Fuzzy Decision Making in Embedded System Design

Alessandro G. Di Nuovo
adinuovo@diit.unict.it

Maurizio Palesi
mpalesi@diit.unict.it

Davide Patti
dpatti@diit.unict.it

Dipartimento di Ingegneria Informatica e delle Telecomunicazioni
Università degli Studi di Catania
Catania, Italy

ABSTRACT

The use of Application Specific Instruction-set Processors (ASIP) is a solution to the problem of increasing complexity in embedded systems design. One of the major challenges in ASIP design is Design Space Exploration (DSE), because of the heterogeneity of the objectives and parameters involved. Typically DSE is a multi-objective search problem, where performance, power, area, etc. are the different optimization criteria. The output of a DSE strategy is a set of candidate design solutions called a Pareto-optimal set. Choosing a solution for system implementation from the Pareto-optimal set can be a difficult task, generally because Pareto-optimal sets can be extremely large or even contain an infinite number of solutions. In this paper we propose a methodology to assist the decision-maker in analysis of the solutions to multi-objective problems. By means of fuzzy clustering techniques, it finds the reduced Pareto subset, which best represents all the Pareto solutions. This optimal subset will be used for further and more accurate (but slower) analysis. As a real application example we address the optimization of area, performance, and power of a VLIW-based embedded system.

Categories and Subject Descriptors

I.6.7 [**Simulation and Modeling**]: Simulation Support Systems—*Environments*; J.6 [**Computer Applications**]: Computer-Aided Engineering—*Computer-aided design*

General Terms

Algorithms, Design

Keywords

Multi-objective optimization, Pareto-set reduction, clustering, decision making.

1. INTRODUCTION

The major task of system level design is to generate design concepts, evaluate them, and select one or more best concepts for further refinement in the latter design stages. Concept selection is important, because poor selection of a design concept can rarely be compensated for at later design stages and incurs great redesign costs. It has been recognized that nearly 75% of product life-cycle cost is committed by the end of conceptual design [19].

Embedded system design can be viewed as a complex design space exploration (DSE) problem that attempts to optimize conflicting criteria, for example maximizing performance while minimizing energy and area requirements. The optimization involves the simultaneous consideration of several incomparable and often competing objectives.

Many multi-objective DSE approaches have been proposed in the literature [10, 13, 12, 1, 6, 11]. Although they operate at different level of abstraction and deal with different optimization objectives, the overall goal is always the same: find a good approximation of the Pareto-optimal surface (i.e., the image of the Pareto-optimal set in the space of objectives). Unfortunately, these approaches themselves do not distinguish between solutions; all the solutions found are therefore considered to be equivalent to each other. This means that the designer is given no indications about the choice of the most effective solution to be implemented.

When properly formulated, the goodness or desirability of a design is almost fully represented by its location in the objective space. For example, when two points are near each other in the objective space, that closeness is supposed to indicate the designer's indifference to their relative goodness. That is, they are of nearly identical desirability to the designer, regardless of their otherwise different features, or potentially different relative locations in the design variable space. If this is not the case, then the objective space becomes less useful or meaningful, and the Pareto frontier also loses its value. These observations explain why we do not focus on the design variable space. If a certain geometric feature is desirable (say, a smaller number of functional units), then that feature should simply become part of the decision space. In that case, two designs with differing numbers of units will simply not be neighbors in the objective space.

Once the set of Pareto-optimal design points is identified from a multi-objective design space exploration, the system designer has to select a design point that is a trade off between the various objective functions. But as no further information is given, none of the Pareto-optimal solutions can be said to be superior to the others. It is also difficult to assess the performance of a design concept that is just a rough idea or sketch at this stage. Experience shows that in the presence of complex design spaces, the number of different Pareto-optimal solutions can be overwhelming. So an auto-mated decision-making process is needed, in which the designer has to handle only some of the Pareto-optimal designs. In fact, from the designer's point of view, presenting all the Pareto-optimal solutions found is useless when their number is huge i.e. it exceeds reason-

CODES+ISSS'06, October 22–25, 2006, Seoul,Korea.

able bounds. Thus, the motivation for the current work stems from challenges encountered during the post-Pareto analysis phase.

