17. Prikladnye nechetkie sistemy [Applied fuzzy systems]: Translation from Japanese K. Asai, D. Vatada, S. Ivai i dr., Under ed. T. Terano, K. Asai, M. Sugeno. Moscow: Mir, 1993, 386 p.
18. *Kureychik V.M.* Osobennosti postroeniya sistem podderzhki prinyatiya resheniy [Features of decision making support system design], *Izvestiya YuFU. Tekhnicheskie nauki* [Izvestiya SFedU. Engineering Sciences], 2012, No. 7 (132), pp. 92-98.
19. Iskusstvennye immunnye sistemy i ikh primenenie [Artificial immune systems and their applications], Under ed. Dasgupty D., Translation from English, Under ed. Romanyukhi A.A. Moscow: Fizmatlit, 2006, 344 p.
20. *Kofman A.* Vvedenie v teoriyu nechetkikh mnozhestv [Introduction to the theory of fuzzy sets], Translation from English. Moscow: Radio i svyaz', 1982, 432 p.
21. *Zade L.A.* Fuzzy sets, *Information and Control*, 1965, Vol. 8, pp. 338.
22. *Kureychik V.M., Lebedev B.K., Lebedev O.B.* Poiskovaya adaptatsiya: teoriya i praktika [Search adaptation: theory and practice]. Moscow: Fizmatlit, 2006, 272 p.

Статью рекомендовал к опубликованию к.т.н. О.Г. Солопова.

Чернышев Юрий Олегович – Донской государственный технический университет; e-mail: myvnn@list.ru; 344000, г. Ростов-на-Дону, пл. Гагарина, 1; тел.: 88632738582; кафедра автоматизации производственных процессов; профессор.

Венцов Николай Николаевич – e-mail: vencov@list.ru; кафедра информационных технологий; доцент.

Панасенко Павел Александрович – Филиал военной академии связи (г. Краснодар); e-mail: we_panasenko_777@mail.ru; 350035, г. Краснодар, ул. Красина, 4; адъюнкт.

Chernyshev Yury Olegovich – Don State Technical University; e-mail: myvnn@list.ru; 1, Gagarin sq., Rostov-on-Don, 344000, Russia; phone: 88632738582; the department of automation of productions; professor.

Ventsov Nikolay Nikolaevich – e-mail: vencov@list.ru; the department information technologies; associate professor.

Panasenko Pavel Alexandrovich – Branch of the Military Academy of Telecommunications (Krasnodar); e-mail: we_panasenko_777@mail.ru; 4, Krasina, Krasnodar, 350035, Russia; adjunct.

УДК 681.325

Б.К. Лебедев, В.Б. Лебедев, О.Б. Лебедев

РЕШЕНИЕ ЗАДАЧИ СИМВОЛЬНОЙ РЕГРЕССИИ МЕТОДАМИ ГЕНЕТИЧЕСКОГО ПОИСКА*

Рассматриваются новые принципы решения задачи множественной нелинейной символьной регрессии на основе идей генетического программирования. Решение представляется в виде трех хромосом. Предлагаются способы представления деревьев с произвольной локальной степенью вершин в виде линейной записи. Разработаны структуры и принципы кодирования и декодирования хромосом, несущих информацию о структуре дерева и имеющих гомологичные структуры. Структуру бинарного дерева можно задать, используя на базе алфавита $\boldsymbol{A} = \{\boldsymbol{O}, \bullet\}$ *польское выражение. Определены основные свойства польского выражения, выполнение которых необходимо, чтобы ему соответствовало бинарное дерево. Предложен линейный алгоритм восстановления дерева по польскому выражению. Рассмотрены структура и принципы кодирования и декодирования хромосомы, для представления польского выражения. Разработана структура и принципы формирования линейной записи для иерархического дерева без ограничений на локальную степень внутренних вершин. На основе анализа определены свойства таких записей. Структура и*

* Работа выполнена при финансовой поддержке программы развития научного потенциала высшей школы РНП.2.1.2.1652 и грантов РФФИ № 12-01-00100, № 10-07-00055.

принципы кодирования хромосомы для представления древовидной записи разработаны с учетом вышеперечисленных свойств. Основными генетическими операторами являются кроссинговер и мутация. У описанной выше структуры хромосом гены, расположенные в одних и тех же локусах, являются гомологичными. Мутация заключается в принятии геном случайного значения из заданного диапазона значений для гена в данном локусе. Реализация кроссинговера осуществляется путем обмена гомологичной пары генов. Разметка терминального и функционального множества вершин дерева задается двумя дополнительными хромосомами. Отличительной особенностью способов представления деревьев в виде линейной записи исключают возможность потери элементов терминального множества, но при этом модель может быть произвольной суперпозицией функций из некоторого набора. Многохромосомные представления решений позволили создать иерархические структуры генетических операторов, что дает возможность организовать целенаправленный поиск и расширяет возможности и спектр решаемых задач символьной регрессии. При больших размерностях временные показатели разработанного алгоритма превосходят показатели сравниваемых алгоритмов при лучших значениях целевой функции. При больших размерностях временные показатели разработанного алгоритма превосходят показатели сравниваемых алгоритмов при лучших значениях целевой функции. Программа решения задачи множественной нелинейной символьной регрессии на основе идей генетического программирования была реализована на языке С++ для IBM/PC. Экспериментальная временная сложность алгоритма на одной итерации при фиксированных значениях управляющих параметров составляет O(nlgn), а временная сложность существующих алгоритмов – $O(n^2)$, где n – мощность терминального множества.

Регрессионный анализ; множественная нелинейная символьная регрессия; метод наименьших квадратов генетическое программирование; польское выражение; иерархическое дерево; структура и принципы кодирования хромосомы; генетические операторы.

