УДК 519.71

О.Г. Руденко, С.В. Мирошниченко, А.А. Бессонов

Программирование с экспрессией генов: способы кодирования и создания синтаксических деревьев

Проанализированы алгоритм программирования с экспрессией генов, различные схемы кодирования синтаксических деревьев для их последующей линеаризации и хранения, описана процедура создания начальной популяции. Разработана универсальная методика, позволяющая определить производительность модификаций алгоритма по любым метрикам. Сравнение схем кодирования позволило определить область их эффективного применения.

The gene expression programming algorithm is analyzed, various coding schemes of syntax trees for subsequent linearization and storage are considered, the procedure for creating the initial population is described. A universal method allowing to determine the algorithm modifications performance on any metrics is developed. The comparison of the coding schemes allows to determine the area of their effective application.

Проаналізовано алгоритм програмування з експресією генів, різні схеми кодування синтаксичних дерев для їх подальшої лінеаризації та зберігання, описано процедуру створення початкової популяції. Розроблено універсальну методику, яка дозволяє визначити продуктивність модифікацій алгоритму за будь-якими метриками. Порівняння схем кодування дозволяє визначити галузь їх ефективного застосування.

Введение. Эволюционные вычисления – термин, применяемый к техникам глобальной оптимизации, характерной особенностью которых является подобие их процессов биологической эволюции. Вместо работы с одним определенным вариантом решения, эволюционные техники начинают процесс поиска с популяции вариантов решения, созданных случайным образом. Начальная популяция эволюционирует в набор улучшенных решений, итеративно проходя три этапа: отбор, рекомбинацию и мутацию. Наиболее приспособленные решения отбираются для участия в рекомбинации с целью создания нового поколения, а оператор мутации вносит в него разнообразие. В ходе этого процесса лучшие особи передают свои характеристики следующим поколениям. При достаточном количестве итераций эволюционные алгоритмы способны находить решения многих сложных задач.

В качестве функции, определяющей количественную оценку особи – степень соответствия построенной модели условию задачи – и называемой функцией фитнеса (*fitness function*), чаще всего применяется значение среднеквадратичного отклонения (СКО). При таком подходе генетическое программирование и другие эволюционные техники могут за короткое время находить оптимальные либо околооптимальные решения задач и ситуаций. Использование этих техник не требует индивидуального ручного поиска оптимального решения каждой задачи – описанный процесс автоматизирован и выполняется компьютером, однако для этого необходимы значительные вычислительные ресурсы [1].

Генетические алгоритмы (ГА) были созданы в 1960 годы, когда Джон Холланд применил теорию биологической эволюции к компьютерным системам [2]. Как и все эволюционные компьютерные системы, ГА представляет собой упрощенную модель биологической эволюции. При таком подходе варианты решения задач кодируются символьными строками (обычно состоящими из нулей и единиц), и популяция этих вариантов решения участвует в процессе эволюции, цель которой – получение наилучшего решения.

Развитие идеи кодирования нелинейных структур разных размеров и форм Джоном Козой привело к появлению в 1992 г. генетического программирования (ГП) [3].

Программирование с экспрессией генов (ПЭГ), как в целом и ГА и ГП, – это генетический алгоритм с такими характерными чертами, как популяция особей, их отбор на основе приспособленности (фитнеса) и изменчивость на основе генетических операторов [4]. Фундаментальное отличие этих трех алгоритмов заключается в природе особей: в ГА это строки

фиксированной длины (хромосомы); в ГП – нелинейные сущности переменной длины и формы (синтаксические деревья). В ПЭГ особи кодируются в виде строк фиксированной длины (называемых геномом, либо хромосомами [5]), которые затем декодируются для получения синтаксических деревьев.

Различие между ГА и ГП выражено не сильно: обе системы используют один вид объектов, который служит как генотипом, так и фенотипом. Такой подход имеет одно из двух ограничений: простота применения генетических операторов к объектам означает их недостаточную выразительность и сложность этих объектов (в случае ГА), а их сложность ведет к трудностям при воспроизводстве и модификации (в случае ГП).

ПЭГ лишено указанных ограничений и обладает следующими преимуществами:

- простота хромосом: линейность, компактность, относительно небольшой размер, простота применения операторов, таких как репликация, мутация, рекомбинация, перенос и пр.;
- экспрессия (декодирование компактной структуры) генома порождает фенотип – синтаксическое дерево, которое может представлять математическую формулу либо программу. Значения вычисленной формулы, выполненной программы служат для численной оценки приспособленности особи с помощью функции фитнеса.

Таким образом, участие хромосомы в процессе воспроизведения зависит от фитнеса дерева, которое она кодирует. Для этого требуется универсальная система перевода формата генома в формат дерева и обратно, при которой любое изменение хромосомы с помощью операторов всегда приводит к синтаксически корректному дереву, обеспечивая приспосабливаемость и эволюционирование.

Исходный алгоритм ПЭГ

На рис. 1 показана блок-схема алгоритма ПЭГ [4].

Процесс начинается с конструирования популяции случайным образом созданных хромосом. После экспрессии (декодирования) хромосом каждая особь выполняется на заданном наборе входных данных (в виде координат точек для математических задач, таблиц истинности при поиске булевых выражений, наборов признаков при решении задач классификации и т.д.). На основе результатов выполнения особей вычисляется фитнес каждой из них, позволяющий отобрать и подвергнуть изменениям особи, нацеленные на получение потомства – популяции с новыми характеристиками. С этой популяцией проводятся те же операции: декодирование генома, отбор, воспроизведение с модификациями. Процесс повторяется заданное количество итераций либо до получения решения.

