Yeroshenko Ilya Nikolaevich
Taganrog Institute of Technology – Federal State-Owned Educational Establishment of Higher Vocational Education "Southern Federal University".
E-mail: lutaries@rambler.ru.
44, Nekrasovsky, Taganrog, 347928, Russia.
Phone: +78634644031.

УДК 658.512.2.011.5

М.В. Лисяк

ПРИМЕНЕНИЕ ГЕНЕТИЧЕСКОГО ПРОГРАММИРОВАНИЯ ДЛЯ РЕШЕНИЯ ЗАДАЧИ СИМВОЛЬНОЙ РЕГРЕССИИ*

Рассматривается приложение метода генетического программирования для решения задачи символьной регрессии. Входными данными задачи являются множество переменных и констант (аргументов функции), множество элементарных функций, набор фитнес-тестов (таблица эталонных значений искомой функции при различных значениях аргументов). Результатом решения задачи является символьная запись математического выражения, наилучшим образом описывающего заданную фитнес-тестами функциональную зависимость. Приводится описание задачи и метода её решения, способ кодирования решений и оценки их качества, используемый алгоритм генетического поиска, модификации стандартных генетических операторов, параметры алгоритма поиска. Приводится пример работы программы, реализующей описанный метод.

Генетическое программирование; генетический алгоритм; символьная регрессия; оценки фитнеса; репродукция; кроссинговер; многократная селекция.

M.V. Lisyak

APPLICATION OF GENETIC PROGRAMMING FOR DECISION OF SIMBOLIC REGRESSION PROBLEM

There is considered the decision of symbolic regression problem by means of genetic programming method. The problem input data include the set of independent variables and constants (arguments of function), the set of elementary functions, list of fitness-cases (table containing reference values of unknown function at different values of the arguments). Result of the problem decision is mathematical expression in symbolic form, which fits the functional dependence, described by fitness-cases, in the best way. There are described the problem and the method of decision, the way of alternative decisions representation and quality measuring, algorithm of genetic search, modifications of standard genetic operators, parameters of search algorithm. The result of running a program, which realizes described method, is considered.

Genetic programming; genetic algorithm; symbolic regression; fitness measures; reproduction; crossover; over-selection.

Введение. Символьной регрессией называется процесс поиска математического выражения, наилучшим образом описывающего функциональную зависимость, заданную некоторым набором числовых данных. Поиск осуществляется путем перебора произвольных композиций элементарных функций из заданного множества.

Исходными данными задачи являются множество независимых переменных, множество элементарных математических функций, которые могут входить в состав искомой функции, а также таблица значений искомой функции при различ-

* Работа выполнена при поддержке: РФФИ (гранты № 09-07-00318, № 10-07-00538), г/б № 2.1.2.1652.

ных значениях независимых переменных. В процессе регрессии происходит обработка символьной информации, алгоритм поиска анализирует, насколько точно каждое полученное математическое выражение описывает функциональную зависимость, и выбирает оптимальное с точки зрения точности символьное выражение. В отличие от линейной, квадратичной и других видов регрессии, где требуется определить числовые коэффициенты для имеющейся математической модели, символьная регрессия подразумевает как определение коэффициентов, так и построение оптимальной модели.

Эффективным способом решения задачи символьной регрессии является метод генетического программирования, предложенный Дж. Козой (*John Koza*). Помимо символьной регрессии с помощью генетического программирования могут быть решены такие задачи как оптимальное управление (*optimal control*), планирование (*planning*), индукция последовательностей (*sequence induction*), автоматическое программирование (*automatic programming*), игровые стратегии (*game playing strategies*), эмпирические исследования и прогнозирование (*empirical discovery and forecasting*), символьное интегрирование и дифференцирование (*symbolic integration and differentiation*), определение обратных функций (*inverse problems*), нахождение математических тождеств (*discovering mathematical identities*), классификация и создание деревьев решений (*classification and decision tree induction*), автоматическое программирование клеточных автоматов (*automatic programming of cellular automata*), эволюция эмерджентного поведения (*evolution of emergent behavior*).

Основной предпосылкой к развитию генетического программирования явилась идея Дж. Козы о том, что множество задач из различных предметных областей могут быть сформулированы в виде задачи индуктивного поиска оптимальной компьютерной программы на множестве всех допустимых компьютерных программ. Под компьютерной программой понимается любая древовидная структура, преобразующая входные данные в желаемый выход. В зависимости от предметной области решаемой задачи программе может соответствовать формула, план, управляющая стратегия, вычислительная процедура, модель, дерево решений, игровая стратегия, план действий робота, функция переходов состояний, последовательность операций или композиция функций [1].

