STUDIA INFORMATICA
Nr 1/2(7) Systemy i technologie informacyjne 2006

The Tabu Search approach in coherent co-synthesis of multiprocessors systems

Mieczyslaw Drabowski[1], Krzysztof Czajkowski[1]
[1] Cracow University of Technology, Faculty of Electrical and Computer Engineering, Warszawska 24, 31-155 Krakow, Poland

Abstract. This paper presents the use of Tabu Search algorithm for solving the problems of coherent synthesis of multiprocessor computer systems. The paper includes a coherent solution of both optimization of partition resources and optimization of tasks scheduling. This publication shows results of computational experiments for different instances of system synthesis problems.

Keywords. Synthesis of system, coherent, identification resources, task scheduling, NP-complete problem, heuristic algorithm, tabu search algorithm.

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 multiprocessor 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.

A specification describing a computer system may be provided as a set of interactive tasks. In any computer system certain tasks are implemented by hardware. The basic problem of system synthesis is partitioning system functions due to their hardware and software implementation. The goal of the resources assignment is to specify what hardware and software resources are needed for the implementation and to assign them to specific tasks of the system, even before designing execution details. In the synthesis methods used so far ([4], [13], [16]) software and hardware parts are developed separately and then composed, what results in cost increasing and decreasing quality and reliability of the final product. Task scheduling is one of the most important issues occurring in the synthesis of operating systems responsible for controlling allocation of tasks and resources in computer systems.

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, tasks scheduling, 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 [6], [7], [11]. In this approach a combined search for optimal partition resources and optimal tasks scheduling occur in Figure 1. As optimality criterion for resources partition - a minimization of system cost was assumed and as optimality criterion for tasks scheduling - a minimization of time execution of all tasks by the system was assumed.

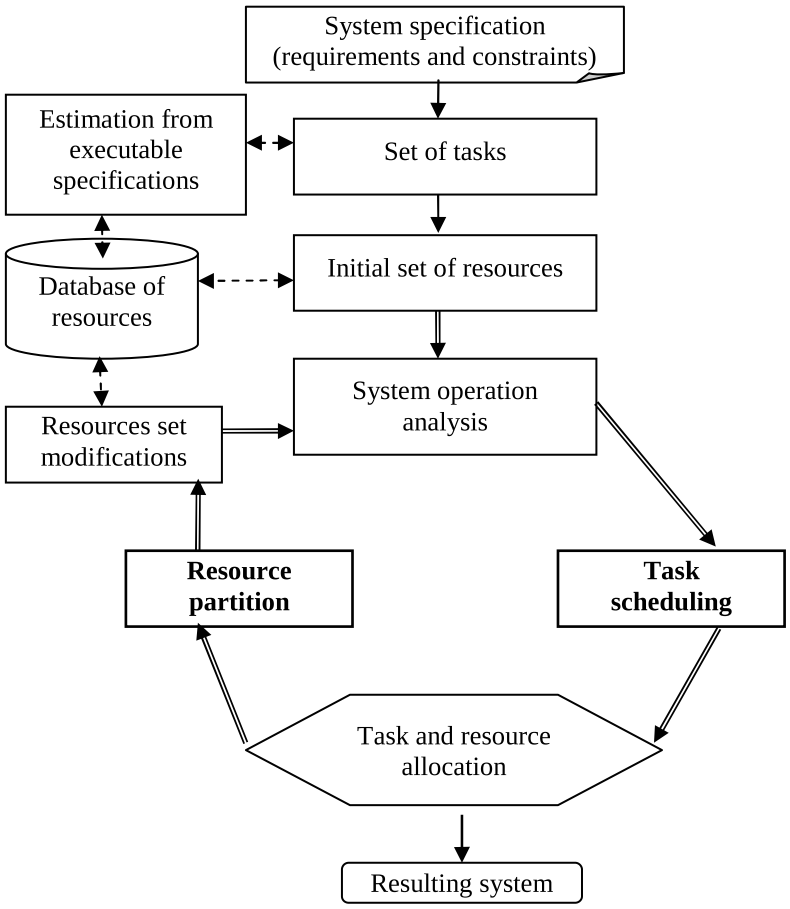
System specification (requirements and constraints)
Estimation from executable specifications
Set of tasks
Database of resources
Initial set of resources
System operation analysis
Resources set modifications
Resource partition
Task scheduling
Task and resource allocation
Resulting system

