

Math-Net.Ru

Общероссийский математический портал

О. Г. Монахов, Э. А. Монахова, Параллельный алгоритм многовариантного эволюционного синтеза нелинейных моделей, *Сиб. журн. вычисл. матем.*, 2017, том 20, номер 2, 169–180

DOI: <https://doi.org/10.15372/SJNM20170205>

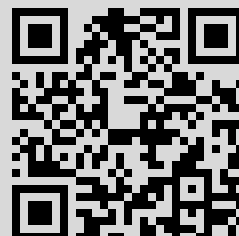
Использование Общероссийского математического портала Math-Net.Ru подразумевает, что вы прочитали и согласны с пользовательским соглашением

<http://www.mathnet.ru/rus/agreement>

Параметры загрузки:

IP: 178.69.47.199

23 февраля 2023 г., 14:39:34



УДК 519.7, 519.8

Параллельный алгоритм многовариантного эволюционного синтеза нелинейных моделей*

О.Г. Монахов, Э.А. Монахова

Институт вычислительной математики и математической геофизики Сибирского отделения Российской академии наук, просп. Акад. М.А. Лаврентьева, 6, Новосибирск, 630090

E-mails: monakhov@rav.sccc.ru (Монахов О.Г.), emilia@rav.sccc.ru (Монахова Э.А.)

Монахов О.Г., Монахова Э.А. Параллельный алгоритм многовариантного эволюционного синтеза нелинейных моделей // Сиб. журн. вычисл. математики / РАН. Сиб. отд-ние. — Новосибирск, 2017. — Т. 20, № 2. — С. 169–180.

Предложен параллельный алгоритм для решения проблемы построения нелинейных моделей (математических выражений, функций, алгоритмов, программ) на основе заданных экспериментальных данных, множества переменных, базовых функций и операций. Разработанный алгоритм многовариантного эволюционного синтеза нелинейных моделей имеет: линейное представление хромосомы, модульные операции при декодировании генотипа в фенотип для интерпретации хромосомы как последовательности команд, многовариантный метод для представления множества моделей (выражений) с помощью одной хромосомы. Проведено сравнение последовательной версии данного алгоритма со стандартным алгоритмом генетического программирования и алгоритмом декартового генетического программирования и показано его преимущество по сравнению с указанными алгоритмами как по времени поиска решения (более чем на порядок в большинстве случаев), так и по вероятности нахождения заданной функции (модели). Проведены эксперименты на параллельных суперкомпьютерных системах и получены оценки эффективности предложенного параллельного алгоритма, демонстрирующие линейные ускорение и масштабируемость.

DOI: 10.15372/SJNM20170205

Ключевые слова: *параллельный многовариантный эволюционный синтез, генетический алгоритм, генетическое программирование, декартово генетическое программирование, нелинейные модели.*

Monakhov O.G., Monakhova E.A. A parallel algorithm of the multivariant evolutionary synthesis of nonlinear models // Siberian J. Num. Math. / Sib. Branch of Russ. Acad. of Sci. — Novosibirsk, 2017. — Vol. 20, № 2. — P. 169–180.

A parallel algorithm for solving the problem of constructing of nonlinear models (mathematical expressions, functions, algorithms, programs) based on given experimental data, a set of variables, basic functions and operations is proposed. The proposed algorithm of the multivariant evolutionary synthesis of nonlinear models has a linear representation of the chromosome, the modular operations in decoding the genotype to the phenotype for interpreting a chromosome as a sequence of instructions, the multivariant method for presenting a multiplicity of models (expressions) using a single chromosome. A comparison of the sequential version of the algorithm with a standard algorithm of genetic programming and the algorithm of the Cartesian Genetic Programming offers advantage of the algorithm proposed both in the time of obtaining a solution (by about an order of magnitude in most cases), and in the probability of finding a given function (model). In the experiments on the parallel supercomputer systems, estimates of the efficiency of the proposed parallel algorithm have been obtained showing linear acceleration and scalability.

Keywords: *parallel multivariant evolutionary synthesis, genetic algorithm, genetic programming, Cartesian genetic programming, nonlinear models.*

*Работа выполнена при поддержке РФФИ (проект № 14-01-00031).

1. Введение и основные определения

Рассматривается решение проблемы построения нелинейных моделей, представленных в виде математических выражений, функций, формул, алгоритмов и программ, на основе заданных экспериментальных данных, множества переменных, базовых функций и операций. Задачей является поиск математического выражения f , наилучшим образом описывающего нелинейную вычислительную модель, заданную совокупностью входных X и выходных Y экспериментальных данных, т. е. требуется подобрать такую функцию $Y = f(X)$, которая отображает зависимость Y от X с минимальной погрешностью (иногда эту задачу называют символьной регрессией или идентификацией системы). Поиск осуществляется на основе заданного множества базовых функций, операций и переменных, с помощью которых автоматически создаются аналитические выражения (формулы), представляющие модель, и компьютерные программы для их вычисления.