Переход от понятий конкретной предметной области к терминологии компьютерных программ позволяет использовать при решении задач такие полезные свойства программ как гибкость и универсальность. Гибкость подразумевает способность к изменению формы, размера и структурной сложности альтернативных решений, а гибкость – возможность использовать единые средства индукции программ для решения задач различного рода.

В общем случае задачи, требующие обработки символьной информации, не могут быть решены с помощью генетических алгоритмов. Тем не менее, изменение стандартного способа представления решений (изменение структуры хромосом) и модификация генетических операторов делают генетический поиск на множестве символьных данных возможным [2].

Генетическое программирование является развитием генетических алгоритмов в сторону повышения сложности адаптирующихся структур. Если в генетических алгоритмах адаптирующейся структурой является линейная хромосома фиксированного или изменяющегося размера, то в генетическом программировании адаптирующиеся структуры представляют собой деревья различных форм и размеров [3].

Исходные данные. В задаче символьной регрессии каждое альтернативное решение представляет собой математическое выражение в символьном представлении, аналитически описывающее функцию из множества допустимых решений.

Множеству допустимых решений принадлежат все синтаксически корректные функции, полученные композицией элементов двух множеств – множества операторов (*function set*) и множества терминалов (*terminal set*). Множество терминалов состоит из независимых переменных и констант предметной области, множество операторов содержит элементарные математические функции [4].

Множество терминалов и множество операторов являются исходными данными для символьной регрессии методом генетического программирования, и должны быть сформированы так, чтобы удовлетворять следующим условиям:

1) условие замыкания (*closure*): каждая элементарная функция из множества операторов должна принимать в качестве аргументов любые значения и типы данных, которые могут быть возвращены любой функцией из множества функций или могут принадлежать множеству терминалов;
2) условие достаточности (*sufficiency*): множество терминалов и множество операторов должны быть достаточно полными для того, чтобы с помощью функции, полученной композицией их элементов, было возможно описать решение поставленной задачи с требуемой точностью.

Наряду с множествами терминалов и операторов в качестве исходных данных используются наборы значений независимых переменных и соответствующие им эталонные значения искомой функции, называемые фитнес-тестами (*fitness-cases*). На основании фитнес-тестов происходит оценка качества решений.

Представление решений. Математические выражения кодируются с помощью деревьев, называемых программами. Узлам дерева соответствуют элементы множества операторов, а листьям – независимые переменные и константы из множества терминалов (рис. 1).

В отличие от линейных хромосом, которые используются в генетических алгоритмах, программы являются активными структурами. Каждая программа выполняется на всех наборах значений переменных, на выходе получая значения математического выражения. Вычисление значения выражения производится обходом дерева в ширину (левое поддерево, затем правое поддерево, затем узел). Найденное значение сравнивается с эталонным, и таким образом, путем оценки точности соответствия найденного математического выражения искомой функции, определяется качество программы.

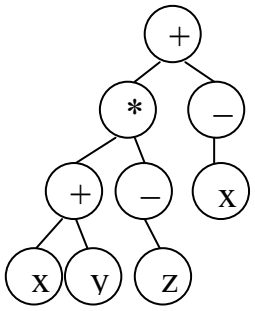

Рис. 1. Представление решений с помощью деревьев

Целевая функция. Традиционно для задачи символьной регрессии целевая функция описывает сумму абсолютных погрешностей значений математического выражения на всех фитнес-тестах. Целевая функция определяется по методу первичного фитнеса (*raw fitness*), т.е. в исходных понятиях задачи:

$$r(i) = \sum_{j=1}^{N} |S(i,j) - S_{et}(j)|, \; r(i) \to min, \qquad (1)$$

где r(i) – значение первичного фитнеса i-го математического выражения из популяции математический выражений; S(i,j) – значение i-го математического выражения на j-ом фитнес-тесте; S_{et}(j) – эталонное значение искомого выражения на j-ом фитнес-тесте; N – количество фитнес-тестов.

Для задачи символьной регрессии известно оптимальное значение первичного фитнеса, равное 0. Это позволяет использовать для вычисления целевой функ-

ции не только первичный фитнес, но также скорректированный и нормализованный виды фитнеса.

