УДК 519.71

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

Программирование с экспрессией генов: генетические операторы

Проанализированы методы и средства, направленные на ускорение процесса получения модели и повышение ее качества с помощью алгоритма программирования с экспрессией генов. В ходе экспериментов выявлен ряд ограничений производительности традиционного алгоритма, в связи с чем рекомендуется направить дальнейшие исследования на создание подходов и методов, способствующих преодолению таких ограничений.

Проаналізовано методи та засоби, спрямовані на пришвидшення процесу отримання моделі і підвищення її якості за допомогою алгоритму програмування з експресією генів. В процесі експериментів виявлено низку обмежень продуктивності традиційного алгоритму, у зв'язку з чим рекомендовано спрямувати подальші дослідження на створення підходів і методів, які сприяли б подоланню таких обмежень.

Введение. Эволюционные алгоритмы (ЭА) представляют собой направление в искусственном интеллекте, использующем и моделирующем биологическую эволюцию. Среди ЭА – стохастических и включающих в себя эволюционное программирование, эволюционные стратегии, генетические алгоритмы, генетическое программирование, в частности, программирование с экспрессией генов, наиболее распространены генетические алгоритмы (ГА).

ГА – это метод машинного обучения, основанный на механизмах отбора в природе, которые ведут случайный и параллельный поиск решений, оптимизирующих заранее определенную фитнес-функцию.

Генетическое программирование (ГП) создано в качестве самостоятельного направления в области эволюционных вычислений, несмотря на то, что этот метод может быть также интерпретирован как особый класс ГА.

В ГП адаптированы и применены аналогичным ГА образом основные механизмы отбора, рекомбинации и мутации. Более общее представление проблем в ГП позволяет определить особь популяции как структуру, формулу или даже в более общем случае как программу. Это позволяет рассматривать новые области применения ЭА.

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).

Программирование с экспрессией генов (ПЭГ) реализуется с помощью алгоритма [1]:

1. Создание начальной популяции
2. Декодирование особей
3. Выполнение программ особей
4. Вычисление фитнес-функции (ФФ) особей
5. Проверка критерия останова алгоритма (достигнута требуемая точность либо истекло лимитированное время выполнения)
6. Копирование лучшей особи (элитизм)
7. Отбор
8. Репликация
9. Мутация
10. Операторы переноса
11. Операторы рекомбинации
12. Возврат к п. 2

В ПЭГ применяются следующие операторы: репликации, мутации, копирования со сдвигом, копирования со сдвигом в корень, перестановки генов, одно- и двухточечной рекомбинации, генной рекомбинации.

Репликация представляет собой самый простой оператор, не вносящий разнообразие в геном, и используется в паре с оператором отбора для копирования особей в новую популяцию в соответствии со значением ФФ и случайностью, вводимой оператором рулетки.

Мутация (замена символа – элемента гена, соответствующего узлу дерева) может возникнуть в любом месте хромосомы. Вероятность оператора, как правило, задается эквивалентной двум мутациям в хромосоме. Элементы,

подверженные мутации в голове гена, могут быть изменены на любой другой символ (функцию или терминал), в хвосте – только на терминал. Данное ограничение должно гарантировать сохранение структуры хромосомы и обеспечение декодирования синтаксически правильного дерева. Среди всех модифицирующих операторов мутация наиболее существенный и эффективный, так как в отличие от остальных, комбинирующих существующие участки генома, мутация направлена на создание новых элементов, а потому вносит наиболее радикальные изменения, расширяя область поиска решений.

Рекомбинация – операция по перемещению последовательных участков генома в пределах хромосомы.

Цель данной статьи – исследование влияния выбора генетических операторов на свойства алгоритма ПЭГ.

В статье [2] описана методика оценки эффективности модификаций алгоритма ПЭГ, заключающаяся в статистической обработке результатов множества запусков с целью получения среднеквадратичной ошибки (СКО) наилучшей модели среди всех запусков (e_b) и доли успешных запусков (r_f), т.е. таких, в ходе которых была получена модель с приемлемой для данной задачи точностью.

Для тестирования использованы модели синусоиды (*sin*), функция Розенброка (*rosen*) и сумма четырех сигмоид (*sigmas*).

Эти же методика и модели использованы в данной работе для сравнения генетических операторов.

Операторы отбора

Процесс эволюции основан на генетической изменчивости и процедуре отбора. Данные механизмы задействованы во всех эволюционных алгоритмах. Однако способ обеспечения генетической изменчивости, который можно было бы назвать лучшим, не был выявлен. Исследователи разделяют эти способы на мутацию и рекомбинацию. Успешность эволюции также существенно зависит от применяемых алгоритмом генетических операторов, размера и состава начальной популяции.

Сравнивая эффективность стандартного (традиционного) алгоритма ПЭГ при разных размерах популяции, следует учесть, что время выполнения алгоритма прямо пропорционально размеру популяции. Поэтому для оценки данного фактора в условиях ограниченного времени каждому запуску было отведено фиксированное время работы, чтобы алгоритмы с большим размером популяции имели возможность рассчитать меньшее число поколений. Результаты эксперимента для различного количества особей, приведенные в табл. 1, свидетельствуют о наличии пика эффективности при размере популяции 80–120 особей.

