Tuning Methodologies for Parameterized Systems Design

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

Key words: Design space exploration, genetic algorithms, system-on-chip architectures, Pareto-optimal configurations.

Abstract: In this paper we present some new solutions for design space exploration of parameterized systems. The approaches use multi-objective optimisation techniques based on the concept of Pareto-optimality to determine the power/performance trade-off front for a highly parametric system-on-a-chip for digital camera applications. The approaches used are purely heuristic, genetic algorithm-based and a combination of the two. The results obtained demonstrate the effectiveness of the approaches in terms of both validity and efficiency, measured as the number of simulations run.

1. INTRODUCTION

With current technology it is possible to integrate hundreds of millions of transistors on a single silicon die. This large number of transistors means that a single chip can contain a complete processing systems with high-performance CPUs, DSPs, co-processors, memories, and analog blocks that communicate via an extremely complex interconnection network. Unfortunately, the technological gap between the number of transistors that could be used and the number actually used is constantly on the increase [1].

This problem has partially been solved by the IP-based design approach that uses pre-designed and pre-verified cores as building blocks (as gates were previously used). These cores can also be supplied by third parties and can be in different forms (soft-, hard-, firm-cores) [2].

A new approach to IP-based design, called *configure-and-execute* was proposed in [3] and is based on the presence of a highly parametric pre-designed system-on-a-chip or (SOC platform) which is configured according to the application or set of applications it will have to execute.

Once the designer has chosen the SOC platform on which the application is to be mapped, he passes to the *tuning* phase to search for a parameter

configuration such as to optimise an objective function that almost always depends on three variables—area, power and performance.

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.

This paper presents three new approaches to design space exploration (DSE) for parameterized systems. The approaches use multi-objective optimisation techniques based on the concept of Pareto optimality. The first approach is an extension of sensitivity analysis [4] to multi-objective optimisation, the second uses multi-objective evolutionary programming techniques, and the third is a combination of the previous two. The results obtained in a case study (a highly parametric SOC for digital camera applications) show the effectiveness of using evolutionary programming techniques for DSE in terms of both accuracy and efficiency, measured as the number of simulations run.

The paper is structured as follows. In Section 2 the problem will be stated and a survey of multi-objective optimisation techniques based on genetic algorithms will be given. In Section 3 three new approaches to design space exploration will be presented and then applied to determine the power/performance trade-off for a highly parametric system described in Section 4 together with the results obtained. Finally, Section 5 provides our conclusions and indications as to future developments.

2. PROBLEM FORMULATION

Let there be a parametric system S in which each of the N parameters p_i can take a value belonging to a finite set $V^{(i)}$. We will call a configuration of the system S an N-tuple $(v_1,v_2,\ldots,v_N)$ with $v_i \in V^{(i)}$ such that $p_1=v_1$, $p_2=v_2$ and so on. We will indicate the space of configurations of the system S, i.e. all the configurations the system can possibly have, as $C(S)$. This set will correspond to the cartesian product of the sets of variation of each system parameter and its size will therefore be $\prod^{N}_{i=1} V^{(i)}$. Let there be M cost functions $f_i: C(S) \rightarrow \mathrm{R}$ (i=1,2,…,M) which associate a real value with each possible configuration of the system. The problem is to find an efficient way to search for the Pareto-optimal configurations given the cost functions f_i.

Problems of this kind are particularly felt in VLSI design, especially that of systems-on-chips (SOCs). A system involving the interconnection of several parametric, heterogeneous functional blocks has to be configured in such a way as to find the Pareto-optimal configurations which are typically in terms of area, power and execution time. Beside the basic problem of the enormous size of the space of possible system configurations, another great problem is that of evaluating the costs functions for a given configuration. This is a computationally onerous task, especially for complex systems like SOCs. First of all, in fact, it requires the system to be configured according to the given configuration, then execution of the application, and finally measurement (or estimation) of the variables of interest. It is therefore clear that a technique to search for the Pareto-optimal configurations of a parameterised SOC based on an exhaustive approach is unfeasible.

The use of heuristic techniques can help to reduce the space of configurations that have to be analysed by identifying and discarding any Pareto-dominated [4] configurations. Another approach presented in [5] proposes the use of genetic algorithms as an efficient technique for DSE.

Both techniques reduce the problem of multi-objective optimisation to one of scalar optimisation by aggregation of the objective functions [7],[8].

The main disadvantage to 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. These problems are solved by Pareto-based approaches that select Pareto non-dominated individuals from the rest of the population. These individuals are then assigned the hightest rank and eliminated from further contention. Another set of Pareto non-dominated individuals are determined from the remaning population and are assigned the next hightest rank. The procedure is repeated until the whole population is suitably ranked.

