *Asia South-Pacific Design Automation Conference (ASP-DAC)*, Jan. 2001.

[9] C. A. C. Coello. A comprehensive survey of evolutionary-based multiobjective optimization techniques. *Knowledge and Information Systems. An International Journal*, 1(3):269--308, Aug. 1999.

[10] F. Vahid and T. Givargis. Platform tuning for embedded systems design. *IEEE Computer*, 34(3):112--114, Mar. 2001.

[11] C. M. Fonseca and P. J. Fleming. An overview of evolutionary algorithms in multiobjective optimization. *Evolutionary Computation*, 3(1):1--16, 1995.

[12] D. A. V. Veldhuizen and G. B. Lamont. Multiobjective evolutionary algorithms: Analyzing the state-of-the-art. *Evolutionary Computation*, 8(2):125--147, 2000.

[13] J. D. Schaffer. Multiple objective optimization with vector evaluated genetic algorithms. In L. Erlbaum, editor, *Genetic Algorithms and their Applications: Proceedings of the First International Conference on Genetic Algorithms*, pages 93--100, 1985.

[14] E. Zitzler and L. Thiele. Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach. *IEEE transactions on Evolutionary Computation*, 4(3):257--271, Nov. 1999.

[15] E. Zitzler, K. Deb, and L. Thiele. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation*, 8(2):173--195, 2000.

[16] M. Wall. *GAlib: A C++ Library of Genetic Algorithm Components*. Mechanical Engineering Department, Massachusetts Institute of Technology, Aug. 1996.

[17] C. A. C. Coello. Treating constraints as objectives for single-objective evolutionary optimization. Technical report, Laboratorio Nacional de Informatica Avanzada, Rebsamen 80, Xalapa, Veracruz 91090, Mexico, 2000.

[18] The UCR Dalton Project IP-Based Embedded System Design. http://www.cs.ucr.edu/~dalton/.