Slovak University of Technology in Bratislava
Institute of Information Engineering, Automation, and Mathematics

PROCEEDINGS

of the 18th International Conference on Process Control

Hotel Titris, Tatranská Lomnica, Slovakia, June 14 – 17, 2011

ISBN 978-80-227-3517-9

http://www.kirp.chtf.stuba.sk/pc11

Editors: M. Fikar and M. Kvasnica

Ošmera, P.: Transplant Evolution for Optimization of General Controllers, Editors: Fikar, M., Kvasnica, M., In *Proceedings of the 18th International Conference on Process Control*, Tatranská Lomnica, Slovakia, 366–372, 2011.

Full paper online: http://www.kirp.chtf.stuba.sk/pc11/data/abstracts/059.html

Transplant Evolution for Optimization of general Controllers

Jindřich Petrucha
European Polytechnical Institute Kunovice,
Osvobození 699, 686 04 Kunovice, Czech Republic
e-mail:petrucha@edukomplex..cz

Pavel Ošmera
Institute of Automation and Computer Science
Brno University of Technology
Faculty of Mechanical Engineering
Brno, Czech Republic
osmera@fme.vutbr.cz

Milos Seda
Institute of Automation and Computer Science
Brno University of Technology
Faculty of Mechanical Engineering
Brno, Czech Republic
seda@fme.vutbr.cz

ABSTRACT
This paper describes a new method of evolution that is named Transplant Evolution (TE). None of the individuals of the transplant evolution contains genotype. Each individual of the transplant evolution contains only phenotype. Reproduction methods as crossover and mutation work and store only the phenotype. The hierarchical structure of grammar-differential evolution that is used for finding optimal structures and parameters of general controllers is described. structure of solution with unknown abstract parameters and the DE optimizes the parameters in this structure. The parameters are real numbers. The real numbers are not easy find directly in TE without DE. For evaluation of quality of found control equation are described new methods, which allow us evaluate their quality. It can be used in the case when the simulation of control process cannot be finished. In results are shown some practical application. In all results we received the control equation that reached better quality of control process, than classical PSD controllers and Takahashi`s modification of PSD controller.

Categories and Subject Descriptors
D.3.2

General Terms
Algorithms, Design

Keywords
Transplant evolution, grammatical-differential evolution, object trees, hierarchical structures, algebraic reducing of trees, crossover by linking.

1. INTRODUCTION
The aim of this paper is to describe a new optimization method that can create control equations of general regulators. For this type of optimization a new method was created and we call it Two-Level Transplant Evolution (TLTE). This method allowed us to apply advanced methods of optimization, for example direct tree reducing of tree structure of control equation. The reduction method was named Arithmetic Tree Reducing (ART). For optimization of control equations of general controllers is suitable combine two evolutionary algorithms. Main goal in the first level of TLTE is the optimization of structure of general controllers. In the second level of TLTE the concrete parameters are optimized and the unknown abstract parameters in structure of equations are set. The method TLTE was created by combination of Transplant Evolution method (TE) [1,2,3,8,9,10] and Differential Evolution method (DE) [7]. The Transplant Evolution (TE) optimizes

2. THE PRESENTATION OF OBJECT TREE STRUCTURES

The phenotype representation of the individual is stored in the object tree structure. Each of nodes in the tree structure, including the sub-nodes, is an object that is specified by a terminal symbol and the type of terminal symbols. All nodes are independent and correctly defined mathematical functions that can be calculated, e.g. the function x-3, shown on Fig. 1, is a tree structure containing a functional block (sub-tree).

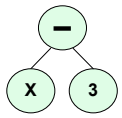

Fig. 1. Function block

Creating the object tree is a key part of GEOS, which this method differs from other evolutionary algorithms. When the object tree is generated, similar methods to a traditional grammatical evolution are used. But the GEOS does not store the genotype, because the production rules are selected by randomly generated genes that are not saved in chromosomes of individuals. The final GEOS's individual contains only phenotype expressed in an object tree structure.

The algorithm of GEOS uses a generative grammar [4,5,6] whose translation process starts from the initial symbol *S* and continues randomly with using the rules of defined grammar [2]. The basic procedure of the translation algorithm is shown on Fig. 2 where is explain to why is unnecessary to store the genotype.

