Evolutionary synthesis of nonlinear models based on metaheuristic programming and templates

Oleg Monakhov
Institute of Computational Mathematics and Mathematical Geophysics, SB RAS, Novosibirsk, Russia
E-mail: monakhov@rav.sscc.ru

Abstract. The application of a metaheuristic programming (MP) algorithm based on various bio-inspired algorithms for the evolutionary (metaheuristic) synthesis of nonlinear models is considered. The described approach of evolutionary synthesis uses a sequential operator structure of the chromosome, templates (with some undefined functions and parameters) of the models and the given sets of pairs of input and output data. The influence of the degree of specialization of the template on the characteristics of the search algorithm in the metaheuristic synthesis of nonlinear models defined by ordinary differential equations with undetermined functions and parameters is studied. Estimates of the complexity of this algorithm are obtained, and a significant reduction in the search time when using templates in the process of the evolutionary synthesis is shown.

1. Introduction and Basic Definitions
A problem of the synthesis of mathematical expressions (formulas) and parameters for representation of nonlinear models based on a given template of the model, experimental data, sets of variables, basic functions and operations is considered. A template (skeleton) is a control parameterized structure of the algorithm that describes the scanning order of the data structures of the algorithm and determines the dynamics of the computing process in space-time coordinates. The template $TM$ of the model is a mathematical description of the model with some undetermined functions and parameters $f^*$. It is needs to find a mathematical expression $f$ which best describes a nonlinear computation model defined by the given template $TM$, a set of the input $(X)$ and output $(Y)$ experimental data, i.e., it is necessary to select a function (expression) $f$ that a model $Y = TM(f, X)$ would represent the dependence of $Y$ on $X$ with a minimum error (also, this problem is called the symbolic regression).

For constructing the expression $f$, we use a given set of elementary (basic) functions and operations $F_1 = \{ f_i \mid f_i : R \times \ldots \times R \to R \}$ and a given set of variables and constants $T = \{ x_i, c_i \}$, and also we use bio-inspired (nature-inspired) algorithms for the automatic creation and optimization of analytical expressions (formulas) representing the model. The objective function (the fitness function) $FF$ calculates the following error of the model with the used function $f$: 

$$FF = \sum_{i=1}^{N} (TM(f, X_i) - Y_i)^2$$, 

where $TM(f, X_i)$ is the value of the model $Y$ for the input $X_i$.
where \( N \) is the number of the experimental data. The algorithm is aimed at determining:

\[
\min_{f \in D(F_1, T)} FF(f),
\]

where \( D(F_1, T) \) is the set of models defined by a set of basic functions and operations and a set of variables and constants.

For solving this problem (as symbolic regression without templates) the genetic programming (GP) \[1],[2\] is mainly used, which is based on a genetic algorithm for automatic synthesis of programs. Genetic operators in the GP use tree structures of chromosomes to represent after an interpretation some expressions and functions.

In this study, we use a new metaheuristic programming method (MP) for the evolutionary synthesis of nonlinear models, which is based on the use of: (1) representing the genotype of nonlinear models as a simple vector of real numbers (rather than representing more complex structures — trees, networks or programs traditional for genetic programming), (2) new algorithms for converting these vectors (genotype) to phenotype (expressions and programs for representing a nonlinear model), (3) organization of the evolutionary search process on the set of these vectors using the usual metaheuristic, bio-inspired optimization algorithms \[3\], \[4\], with simple evolutionary search operators (rather than specialized operators for more complex structures), (4) a multivariate coding method for several solutions in one genotype.

Note that the MP method is a further development and generalization of the multivariate evolutionary synthesis (MVES) method \[5\], \[6\] proposed earlier by the author, it is more universal and unified for the use with many metaheuristic (bio- and nature-inspired) algorithms, and does not only use one discrete genetic algorithm (GA), as the MVES method, but uses chromosomes as a vector of real numbers, and not as a vector of integers as in the MVES method and, accordingly, uses other chromosome decoding operations.

2. Metaheuristic Programming for the Nonlinear Models Synthesis

Metaheuristic programming for the synthesis of nonlinear models is based on nature-inspired (evolutionary) computations and modeling of natural processes in the population of individuals, each one being a point in the space of solutions to the optimization problem \[3\], \[4\]. Individuals (agents, ants, bees, fireflies, particles, students, etc.) are data structures (chromosomes), namely sequences of real numbers (vectors) that encode mathematical expressions (formulas and programs). Each population is a set of chromosomes, and each chromosome in this algorithm determines a set of expressions (formulas) arising from it after decoding. The main idea of the synthesis algorithm is the evolutionary transformation of a set of chromosomes (formulas) in the process of natural selection for the survival of the "strongest" (with the extremum value of the objective function). To create the next generation of the population (subsequent iteration), new individuals are created using migration operators (in the case of the standard genetic algorithm, the migration operators are selection, mutation and crossover). The objective of the operator migration is to move towards the extremum of the objective function.