In this paper we propose a fuzzy clustering approach to assist the decision-maker in analysis of the solutions to multi-objective problems. Although the proposed approach is general, we apply it to optimization of the design of a parameterized embedded computer system. Fuzzy approaches are often used in decision-making processes, especially when sources of uncertainty are involved [23, 18]. Specifically, this paper develops a methodology that allows us to identify a minimal number of Pareto solutions needed to adequately represent the multi-objective design space, thus providing the designer with a small number of disparate design alternatives from which to choose the final design. In addition, by using information provided by the fuzzy clustering algorithm it is possible to design a fuzzy rule based classifier starting from the geometric properties discovered [5]. The fuzzy if-then rules are easy to read and understand, so they could be very useful for educational purposes.

2. RELATED WORK

Engineering design generally involves assessing and managing tradeoffs between conflicting design objectives. Multi-objective optimization is a powerful method that is commonly used to mitigate this conflict and arrive at a final design. The concept of *Pareto optimality* is central to multi-objective optimization. The complete set of such solutions is referred to as the *Pareto surface* of *Pareto-hyper-surface* in n-dimension. We will use the term *Pareto set* to refer to a partial discrete representation of the Pareto hyper-surface. In this area the most significant problems are related to: (i) the definition of methods to obtain a comprehensive Pareto set and (ii) the development of strategies to obtain a significantly reduced Pareto set for effective decision making.

The first problem has been addressed in several work. In the field of embedded system design we refer to [1]. The second problem is the main topics of this paper. In [9] Das suggests that some Pareto points should be viewed as superior to others. Das introduces the concept of order of efficiency, which provides a notion that is stronger than Pareto optimality and allows to set up a preference ordering amongst various alternatives that are Pareto optimal. The same author in [8] provides a problem formulation for identifying the so-called *knee* of the Pareto frontier. The knee is characterized as the region of the curve that protrudes the furthest away from the line connecting the endpoints of the Pareto frontier. The purpose for identifying the knee is to direct the decision maker to a region at the middle of the Pareto frontier. In [3] Di Barba states that, in several cases, the identification of a few non-inferior solutions is often sufficient for the design purposes. Mattson *et al.* in [18] present a general method to obtain smart Pareto sets in n-dimension. The smallness of the set and the desired degree of tradeoff among objectives in the resulting set are controlled by the designer. An outranking preference model based on the possibility theory has been developed by Wang in [23]. It allows to model the imprecise preference relation between each pair of design concepts. A method based on a data mining technique has been proposed by Taboada and Coit in [22] to assist the decision-maker in the analysis of the solutions of multi-objective problems. The method provides the decision-maker a smaller set of optimal tradeoffs either k different selections or the *knee* cluster.

In this paper we present a novel approach that not only allows to extract the optimal sub-set of representative solutions from a Pareto set but also ranks all the Pareto points indicating their similarity with the most representative solutions. This ranking allows the designer to replace one suggested solution of the subset with an other one when the first one proved to be unfeasible in the latter stages of the production process.

Using a clustering algorithm, which equally divides the Pareto set into an optimum number of clusters is useful to determine a more balanced Pareto subset. In fact all the DSE algorithms are not able to find the whole Pareto-set, because of the long time needed by simulations. The algorithm allows to identify those single points that differ in meaningful way, erasing eventual unbalanced clusters and re-establish the proportions, that they can above all be lost from the used algorithm more, that genetic one for via of its stochastic nature.

The approach is presented in section 3, then in section 5 a real world application is studied with numerical examples using the framework described in section 4. Finally section 6 gives our conclusions.

3. FUZZY DECISION MAKING METHOD

The fuzzy method for decision making we propose is based on the most famous unsupervised fuzzy clustering algorithm, the Fuzzy C-Means (FCM) [4].

To assess clustering performances we use the Xie-Beni (XB) cluster validity index [24] as the underlying optimizing criterion since it has been shown to be better able to indicate the correct number of clusters in several cases [20].