Одним из подходов к решению данной задачи является генетическое программирование (ГП) [1–3], которое представляет собой одно из направлений в эволюционном моделировании [4, 5] и ориентировано, в основном, на решение задач автоматического синтеза программ на основе обучающих данных путем эволюционного поиска моделей, минимизирующих погрешность отображения. Хромосомы, или структуры, которые автоматически генерируются с помощью генетических операторов в ГП, являются (после интерпретации) выражениями и реализующими их компьютерными программами различной величины и сложности.

В данной работе представлен новый параллельный алгоритм многовариантного эволюционного синтеза (МВЭС) нелинейных моделей, имеющий более высокую эффективность эволюционного поиска по сравнению с известными ранее системами генетического программирования, и показана принципиальная работоспособность предлагаемого метода на простых примерах, когда множество базовых функций включает множество элементарных функций, необходимое для построения используемых тестовых функций.

В следующей пункте будут приведены основные этапы моделирования процесса эволюции в системах генетического программирования и варианты представления решений (хромосом) в двух наиболее известных из них. Далее описан новый алгоритм многовариантного эволюционного синтеза нелинейных моделей, и в конце статьи приведены экспериментальные оценки эффективности предложенного алгоритма, его параллельной версии и рассмотренных ранее алгоритмов.

2. Представление решений в системах генетического программирования

Рассмотрим основные этапы моделирования процесса эволюции в различных системах генетического программирования и покажем варианты представления решений (хромосом) в двух наиболее известных из них: стандартное генетическое программирование (ГП, genetic programming) [1] и декартово генетическое программирование (ДГП, cartesian genetic programming)[6].

Этапы моделирования процесса эволюции в системах генетического программирования заключаются в следующем.

1. Создание первоначальной популяции из случайно сгенерированных решений (хромосом). Отметим, что решение в хромосоме представлено, как правило, в закодированном виде — в виде генотипа.

2. Оценка популяции по фитнес-функции (fitness function, функция пригодности, функция приспособленности), которая показывает, насколько хорошо каждый индивид решает заданную проблему. При этом происходит декодирование генотипа в фенотип для интерпретации хромосомы как программы для вычисления фитнес-функции.
3. Создание популяции следующего поколения с помощью следующих эволюционных операторов:
 - 3.1. Выбор лучшего решения в популяции и копирование его в следующее поколение.
 - 3.2. Создание новых хромосом методом кроссовера.
 - 3.3. Создание новых хромосом методом мутации.
4. Повторение пунктов 2 и 3 пока решение по заданному критерию не будет найдено или не будет достигнуто максимальное число поколений.

Приведенные базовые этапы моделирования процесса эволюции не исключают существование различных стратегий организации популяции и применение других эволюционных операторов.

Поиск решения в процессе эволюции осуществляется на основе заданного множества базовых (элементарных) функций (function set, напр., $F_1 = \{+, -, /, *, \sin, \exp\}$) и заданного множества свободных проблемно-ориентированных переменных и констант (терминальное множество — terminal set, напр., $T_1 = \{x, y, a, 1, 2, 3.14\}$), из которых строится требуемое математическое выражение (модель, программа).

Генетическое программирование. При стандартном генетическом программировании [1] решение представляется в виде дерева вместо строки (вектора) чисел (целых, действительных, двоичных), используемой в генетическом алгоритме (ГА) [4, 7]. При этом функции из набора базовых (элементарных) функций становятся внутренними узлами дерева решения, а элементы терминального множества (переменные и константы) становятся листьями (концевыми вершинами) дерева (рисунок 1).

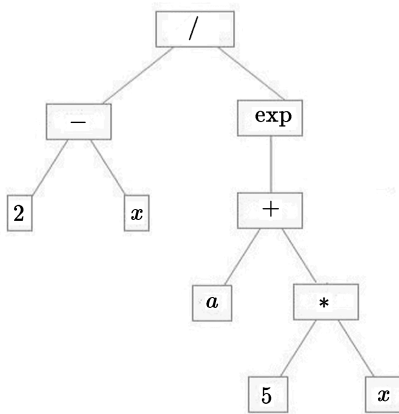


Рис. 1. Пример дерева решения при стандартном генетическом программировании

На рис. 1 показан пример дерева решения для выражения $f = (x - 2)/e^{5x+a}$ при множестве базовых функций $F_1 = \{+, -, /, *, \sin, \exp\}$ и терминальном множестве $T_1 = \{x, a, 2, 5\}$. Операция кроссовера в этом случае состоит в порождении двух новых особей путем обмена частями хромосом родителей (обмен случайно выбранными поддер-