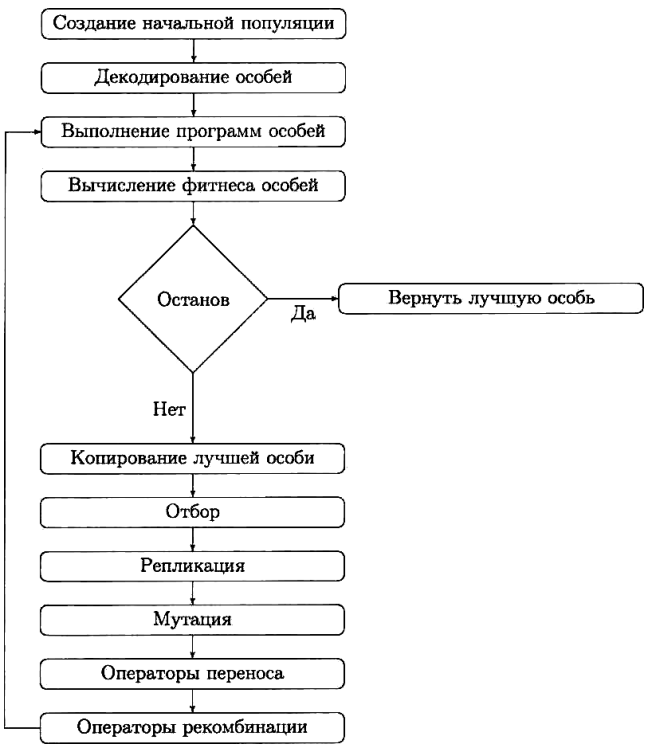

Рис. 1

Воспроизведение включает в себя не только репликацию, но и выполнение генетических операторов, вносящих разнообразие. Во время репликации геном особи в точности, без изменений копируется и переносится в новое поколение. Очевидно, этот оператор не способен вносить разнообразие, потому требуется вмешательство оставшихся операторов, случайным образом выбирающих особи, к которым они будут применены. Так, в ПЭГ особь может быть модифицирована сразу несколькими операторами, а может оказаться и не измененной [5].

Открытые рамки считывания. Структурная организация генома в ПЭГ более понятна в биологической терминологии открытых рамок считывания (*open reading frames – ORF*) – последовательностей генома, начинающихся стартовым кодоном, завершающихся стоповым кодоном и содержащих кодирующие элементы между ними. В ПЭГ последовательность всегда начинается с первого элемента гена, но ее конец не всегда совпадает с последним элементом. Таким образом, в ПЭГ нередка ситуация, когда участок от последнего элемента последовательности до конца гена не участвует в построении дерева, такие участки называют некодирующими, либо *нитронами*.

В качестве примера рассмотрим следующее алгебраическое выражение:

$$\sqrt{(a+b)\times(c-d)}\,. \qquad (1)$$

Выразить формулу (1) можно в виде синтаксического дерева (рис. 2).

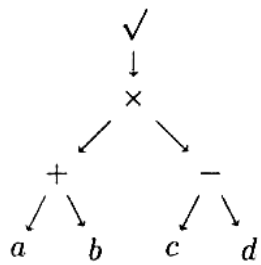

Рис. 2.

Количество ответвлений (дочерних узлов) у узлов, представляющих функцию, равно количеству аргументов этой функции. Узлы, соответствующие терминалам (переменным константам), не имеют дочерних узлов. Изображенное дерево и есть фенотипом особи в ПЭГ, в то время как генотип записывается следующим образом (в верхнем ряду показаны индексы узлов, в нижнем – значения закодированных символов):

01234567

Q + –abcd*,

где очередность элементов обусловлена обходом графа в ширину. Это выражение – открытая рамка считывания, в ПЭГ такие записи называются *K*-выражениями, а язык преобразования *K*-выражения в дерево и обратно – языком *KARVA*. Построение дерева из генома завершается, когда каждому ответвлению узла-функции поставлен в соответствие дочерний узел. Листья дерева, следовательно, – это терминалы.

Такая структура генома ПЭГ позволяет кодировать деревья различных размеров и форм, оперируя генами фиксированной длины, каждый из которых состоит из рамки считывания переменной длины и, при необходимости, некодирующего участка для заполнения пространства между концом кодирующей последовательности и концом гена. Хромосома особи состоит из одного или нескольких таких генов. Функция некодирующих участков, таким образом, состоит в обеспечении того, что длина рамки считывания всегда будет меньше или равна длине гена. Это позволяет гарантировать декодирование синтаксически правильного дерева после любых модификаций генома без ограничений и необходимости проведения сложных процедур проверки и редактирования. Отсутствие множества ограничений на генетические операторы есть коренным отличием ПЭГ от ГП.

Геном в ПЭГ. Каждый ген ПЭГ состоит из головы и хвоста. Голова содержит символы как функций, так и терминалов. Хвост содержит исключительно терминалы. Для каждой конкретной задачи подбирается длина головы гена h. Обозначим максимальное количество аргументов среди всех функций функционального множества как n (для большинства арифметических функций $n = 2$; для оператора «ЕСЛИ ТО ИНАЧЕ» $n = 3$).

