DEVELOPMENT OF PATTERNS OF EVOLUTIONARY METHODS FOR DIAGNOSING DIGITAL DEVICES

Ramzi ZOUAOUI[1], Imen HERZI[2]

[1]*Professor Assistant, LIPAH-LR 11ES14, University of Tunis El Manar, 2092 Tunis Tunisia*
[2]*Professor Assistant, LIPAH-LR 11ES14, University of Tunis El Manar, 2092 Tunis Tunisia*

---***---

Abstract - *The paper proposes classification of existing evolutionary methods for generating identifying sequences of digital devices as well as development the templates for constructing such methods and their algorithmic implementation, which are based on this classification. Such classification and built templates are part of the methodology for the synthesis of evolutionary methods of diagnosing digital devices aimed to accelerate the development of new methods by unifying their components.*

Key Words: Reliability, diagnostics, digital device, evolutionary computation, classification, genetic algorithm, simulated annealing.

1. INTRODUCTION

Currently, the entire design process of Digital Devices (DD) is carried out using specialized Computer Aided Optimization CAO systems. This allows us to design highly reliable DD. During the different stages of this process, the developer is faced with the need to create input Identification Sequences (IS) of different classes: characteristics, testing, etc.
Traditional methods of constructing such types of sequences for sequential DD are adaptations of the corresponding methods developed for combinatorial devices. The internal flaws inherent in these methods have given rise to a new and evolving paradigm of development methods.
Often, researchers use Genetic Algorithms (GA) to solve problems in diagnosing DD [1–2]. The objective of GA is to repeat the natural mechanism of improving the properties of individuals by adapting to the solution of the problem of natural evolutionary mechanisms: crossing, survival, physical form, etc.
In Italy, at the University of Turin, researchers, in fact, were pioneers in this field and proposed a large number of GA for the construction of IS-DD [3-5]. In particular, algorithms for the generation of verification tests, initialization sequences and verification of the equivalence of behavior of synchronous sequential DD have been developed. The binary input sequence acts as an individual in algorithms and the set of individuals forms a population. For individuals, a set of evolutionary operations is applied: selection, crossing, mutation.

A group from the University of Illinois worked in the same direction [6-7]. In general, this approach has been very successful and has been developed in many works by other authors [9–10].

National authors [11–12] have also made an important contribution to this field. At the same time, we note efforts to develop versions of parallel methods [13] that have extended GA to a multiple observation strategy [14] and to test new types of flaws [15].

Often, the methods developed are very close ideologically and differ only in certain details: the modeling method used to evaluate individuals, the heuristic methods used, etc.

Another frequently used evolutionary paradigm is the method (strategy) of Simulating Annealing SA [16]. The main difference of this paradigm compared to GA lies in the evolution of a potential solution, called configuration. The process of finding the optimal solution is constructed in such a way that at the beginning of it there are frequent disturbances of the solution (deterioration of properties) and they disappear at the end of the search. The application of SA to the resolution of diagnostic problems has generally followed the same process as GA [17-18], and the methods developed show a fairly high efficiency in terms of corresponding tasks.

Traditionally, associated artificial intelligence algorithms also include ant algorithms [19], swarm bee algorithms (swarm intelligence) [20] and a number of metaheuristics little known and rarely used in diagnostic tasks, for example taboo research [21]. Algorithms of this type have been applied to solving problems in the diagnosis of digital circuits [22] and have sometimes given good results [23, 24]. However, we will not consider them in this article because, on the one hand, their use in technical diagnostic tasks is very limited, and on the other hand, they are not evolutionary algorithms (EA) in the sense of the established classification [2].

A feature of all the works mentioned above is that the development of methods is based on a heuristic approach. When developing a new method and, depending on the conditions of the problem, coding, scalable operations are chosen, evaluation functions are constructed, etc. That is, the typical process for the development of a unique genetic algorithm to solve a specific problem is fully executed.

On the other hand, the presented basis of the methods developed makes it possible to make appropriate generalizations and to form a methodology for the synthesis of evolutionary methods of construction of IS. No research in this direction has been carried out. In the meantime, such a methodology should, due to the unification of the approach and the development of common components of evolutionary methods, increase the speed and quality of the development of new methods.

Part of this methodology is a classification (hierarchy) of evolutionary methods, reflecting an understanding of their place in solving the corresponding IS generation problems. This article proposes a variant of a similar hierarchy of evolutionary methods, including population and non-population paradigms, as well as method models of the components of such a hierarchy.