Figure 1. The process of coherent synthesis of computer system

Synthesis of multiprocessor system consists of resource identification and task scheduling problems that are both NP-complete. Algorithms for solving such problems are usually based on heuristic approaches. The objective of this paper is to present the concept of combined approach (based of Tabu Search rule) to the problem system synthesis, i.e. a coherent solution to task scheduling and resource assignment problems.

2 An example of the use of Tabu Search method

The general idea of Tabu Search method (TS) is presented in [14]. The algorithm course is as follows:

```
TS:
generate a single initial solution
set tabu list length, t
until termination criteria is met                              {
   for neighborhood i = 1 to N      {
       make a move to a neighboring solution
       evaluate
       record in ranked order, j = 1 to N
       ++ i                                      }
   for j = 1 to N                    {
       if solution j is not tabu, break to GO
       else if solution j is better than best so far, break to GO
       ++ j                                      }
  GO: add solution j to tabu list
  if tabu list length > t, remove oldest solution from it
  if solution j is better than best so far
  best so far = solution j                                     }
return best so far
```

Below there is presented a simple example of the use Tabu Search method [8] as a solution to the classic problem of optimizing preemptable and independent tasks scheduling, to minimize the tasks schedule length (C_{max}) on identical processors (for this problem are exact polynomial algorithms, which use McNaugton's rule [3]). It is assumed that 2 identical processors and 5 independent tasks are available. The following times (in conventional units) of individual tasks are accepted: $t_1 = 1$, $t_2 = 1$, $t_3 = 1$, $t_4 = 2$, $t_5 = 1$.

The movement is described over 6 parameters: λ^1, λ^2, λ^3, λ^4, λ^5, λ^6, where

λ^1 – stands for the number of processor, on which the chosen task to transfer is being carried out,

λ^2 – the number of position in scheduling, on which the chosen task to transfer is located,

λ^3 – the number of task to transfer during movement,

λ^4 – the number of processor, onto which the tasks will be transferred,

λ^5 – the number of position in scheduling, onto which the task will be transferred,

λ^6 – the number of task, which is located on λ^5 position and λ^4 processor.
First schedule was shown in Figure 2a, for iteration 0.

The realization of algorithm and the Gantt chart for the tasks system by iterations from 1 to 10 are showing in Figure 2b.
On the tabu list there are movement attributes: name tag of task to transfer in the movement (λ^3) and the position, onto which this task will be transferred (λ^5).

No. of iteration	Processor	Position in schedule				
		1	2	3	4	5
0	P1	1		4		5
	P2	3	2		4	
1	P1			4		5
	P2	3	2	1	4	
2	P1	4		4	5	
	P2	3	2	1		
3	P1	4		4	5	1
	P2	3	2			
4	P1	4	4	5	1	
	P2	3	2			
5	P1	4	4	5	1	
	P2	2	3			
6	P1	4	4	5	1	
	P2		3		2	
7	P1	4	4	5	1	
	P2	3			2	
8	P1	1	4	5	4	
	P2	3			2	
9	P1	1	4	5	4	
	P2	3	2			
10	P1	1	4	5		
	P2	3	2	4		

Figure 2a. A Gantt chart for the task system

The example assumes that the length of tabu list equals 5 and the following initial solution is accepted.