Тогда минимальную возможную длину хвоста гена t можно вычислить по следующей формуле:

$$t(h, n) = h \times (n - 1) + 1.$$

Следует отметить, что такая структура соответствует схеме кодирования путем обхода дерева в ширину, принятого в оригинальном алгоритме ПЭГ. Другие схемы кодирования, которые будут описаны, не требуют условного деления генома на голову и хвост.

Еще один параметр, требующий подбора под каждую задачу – количество генов (участ-

ков кода фиксированной длины, содержащих одно дерево) в хромосоме. Обычно используется больше одного гена, и такие хромосомы называют мультигенными. Длина всех генов устанавливается равной для простоты применения операторов. Хромосома представляет собой символьную строку, полученную путем конкатенации всех генов.

Таким образом, ген – объект, состоящий из пяти атрибутов: $G = (E, T, F, Op, S)$, где E – генотип, T – терминальное множество, F – функциональное множество, Op – множество операторов, S – значение фитнеса на определенном элементе данных. Пример полной записи гена таков:

(«*++–*aabcd*», «*abcd*», «+–*/», 0).

Хромосома – объект, состоящий из четырех атрибутов: $C = (G, L, Op, S)$, где G – набор генов, L – связующий оператор. Пример полной записи хромосомы:

$C1 = (\{G0, G1, G2\}$, «*ab*», +,«+ –*/», 0,7).

Пример символьной записи хромосомы:

012345678 012345678 012345678

*–b*babbab *Qb + abbba –*Qabbaba.*

Данная хромосома содержит три гена, каждый из которых начинается с позиций, обозначенных индексом ноль. Окончание рамки считывания каждого гена можно определить лишь после построения закодированного в нем синтаксического дерева; в приведенном примере декодирование первого гена завершается на позиции 4 (последний элемент), второго и третьего – на позиции 5.

Каждое дерево может использоваться как в отдельности, с расчетом фитнеса для каждого гена, так и в совокупности, формируя более сложное дерево, фитнес которого и отражает приспособленность особи. Во втором случае каждое субдерево есть отдельной сущностью, компонентом иерархической структуры, представляющей большую ценность, чем сумма ее частей. Независимая друг от друга эволюция генов хромосомы как отдельных блоков иерархической системы при мультигенном подходе позволяет более эффективно решать сложные задачи.

Взаимодействие субдеревьев происходит с помощью связующих функций, для алгебраических выражений это, как правило, функция арифметического суммирования, для булевых – логическое ИЛИ. Обычно связующая функция устанавливается априори, однако может также быть добавлена в геном и выбираться в процессе эволюции. На рис. 3 показано синтаксическое дерево, декодированное из рассматриваемой трехгенной хромосомы с априорно заданной связью с помощью функции суммирования.

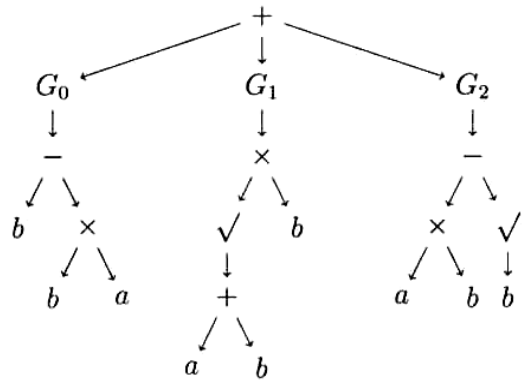

Рис. 3

Такой простой способ кодирования синтаксических деревьев в виде линейных структур может быть использован и в других видах эволюционных алгоритмов, например в иммунных сетях [6].

Сведение к ГА. Простейшим представлением генома в ПЭГ является ген из одного терминала. Организация хромосомы, состоящей из одноэлементных генов делает ПЭГ эквивалентным ГА. Однако для решения комбинаторных задач требуются особые связывающие функции, отличные от простых арифметических и булевых. Например, для задачи коммивояжера связь отображает расстояние между городами, представленными соответствующими генами. Если обозначить каждый город из девяти отдельным символом, то следующая хромосома из девяти элементов представит один из возможных путей обхода: *CADEBHFIG*.

Если оптимизационная задача содержит N классов терминалов, хромосома может состоять из N групп одноэлементных генов, называемых мультигенными группами (МГГ –

multigene family, *MGF*) [7]. Например, при отображении одного шестиэлементного множества {1, 2, 3, 4, 5, 6} в другое {*A*, *B*, *C*, *D*, *E*, *F*} хромосома будет состоять из двух шестиэлементных групп:

012345 012345

632451 *EDFCBA*

Особенность МГГ – то, что она не содержит повторяющихся элементов, а отображает перестановку элементов некоторого конечного множества. Потому операторы мутации и рекомбинации к такой структуре генома неприменимы в силу вероятности некорректных последовательностей с повторяющимися и отсутствующими элементами. Следовательно, для внесения генетического разнообразия должны быть созданы новые операторы, порождающие корректные структуры.