2. SCALABLE ALGORITHMS AT ONE AND TWO LEVELS FOR THE CONSTRUCTION OF IDENTIFICATION SEQUENCES OF DIGITAL DEVICES

- Scalable algorithms at a single level

- Two-tier Evolutionary Algorithms (EA).

In the event that a method of finding a solution (constructing a sequence) can find it in an EA call, we will talk about one-level scalable algorithms (Fig -1a) or a one-level application scheme to apply EA.

A characteristic of one-level evolutionary methods is that their purpose is defined only once and is known before the start of the algorithm. For such an EA, the formalization of an objective is expressed in the form of a function of evaluation of potential solutions. Achieving this goal shows the completion of the algorithm as a whole. In fact, the whole method of constructing a sequence is an evolutionary algorithm. Thus, the structure of methods of this type is such that the main evolutionary cycle of building new solutions is also the most external.

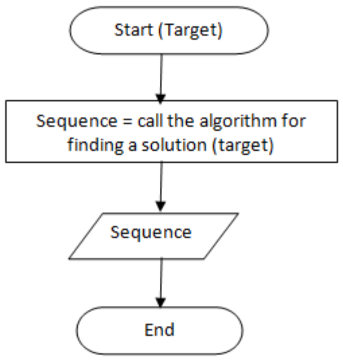
Start (Target)
Sequence = call the algorithm for finding a solution (target)
Sequence
End

Fig -1a: One-level algorithm

The problem is that solutions based on a single-level model, can include special cases of problems of construction of characteristic sequences of different types:

- Initialization sequences: transfer of the device from an undefined initial state to a given start (start);

- State-Obtaining Sequences): here we determine the final state that must be reached after the application of the input sequence;

- Verification of the equivalence of the behavior of two given devices;

- Estimates of the heat dissipation parameters of a given device (mono-cycle peak, n-cycles and stable).

In the event that the complexity of the problem does not allow this method to find a solution in a single call of the EA research, we will talk about two-level methods (Fig -1b) or a two-level system to apply the EA. The methods of this class provide an iterative construction scheme. Each iteration consists of two phases. The first phase is dedicated to the search for an intermediate (local) target, if this objective is achieved, then a call is made from the EA for the search for a solution for this local target that formulates the second phase of the iteration. The iterative search for intermediate objectives and their achievements leads to a solution to the global problem. In this case we call the first phase of the search for a local target as the top level of EA while the second phase of achieving that local goal as being the lower level of EA. With this structure, the second phase of the algorithm corresponds to a single-level EA of IS identification sequence construction. In the construction problems of the input IS, usually the final solution (sequence) is often constructed according to the additive principle (it is a set of intermediate solutions).

That is, these tasks are naturally projected onto EA's scheme at two levels.

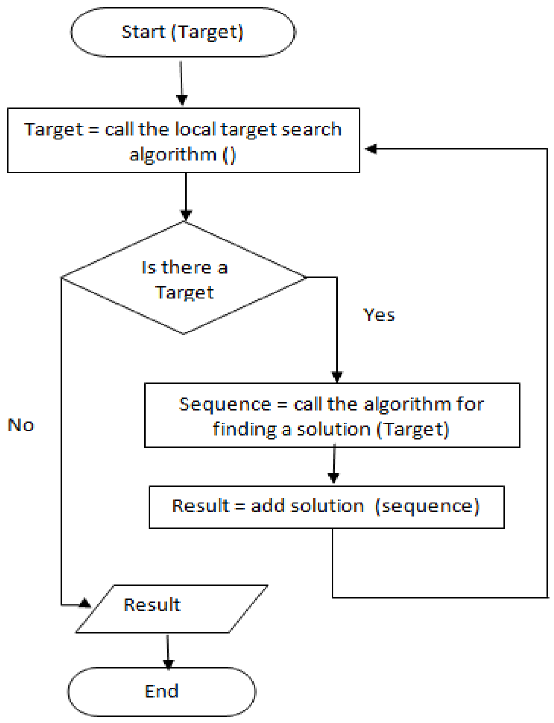
Start (Target)
Target = call the local target search algorithm ()
Is there a Target
Yes
No
Sequence = call the algorithm for finding a solution (Target)
Result = add solution (sequence)
Result
End

Fig -1b: Two-level algorithm

The EA algorithm with two levels of IS identification sequence construction depends on:

- Methods of construction of verification tests using different strategies (state confirmation based on GA or the propagation of the influence of a defect based on GA);

- The method of construction of diagnostic tests;

- The method of construction of the energy efficiency tests of digital devices DD which is a modification of the method of carrying out the tests; in this case, for the sequences under construction, we study not only the verification properties but also in addition and in particular the parameter of heat dissipation.

