computing@computingonline.net
www.computingonline.net

Print ISSN 1727-6209
On-line ISSN 2312-5381
International Journal of Computing

PARTITIONING AND SCHEDULING RESOLUTION PROBLEMS BY BEES MATING STRATEGY IN DRES' SYSTEMS

Yahyaoui Khadidja

Department of Math and Computer Science, Faculty of Science and Technology
University of Mascara, 29000 Algeria
yahyaouikhadidja@yahoo.fr, yahyaouiinfo@gmail.com

Abstract: In last years, several approaches have been proposed for solving the Hardware/Software partitioning and scheduling problem in dynamically reconfigurable embedded systems (DRESs), directed by metaheuristic algorithms. Honey Bees Mating Optimization (HBMO) algorithm is one of these advanced methods. It is a nature inspired algorithm which simulates the process of real honey-bees mating. In this work, we propose a variant of the Honey-bee Mating Optimization Algorithm for solving Hardware/software (HW/SW) partitioning and scheduling problems in DRESs. The algorithm is used in a hybrid scheme with other metaheuristic algorithms for successfully solving these problems. More precisely, the proposed algorithm (HBMO_ DRESs) combines a Honey Bees Mating Optimization (HBMO) algorithm, the Tabu Search (TS) and Simulated Annealing (SA)). From an acyclic task graph and a set of Area-Time implementation trade off points for each task, the adopted method performs HW/SW partitioning and scheduling such that the global application execution time is minimized. Comparing the proposed method with Genetic Algorithm and Evolutionary Strategies (ES), the simulation results show that the proposed algorithm has better convergence performance. *Copyright © Research Institute for Intelligent Computer Systems, 2016. All rights reserved.*

Keywords: Dynamic Reconfiguration, HW/SW partitioning, scheduling, HBMO, optimization.

1. INTRODUCTION

With the rapid development of integrated circuit manufacturing technology, embedded systems are widely used in a variety of complex applications. How to shorten the product-to-market cycle and accelerate development efficiency has become a major concern in the field of embedded system design. HW/SW co-design remedies many shortcomings of traditional design methods and becomes a kind of mainstream technology of embedded systems development. Dynamically reconfigurable embedded systems (DRESs) target an architecture consisting of general purpose processors and field programmable gate arrays (FPGAs), HW/SW partitioning is a key step of HW/SW co-design and it plays a crucial role in improving the system performance. The decision about the functional blocks to be implemented in software or in hardware in the dynamically reconfigurable embedded systems is based on the experience of the designer and/or making a brief exploration of the design space. This procedure, in addition to not complying with any methodology, does not ensure an optimal result, since for obtaining the best configuration it is necessary to solve an optimization problem which in most of its formulations is NP-hard [1]. In most cases, the HW/SW problem is driven by optimization aim such as the area of hardware, execution time, power consumption and communication cost [2]. Consequently, many researchers employed heuristics based on Genetic Algorithms [3-5], Simulated Annealing [6], Tabu Search [7], Ant Colony Optimization (ACO) [8] and bee swarms [9] to obtain sub-optimal solutions. The IA techniques are also used to resolve the HW/SW partitioning and scheduling problem. In [10], the authors apply a Fuzzy Logic for Hardware/Software Partitioning in Embedded Systems. A Neural Networks Based Approach for the Scheduling of Reconfigurable Embedded Systems supporting FPGA as reconfigurable component is applied in [11].

The paper introduced an automatic partitioning/scheduling approach for DRESs (HBMO_DRESs) that uses the combination of global search represented by HBMO as main strategy with two local searches HW_TS and HW_SA in order to obtain minimal execution time taking into account all temporal (hardware and software execution time), spatial (hardware area) constraints and dependencies (Acyclic Graph tasks) contraints. This work presents the own contributions as following:

- The combination of the global search represented by HBMO with two local search Tabu Search (TS) and Simulated annealing (SA) for the resolution of HW/SW partitioning and scheduling problems in order to intensify the search in promising zones detected by the HBMO exploration process.
- Comparative study of the related works of the HBMO approach used to resolve different optimization problems.
- The differences and similarities found between the using of the HBMO_DRESs approach to revolve our problematic (partitioning and scheduling problem in dynamically reconfigurable embedded systems) and the HBMO related works.

The paper proceeds as follows. In the beginning, the introduction is given. The HBMO and related works are described in Section 2. Section 3 presents the HW/SW partitioning and scheduling formulation problems. The HBMO_DRESs approach is presented in Section 4. In the 5 section, the simulation and experimental results are given. The paper finishes with the conclusion presented in the section 6.

2. HBMO AND RELATED SEARCH WORKS

The HBMO metaheuristic has been successfully used as an efficient and able approach to solve different complex problems like: Real time scheduling, [14], Open vehicle routing [15], Partitioning and scheduling in Multiprocessor systems [16], Dynamic Multi machine power system stabilizers problem [17] problem, Educational Timetabling Problems [18]. The table 1 and table 2 presented different HBMO approach aspects according to the kind of the cited problems above. Also, the differences and similarities between the basic HBMO approach [13] and the different HBMO were used to resolve these problems.

