A Knowledge-based System for Hardware-Software Partitioning*

M. L. López‡, C.A. Iglesias**† and J. C. López‡
‡ Dep. Ing. Electrónica, ETSIT, Univ. Politécnica Madrid, E–28040 Madrid, Spain, {marisa,lopez}@die.upm.es
** Dep. TSC e Ing. Telemática, ETSIT, Univ. de Valladolid, E–47011 Valladolid, Spain, {cif@tel.uva.es}

Abstract

This paper presents SHAPES, a tool for hardware-software partitioning. It is based on two main paradigms: the implementation of the partitioning tool by means of an expert system, and the use of fuzzy logic to model the parameters involved in the process.

1 Introduction

Hardware-software partitioning deals with the assignment of parts of a system description to heterogeneous implementation units: ASICs, standard or embedded processors, memories, etc. This is a key task in HW-SW co-design, because the decisions that are made at this time impact directly on the performance and cost of the final implementation. To achieve such a difficult labor, complex algorithms have been developed in different co-design environments [2, 1, 3, 4]. In this paper artificial intelligent techniques are proposed for the partition of complex systems.

SHAPES (Software-HArdware Partitioning Expert System) is a fuzzy expert system which provides an easy way to address HW-SW partitioning. This tool considers two important aspects of general optimization problems: the possibility of dealing with imprecise and uncertain values (by means of the definition of fuzzy magnitudes), and the use of the expertise of the system designer in the decision making process.

2 Rationale for Hw-Sw Partitioning using a FuzzyExpert System

In the traditional design style, heterogeneous system development was fully characterized by an initial partitioning phase performed manually by the system designer. It was based on different pieces of estimation and basically the designer knowledge. To automate this partitioning phase, it is necessary to mimic the way a skilled designer performs this step.

We propose a different approach based on expert systems technology, that uses the designers' knowledge acquired in manual partitioning, in the form of if-then rules. Since this knowledge is imprecise in nature, we use fuzzy logic [5] to model the variables of the rules.

3 Partitioning Model

The input to the partitioning process is an execution flowgraph which comes from the initial system specification. In this graph (directed and acyclic), nodes stand for tasks and edges represent data and control dependencies (coarse granularity).

Every graph node i is labeled with additional information: hardware area (ha_i), hardware execution time (ht_i), software execution time (st_i) and the average number of times the task is executed (n_i). Edges have also associated a communication value ($comm_{ij}$) obtained from: the transfer time ($t_{trans_{i,j}}$), the synchronization time ($t_{synch_{i,j}}$) and the average number of times the communication takes place n_{ij}.

The target architecture considered consists of one processor running the software, one ASIC and a shared memory accessed through a common bus. The output of the partitioning tool is not only an assignment of blocks, but also their scheduling and the communication values produced in the interface.

4 SHAPES Architecture

The architecture of SHAPES is described in figure 1. This blockdiagram describes the design flow followed by the expert system, whose modules will be explained in the following sections.

4.1 Classification Module

The first step in SHAPES is to determine which blocks of the initial specification are more suited to be implemented in special purpose hardware or as software running on a standard processor. This step is performed by a rule based classifier module.

The crisp output of the classifier module is the set of input tasks ordered by their implementation degree: value 0 stands for hardware and 1 for software. If no rule is activated for a given input task, the implementation degree is unknown (0.5). Furthermore, in that case, the tool can be configured to ask the designer an

*This work has been funded by projects HADA (CICYT TIC97-09 28) and BELSIGN (HCM CHRX-CT94-0459)

†This research was done while the author was visiting the Dep. Ing. Sistemas Telemáticos (Univ. Polit. Madrid).

0-8186-8359-7/1998/$10.00 © 1998 IEEE

implementation value for the specific task and increase the knowledge base.