In the case of EA, research into one- or two-level schemes shows the possibility of considering them as algorithms based on the evolution of a population of solutions as well as algorithms with a single scalable solution. In the case of evolutionary methods, we will study the GA genetic algorithms, the methods with the evolution of a single solution presented by the method of annealing simulation SA.

Thus the majority of EA construction of IS identification sequences can be represented in the form of a hierarchy, which is represented in Fig -2. The classification characteristics of EA in this hierarchy are:

- The population of the evolution of the solution: a population or a single solution.

- The flatness (level) of the algorithm: one or two levels.

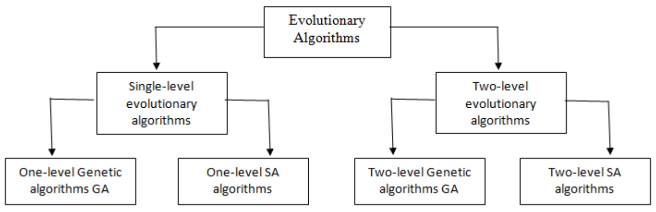
Evolutionary Algorithms
Single-level evolutionary algorithms
Two-level evolutionary algorithms
One-level Genetic algorithms GA
One-level SA algorithms
Two-level Genetic algorithms GA
Two-level SA algorithms

Fig -2: The Classification of evolutionary algorithms for the constructing of identification sequences

In addition, below for developed methods that use GA as a search method, we will assign the name GA-method, and for algorithms having a search based on the annealing simulation algorithm - SA-method.

In the continuation of our contribution, we will develop one- and two-level models based on GA-method for the construction of IS digital devices and similarly models based on SA-method.

3. SINGLE-LEVEL GA MODEL FOR THE CONSTRUCTION OF IS IDENTIFICATION SEQUENCES OF DIGITAL DEVICES DD

Formally, GA is defined as follows:

Let *Ind* be a set of individuals;

$Pop = \{pop | pop \subseteq Ind, pop < \infty\}$: The set of possible populations of finite sizes. So GA is an ordered collection of objects:

$$AG = (Ind, Sel, Cross, Mut, O, Fit, pop_{ini}, N_{ind}, l, p_{croi}, p_{mut}).$$

Where: $Sel : Pop \rightarrow Ind$: selection operation: selects one or more individuals from the given population to perform genetic operations; generally, the highest probability of being chosen as a parent is to have a fitness function the highest.

$Cross : Ind \times Ind \rightarrow Ind$: Crossing operation: by the union of two given individuals is built a new individual in accordance with the chosen crossing rule;

$Mut : Ind \times Ind$: Mutation operation: construction of a new individual by applying the mutation rule to the given individual;

$O : Ind \rightarrow R$: Evaluation function;

$Fit : Ind \times Pop \rightarrow R$: Fitness Function: The evaluation function must be distinguished from the fitness function, since the latter shows the quality of an individual compared to the others in the population. To calculate it, it is necessary to know not only the estimate of one individual, but also the estimate of all other individuals in the population;

$pop_{ini} \subseteq Ind$: Initial population of individuals; it is often built at random during its implementation;

$N_{ind}(pop) = |pop|$: Population size, which specifies the number of individuals included in the pop population.

l : the length of an individual in bits for binary encoding;

p_{croi} et p_{mut} : The probabilities of using the crossing and mutation operators respectively.

The goal of GA is to find the individual with the highest score: $O \rightarrow max.$

It can be seen that for the implementation of a specific GA-oriented method for the construction of IS, a number of its components must be specified constructively: the reproduction operator, the evaluation function, the method of construction of the fitness function, etc... Often, when constructing a corresponding GA-method, the specific implementation of these algorithm components presupposes their experimental justification. The final implementation of the method includes the type of operator, function, and so on, which gives the best numeric results. We'll call these components implementation-dependents.