3.1 Fuzzy C-Means

Indicating the degree of membership (or "similarity") of a pattern to every group is called "Fuzzy Clustering"; this is different from "Hard Clustering", which simply associates them. In "hard" approaches, a data set is subdivided into separate partitions in which each pattern belongs to a single cluster. In the fuzzy approach, on the other hand, partitions are not created: elements are associated to each group, with a degree of membership. The result yielded by this algorithm is thus not a simple partitioning (which can, however, still be achieved by assigning each pattern to the group with which it has a higher numerical affinity, for example), but more detailed information on the relations between the patterns and groups. There are two main groups of Fuzzy Clustering techniques; the first includes algorithms that directly use fuzzy sets, while the second comprises ad-hoc algorithms such as Fuzzy C-Means. Further details of these and all other types of clustering are to be found in [15].

The FCM version we implemented is the classical one, which proposes to minimize the following objective function with respect to fuzzy memberships $U = [u_{ij}]$ and cluster centroids $C = [\mathbf{c}_j]$:

$$J(U, C; X) = \sum_{j=1}^{K} \sum_{i=1}^{N} u_{ij}^{m} \cdot d(\mathbf{x}_i, \mathbf{c}_j) \tag{1}$$

where $\mathbf{c}_j$ is the prototype of the j-th cluster and $d(\bullet, \bullet)$ is a distance metric appropriately chosen from the pattern space, $\mathbf{x}_i$ is the i-th pattern, u_{ij} is the degree of truth of the i-th pattern in the j-th cluster, raised to the "fuzzyfier" m. K and N are respectively the number of clusters and the number of patterns. m is a parameter on which the degree of fuzzyfication depends: as its value increases, so does the degree of uncertainty, until it settles at $u_{ij} = 1/K \; \forall \, i, j$, whereas when it gets close to 1 the result is an hard partitioning (i.e. u_{ij} becomes a binary variable which is equal to 1 if the i-th pattern belongs to the j-th group, otherwise it is 0). The FCM algorithm can be summarized in the following steps :

1. Let N and $K \in [2, N[$ be the cardinality of the set of patterns and the initial number of clusters respectively.

2. At step $p = 0$, randomly initialize the matrix $U^{(0)} = [u_{ij}]$, which satisfy the following constraint

$$\sum_{j=1}^{K} u_{ij} = 1 \quad \forall i \in \{1, 2, ..., N\} \tag{2}$$

3. At step $p > 0$, calculate the centroids $C^{(p)}$, i.e. the prototype vectors $\mathbf{c}_j$, starting from $U^{(p-1)}$:

$$\mathbf{c}_j = \frac{\sum_{i=1}^{N} u_{ij} \cdot \mathbf{x}_i}{\sum_{i=1}^{N} u_{ij}} \tag{3}$$

4. Update the matrix $U^{(p-1)}$, obtaining $U^{(p)}$:

$$u_{ij} = \frac{1}{\sum_{k=1}^{K} \left[\frac{d(\mathbf{x}_i, \mathbf{c}_j)}{d(\mathbf{x}_i, \mathbf{c}_k)}\right]^{2/(m-1)}} \tag{4}$$

5. Verify whether the STOP criterion is satisfied; if it is not, return to point 3.

The STOP criterion normally chosen is $\left\| U^{(p)} - U^{(p-1)} \right\| < \varepsilon$, with $\varepsilon \geq 0$. In order to avoid long calculation time it is preferable to choose a certain number of iterations S as the STOP criterion; in this case the first condition is also applied and the algorithm is stopped when one of the two is met.

However, the FCM algorithm needs to be invoked repeatedly, using different initial conditions (i.e. different matrices $U^{(0)}$) and extracting the one that minimizes the validity index. An alternative to repeated tests would be to use a genetic algorithm to determine the best initial conditions, which minimize either the objective function or the validity index. This method was investigated in [14]. The authors reported an increase in computing time of about two orders of magnitude without obtaining significantly better results than those achieved in the "trial-and-error" approach we used.

The FCM implementation we used in this work has $m = 2$, $\varepsilon = 10^{-5}$, $S = 100$, and uses the Euclidean metric as distance measure.