Скорректированный фитнесс (adjusted fitness) является дополнительным способом оценки качества решений и вычисляется следующим образом:

$$a(i) = \frac{1}{1+r(i)},\ a \to max,\ \text{a(i)} \in (0, 1], \tag{2}$$

где a(i) – значение скорректированного фитнеса i-го математического выражения; r(i) – значение первичного фитнеса i-го математического выражения.

Нормализованный фитнес (normalized fitness) также является дополнительным и вычисляется на основе скорректированного:

$$n(i) = \frac{a(i)}{\sum_{k=1}^{M} a(k)}, n \to max, n(i) \in (0,1), \tag{3}$$

$$\sum_{i=1}^{M} n(i) = 1, \tag{4}$$

где *n(i)* – значение нормализованного фитнеса *i*-го математического выражения; *a(i)* – значение скорректированного фитнеса *i*-го математического выражения; *M* – размер популяции математических выражений.

Особенностью нормализованного фитнеса является то, что при таком способе вычисления целевой функции, сумма значений целевой функции особей всей популяции равна 1 [4].

Пример вычисления значений целевой функции математических выражений на основе различных видов фитнеса приведен в табл. 1.

Таблица 1

Фитнес-тесты				**Популяция математических выражений**				**Лучшее значение целевой функции по популяции**	**Среднее значение целевой функции по популяции**
j	*x*	*y*	$S_{et}(j)$	*S(1, j)*	*S(2, j)*	*S(3, j)*	*S(4, j)*		
1	0	0	1	0	-4	1	6		
2	1	1	2	2	10	2	5		
3	2	2	3	5	13	3	-10		
Первичный фитнес				3	23	0	21	0	11.75
Скорректированный фитнес				0,25	0,0416	1	0,0454	1	0,3342
Нормализованный фитнес				0,1869	0,0311	0,7478	0,0339	0,7478	0.25

Преимущество, которое предоставляют скорректированный и нормализованный фитнес, заключается в увеличении различий между близкими значениями целевой функции по мере приближения их к оптимуму. Это позволяет повысить вероятность отбора лучших решений за счет увеличения разницы между значениями целевой функции близких по качеству решений, но замедляет работу генетического поиска за счет временных затрат на пересчет фитнеса. Скорректированный и нормализованный фитнес целесообразно использовать, только если селекция особей осуществляется пропорционально фитнесу.

Алгоритм генетического поиска. Генетическое программирование предполагает осуществление генетического поиска путем выполнения следующих этапов:

1. Создание начальной популяции случайных программ, полученных композицией элементов множества терминалов и множества операторов в соответствии с синтаксическими правилами.

2. Итеративное исполнение следующих шагов до тех пор, пока не выполнится заданное число операций, либо не будет достигнут известный глобальный оптимум:
   2.1. Выполнение каждой программы популяции на наборе фитнес-тестов, вычисление значений целевой функции.
   2.2. Создание новой популяции компьютерных программ путем применения генетических операторов. Особи для выполнения генетических операторов выбираются из популяции в соответствии со стратегией селекции.
3. Определение результата генетического поиска.

Для того чтобы выполнение генетического поиска по методу генетического программирования стало возможным, необходимо предусмотреть специальные процедуры формирования начальной популяции программ, модифицировать стандартные генетические операторы, адаптировав их для работы с древовидными структурами.

Создание начальной популяции. Формирование начальной популяции решений может осуществляться тремя способами:

1) полная генерация (*full*): для любого дерева популяции длина пути от корня до каждого листа равна заданной длине, при этом все деревья имеют одинаковые форму и высоту;
2) культивирование (*grow*): для каждого дерева популяции длина пути от корня до любого его листа не превышает заданной максимальной длины, при этом все деревья популяции приобретают разные форму и высоту;
3) объединенный метод (*ramped half-and-half*): является комбинацией предыдущих методов, при котором, деревья разной высоты генерируются с одинаковой частотой.

Наилучшее качество решений исходной популяции позволяет получить метод *ramped half-and-half*, общим для всех способов генерации является то, что деревья создаются в соответствии с синтаксическими правилами, порождаются только выполнимые математические выражения [5].

Модификация генетических операторов. Основными операторами, используемыми в генетическом программировании, являются репродукция и кроссинговер, аналогичные операторам простого генетического алгоритма с поправкой на синтаксические правила рекомбинации деревьев [6].

Оператор репродукции работает с одной родительской программой и получает одну программу-потомка путем выполнения двух шагов:

1) выбор программы из текущей популяции на основе значения её целевой функции в соответствие со стратегией селекции;
2) копирование выбранной программы в следующую популяцию без внесения изменений.