Один из таких операторов – инверсия, впервые описанная в [1], суть которой состоит в изменении порядка следования элементов в случайным образом выбранного участка МГГ. Так, например, результатом применения к хромосоме из предыдущего примера может стать следующий геном:

012345 012345

632451 *ECFDBA*

Из проведенных экспериментов следует, что оператор инверсии обладает более сильной изменяющей способностью и, следовательно, наиболее способствует скорейшему обнаружению решения. Оператор перемещения гена располагает случайным образом выбранный ген по другой позиции, сохраняя порядок следования остальных генов группы. Использование данного оператора вместе с более мощными, такими как инверсия, может быть полезно для точной подстройки решения. Однако эксперименты показали [7], что инверсия, используемая в одиночку, более эффективна.

Оператор перестановки генов меняет местами два случайным образом выбранных гена в пределах МГГ. Исследование оператора перемещения участков гена показало его крайнюю неэффективность – особи-потомки имели, как правило, худший фитнес, чем родительские особи.

Иерархическая структура дерева. В исходном алгоритме ПЭГ гены мультигенной хромосомы – это не связанные между собой субдеревья, объединяемые связующим оператором (например, арифметическим сложением). Каждый из генов задает свою собственную функцию, обнаруженную эволюционным алгоритмом. В терминологии ПЭГ эти функции называются *автоматически определенными* (*ADF – automatically defined function*). Такие обособленные функции могут использоваться как законченный блок при построении итогового синтаксического дерева. Для этого требуется усложнить структуру хромосомы – выстроить иерархию генов. В такой сложной хромосоме первые *N* генов задают функции, а следующие *M* генов представляют собой клетки – деревья, терминальное множество которых состоит из *N* терминальных символов, обозначающих функции первых *N* генов хромосомы. Каждая клетка – это отдельный способ комбинации генов-функций. Рассмотрим хромосому, состоящую из трех генов-функций и двух клеток (рис. 4):

0123456 0123456 0123456 012345678 012345678

**Q-bbab Q*baabb -/abbab* *+21*Q*1102 /*21+1011

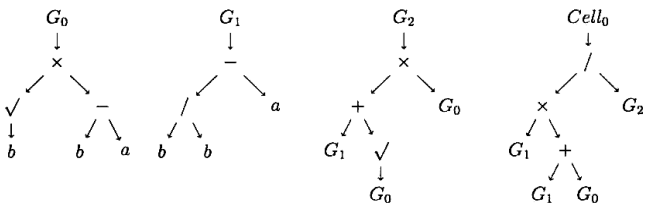

Рис. 4

Многоклеточные хромосомы могут быть поразному использованы при решении задач: либо как отдельные выходы (например, отражающие различные классы в задаче многокласссовой классификации), либо как задействование какой-либо одной (обычно лучшей) из них в качестве единственного выхода.

В ходе экспериментов [8] выяснено, что при решении простых задач наиболее эффективной показала себя структура хромосомы, состоящая всего из одного гена-функции и одной клетки.

Для повышения выразительной силы мультигенной хромосомы можно применить другой, более простой вариант иерархической структуры: любой ген (кроме первого по порядку) может использовать значения генов, предшествующих им в хромосоме [9]. Для этого также используются терминальные символы, соответствующие индексу гена, на значение которого указывает ссылка. Продемонстрировать преимущества такого подхода можно на примере функции a^{2^n} при функциональном множестве {+, –, ×, /}. Традиционное кодирование ПЭГ требует в этом случае $2^{n+1} - 1$ символов 2^n символа a и $2^n - 1$ символов *, строка генома выглядит так: *******$aaaaaaaa$. При подходе *MERGE* (*Multi ExpRession GEne programming* – ПЭГ с несколькими выражениями) для этого требуется всего $3 \times n$ символов и хромосома из трех генов:

0: *aa

1: *00

2: *11,

где 0, 1 – терминальные символы, кодирующие ссылки на гены с индексами ноль и единица. Для гарантии отсутствия циклических ссылок ген *ноль* не может ссылаться на другие, ген *единица* может ссылаться на ген *ноль*, ген *два* – на гены *ноль* и *единица*, ген *три* – на гены *ноль*, *единица*, *два* и т.д.

Кроме того, такая структура мультигенной хромосомы позволяет использовать значения каждого из своих генов как самостоятельный геном особи, имеющий собственное значение фитнеса. Максимальный из фитнесов генов выбирается фитнесом всей хромосомы.

Экспериментальная проверка подхода ПЭГ с несколькими выражениями показала его успешность в решении задач даже короткими генами с длиной головы в один–два элемента. Помимо этого, выяснилось, что установка длины гена выше оптимальной не ведет к падению производительности алгоритма, в то время как чрезмерная сложность особей, кодируемых длинным геномом в исходном ПЭГ, не позволяла быстро находить приемлемые решения.

В работе [10] предложена другая модификация автоматически определяемых функций, описанных в [8]. Символ ссылки на ген представляет собой уже не терминал, отображающий значение запрашиваемого гена для этого набора данных, а элемент особого функционального множества, декодируемого как узел дерева с дочерними узлами-аргументами функции. Отличие состоит в том, что запрашиваемый ген оперирует аргументами, подаваемыми геном-клеткой, а не переменными выборки данных. Модификация позволила [10] на треть ускорить сходимость.

