PAR-SYNTHESIS OF MULTIPROCESSORS PARALLEL SYSTEMS

Mieczysław Drabowski
Cracow University of Technology
Warszawska 24 Kraków 31-155
Poland

SUMMARY
The paper includes a new model of par-synthesis (coherent co-synthesis), this is characteristic for the problems of computer system design. Optimal task scheduling, partition at resources, allocation task and resource are basic problems in high-level synthesis of computer systems. The goals of synthesis of computer systems are to find an optimum solution satisfying the requirements and constraints enforced by the given specification of the system. The following criteria of optimality are considered: cost of system implementation and operating speed. A specification describing a computer system may be provided as a set of interactive tasks (processes, functions). The synergic solution is a result of cooperation between the scheduling algorithms and the algorithm responsible for resource partition. Par-synthesis may have a practical application in developing tools for computer aided rapid prototyping of such systems.
Index Terms:
partition at resources; task scheduling; co-synthesis; par-synthesis; schedule length;

1. INTRODUCTION

The goal of high-level system synthesis is the optimal choice of system resources and the optimal execution of all functions which are performed by the embedded system. The system which is being designed has to satisfy all given requirements and constraints. They may be presented as a set of tasks with given characteristics and they should be carried out by the system resources. System, which is being constructed, should be optimal in the sense of accepted criteria, such as the cost of implementation, its operating speed, reliability or power consumption. To make a successful computer system synthesis the following problems should be resolved: resources identification, i.e. functions partition on the part carried out by hardware and part done by software, system tasks scheduling, system tasks and resources allocation. For the efficient system synthesis a coherent and multi-criteria solution is required to the problems of resources partition, tasks scheduling, tasks and resources allocation. The coherent approach for computer systems synthesis was presented in [3]. The classical process co-synthesis [2, 5] – hardware and software synthesis – consists of the following general stages (Fig. 1).

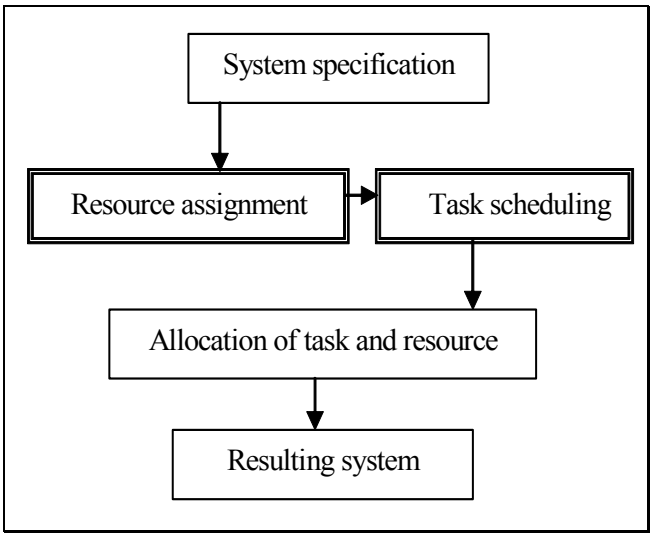

Fig. 1 The process synthesis

1. Specification of the designed system in terms functional and behavioural – requirements and limitations analysis. The system description in a high-level language, abstracting from the physical implementation.
2. Selection of the system architecture and control.
3. Resource distribution – architecture development.
4. Task scheduling – system control development.
5. Assigning the system functions to the architecture elements – generating the system modular architecture, control adaptation and the whole system integration.

In par-synthesis approach a combined search for optimal partition resources and optimal tasks scheduling occur (Fig. 2). As optimality criterion for resources partition - a minimization of system cost was assumed and as optimality criterion for tasks scheduling (schedule length) - a minimization of time execution of all tasks by the system was assumed.