3.2 Cluster Validation Index

Cluster validation is generally a measure of separation among clusters and cohesion within clusters [20]. For our purposes we need a cluster validity measure which satisfies the following requirements: 1) It has intuitive meaning; 2) it is easy to compute; 3) it is mathematically justifiable. Xie and Beni proposed one such cluster validity criterion, based on a validity function, which identifies overall compact and separate fuzzy c-partitions without assumptions as to the number of substructures inherent in the data. The Xie-Beni index is defined as the ratio of compactness σ/N of the total variation to the minimum separation sep of the clusters, where σ and sep can be written as

$$\sigma(U, C, X) = \sum_{j=1}^{K} \sum_{i=1}^{N} u_{ij}^2 \left\| \mathbf{x}_i - \mathbf{c}_j \right\|^2 \tag{5}$$

and

$$sep(C) = \min_{j \neq k} \{ \|\mathbf{c}_j - \mathbf{c}_k\|^2 \} \tag{6}$$

The XB index is then defined as

$$XB(U, C, X) = \frac{\sigma(U, C, X)}{N \cdot sep} \tag{7}$$

Note that, when the partitioning is compact and good, the value of σ should be low, while sep should be high. Therefore, the Xie-Beni index (XB) should have a low value when the data has been appropriately clustered.

3.3 Data treatment

When a data set, e.g. representing the Pareto-front of an embedded system, characterized by heterogenous features is to be analyzed, a good practice is to normalize the patterns in order to avoid bias in the distance metric. For example if we consider the number of CPU cycles to measure performances and Watts to measure power consumption, we will have values that differ by several orders of magnitude. For the clustering algorithm, which bases the clustering on distances between patterns, the performance weight in decision making will be thousands times higher than power dissipation, i.e. the decision will be made considering only the performances. To avoid this problem, we chose to normalize the patterns into the hypercube $[0, 1]^D$, where D is the number of features. Let $\mathbf{x}_i = \{x_{i1}, x_{i2}, ..., x_{iD}\}, i = 1, 2, ..., N$ be the i-th data pattern, the normalized value of its j-th feature is obtained by :

$$x_{ij} = \frac{x_{ij} - \min\limits_{1 \leq k \leq N}(x_{kj})}{\max\limits_{1 \leq k \leq N}(x_{kj}) - \min\limits_{1 \leq k \leq N}(x_{kj})} \quad \forall j \in \{1, 2, ..., D\} \tag{8}$$

Sometimes it may be desirable to give more importance to some features. This need could be take in account using a weighted Euclidean metric as distance measure in (4), (5) and (6):

$$d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{j=1}^{D} w_j^2 (x_j - y_j)^2} \tag{9}$$

where w_d are weights chosen on the basis of the d-th attribute significance. For instance if a low power consumption is more important than area requirement, it can be useful assign to the first feature a weight higher than the one of the second.

3.4 The fuzzy decision making method for DSE

The methodology we propose can be summarized in the following steps:

Algorithm 1 The fuzzy decision making Fuzzy method for DSE.

1. By using a multi-objective DSE stategy, derive the set of Pareto-optimal system configurations P.
2. Remove from P all the system configurations which do not satisfy the design constraints.
3. Determine the $K_{optimum}$ number of clusters in which the Pareto set can be partitioned. The $K_{optimum}$ could be defined as the number of clusters K which minimizes the Xie-Beni index among a predefined range of clusters.
4. Use the optimum partition to choose between the $K_{optimum}$ elements to use in further stage of design. (For example, in this work, we chose the $K_{optimum}$ points which had the highest fuzzy membership with the $K_{optimum}$ clusters).

Of course in step 3 it is possible to set the number of clusters $K_{optimum}$ a priori.

The $K_{optimum}$ solutions extracted will be the reduced Pareto-set from which the designer will extract the entry point for the next stages in the design flow. The best representative solutions, however, could be easily replaced by other ones belonging to the same cluster, according to the fuzzy memberships. In fact the $K_{optimum}$ solutions are useful to figure out the general characteristics (i.e. the profiles) of the clusters. Unfortunately, in the following design phases, these candidate solutions can fall into unfeasible regions of the design space. Searching a new valid candidate solution is a difficult task, because a lot of design decisions derive from previous assumptions. For example let us consider the design of portable embedded system whose battery was chosen to meet a certain power requirement. If power requirements vary significantly a different battery has to be used. Changing battery has a strongly impact on system costs and sizes. So a substantial redesign of the entire system is needed. The best choice is a configuration similar to the original one. In our method this condition of "similarity" is measured by the fuzzy degree of membership of the Pareto-solutions to the clusters associated by FCM algorithm.