Схемы кодирования дерева

Префиксное кодирование. В исходном алгоритме ПЭГ декодирование дерева выполняется путем линейного извлечения элементов генома и установки в узлы формируемого дерева при его обходе в ширину. При замене обхода дерева в ширину обходом в глубину повышается связность генома – уменьшается расстояние между элементами, соответствующими соседним узлам дерева, а субдерево кодируется непрерывной строкой в геноме. Такой подход способствует сохранению субструктур, что делает менее разрушительным применение операторов скрещивания [11]. Сохранение в популяции субструктур (субдеревьев) лучших особей и их рекомбинация позволяет ускорить сходимость алгоритма: оптимальное решение обнаруживается на более ранних поколениях, т.е. в несколько раз быстрее, качество оптимального решения зачастую выше, чем результат работы оригинального алгоритма ПЭГ. Отдельные поддеревья лучших особей можно сохранять в отдельной, так называемой элитной группе [12], кодируя их дополнительными терминальными символами.

Поскольку при таком подходе нарушается гарантия получения синтаксически корректного дерева из любого генома, сформированного согласно правилам, требуется ввести процедуру динамической валидации дерева. К участию в эволюционном процессе допускаются только те особи, декодирование которых порождает корректное дерево. Такая система получила название P-ДЕР.

Для взаимодействия между особями и элитной группой были введены два новых генетических оператора: оператор сжатия, заменяющий последовательность в геноме особи соответствующим терминальным символом из элитной группы, и оператор раскрытия, выполняющий обратную операцию – замену терминального символа на соответствующую последовательность из элитной группы. Внесение данной модификации в Р-ДЕР привело к обнаружению решений, имеющих значительно меньшую погрешность.

Кодирование с наложениями. В работе [13] рассматривается разновидность декодирования генома путем обхода дерева в глубину, когда последовательно идущие субдеревья накладываются друг на друга. Элементы генома последовательно считываются:

- если текущий символ принадлежит к терминальному множеству – поместить его в синтаксическое дерево как листовой узел;
- если текущий символ принадлежит к функциональному множеству – поместить его в синтаксическое дерево как узел с количеством дочерних узлов, равным количеству аргументов помещаемой функции; первый дочерний узел – следующий в геноме за текущим, второй – следующий за ним и т.д.; дочерние субдеревья заполняются, следуя этой же процедуре;
- при выходе за пределы генома в качестве текущего символа берется первый элемент терминального множества.

Коренное отличие описанного способа декодирования от методов обхода в ширину и в глубину заключается в том, что элементы генома могут быть задействованы в построении дерева неоднократно, и размер получаемого дерева, как правило, бывает больше при равном размере генома.

Рассмотрим на примере строки $**a+*bc$, синтаксическое дерево, образующееся при ее декодировании (рис. 5).

При использовании традиционного метода декодирования, принятого в ПЭГ, данному дереву соответствует строка $**aa + *bbc$, содержащая на два символа больше.

Выразительная сила генома при декодировании с наложениями значительно выше, что показано в табл. 1, отражающей соответствие максимального возможного размера (количества узлов) синтаксического дерева при равных размерах гена.

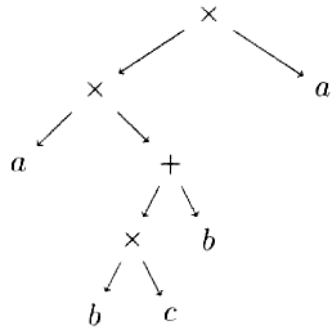

Рис. 5

Таблица 1. Сравнение выразительной силы ПЭГ с наложениями

Размер гена	ПЭГ	ПЭГ с наложениями
3	3	9
5	5	25
7	7	67
9	9	177
11	11	465
13	13	1219
15	15	3193
17	17	8361

Повышение выразительной силы генома позволяет использовать более короткие хромосомы, что, в свою очередь, значительно понижает вычислительную сложность алгоритма: разница в скорости обработки экспоненциально растет, при длине хромосомы в 23 элемента ускорение достигает 10 раз. Алгоритм, основанный на таком подходе, получил название *EAOGE – Evolutionary Algorithm based on Overlapped Gene expression* – эволюционный алгоритм на основе ПЭГ с наложениями.

Нейтральные гены

После декодирования хромосомы ПЭГ и конструирования синтаксического дерева количество узлов полученного дерева в предельном случае будет равно размеру хромосомы, однако в большинстве случаев будет меньшим. Те элементы в хвостах генов, которые не были использованы при построении дерева, называются некодирующими участками. Кроме того, полученное дерево может содержать выраже-

ния вида ($a - a$), ($a \times 0$), ($a - 0$) и другие, которые при семантическом анализе упрощаются, устраняя данные интроны и образуя более компактное дерево. Множество подобных участков – часто повторяющихся последовательностей, интронов, псевдогенов и прочего – встречается в природе в ДНК живых существ. Потому логично предположить, что и в искусственном геноме нейтральные участки могут быть полезны. Среднее количество нейтральных последовательностей в популяции растет с увеличением как количества генов в хромосоме, так и размера каждого из них.

Исследования работы алгоритма с различной длиной генома показали [14], что эволюционирование особей с высокой степенью избыточности генома происходит эффективней. Это связано с тем, что у каждого решения, представленного в наиболее компактной форме, существует множество менее компактных вариантов представления, увеличенных в размерах посредством интронов. Кроме того, накапливаемые в некодирующих участках мутации могут быть использованы в дальнейшем, если изменения в голове гена (добавление функциональных узлов) приведут к увеличению размера дерева.