вьями для древовидной структуры хромосомы). Операция мутации при этом состоит в изменении значения случайно выбранной вершины в представлении функции f на другую случайно выбранную величину из множества допустимых значений. Отметим, что при древовидном представлении решения в ходе процесса эволюции возникает эффект неоправданного роста выражений и появления функционально лишних частей (интронов), что ведет к снижению эффективности эволюционного поиска [2].

Декартово генетическое программирование. Декартово генетическое программирование (ДГП) [6] является формой генетического программирования, которая использует хромосому фиксированного размера, представленную последовательностью целых чисел (генотип), кодирующую двумерную функциональную сеть (фенотип). Хромосома представляет в закодированном виде многокаскадную сеть из узлов, каждый узел которой состоит из базовой функции и набора связей с узлами предыдущих каскадов — входных переменных функции в узле. Первый каскад представляет входные переменные из терминального множества, а последний каскад представляет выходные переменные всей сети. Пути, соединяющие входные и выходные узлы и проходящие через ряд функциональных узлов, определяют аналитические выражения, состоящие из функций и входных переменных, принадлежащих данным путям. На рис. 2 показан пример функциональной сети для получения выражений $f = (x + 2)/e^{ax-5}$ и $g = \sin(e^{a+5}) - 2x$. Хромосома, определяющая функциональную сеть, состоит из трех типов генов (векторов целых чисел): функциональные гены, описывающие функции в узлах; гены связи, описывающие соединения между узлами; выходные гены, описывающие выходные узлы. Одна функциональная сеть может задавать несколько аналитических выражений, соответствующих разным выходным узлам. Вычисление фитнес-функции происходит на всей сети для каждого набора входных данных. Полученные выходные значения сравниваются с требуемыми значениями и отбираются сети с минимальными рассогласованиями этих значений. В ДГП эволюционный процесс наиболее часто организуется с использованием эволюционной стратегии типа (1 + 4)-ES [6] путем генерации случайных хромосом, оценки их фитнес-функции и их мутации (как правило, с использованием алгоритма точечной мутации) и отбора кандидатов для создания следующего поколения.

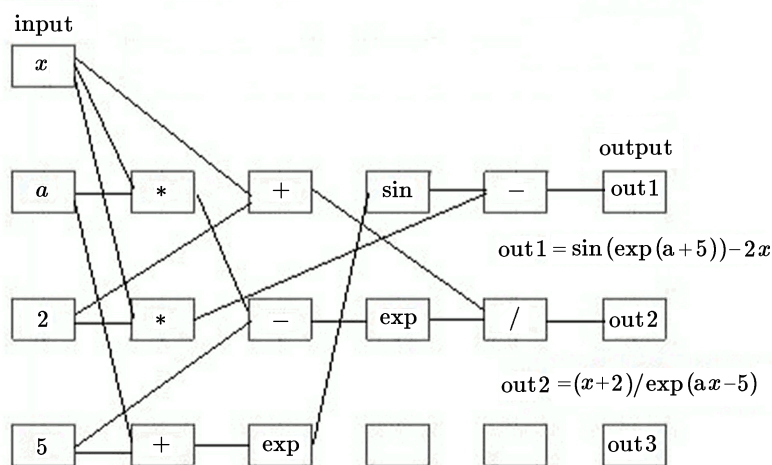


Рис. 2. Пример функциональной сети для представления решений в декартовом генетическом программировании

3. Алгоритм многовариантного эволюционного синтеза

Алгоритм многовариантного эволюционного синтеза (МВЭС) основан на эволюционных вычислениях и моделировании процесса естественного отбора в популяции особей, каждая из которых представляет точки в пространстве решений задачи оптимизации, а не единственное решение как в ГП и ДГП. Особи являются структурами данных — хромосомами — последовательностями целых чисел, которые представляют в закодированном виде математические выражения (формулы, программы). Каждая популяция является множеством хромосом, и каждая хромосома в данном алгоритме определяет множество выражений (формул), образующихся из нее после декодирования.