Ignoring the implementation of such components, it is possible to develop a generalized model for GA-methods for IS construction using a single-level application model. Such a GA model will include both the coding of individuals, a set of genetic operations (selection, crossing, mutation) and the basic structure of the GA-method. With this dependency of realization, implementation dependents should only be appointments and their concrete content must be specific for that model and refers to the construction of a specific method, its algorithmic implementation and its customization.

Let be the coding of individuals of the sequences, the coding of the populations and the evolutionary operations (selection, crossing and mutation) selected according to the objectives of the method. These components are well known and described for example in [12].

In addition, later for the construction of a single-level GA-method model, we will assume that for each individual sequence S, there is a known method for calculating its evaluation function (A_0, S) , specified constructively by the procedure $estimate\_individus\ (A_0, S)$ with A0 the digital devices processed. Most often and during the calculation of the evaluation functions, an explicit modeling occurs of the behavior of device A0 when sending the sequence S to its input. Depending on the specific task, a correct simulation or a simulation with defects can be used [25].

The GA as a whole exposes an iterative construction of new populations of potential solutions that are executed until the shutdown criterion is met. The enlarged pseudocode of the single-level GA model of a GA-method for is construction can be presented in the following form:

Algorithm A1

```
Onelevel_GA_construction_SI(A0, paramètres)
{
Preliminary_processing (A0) ;
$pop_{ini} = Construction\_initial\_population()\ (N_{ind}, L)$;
$Estimate\_population(pop_{ini}, A_0, N_{ind})$;
$pop_i = pop_{ini}$;
Number_population=0 ;
  // main cycle by generation
While(Stop_Criterion_Not_Reached ())
  {
$Calculate\_fitness\text{-}fonction(pop_i, N_{ind})$;
   // intermediate population build cycle
While(New_Population_is_being_built ()) ;
{
    Parents = Operation_Selection
    Descendants = Operation_Crossing (Parents);
Descendants = Operation_mutation (Descendants);
Add _in_New_Population (Descendants);
  } //end While – construction of a new population
$pop_i$ = Build_New_Population();
Estimate_population($pop_i, A_0, N_{ind}$);
Number_population ++ ;
Adaptation_Parameters ();
  } //end While – Stop_Criterion _Reached
Sort_Population_By_Rating ($pop_i$);
// Solution = best individual in the last population
  Solution= Current_Population [0] ;
  } //end Onelevel_GA_construction_SI
```

Let's explain the pseudo-code of the method. The following variables are used:

A_0- processed digital apparatus,

$pop_i et\ pop_{ini}$ – Respectively Current population and initial iteration population,

L - Initial length of sequences of individuals indicating the number of sets of inputs,

Parents - individuals selected for genetic crossbreeding and mutation operations,

Descendants - individuals resulting from genetic operations,

Number_population - iteration counter, indicates the number of the current population.

In code, functions have the following overhead:

Construction_initial_population() - Implements a strategy to build an initial population of individuals;

Estimate_population - Calculates an evaluation function for all individuals in a given population.

Operation_Selection (), Operation_Crossing(), Operation_mutation() – perform genetic selection, crossbreeding and mutation operations respectively;

Add_individual_population - adds an individual to a given population

Preliminary_processing - calculates the static parameters of the evaluation function [26].

It can be seen that the given pseudo code implements the GA main cycle to find a solution. Iterations of building new populations within the GA stop when one of the following conditions is met: Iterations of building new populations within the GA stop when one of the following conditions is met: the exact solution of the problem is found, the limit number of iterations is reached, the specified number of iterations does not improve the estimate of the best individual. After stopping the iterations, the solution to the problem takes the sequence-individual with the best estimate in the last population reached.

With this construction of the GA method model, all implementation-dependent mechanisms are hidden in the corresponding procedures and are specified during implementation.

Thus, based on this model, other private GA methods for constructing ISs will be constructed by specifying an evaluation function and populating the implementation-dependent components. It is on the basis of this model that we implement all the single-level GA methods listed in Section 2. In fact, they differ in the type of sequence-individual evaluation function and in the heuristics used.