The suggested coherent analysis consists of the following steps [4]:

Manuscript received October 5, 2008
Manuscript revised October 20, 2008

1. Specification of requirements for the system to be designed and its interactions with the environment.
2. Specification of tasks, including evaluation of task executive parameters using available resources (e.g. execution times).

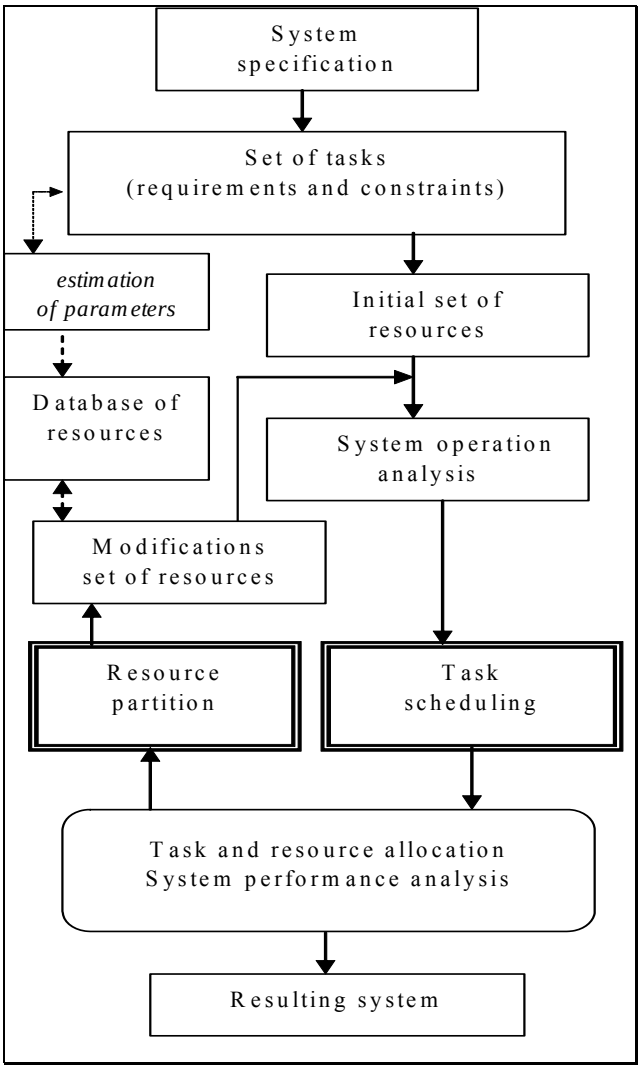

Fig. 2 The process of coherent synthesis of computer system

3. Assuming the initial values of resource set and task scheduling – initial resource set and task schedule should be admissible, i.e. should satisfy all requirements in a non-optimum way.
4. Task scheduling and resource partitioning.
5. Evaluating the operating speed and system cost, multi-criteria optimization.
6. The evaluation should be followed by a modification of the resource set, a new system partitioning into hardware and software parts and an update of tasks schedule (step 4).

Iterative calculations are executed till satisfactory design results are obtained – i.e. optimal (or sub-optimal) system structure and schedule. The designed system should be fast and cheap.
The problems scheduling tasks to minimize schedule length and assignment resources to minimize cost belong to NP-complete class [1]. We shall present mete-heuristic algorithms which were applied: Tabu Search algorithm and Genetic algorithm.