The stages of simulation of the nature-inspired process in the population algorithms are based on the following template:

1. Creating an initial population from randomly generated chromosomes as sequences of real-valued numbers.
2. Evaluating the population by the fitness function, which shows how well each individual solves a given problem. In so doing, a genotype (a real-valued vector) is decoded to a phenotype (a function and a program) for the calculation of a fitness function.
3. Creating a next-generation population using the migration (evolutionary) operators (specified for each nature-inspired algorithm) to move the individuals in this population to the extremum of a objective function.
4. Repeating points 2 and 3 until a solution meeting specified criteria is found or the maximum number of generations is reached.
In the metaheuristic programming proposed, a unified approach is offered to decode the main data structures, chromosomes for various nature-inspired algorithms. This approach relies on representing the chromosome linear structure as a sequence of three-address instructions and producing a linear operator structure for calculating expressions (formulas) encoded in the chromosome. To this end, a sequence of real-valued numbers (a chromosome) is divided into groups of three elements (triplets) \((h_1, h_2, h_3)\) with \(0 < h_i < 1\). Each such group is interpreted as a three-address instruction as follows: \(<oper><adr_1><adr_2>\), where the operation \(oper\) is applied to operands in the instructions with the numbers \(adr_1\) and \(adr_2\), which are calculated by the following formulas:

\[
\begin{align*}
oper & = \lfloor h_1 \cdot |F| \rfloor, \\
adr_1 & = \lfloor (h_2 \cdot (I-1)) \rfloor + 1, \\
adr_2 & = \lfloor (h_3 \cdot (I-1)) \rfloor + 1, \\
\text{if } oper = 0 \text{ then } adr_1 & = \lfloor (h_2 \cdot |T|) \rfloor + 1,
\end{align*}
\]

where \(|F|\) is the cardinal number of a set of basic functions, \(oper\) is the element number in this set, i.e., the number of the function being executed in the current instruction, \(I\) is the number of the current instruction, \(adr_1, adr_2\) are the numbers of the preceding instructions, the execution results whose are used as operands in the current instruction, and \(|T|\) is the cardinal number of the terminal set. If \(oper=0\), then the instruction is interpreted as a loading operator, and the terminal symbol with the number \(adr_1\) is loaded.

Thus, decoding the chromosome into an expression (function) leads to the representation of the function in the form of an interpreted code. Each code instruction will be considered as a separate function, which may include all the previous instructions. The first operation will always call a variable or constant in this function. At the run time, only the previos instructions with lower numbers are used for the possible arguments of the current instruction. The genetic solution in this case is a set of sequences from the first operator to each current one. This allows one unlike the standard GP, to simultaneously evaluate a set of expressions in the form of a sequence of the operators and reduce the time to search for the optimal solution. Here, the evaluation of a particular chromosome from a set of obtained variants (versions) is selected as an expression evaluation that has a minimum value of the objective function.

The advantage of this approach is a simple algorithm for decoding a real-valued chromosome (vector) into a set of commands that allows these vectors to be used within a unified approach to finding the optimal models using various nature-inspired population algorithms \([3],[4]\). In this study, within metaheuristic programming approach, three different nature-inspired algorithms were used to search for optimal models: genetic algorithm (GA)\([3],[4]\), differential evolution (DE)\([7],[8]\) and particle swarm optimization (PSO)\([9]\).

### 3. Experimental Results

We investigate the efficiency of the metaheuristic programming using three different nature-inspired algorithms (genetic algorithm (GA), differential evolution (DE) and particle swarm optimization (PSO)) for the problem of the search the analytical functions and parameters of models based on the given experimental data, the sets of variables, basic functions and operations. We used three systems of ordinary differential equations with some undefined functions and parameters. The metaheuristic programming synthesis with templates is implemented in MATLAB system, on the processor Intel Core i5-8265U, 1.8 GHz with memory 8 GB.