B.K. Lebedev, V.B. Lebedev, O.B. Lebedev

THE SOLUTION OF THE SYMBOLIC REGRESSION PROBLEM BY GENETIC SEARCH METHODS

The paper discusses new approaches to solving the problem of multiple nonlinear symbolic regression based on the ideas of genetic programming. The solution presented in the form of three chromosomes. Suggests ways to represent trees with arbitrary local degree of the vertices in the form of linear recording. Developed the structure and principles of encoding and decoding of chromosomes, which carry information about the tree structure and having a homologous structure. The structure of a binary tree can be set using Polish expression on the basis of the alphabet A = {O •}. Defined the basic properties of Polish expressions, which are required to be consistent with a binary tree. We propose a linear algorithm to reconstruct a tree in Polish expression. The structure and principles of encoding and decoding a chromosome to represent the Polish expression. The structure and principles of linear recording for the hierarchical tree without constraints on the local degree of internal vertices. Based on the analysis of defined properties such records. The structure and encoding of a chromosome to represent a tree record developed taking into account the above properties. The main genetic operators are crossover and mutation. In the above-described structure of chromosomes genes located in the same loci are homologous. Mutation is the adoption of the genome random values from a specified range of values for the gene in this locus. Implementation of the crossover is performed by exchanging homologous pairs of genes. The layout of the terminal and functional set of nodes in the tree is determined by two additional chromosomes. A distinctive feature of the ways to represent trees in a linear recording exclude the possibility of the loss of the elements of the terminal set, but the model can be arbitrary superposition of functions from some set. Multiple chromosome presenting solutions has allowed us to create a hierarchical structure of genetic operators that allows you to organize targeted searches and expands the possibilities and range of tasks symbolic regression. For large dimensions the temporary performance of the developed algorithm are higher than those of the compared algorithms with the best values of the objective function. For large dimensions the temporary performance of the developed algorithm are higher than those of the compared algorithms with the best values of the objective function. Program solving the problem of multiple nonlinear symbolic regression based on the ideas of genetic programming

was implemented in C++ for IBM/PC. Experimental time complexity of the algorithm for one iteration with fixed values of control parameters is O(nlgn), and time complexity of the existing algorithms is O(n2), where n is a power terminal set.

Regression analysis; multiple nonlinear symbolic regression; method of least squares genetic programming; Polish expression; the hierarchical tree; the structure and coding of the chromosomes; the genetic operators.

Введение. Знания, добываемые методами Data mining, принято представлять в виде *закономерностей (паттернов)* [1]. В качестве таких выступают:

- ассоциативные правила;
- деревья решений;
- кластеры;
- математические функции.

Основу методов Data Mining составляют всевозможные методы классификации, моделирования и прогнозирования [1, 2], Задачи, решаемые методами Data Mining, принято разделять на описательные (англ. *descriptive*) и предсказательные (англ. *predictive*).

В описательных задачах самое главное – это дать наглядное описание имеющихся скрытых закономерностей, в то время как в предсказательных задачах на первом плане стоит вопрос о предсказании для тех случаев, для которых данных ещё нет.

К описательным задачам относятся:

- поиск ассоциативных правил или паттернов (образцов);
- группировка объектов, кластерный анализ;
- построение регрессионной модели.

К предсказательным задачам относятся:

- классификация объектов (для заранее заданных классов);
- регрессионный анализ, анализ временны́х рядов.

В большинстве численных методов идентификации для аппроксимации экспериментальных (статистических) данных используются регрессионные модели [3, 4]. Регрессия – это оценка функциональной зависимости условного среднего значения результативного признака Y от факторных признаков $X= (x_1 ,x_2 ,...,x_n)$, т.е. регрессия – это некоторая усредненная количественная зависимость между выходными и входными переменными $Y=F(X)$. В регрессионном анализе задача регрессии решается путем выбора функциональной формы и последующим нахождением ее численных коэффициентов (любым подходящим методом). Например, линейная – $(y = a_0 + a_1 x)$, квадратичная - $y = a_0 + a_1 x + a_2 x^2$, полиномиальная регрессия и др. [5]. Очевидно, что качество аппроксимации при данном подходе напрямую зависит от выбора конкретной параметрической модели.

Регрессия бывает двух видов: парная (линейная и нелинейная) и множественная (линейная и нелинейная). Разница между ними в виде уравнения и количестве независимых переменных. Логично, что парная регрессия – это когда одна зависимая переменная и одна независимая, в множественной – независимых переменных несколько. В природе имеет место исключительно множественная регрессия, так как нельзя ограничить внешнее влияние на какое-то явление строго одним фактором. Множественный регрессионный анализ может применяться как в исследовательских целях, так и для решения прикладных задач. Обычно множественная регрессия применяется для изучения возможности предсказания некоторого результата по ряду предварительно измеренных характеристик. Также помимо предсказания и определения степени его точности множественная регрессия позволяет определить и то, какие показатели, или независимые переменные, наиболее существенны и важны для предсказания, а какие переменные можно просто исключить из анализа [6–8].

Различают *линейную* и *нелинейную* регрессию. Если регрессионная модель не является линейной комбинацией функций от параметров, то говорят о нелинейной регрессии. При этом модель может быть произвольной суперпозицией функций из некоторого набора. Нелинейными моделями являются, экспоненциальные, тригонометрические и другие (например, радиальные базисные функции или персептрон Розенблатта), полагающие зависимость между параметрами и зависимой переменной нелинейной [6–9].

Регрессия тесно связана с классификацией. Термин *алгоритм* в классификации мог бы стать синонимом термина *модель* в регрессии, если бы алгоритм не оперировал с дискретным множеством ответов-классов, а модель – с непрерывно-определенной свободной переменной.