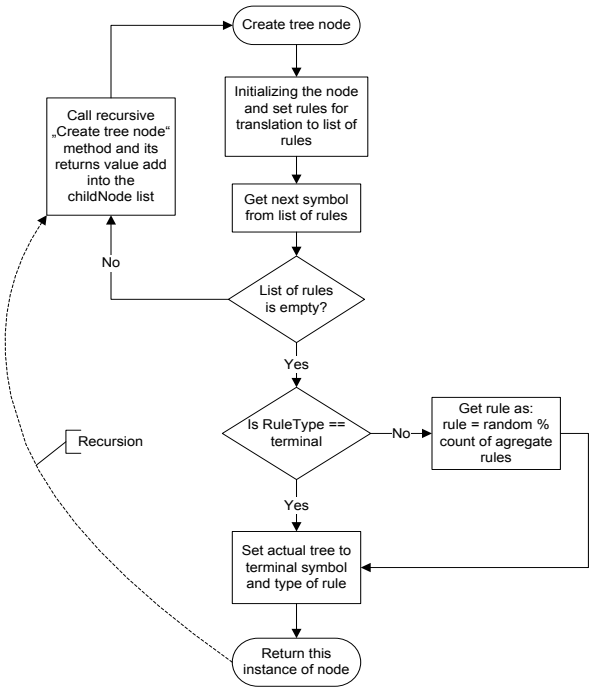

Fig. 2 Flowchart creation of object tree

3. CROSSOVER

The crossover is a distinctive tool for genetic algorithms and is one of the methods in evolutionary algorithms that are able to acquire a new population of individuals. For crossover of object trees can be used following methods:

Crossover the parts of object trees (sub-trees)

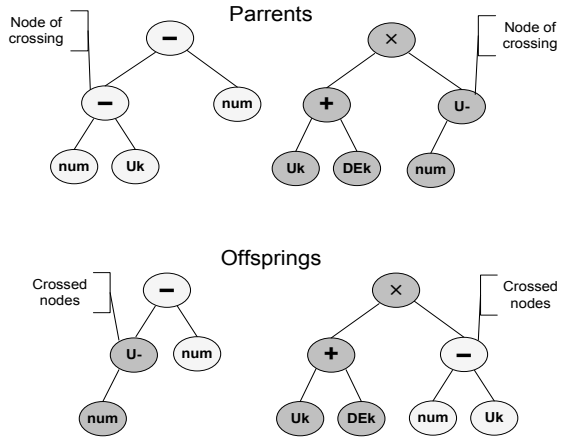

Fig. 3. Classical Crossover (CC)

The method of crossover object trees is based on the selection of two parents from the population and changing each other part of their sub-trees. For each of the parents cross points are randomly selected and their nodes and sub-trees are exchanged. This is the principle of creating new individuals into subsequent population as is shown on Fig. 3.

Crossover by linking trees or sub-trees

This method, as well as the previous one, is based on the crossover of two parents who are selected from the previous population. But the difference is in the way how the object trees are crossed. This method, unlike the previous one, does not exchange two randomly selected parts of the parents but parts of individuals are linked together with new and randomly generated node. This node will represent a new root of the tree structure of the individual. This principle is shown on Fig. 4.

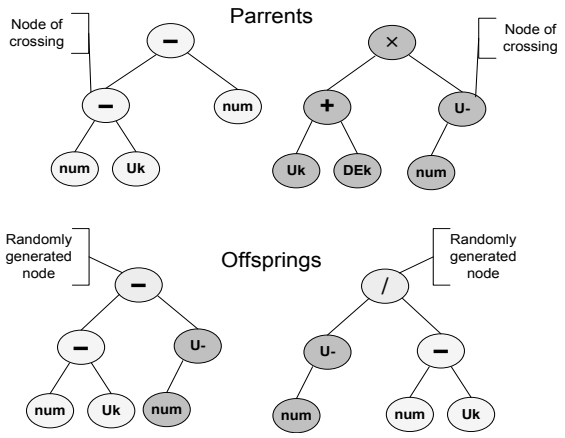

Fig.4. Crossover by linking method

4. MUTATION

Mutation is the second of the operators to obtain new individuals. This operator can add new structures, which are not included in the population so far. Mutation is performed on individuals from the old population. In the selected individual are randomly chosen nodes which are then subjected to mutation. The mutation operator can be subdivided into two types:

- Non-structural Mutation (NM)
- Structural Mutation (SM)

Non-structural Mutation (NM)

Non-structural mutations do not affect the structure of already generated individual. In the individual who is selected for mutation, chosen nodes of object sub-tree are further subjected to mutation. The mutation will randomly change chosen nodes, whereas used grammar is respected. For example it means that mutated node, which is a function of two variables (i.e. + - × ÷) cannot be changed by node representing function of one variable or only a variable, etc. see Fig. .

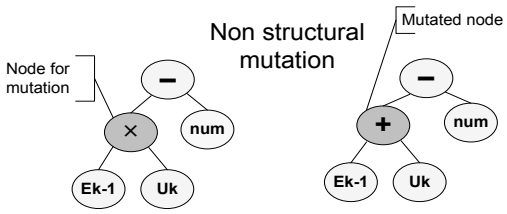

Fig. 5. Nonstructural mutation

Structural Mutation (SM)

Structural mutations, unlike non-structural mutations, affect the tree structure of individuals. Changes of the sub-tree by extending or shortening its parts depend on the method of structural mutations. Structural mutation can be divided into two types: Structural mutation which is extending an object tree structure (ESM) and structural mutation which is shortening a tree structure (SSM). This type of mutation operator can be subdivided into two types:

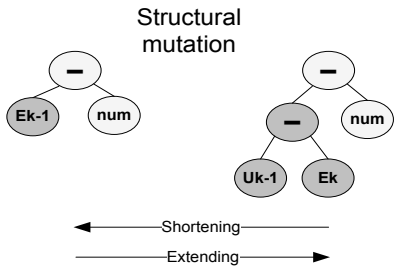

Fig. 6. Structural mutation

Extending Structural Mutation (ESM)
In the case of the extending mutation, a randomly selected node is replaced by a part of the newly created sub-tree that respects the rules of defined grammar (see fig. 3). This method obviously does not always lead to the extension of the sub-tree but generally this form of the mutation leads to extension of sub-tree. (see Fig.).

Shortening Structural Mutation (SSM)
Conversely the shortening mutation replaces a randomly selected node of the tree, including its child nodes, by node which is described by terminal symbol (i.e. a variable or a number). This type of mutation can be regarded as a method of indirectly reducing the complexity of the object tree (see Fig.).

The complexity of the tree structure can be defined as the total number of objects in the tree of individual.

5. DIRECT TREE REDUCTION

The minimal length of an object tree is often one of the factors required in the optimal problem solution. This requirement can be achieved in several ways:

- By penalizing the part of the individual fitness which contains a complex object tree,
- Method of targeted structural mutation of individual (see SSM),
- The direct shortening of the tree using algebraic adjustments - algebraic reducing tree (ART).

The last-mentioned method can be realised by the GEOS, where all of individuals does not contain the genotype, and then a change in the phenotype is not affected by treatment with genotype. The realisation of above mentioned problem with individual, which use genotype would be in this case very difficult. This new method is based on the algebraic arrangement of the tree features that are intended to reduce the number of functional blocks in the body of individuals (such as repeating blocks "unary minus", etc.). The method described above is shown on Fig. and Fig. .