GSAT: Greedy SAT
SLS: Simple Local Search,
GLS: Guided Local Search,
HSAT: hill-climbing procedures for SAT
ENS: Expanding Neighborhood strategy
CRLSMS:Circle Restricted Local Search Moves Strategy.

Table 1. Different aspects of HBMO Parameters

Parameters	HBMO-Basic Algorithm [13]	HBMO_Real Time scheduling [14]	HBMO_Open Vehicle Routing [15]
Number of Queens	01	01	01
Initial population Generation	Randomly	Randomly, Genetic algorithm	Randomly
Local search	(GSAT)	Stochastic process of the drone selection	Annealing condition
Brood Generation	Crossover haploid and mutation	Genetic crossover operator	Crossover based on adaptive memory procedure
Improvement's broods	Mutation FLIP	Simulated annealing, Tabu search	ENS, Circle Restricted Local Search Moves Strategy, (CRLSMS)
Resultants broods	All broods are killed	All broods will be used in the next mating flight	All the brood to the population of drones
Fitness function	Fitness function	minimizing makespan	Minimizing the total distance traveled

Table 2. Different aspects of HBMO Parameters

Parameters	HBMO_real time _multiprocessors System Partitioning/ scheduling [16]	HBMO_ Dynamic Multi machine power system stabilizers [17]	HBMO_Educ Timetabling Problems [18]
Number of Queens	1≤Number queen≤20	01	01
Initial population Generation	Randomly	Randomly	Generated by (LS+LD+LE)
Local search	Annealing conditions	Probabilistic decision rule	Simple descent algorithm
Brood Generation	Crossover haploid	Cross-over ring	haploid
Improvement's broods	SL, GLS, GSAT and HSAT	Mutation operators	Kempe chain neighbourhood (Shaking procedure)
Resultants broods	Delete the Eggs from the list	Discard the all previous trial solutions	All the brood will be used in the next mating flight
Fitness function	Cost function (space, WCET, Communication).	minimizing the power system stability	Timetable quality or penalty cost

3. HW/SW PARTITIONING AND SCHEDULING FORMULATION PROBLEM

Scheduling is the process of determining a starting time for each software/hardware task of the system. When it is used with partitioning in codesign, it also allows to determine the execution time of the system under design when a partitioning solution is determined (mapping of the software and hardware implementation tasks on the architecture target supporting a FPGA). The two problems of partitioning and scheduling are interrelated and tightly coupled. Each of them is known to be NP-Hard, and the fact that they have to be solved simultaneously complicates their resolution [16]. This is why the only feasible way to solve them is to use heuristics, since exhaustive methods take a prohibitive execution time to find the best solution as soon as the system becomes large. In this study, our main goal of partitioning is to select whether to put each task of the application model into SW (task assigned to the processor) or HW (task assigned to the FPGA) such that the whole execution time is minimized and the area occupation is maximized while reducing the overhead. Each application task is defined as a 5-tuple: T_i = (ha_i w_i, h_i, st_i, c_i, ht_i) where $ha_i = w_i * h_i$ denote area, width and height, st_i, c_i and ht_i represent software execution time, communication cost and hardware execution time of the task. The reconfiguration time of the task can be considered as a part of its hardware execution time (ht_i). In this work, we focus on dependent tasks which may be represented as a directed task graph (DAG). The nodes present the tasks with hardware implementation/software implementation. The edges present the dependencies between the tasks. Each edge is evaluated by the cost of communication type as showed in Fig. 1.

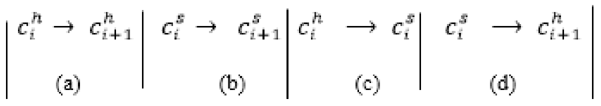

$c_i^h \rightarrow c_{i+1}^h$
(a)
$c_i^s \rightarrow c_{i+1}^s$
(b)
$c_i^h \longrightarrow c_i^s$
(c)
$c_i^s \longrightarrow c_{i+1}^h$
(d)

Fig. 1 – Type of communication between the software and hardware tasks

With *i*=1 to *n* (*n*: number of tasks):

C_i : represents the communication time cost.
h : hard implementation mode
S : Soft implementation mode

(a) The communication time between node *i* and node (*i*+1) if both tasks blocks are assigned to hardware.
(b) The communication time between node *i* and node (*i*+1) if both tasks blocks are assigned to software.
(c) The communication time between node *i* and node (*i*+1) if task *i* is assigned to hardware and task *i*+1 is assigned to software.
(d) The communication time between node *i* and node (*i*+1) if task *i* is assigned to software and task *i*+1 is assigned to hardware.

