UDC 004.896

Ia. Vasyliuk, V. Teslyuk, P. Denysyuk
Lviv Politechnic National University,
Computer Aided Systems Department

METHOD AND PROGRAM MODEL OF MICROELECTROMECHANICAL SYSTEMS COMPONENTS SYNTHESIS BASED ON GENETIC ALGORITHM AND ONTOLOGY MODELS

In this paper the general process and the concurrent synthesis realization model of microelectromechanical systems, which is based on developed genetic algorithm, are described. As the synthesis task in the sphere of complex microsystems is very comprehensive and time-consuming, the actuality of performance and speed issues to generate the novel system and its components constructions is still up-to-date unsolved item. The developed model facilitates and accelerates the synthesis of the new and unique microelectromechanical systems structures.

Key words: microelectromechanical systems (MEMS), computer-aided systems (CAD), ontology, genetic algorithm (GA), stochastic methods, concurrency.

Описано загальний процес і паралельну модель реалізації синтезу мікроелектромеханічних систем, що ґрунтуються на розробленому генетичному алгоритмі. Оскільки завдання синтезу у сфері комплексних мікросистем є дуже складним у своїй структурі і затратним у часі, актуальність питань виконання і швидкості, щоб згенерувати новітню систему і її структурні компоненти, є все ще невирішеною сьогодні. Розроблена модель сприяє і прискорює синтез нових і унікальних мікроелектромеханічних структур системи.

Ключові слова: мікроелектромеханічні системи (МЕМС), системи автоматизованого проектування (САПР), онтологія, генетичний алгоритм (ГА), стохастичні методи, паралелізм.

Introduction

For present microelectromechanical systems increase their important part in everyday human activities and different spheres. For instance, nowadays microsystems began to be actively applied even in micro liquid, chemical and biological systems, which are built with micromachining technologies usage, etc. The total appliance of microsystems is stipulated by great number of advantages: microsize, low costs of fabrications, reliability, and lightness [1].

Actually, for optimization and synthesis issues solving the genetic algorithm is widely used. That is why according to sphere of implementation, business, mathematics, and manufacturing many realization variations exist. Such prevalence is caused by the simplicity in genetic algorithm implementations, adaptability and quick solution search [2, 3, 4, 5]. To principal peculiarities of evolutionary algorithm belong: the need of little information for some solutions generation, ability to decode and change the designed solution and estimate the sustainability and performance of developed decision. These assets convinced to apply GA for design of new-made solution – microelectromeachanical systems.

Genetic Algorithm implementation distinctions

For microelectromechanical systems structure synthesis and their components the great quantity of methods are well-known. For MEMS synthesis and optimization as the basis was chosen the genetic algorithm. In comparison with other design approaches our developed realization of GA has a lots advantages: 1) concurrent model of implementation: 2) discrete calculation of fitness function according to externally or internally defined constraints; 3) generation of modern, non-standard designs as output.

During the search process of novel solutions and their further optimization the main principle of genetic algorithm provides to pursuit the most optimal microsystem structure or components, their relations at different system levels. The searching is fulfilled in the cycle till the moment of the best solution finding – the satisfaction of internally determined fitness function [6], the rules of MEMS elements combinations, microsystems' design laws, physical ones, and conformity to external constraints defined by designer. At Fig.1 the functioning process of genetic algorithm is illustrated:

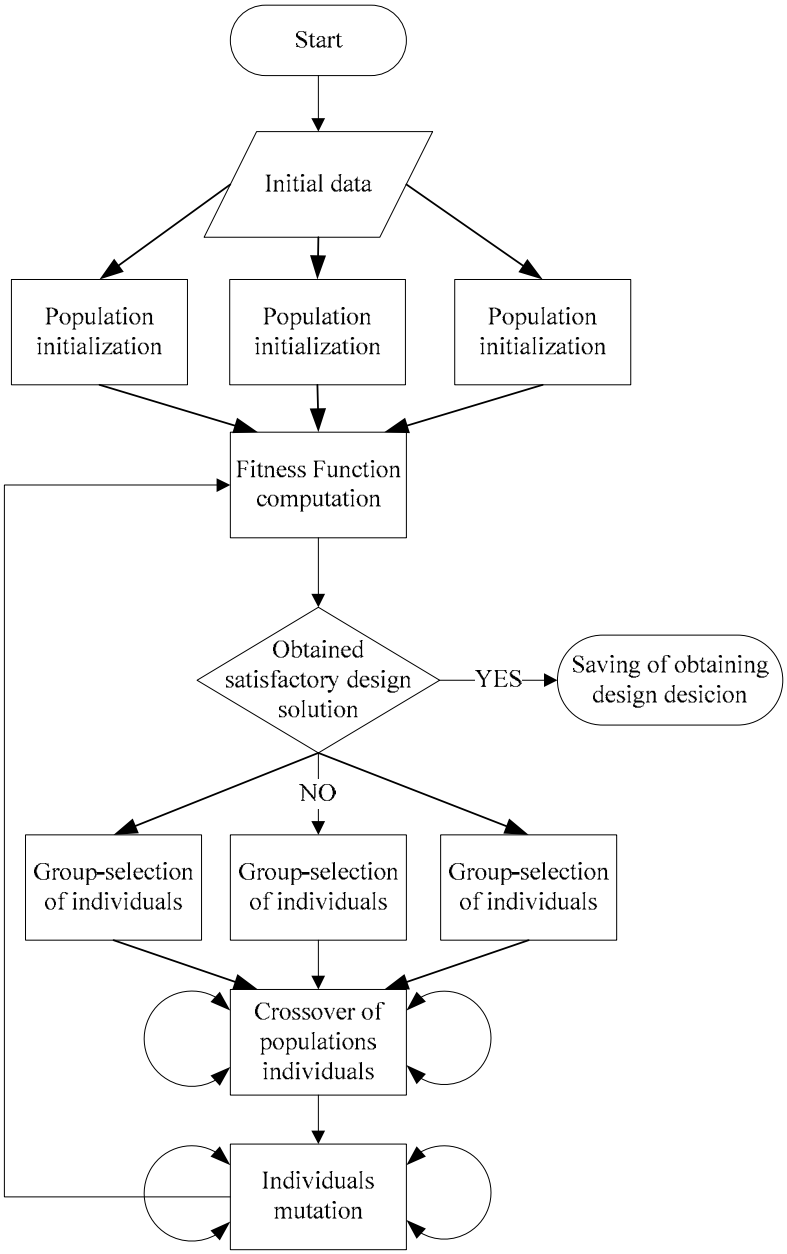

Fig. 1. Diagram of functioning concurrent model based upon genetic algorithm

The key point of successful GA work linearly depends on implementation procedure of iterative production and pursuit of the most viable individual's generation – as new populations – as new design solutions.

Concurrent GA Model

After examine of numbers of implemented synthesis algorithms models for new decisions creation as outcome was referenced to develop the specific module which should realize the concurrent model of genetic algorithm proceeding for rapid search of the optimum. As an example, for projection of new

structure and elements values of accelerator the algorithm will consist of beneath provided steps-functional blocks:

Block №1: the initialization of accelerometer elements values, based on specific template which is fixed out from accelerometer ontology model;

Block №2: the creation of several individuals-accelerometers populations in parallel threads;

Block №3: accordance computation of proliferated design to MEMS projection rules [7] and defined fitness function value;

Block №4: (in case of appropriate solution absence) the grouping of produced individuals and selection the best ones according to Pareto law [8];

Block №5: the crossover of most fitted individuals with crossover probability 0.7;

Block №6: the mutation process to new formed population, mutation coefficient – 0.13.

GA Concurrent Model Implementation

The module which is responsible for concurrent model implementation of genetic algorithm for microsystem synthesis and optimization is written in object-oriented language JAVA [9]. The main object in developed implementation of concurrent model is the *class MemsGAEn gine <T>*. In this class the initialization process of all needed objects for successful GA functionality execution is started:

public *MemsGAEngine (CandidateFactory<T> candidateFactory,*
EvolutionaryOperator<T> evolutionScheme,
*FitnessEvaluator<? **super** T> fitnessEvaluator,*
*SelectionStrategy<? **super** T> selectionStrategy,*
Random rng)
{};

where *1) Cand idateFactory<T>* – the creation of elementary population based upon individulas – ready-made design decisions (obtained from ontology models);

2) EvolutionaryOperator<T> – the set of operator for generation of novel populations;

*3) FitnessEvaluator<? **super** T>* – the computation of fitness function logic for every individual-decision;

*4) SelectionStrategy<? **super** T>* – the selection strategy of the most appropriate generated solutions;

5) Random rng – the probability coefficient applied to individuals in their creation and selection stages.

For possibility at back-end side to change grouping of design solutions, the specific selection carrying out, and mutation it was applied the methodologies of evolutionary strategies, which are implemented upon the *class MemsEvolutionStrategy<T>,* and "islands" models. This developed model enables to isolate the determined number of populations and as result to synthesize numbers of fresh, non-standard design decisions of microelectromechanical systems and its components structures. In case of solution absence automatically the deisolation process starts automatically to the populations, and the evolutionary strategy is used to combine the individuals which almost appropriate to fitness function value. And the GA cycle launches again.