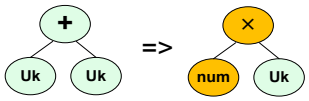

Fig. 7. ART – substitution of nodes

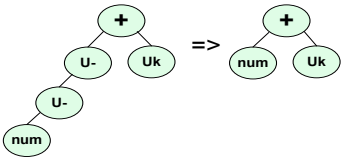

Fig. 8. ART – reduction multiple unary minus

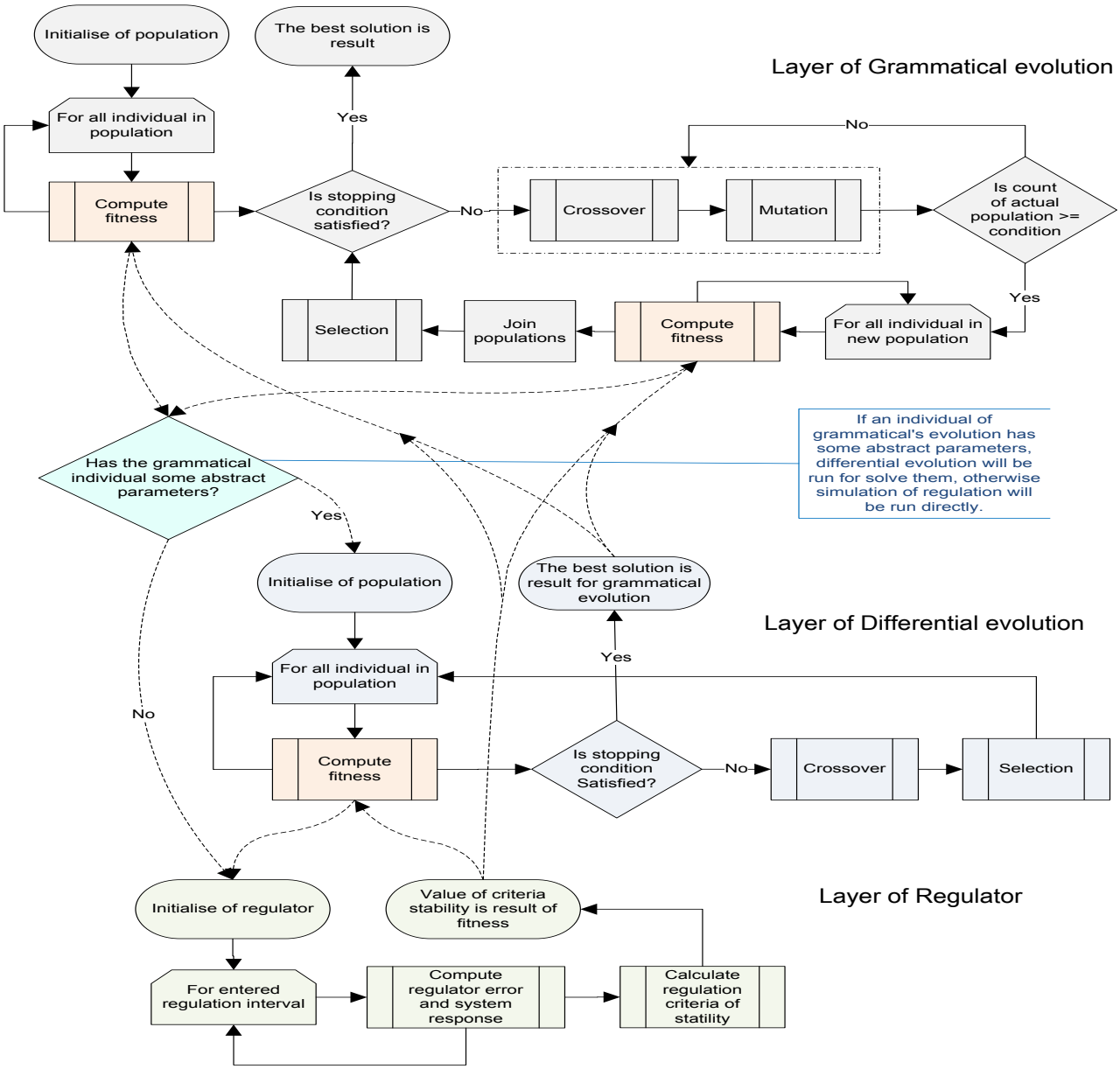

Fig. 9. Flowchart of TE (GDEOS) for controller

In view of the object tree complexity of the individual and also for subsequent crossover is preferable to have a function in the form $x = 3a$ than $x = a + a + a$, or more generally $x = n \times A$. Another example is the shortening of the function $x = -(-a)$, where is preferable to have the form $x = a$ (it is removing redundant marks in the object tree individual). The introduction of algebraic modifications of individual phenotype leads to the shorter result of the optimal solution and consequently to the shorter presentation of the individual, shortening the time of calculation of the function that is represented in object tree and also to find optimal solutions faster because of higher probability of crossover in the suitable points with higher probability to produce meaningful solutions. The essential difference stems from the use of direct contraction of trees, which leads to significantly shorter resulting structure than without using this method.