Критерием качества приближения (целевой функцией) обычно является *среднеквадратичная ошибка:* сумма квадратов разности значений модели и зависимой переменной для всех значений независимой переменной в качестве аргумента.

Задача символьной регрессии заключается в нахождении математического выражения в символьной форме, аппроксимирующего зависимость между конечным набором значений независимых переменных и соответствующими значениями зависимых переменных. Таким образом, символьная регрессия дает нам не только вычислительную процедуру, но и формулу (символьное математическое выражение), которую можно было бы подвергнуть содержательному анализу, упростить, а затем и уточнить. Однако на современном этапе методы символьной регрессии разработаны не достаточно хорошо. В последние годы интенсивно разрабатывается научное направление с названием «Природные вычисления» (Natural Computing), объединяющее математические методы, в которых заложены принципы природных механизмов принятия решений Генетическое программирование (ГП) – один из самых многообещающих подходов в данном направлении [12].

Одной из часто используемых моделей при решении многих задач является дерево, в частности бинарное дерево [10, 11]. В начале девяностых Koza J.R. из Стенфордского университета была разработана область эволюционных вычислений под названием генетическое программирование (ГП) [13, 14]. Его основной идеей была идея использования эволюционных алгоритмов для создания компьютерных программ. Для представления программ был использован язык программирования LISP (LISP S-expression), в котором программы могут легко рассматриваться как структуры дерева. Поэтому, вместо обычного использования двоичных последовательностей для отображения решения, в генетическом программировании в качестве хромосом использовались деревья. Вершины дерева являются элементами одного из двух множеств. Множество всех возможных внутренних вершин дерева называется функциональным множеством *F*. Множество всех возможных внешних вершин дерева называется терминальным множеством *T*. Элементы функционального множества обычно являются рабочими блоками программы (процедурами, функциями), элементы терминального множества – входными данными (переменными и константами). Внутренние вершины дерева (функциональное множество), обычно соответствуют следующим типам функций: арифметические операции - (+, -, *, \, %, и т.д.); математические функции - (синус, косинус, тангенс, логарифм и т.д.); булевские функции – (и, или, не, и т.д.); условные операторы – (если ... тогда ... иначе); операторы циклов – (до тех пор ... пока); любая другая функция из предметной области задачи. Терминалы – листья дерева, соответствуют либо переменной данной области задачи, либо постоянной. Например, выражение $x^2 + y$ может быть представлено деревом, показанным на рис. 1.

С одной стороны, представление решений в виде деревьев значительно расширяет сферы приложения идей генетического поиска. С другой стороны, применение стандартных механизмов генетического поиска приводит к возникновению нелегальных решений, хромосомы не гомологичны и могут иметь различную длину, что усложняет генетические операторы. Это является побудительной причиной разработки новых механизмов генетического поиска, у которых отсутствуют вышеперечисленные недостатки.

Рассматриваются новые принципы решения задачи множественной нелинейной символьной регрессии на основе идей генетического программирования. Предлагаются способы представления деревьев с произвольной локальной степенью вершин в виде линейной записи. Разработана структура и принципы кодирования и декодирования хромосом, несущих информацию о дереве и имеющих гомологичные структуры. Разработаны модифицированные генетические операторы, при выполнении которых не возникают хромосомы с нелегальными структурами.

Постановка задачи. Символьная регрессия заключается в построении математического выражения F, задаваемого примерами пар (x_1, y_1), (x_2, y_2),…, (x_n, y_n), где x_i, и y_i – входные и выходные записи. Обозначим как y_i^* значение выходной записи, получаемой с помощью выражения F. Для оценки математического выражения F введем критерий

$$D=\Sigma|y_i - y_i^*|^2.$$

Для решения задачи символьной регрессии с помощью генетического программирования необходимо выполнить следующие подготовительные шаги.

На первом этапе регрессионного анализа данные наблюдений или эксперимента представляют графически. Зависимость между переменными X и Y изображают точками на координатной плоскости (x, y) и соединяют их ломаной линией. Этот ломаный график называется *эмпирической линией регрессии Y по X*. По виду эмпирической линии регрессии делают предположение о виде (форме) зависимости переменной Y от X. В простейшем случае предполагают линейную зависимость.

На втором этапе по *эмпирической линии регрессии* определяют множество термов, из которых будет строиться решение. В задаче символьной регрессии терминальное множество содержит набор переменных x_i, $i = \overline{1,N}$, где N – размерность поставленной задачи) и набор констант $const_j$, $j = \overline{1,\ K}$, т.е. $T=\{X \cup C\}$.

На третьем этапе пользователь метода должен определить множество функций, которые будут использованы для построения решений. Пользователь должен априори предполагать некоторую комбинацию функций, которые могли бы содержаться в решении задачи. Функциональное множество может содержать: арифметические операции, математические функции, булевы операции, специальные предопределенные функции.

На четвертом этапе для оценки уравнения символьной регрессии задается целевая функция. Если вид функции φ в уравнении регрессии выбран, то для оценки неизвестных параметров используется *метод наименьших квадратов* (*МНК*). Согласно методу неизвестные параметры функции выбираются таким образом, чтобы сумма квадратов отклонений экспериментальных (эмпирических) значений y_i от их расчетных (теоретических) значений была минимальной

Основу генетического алгоритма составляют принципы кодирования и декодирования хромосом, генетические операторы и структура генетического поиска [12].

Способы представления исходной формулировки задачи в виде трех компонент в очень большой степени определяют усилия, необходимые для ее решения [15]..