4. SIMULATION FRAMEWORK

To evaluate and compare the performance indexes of different architectures for a specific application, one needs to simulate the architecture running the code of the application. In addition, to make architectural exploration possible both the compiler and the simulator have to be retargetable. Trimaran [7] provides these tools and thus represents the pillar around which the EPIC-Explorer was constructed [2] to provide a framework that allows to evaluate any instance of the architecture in terms of area, performance and power. The EPIC-Explorer platform, which can be freely downloadable from the Internet [21], allows the designer to evaluate any application written in C and compiled for any instance of the platform, for this reason it is an excellent testbed for comparison between different design space exploration algorithms.

4.1 Parameterized VLIW Architecture

The Trimaran system is based on HPL-PD [16] which is a parametric processor meta-architecture designed for research in instruction-level parallelism of EPIC/VLIW architectures[1]. The HPL-PD opcode repertoire, at its core, is similar to that of a RISC-like load/store architecture, with standard integer, floating point (including fused multiply-add type operations) and memory operations.

Architectural parameters can be classified in three main categories: *register files*, *functional units* and *memory sub-system*. The first two depend on the implementation of the VLIW core and regard the size of the register files, in terms of the number of registers contained in each of them, and the number of functional units for each type of unit supported. As far as the former are concerned, five different types of register file can be identified: GPR (32-bit registers for integers), FPR (64-bit registers for floating point values) PR (1-bit registers used to store the Boolean values of predicated instructions), BTR (64-bit registers containing information about possible future branches) and CR (32-bit control registers containing information about the internal state of the processor). The functional units involved are: *Integer units*, *floating point units*, *memory units* (associated with load/store operations) and *branch units* (associated with branch operations). As regards the memory subsystem, the parameters that can be modified are the *size*, *associativity* and *block size* for each of the three caches: First-level data cache (L1D), first-level instruction cache (L1I) and second-level unified cache (L2U).

[1]EPIC is an extension of the VLIW approach; where it is not necessary to make a distinction we will only use the term VLIW for the sake of simplicity.

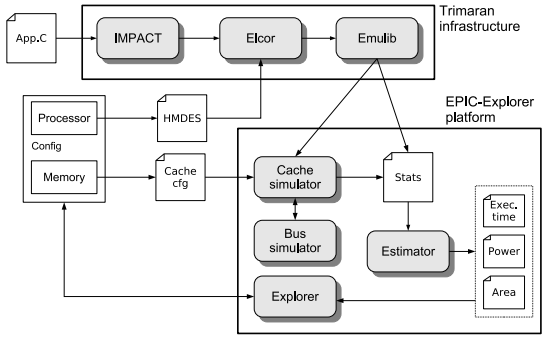

Figure 1: Block diagram of the framework.

4.2 Evalutation Flow

We will now show a functional scheme highlighting the main blocks of the EPIC-Explorer framework and the interface with the Trimaran tools. The input for the whole evaluation flow is the source of the application involved in the optimization and the configuration of the architecture being evaluated. With reference to Figure 1 this input is represented by the blocks *Application.C* and *Configuration*. The application (Application.C) is first compiled by the Trimaran's compiler front-end (IMPACT). This front-end performs ANSI C parsing, code profiling, classical code optimizations and block formation. The intermediate code produced, together with the High Level Machine Description Facility (HMDES) machine specification [16], represents the Elcor input. The HMDES is the machine description language used in Trimaran, which describes a processor architecture from the compiler's point of view. With reference to the tunable parameters outlined in the previous subsection, the hmdes file specifies the number and type of register files and functional units in the VLIW core. Elcor is the Trimaran's back-end VLIW compiler for the HPL-PD architecture and it is parameterized by the hmdes machine. It performs three tasks: Code selection and scheduling, register allocation, and machine dependent code optimizations. At the end of the compilation flow, Trimaran supplies a simulation library (Emulib) which makes it possible to execute the VLIW code produced by Elcor, generating a file (Stats) containing the execution statistics (e.g., instruction mix, execution cycles, utilization of functional units, etc.). A cache simulator, along with a bus simulator, is used to gather information about the behavior of the memory hierarchy in terms of miss rate and data/address traffic on the interconnection buses.