Оператор кроссинговера вносит изменения в популяцию путем создания новых программ, состоящих из частей родительских программ. Кроссинговер использует две родительские программы и производит двух потомков по следующей схеме:

1) выбор двух родительских программ из текущей популяции на основе значений целевой функции в соответствие со стратегией селекции;
2) случайный выбор точки кроссинговера для каждого родителя;
3) получение программ-потомков с помощью рекомбинации фрагментов родительских программ, определяемых точками кроссинговера;
4) копирование программ-потомков в новую популяцию, если их размер не превышает допустимый, в противном случае вместо слишком длинного потомка в новую популяцию копируется один из родителей.

Точки кроссинговера выбираются случайным образом среди узлов деревьев и определяют фрагменты родительских программ, которые подвергаются рекомбинации. Каждый фрагмент представляет собой поддерево, корнем которого является точка кроссинговера, а ветви и узлы лежат ниже этой точки. В частном случае фрагмент может состоять из одного терминала. Первый потомок получается отделением фрагмента от первого родителя и вставкой вместо него в родительское дерево фрагмента второго родителя. Второй потомок получается аналогичным способом (рис. 2). Таким образом, новые математические выражения составляются в соответствие с синтаксическими правилами, и описанный оператор кроссинговера не порождает некорректных решений.

В отличие от генетических алгоритмов, где для обоих родителей в операторе кроссинговера выбирается одна точка кроссинговера, метод генетического программирования предполагает выбор двух различных точек кроссинговера. Такой подход позволяет получать потомков различных форм и размеров. Модифицированный кроссинговер вносит в популяцию большее разнообразие по сравнению со стандартным.

Селекция родительских программ для выполнения генетических операторов производится пропорционально фитнесу (методом вращения колеса рулетки), на основе заданной шкалы, используются также турнирная и элитная селекции. Допускается многократная селекция одной и той же программы в качестве родителя при выполнении операторов репродукции и кроссинговера *(over-selection)*. Многократная селекция не приводит к предварительной сходимости алгоритма, поскольку оператор кроссинговера порождает большое разнообразие новых решений, препятствуя попаданию в локальный оптимум.

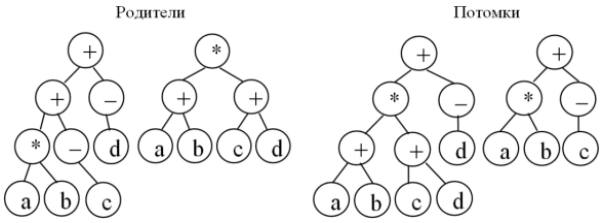

Рис. 2. Пример выполнения оператора кроссинговера над математическими выражениями

Простейший поисковый алгоритм на основе генетического программирования использует только операторы репродукции и кроссинговера. Более сложные алгоритмы генетического поиска могут содержать дополнительные модифицированные операторы мутации, перестановки, редактирования, децимации и инкапсуляции.

Параметры алгоритма поиска. Генетическое программирование использует основные и дополнительные числовые параметры, управляющие работой алгоритма поиска.

К основным числовым параметрам относятся размер популяции и максимальное количество поколений, определяющее максимальное время работы алгоритма.

К дополнительным числовым параметрам относятся:

1) вероятность кроссинговера, определяющая долю особей в каждой популяции, к которым применяется оператор кроссинговера;
2) вероятность репродукции, определяющая долю особей в каждой популяции, к которым применяется оператор репродукции;

3) вероятность выбора в качестве точки кроссинговера функционального узла дерева;
4) максимальный допустимый размер программы, полученной с помощью кроссинговера, который определяется как наибольшая допустимая высота дерева;
5) максимальный допустимый размер программы в начальной популяции.

Пример решения задачи символьной регрессии.

Исходные данные. Множество терминалов содержит две переменные: *{x, y}*. Множество операторов содержит бинарные математические операции сложения (+), умножения (*), безопасного деления (/) и унарную операцию минуса (-). Безопасное деление может принимать 0 в качестве аргумента, возвращая при этом достаточно большое целое число. Использование унарного минуса в качестве альтернативы бинарному вычитанию, не обладающему коммутативностью, повышает разнообразие решений, тем самым уменьшая вероятность попадания алгоритма в локальный оптимум.

Фитнес-тесты представлены в табл. 2.