No.	Move $\lambda^1, \lambda^2, \lambda^3, \lambda^4, \lambda^5, \lambda^6$	Tabu list
1	P1,1,1 P2,3,*	(1,3)
2	P2,4,4 P1,1,*	(4,1) (1,3)
3	P2,3,1 P1,5,*	(1,5) (4,1) (1,3)
4	P1,3,4 P1,2,*	(4,2) (1,5) (4,1) (1,3)
5	P2,1,3 P2,2,2	(3,2) (4,2) (1,5) (4,1) (1,3)
6	P2,1,2 P1,4,*	(2,4) (3,2) (4,2) (1,5) (4,1)
7	P2,2,3 P2,1,*	(3,1) (2,4) (3,2) (4,2) (1,5)
8	P1,4,1 P1,1,4	(1,1) (3,1) (2,4) (3,2) (4,2)
9	P2,4,2 P2,2,*	(2,2) (1,1) (3,1) (2,4) (3,2)
10	P1,4,4 P2,3,*	(4,3) (2,2) (1,1) (3,1) (2,4)

Figure 2b. The example of algorithm realization

The solution for schedule length is equal 3 and is optimum.

3 Tabu Search algorithm for resources partition problems

For resources partition problem the initial solution is generated quasi-randomly or by means of other heuristic algorithms. It is the first correct solution and it is not optimal. Further optimization is conducted by Tabu Search algorithm. For the solution of this problem, it is assumed that a movement is the swapping of one of resources for the other with simultaneous transfer of all performing tasks, which use this resource.

Movement is realized as follows:

- from resources so far allocated to tasks, the resource to be swapped is drawn (let it be PI resource),
- from the set of available resources, a resource is drawn (let it be PO resource),
- the tasks allocated to PI resource will be transferred and allocated to PO resource.

In the case, when empty processor (0) is drawn from the resources set, all tasks, which are being carried out on the processor chosen at the first step, will be removed to other processors. Movement parameters are stored in the short term memory, i.e. on the Tabu list. The optimization criteria are cost and speed of solution.

4 Tabu Search algorithm for tasks scheduling problems

For tasks scheduling problem the initial solution is generated quasi-randomly or by means of other heuristic algorithms and further optimization is conducted by Tabu Search algorithm. Initial solutions obtained meet all requirements, which are specified for input data. In the algorithm, which resolves optimization problems of tasks scheduling, the movement is defined as task transfer (or given part of task) carried out on one of the processors onto other processor and onto a different position in the schedule. In the case when we want to transfer selected task (or its part) onto the position where there is another task (or its part), the tasks (or their parts) will be swapped. Such a movement was described unambiguously by following six attributes.

1. The number of task to transfer during a movement.
2. The number of processor, on which a selected task is being carried out.
3. The number of task position from pt. 1 in the schedule on the processor from pt. 2.
4. The number of position in the schedule, onto which we want to transfer a selected task.
5. The number of processor, onto which we want to transfer a selected task.
6. The number of task, which is located in position from pt. 4 and the processor from pt. 5.

For nonpreemptable tasks scheduling problems during movement – whole tasks are transferred and for preemptable tasks – their selected parts are transferred. To specify the above mentioned attributes is necessary for movement making in a given schedule. But Tabu Search method alone does not determine which factors should be taken into consideration in defining movements, what features they should have and what criteria for comparing them are. In task scheduling problems, for various types of input data, movement definition and individual parts of it have significantly different meaning for determining whether a given movement is identical to another one.

For example, in scheduling unit-time and nonpreemtable tasks on parallel identical processors, the position of given task is insignificant, however, when task are preemptable, the position, on which a given task will be transferred, contains important information, determining subsequent movements and the quality of obtained solution. In the developed approach, the function which compares movements is flexible, its operation is dependent on parameters defined by input data inserting. In tasks scheduling problems neighbourhood is a set with a great number of elements. Moreover, the ways of generating them differ from one another depending on the type of problem. Obtaining complete content of environment in the middle of every iteration causes, significant extending of algorithm work, therefore determining a new movement is partially random. At the beginning the task to be moved is drawn; next, processors are chosen on which random task is carried out.