Методика тестирования

Создание начальной популяции. В изученных работах о ПЭГ не уделяется внимания процедуре создания начальной популяции, а именно конструированию случайной особи. Указано, что строка хромосомы обходится с начала и по порядку, и каждый символ задается случайным образом: тип узла может быть равновероятно задан как функциональный элемент, как переменная (подаваемая на вход модели) и как числовая константа. Очевидно, что размер дерева определяется количеством подряд (близко) идущих функциональных узлов, начиная с корневого узла (первого символа хромосомы): из условия равновероятности типов узлов (три типа – 33,333 процента каждый) следует, что вероятность создания дерева, состоящего всего из одного элемента (терминала: переменной либо константы), составляет 66,67 процента. При этом можно заметить, что средняя длина получаемых деревьев не зависит от задаваемой длины хромосомы, но определяется установленной вероятностью появления того или иного типа.

При оригинальном кодировании элементов путем обхода узлов дерева в ширину логично предположить, что вероятность появления функциональных узлов в головной части хромосомы следует повысить, в то время как в хвостовой части могут быть помещены исключительно терминальные узлы. В таком случае определение оптимальных численных значений вероятностей представляет собой отдельную задачу для исследования. Кроме того, данный подход не применим к другим способам кодирования, поскольку не обладает должной универсальностью.

Для решения указанной проблемы и закрепления взаимосвязи между размером дерева и длиной генома была разработана следующая процедура инициализации. Обход строки так же выполняется посимвольно, начиная с ее начала (независимо от способа кодирования). При этом не составляет трудности определить местонахождение кодируемого символом узла в дереве, а именно его глубину: первый символ – это корневой узел, далее – в соответствии со способом кодирования.

Вырожденное дерево, состоящее из одного терминала, как правило, не представляет интереса, так как решаемая таким образом задача не требует применения ПЭГ. В таком случае вероятность задания функционального типа корневому узлу устанавливается равным 100 процентам. В то же время для ограничения размера дерева заданным значением (несмотря на способность префиксного кодирования и ПЭГ с наложениями создания деревьев, превышающих в размере длину генома) в целях удобства пользования уместно ограничить узлы максимальной возможной глубины листовыми. Следовательно, вероятность появления функционала в указанных местах равна нулю. Требует решения задача распределения вероятности функциональных узлов в зависимости от глубины вложенности.

Предлагается использование следующей формулы:

$$P_{funk}\left(level, maxlevel\right) = \sqrt{1-\left(\frac{level-1}{maxlevel-1}\right)^2},$$

где *level* – глубина узла, *maxlevel* – максимально возможная глубина дерева (рис 6).

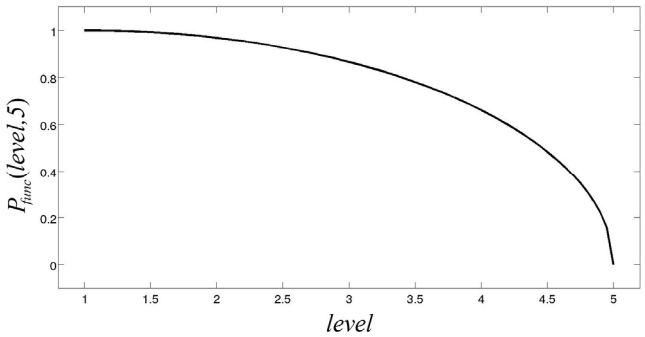

Рис. 6

Таким образом, процедура инициализации задает узлу глубины *level* тип функции при выполнении условия *random*(0... 1) < P_{func}(*level*, *maxlevel*), обеспечивая сбалансированную начальную популяцию особей прогнозируемого размера. Данный метод конструирования новой особи случайным образом показал свою эффективность и был использован во всех экспериментах.

Методы оценки производительности. Для проведения экспериментов была реализована система ПЭГ на языке программирования *C*, принимающая в качестве аргумента файл с набором точек (заданных координатами) и выдающая лучшую обнаруженную формулу, набор точек – модель, полученную с использованием этой формулы, – и динамику СКО лучшей особи с течением времени. В исходный алгоритм затем поочередно добавлялись описанные далее модификации, с возможностью включения и исключения каждой из них в любых комбинациях. Так, например, при каждом запуске можно выбрать оператор селекции (рулетку, отбор по плотности либо турнир), вероятность мутации, способ кодирования и др.

Рандомизированная природа алгоритма ПЭГ приводит к необходимости использовать статистические методы для оценки его эффективности. Для этого каждая исследуемая комбинация модификаций алгоритма запускалась множество раз (обычно 100), анализу подвергались значения СКО лучших моделей, полученных в ходе каждого запуска. Наиболее интересные показатели – СКО наилучшей модели среди всех запусков e_b и доля успешных запусков, т.е. таких запусков, в ходе которых была получена модель с приемлемой для данной задачи точностью r_f.

Обнаружено, что в большинстве случаев подбор комбинаций параметров и модификаций позволяет добиться улучшения какой-либо одной из характеристик e_b и r_f, но не обеих одновременно. Направленность на снижение e_b наиболее оправдана в тех случаях, когда процесс поиска модели не стеснен во времени, а целью является получение наиболее точной модели. Повышение доли успешных запусков r_f есть целью при поиске оптимального алгоритма решения задач, требующих быстрого построения моделей с приемлемой заданной точностью.