Then communication cost is ignored if two tasks *i* and *i*+1 are executed on the same processor.

$$\text{Let} : T = \{T1, T2, .., T3\} \tag{1}$$

The mathematic formulation of the task model adopted as following:

$$x_{i=} \begin{cases} 1 \; if\; T_i \; is\; assigned\; on\; FPGA \\ 0 \; if\; \; T_i \; \; is\; \; assigned\; on\; proce. \end{cases} \tag{2}$$

The total system cost communication, execution time and area occupation were refered respectively to “(3)”, “(4)” and “(5)”.

$$\mathrm{c_i = x_i * x_{i+1} * c_i^{hh} + x_i * (1 - x_{i+1}) * c_i^{hs}} \\ +(1 - x_i) * x_{i+1} * c_i^{sh} \tag{3}$$

$$\mathrm{T_{execution}} = \sum_{i=1}^{N}([1 - x_i) * st_i + x_i * ht_i] + \sum_{1}^{N} c_i \tag{4}$$

$$T_{Occupation} = \sum_{i=1}^{N}[x_i * ha_i] \tag{5}$$

4. HBMODRESS APPROACH FOR PARTITIONING/SCHEDULING

4.1 PRESENTATION OF HBMO

Mating in Honey-Bees Optimization algorithm (HMBO) belongs to the category of methods dedicated to resolve the optimization problems of kind NP-complete. It has been introduced for the first time in 2001 by Abbass et al [13], and used by the authors to solve the 3-Sat problem. HBMO was found to outperform some well known algorithms. However, it has not been applied to dynamically reconfigurable embedded systems system scheduling/partitioning HW/SW problems. In this paper, we adapted it to the partitioning problem in codesign.

Optimization using artificial bees is a intelligent technique inspired by the biological process of bees’ reproduction. It is an evolutionary method which includes, nevertheless, strategies based on the neighborhood approach. The reproduction of bees has been modeled to solve optimization problems. It gave birth to the Marriage in honey-Bees Optimization algorithm (HBMO). In this algorithm, artificial queen, workers, drones and broods are used as follows:

Artificial queen has a genotype, set of genes which constitutes its heredity. The queen’s genotype can be considered as a complete solution to the problem. Energy and speed are associated to a queen and used during her mating-flight. In addition, each queen has a spermatheca that collects the sperms of the drones encountered. The queen lays the eggs that become new queens which consist of new solutions. The process of reproduction tends to improve the genotype of the queen through the generations and thus improve the initial solution.

Artificial workers are heuristics that are specific to the problem. Their role is to improve the genotypes of broods (future queens), making it possible to obtain better solutions. Thus, in the artificial model, taking care of eggs corresponds to improving their genotype.

Artificial Drones have only half of a genotype, they are haploid. In the artificial model, drones have a complete genotype and a mask being used to mask (selected randomly) half of the genes. The non masked half of the genotype constitutes the sperm of the drone. At the time of fecundation, the male genes (elements of its genotype) are crossed-over with the genotype of the queen to form a new genotype (complete solution).

The mating-flight can be considered as a series of transitions among a set of states. Each queen goes from a state to another according to her speed and mats herself with the drones met at each state according to a probability rule. A state is in fact a drone; as a result, both terms have the same significance in the algorithm. At the beginning of each mating-flight, each queen has an initial energy that decreases during the flight and the queen returns to her nest when its energy reaches a critical point or when its spermatheca is full. The probability that a queen R mats herself with a drone B is given by the following formula:

$$Prob(D) = e^{[-\Delta(f\Delta)/speed(t)]} \quad (6)$$

where Prob(R,B) represents the probability of a successful mating between queen R and drone B. Difference represents the absolute difference between the ‘‘fitness’’ of the queen and that of the drone. Speed (t) is the speed of the queen at the time *t*. A quick look at this equation shows that the probability of a successful mating is higher either at the beginning of the flight (because the energy of the queen is the highest and thus its speed too), or when the fitness of the drone is close to that of the queen.

$$Speed(t+1) = \propto * \, Speed(t) \quad (7)$$
$$Energy(t+1) = \propto * \, Energy(t) \quad (8)$$

where factor $\alpha \in (0, 1)$ is the amount of speed and energy. An analogy between the naturel honey bee entities and artificial honey bee entities are summarized in Table3.

Table 3 presents the analogy between the natural bees and the artificial bees.

Table 3. Analogy between the natural bees and the artificial bees

Natural honey bee entities	**Artificial honey bee entities**
Bees population	All feasible solutions
Queen	Best and optimal solution
Drones	Incumbent solutions
Broods	New trial solutions

The description of the proposed system: hardware and software environement was presented by Fig. 2.

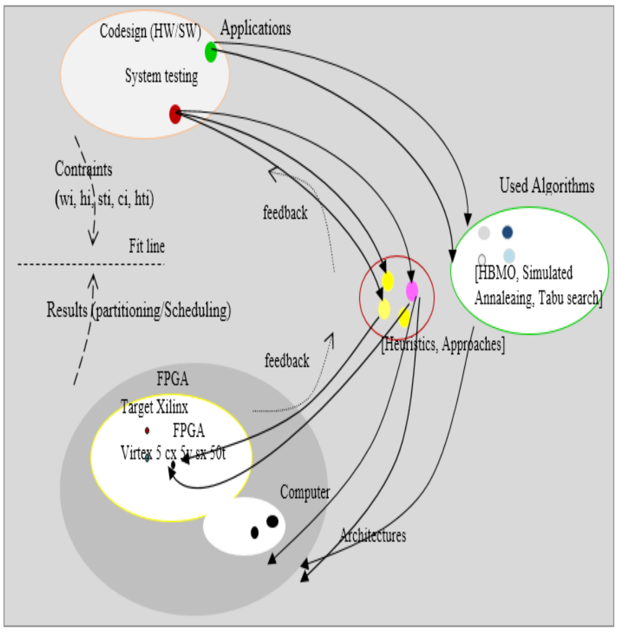


Fig 2. – Description of system: hardware and software environement

Hardware environement contains one processor, one dynamically reconfigurable device of type FPGA, one bus and one chared memory. The bus seves as communication channel connecting processor and FPGA.