Существует две схемы скрещивания в методе генетического программирования: стандартное скрещивание (standard crossover) и одноточечное скрещивание (one-point crossover). Стандартное скрещивание осуществляется следующим обра-

зом. Выбираются родительская пара. У каждого из родителей выбирается точка скрещивания (дуга в графе). Родители обмениваются генами (поддеревьями), находящимися ниже точки скрещивания. При одноточечном скрещивании у родительской пары выбирается общая точка скрещивания, далее скрещивание осуществляется по стандартной схеме. Общая точка выбирается в общей области деревьев родителей, получаемой наложением одного дерева на другое начиная с корня. Такие схемы скрещивания приводят к тому, что получаемые хромосомы имеют различную длину. При аппроксимации функциональных зависимостей может возникнуть ситуация, когда в результате применения оператора скрещивания или мутации будет потеряны одна или несколько переменных.

Гомологичными называют хромосомы, имеющие общее происхождение и поэтому морфологически и генетически сходные, т.е. при применении стандартных генетических операторов не образуются недопустимые хромосомы.

В негомологичных хромосомах не может быть двух генов с одинаковым значением. Для негомологичных хромосом применяют различные специальные генетические операторы, которые не создают недопустимых решений. В связи с этим трудоемкость алгоритмов реализующих генетические операторы для негомологичных хромосом больше, что увеличивает трудоемкость генетического алгоритма в целом. Это обстоятельство является побудительным мотивом исследований и разработок гомологичных структур хромосом.

Структуры хромосом для деревьев. Рассмотрим структуру выражения для описания бинарного дерева. Введём алфавит A = {O , •}.

Структуру бинарного дерева можно задать, используя на базе алфавита A польское выражение, где **O** соответствует листьям дерева, а • – соответствует внутренним вершинам дерева.[16-18]. Каждая внутренняя вершина подвергается бинарному ветвлению. Польские выражения для деревьев, представленных на рис. 2,а и 2,б, имеют вид **O O • O O • O • •** и **O O • O O • • O •**.

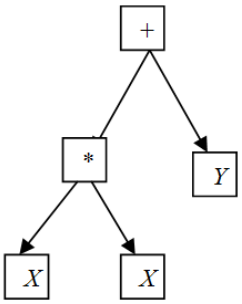
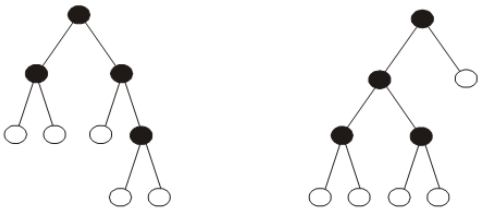


Рис. 1. Дерево выражения $x^2 + y$



а б

Рис. 2. Деревья выражений ***O O • O O • O • •; O O • O O • • O •***

Процесс восстановления дерева по польскому выражению достаточно прост. Последовательно, слева направо просматривается польское выражение и отыскиваются буквы типа •, соответствующие внутренним вершинам. Каждая внутренняя вершина объединяет два ближайших подграфа, сформированных на предыдущих шагах и расположенных в польской записи слева от знака •. Ниже, с помощью скобок показаны подграфы, образованные при просмотре польского выражения слева направо:

{(**O O** •) [(**O O** •) **O** •] •} и {[(**O O** •) (**O O** •) •] **O** •}.

Отметим основные свойства польского выражения, выполнение которых необходимо, чтобы ему соответствовало бинарное дерево.

Обозначим через n_o – число элементов польского выражения типа **O**, а через $n_\bullet$ – число элементов типа •. Для дерева всегда выполняется равенство $n_o = n_\bullet + 1$.

Если в польском выражении провести справа от знака • сечение, то слева от сечения число знаков **O** больше числа знаков • по крайней мере на единицу: $n_o - n_\bullet \geq 1$. Первый знак • в польском выражении (при просмотре слева направо) может появиться только после двух знаков **O**.

Пронумеруем позиции между знаками **O** как показано ниже.

OO $\underline{1}$ **O** $\underline{2}$ **O** $\underline{3}$ **O** $\underline{4}$ … **O** $\underline{n_o - 1}$

Максимальное число знаков •, которое может появиться в i-й позиции, равно номеру позиции i. Напомним, что общее число $n_\bullet = n_o - 1$.

Если польское выражение обладает вышеперечисленными свойствами, то ему соответствует бинарное дерево.

Рассмотрим структуру и принципы кодирования и декодирования хромосомы, для представления польского выражения [3].

Хромосома H имеет вид: $H=\{g_i \mid i=1,2,\ldots, n_\bullet\}$.

Будем использовать строку с n_o знаками O в качестве опорного множества для построения польского выражения при декодировании хромосомы.

Число генов в хромосоме равно $n_\bullet$, т.е. числу знаков '•'. Значение $Z(g_i)$ гена g_i колеблется в пределах от i до $n_\bullet$, т.е. $i \leq Z(g_i) \leq n_\bullet$. Значение гена указывает номер позиции между знаками **O** опорного множества, в которую необходимо поместить знак •. Декодирование хромосом, т.е. построение польской записи, осуществляется следующим способом. Формируется базовое множество символов **O** мощностью $n_o = n_\bullet + 1$ и определяется $n_\bullet$ позиций, расположенных между символами **O**, для помещения в них символов •. Затем последовательно выбираются гены, определяются задаваемые ими номера позиций, в которые и помещаются знаки •.

Например: пусть для $n_\bullet = 4$ имеется хромосома H = < 4,2,2,4 >. Это значит, что число n_o=5, а число позиций – 4. Два знака • назначаются во 2-ю позицию, а два знака • – в 4-ю. Польское выражение, соответствующее хромосоме, имеет вид:

O O O • • O O • •

Дерево, соответствующее данному польскому выражению, имеет вид, представленный на рис. 3.