Among many processors only one is drawn. In the case of preemptable task only a part of task to be moved is drawn. The target place of task moving is determined by drawing a random processor and place in the schedule. Verifying whether a given movement belongs to the environment happens during its generating, but complete surroundings content is never known. The results of this solution to the way of generating movement are twofold. In states, when actual solution is distant form optimal one and many possible movements exist, algorithm quickly approaches the optimal solution. However, if optimal solution is close and only few possible movements exist, many wrong moves are generated and algorithm significantly slows down.

Short term memory consists of all newly made movements. For the sake of constant number of elements on the list, each new element added causes automatic removal of the element from the end of the list. A list of generated solutions was used as long term memory. Information obtained on its basis: total number of iterations executed from the beginning of algorithm action, number of iterations obtained without better solution, the number of recent returns to initial solution, present value of parameter which determines diversity in environment searching. To accelerate the work of algorithm the initial solution and the best solution so far, are retained during algorithm action. At the Tabu Search algorithm implementation for tasks scheduling problems the considered criterion of quality is minimizing tasks scheduling length.

5 Tabu Search algorithm for coherent optimization resources partition and tasks scheduling

In Tabu Search algorithm for coherent optimization of resources partition and tasks scheduling, like in both previous algorithms, 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 similarly to Tabu Search algorithm for resources partition, 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 neighbourhood is double set, separate for task scheduling and resources partition. New movement creation is partially random. The mechanisms are parallel to those in programs for tasks scheduling problems. 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 neighbourhood searching. The criteria, which are being considered, are concurrently the total cost and task scheduling length.

6 Results of computational experiments

As mentioned earlier, the problems presented in this work (resource partitioning, task scheduling) belong to NP-complete class. We shall present the final results of the computer experiments obtained with coherent and non-coherent approach.

Algorithm Tabu Search it is a meta-heuristic algorithm, which were applied in computational experiments [10]. The optimality criteria used in our experiments were minimum cost of the system and minimum processing time (task scheduling length C_{max}).

6.1 The influence of tasks preemptability on the quality of computations

All the tests were executed into independent tasks, without additional resources. The number of processors in database of resources is 10, maximum number of processors, which algorithm can use is 5, and the cost of memory equaled 0.2 was taken into consideration.

Constraints of system: maximum cost is 50 units and maximum operating time is 50 units. The parameters of algorithm are: maximum time of computation step 1000 (ms), the length of tabu list 30.

The results were registered after 300 (sec) of computation – Table 1. The results of the experiment show that tasks preemptability influences the number of executed iterations. It results from the increase in the number of movements, because only parts of tasks, not whole tasks, are moved. The obtained cost is lower in case of preemptable tasks because they are usually placed on a smaller number of processors with a similar C_{max} value.

Table 1. The influence of tasks preemptability on the quality of computations

No. of tasks	Tasks	C_{max}	Cost	No. of iterations
10	Nonpreempt.	0,60	17,4	78
10	Preemptable	0,67	16,4	127
20	Nonpreempt.	1,79	22,7	122
20	Preemptable	1,89	21,7	143
30	Nonpreempt.	1,95	24,5	158
30	Preemptable	1,99	18,2	143
40	Nonpreempt.	2,35	18,8	210
40	Preemptable	2,45	22,0	187
50	Nonpreempt.	3,23	25,8	234
50	Preemptable	3,43	19,4	267
75	Nonpreempt.	4,12	22,2	298
75	Preemptable	7,78	21,5	309
100	Nonpreempt.	15,6	25,6	432
100	preemptable	16,1	19,4	474

6.2 The influence of additional resources on the quality of computations and number of iterations

The tests were executed into independent and nonpreemtable tasks.

The number of processors in the database of resources is 10 and maximum number of processors, which algorithm can use is 5. The cost of memory equaled 0.2 was taken into consideration.

Constraints of system: maximum cost is 50 units, maximum operating time is 50 units. The parameters of algorithm are: maximum time of computation step 1000 (ms), the length of tabu list 40.

The results were registered after 300 (sec) of computation – Table 2.