Software environement defines the target application: codesign and the proposed algorithms: HBMO, Tabu search annealing Simulated used to resolve the partitioning and scheduling problem. In the following, we detailed the adopted resolution problem.

4.2 ALGORITHMS OF RESOLUTION

The proposed approach combines the Honey Bees Mating Optimization (HBMO) algorithm with the local searches: WH_S based on simulated annealing [19] and WH_TS based on Tabu Search [20]. HBMO was found to outperform some better known algorithms. However, it has not been applied to dynamically reconfigurable embedded systems system scheduling/partitioning HW/SW problems. In this section, we present the approach adapted to resolve the issues addressed. To start with, a set of parameters are explained in the Table 4.

Table 4. Parameters used for HW/SW partitioning/scheduling resolution problem (HBMO_ DRESs)

Parameters	HBMO_HW/SW partitioning/Scheduling embedded system Proposed
Number of queen	01
Initial population Generation	Randomly
Local search	stochastic process of the drone selection
Brood Generation	Genetic crossover operator
Improvement 's broods	Simulated annealing, Tabu search
Resultant broods	All broods will be used in the next mating flight
Fitness function	Cost function= Minimizing_execution_time and maximizing_FPGA ressources occupation

Algorithm explains the HBMO_DRESs approach

```
Begin
Initialization
Generate the initial population of the bees randomly
Evaluate the fitness of the scheduling (time execution cost, area cost);
Select the best plan of HW/SW partitioning of the population of HW/SW partitioning plans which represents the queen ;
Sperm: Size of the spermtheca;
E (t) and S (t): Energy and Speed in [0.5, 1];
α: factor of reduction of energy and speed in [0,1];
M: maximum number of mating flights;
For i:=0 to M (mating flights)
do while E(t) > 0 and Sperm is not full Select a drone
If the drone passes the probabilistic condition
Add sperm (plan) of the drone in the spermatheca
Endif
S(t+1) = α *S(t)
E(t+1) = α * E(t)
Enddo
For j = 1 to Size of Spermatheca
Select a sperm from the spermatheca;
Generate a brood by applying Genetic crossover operator (the queen's genotype with the selected sperm);
Improve the brood's fitness by applying the workers (SA (Simulated annealing & TS (Tabu search))
If the brood's fitness is better than the queen's fitness
Replace the queen with the brood
else Add the brood to the population of drones
Endif
Replace the drones by the broods
Enddo (for)
Enddo
Return The Queen (Best Solution Found)
End.
```

HBMO_DRESs approach follows three highly dependent stages in the life of an embedded application: HW/SW Partitioning scheduling. The process is presented in Fig. 3.

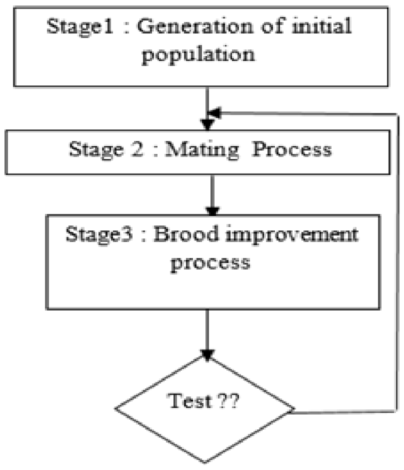

Stage1 : Generation of initial population
Stage 2 : Mating Process
Stage3 : Brood improvement process
Test ??

Fig. 3 – HBMO_DRESs stages process

In the following, more information is given about the Fig. 3 process.

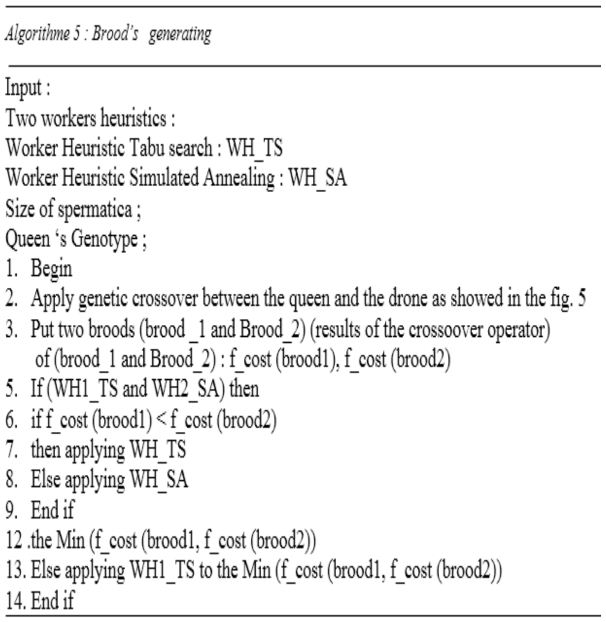
Stage1: The initial population was generated randomly

In the begining of the HBMO algorithm, initial population of partitioning and scheduling must be generated. In our work, this generation was made randomly. Only the feasible solutions with all precedence constraints are respected will be considered.