Рассмотрим теперь структуру и принципы формирования линейной записи для иерархического дерева без ограничений на локальную степень внутренних вершин. Запись представляет собой набор элементов $A=\{a_i \mid i=1,2,\ldots,l\}$, где l – число вершин дерева. Причем n элементов $a_i \in A_o$ соответствуют листьям дерева, а m элементов $a_i \in A_v$ соответствуют внутренним вершинам n-арного дерева разрезов. $n+m=l$, $A=A_o \cup A_v$.

Формирование дерева в соответствии с древовидной записью осуществляется на основе иерархического подхода при просмотре записи слева направо, начиная с первого элемента.

Будем говорить, что вершина x_i в соответствии с записью расположена слева от x_j, если в записи элемент a_i расположен левее a_j. Древовидная запись организована так, что каждая внутренняя вершина x_i с одной стороны является корнем некоторого поддерева для вершин расположенных слева от x_i, а с другой стороны может быть дочерней вершиной для некоторой внутренней вершины расположенной в соответствии с записью справа от x_i. Последнему элементу a_n списка A соответствует вершина x_l, являющаяся корнем всего дерева.

Значением элемента $a_i \in A_v$ соответствующего внутренней вершине x_i является число поддеревьев, корни которых являются дочерними вершинами вершины x_i. Описания поддеревьев расположены в линейной записи непосредственно слева от a_i. Если $a_i \in A_o$, то $a_i = 0$, т.е. x_i является листом дерева.

Рассмотрим запись 0 0 0 3 0 0 2 0 0 2 0 0 0 2 3,3. Выделим с помощью скобок иерархически вложенные друг в друга поддеревья, образованные при просмотре записи слева направо:

((0,0,0,3),(0,0,2),((0,0,2),0,(0,0,2),3),3). (1)

Соответствующее дерево представлено на рис. 4.

Запись дерева обладает следующими свойствами:

- число элементов a_i записи с нулевым значением равно числу листьев дерева;
- число элементов a_i записи с ненулевым значением колеблется от 1 до *n-1*;
- значение любого элемента $a_i \in A_v$, соответствующего внутренней вершине x_i, равно или больше двух ($a_i \geq 2$), так как при разбиении любая внутренняя вершина связана минимум с двумя дочерними вершинами;
- a_1 и a_2 всегда равны нулю ;
- между номером позиции *k* и значениями элементов, расположенных в позициях с 1-й по *k*-ю, существует зависимость

$$\sum_{i=1}^{k} a_i < k,$$

т.е. сумма значений элементов, расположенных в позициях с1-й по *k*-ю меньше числа позиций *k*; для записи $A=\{a_i \mid i=1,2,...,l\}$ существует зависимость

$$\sum_{i=1}^{k} a_i = l-1$$

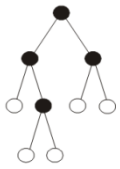


Рис. 3. Дерево выражения ООО••ОО • •

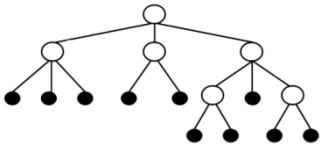


Рис. 4. Дерево выражения 0,0,0,3,0,0,2,0,0,2,0,0,0,2,3,3

Если запись удовлетворяет перечисленным свойствам, то она является древовидной и ей соответствует некоторое дерево.

Структура и принципы кодирования хромосомы для представления древовидной записи разработаны с учетом вышеперечисленных свойств.

Будем использовать множество элементов a_i с нулевым значением в качестве опорного множества для построения древовидной записи при декодировании хромосомы.

Расположим между этими элементами множество позиций, как показано ниже.

0 0 *1* **0** *2* **0** *3* … *n-2* **0** *n-1*

Если число элементов $a_i \in A_o$ равно *n*, то число позиций равно *n-1*. В каждую позицию может быть помещено несколько элементов с ненулевыми значениями, но при этом должны быть соблюдены вышеперечисленные свойства.

Хромосома имеет вид $H=\{g_i \mid i=1,2,..., n-1\}$.

Число генов в хромосоме равно *n-1*, где *n* – число листьев дерева. Ген g_i может принимать значение в интервале от *i* до *n-1*, кроме того, ген может быть либо помеченным, либо нет. Значение гена указывает номер позиции. Возможны два варианта действий. Если ген непомечен, то в позицию, соответствующую значе-

нию гена, к ненулевым элементам последним справа записывается элемент со значением, равным двум. Если же ген помечен, то в соответствующей позиции значение последнего справа ненулевого элемента увеличивается на единицу, а при отсутствии в позиции ненулевых элементов в нее записывается элемент со значением, равным двум. Метки хромосом задаются вектором $M=\{m_i \mid i=1,2,\ldots, n-1\}$. m_i=1, если ген g_i помечен; m_i=0, если ген g_i непомечен. Таким образом, древовидная запись кодируется парой хромосом. Для разметки листьев дерева используется третья хромосома $R=\{r_i \mid i=1,2,\ldots, n\}$. Пример: пусть задана пара хромосом

H = {2,2,4,6,9,9,9,9,9};
M = {0,1,0,0,0,0,1,0,1}.

Поскольку число генов в хромосоме H равно 9, число нулевых элементов в древовидной записи равно 10. После декодирования пары хромосом H и M полученная древовидная запись имеет вид **((0,0,0,3),(0,0,2),((0,0,2),0,(0,0,2),3),3)**, а соответствующее ей дерево представлено на рис. 4.

Предложенные структуры хромосом имеют линейную пространственную сложность.

Разметка множества вершин дерева кодируется двумя хромосомами. Хромосома *H1* несёт информацию о разметке листьев дерева. Хромосома *H2* содержит информацию о о разметке внутренних вершин дерева.