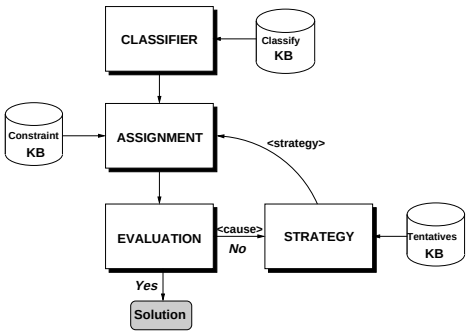

Figure 1: Block Diagram of the Expert System

4.2 Assignment Module

This module allocates the system tasks to HW or SW taking as inputs the ordered set of tasks and the system constraints (max. hardware area, A, and max. execution time, T). The output is an implementation *threshold* which determines the HW-SW boundary.

The knowledge base has rules that characterize the specification constraints to determine a reliable partition. The output data (a crisp threshold) is obtained after estimating the "hardness" of the specification requirements: how critical (or not) the constraints are regarding the extreme values of the system.

4.3 Partition Evaluation

The goodness of the first partition obtained after assignment is computed by the evaluation module. In this module three parameters have been considered:

- The estimation of the needed hardware area.
- The scheduling of the assigned task graph, which gives the final execution time.
- Computation of the final communication cost.

For the tasks assigned to hardware, a rough area estimation is performed. Area estimates correspond just to one of the possible points of the design space. Task scheduling is implemented by means of a list-based algorithm that gives as output the final execution time for that partition and the communication cost.

Once these parameters have been calculated, the evaluation module determines if the proposed partition is feasible. Extra information is then originated: if the constraints have not been met, the degree of violation, otherwise the available margin. The strategy module must start working to refine the partition.

4.4 Strategy Module

When a partition is not feasible, another partition must be proposed. There are two non-exclusive alternative strategies that are selected taking into account a rules base of previous experiences and some results of the evaluation module. These alternatives are:

1. To migrate tasks from hardware to software or vice versa according to the constraints. This is carried out by tuning the threshold.
2. To improve the partition attending to minimize the communication on the HW-SW boundary.

The output of this module is fed back to the assignment module, and the partitioning process starts again. This iteration is performed until either a feasible partition is found or the strategy module decides that no solution can be found with the initial constraints. In this case, SHAPES asks the designer to change the system requirements and gives advice on how to do that. Otherwise, no solution can be given.

4.4.1 Communication based Reordering

Attending to the communication penalty, a new configuration can be proposed by reordering according to the *Communication Rate factor*. This factor considers the communications of the task i with the tasks placed in the other partition, being relative to the total number of communications of the task.

The tasks affected by communication penalty update their implementation value. If the communication factor is high and the task is close to the threshold, the kind of implementation can be even changed.

5 Conclusions and Future Work

This paper has shown that artificial intelligence techniques are attractive alternatives to implement system design tools.

The advantages of our approach are the declarative specification of the knowledge bases, that contain the expertise of the designers and allow its modification by the user. This tuning of the system rules cannot be carried out following traditional approaches. In addition, the total computation time is very short.

References

[1] R. Ernst, J. Henkel, and T. Benner. Hardware-Software Cosynthesis for Microcontrollers. *IEEE Design & Test of Computers*, pages 64–75, Dec 1993.

[2] R. K. Gupta and G. De Micheli. HW-SW Cosynthesis for Digital Systems. *IEEE Design & Test of Computers*, pages 29–41, 1993.

[3] A. Kalavade and E. A. Lee. The extended partitioning problem: Hardware/software mapping, scheduling and implementation-bin selection. *Journal of Design Automation of Embedded Systems*, 2(2):125–164, March 1997.

[4] F. Vahid and D. D. Gajski. Clustering for Improved System-level Functional Partitioning. In *Proc. ISSS'95*, pages 28–33, 1995.

[5] L. A. Zadeh. Outline of a new approach to the analysis of complex systems and decision processes. *IEEE trans. on System Man and Cibernetics*, pages 28–44, Jan 1973.