4. TWO-LEVEL GA MODEL FOR THE CONSTRUCTION OF IS IDENTIFICATION SEQUENCES OF DIGITAL DEVICES

Single-level GA methods for IS construction should be considered as the first step in the application of GA to solve technical diagnostic problems. However, a number of problems cannot be solved with a single GA search call because the solution of the problem requires the search for several intermediate solutions. Since several subtasks are posed in the search process at different times, a parameter (parameters) appears in the evaluation function, which depends on the current local objective

(Fig. 1b). We consider cases where the search for a solution for local goals is reduced to calling a procedure that, to one degree or another, uses scalable search based on GA. We will consider this scheme of building a solution to the problem as a two-level scheme of application of GA, and the methods developed on its basis as two-level GA methods.

We will develop a model for the top level of two-tier methods for the construction of the IS in which GA acts as an evolutionary search procedure.

An enlarged synoptic diagram of the two-level GA methods is illustrated in Fig.1b. The centerpiece of all methods based on this schema is the target object. The nature of this object is specified (ensured) when constructing a specific method. The top-level main loop (cycle) of these methods has two steps:

- Find the current target; if it is impossible to find such a target, then the method ends;

- Seek a solution leading to the achievement of the current Objective.

Let be given a A_0 and a coding solution. The top-level model for two-level GA methods for IS construction can be represented as follows.

Algorithm A2

```
Towlevel_GA_construction_SI($A_0$, Parameters)
{
Preliminary_processing ($A_0$) ;
While(Stop_Criterion_Not_Reached ())
  {
Target = Select_Current_Target ($A_0$, Parameters);
if(Target ==not_ Target)
return ; // No Target - algorithm termination
else
   {    // target is found - GA call to search for a local
solution
    S=GA_construction_IS($A_0$, Target, Initial_Population) ;
if(S==NULL)  // sequence not built
Mark_As_Unreachable (Target);
else
      {
AddIn_Test (S);
Additional_ Check ($A_0$,S);
      } // end else - sequence built
}   // end else – have chosen a target
}    // end while - did not reach the stopping criterion
}     // end Towlevel_GA-method
```

Preliminary processing, presented in the code by the procedure of the same name, may include the construction of a set of intermediate Objectives, the preparation of a data structure to work with them later (lists, tables, etc.), as well as the calculation of the static parameters of the evaluation functions [26].

If the current objective is found in the iteration then we call on the GA-method which must generate an intermediate sequence S. At the same time, structurally, such a GA is presented at a single level and corresponds to the model of method A1. The initial population of the GA of the lower level can be constructed, for example, when searching for the objective at the higher level. This approach is implemented in [3, 9, 12].

After constructing the intermediate sequence S and depending on the specific method, additional modeling can be carried out [3, 12]. The introduction of such heuristics is related to the fact that the intermediate sequence S constructed for a local purpose may have broader identification properties. This fact is established in the procedure *[Additional_Check ()].*

The resulting solution is constructed according to the additive principle: the final sequence consists of a set of intermediate sequences and the different optimization procedures can be applied to it.

Another feature of this model is that GA is used precisely to achieve the goals. In general, GA can be used at the research stage but for us such use is secondary.

This model is generalized for a fairly broad class of methods. Application of different types of "Purpose" objects and the different GA search procedures at the lower level generate a variety of different methods.

Let's show how it works with examples. In the problem of the construction of validation tests the set of objectives is the set of defects. Here, at the top level, an uncontrolled defect is selected as the target, for which a control sequence is constructed at the lower level. In fact, at the lower level, we consider the problem of checking the equivalence of the behavior of two digital circuits (repairable and non-repairable) and the method of solving for which is built according to the model at a level A1.

In the task of building diagnostic tests, the goal is summed up in a set of indistinguishable defects. At the lower level, using a single-level GA, a sequence is constructed that divides the given set into smaller ones up to those that contain exactly one defect. The difference here is also that the evaluation function shows the difference in behavior not for one couple but for several devices.

The two-level scheme is applicable not only for the case where the target of the higher level is a defect but also for a set of defects. For example, in the task of testing functional blocks of an arithmetic-logical device (DAL) [9, 12], an operation distinct from the functional level actually acts as an objective.

Then, at the lower level of the model for this operation, a set of operands is constructed in such a way that when applied to the input of the test scheme considered, the maximum coverage of all the defects considered of the structural level will be obtained. That is, the task at the lower level is again reduced to the task of checking the behavior of the device.

Analysis of known two-level GA-methods [1, 3–12] shows that at the lower level, two single-level main GA-methods are used: reaching a given state in the digital circuit and checking the equivalence of the behaviors of two or more devices.