Если допускается повторение меток, то значением гена в *H1* или *H2* является метка, которой помечается соответствующая вершина дерева.

Для случая неповторяющихся меток разработана структура и принципы кодирования хромосом, обладающих свойством гомологичности.

Пусть n – число вершин. Хромосома *H1* имеет вид: $H1 = < g_1, g_2,\ldots, g_{n-1} >$. В результате декодирования строится вектор R, задающий разметку вершин.

Каждый ген g_i может принимать значение в интервале от 1 до (n+1- i). Например: для n = 8; $1 \le g_1 \le 8$; $1 \le g_2 \le 7$; $1 \le g_3 \le 6$;…; $1 \le g_7 \le 2$.

Декодирование хромосомы *H1* производится следующим образом. Пусть для n = 8 имеется хромосома $H1$ = <3,5,3,4,4,2,2> , и пусть имеется опорный вектор B^1 = <*a,b,c,d,e,f,g,h*>, число элементов которого равно n. Рассматриваем по порядку гены хромосомы и в соответствии с их значениями выбираем элементы в опорном векторе и записываем их в порядке выборки в вектор R.

Значение g_1 = 3. Выбираем в B^1 элемент b^1_j (j= g_1 =3, b^1_3 =*c*) и записываем его на первое место формируемого вектора R, т.е. $r_1 = b^1_3$ =*c*.

Удаляем элемент b^1_3 из B^1 и получаем вектор B^2 = <*a,b,d,e,f,g,h*>, содержащий 7 элементов. Следующим выбирается g_2 , g_2 = 5. Отыскиваем элемент b^2_5 вектора B^2 . b^2_5 = *f*. Следовательно, r_2 = *f*. Удаляем из B^2 элемент b^2_5 , получаем вектор B^3 = <*a,b,d,e,g,h*>. Далее:

g_3 = 3, b^3_3 = *d*, r_3 = *d*, B^4 = <*a,b,e,g,h*> ; g_4 = 4, b^4_4 = *g*, r_4 = *g*, B^5 = <*a,b,e,h*>;
g_5 = 4, b^5_4 = *h*, r_5 = *h*, B^6 = <*a,b,e*>; g_6 = 2, b^6_2 = *b*, r_6 = *b*, B^7 = <*a,e*>;
g_7 = 5, b^7_2 = *e*, r_7 = *e*, B^8 = <*a*>; и наконец, $r_8 = b^8_1$ = *a*.

В итоге получаем вектор = <*c,f,d,g,h,b,e,a*>, задающий разметку множества вершин.

Генетические операторы. Основными генетическими операторами являются кроссинговер и мутация. У описанной выше структуры хромосом гены, расположенные в одних и тех же локусах, являются гомологичными. Реализация кроссинговера осуществляется следующим образом [16, 17].

У предварительно выбранной родительской пары хромосом (на основе использования «принципа рулетки») последовательно просматриваются гомологичные пары генов, и с вероятностью P_k осуществляется обмен генами. При таком подходе степень модификации родительских хромосом определяется значением параметра P_K.

Суть оператора мутации – в произвольном изменении значений генов. Реализация оператора мутации осуществляется следующим образом. Последовательно просматриваются хромосомы, и с вероятностью P_{M1} они подвергаются мутации. Если хромосома мутирует, то последовательно просматриваются локусы хромосомы, и с вероятностью P_{M2} осуществляется мутация гена в рассматриваемом локусе. Мутация заключается в принятии геном случайного значения из заданного диапазона значений для гена в данном локусе.

Представление решения в виде набора хромосом дает возможность использовать оператор комбинирования набором хромосом в одном решении, что является приближением к естественной эволюции.

С другой стороны, представление решения набором из *n* хромосом дает возможность организации поиска решений в различных постановках, оставляя отдельные виды хромосом неизменными в процессе генетического поиска.

Очевидно, что фиксация отдельных хромосом в некоторой постановке приводит к сужению пространства поиска, и при этом возможна потеря оптимальных решений. В этой связи представляется целесообразным комбинирование отдельными постановками при поиске оптимального решения.

В общем случае возможны три подхода к комбинированию постановок: последовательный, параллельный и параллельно-последовательный.

При последовательном подходе на каждом *i*-м этапе осуществляется генетический поиск путем модификации хромосом, входящих в заданный для этого этапа набор NH_i модифицируемых типов хромосом. Это означает, что в полном объеме используется кроссинговер *K1*, заключающийся в комбинировании наборов хромосом, входящих в решение, а кроссинговер *K2* и мутация применяются только к тем хромосомам, которые входят в набор типов модифицируемых хромосом.

Приведем комбинацию, при которой в наборы входят по одному типу хромосом: $NH_1=\{H1\}$; $NH_2=\{H2\}$; $NH_3=\{H3\}$. В набор может входить от одного до четырех типов хромосом. На первом этапе в качестве исходной служит популяция $П_u$. На втором – популяция $П_1$, сформированная после отработки первого этапа, и т.д. Отметим возможность циклического повторения этапов.

При параллельном поиске производится распараллеливание процесса генетического поиска.

Вначале формируется исходная популяция. Для каждой параллельной ветви задается набор – NH_i типов хромосом, подвергающихся модификации. Затем на первом шаге, на базе этой исходной популяции, на каждой параллельной ветви осуществляется генетический поиск путем модификации хромосом, входящих в соответствующий набор типов модификаций хромосом.

После некоторого числа генераций (поколений) осуществляется случайное или направленное перемещение хромосом между любыми популяциями $П_1 – П_3$, сформированными на данный момент на параллельных ветвях. После этого на втором шаге модифицированные популяции $П_{10} – П_{30}$, вновь подвергаются обработке генетическими процедурами на параллельных ветвях. Число шагов является управляющим параметром.