6. HIERARCHICAL STRUCTURE OF TE (GDEOS) FOR OPTIMISATION OF THE CONTROLLER

The hierarchical structure of the transplant evolution can be used for optimisation of the structure and parameters of a general controller. This structure contains three layers. First two layers (GE + DE) are contained in TE. Those two layers are used for

optimisation of the structure and parameters of general controller. The third layer which is named layer of controller is used for computation of fitness in TE.

At the beginning of GDEOS an initial population is created (see Fig. 2) and then fitness of individuals is calculated. In the case of finding the optimal solution in the first generation, the algorithm is terminated, otherwise creates a new population of individuals by crossover and mutation operators, with the direct use of already created parent's object tree structures (it is analogy as transplantation of already created organs, without necessary know-ledge of DNA – "Transplant Evolution (TE)"). If the result of GDEOS needs some numerical parameters (for example *num* in (Weisser 2010)), the second level with Differential Evolution (DE) is used for optimization their parameter setting. The DE gives better results in finding optimal values of unknown numerical parameters that are expressed in the form of real numbers, then in the GE. Due to the use of GDEOS for optimization of controllers in the next stage of calculation of fitness is model of controller used which is represented by the equation in incremental form (recurrent algorithm). Quality of controller is determined depending on the type of criterial function (see equation 3). For fitness calculation are various criterial functions used. Basic criterion is linear control area, quadratic control area, linear or quadratic control area extended with overshoot, oscillation of action value of the controller.

The flowchart of TE (GDEOS) for a controller is shown on Fig.

7. RESULTS

The TE and TE + ART methods for optimization of equation for general controller were compared.

The resulting form of the recurrent equation of general controller without using the direct method shortening of the tree (ART) is following (equation 1):

$$u_k=((((((E_k-(-((E_k+E_k))))\times3)\times2)+E_{k-3})-(-(2)))-(-(((((((E_{k-3}+((((((E_k\times3)+(E_k+(((dE_{k-1}+E_k)\times2)\times1.63)))\times2)\times2)+(-((((E_{k-4}+(E_k-(-(3))))+E_{k-2}))))\times3))+(-(((E_{k-4}+(E_k-E_k))+E_k-2))))\times2)+((dE_{k-1}+(4.47-((((E_k-(-((((((((((E_k\times3)+(E_k+(((dE_{k-1}+E_k)\times2)\times2)))\times2)\times2)\times2)+(-(3.61)))+E_k)+E_k)+(-((3+E_{k-2})))))))+((E_k\times2)+E_k))+E_k)-(-((((E_k+2)\times(3\times((E_{k-4}+2)+E_{k-2})))))))-((((E_k+((((E_k+2)+E_k)+E_k)+(E_k/(-((6.88-((dE_{k-1}+(1.79-E_{k-3}))-2)))))))\times2)\times3)+(-(((E_{k-4}+((E_k+E_k)\times2))+E_k))))))+E_k)-(-(E_{k-2})))))$$

(1)

The resulting form of the recurrent optimization algorithm in the case with using the direct method of contraction tree is following (equation 2):

$$u_k = (E_{k-3} - E_k \times 1.93) \times 33.97 + E_{k-1} + E_{k-2} \quad (2)$$

As you can see, the resulting lengths of recurrent equation of the general controller, is shorter in case of using TE + ATR then TE without ART.

Bellow is shown result of optimisation parameters of PSD controllers and optimisation of the structure and parameters of general controllers. The parameters of PSD controllers were optimised with using DE and structure and parameters of general controller were optimised with using TE + ART method.

The basic criterion of minimal integral control area was used as criterial function for optimisation of PSD or general controllers, (see equation 3)

$$J = \int_0^\infty |e(t) - e(\infty)|\, dt \quad (3)$$

On the Fig. and Fig. are result of optimisation of PSD controller and general controller to control the identical system with 5 second time delay.