Thus, the algorithmic implementation of all known GA-methods at two levels is based on the models of algorithms A1 and A2.

Since model A2 only shows the location of the search procedures based on GA, then without loss of generality, this model can be applied in case the search for a solution at the lower level will be carried out using another EA, for example the Simulated Annealing. From this point of view, this model can be considered common for two-level GA-methods and two-level SA-methods.

5. CONCLUSIONS

In our article we propose a classification of the evolutionary methods of construction of the identification sequences of digital circuits of different classes. Based on this classification, one- and two-level evolutionary method models have been developed, which serve as a direct basis for their algorithmic implementation. This accelerates the development of new efficient methods to generate diagnostic sequences of different classes, which are needed in the design of digital circuits. Later and obviously, this completed hierarchy should include parallel methods. And if for the GA of construction of parallel methods identification sequences SI are well developed (schemes "of islands" and "master worker"), then the approval of the SA-method methods to solve the problems considered is still to come.

REFERENCES

[1] Goldberg D.E. Genetic Algorithm in Search, Optimization and Machine Learning / Goldberg D.E. – Boston: MA: Addison-Wesley Longman Publishing Co., 1989. – 412 p.

[2] СкобцовЮ.А. Основыэволюционных вычислений / Скобцов Ю.А. – Донецк: ДонНТУ, 2008. –326 с.

[3] Corno F. Experiences in the use of evolutionary techniques for testing digital circuits / F. Corno, M. Sonza Reorda, M. Rebaudengo // Proc. of Conf. Applications and science of neural networks, fuzzy systems, and evolutionary computation. – San Diego CA, 1998. – P. 128 – 139.

[4] Corno F. Comparing topological, symbolic and GA-based ATPGs: an experimental approach / F. Corno, P. Prinetto, M. Rebaudengo [et al.] // Proceedings of the IEEE International Test Conference on Testand Design Validity. – Washington (USA), 1996. – P. 39 – 47.

[5] Corno F. A Genetic Algorithm for the Computation of Initialization Sequences for Synchronous Sequential Circuits / F. Corno, P. Prinetto, M. Rebaudengo [et al.] // Proc. ATS '01 Proceedings of the 10th Anniversary Compendium of Papers from Asian Test Symposium 1992 – 2001. – 2001. – P. 213.

[6] Hsiao M.S. Dynamic state traversal for sequential circuit test generation / M.S. Hsiao, E.M. Rudnick, J.H. Patel // ACM Transactions on Design Automation of Electronic Systems (TODAES). – 2000. – Vol.5, Is. 3. – P. 548 – 565.

[7] Yu X. Diagnostic Test Generation for Sequential Circuits / X. Yu, J. Wu, E.M. Rudnick // Proc. of International Test Conference. – Atlantic City, NJ, 2000. – P. 225 – 234.

[8] Srinivas M. A simulation-based test generation scheme using genetic algorithms / M. Srinivas, L.M. Patnaik // Proc. Int. Conf. VLSI Design. – Bombay, India, 1993. – P. 132 – 135.

[9] Skobtsov Y.A. Evolutionary Approach to Test Generation for Functional BIST / Y.A. Skobtsov,D.E. Ivanov, V.Y. Skobtsov [et al.] // 10 European Test Symposium. Informal Digest of Papers. Digest of Papers, (May 22 – 25, 2005). – Tallinn, Estonia, 2005. – P. 151 – 155.

[10] Saab D.G. Iterative [Simulation-Based+Deterministic Techniques] = Complete ATPG / D.G. Saab, Y.G. Saab, J. Abraham // Proc. Int. Conf. on Computer Aided Design. – San Jose, CA, 1994. – P. 40 – 43.

[11] Иванов Д.Е. Генетические алгоритмыв диагностикеи проектировании цифровых схем/Д.Е. Иванов, Ю.А. Скобцов, В.Ю. Скобцов// Искусственный интеллект. – 2002. – № 2. – С. 250 – 258.

[12] Иванов Д.Е. Генетические алгоритмы построения входных идентифицирующих последовательностей цифровых устройств/ Иванов Д.Е. -Донецк, 2012. - 240 с.

[13] Ivanov D.E. Distributed Genetic Algorithm of Test Generation For Digital Circuits / D.E. Ivanov,Y.A. Skobtsov, A.I. El-Khatib // Proc. of the 10th Biennial Baltic Electronics Conference. – Tallinn Technical University, 2006. – P. 281 – 284.