Основная идея алгоритма синтеза состоит в эволюционном преобразовании множества хромосом (формул) в процессе естественного отбора с целью выживания “сильнейшего”. В нашем случае этими особями являются выражения, имеющие наименьшее значение целевой функции. Алгоритм начинается с генерации начальной популяции. Все особи в этой популяции создаются случайно, затем отбираются наилучшие особи путем декодирования генотипа (хромосомы) в фенотип (выражение) и вычисления фитнес-функции. Для создания популяции следующего поколения (следующей итерации) новые особи формируются с помощью генетических операций селекции (отбора), мутации и кроссовера.

Примем, что целевая функция (фитнес-функция (FF), функция качества, функция пригодности, функция приспособленности) вычисляет сумму квадратов отклонений выходных данных выражения $Y'_i = f(X_i)$ от заданных эталонных значений Y_i для определенных наборов множества входных данных выражения X_i ($1 \leq i \leq N$): $FF = \sum_{i=1}^N (f(X_i) - Y_i)^2$, где N — число экспериментальных данных. Целью алгоритма является поиск минимума FF . На практике, если получалось несколько решений с одинаковым значением целевой функции, то выбиралось решение с меньшей оценкой структурной сложности, т. е. с меньшей общей длиной (суммой числа элементов) формул решения.

Для декодирования основной структуры данных — хромосомы — был предложен подход, основанный на представлении линейной структуры хромосомы как последовательности трехадресных команд (инструкций) и получении последовательной операторной структуры для вычисления выражений (формул), закодированных в хромосоме. Для этого последовательность целых чисел (хромосома) разбивается на группы из трех членов (триплеты (h_1, h_2, h_3)), а каждая такая группа интерпретируется как трехадресная инструкция следующим образом: $\langle \text{oper} \rangle \langle \text{adr1} \rangle \langle \text{adr2} \rangle$, где операция $\langle \text{oper} \rangle$ выполняется над операндами, расположенными в инструкциях с номерами $\langle \text{adr1} \rangle$ и $\langle \text{adr2} \rangle$, которые вычисляются по следующим формулам:

$$\begin{aligned} \text{oper} &= h_1 \bmod |F|, \\ \text{adr1} &= (h_2 \bmod (I - 1)) + 1, \\ \text{adr2} &= (h_3 \bmod (I - 1)) + 1, \\ \text{if } \text{oper} = 0 \text{ then } \text{adr1} &= (h_1 \bmod |T|), \end{aligned} \tag{1}$$

где $|F|$ — мощность множества базовых функций, oper — номер элемента в этом множестве, т. е. номер исполняемой функции в текущей инструкции, I — номер текущей инструкции, adr1 и adr2 — номера предыдущих инструкций, результаты исполнения которых используются как операнды в текущей инструкции, $|T|$ — мощность терминального множества. Если $\text{oper} = 0$, то инструкция интерпретируется как оператор загрузки и происходит загрузка терминального символа с номером adr1 .

Таким образом, результат декодирования хромосомы в выражение (функцию) заключается в представлении функции интерпретируемым кодом, каждая инструкция которого будет считаться отдельной функцией (включающей все предыдущие инструкции). Первой операцией всегда будет инструкция вызова переменной данной функции. Выполнение происходит сверху вниз, и аргументами инструкций могут выступать лишь записи, расположенные ранее в инструкциях с меньшими номерами, поэтому исполнение получается линейным. Генетическое решение в этом случае представляет сразу множество функций-выражений (последовательностей от первого до каждого текущего оператора), что позволяет, в отличие от стандартного ГП, оценить одновременно множество выражений, представленных операторной последовательностью, и сократить время поиска оптимального решения. При этом в качестве оценки данной хромосомы из множества полученных вариантов выбирается оценка выражения с минимальным значением целевой функции. Переменные и константы формулы f образуют множество терминальных символов T , а операции, используемые в формуле f , образуют множество нетерминальных символов F .

Пусть $f = (x - 2)/e^{ax+5}$ при $F = \{L, +, -, /, *, \exp, \sin\}$, $|F| = 7$ (L — операция загрузки (вызова) переменной или константы из множества T с номером, вычисленным по первому адресу), $T = \{x, a, 2, 5\}$, $|T| = 4$. Пример представления данного выражения f в виде последовательной операторной структуры с интерпретируемым кодом и в символьном виде показан в таблице 1, где I — номер инструкции, K — хромосома (238, 224, ..., 215), разбитая на триплеты, $M = \text{опер} : \text{adr1} : \text{adr2}$ — результат декодирования триплета с помощью модульных операций (1), CM — сама инструкция в мнемонической записи, E — получаемое выражение (формула) для данной инструкции в символьном виде, FC — значение выражения для каждой инструкции (при $x = 1$, $a = 2$). Приведем пример декодирования инструкции $I = 5$ по формулам (1): $K = 060 : 071 : 196$, номер операции $\text{опер} = 60 \bmod 7 = 4$ — это операция умножения $*$ в F (здесь первый элемент L имеет номер 0), первый операнд $\text{adr1} = (71 \bmod (5 - 1)) + 1 = 4$; для инструкции с номером $I_1 = 4$ имеем $E_4 = a$ и $F_4 = 2$, второй операнд $\text{adr2} = (196 \bmod (5 - 1)) + 1 = 1$; для инструкции с номером $I_2 = 1$ имеем $E_1 = x$ и $F_1 = 1$, отсюда $M_5 = 4 : 4 : 1$, $CM_5 = * 4, 1$, $E_5 = E_4 * E_1 = ax$, $F_5 = F_4 * F_1 = 2$.