On the Fig. is shown regulatory process of PSD controller. The parameters of PSD controller were optimised with using DE.

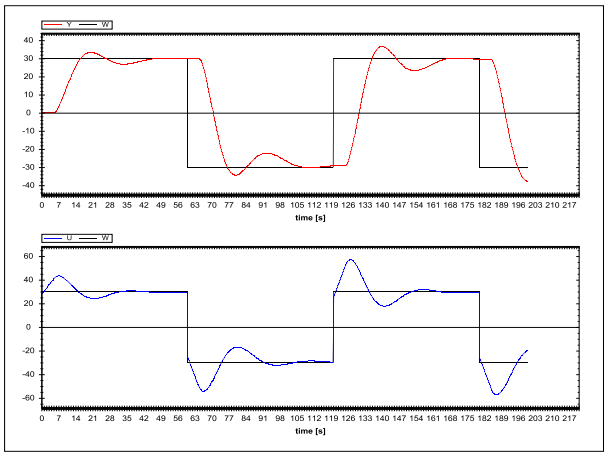

Fig.10. Regulatory process of PSD controller for second order system with 5s time delay

(Top figure shows the system response and on the bottom part of the figure is shown the action output of controller)

On the Fig. is shown regulatory process of general controller. The structure and parameters of this controller was optimised with using TE (GDEOS) + ART method. The equation of general controller is following (see equation 4).

$$u_k = E_k \times 23.01 + 10.91 \times E_{k-3} + E_{k-1} \times (-33.91) + U_{k-1} \quad (4)$$

On the Fig. and

Fig. are result of optimisation of PSD controller and general controller to control the identical system with 2 second time delay.

On the Fig. is shown regulatory process of PSD controller. The parameters of PSD controller were optimised with using DE.

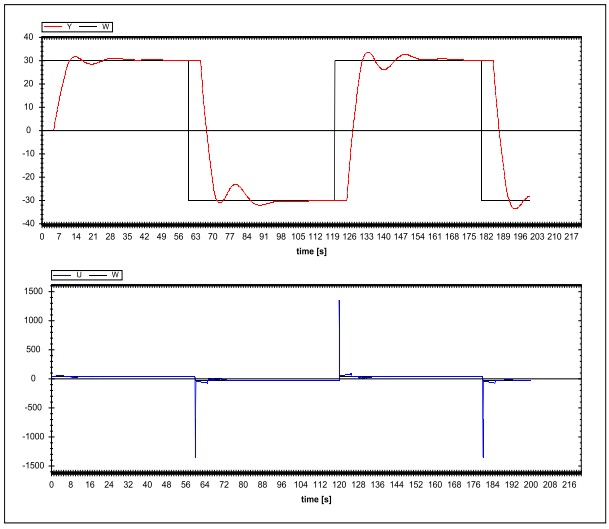

Fig. 11 Regulatory process of general controller for second order system with 5s time delay

(Top figure shows the system response and on the bottom

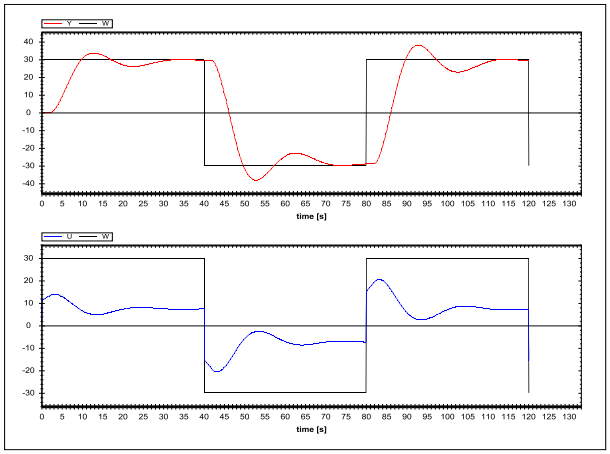

Fig. 12 Regulatory process of PSD controller for second order system with 2s time delay

(Top figure shows the system response and on the bottom part of the figure is shown the action output of controller)