Stage 2: Broods generating process

The main Operation to generate a new brood in the HBMO_DRESs approach is to apply the genetic crosser to every operation of crossing the queen with drone. The genotypes resulting from the queen with the drone.

This process is presented in Fig. 4.



Algorithme 5 : Brood's generating

```
Input :
Two workers heuristics :
Worker Heuristic Tabu search : WH_TS
Worker Heuristic Simulated Annealing : WH_SA
Size of spermatica ;
Queen 's Genotype ;
1. Begin
2. Apply genetic crossover between the queen and the drone as showed in the fig. 5
3. Put two broods (brood _1 and Brood_2) (results of the crossoover operator)
   of (brood_1 and Brood_2) : f_cost (brood1), f_cost (brood2)
5. If (WH1_TS and WH2_SA) then
6. if f_cost (brood1) < f_cost (brood2)
7. then applying WH_TS
8. Else applying WH_SA
9. End if
12 .the Min (f_cost (brood1, f_cost (brood2))
13. Else applying WH1_TS to the Min (f_cost (brood1, f_cost (brood2))
14. End if
```

Fig. 4 – Broods generation process

Stage 3: Broods improvement

The improvement process is realized by two methods: simulated Annealing (SA) and Tabu Search (TS). The genotypes resulting from the queen with the drone in the broods generating stage are improved by these workers through SA and TS. This stage of HBMO_DRESs generates broods solutions: HW/SW partitioning. Thus the brood's improvement algorithm is applied to every operation of crossover of the queen with the drone. This process is presented by the Fig. 5.

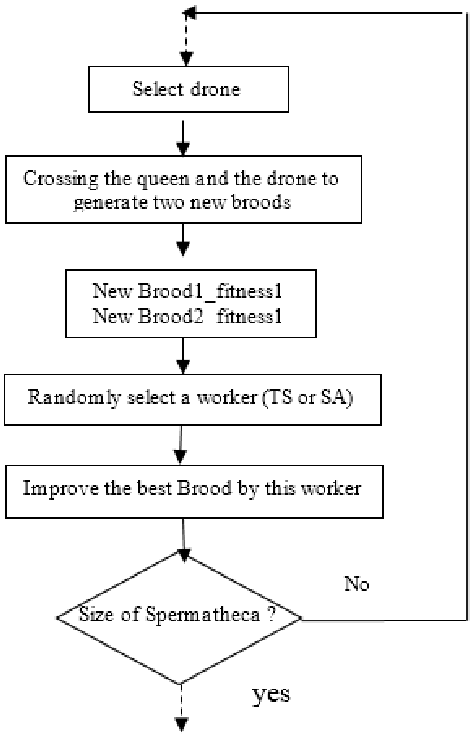

Select drone
Crossing the queen and the drone to generate two new broods
New Brood1_fitness1
New Brood2 fitness1
Randomly select a worker (TS or SA)
Improve the best Brood by this worker
Size of Spermatheca ?
No
yes

Fig. 5 – Broods improving process

5. SIMULATION AND EXPERIMENTAL RESULTS

In this section results of the HBMO for HW/ SW partitioning and scheduling are compared to those given by the genetic algorithm (GA) and Evolutionary Strategies (ES). The JCreator version 4 has been used to implement the approach, jfreechart: version 1.0.13 for the realization of the graphic results and the Graph stream version gs-core-1.2 as tools to generate several task Directed Acyclic Graphs (DAG). These generated graphs are used as the benchmark to evaluate the result quality of our HW/SW partitioning and scheduling. In the graph, Each node i is represented by the quadruplet (st_i,ht_i, c_i, ha_i (w_i, h_i)) contains software execution time (st_i), hardware execution time (ht_i), cost communication (c_i) and hardware area occupied (ha_i)such the area ha_i is defined by the product between the width (w_i) and the height (h_i). It specifies the number of CLBs necessary for the execution of the task in FPGA. The values of software execution time (st_i), hardware area occupied (ha_i) and hardware execution time (ht_i) were generated randomly. Since the execution time for one node implemented in software is generally greater than in a hardware implementation. The execution time of hardware (FPGA) equals one third to one fifth of the execution time of software [21].

In this section, we study the effectiveness of HBMO_DRESs approach. For showing this result the comparisons were made on the values found by the application of each algorithm: HBMO_DRESs GA (Genetic Algorithms) and Evolutionary Strategies (ES). The parameters of the algorithms (HBMO, GA and ES) have been selected after thorough testing. A number of different values were tested and the ones selected are those that yielded the best results in both solution quality and computational time. The tests were performed with the following parameters:

The HBMO parameters: the initial speed was set to 0.8, the initial energy was set to 0.7, Alpha (the factor that influences the speed of the queen) to 0.2, Mating flight was set 50 and spermatica was set to 6.

The GA parameters: number of generation was set to 100, crossover probability was set 0.6 and mutation probability was set 0.2.