2. TABU SEARCH ALGORITHM FOR PAR-SYNTHESIS

In Tabu Search algorithm for coherent optimization of resources partition and tasks scheduling the initial solution is generated quasi-randomly or by means of other heuristic algorithms. Obtained in this way initial solutions are not optimal but they meet all requirements specified in input date and further optimization is done by Tabu Search algorithm. In the algorithm, which is a coherent approach to tasks scheduling optimization and resources partition problems, the movement is defined as a transfer of specified part of task (or whole task), which is being carried out on one of processors, to another processor and to other position in schedule. In comparison with tasks scheduling algorithm, the difference is that movements, without tasks and position specifying are possible, so exchange of processors for other once in the resource set is possible. Tasks, which are carried out on a swapped processor, are transferred to a new processor taken from resource set or other active processors. Such a situation is likely to happen when task are dependent and the new processor is slower than the previous one or in the case of problems with additional resources. In the case of coherent approach to task scheduling and resources partition functional surrounding as well as neighborhood is double set, separate for task scheduling and resources partition. New movement creation is partially random. First, movements for tasks scheduling are generated, after they have been made and after some possible returns to the best so far solution found, movement for resources partition problem is generated. This kind of solution permits a coherent resources partition and various task schedules verification for these tasks resources partitions. Short term memory consists of all recently made movements. Apart from modification characterized in previous chapters, in this case new modifications are introduced. Tabu list associated with resources partition has mechanisms which allow for combining each element from the list of movements for resources partition with the list of movements for tasks scheduling. Later on, the list searching is only limited to finding the movements for dividing and to checking for this movement whether the movement for scheduling exists on the joined to it list. It is

a simple mechanism which allows efficient and effective retaining of movements in memory and searching for them. A list of generated solutions was used as a long term memory. Pieces of information obtained on the basis of values of these variables are as follows: the total number of iterations performed from the beginning of algorithm action, the number of iterations without acquiring a better solution, the number of recent returns to the initial solution, the present value of parameter which describes diversity in neighborhood searching. The criteria, which are being considered, are concurrently the total cost and task scheduling length (C_{max}).

3. GENETIC ALGORITHM FOR PAR-SYNTHESIS

In genetic algorithm [4] with Boltzmann tournament selection strategy to eliminate solution convergence in genetic algorithms, we use data structures which ensure locality preservation of features occurring in chromosomes and represented by a value vector. Locality is interpreted as the inverse of the distance between vectors in an n-dimension hypersphere. Then, crossing and mutation operators are data exchange operations not between one-dimensional vectors but between fragments of hyperspheres. Thanks to such an approach, small changes in a chromosome correspond to small changes in the solution defined by the chromosome. The presented solution features two hyperspheres:

- Task hypersphere – two-dimensional, representing the task graph structure. Each of the vertexes is defined by two coordinates: an indicator obtained through topological sorting (the tasks are "closest" if one of them is a direct successor of the other), and an indicator calculated from the BFS algorithm (parallel tasks are equally distant from the beginning of the graph).
- Resource hypersphere – three-dimensional, representing the dependencies of resource features. Each of the resources may be defined by the following coordinates - cost, speed and power consumption.

The solutions sharing the same allocations form the so-called clusters. The introduction of solution clusters separates solutions with different allocations from one another. Such solutions evolve separately, which protects the crossing operation from generating defective solutions. There are no situations in which a task is being allocated to a non-allocated resource. Solution clusters define the structures of the system under construction (in the form of resources for task allocation). Solutions are the mapping of tasks allocated to resources and task scheduling. During evolution, two types of genetic operations (crossing and mutation) take place on two different levels (clusters and solutions).

A population is created whose parameters are: the number of clusters, the number of solutions in the clusters, the task graph and resource library. For the synthesis purposes, the following criteria and values are defined: optimization criteria and algorithm iteration annealing criterion if solution improvement has not taken place, maximum number of generations of evolving solutions within clusters, as well as the limitations - number of resources, their overall cost, total time for the realization of all tasks, power consumption of the designed system and, optionally, the size of the list of the best and non-dominated individuals.

4. RESULTS OF COMPUTATIONAL EXPERIMENTS

4.1. Comparison of par-synthesis (coherent) and co – synthesis (incoherent)

As mentioned earlier, the problems presented in this work belong to NP-complete class. We shall present the final results of the computer experiments obtained with non-coherent and coherent approach. Meta-heuristic algorithms were applied: genetic algorithms. The optimality criteria used in our experiments were minimum cost of the system and minimum processing time. Analyzing the presented results one may conclude that the par-synthesis obtains better solutions in terms of the implementation costs criterion, as well as in terms of the operating speed criterion for the designed implementations. These results are collected in the charts as depicted in Fig. 3 and Fig. 4.