Таблица 1. Пример представления выражения в виде хромосомы (K), последовательной операторной структуры (CM) и в символьном виде (E)

I	K	M	CM	E	FC
1	238 : 224 : 197	0 : 0 : -	$L x$	$E_1 = x$	$F_1 = 1$
2	112 : 010 : 155	0 : 2 : -	$L 2$	$E_2 = 2$	$F_2 = 2$
3	065 : 168 : 243	1 : 1 : 2	$- 1, 2$	$E_3 = x - 2$	$F_3 = -1$
4	028 : 121 : 147	0 : 1 : -	$L a$	$E_4 = a$	$F_4 = 2$
5	060 : 071 : 196	4 : 4 : 1	$* 4, 1$	$E_5 = ax$	$F_5 = 2$
6	091 : 135 : 181	0 : 3 : -	$L 5$	$E_6 = 5$	$F_6 = 5$
7	134 : 112 : 215	2 : 5 : 6	$+ 5, 6$	$E_7 = ax + 5$	$F_7 = 7$
8	068 : 181 : 189	5 : 7 : -	$\exp 7$	$E_8 = e^{ax+5}$	$F_8 = 1096.6$
9	045 : 090 : 215	3 : 3 : 8	$/ 3, 8$	$E_9 = (x - 2)/e^{ax+5}$	$F_9 = -0.00091$

Отметим, что для данной хромосомы K определяется множество выражений E , вычисляется для каждого выражения значение целевой функции и в качестве оценки данной хромосомы из множества полученных оценок выбирается оценка выражения с минимальным значением целевой функции. Таким образом, в отличие от алгоритмов ГП и

ДГП, которые кодируют одно решение в хромосоме, данный алгоритм кодирует несколько решений в хромосоме (несколько решений в хромосоме кодируется также в Multi Expression Programming (MEP) [8], но без использования хромосомы как простого вектора целых чисел и стандартного ГА для их эволюции, а организует эволюцию на множестве программ).

В процессе эволюции популяции хромосом (векторов целых чисел) в МВЭС используются операторы стандартного генетического алгоритма. Зададим начальную популяцию, состоящую из M произвольных случайных векторов целых чисел заданной длины l , каждое из которых находится в интервале $[0, b]$ (обычно $b = 255$, l кратно 3). Затем будем применять генетические операторы мутации, кроссовера и селекции к данной популяции.

Мутация — это замещение с вероятностью $p_m \in [0, 1]$ каждого элемента вектора, независимо от остальных, случайным числом от 1 до b .

Кроссовер. Из популяции, состоящей из M векторов, M раз выбираются произвольные пары, и с вероятностью $p_c \in [0, 1]$ к данной паре применяется операция кроссовера: данная пара векторов делится в произвольной позиции на две части и производится их обмен (отметим, что с вероятностью $1 - p_c$ кроссовер к данной паре не применяется).

Селекция. Вычисляются целевые функции новых векторов, полученных посредством мутации и кроссовера, при этом происходит декодирование векторов (хромосом) в выражения и программы описанным выше процессом. Если они имеют целевые функции меньше, чем некоторые векторы популяции, тогда “наихудшие” векторы (с большим значением целевой функции) в популяции замещаются новыми “наилучшими” (с меньшим значением целевой функции).

Для поиска оптимума заданной целевой функции FF итерационный процесс вычислений в генетическом алгоритме организован следующим образом.

Первая итерация: порождение начальной популяции. Все особи популяции создаются с помощью случайного оператора, порождая значения для каждого элемента вектора в начальной популяции, равномерно распределенные от 1 до b , с последующим вычислением целевой функции.

Промежуточная итерация: шаг от текущей к следующей популяции. Основным шагом алгоритма состоит в создании нового поколения особей на основе текущей популяции, используя операции мутации, кроссовера и селекции. На каждой итерации происходит M (величина популяции) попыток выбора пар особей, к которым применяются операции кроссовера (с вероятностью p_c), мутации (с вероятностью p_m) и селекции.