Table 2. The influence of additional resources on the quality of computations and number of iterations

No. of tasks	No. of additional resources	C_{max}	Cost	No. of iterations
10	1	1,64	8,5	137
10	2	1,64	13,2	125
10	5	2,25	7,4	134
20	1	2,97	17,9	178
20	2	3,14	12,4	201
20	5	4,21	12,6	209
30	1	4,47	14,8	267
30	2	4,87	18,9	256
30	5	6,78	12,1	270
50	1	8,24	14,8	458
50	2	9,43	25,5	457
50	5	13,56	12,4	466
75	1	16,89	25,6	789
75	2	21,31	30,2	845
75	5	27,32	28,5	838

The results of the experiment show that additional resources influence the number of executed iterations. The increase in additional resources must influence the scheduling time and cost, and when the number of tasks is large the quality of results deteriorates.

6.3 The influence of precedence constraints on the quality of computations and number of iterations

All the tests were executed with following assumptions: the tasks are dependent, the number of processors in database of resources is 10, maximum number of processors, which algorithm can use is 5, the cost of memory equaled 0.2 was taken into consideration.

Constraints of system: maximum cost is 50 units, maximum operating time is 50 units. The parametrs of algorithm are: maximum time of computation step 1000 (ms), the length of tabu list 40. The results were registered after 300 (sec) of computation – Table 3.

The increased in the number of edges in the graph actually does not change the number of iterations, which is a consequence of movement definition and implementation of the method executing it.

Table 3. The influence of precedence constraints on the quality of computations

No. of tasks	No. of edges	C_{max}	Cost	No. of iterations
20	21	1.67	13.11	440
20	32	1.78	15.70	539
30	38	1.98	18.25	679
30	49	1.77	18.25	546
40	30	2.23	20.03	789
40	59	2.45	17.14	235
50	70	3.12	21.45	478
50	81	3.21	19.45	365
60	85	7.45	19.23	509
60	95	8.32	20.89	417

6.4 The influence of cost constraints on the operation of algorithm

Algorithm was tested with following assumptions: the number of tasks 40, tasks are dependent and nonpreemptable, the number of processors in database of resources is 10, maximum number of processors, which algorithm can use is 8, cost of memory which is taken into consideration equals 0.2. Constraints of system: maximum operating time is 50 units.

The parameters of algorithm: maximum time of computation step 1000 (ms), the length of tabu list 40 – Table 4.

Table 4. The influence of cost constraints – minimization of time

Cost constraint	Operating speed	Cost	Time of computation
5	21,45	5,81	146,53
8	14,34	8,23	137,14
12	11,27	10,12	142,82
16	9,56	14,15	165,21
20	5,33	19,35	176,32
25	4,21	21,90	185,12

The speed of the system increases with the decrease in cost constraints. Thus, algorithm reaches a greater possibility of resources allocation and tasks scheduling.

6.5 The influence of operating time constraints on algorithm

The influence of time constraints on algorithm was researched with identical data as in the previous case. The only difference applies to the maximum operating time constraints valued 25 – (Table 5).

You can also observe that the increased in the value of time criterion causes the decrease in the cost of identified resources.

Table 5. The influence of time constraints – minimization of cost

Operating time constraints	Cost	Operating speed	Time of computation
5	14,6	4,2	207
8	13,21	4,3	178
12	11,1	8,42	265
16	12,3	10,6	143
20	11,2	13,21	121
25	13,6	12,11	106