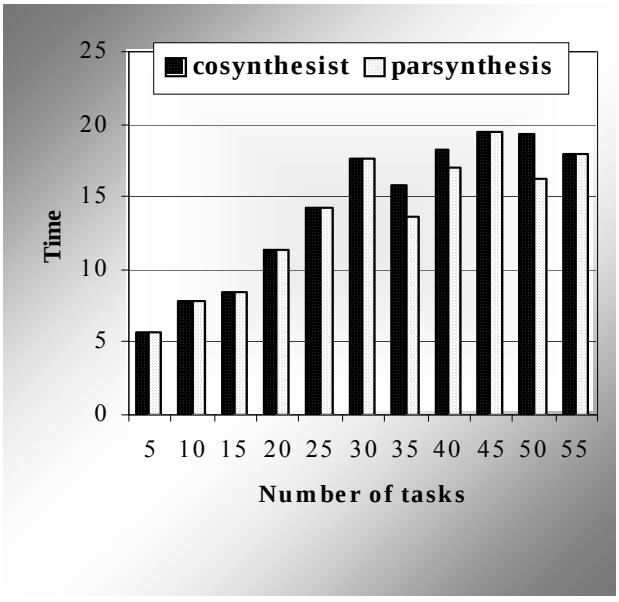

Fig. 3 Minimum processing time

As for the time minimization both algorithms produce similar values of the cost for all sample task sets. Par-synthesis algorithms improve the task processing time substantially, in particular for graphs with more then 30 tasks. For instance, 15% improvement of total processing time was obtained for graphs with 50 tasks.
The diagram presenting the relationship between the costs and the number of tasks (Fig. 5) suggests that the solutions obtained by coherent algorithms are considerably cheaper than the ones obtained by a non-coherent algorithm. A coherent algorithm obtains similar task time performance times in cheaper structures that the ones defined by the non-coherent algorithm. This is particularly visible for the case of graphs with 45 tasks.