Together with the configuration of the system, the statistics produced by simulation contain all the information needed to apply the area, performance and power consumption estimation model implemented in *Estimator* component of EPIC-Explorer. The results obtained by these models are the input for the *Explorer* component. This component executes an optimization algorithm, the aim of which is to modify the parameters of the configuration so as to minimize the three cost functions (area, execution time and energy/power consumption).

5. A CASE STUDY

5.1 Design space

The design space explored is reported in Table 1.

The performance statistics produced by the simulator are ex-

Table 1: Parameters Space.

Parameter	Parameter space
GPR / FPR	16,32,64,128
PR / CR	32,64,128
BTR	8,12,16
Integer/Float Units	1,2,3,4,5,6
Memory/Branch Units	1,2,3,4
L1D/I cache size	1KB,2KB,...,128KB
L1D/I cache block size	32B,64B,128B
L1D/I cache associativity	1,2,4
L2U cache size	32KB,64KB...,512KB
L2U cache block size	64B,128B,256B
L2U cache associativity	2,4,8,16
Space size	7.7397×10^{10}

pressed in clock cycles. To evaluate the execution time it is sufficient to multiply the number of clock cycles by the clock period. This was set to 200MHz, which is long enough to access cache memory in one single clock cycle.

5.2 Real world example

By using the framework described in section 4 we conducted an exploration of the small-area/low-power/high-performance design space for the jpeg benchmark from the media benchmark suite [17].

The aim was to determine the optimal design of an embedded system (for example a digital camera application), based on a VLIW processor, which is able to compress and decompress jpeg pictures. The system has some requirements to be met, i.e. constraints in terms of codec performance, power consumption and area occupied.

The first step was to explore the space presented in Table 1 by means of the multi objective GA approach [1]. From the Pareto-set obtained after 100 generations, we eliminated the Pareto points which did not respect the constraints, reducing them to 175 points. As we expected, this number is still too high, so we applied the methodology presented in this paper to find the best subset, which represents all the Pareto solutions.

First of all we apply the FCM algorithm, with cluster number varying in the range [2,40], in order to determine the best number of clusters in which the Pareto set can be optimally partitioned. Results are summarized in Figure 2 and Table 2.

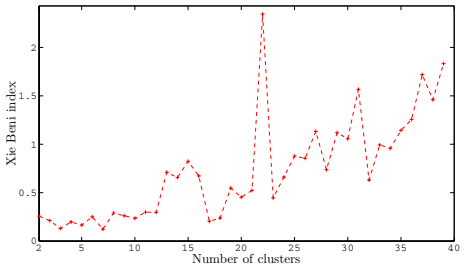

Figure 2: Xie-Beni cluster validation index values for jpeg Pareto set, with varying number of clusters.

According to Xie-Beni cluster validity measure, the best number of cluster is 8, with $XB_{index} = 0.1236$, however a good choice can be also 4 clusters, with $XB_{index} = 0.1311$.

In this paper we chose to analyze the 4-cluster solution so as to achieve greater clarity in representing the configurations obtained.

From Table 3 we observe that clusters are representative of three different scenarios. Low-cost configurations belong to Cluster 1,

Table 2: Xie-Beni cluster validation index with varying number of clusters.

Clusters N.	2	3	4	5	6	7
XB_{index}	0.2577	0.2117	0.1311	0.1979	0.1651	0.2499
Clusters N.	8	9	10	11	12	13
XB_{index}	0.1236	0.2902	0.2617	0.2356	0.2986	0.2962

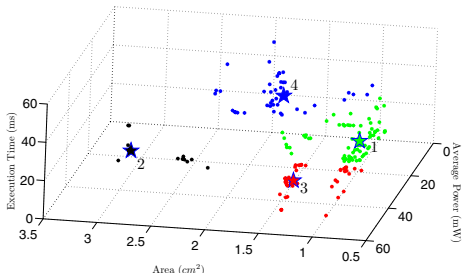

Figure 3: The four configuration which best represent the Pareto-set.

high performance configurations to Clusters 2 and 3, and low-power configurations to Cluster 4. If, for instance, the target system is destined for low-end market, the system configuration to be implemented is chosen from Cluster 1. So, the candidate configuration reported in the second column of Table 3, might be taken as starting design solution in the next phases of the design flow. However, there is a possibility that the candidate configuration is not a valid solution for the current design (e.g., due to the lack of some particular block configuration in the current design library). In this case, the fuzzy ranking approach allows to extract (from the same cluster) backup solutions which are valid for the current target design.