The ES parameters: the size of population was set to 100, the two mutations operators M1 and M2 were applied with Probability P1 and P2=1-P1 respectively. We evaluated the performance of these operators for various values of P1 and P2 to see the effect on the converegence of the ES. P1 $\approx$ P2 will give reasonables convergences rates. The number of generation was 40.

5.1 SIMULATION RESULTS

In each execution evaluations of both objective functions were made: minimization of execution time and maximization of occupation area. This is illustrated by the graphs presented in the Fig. 6 and Fig. 7:

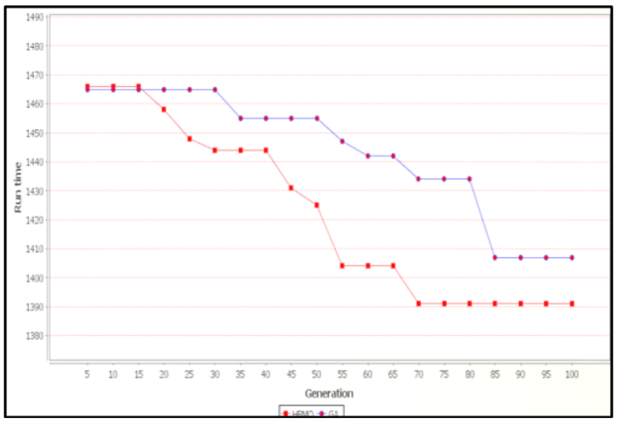


Fig. 6 – Comparative results between the graphics simulations HBMO_DRESs approach genetic algorithms GA and evolutionary strategies (ES) (case of minimization of execution time)

In Fig. 6, we note that: Initialy, the HBMO_DRESs _partitioning, GA_partitioning and ES_partitioning are generated aleatory. The initial values execution time respectively of HBMO_DRESs partitioning, GA_ partitioning and ES_partitioning were 1465 time unit of each one. The execution time was decreased gradually after each iteration for the strategies adopted (HBMO, GA and ES). The final values showed in the grahic results were 1391 time unit (HBMO), 1408 time unit (GA). And ...... time unit (ES). The best value was given by the HBMO algorithm.

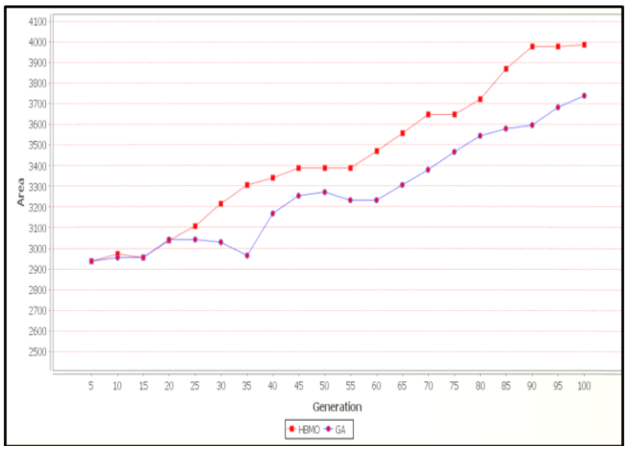


Fig. 7 – Comparative results between the graphics simulations HBMO_DRESs approach, Genetic Algorithms GA and ES (case of maximizing the space used for FPGA size (FPGA area) = 4100 CLBs

In Fig. 7, the graphic results show that each strategy adopted HBMO_DRESs or Genetic algorithms (GA) looked for to maximize the exploitation of FPGA occupation. In the initial aleatory generation, the size occupied respectively by HBMO_DRESs partitioning and GA_partitioning are: 2950 CLBs. After more generations, the size occupied by HBMO_partitioning and GA_partitioning increased. The majority of tasks will have hardware implementations. But we note that the best result is given by the HBMODRESs approach, all the FPGA area was occupied.

In order to show the efficiency of the HBMO_DRESs, we use an exhaustive search by different benchmarks of DAGs (20 tasks, ...,250 tasks) and compared the resuts GA method. A coefficient of improvement ration is calculated. The quality is given in terms of the improvement rate of the solutions found by the application of HBMO compared to those found by the application of genetic algorithm (GA).

Through the exhaustive search, the following graphs summarized the found simulation results with different FPGA area (number of CLBs: 350, 200, 150).

As shown in the following Fig. 8, the improvement is more obvious through the optimization of the partitioning/scheduling HBMO_DRESs approach, compared with the GA algorithm and ES algorithm.

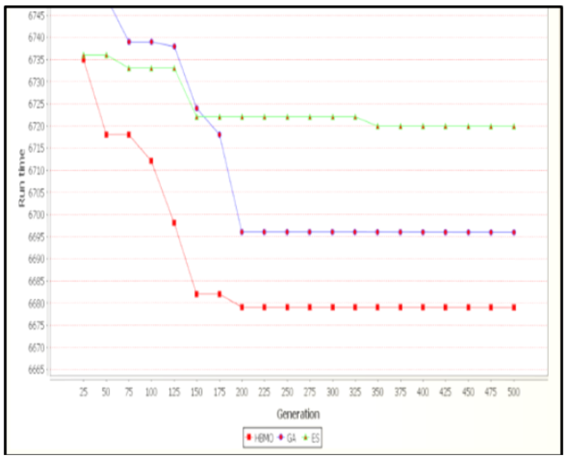


Fig. 8 – Comparative results between the graphics Simulations HBMO_DRESs approach , genetic algorithms GA and Evolutionary Strategies (ES)(FPGA area =350 ClBs)

As shown in Fig. 8, the improvement is the largest when the number of generations is between 25 and 115 generation, and the corresponding rate is 40.7%.