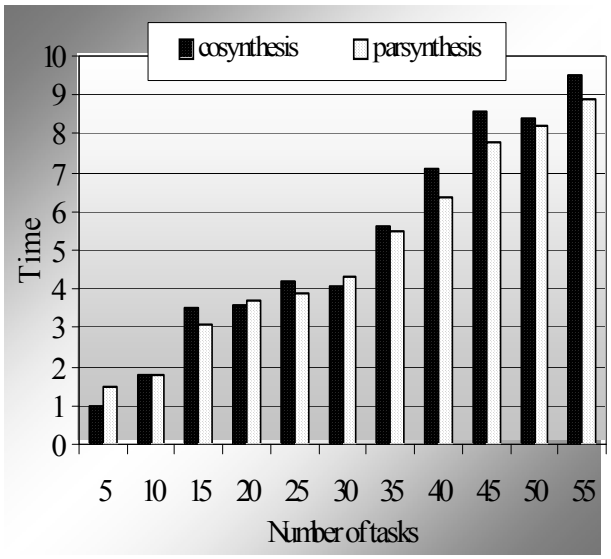

Fig. 4 Minimum system cost

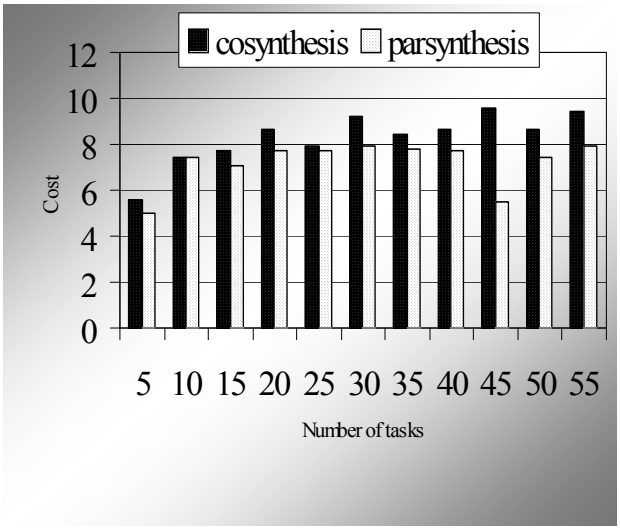

Fig. 5 Minimum system cost

Conclusion
Coherent synthesis gives in many cases better results that incoherent synthesis. Enabling the algorithms to verify and divide the resources once more during the course of operation, positively influence the result.

4.2. The comparison genetic algorithm with Tabu Search algorithm in par-synthesis

The results obtained by genetic algorithm and Tabu Search algorithm will be presented on the basic of the following example.

4.2.1. Minimize schedule length
For schedule length optimization, the algorithms were given the following resources: memory (< 10 GB), storage (<10 TB), processors (universal and identical < 6), dedicated (< 5).
The above example shows, that genetic algorithm has the advantage over Tabu Search algorithm in the wide interval. The first one gives better scheduling in many cases – Fig. 6

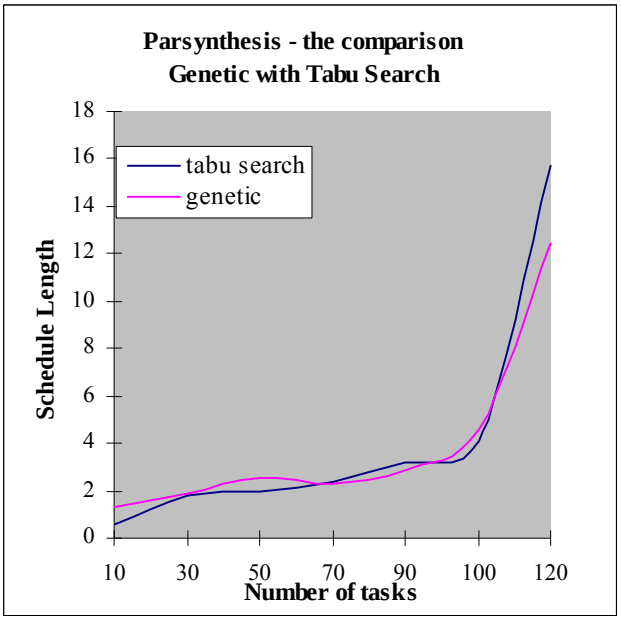

Fig. 6 Comparison genetic and tabu search algorithms for par-synthesis. Dependence: the schedule length and number of tasks (non-preemptive and dependent tasks).

4.2.2. Minimize cost

For cost optimization, the algorithms were given the following resources: memory (< 10 GB), storage (<10 TB), processors (universal and identical < 6), dedicated (< 5).

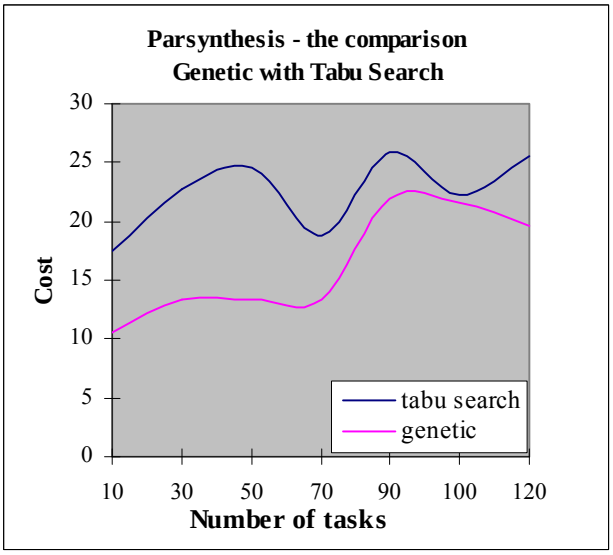

Fig. 7 Comparison genetic and tabu search algorithms for par-synthesis. Dependence: the cost and number of tasks (non-preemptive and dependent tasks).