Results of GA concurrent model functioning to MEMS Synthesis

Afterwards of GA concurrent model implementation numbers of experiments were passed for test purpose and definition of two key milestones: a) time costs of sequential and concurrent GA models for microsystems synthesis and optimization of its structures; b) the population quantity in which was obtained the most sufficient and satisfactory solutions to input constraints. The table №1 presents the correlation of two genetic models:

Table 1

Time spent metrics correlation of sequential and concurrent GA models

Population number, n	Time costs (minute)		Decreased time costs	% correlation
	Sequential model	Concurrent model		
2	~46	~35	11	23,91
6	~109	~85	24	22,01
9	~180	~127	53	29,44
14	~286	~204	82	28,67
19	~477	~394	83	17,40

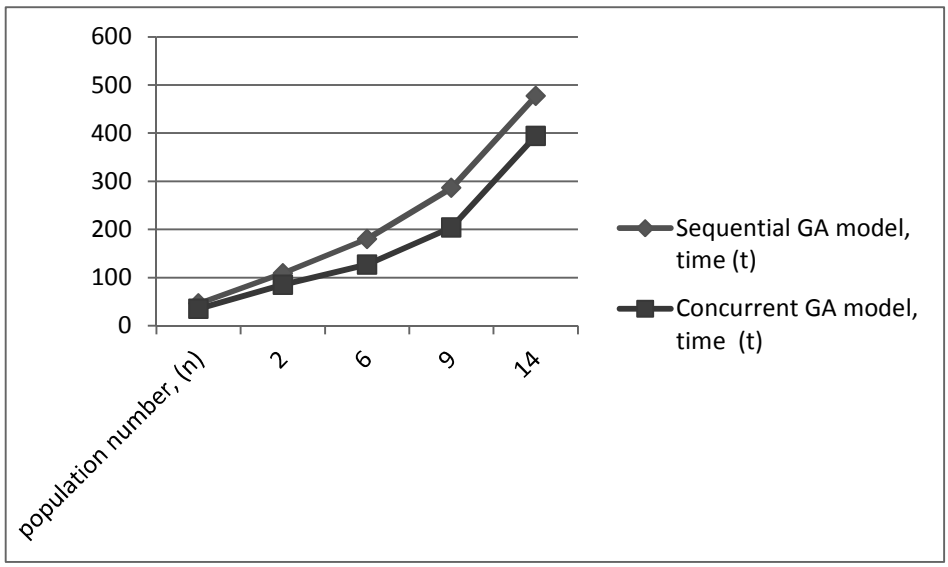

Fig. 2. Correlation diagram of sequential and concurrent GA models functioning

Based upon obtained outcomes of sequential and parallel genetic models it can be summarized that the most suitable design solutions were found in created fourteen populations of individuals where the spent time metric in percentage interrelation reached 29.44 value.

Conclusion

The genetic algorithm is grounded upon of nature principles. Every produced population mimics the searched solution which is mostly appropriate to externally defined fitness function and input design constraints. Via selection, crossover and mutation processes the accommodation of populations' individuals is provided. Thereby, genetic algorithm enables to find quickly design solutions.

According to requirements at the initial stage of concurrent GA model implementation was developed software module. It simulated the GA process appropriately to MEMS design specific rules with use of JAVA language. The obtained results were presented in table 1 and figure 2, and enable to made strict conclusion that concurrent GA model decreased the time cost (on average on ~24%) at novel microsystems and its elementary structure synthesis.

1. Теслюк В.М. Моделі інформаційних технологій синтезу мікроелектромеханічних систем: Монографія. – Львів: Вид-во ІПІ "Вежа і Ко", 2008. – 192 с. 2. Deb K. Multi-Objective Optimization Using Evolutionary Algorithms, Wiley and Sons, Inc., New York, NY, 2001. 3. Goldberg D.E. Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, 1989. 4. Tamaki H., Kita H., Kobayashi S. "Multi-Objective Optimization by Genetic Algorithms: A Review", Proc. of 1996 IEEE Int. Conf. on Evolutionary Computation (ICEC'96), 1996, pp. 517–522. 5. Coello C.A. "An Updated Survey of Evolutionary Multiobjective Optimization Techniques : State of the Art and Future Trends," 1999 Congress on Evolutionary Computation, Vol. 1, Washington, D.C., July 1999. 6. Гладков Л.А., Курейчик В.В., Курейчик В.М. Генетические алгоритмыю. – М.: Физматлит, 2006. – 320 с. 7. Lindroos V., Tilli M., Lehto A., Motook T. Silicon-On-Glass MEMS Design Handbook. Michigan, 2007. – 26 p. 8. http://en.wikipedia.org/wiki/Pareto's_law. 9. http://docs.oracle.com/javase/specs/