3. NEW MULTI-OBJECTIVE APPROACHES FOR SYSTEM LEVEL EXPLORATION

In the following subsections, the approaches based on sensitivity analysis and GAs will be extended to conduct a proper multi-criteria analysis of the notion of Pareto optimum.

3.1 Pareto-Based Sensitivity Analysis

The sensitivity analysis (SA) presented in [4] reduces the space of possible configurations in two phases. The first phase identifies the parameters which most influence the objective function to be optimised

(sensitivity analysis phase (SAP)). For a system with P parameters, determination of the degree of sensitivity of each parameter consists of fixing $P-1$ parameters and varying one of them, determining the maximum range of variation of the objective function. One way to fix the parameters is to consider the mean value of their variation set.

The next phase, design space exploration phase (DSEP), identifies the optimal value for each parameter, from the most to the least sensitive. If $V^{(i)}=\{v_1^{(i)},...,v_{Ni}^{(i)}\}$ is the set of values the parameter p_i can take, the number of configurations to be evaluated goes down from $\prod_{i=1}^{P} N_i$ to $\sum_{i=1}^{P} N_i$.

To understand how this approach works, let us consider the following example. Let us assume that we want to find the configuration of a cache memory in terms of size (s), block size (b) and associativity (a) that minimises the power-delay product (pd). Let S, B and A respectively be the set of possible values of the parameters s, b and a. The SAP is performed as follows: we fix $b=b_0$ and $a=a_0$ and s is made to vary in S, obtaining the set $PD=\{PD_1,PD_2...,PD_{|S|}\}$ of values of pd. If $pd_{max}=\max(PD)$ and $pd_{min}=\min(PD)$, the sensitivity of the parameter s is $s_S=pd_{max} - pd_{min}$. The same procedure is repeated for the remaining parameters s_B and s_A.

Under the hypothesis that $s_S>s_A>s_B$, we set $a=a_0$, $b=b_0$ and vary $s\in S$, obtaining $PD=\{PD_1,..,PD_{|S|},\}$ values of pd. If $s_{opt}=\min(PD)$, we fix $s=s_{opt}$ and $b=b_0$ and vary $a\in A$. Proceeding as previously, a_{opt} is determined. In short, having fixed $s=s_{opt}$, $a=a_{opt}$ and varying $b\in B$ we determine b_{opt}. The configuration $\langle s_{opt}, b_{opt}, a_{opt}\rangle$ will determine a pd value close to pd_{min}. Algorithm 1 shows the generalization of the DSEP assuming all parameters sorted by decreasing values of sensitivity.

Algorithm 1: Monono-objective sensitivity analysis (DSEP)

```
for all i ∈ {1,2,…,P}
   for all j ∈ {1,2,…,i-1}
       vj = vj^opt
    end for
    for all j ∈ {i+1,i+2,…,P}
       vj = vj0
    end for
    vi^opt = v1^(i)
   for all j ∈ {2,3,…,Ni}
      if f(v1,…, vj^(i),…,vP) < f(v1,…, vi^(opt),…, vP)
         vi^(opt) = vj^(i)
      end if
   end for
end for
```

To overcome the limits of a mono-objective approach we extended the technique based on sensitivity analysis to perform multi-objective optimisation based on the notion of Pareto optimum (PBSA).

The SAP was modified by defining a new metric to measure the sensitivity of a parameter. We have defined the sensitivity s_i of the i-th parameter as:

$$s_i = \max_{h,k \in \{1,2,\ldots,N_i\}} dist(\underline{o}_h^{(i)}, \underline{o}_k^{(i)})$$

where dist is the Euclidean distance and $\underline{o}_j^{(i)}$ are the N_i points in the *n*-dimensional space obtained by fixing *P-1* parameters and varying $p_i \in V^{(i)}$.

Indicating with S_i a parameter order by decreasing degrees of sensitivity, i.e. such that $s_{Si} > s_{Si+1}$, we defined the DSEP as follows.

Having fixed $p_{S2}, \ldots, p_{SP}$ parameters, $p_{S1} \in V^{(S1)}$ is made to vary. From the N_{S1} points obtained, whose components represent the objective values, the non-dominated configurations are extracted and accumulated in a set *P*. At the second iteration for each configuration in the set *P*, $p_{S2} \in V^{(S2)}$ is made to vary. From the $N_{S1} \times |P|$ obtained, the non-dominated configurations are extracted and accumulated in the set *P*. The procedure is repeated for all the parameters p_{Si} with $i < \nu$, where ν is the first index such that $s_{Si} < T$ and *T* is a *sensitivity threshold*. Parameters whose sensitivity is less than *T* influence in a limited way the value of the objective functions. So they are fixed to a reference value. At the end of the algorithm the configurations in *P* will represent the trade-off surface identified. The Algorithm 2 gives the pseudo-code of the DSE procedure.