При параллельно-последовательном подходе на каждой параллельной ветви реализуется последовательная комбинация постановок.

Как видно из алгоритмов, реализующих операторы кроссинговера и мутации, оценка их временной сложности имеет вид *O(n)*.

Заключение. Рассматриваются новые принципы решения задачи множественной нелинейной символьной регрессии на основе идей генетического программирования. Предлагаются способы представления деревьев с произвольной локальной степенью вершин в виде линейной записи. Разработаны структуры и принципы кодирования и декодирования хромосом, несущих информацию о дере-

ве и имеющих гомологичные структуры. Разработаны модифицированные генетические операторы при выполнении которых не возникают хромосомы с нелегальными структурами.

Рассмотренные в работе новые принципы и способы кодирования и декодирования хромосом для представления деревьев исключают некорректные решения, отличаются простотой и линейными оценками временной и пространственной сложности, что упрощает использование генетических операторов и позволяет использовать модификации генетических операторов, близких к естественным. Отличительной особенностью способов представления деревьев в виде линейной записи исключают возможность потери элементов терминального множества, но при этом модель может быть произвольной суперпозицией функций из некоторого набора.

Многохромосомные представления решений позволили создать иерархические структуры генетических операторов, что дает возможность организовать целенаправленный поиск и расширяет возможности и спектр решаемых задач символьной регрессии. При больших размерностях временные показатели разработанного алгоритма превосходят показатели сравниваемых алгоритмов при лучших значениях целевой функции.

При больших размерностях временные показатели разработанного алгоритма превосходят показатели сравниваемых алгоритмов при лучших значениях целевой функции.

Экспериментальная временная сложность алгоритма на одной итерации при фиксированных значениях управляющих параметров составляет $O(nlgn)$, а временная сложность существующих алгоритмов [3–15] – $O(n^2)$, где n – мощность терминального множества.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Han J., Kamber M.* Data mining: Concepts and Techniques. – Morgan Kaufmann Publishers. – 2001.
2. *Ian H. Witten, Eibe Frank and Mark A.* Hall Data Mining: Practical Machine Learning Tools and Techniques. – 3rd Edition. – Morgan Kaufmann, 2011.
3. *Радченко С.Г.* Методология регрессионного анализа: Монография. – К.: Корнийчук, 2011. – 376 с.
4. *Дрейпер Н., Смит Г.* Прикладной регрессионный анализ. – М.: Издательский дом «Вильямс», 2007. – 912 с.
5. *Стрижов В.В., Крымова Е.А.* Методы выбора регрессионных моделей. – М.: ВЦ РАН, 2010. – 60 с.
6. *Барсегян А.А., Куприянов М.С., Степаненко В.В., Холод И.И.* Методы и модели анализа данных: OLAP и Data Mining. – СПб.: БХВ-Петербург, 2004. – 336 с.
7. *Сергиенко В.И., Бондарева И.Б.* Математическая статистика в клинических исследованиях. – 2-е изд., перераб. и доп. – М.: ГЭОТАР-Медиа, 2006. – 304 с.
8. *Konar A.* Artificial intelligence and soft computing: behavioral and cognitive modeling of the human brain. – CRC Press LLC. – Boca Raton, Florida, 2000.
9. *Лаваньини И., Маньо Ф., Сералья Р., Тральди П.* Количественные методы в масс-спектрометрии. – М.: Техносфера, 2008. – 176 с.
10. *Лебедев В.Б., Лебедев О.Б.* Роевой интеллект на основе интеграции моделей адаптивного поведения муравьиной и пчелиной колоний // Известия ЮФУ. Технические науки. – 2013. – № 7 (144). – С. 41-47.
11. *Лебедев Б.К., Лебедев О.Б.* Моделирование адаптивного поведения муравьиной колонии при поиске решений, интерпретируемых деревьями // Известия ЮФУ. Технические науки. – 2012. – № 7 (132). – С. 27-35.
12. *Alpert C.J., Mehta D.P., and Sapatnekar S.S.* Handbook of Algorithms for Physical Design Automation. Boston, MA: Auerbach, 2009.

13. *Koza J.R.* Hierarchical genetic algorithms operating on populations of computer programs. In N.S. Sridharan (Ed.), Eleventh International Joint Conference on Artificial Intelligence. – Morgan Kaufmann. – 1989. – P. 768-774.
14. *Koza J.R.* Genetic Programming: On the Programming of Computers by means of Natural Selection. – Cambridge MA, MIT Press, 1992.
15. *Kureichik V.M., Lebedev B.K., Lebedev O.B.* A hybrid partitioning algorithm based on natural mechansms of decision making // Scientific and Technical Information Processing. – 2012. – № 39 (6). – P. 317-327.
16. *Lebedev B.K. and. Lebedev V.B.* Synthesis of Mathematical Expressions by Methods of Genetic Search // Proceedings of the International Scientific Conferences "Intelligent Systems (IEEE AIS'06)"and "Intelligent CAD's (CAD- 2006)". Scientific publication in 3 volumes. – M.: Physmathlit, 2006. – Vol. 3. – P. 29-34.
17. *Лебедев Б.К., Лебедев В.Б.* Эволюционная процедура обучения при распознавании образов деревьями // Известия ТРТУ. – 2004. – № 8 (43). – С. 83-88.
18. *Kureichik V.M., Lebedev B.K. and Lebedev V.B.* VLSI Floorplanning Based on the Integration of Adaptive Search Models. ISSN 1064_2307 // Journal of Computer and Systems Sciences International. – 2013. – Vol. 52, No. 1. – P. 80-96.