We also applied the approach using different weights for the objectives. The weights chosen are: 1 for area, 5 for average power and 2 for execution time. Table 4 shows that the number of optimal clusters is very small. As the Xie-Beni index values associated to solutions with two and three clusters are very close they could be considered equivalent. In this scenario, as we expected, the higher weight of the average power consumption leads to two partitions, which can be associated with the low power and high consumption profiles. An additional profile (medium consumption) is obtained when the Pareto set is partitioned in three clusters, as reported in Table 5.

6. CONCLUSION

In this paper we have presented a technique for extracting the most representative solutions of a Pareto-set. Although the technique is of general applicability, it has been illustrated and evaluated to assist the designer of an embedded system in choosing the optimal system configuration in terms of area, power consumption and performance. The large number of tradeoff configurations obtained during the DSE phase is the entry point for the decision maker, which uses classification techniques based on fuzzy logic to extract a subset that is representative of the Pareto-set. The reduced cardinality of this subset allows the designer to base his choice on a very limited number of possible configurations. If the solution suggested by the fuzzy decision maker, although optimal as regards the objectives being optimized, is unacceptable (e.g. because of the value of some parameters not considered during the exploration phase), the method proposed allows backup solutions with similar characteristics in the space of objectives to be extracted.

Table 3: The four clusters and the candidate configurations.

Cluster	1	2	3	4
Number of points	73	15	44	43
Candidate configurations				
Area (cm^2)	1.0096	3.0380	1.4282	1.8074
Average Power (mW)	17.7230	26.5450	36.0510	9.7102
Execution Time (ms)	14.2830	8.9674	7.5872	28.0850
IU	3	4	3	2
FPU	1	2	1	2
BU	1	1	1	1
MU	1	2	2	2
GPR	64	64	64	64
FPR	64	32	32	32
PR	64	64	64	32
CR	32	64	32	32
BTR	16	16	16	32
L1D-S	8192	8192	16384	16384
L1D-B	64	16	64	64
L1D-A	1	4	4	4
L1I-S	4096	4096	4096	4096
L1I-B	32	8	8	32
L1I-A	4	2	1	2
L2U-S	131072	524288	262144	131072
L2U-B	32	128	32	128
L2U-A	16	16	16	16

Table 4: Xie-Beni cluster validation index with varying number of clusters and weighted objectives.

Clusters N.	2	3	4	5	6	7
XB_{index}	0.0955	0.0967	0.1326	0.1952	0.1988	0.1456

Table 5: The power profiles and the candidate configurations.

Power profile	Low	Medium	High
Area (cm^2)	1.5104	1.1028	1.4174
Average Power (mW)	10.3978	23.1029	36.0510
Execution Time (ms)	24.7285	9.4007	7.4996

7. ADDITIONAL AUTHORS

Giuseppe Ascia and Vincenzo Catania (DIIT, Universita' di Catania, emails: {gascia,vcatania}@diit.unict.it)

8. REFERENCES

[1] G. Ascia, V. Catania, and M. Palesi. A multi-objective genetic approach for system-level exploration in parameterized systems-on-a-chip. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 24(4):635–645, Apr. 2005.

[2] G. Ascia, V. Catania, M. Palesi, and D. Patti. EPIC-Explorer: A parameterized VLIW-based platform framework for design space exploration. In *First Workshop on Embedded Systems for Real-Time Multimedia (ESTIMedia)*, pp. 65–72, Newport Beach, California, USA, Oct. 3–4 2003.