7 The comparison of results received at non-coherent and coherent synthesis

7.1 Minimization of system cost

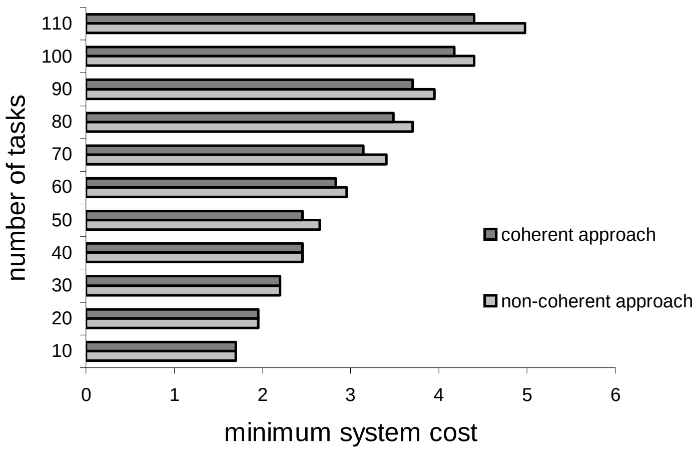

Figure 3. Comparison of coherent and non-coherent approaches – minimization of system cost

7.2 Minimization of operation time

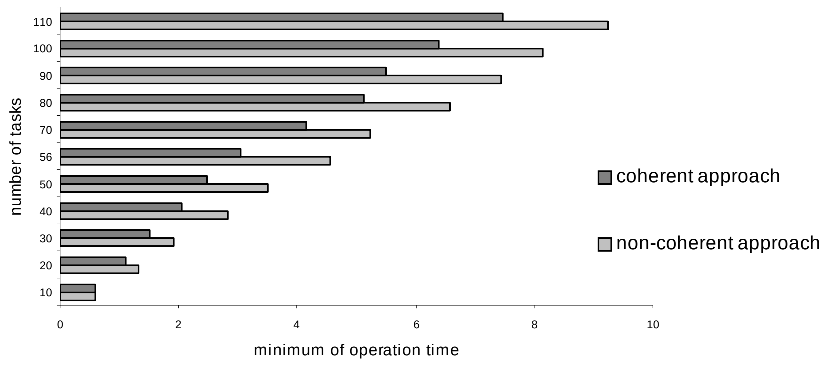

Figure 4. Comparison of coherent and non-coherent approaches – minimization of operation time

Charts (Figure 3 and Figure 4) show that in the case the results obtained for coherent approach the improvement of the received results was greater, especially for optimization based on minimization of operation time criterion.

7.3. The influence of the type of problem on the number of iterations in algorithm

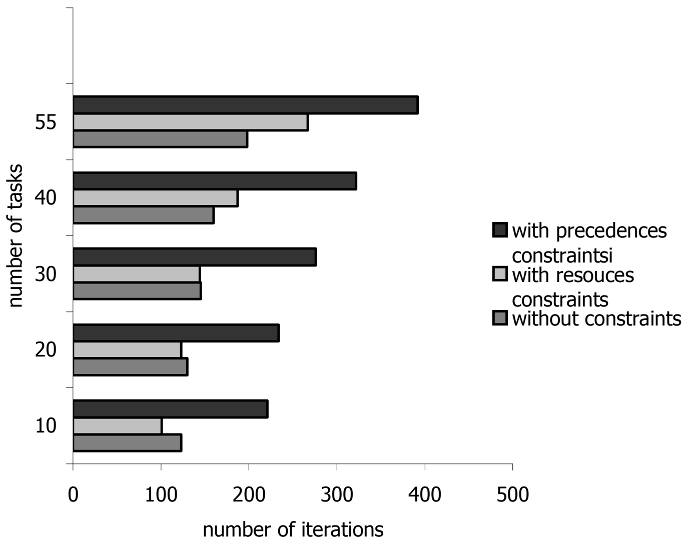

Figure 5. The influence of the type of problem on the number of iterations

The increase in complexity of the problem causes (Figure 5) the increase in the number of iterations which is necessary for reaching the satisfactory result.

8 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 computational experiments show, that Tabu Search algorithm can resolve task scheduling and resource partition problems. For coherent approach a greater improvement of received solutions was indicated considering cost criterion and optimizing based on time criterion as well.

This conforms the correctness of the concept of joint development of hardware and software parts in system design. The approach presented for coherent co-synthesis and the first encouraging experimental results allow a further research in this area.