REFERENCES

1. *Han J., Kamber M.* Data mining: Concepts and Techniques. Morgan Kaufmann Publishers, 2001.
2. *Ian H. Witten, Eibe Frank and Mark A.* Hall Data Mining: Practical Machine Learning Tools and Techniques. 3rd ed. Morgan Kaufmann, 2011.
3. *Radchenko S.G.* Metodologiya regressionnogo analiza: Monografiya [Methodology regression analysis: Monograph]. K.: Korniychuk, 2011, 376 p.
4. *Dreyper N., Smit G.* Prikladnoy regressionnyy analiz [Applied regression analysis]. Moscow: Izdatel'skiy dom «Vil'yams», 2007, 912 p.
5. *Strizhov V.V., Krymova E.A.* Metody vybora regressionnykh modeley [Methods selection of regression models]. Moscow: VTs RAN, 2010, 60 p.
6. *Barsegyan A.A., Kupriyanov M.S., Stepanenko V.V., Kholod I.I.* Metody i modeli analiza dannykh: OLAP i Data Mining [Methods and models of data analysis: OLAP and Data Mining]. St. Petersburg: BKhV-Peterburg, 2004, 336 p.
7. *Sergienko V.I., Bondareva I.B.* Matematicheskaya statistika v klinicheskikh issledovaniyakh [Mathematical statistics in clinical research]. 2nd ed. Moscow: GEOTAR-Media, 2006, 304 p.
8. *Konar A.* Artificial intelligence and soft computing: behavioral and cognitive modeling of the human brain. CRC Press LLC. Boca Raton, Florida, 2000.
9. *Lavan'ini I., Man'o F., Seral'ya R., Tral'di P.* Kolichestvennye metody v mass-spektrometrii [Quantitative methods in mass spectrometry]. Moscow: Tekhnosfera, 2008, 176 p.
10. *Lebedev V.B., Lebedev O.B.* Roevoy intellekt na osnove integratsii modeley adaptivnogo povedeniya murav'inoy i pchelinoy koloniy [Swarm intelligence on the basis of the adaptive behaviour models integration of the ant and beer colonies], Izvestiya YuFU. Tekhnicheskie nauki [Izvestiya SFedU. Engineering Sciences], 2013, No. 7 (144), pp. 41-47.
11. *Lebedev B.K., Lebedev O.B.* Modelirovanie adaptivnogo povedeniya murav'inoy kolonii pri poiske resheniy, interpretiruemykh derev'yami [Modelling of an ant colony adaptive behaviour by search of the decisions interpreted by trees], *Izvestiya YuFU. Tekhnicheskie nauki* [Izvestiya SFedU. Engineering Sciences], 2012, No. 7 (132), pp. 27-35.
12. *Alpert C.J., Mehta D.P., and Sapatnekar S.S.* Handbook of Algorithms for Physical Design Automation. Boston, MA: Auerbach, 2009.
13. *Koza J.R.* Hierarchical genetic algorithms operating on populations of computer programs. In N.S. Sridharan (Ed.), Eleventh International Joint Conference on Artificial Intelligence. Morgan Kaufmann, 1989, pp. 768-774.
14. *Koza J.R.* Genetic Programming: On the Programming of Computers by means of Natural Selection. Cambridge MA, MIT Press, 1992.
15. *Kureichik V.M., Lebedev B.K., Lebedev O.B.* A hybrid partitioning algorithm based on natural mechansms of decision making, *Scientific and Technical Information Processing*, 2012, No. 39 (6), pp. 317-327.

16. *Lebedev B.K. and. Lebedev V.B.* Synthesis of Mathematical Expressions by Methods of Genetic Search, *Proceedings of the International Scientific Conferences "Intelligent Systems (IEEE AIS'06)"and "Intelligent CAD's (CAD- 2006)". Scientific publication in 3 volumes.* Moscow: Physmathlit, 2006, Vol. 3, pp. 29-34.
17. *Lebedev B.K., Lebedev V.B.* Evolyutsionnaya protsedura obucheniya pri raspo-znavanii obrazov derev'yami [Evolutionary procedure learning in pattern recognition trees], Izvestiya TRTU [Izvestiya TSURe], 2004, No. 8 (43), pp. 83-88.
18. *Kureichik V.M., Lebedev B.K. and Lebedev V.B.* VLSI Floorplanning Based on the Integration of Adaptive Search Models. ISSN 1064_2307, *Journal of Computer and Systems Sciences International*, 2013, Vol. 52, No. 1, pp. 80-96.

Статью рекомендовал к опубликованию д.т.н., профессор Я.Е. Ромм.

Лебедев Борис Константинович – Южный федеральный университет; e-mail: lebedev.b.k@gmail.com; 347928, г. Таганрог, пер. Некрасовский, 44; тел.: 88634371651; кафедра систем автоматизированного проектирования; профессор.

Лебедев Владимир Борисович – e –mail: lebvlad@rambler.ru; тел.: 88634371743; кафедра системного анализа и телекоммуникаций; доцент.

Лебедев Олег Борисович – e-mail: lbk@tsure.ru; тел.: 886343711651; кафедра систем автоматизированного проектирования; доцент.

Lebedev Boris Konstantinovich – Southern Federal University; e-mail: lebedev.b.k@gmail.com ; 44, Nekrasovsky, Taganrog, 347928, Russia; phone: +78634371651; the department of computer aided design; professor.

Lebedev Vladimir Borisovich – e-mail: lebvlad@rambler.ru; phone: +78634371743; the department of system analysis and telecommunications; associate professor.

Lebedev Oleg Borisovich – e-mail: lbk@tsure.ru; phone: +78634371651; the department of computer aided design; associate professor.