Algorithm 2: Pareto-based objective sensitivity analysis (PBSA)

```
Require S1,S2,…,Sm //sorted by sensitivity parameter's index
ND = p*    // initialize non-dominated set
i = 1 // high sensitive parameter index
Repeat
  C={}
  for all c ∈ ND
    for all v ∈ V(Si)
      c[Si]=v
      C = C ∪ c
   end for
  end for
  ND =ND ∪ C
  ND = ND \ Dominated(ND) // remove dominated solutions
  i = i + 1 // next high sensitive parameter
until (i>P OR Sensitivity(Si) < T)
```

3.2 Pareto-based Genetic Algorithms

In this paper we consider multiobjective optimisation techniques that use Pareto-based GAs. More specifically, we chose the *Strength Pareto EA* (SPEA) [9] 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 1*. An initial population of configurations is simulated to obtain an estimate of the parameters to be optimised. Together with the constraint specifications, these parameters will be used by the genetic exploration algorithm to generate the next population to be evaluated. The cycle is repeated until a stop condition is met and Pareto-optimal solutions are provided.

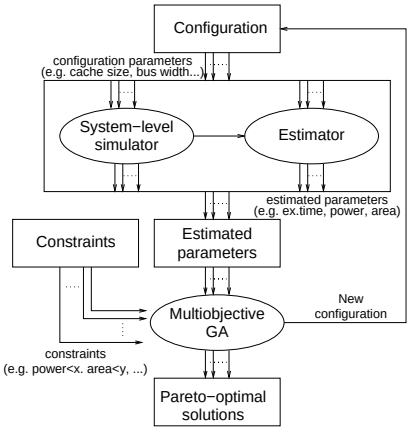

Figure 1. GA Design flow.

The algorithm was implemented using GAlib [10] (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 [11] as rejection of unfeasible individuals.

3.3 A Mixed Approach

A mixed approach can be used to exploit the potential of the two approaches presented in the previous subsections. The advantage of sensitivity analysis lies in its reduction of the configuration space by neglecting parameters that have less effect on the objective functions. The genetic approach, on the other hand, allows for rapid exploration of the space of configurations. We could therefore envisage defining the chromosome, using only the most sensitive parameters determined by the sensitivity analysis, to define a new mixed approach which we will call SAGA (Sensitivity Analysis Genetic Algorithm).

Obviously the trade-off front obtained by SAGA will not be better than that obtained by the pure GA approach, given that the space of configurations on which SAGA operates is a subspace of the space on which the GA approach operates. As compared with PBSA, on the other hand, the solutions found will be better, as the parameter tuning is not constrained: any combination of parameter values is admissible.

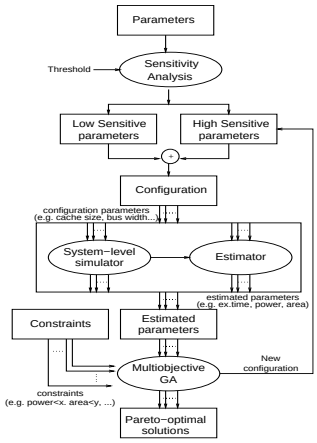

Figure 2. Mixed approach design flow.

The flow of operations performed by SAGA, *Figure 2*, is divided into two stages. In the first, the sensitivity analysis is performed and the parameters that exceed a certain sensitivity threshold are determined. These parameters will define the chromosome of the genetic algorithm that will be applied in the second stage. The remaining parameters that remain below the threshold are set to a reference value (e.g. the mean value of their range of variation).

4. EXPERIMENTAL RESULTS

In this section we will first describe the parametric reference architecture and then apply the approaches described in the previous section.

4.1 Reference Architecture

To test the methodologies proposed in Section 3 we used the architecture shown in *Figure 3*. It is a highly parametric SOC for digital camera applications developed under the Dalton Project [12]. 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. Each core is parametric. 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.

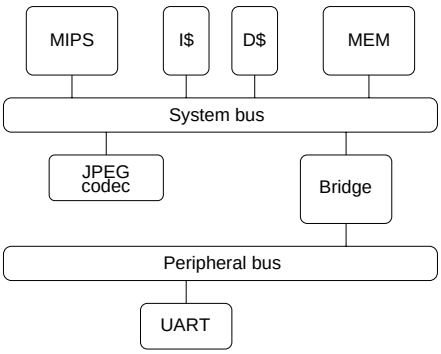

Figure 3. Reference architecture.