Последняя итерация: критерий остановки. Алгоритм завершается, когда найден вектор с $FF = 0$ или после заданного числа итераций (генераций) t .

Отметим, что разработанный алгоритм многовариантного эволюционного синтеза нелинейных моделей объединяет преимущества генетического алгоритма (простые генетические операции над целочисленными векторами, постоянный размер которых предотвращает эффект неоправданного роста выражений) и генетического программирования (автоматический синтез математических выражений и реализующих их компьютерных программ различной величины и сложности). Разработанный алгоритм совместно использует линейное представление хромосомы, простые модульные операции (1) при декодировании генотипа в фенотип для интерпретации хромосомы как последовательности команд, многовариантный метод для представления множества моделей (выражений) с помощью одной хромосомы.

4. Экспериментальные результаты

Для сравнения эффективности алгоритма многовариантного эволюционного синтеза с другими алгоритмами генетического программирования для задачи поиска аналитического описания модели (символьной регрессии) на основе заданных экспериментальных данных, множества переменных, базовых функций и операций были выбраны шесть тестовых функций:

$$\text{Test1: } x^4 + x^3 + x^2 + x.$$

$$\text{Test2: } \sin(x^2 + x^4).$$

$$\text{Test3: } \sin(\exp(\sin(\exp(\sin(x)))))$$

$$\text{Test4: } \sin(x^3) + e^x.$$

$$\text{Test5: } \sin(2x) + 1/x^2 - x^3.$$

$$\text{Test6: } (x + 2)/e^{(ax-5)}.$$

В экспериментах использовались значения каждой функции в 20-ти случайных точках в диапазоне (0,2), множество базовых функций для Test1–Test6: $F_1 = \{+, -, *, /, \sin, \exp\}$, терминальные символы для Test1–Test5: $T_1 = \{x\}$, а для Test6: $T_2 = \{x, a, 2, 5\}$.

Использовались следующие параметры алгоритмов: вероятность кроссовера — 0.80; вероятность мутации — 0.2; размер популяции — 200; максимальное число генераций — 500. Длина хромосом равна 42, для ГП задается начальная глубина дерева выражения, равная 4, для ДГП задаются размеры функциональной сети $42 = 2 * 21$ и максимальное число генераций эволюционной стратегии (1+4)-ES для функций Test1–Test6 равно 100000. Для каждого алгоритма и каждой тестовой функции программа исполнялась 100 раз и результаты усреднялись.

Один из основных показателей, используемых для измерения эффективности эволюционных алгоритмов — это вероятность (частота) успеха — вероятность того, что алгоритм обнаружил (синтезировал) выражение, в точности совпадающее с эталонной функцией. Это отношение числа успешных опытов, когда алгоритм обнаружил правильное выражение, к общему числу опытов с использованием заданных параметров. В табл. 2 представлены вероятности (частоты) успеха для различных алгоритмов. Из таблицы видно, что вероятность успеха у МВЭС почти во всех случаях выше, чем у других алгоритмов (кроме одного случая — Test4, где вероятность успеха у МВЭС равна ДГП). Это объясняется способностью МВЭС представлять несколько выражений в одной хромосоме, что приводит к более высокому шансу обнаружить решение.

Таблица 2. Частота успешного поиска тестовых функций

Тестовые функции	ГП	ДГП	МВЭС
Test1: $x^4 + x^3 + x^2 + x$	0.01	0.69	0.8
Test2: $\sin(x^2 + x^4)$	0.006	0.42	0.45
Test3: $\sin(\exp(\sin(\exp(\sin(x)))))$	0.26	0.5	0.75
Test4: $\sin(x^3) + e^x$	0.048	0.67	0.67
Test5: $\sin(2x) + 1/x^2 - x^3$	0	0.14	0.22
Test6: $(x + 2)/e^{(ax-5)}$	0	0	0.45

Второй показатель, используемый для оценки эффективности эволюционных алгоритмов в данной работе — это среднее время поиска тестовых функций, т. е. среднее время исполнения алгоритма, до того, как алгоритм обнаружил (синтезировал) заданное выражение либо выполнил требуемое число поколений. В табл. 3 представлено сред-

нее время поиска тестовых функций для различных алгоритмов. По степени уменьшения времени решения алгоритмы можно упорядочить в следующей последовательности: ГП > ДГП > МВЭС. Из данной таблицы видно, что МВЭС имеет меньшее время поиска решения (более чем на порядок в большинстве случаев, за исключением Test6, где время решения МВЭС меньше в 4.7 раза). Это можно объяснить: 1) простыми генетическими операциями и структурами у МВЭС и 2) прямым исполнением инструкций во время декодирования хромосом “на лету”, без получения всего выражения отдельно в виде текстовой строки и ее интерпретации, как в других алгоритмах.