On the Fig.13 is shown regulatory process of general controller. The structure and parameters of this controller was optimised with using TE (GDEOS) + ART method. The equation of general controller is following (see equation 5).

$$u_k = (32.55 \times E_{k-5}) + (58.79 \times E_k) + 7.08 + (-89.97 \times E_{k-2}) \qquad (5)$$

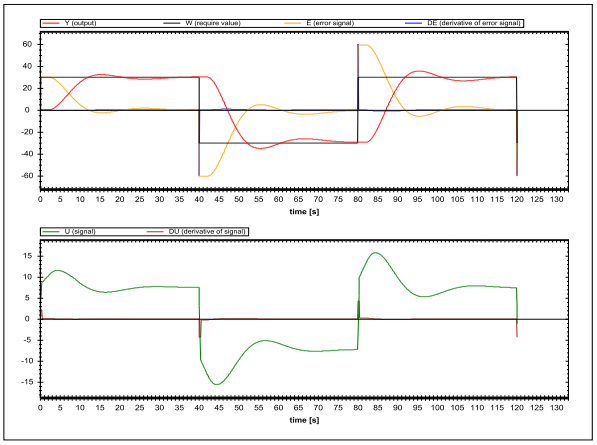

Fig. 13. Regulatory process of general controller for second order system with 2s time delay

(Top figure shows the system response and on the bottom part of the figure is shown the action output of the controller)

We tested the TLTE method for optimization of recurrent equation of general controllers. There is some results of optimization for one following system:.

Integral system with transport delay

$$G_S(S) = \frac{5}{s(9s+1)} e^{-2s} \qquad (5)$$

In Fig.14 we compare 3 types of controllers. There is one PSD controller marked PSD_DE and two general controllers marked General_DE and General_TLTE. The curve marked PSD_DE is PSD controller. Parameters (Kr, Ti, Td) of this controller were optimized by Differential Evolution (DE). The curve marked General_DE is for general controller which has the control equation in PSD equation form, but parameters q0, q1, q2 were optimized directly by DE. The curve marked General_TLTE is for general controller with general control equation that was optimized by Two-Level Transplant Evolution (TLTE). As you can see, the best result gives the General_TLTE. In this case we receive the recurrent control equation with following form:

```
Uk = (Ek_5+(((-((Ek_5*(-4,69210604912152))-(-(((((Yk*5,14847786853854)*
     Yk_3*1,59643919322167))+((Ek_5+22,2867094434306)*Ek_2))*
     (Uk_2*(-3,80995100289523E-06))))))+((11,9436857350138*Ek)+
     (((-0,322492294234783)*Uk_3)+((-18,5463669625012)*Ek_2))))+
     (Ek_1+(-0,00399257533723234))))
```