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 [13].

4.2 Experiments

The three approaches were compared considering three different applications. The first, *image*, copies a bitmap from one memory region to another. The second, *key*, works on large-size matrices. The third, *matrix*, performs arithmetical operations on two 10x10 matrices of integers.

Figure 4 gives the power/execution-time trade-off for the *image* application, obtained using the three approaches. PBSA was executed with different threshold values. As was to be expected, when the threshold decreases more parameters are taken into consideration, thus leading to an improvement in the solutions obtained but an increase in the number of simulations required (respectively 1780, 4958 and 8633 for thresholds of 10%, 5% and 1%). SAGA gives the same results as PBSA with a 1% threshold but after only 30 generations with internal and external populations of 50 individuals, and a total of 2238 simulations. With GA, after 50 generations with internal and external populations of 50 individuals, a total of 4581 simulations gives dominant solutions as compared to those obtained with PBSA and SAGA.

The same qualitative results are obtained in the other two applications -- *key* and *matrix* and *Table 1* gives the number of simulations run by each

approach and the percent gain over PBSA. The results were obtained using a 1% threshold for both PBSA and SAGA. For SAGA and GA we used an internal and external population of 50, a crossover probability of 0.9 and a mutation probability of 0.01. GA and SAGA respectively converged after 50 and 30 generations for *image* and *key* and after 40 and 20 generations for *matrix*.

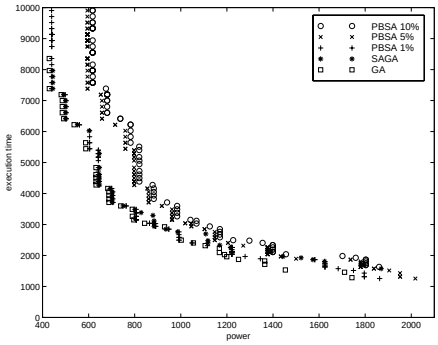

Figure 4. Power/execution time trade-off for the *image* application.

Benchmark	PBSA	GA		SAGA	
	simulations	simulations	% saving	simulations	% saving
Image	8633	4581	47.5	2238	74.1
Key	11019	4587	58.4	2642	76
Matrix	4372	3751	14.2	1714	60.8
Total	24024	2919	46.2	6594	72.6

Table 1. Number of simulations run by each approach and the percent gain over PBSA.

In short, the solutions obtained with GA dominate those obtained with PBSA and are achieved with on average 46% fewer simulations. If we are willing to sacrifice the quality of the solutions to obtain an increase in efficiency, we can use the mixed SAGA approach, which gives solutions close to those obtained with GA but with 72% fewer simulations than PBSA.

5. CONCLUSIONS

In this paper we have presented three approaches to design space exploration for parameterized systems. The first approach is heuristic and is based on sensitivity analysis to reduce the space of configurations to be explored. The second uses multi-objective genetic algorithms, while the third is a mixture of the previous two, using sensitivity analysis to limit the space

of configurations and genetic algorithms to explore the subspace thus obtained. All three approaches were applied to determine the power/performance trade-off of a highly parametric architecture implementing a SOC for digital camera applications during the execution of different applications. The approaches were evaluated in terms of both the quality of the solutions obtained (using the concept of Pareto dominance) and efficiency, measured as the number of simulations required to determine the trade-off front. The results obtained show the effectiveness of the pure genetic approach as regards the quality of the solutions obtained, and the mixed approach in terms of efficiency.

6. REFERENCES

[1] "International technology roadmap for semiconductors", SIA, 1999.

[2] R. Rajsuman, *System-on-a-Chip Design and Test*, Artech House, 2000.

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

[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*, Copenhagen, Denmark, Apr. 25-27 2001, pp. 260-265.

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

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

[7] D. Van Veldhuizen and G. B. Lamont, "Multiobjective evolutionary algorithms: Analyzing the state-of-the-art", *Evolutionary Computation*, vol. 8, no. 2, pp. 125-147, 2000.

[8] J. D. Schaffer, "Multiple objective optimization with vector evaluated genetic algorithms", *First International Conference on Genetic Algorithms*, 1985, pp. 93-100.

[9] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach", *IEEE transactions on Evolutionary Computation*, vol. 4, no. 3, pp. 257--271, Nov. 1999.

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

[11] C. A. Coello Coello, "Treating constraints as objectives for single-objective evolutionary optimisation", Tech. Rep., Laboratorio Nacional de Informatica Avanzada, Rebsamen 80, Xalapa, Veracruz 91090, Mexico, 2000.

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

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