Таблица 3. Среднее время поиска тестовых функций

Тестовые функции	ГП	ДГП	МВЭС
Test1: $x^4 + x^3 + x^2 + x$	305	170.6	4.54
Test2: $\sin(x^2 + x^4)$	307	337.6	10.8
Test3: $\sin(\exp(\sin(\exp(\sin(x)))))$	249	496	6.6
Test4: $\sin(x^3) + e^x$	292	208.4	10.5
Test5: $\sin(2x) + 1/x^2 - x^3$	331.4	428	1.24
Test6: $(x + 2)/e^{(ax-5)}$	376	544.5	80.9

Параллельная версия алгоритма многовариантного эволюционного синтеза использует метод распараллеливания по данным на основе схемы “мастер-рабочие”. В параллельной версии данного алгоритма целая популяция делится на подпопуляции, каждая из которых назначается процессом-мастером процессу-рабочему на соответствующий процессор (ядро) параллельной системы. Эволюционные процессы исполняются независимо процессами-рабочими и после их завершения результаты пересылаются процессу-мастеру, который выполняет сбор и статистическую обработку результатов.

Параллельный алгоритм многовариантного эволюционного синтеза был реализован на ресурсах Сибирского суперкомпьютерного центра (ССКЦ) ИВМиМГ СО РАН и Информационно-вычислительного центра (ИВЦ) Новосибирского национального исследовательского государственного университета (НГУ). Эксперименты выполнялись на кластерной суперЭВМ ССКЦ (каждый вычислительный модуль которой состоит из двух четырехядерных процессоров Intel Xeon E5540, 2.53 ГГц) и на кластерной суперЭВМ ИВЦ (каждый вычислительный модуль которой состоит из двух 12-ти ядерных процессоров Intel Xeon E5-2680v3, 2.5 ГГц). Общее число используемых ядер изменялось от 8 до 312. Программа написана на языке Си с использованием библиотек параллельного программирования OpenMP и MPI. На каждом вычислительном модуле использовались один поток MPI (для организации коммуникаций) и все доступные ядра для выполнения вычислений (по одному потоку OpenMP на каждое ядро). В конце итераций среди всех потоков собираются статистика и лучшее решение, такой подход минимизирует взаимодействия и позволяет получить линейные масштабирование и ускорение для параллельного алгоритма.

В табл. 4 и табл. 5 приведены результаты экспериментов для параллельной реализации алгоритма многовариантного эволюционного синтеза при поиске тестовой функции Test6 для числа вычислительных модулей BM , числа ядер Kn , размера популяции Pop и времени исполнения T (в секундах). Полученное число успешных опытов — Suc , когда алгоритм обнаружил правильное выражение, и Sp — полученное ускорение по отношению к одному вычислительному модулю. Из таблиц видно, что при линейно возрастающем объеме вычислений (при линейном росте величины популяции) время исполнения остается постоянным (с отклонениями не более 1.2 %), что свидетельствует об отличном

масштабировании, высокой эффективности распараллеливания и линейном ускорении параллельного алгоритма многовариантного эволюционного синтеза. Приведем два примера выражений, полученных в результате эксперимента для тестовой функции Test6: $f_1 = (x + 2) * \exp(5 - ax)$, $f_2 = (2 + x) / \exp(x - (x - ax + 5))$.

Таблица 4. Результаты экспериментов для параллельной реализации алгоритма МВЭС на суперЭВМ ССКЦ ИВМиМГ СО РАН

$BM(Kn)$	Pop	T	Suc	Sp
1 (8)	1600	2419	400	1
2 (16)	2 * 1600	2400	822	2
3 (24)	3 * 1600	2410	1236	3
4 (32)	4 * 1600	2402	1755	4
6 (48)	6 * 1600	2408	2581	6
8 (64)	8 * 1600	2403	3535	8
10 (80)	10 * 1600	2441	4163	10
12 (96)	12 * 1600	2417	5100	12
14 (112)	14 * 1600	2409	5979	14
16 (128)	16 * 1600	2414	6900	16
18 (144)	18 * 1600	2407	7568	18
20 (160)	20 * 1600	2446	8488	20

Таблица 5. Результаты экспериментов для параллельной реализации алгоритма МВЭС на суперЭВМ ИВЦ НГУ