With the increase in the number of generations, the HBMO_DRESs partitioning/sheduling strategy has remarquable ameliored results than GA and ES algorithms, So, the adopted HBMO_DRESs strategy had fully utilized hardware area to reduce the total execution time as much as possible.

In the Fig. 9 and the Fig.10 the same remarks are given for the simulation results in Fig. 8: The HBMO_DRESs approach was always the best approach. But these figures showed that when the size of the FPGA area decreases the ES algorithm has a best convergence to the execution time (diminition of computation time) than the GA approach.

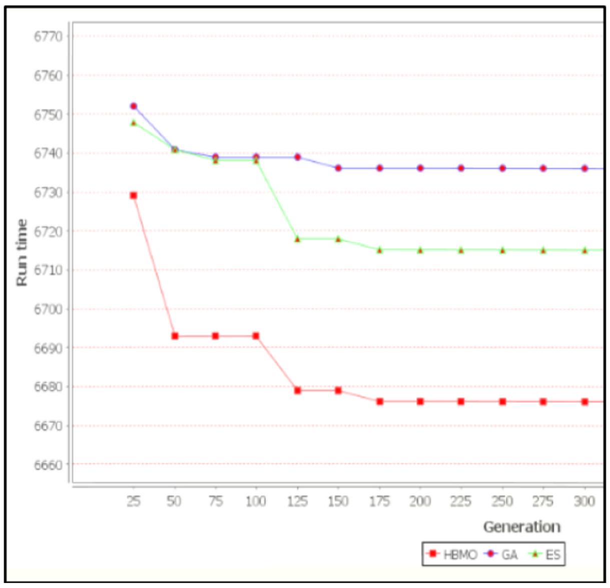


Fig. 9 – Comparative results between the graphics simulations HBMO_DRESs approach, genetic algorithms (GA) and Evolutionary Strtegis (ES) (FPGA area = 200CLBs)

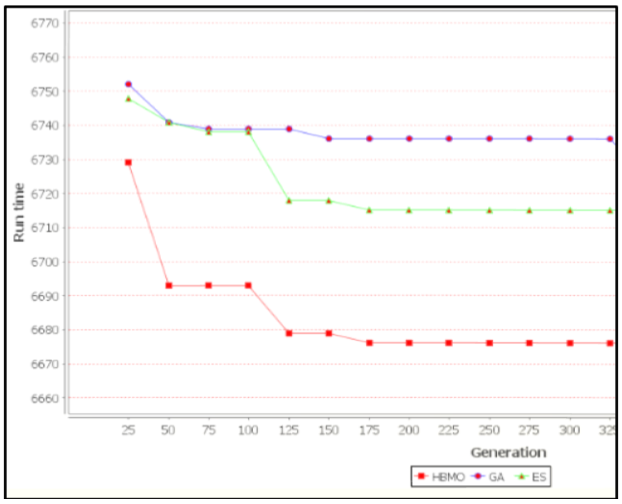


Fig. 10 – Comparative results between the graphics simulations HBMO_DRESs approach, genetic algorithms (GA) and Evolutionary Strategies (ES) (FPGA area = 150CLBs)

6. CONCLUSIONS

In this paper, a task of partitioning and scheduling approach with collective bees behavior (HBMO: Honey Bee Mating Optimization) has been proposed for Dynamically Reconfigurable Embedded systems augmented with the reconfigurable circuit. The HW/SW partitioning and scheduling algorithm has been designed to further minimize the overall execution time and to maximize the occupation of the reconfigurable ressource area. The proposed approach takes into consideration the overheads imposed by the reconfigured devise FPGA in terms of number of ressources, speed of tasks execution and incressing of parallelism degree of application.The application of the adopted method shows that the two problems: partitioning and scheduling are hence interrelated and tightly coupled because the partitioning is included in the design space exploration while scheduling allows the evaluation of each solution. To test the validity of these algorithms, the results are obtained by HW/SW partitioning and scheduling different groups of task graphs on different target systems (using different number of resources: size of FPGA). We compared the adopted approach, HBMO_DRESs, to GA method and ES, The obtained simulation results showed the optimality for scheduling process and the effectiveness of the adopted approach (HBMO_DRESs) in the all cases of examples and they showed the reduction of the overall execution time up to 40.7 % compared with GA approach and 37.2 compared with the ES approach.

7. REFERENCES

[1] Y. Jing, J. Krang, J. Du and B. Hu, “Application of improved simulated annealing optimization algorithms in hardware/software partitioning of the system on chip”, *Springer*, pp. 532-540, 2011.