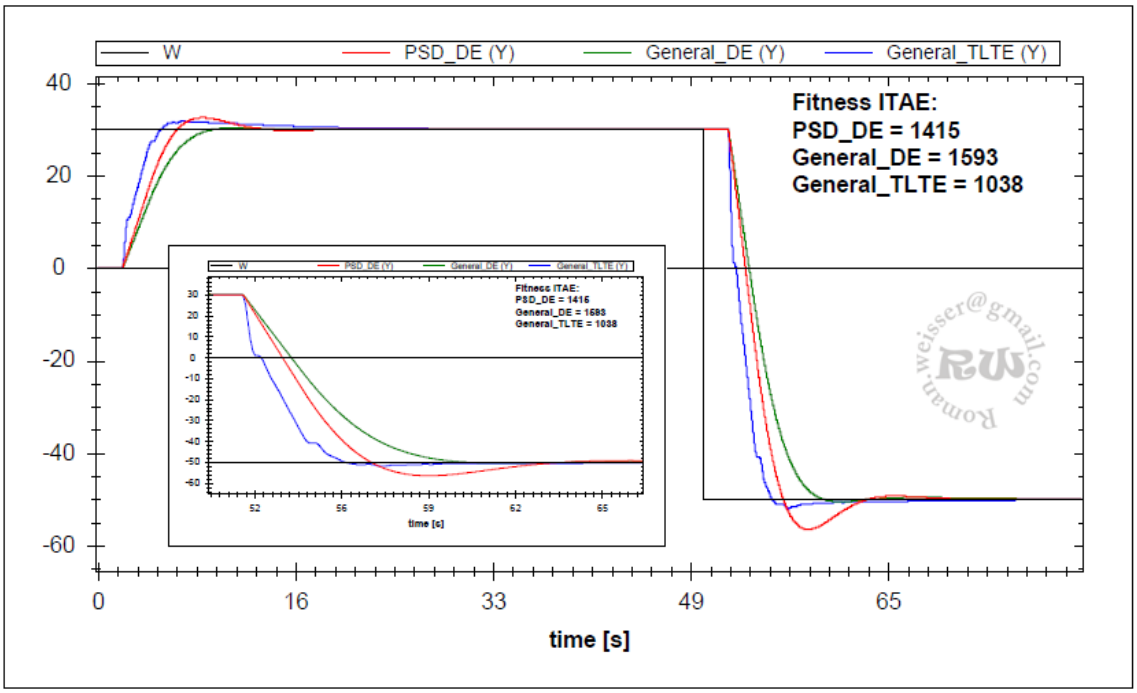

Fig.14. Step response for integration system with time delay

8. CONCLUSION

The Two-Level Transplant Evolution (TLTE) was successfully use for automatic generation of control programs of general controllers. We tested this algorithm on many problems, only one example was described in this paper. We hope that this new method of controller design will be use in practice, not only for simulation.

Although we are at early stages of experiments, but it seems that it is possible to use parallel grammatical evolution with backward processing to generate combinatorial logic circuits. The grammatical algorithm can be outperformed with algorithms, which are designed specifically for this purpose.

9. ACKNOWLEDGMENTS

This work has been supported by Czech Ministry of Education No: MSM 00216305529 Intelligent Systems in Automation and GA ČR No: 102/09/1668.

10. REFERENCES

[1] Koza J.R. 1992: Genetic Programming: On the Programming of Computers by Means of Natural Selection, The MIT Press

[2] Kratochvíl O. and Ošmera P. and Popelka O. 2009: Parallel grammatical evolution for circuit optimization, in Proc. WCECS, World Congress on Engineering and Computer Science, San Francisco, 1032-1040.

[3] Li Z. and Halang W. A. and Chen G. 2006: Integration of Fuzzy Logic and Chaos Theory; paragraph: Osmera P.: Evolution of Complexity, Springer, 527 – 578.

[4] O'Neill M. and Ryan C. 2003: Grammatical Evolution: Evolutionary Automatic Programming in an Arbitrary Language Kluwer Academic Publishers.

[5] O'Neill M. and Brabazon A. and Adley C. 2004: The Automatic Generation of Programs for Classification Problems with Grammatical Swarm, Proceedings of CEC, Portland, Oregon, 104 – 110.

[6] Piaseczny W. and Suzuki H. and Sawai H. 2004: Chemical Genetic Programming – Evolution of Amino Acid Rewriting Rules Used for Genotype-Phenotype Translation, Proceedings of CEC, Portland, Oregon, 1639 - 1646.

[7] Price K. 1996. Differential evolution: a fast and simple numerical optimizer, Biennial Conference of the North American Fuzzy Information Processing Society, NAFIPS, IEEE Press, New York, NY, 524-527.

[8] Rukovanský I. Optimization of the throughput of Computer Network Based on Parallel EA. In Proceedings of the World Congress on Engineering and Computer Science WCECS 2009, San Francisco, CA, Oct. 20-22, 2009, Vol. II, pp. 1038-1043

[9] Weisser R., Ošmera P., Matoušek R., Transplant Evolution with Modified Schema of Differential Evolution: Optimization Structure of Controllers. In International Conference on Soft Computing MENDEL. Brno : MENDEL, 2010.

[10] Weisser R., Ošmera P., Šeda, M., Kratochvíl, O. Transplant Evolution for Optimization of General Controllers. In European Conference on Modelling and Simulation. 24th. Kuala Lumpur (Malaysia) : ECMS 2010. s. 250 -- 260.