[3] P. D. Barba. A fast evolutionary method for identifying non-inferior solutions in multicriteria shape optimization of a shielded reactor. *The International Journal for Computation and Mathematics in Electrical and Electronic Engineering*, 20(3):762–776, Sept. 2001.

[4] J. C. Bezdek. *Pattern Recognition with fuzzy objective function algorithms*. Plenum Press, New York, 1981.

[5] J. C. Bezdek, J. Keller, R. Krishnapuram, and N. R. Pal. *Fuzzy Models and algorithms for pattern recognition and image processing*. Springer, 1999.

[6] B. Chakraborty, T. Chen, T. Mitra, and A. Roychoudhury. Handling constraints in multi-objective ga for embedded system design. In *VLSI Design*, pp. 305–310, 2006.

[7] L. N. Chakrapani, J. C. Gyllenhaal, W. mei W. Hwu, S. A. Mahlke, K. V. Palem, and R. M. Rabbah. Trimaran: An infrastructure for research in instruction-level parallelism. In *LCPC, LNCS 3602*, pp. 32–41, 2004.

[8] I. Das. On characterizing the "knee" of the Pareto curve based on normal-boundary intersection. *Structural and Multidisciplinary Optimization*, 18(2-3):107–115, Oct. 1999.

[9] I. Das. A preference ordering among various Pareto optimal alternatives. *Structural and Multidisciplinary Optimization*, 18(1):30–35, Aug. 1999.

[10] M. Eisenring, L. Thiele, and E. Zitzler. Conflicting criteria in embedded system design. *IEEE Des. Test*, 17(2):51–59, 2000.

[11] S. Eyerman, L. Eechhout, and K. D. Bosschere. Efficient design space exploration of high performance embedded out-of-order processors. In *DATE*, 2006.

[12] W. Fornaciari, D. Sciuto, C. Silvano, and V. Zaccaria. A sensitivity-based design space exploration methodology for embedded systems. *Design Automation for Embedded Systems*, 7:7–33, 2002.

[13] T. Givargis and F. Vahid. Platune: A tuning framework for system-on-a-chip platforms. *IEEE Trans. on Computer Aided Design of Integrated Circuits and Systems*, 21(11):1317–327, Nov. 2002.

[14] L. O. Hall, I. B. Ozyurt, and J. C. Bezdek. Clustering with a genetically optimized approach. *IEEE Trans. on Evolutionary Computation*, 3(2):103–112, 1999.

[15] A. Jain, M. Murty, and P. Flynn. Data clustering: A review. *ACM Computing Surveys*, 31(3), 1999.

[16] V. Kathail, M. S. Schlansker, and B. R. Rau. HPL-PD architecture specification: Version 1.1. Technical report, HP Laboratories, 2000.

[17] C. Lee, M. Potkonjak, and W. H. Mangione-Smith. MediaBench: A tool for evaluating and synthesizing multimedia and communications systems. In *International Symposium on Microarchitecture*, Dec. 1997.

[18] C. A. Mattson, A. A. Mullur, and A. Messac. Smart pareto filter: Obtaining a minimal representation of multiobjective design space. *Eng. Optimiz.*, 36(6):721–740, 2004.

[19] J. L. Nevins and D. E. Whitney. *Concurrent Design of Products and Processes: A Strategy for the Next Generation in Manufacturing*. Mcgraw-Hill, 1989.

[20] N. R. Pal and J. C. Bezdek. On cluster validity for the fuzzy c-means model. *IEEE Trans. on Fuzzy Systems*, 3:370–379, 1995.

[21] D. Patti and M. Palesi. EPIC-Explorer. http://epic-explorer.sourceforge.net/.

[22] H. A. Taboada and D. W. Coit. Data mining techniques to facilitate the analysis of the Pareto-optimal set for multiple objective problems. Technical Report 06-001, Department of Industrial and Systems Engineering, Rutgers University, Piscataway, NJ 08854, USA, 2002.

[23] J. R. Wang. Ranking engineering design concepts using a fuzzy outranking preference model. *Fuzzy Sets and Systems*, 119:161–170, 2001.

[24] X. L. Xie and G. Beni. A validity measure for fuzzy clustering. *IEEE Trans. Pattern Anal. Mach. Intell.*, 13(8):841–847, 1991.