For example, other heuristics may be applied, including hybrid ones, e.g. evolution algorithm supported by simulated annealing algorithm or ant-colony algorithm. One may also specify additional optimality criteria, e.g. minimum power consumption of the designed system (which is particularly significant for embedded 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 problem of coherent synthesis is a multi-criteria optimization problem. Taking into account several criteria and the fact that optimization of one criterion results often in worsening of the second one, the Pareto optimization can be a right solution providing a trade-off between multiple criteria. The optimum solution in such case shall be the one not dominated by any other solution in the whole solution space. There is no single optimum in such case, but a set of solutions trying to satisfy contradictory criteria. Thus, as result of a coherent system synthesis, we obtain a set of solutions that are optimal in the Pareto sense.

The above issues are now studied.

References

1. Berrojo L., Corno L., Entrena L., Gonzales I., Lopez C., Sonza Reorda M., Squillero G. (2002). *An industrial environment for high-level fault tolerant structures insertion and validation,* Proc. 20th IEEE VLSI Test Symp., Monterey, CA, USA.
2. Blazewicz J., Ecker. K., Plateau B., Trystram D., (ed.), (2000). *Handbook on parallel and distributed processing,* Springer-Verlag Berlin Heidelberg New York.
3. Blazewicz J., Drabowski M., Weglarz J., (1984). *Scheduling independent 2-procesor tasks to minimize schedule length,* Information Processing Letters.
4. Chen L., Dey S., Sanchez P., Sekar K., Chen Y., (2000). *Embedded hardware and software self-testing methodologies for processor cores,* Proc. 37th Design Automation Conf., Los Angeles, CA.
5. Coffman E.G., Jr. (ed.), (1976). *Computer and jbo/shop scheduling theory,* John Wiley & Sons, Inc., New York.
6. Drabowski M., (2004). *Coherent synthesis of heterogeneous system – a neural approach,* International Conference on Artificial Intelligence, Siedlce, Poland.
7. Drabowski M., (2004). *Coherent synthesis of real-time computer system, in "The currently problems of real time system",* WNT, Warszawa, 2004.
8. Drabowski M., Wantuch E., (2005). *Coherent concurrent task scheduling and resources assignment in dependable system design,* European Safety & Reliability Conference, ESREL'05, Gdansk, Poland.
9. Drabowski M., Czajkowski K., (2005). *Task scheduling in coherent co-synthesis of computer system,* International Multi Conference Advanced Computer Systems, ACS-CISIM, Elk, Poland.
10. Drabowski M., Czajkowski K., (2005). *Coherent synthesis of heterogeneous system – a Tabu search approach,* International Conference on Artificial Intelligence, Siedlce, Poland.
11. Drabowski M., (2002). *Co-synthesis of real time system, Processing of IX Conference Real time systems,* The Silesian University of Technology, Ustron, Poland.
12. Eles P., Peng Z., Kuchcinski K, Doboli A., (1997). *System Level Hardware /Software Partitioning Based on Simulated Annealing and Tabu Search***,** Journ. On Design Automat. For Embedded System.
13. Eles P., Peng Z., Kuchcinski K., Doboli A., (1996). *Hardware/Software Partitionin with Iterative Improvement Heuristics,* Proccedings of the 9th International Symposium on System Synthesis.
14. Glover F., Laguna M., (1997). *Tabu Search,* Kluwer Academic Publishers.
15. Golub M., Kasapovic S., (2002). Scheduling multiprocessor with genetic algorithms, *Proceedings of the IASTED Applied Informatics Conference, Innsbruck.*
16. Hyunok Oh., Soonhoi Ha., (2002). *Hardware-software co synthesis of multi-mode multi-task embedded systems with real-time constraints,* Proceedings of tenth Int. Conf. on Hardware/ Software codesign.
17. Yhang Z., Dick R., Chakrabarty, (2004). *Energy-aware deterministic fault tolerance in distributed real-time embedded systems,* 41st Proc. Design Automation Conf., Anaheim, California.