Таким образом, e_b отражает способность алгоритма к обнаружению лучшей из возможных моделей, а r_f – вероятность получения модели достаточной точности при очередном запуске алгоритма.

Сравнение систем, собранных с различными параметрами, проводилось на трех задачах поиска формул, описывающих такие модели:

- $y(x) = \sin x,\ \ x \in [-5, 4.6]$;
- Функция Розенброка: $z(x, y) = (1 - x)^2 + 100(y - x^2)^2,\ \ x \in [-3, 3], y \in [-3, 3]$;
- Сумма четырех сигмоид:

$$f(x,y) = \frac{1}{2}\exp\left[-\frac{(9x-2)^2+(9y-2)^2}{4}\right] + \frac{3}{4}\exp\left[-\frac{(9x+1)^2}{49}-\frac{(9y+1)^2}{10}\right] + \frac{1}{2}\exp\left[-\frac{(9x-7)^2+(9y-3)^2}{4}\right] - \frac{1}{5}\exp\left[-(9x-4)^2-(9y-7)^2\right],$$

$$x \in [-2,2], y \in [-2,2].$$

При моделировании функций синуса и Розенброка использовано функциональное множество, состоящее из четырех арифметических действий, в последнем случае – полное множество, в состав которого входят арифметические (сложение, вычитание, умножение, деление), тригонометрические (sin, cos, tan, sinh, cosh, tanh), степенные (возведение в степень, извлечение квадратного корня, экспонента, логарифм). Полученные с применением этих формул модели изображены на рис. 7. Значения приемлемых СКО моделей, при которых задача будет считаться решенной, составляют для этих задач, соответственно, 0,08; 0,05 и 0,06.

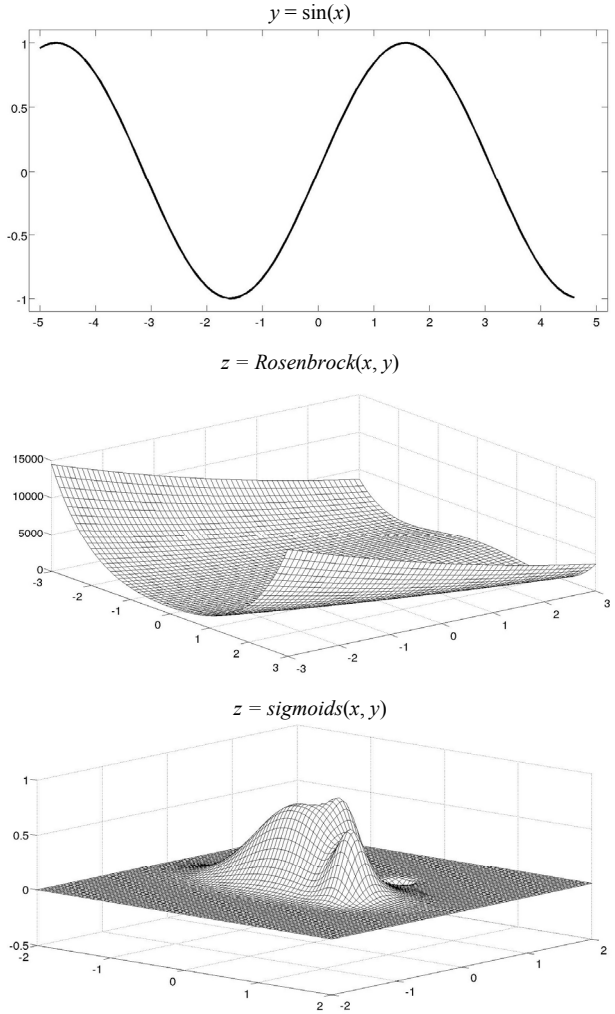

Рис. 7

Результаты экспериментов. Сравнение эффективности реализованной системы ПЭГ с рассмотренными модификациями способа кодирования с исходным алгоритмом приведено в табл. 2, для возможности сопоставления производительности алгоритмов было зафиксировано время, выделяемое каждому запуску.

Таблица 2. Эффективность алгоритма при разной глубине синтаксического дерева

Эксперимент. Глубина	Sin		Rosen		Sigmas	
	e_b	r_f	e_b	r_f	e_b	r_f
3 (фер.)	0,369	0	0,081	0	0,132	0
4 (фер.)	0,075	2	0,034	6	0,064	0
5 (фер.)	0,022	39	0,001	12	0,053	2
6 (фер.)	0,013	22	0,042	2	0,051	1
7 (фер.)	0,034	4	0,079	0	0,108	0
3 (префикс.)	0,369	0	0,087	0	0,132	0
4 (префикс.)	0,076	2	0,000	6	0,072	0
5 (префикс.)	0,011	45	0,033	10	0,047	2
6 (префикс.)	0,011	22	0,035	2	0,070	0
7 (префикс.)	0,042	10	0,081	0	0,086	0
3 (налож.)	0,369	0	0,081	0	0,132	0
4 (налож.)	0,075	2	0,029	И	0,050	6
5 (налож.)	0,030	39	0,017	8	0,038	2
6 (налож.)	0,045	13	0,004	3	0,049	1
7 (налож.)	0,063	2	0,035	1	0,054	1

При небольшой длине генома ПЭГ с наложениями существенно превосходит конкурирующие модификации в способности обнаруживать отличные решения. Дальнейшее увеличение максимально возможного размера дерева до глубины в четыре узла несколько меняет картину: префиксное ПЭГ значительно выигрывает в вероятности успешного обнаружения решения, тогда как выходом ПЭГ с наложениями будет лучшая модель. При допущении деревьев глубиной в пять узлов наилучшие показатели имеет префиксное ПЭГ, показатели всех алгоритмов достигают своего максимума.