The genetic algorithm gives much better solution – Fig. 7.

4.2.3. Dependence of precedence constraints

The algorithms were given the following resources: memory (< 10 GB), storage (<10 TB), processors (universal and identical < 6), dedicated (< 5).

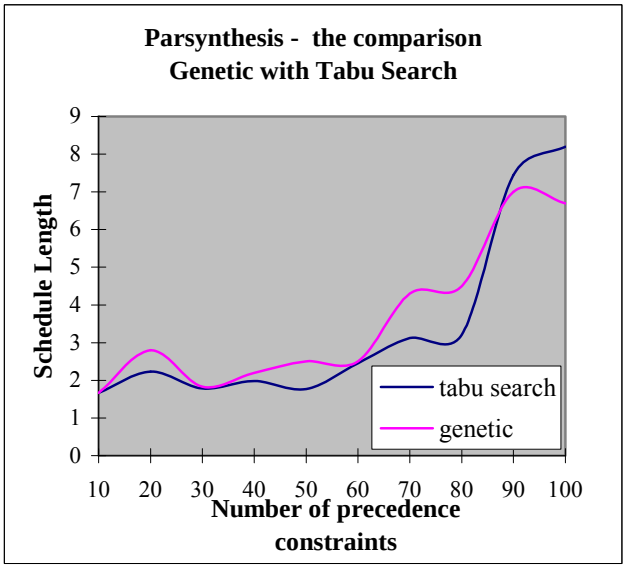

Fig. 8 Comparison genetic and tabu search algorithms for par-synthesis. Dependence: the schedule length and number of precedence constraints.

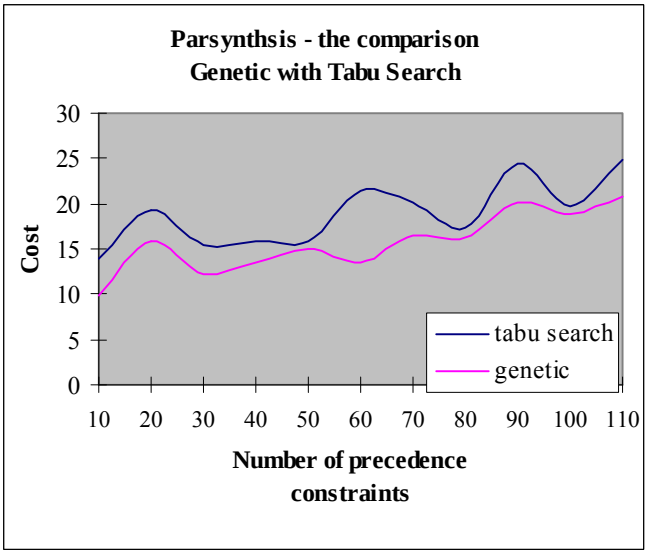

Fig. 9 Comparison genetic and tabu search algorithms for par-synthesis. Dependence: the cost and number of precedence constraints.

The examples (Fig. 8 and Fig. 9) shows dependence cost, schedule length and number of precedence (i.e. number edges in graphs of tasks).