The number of iterations (generations) and the population size for all nature-inspired algorithms and tests were selected as follows: population size: 50, the number of iterations: 100, the length of a sequence of operators (the length of chromosomes): 40. The set of basic
functions used to synthesize formulas is as follows: \( F_1 = \{+,-,*\} \). The set of terminal symbols in this case is \( T = \{x,y,C\} \), where \( C \in \{\alpha,\beta,\gamma,\delta,a,b,c\} \) is a element of the set of random constants. The following parameters of the nature-inspired algorithms are applied: in GA, the crossover probability is 0.7, the probability of mutation is 0.3; in DE, the crossover probability is 0.2, the differential weight is 0.8; in PSO, we have \( w = 0.72984, c1 = 1.4962, c2 = 1.4962 \). In the experiments described, we used the recommended parameter values of the algorithms of their authors in the cited literature to correctly compare the effectiveness. Separate optimization of the algorithm parameters was not performed.

Let us consider the search for some undefined functions and parameters for the Van der Pol oscillator equations for the two-dimensional case:

\[
\frac{dx}{dt} = y, \quad \frac{dy}{dt} = \alpha(1 - x^2)y - x,
\]

for varying degrees of template specialization. Suppose we do not know the parts \( T_i(x,y) \) of the second equation as is shown in the following cases:

\[
\frac{dy}{dt} = T_1(x,y)y - x, \quad \frac{dy}{dt} = T_2(x,y) - x, \quad \frac{dy}{dt} = T_3(x,y).
\]

There be given 50 input/output pairs \(((x,y), \frac{dy}{dt})\) and templates \( TM \) for the right-hand side of the second equation, where \( T_i(x,y) \) are unknown parts (see Table 1). Table 1 shows the characteristics of the metaheuristic programming using three different nature-inspired algorithms (GA, DE, PSO) based on templates, where \( t_A \) is the time (sec) taken by the algorithm \( A \) to find a solution or has fulfilled a required number of generations and \( p_A \) is the probability (frequency) of success, i.e., the probability that the algorithm \( A \) has detected (synthesized) an expression coinciding with a given precision \((\epsilon = 10^{-10})\) to the reference function. This is the ratio of the number of successful experiments, when the algorithm has found a correct expression, to the total number of experiments with the given parameters. For each template the average values obtained from the results of 10 experiments are given.

| \( i \) | TM | \( T \) | \( t_{DE} \) | \( p_{DE} \) | \( t_{GA} \) | \( p_{GA} \) | \( t_{PSO} \) | \( p_{PSO} \) |
|---|---|---|---|---|---|---|---|---|
| 1 | \( \frac{dy}{dt} = T_1(x,y)y - x \) | \( T_1 = \alpha(1 - x^2) \) | 85 | 1 | 113 | 1 | 107 | 1 |
| 2 | \( \frac{dy}{dt} = T_2(x,y) - x \) | \( T_2 = \alpha(1 - x^2)y \) | 486 | 1 | 686 | 0.75 | 1279 | 0.25 |
| 3 | \( \frac{dy}{dt} = T_3(x,y) \) | \( T_3 = \alpha(1 - x^2)y - x \) | 1911 | 0.25 | 1866 | 0.25 | 2102 | 0 |

Consider the search for an equation for another model, i.e. the Lotka - Volterra model of the interaction of two types of the "predator-prey":

\[
\frac{dx}{dt} = x(\gamma - \delta y), \quad \frac{dy}{dt} = y(\alpha - \beta x),
\]

for varying degrees of the template specialization. Assume we do not know the parts \( T_i(x,y) \) of the second equation as is shown in the following cases:

\[
\frac{dy}{dt} = T_1(x,y)y, \quad \frac{dy}{dt} = T_2(x,y). \]

There be given 100 input/output pairs \(((x,y), \frac{dy}{dt})\) and templates \( TM \) for the right-hand side of the second equation, where \( T_i(x,y) \) are unknown parts (see Table 2). Table 2 shows the characteristics of the metaheuristic programming using of three different nature-inspired algorithms (GA, DE, PSO) based on templates, where \( t_A \) is the time (sec) taken by the algorithm \( A \) to find a solution or has fulfilled a required number of generations and \( p_A \) is the probability (frequency) of success.
Table 2. Experimental results for the Lotka-Volterra model with the unknown functions $T_i$.

| $i$ | TM | $T$ | $t_{DE}$ | $p_{DE}$ | $t_{GA}$ | $p_{GA}$ | $t_{PSO}$ | $p_{PSO}$ |
|-----|-----|-----|---------|---------|---------|---------|---------|---------|
| 1   | $\frac{dy}{dt} = T_1(x,y)y$ | $T_1 = (\alpha - \beta x)$ | 2853    | 0.5     | 1748    | 0.5     | 3292    | 0.25    |
| 2   | $\frac{dy}{dt} = T_2(x,y)$ | $T_2 = y(\alpha - \beta x)$ | 4607    | 0       | 3772    | 0.25    | 4566    | 0       |