[14] Скобцов Ю.А. Генерация тестов для последовательностных схем с использованием кратной стратегии наблюдения выходных сигналов/ Ю.А. Скобцов, В.Ю. Скобцов, Ш.Н. Хинди// Науковий вісник Чернівецького університету. – (Серія «Фізика. Електроніка»). – 2008. – Вип. 423. – С. 29– 36.

[15] Skobtsov Yu.A. The crosstalk faults test generation / Yu.A. Skobtsov, V.Yu. Skobtsov // Proc. Of IEEE East-West Design&Test Symposium (EWDTS'10). – St. Peterburg, Russia, 2010. – P. 141 – 144.

[16] Kirkpatrick S. Optimization by simulating annealing / S. Kirkpatrick, C.D. Gelatt, M.P. Vecchi //Science. – 1983. – Vol. 220. – P. 671 – 680.

[17] SAARA: a simulated annealing algorithm for test pattern generation for digital circuits / F. Corno, P. Prinetto, M. Rebaudengo [et al.] // Proc. of the 1997 ACM symposium on Applied computing. – San Jose, California, 1997. – P. 228 – 232.

[18] ИвановД.Е. Применение алгоритмов симуляции отжига в задачах идентификации цифровых схем/ Д.Е. Иванов// Вісник Національного технічного університету "Харківський політехнічний інститут". Збірник наукових праць. Тематичний випуск: Інформатика І моделювання. – Харків: НТУ"ХПІ", 2011. – № 17. – С. 60 – 69.

[19] Maniezzo V. Ant colony optimization: an overview / V. Maniezzo, A. Carbonaro // Essays and Surveysin Metaheuristics. – Norwell: Kluwer Academic Publishers, 2002. – P. 469 – 492.

[20] Blum C. Swarm Intelligence in Optimization. Introduction and Applications. Natural Computing Series/ C. Blum, Li Xiaodong. – Springer-Verlag Berlin Heidelberg, 2008. – P. 43 – 85.

[21] Diaz E. A Tabu Search Algorithm for Structural Software Testing Computers / E. Diaz, J. Tuya // Operations Research. – 2008. – N 35 (10). – P. 3052 – 3072.

[22] Xiaojing H. Ant Colony Optimizations for Initialization of synchronous sequential circuits / H. Xiaojing, S. Zhengxiang // Proc. of IEEE Circuits and Systems International Conf. – Chengdu, China, 2009. –P. 5 – 18.

[23] Farah R. An Ant Colony Optimization Approach for Test Pattern Generation / R. Farah, H.M. Harmanani// Canadian Conference on Electrical and Computer Engineering (CCECE). – Niagara Falls. – 2008.– May. – P. 4 – 7, 1397 – 1402.

[24] Xin F. A Sequential Circuits Test Set Generation Method Based on Ant Colony Particle Swarm algorithm/ F. Xin, F. Shuai // Proc. of National Conference on Information Technology and Computer Science (CITCS 2012). – Atlantis Press, 2012. – P. 205 – 209.

[25] СкобцовЮ.А. Логическое моделированиеи тестирование цифровых устройств/ Ю.А. Скобцов, В.Ю. Скобцов. – Донецк: ИПММНАНУ, ДонНТУ, 2005. – 436 с.

[26] ИвановД.Е. Применение информации структурного уровня в алгоритмах построения идентифицирующих последовательностей/ Д.Е. Иванов// Известия ЮФУ. – (Серия «Техническиенауки»). – 2013. – № 1. – С.149 – 160.

BIOGRAPHIES

RAMZI ZOUAOUI born in Tunisia, 3rd March 1979.
Donetsk National Technical University, diploma of doctoral thesis, 16th June 2011.
Ramzi ZOUAOUI, ASSISTANT PROFESSOR in the University of Tunis El Manar in the Faculty of Mathematical, Physical and Natural Sciences of Tunis. Member of LIPAH-LR 11ES14, University of Tunis El Manar since 3rd September 2012.

Imen HERZI born in Tunisia, 8th August 1983.
Faculty of Sciences of Tunis, diploma of computer Engineer, 16th June 2008.
ASSISTANT PROFESSOR in the Higher Institute of Technological Studies in Nabeul. Member of LIPAH-LR 11ES14, University of Tunis El Manar since 5th October 2014.