[2] R. Ayadi, B. Ouni and A. Mtibaa, “A partitioning methodology that optimizes the communication cost for reconfigurable computing systems,” *International Journal of Automation and Computing (IJAC)*, Institute of Automation and Springer-Verlag Publishers, Vol. 9, No. 3, pp. 280-287, June 2012.

[3] M.B. Abdelhalim, S.E.-D. Habib, “An integrated high-level hardware/software partitioning methodology,” *Des Autom Embed Syst.*, No. 15, pp. 19-50, 2011.

[4] P. Arat, Z.A. Mann, and A. Orbn, “Algorithmic aspects of hardware/software partitioning,” *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, Vol. 10, Issue 1, 136-156, 2005.

[5] S. Luo, X. Ma, Y.i Lu, "An advanced non-dominated sorting genetic algorithm based SOC hardware/software partitioning," *Acta Electronica Sinica*, Vol. 11, pp. 2595-2599, 2009.

[6] S. Dimassi, M. Dijemai, B. Ouni, A. Mtibaa, "Hardware-software partitioning algorithm based on binary search trees and genetic algorithm to optimize logic area for sopc," *Journal of Theoretical and Applied Information Technology (JATIT)*, Vol. 66, No. 3, pp. 788-794, 2014.

[7] F. Vahid, "Modifying min-cut for hardware and software functional partitioning," in *Proceedings of the 5th International Workshop on Hardware/Software Co-Design (CODES/CASHE97)*, Braunschweig, Germany, 1997, pp. 43-48.

[8] F. Ferrandi, P. L. Lanzi, C. Pilato, D. Sciuto, A. Tumeo, "Ant colony optimization for mapping, scheduling and placing in reconfigurable systems," in *Proceedings of the NASA/ESA Conference on Adaptive Hardware and Systems (AHS)*, 2013.

[9] M. Koudil, K. Benatchba, A. Tarabet, E. Batoul Sahraoui, "Using bees to solve partitioning and scheduling problem in codesign," *Applied Mathematics and Computation*, Vol. 186, pp. 1710-1722, 2007.

[10] H. Daz Pando1, S. C. Asensi, R. Seplveda Lima, J. F. Caldern and A. Rosete Suarez, "An application of fuzzy logic for hardware/software partitioning in embedded systems," *Computacion y Sistemas*, Vol. 17, No. 1, pp. 25-39, 2013.

[11] G. Rehaiem, H. Gharsellaoui, S. Ben Ahmed, "A neural networks based approach for the real-time scheduling of reconfigurable embedded systems with minimization of power consumption," in *Proceedings of the International Conference ICIS'2016*, Okayama, Japan, June 26-29, 2016.

[12] K. Yahyaoui, M. Bouchoicha, "A hardware-software partitioning and scheduling approach for dynamically reconfigurable embedded systems," *Proceedings of the International Conference ICHICS'S16*, Morocco, 2016.

[13] H.A. Abbas, "A single queen single worker honey bees approach to 3-Sat problem," in *Proceedings of the Genetic and Evolutionary Computation Conference GECCO'2001*, San Francisco, USA, July 2001.

[14] K. Yahyaoui, F. Debbat, M.F. Khelfi, "HBMO Hybridization: Application to real-time task scheduling," The Mediterranean Journal of Computers Networks (MJCN), No. 2, 2012.

[15] Y. Marinakis, M. Marinaki, G. Dounias, "Honey bees mating optimization algorithm for large scale vehicle routing problems," *Natural Computing*, No. 9, pp. 5-27, 2010.

[16] M. Koudil, K. Benatchba, A. Tarabet, E. Sahraoui, "Using bees to solve partitioning and scheduling problem in codesign," *Applied Mathematics and Computation*, Vol. 186, pp. 1710-1722, 2007.

[17] N. Yousefi and H. Ebrahimian, "Optimal design of multi-machine power system stabilizers using interactive honey bee mating optimization," *Trends in life science (TLS) Dama International Journal*, Vol. 4, Issue 1, 2015.

[18] N. Sabar, M. Ayob and G. Kendall, "Solving examination timetabling problems using honey-bee mating optimization (ETP-HBMO)," in *Proceedings of the Multidisciplinary International Conference on Scheduling: Theory and Applications (MISTA)*, Dublin, Ireland, 2009.

[19] M. Fleischer, "Simulated annealing: Past, present, and future," in Proceedings of the Winter Simulation Conference, 1995.

[20] F. Glover, "Tabu search part I," *ORSA Journal on Computing*, Vol. 1, No. 3, pp. 190-206, 1989.

[21] H. Han, W. Liu, W. Jigang, G. Jiang, "Efficient algorithm for hardware/ software partitioning and scheduling on MPSoC," *Journal of Computers*, Vol. 8, No. 1, pp. 61-68, 2013.

***Yahyaoui Khadidja** received the engineer degree in soft engineering and the M.Tech. degree in industrial computing from Oran university computer science department, Algeria, in 2000 and 2006, respectively. She received PhD degree in computer science in 2013 from Oran University, Algeria. She is currently a Research member of the computer science laboratory at Mascara University. Her main research interests include optimization, artificial intelligence, and reconfigurable embedded systems.*