Таблица 2

№	*x*	*y*	*f(x,y)*
1	0	3	0
2	1	4	7
3	2	5	20
4	3	6	39
5	4	7	64
6	5	8	95
7	6	9	132

Параметры алгоритма генетического поиска. Размер популяции 30, максимальное количество популяций 50, вероятность применения оператора кроссинговера 0.9, вероятность применения оператора репродукции 0.1, вероятность выбора в качестве точки кроссинговера функционального узла дерева 0.5, максимальная допустимая высота дерева исходной популяции 5, максимальна допустимая высота дерева, полученного с помощью кроссинговера 6.

Результаты поиска. Оптимальное математическое выражение, которое было найдено в 18-ом поколении, имеет вид *(x*x)+(((y*x)+(x*x)+x)*, что соответствует функции $f(x, y) = 2x^2+xy+x$. Значение целевой функции выражения 0, найденное решение является глобальным оптимумом. График изменения значений целевой функции в ходе работы алгоритма представлен на рис. 3.

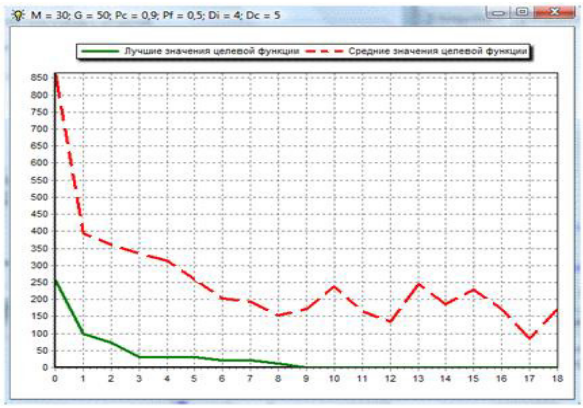

Рис. 3. График изменения значений целевой функции

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *John Koza*. Genetically Breeding Populations of Computer Programs to Solve Problems of Artificial Intelligence. Proceedings of the Second International Conference on Tools for AI. Herndon, Virginia, November 6-9, 1990. Los Alamitos, CA: IEEE Computer Society Press. – P. 819-827.
2. *John Koza*. Integrating Symbolic Processing into Genetic Algorithms. Workshop on Integrating Symbolic and Neural Processes at AAAI-90 in Boston. July 29, 1990.
3. *John Koza*. Genetic Programming: On the Programming of Computers by Means of Natural Selection. Cambridge, MA: The MIT Press, 1992. – 840 p.
4. *Zelinka Ivan*. Symbolic regression – an overview. http://www.mafy.lut.fi/EcmiNL/ older/ecmi35/node70.html.
5. *Ricardo Poli, William B. Langdon, Nicolas F. McPhee*. A field guide to Genetic programming. http://www.gp-field-guide.org.uk. 2008. – 250 p.
6. *Курейчик В.М., Родзин С.И.* Эволюционные алгоритмы: генетическое программирование // Известия академии наук. Теория и системы управления. – 2002. – № 1.

Лисяк Мария Владимировна
Технологический институт федерального государственного автономного образовательного учреждения высшего профессионального образования «Южный федеральный университет» в г. Таганроге.
E-mail: maria-lisyak@yandex.ru.
347928, г. Таганрог, пер. Некрасовский, 44.
Тел.: 88634360524.

Lisyak MariaVladimirovna
Taganrog Institute of Technology – Federal State-Owned Educational Establishment of Higer Vocational Education “Southern Federal University”.
E-mail: maria-lisyak@yandex.ru.
44, Nekrasovskiy, Taganrog, 347928, Russia.
Phone: +78634360524.

УДК 681.3.001.63

О.Б. Лебедев

ПЛАНИРОВАНИЕ СБИС НА ОСНОВЕ МЕТОДА МУРАВЬИНОЙ КОЛОНИИ*

Предлагаются новые технологии, принципы и механизмы решения задачи планирования, использующие математические методы, в которых заложены принципы природных механизмов принятия решений. Для компактного представления решения задачи планирования используется модифицированная польская запись. Это позволило создать пространство решений, в рамках которого организован поисковый процесс, базирующийся на моделировании адаптивного поведения муравьиной колонии. По сравнению с существующими алгоритмами достигнуто улучшение результатов.

Планирование СБИС; муравьиная колония; оптимизация.

O.B. Lebedev

PLANNING VLSI ON THE BASIS OF THE ANT COLONY METHOD

New technologies, principles and mechanisms of the planning VLSI decision using mathematical methods in which principles of natural mechanisms of decision-making are put in pawn are offered. For compact representation of the slicing floorplan a modification Polish expression

* Работа выполнена при поддержке: РФФИ (грант № 09-01-00509) г/б № 2.1.2.1652.