5. CONCLUSIONS

In this paper an attempt of comprehensive and coherent approach to high-level system synthesis is presented. The following system optimization criteria are accepted: minimum operating time and minimum cost. Results of the conducted experiments revealed that a coherent resource partitioning and task scheduling can provide better solutions that those obtained with separated (non-coherent) resource partitioning and task scheduling. The synergic solution is a result of cooperation between the scheduling algorithms and the algorithm responsible for resource partition. The model presented for coherent co-synthesis and the first encouraging experimental results allow a further research in this area [4]. One may also specify additional optimality criteria, e.g. minimum power consumption of the designed system (which is particularly significant for built-in and mobile systems). For the proposed system's relevance to real systems, one should take into account the processes of communication between resources and tasks, preventing resource conflicts, as well as extend the available resources sets, for example by programmable and configurable structures.
The paper describes genetic and Tabu Search algorithms and their implementation for coherent resource partition and task scheduling. The coherent approach in the control generates common and interdependent solutions regarding the system structure (type and configuration of the selected resources), as well as the scheduling of tasks ran on those resources. In the presented approach, the cost of resources (system cost), the time of completing all tasks (system speed) and the communication of the system are optimized. The coherent algorithm yields much (up to 30%) better solutions, which is proved by analytical experiments.

ACKNOWLEDGEMENT

This work was supported by the Polish Ministry of Science as a 2007-2010 research project.

REFERENCES

[1] J. Błażewicz, M. Drabowski, J. Węglarz, "Scheduling multiprocessor tasks to minimize schedule length", IEEE Trans. Computers C-35, No. 5, pp. 389-393, 1986.
[2] Dick R. P., Jha N. K., "MOGAC: A Multiobjective Genetic Algorithm for Hardware-Software Cosynthesis of Hierarchical Heterogeneous Distributed Embedded Systems". In: IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 17, no. 10, Oct. 1998, pp. 920 – 935.
[3] R.P. Dick, N.K. Jha, "CORDS: Co-Synthesis of Reconfigurable Real-Time Distributed Embedded Systems", Proc. Intern. Conf. Computer-Aided Design, p. 62 – 68, 1998.
[4] M. Drabowski, "A Genetic Method for Hardware-Software Par-sythesis", IJCSNS, VOL. 6 No.11, 2006.
[5] Oh. Hyunok, Ha Soonhoi, *"Hardware-software cosynthesis of multi-mode multi-task embedded systems with real-time constraints",* Proceedings of the IEEE/ACM Conference on Hardware Software Codesign, pp. 133-138, 2002.

Mieczyslaw Drabowski, Assistant Professor of Department of Computer Engineering, Faculty of Electrical and Computer Engineering, Cracow University of Technology, received the M. Sc. degree in automatic control and communication from AGH University of Science and Technology, graduated mathematic from Jagiellonian University in Krakow and received the Ph. D. degree (with honors) in computing science from Poznan University of Technology, in 1977, 1979 and 1986, respectively. He has eighteen years of industrial experience in design and engineering of disc drivers and computers and in software engineering: a senior programmer in the Center of Computing in PGGN (1978-1981), a senior engineering specialist and head of Computers Engineering Laboratory of MERA-KFAP (1979-1993) and an expert-computer scientist in ARKA PRESS S.A., FILM S.A., TRAS PRESS S.A. (1993-1996), the deputy director of the Center of Information Technology, Cracow University of Technology in years 1999-2003.Currently he is member of several editorial boards, among others Scientific Journals International, International Association for Development of the Information Society (IADIS), and International Association of Science and Technology for Development (IASTED) on Artificial Intelligence and Soft Computing. His research interests include schedule, assignment and allocation for tasks and resources, dependable and fault tolerant systems, artificial intelligence, operating systems and software engineering, author and co-author of 3 monographs and over 60 papers in major professional journals and conference proceedings.
He lectures at the Cracow University of Technology, teaching undergraduate and graduate courses and supervising master's and engineer's theses (over 80), he has lectured and conducted seminars at the AGH University of Science and Technology, the Pontifical Academy of Theology in Cracow, Tischner European University, Kielce University of Technology, School of Economics and Law in Kielce. Dr. Drabowski is a member of the council of the Polish Information Processing Society.