Таким образом, обе модификации превосходят исходный алгоритм в производительности, однако каждая из них занимает свою нишу, потому выбор одной из них определяется в зависимости от нужд исследователя.

Из таблицы также следует, что удлинение генома ведет к возрастанию пространства поиска, появлению деревьев большего размера, следовательно, ведет к более длительному расчету популяции. Целесообразно сравнивать модели, на построение которых было затрачено равное время. Из таблицы видно, что для каждой задачи имеется некая оптимальная длина

генома. Меньший размер генов не позволяет выразить деревья достаточной сложности, а при большем – требуется на порядки больше вычислительного времени.

Заключение. Итак, проанализирован алгоритм ПЭГ, рассмотрены различные схемы кодирования синтаксических деревьев для их последующей линеаризации и хранения, описана детальная процедура создания начальной популяции, которой не было уделено внимания в изученных работах. Разработана универсальная методика, позволяющая определить производительность модификаций алгоритма по любым метрикам, две из которых отобраны как представляющие наибольший интерес. Сравнение схем кодирования позволило определить область их эффективного применения.

1. *High* Performance Evolutionary Computation / E. Nunez, E. Banks, P. Agarwal et al. – HPCMP Users Group Conf., Nashville, TN, 2006. – P. 354–359.
2. *Holland J.H*. Adaptation in Natural and Artificial Systems // The MIT Press, 1992. – 228 p.
3. *Koza J.R*. Genetic Programming: On the Programming of Computers by Means of Natural Selection, Cambridge, MA, USA: Ibid, 1992. – 840 p.
4. *Ferreira C*. Gene Expression Programming: a New Adaptive Algorithm for Solving Problems // Complex Systems. – 2001. – **13**, N 2. – P. 87–129.
5. *Ferreira C*. Gene Expression Programming in Problem Solving // Soft Comp. and Industry Recent Appl. – Germany: Springer-Verlag, 2001. – P. 635–654.
6. *Karakasis V.K*., *Stafylopatis A*. Efficient evolution of accurate classification rules using a combination of gene expression programming and clonal selection // IEEE Transactions on Evolutionary Computation. – 2008. – **12**, N 6. – P. 662–678.
7. *Ferreira C*. Combinatorial Optimization by Gene Expression Programming: Inversion Revisited // Proc. of the Argentine Symp.on Artificial Intel. – Santa Fe, Argentina, 2002. – P. 160–174.
8. *Ferreira C*. Automatically Defined Functions in Gene Expression Programming // Genetic Systems Programming: Theory and Experiences. – Germany: Springer, 2006. – **13**. – P. 21–56.
9. *MERGE*: A Novel Evolutionary Algorithm Based on Multi Expression Gene Programming / S. Dai, C. Tang, M. Zhu et al. // Proc. of the 4th Int. Conf. on Natural Computation (ICNC'08), Washington, DC, USA // IEEE Comp. Society, 2008. – **01**. – P. 320–324.
10. *GEP-NFM*: Nested Function Mining Based on Gene Expression Programming / T. Li, C. Tang, J. Wu et al. // Ibid. – **06**. – P. 283–287.
11. *Prefix* Gene Expression Programming / X. Li, C. Zhou, W. Xiao et al. // Proc. of Genetic and Evolutionary Computation Conf. (GECCO'2005). – Washington, D.C, USA. – 2005. – P. 25–31.
12. *Direct* Evolution of Hierarchical Solutions with Self-Emergent Substructures / X. Li, C. Zhou, W. Xiao et al. // Proc. of the 4-th Int. Conf. on Machine Learning and Applications (ICMLA'05). – Los Angeles, Calif. // IEEE press, 2005. – P. 337–342.
13. *Evolutionary* Algorithm Based on Overlapped Gene Expression / J. Peng, C. Tang, J. Zhang et al. // Lect. Notes in Comp. Science. – Springer, 2005. – **3612**. – P. 194-204.
14. *Ferreira C*. Genetic Representation and Genetic neutrality in gene Expression Programming // Advances in Complex Syst., 2002. – **5**, N 4. – P. 389–408.

Поступила 02.03.2015
Тел. для справок: +38 057 70-50-862, 70-21-354 (Харьков)
E-mail: *o.bezsonov@gmail.com*, *S.Miroshnichenko@gmail.com*

●

Внимание!

Требования к оформлению статей размещены на сайте журнала: ***http://usim.irtc.org.ua***

Для соответствия журнала современным научно-метрическим базам, авторы должны подать на английском языке:

– расширенную аннотацию на 1–1,5 с. с выделением рубрик: *Introduction*, *Purpose*, *Methods*, *Results*, *Conclusion*);

– фамилию и инициалы автора,

– название статьи;

– ключевые слова (*Keywords*);

– место работы, должность и адрес;

– ученая степень, звание;

– пристатейный список литературы на латинице (для русскоязычных ссылок – транслитерация Ф.И.О. авторов и названия журнала, название статьи – перевод на англ.).