Consider the search for an equation for the third model: the Lorenz system of three ordinary differential equations as a simplified mathematical model for the atmospheric convection:

$$\frac{dx}{dt} = a(y-x), \quad \frac{dy}{dt} = bx - y - xz, \quad \frac{dz}{dt} = xy - cz,$$

for varying degrees of template specialization. Suppose we do not know the parts $T_i(x,y,z)$ of the third equation as is shown in the following cases: $\frac{dx}{dt} = T_1(x,y,z) - cz$, $\frac{dy}{dt} = xy + T_2(x,y,z)$, $\frac{dz}{dt} = T_3(x,y,z)$. There be given 1000 input/output pairs $((x,y,z), \frac{dz}{dt})$ and templates TM for the right-hand side of the third equation, where $T_i(x,y,z)$ are unknown parts (see Table 3). Table 3 shows the characteristics of the metaheuristic programming using of three different nature-inspired algorithms (GA, DE, PSO) based on templates, where $t_A$ is the time (sec) taken of the algorithm $A$ to find a solution or has fulfilled a required number of generations and $p_A$ is the probability (frequency) of success.

Table 3. Experimental results for the Lorenz system with the unknown functions $T_i$.

| $i$ | TM | $T$ | $t_{DE}$ | $p_{DE}$ | $t_{GA}$ | $p_{GA}$ | $t_{PSO}$ | $p_{PSO}$ |
|-----|-----|-----|---------|---------|---------|---------|---------|---------|
| 1   | $\frac{dz}{dt} = T_1(x,y,z) - cz$ | $T_1 = xy$ | 383     | 1       | 389     | 1       | 391     | 1       |
| 2   | $\frac{dz}{dt} = xy + T_2(x,y,z)$ | $T_2 = -cz$ | 1596    | 1       | 1547    | 1       | 1186    | 1       |
| 3   | $\frac{dz}{dt} = T_3(x,y,z)$ | $T_3 = xy - cz$ | 35258   | 0.25    | 27982   | 0.25    | 26920   | 0.25    |

From Table 1 - Table 3 it is seen that an increase in the degree of specialization of the template can significantly reduce the search time. It can be seen from the tables of experiments that the probability of success in DE is in most cases higher than in other algorithms. The algorithms can be arranged in descending order of the obtained average success rate as follows: $DE > GA > PSO$. These tables show that algorithms can not be arranged in decreasing order of the average execution time in all the experiments because DE has less time to find a solution (in most cases) than other algorithms in Table 1, GA has less time to find a solution than other algorithms in Table 2 and PSO has less time to find a solution (in most cases) than other algorithms in Table 3.

Conclusion
The application of a metaheuristic programming (MP) algorithm based on the three bio-inspired algorithms (genetic algorithm, differential evolution and particle swarm optimization) for the evolutionary (metaheuristic) synthesis of nonlinear models is considered. The approach proposed of the evolutionary synthesis uses the sequential operator structure of the chromosome, templates (with some undefined functions and parameters) of the models, sets of variables, basic functions
and the given sets of pairs of input and output data. The influence of the degree of specialization of the template on the characteristics of the search algorithm in the metaheuristic synthesis of nonlinear models defined by ordinary differential equations with undetermined functions and parameters is studied. Estimates of the complexity of this algorithm by the time of the search for a solution and the probability of finding a function (model) are obtained, and a significant reduction in the search time when using templates in the process of evolutionary synthesis is shown.

Acknowledgments
The work has been carried out under the budget project of ICMMG SB RAS N 0315-2019-0006.

References
[1] Koza J 1996 Genetic Programming II: Automatic Discovery of Reuseable Programs (Cambridge, MA: MIT Press)
[2] Langdon W B and Poli R 2002 Foundations of Genetic Programming (Berlin: Springer-Verlag)
[3] Du K L and Swamy M N S 2016 Search and Optimization by Metaheuristics. Techniques and Algorithms Inspired by Nature (Basel: Birkhäuser)
[4] Yang X S 2008 Nature-Inspired Metaheuristic Algorithms (Luniver Press)
[5] Monakhov O G 2013 Science and Education of the Bauman MSTU 3 269
[6] Monakhov O G and Monakhova E A 2017 Numerical Analysis and Applications 10 140
[7] Storn R and Price K 1997 J. Global Optimization 11 341
[8] Zaheer H, Pant M, Kumar S, Monakhov O, Monakhova E and Deep K 2017 International Journal of System Assurance Engineering and Management 8 2170
[9] Bonyadi M R and Michalewicz Z Evolutionary Computation 25 1