$BM(Kn)$	Pop	T	Suc	Sp
1 (24)	4800	1796	1269	1
2 (48)	2 * 4800	1818	2525	2
3 (72)	3 * 4800	1817	3725	3
4 (96)	4 * 4800	1780	4987	4
5 (120)	5 * 4800	1785	6321	5
6 (144)	6 * 4800	1790	7558	6
7 (168)	7 * 4800	1796	8618	7
8 (192)	8 * 4800	1816	10022	8
9 (216)	9 * 4800	1795	11364	9
10 (240)	10 * 4800	1799	12616	10
11 (264)	11 * 4800	1814	13779	11
12 (288)	12 * 4800	1811	14915	12
13 (312)	13 * 4800	1789	16363	13

5. Заключение

Рассмотрен подход к решению проблемы построения нелинейных моделей (математических выражений, функций, алгоритмов, программ) на основе заданных экспериментальных данных, множества переменных, базовых функций и операций. Разработан алгоритм многовариантного эволюционного синтеза таких моделей, объединяющий преимущества генетического алгоритма и генетического программирования. Разработанный алгоритм совместно использует линейное представление хромосомы, модульные опера-

ции при декодировании генотипа в фенотип для интерпретации хромосомы как последовательности команд, многовариантный метод для представления множества моделей (выражений) с помощью одной хромосомы. Предложенный алгоритм многовариантного эволюционного синтеза был реализован, и было проведено его сравнение со стандартным алгоритмом генетического программирования, использующим древовидное представление хромосомы, и алгоритмом декартового генетического программирования, использующим представление программы в виде конечного графа. Проведенные эксперименты показали преимущество предложенного подхода по сравнению с указанными алгоритмами как по времени поиска решения (более чем на порядок в большинстве случаев), так и по вероятности нахождения заданной функции (модели). Проведены эксперименты на параллельных суперкомпьютерных системах и получены оценки эффективности предложенного параллельного алгоритма, демонстрирующие линейные ускорение и масштабируемость.

Литература

1. **Koza J.** Genetic Programming II: Automatic Discovery of Reusable Programs. — Cambridge: MIT Press, 1996.
2. **Langdon W.B., Poli R.** Foundations of Genetic Programming. — Springer-Verlag, 2002.
3. **Poli R., Langdon W.B., and McPhee N.F.** A Field Guide to Genetic Programming. — San Francisco, California, USA: Lulu.com, 2008.
4. **Емельянов В.В., Курейчик В.В., Курейчик В.М.** Теория и практика эволюционного моделирования. — М.: ФИЗМАТЛИТ, 2003.
5. **Монахов О.Г.** Исследование влияния степени специализации шаблонов на пространство поиска при эволюционном синтезе моделей // Прикладная дискретная математика. — 2012. — № 3. — С. 84–94.
6. **Miller J.F.** Cartesian Genetic Programming. — Springer, 2011.
7. **Монахова Э.А., Монахов О.Г.** Поиск рекордных циркулянтных графов с использованием параллельного генетического алгоритма // Дискретный анализ и исследование операций. — 2015. — Т. 22, № 6. — С. 29–39.
8. **Oltean M.** Multi Expression Programming. — Romania: Babes-Bolyai Univ, 2006. — (Technical Report).

*Поступила в редакцию 19 сентября 2016 г.,
в окончательном варианте 20 октября 2016 г.*

Литература в транслитерации

1. **Koza J.** Genetic Programming II: Automatic Discovery of Reusable Programs. — Cambridge: MIT Press, 1996.
2. **Langdon W.B., Poli R.** Foundations of Genetic Programming. — Springer-Verlag, 2002.
3. **Poli R., Langdon W.B., and McPhee N.F.** A Field Guide to Genetic Programming. — San Francisco, California, USA: Lulu.com, 2008.
4. **Emel'yanov V.V., Kureychik V.V., Kureychik V.M.** Teoriya i praktika evolyucionnogo modelirovaniya. — М.: FIZMATLIT, 2003.
5. **Monahov O.G.** Issledovanie vliyaniya stepeni specializacii shablonov na prostranstvo poiska pri evolyucionnom sinteze modeley // Prikladnaya diskretnaya matematika. — 2012. — № 3. — С. 84–94.

6. **Miller J.F.** Cartesian Genetic Programming. — Springer, 2011.
7. **Monahova E.A., Monahov O.G.** Poisk rekordnyh cirkulyantnyh grafov s ispol'zovaniem parallel'nogo geneticheskogo algoritma // Diskretnyy analiz i issledovanie operaciy. — 2015. — Т. 22, № 6. — С. 29–39.
8. **Oltean M.** Multi Expression Programming. — Romania: Babes-Bolyai Univ, 2006. — (Technical Report).