Таблица 1. Эффективность алгоритма при различных размерах популяции

Размер популяции	*sin*		*rosen*		*sigmas*	
	e_b	r_f	e_b	r_f	e_b	r_f
10	0,031	31	0,029	5	0,050	2
20	0,019	39	0,032	7	0,063	0
40	0,022	39	0,029	6	0,047	2
60	0,022	45	0,029	9	0,063	0
80	0,029	45	0,001	18	0,041	5
100	0,009	44	0,001	12	0,043	3
120	0,020	43	0,029	10	0,049	2
140	0,032	37	0,000	12	0,064	0
200	0,031	26	0,027	7	0,066	0
800	0,063	6	0,050	1	0,065	0

Одно из важнейших применений ПЭГ – символьная регрессия, цель которой – поиск выражения, наиболее соответствующего известным заданным значениям с определенной погрешностью. Установка небольшого значения (абсолютного или относительного) допустимой погрешности позволяет обнаружить хорошие решения некоторых математических задач, однако в большинстве случаев задание малой погрешности замедляет процесс эволюции вследствие более строгого отбора особей. Если же задать слишком большую допустимую погрешность, отбору подвергнется множество особей, не являющихся приемлемыми решениями, однако значение ФФ которых будет очень высоким.

Отбор особей в ПЭГ осуществляется пропорциональным оператором рулетки с простым элитизмом: лучшая особь популяции пере-

носится в следующее поколение, шансы остальных прямо пропорциональны значениям их ФФ. Такая форма элитизма гарантирует сохранение лучшего обнаруженного решения. Каждый запуск рулетки выбирает одну особь, соответственно, количество запусков рулетки равно размеру популяции. После отбора новой популяции поочередно выполняются генетические операторы, применяемые к случайным образом выбранным особям. Например, если вероятность оператора составляет 70 процентов, то семь из 10 особей будут им модифицированы. Каждая особь может быть модифицирована сразу несколькими операторами, каждый из которых может быть применен к особи лишь однократно. Это отличает ПЭГ от генетического программирования (ГП), в котором особь не изменяется более чем одним оператором за итерацию. Получаемое таким образом потомство существенно отличается от родительских особей.

Наиболее универсальным оператором отбора с достаточной эффективностью, применяемым в ПЭГ, будет алгоритм рулетки, в котором вероятность дальнейшего участия особи в процессе эволюции напрямую определяется значением ее ФФ, что ведет к сохранению особей только с высоким значением ФФ. Однако если значение ФФ одной из особей популяции в определенный момент значительно превышает значение ФФ остальных, это приводит к застреванию алгоритма в локальном оптимуме и потере множества особей с тенденцией к улучшению с небольшими текущими значениями ФФ.

В работе [3] предложена формула отбора, заимствованная из иммунных алгоритмов и основанная на понятии плотности D особи

$$D(I_k) = \frac{1}{\sum_{i=1}^{N} |f(I_k) - f(I_i)|}, \; k = 1, 2, ..., N$$

и высказано предположение о том, что данная формула отбора гарантирует разнообразие генетического материала в популяции.

Здесь N – количество особей в популяции; f – фитнес-функция.

Значения плотности используются затем для вычисления вероятности отбора

$$P(I_k) = \frac{\sum_{i=1}^{N} D(I_k)}{D(I_k)}, k = 1, 2, ..., N.$$

Таким образом, чем больше особей похожи (по критерию вероятности) на особь I_k, тем меньшую вероятность отбора она имеет. Однако авторами не отмечается тот факт, что при данном подходе особи с принципиально разными синтаксическими деревьями, но равными значениями ФФ будут неотличимы друг от друга.

Сравнение различных подходов к механизму отбора представлено в табл. 2. В данной и в по-

Таблица 2. Эффективность алгоритма с различными операторами отбора и размерами популяции

Схема кодирования	Эксперимент	Число особей	*sin*		*rosen*		*sigmas*	
			e_b	r_f	e_b	r_f	e_b	r_f
1	Рулетка	10	0,033	40	0,029	6	0,076	0
		50	0,029	46	0,000	18	0,038	2
		400	0,067	20	0,034	4	0,047	8
	Отбор по плотности	10	0,032	28	0	6	0,113	0
		50	0,008	40	0,028	6	0,054	2
		400	0,000	100	0,231	0	0,159	0
	Турнирный отбор	10	0,000	100	0,170	0	0,165	0
		50	0,000	100	0,289	0	0,165	0
		400	0,000	100	0,202	0	0,144	0
2	Рулетка	10	0,059	26	0,033	8	0,075	0
		50	0,018	48	0,030	6	0,038	4
		400	0,071	14	0,033	8	0,049	2
	Отбор по плотности	10	0,032	28	0,049	2	0	4
		50	0,035	32	0,033	16	0,039	2
		400	0,000	100	0,279	0	0,165	0
	Турнирный отбор	10	0,584	0	0,289	0	0,165	0
		50	0,000	100	0,160	0	0,142	0
		400	0,000	100	0,253	0	0,161	0
3	Рулетка	10	0,031	38	0,037	4	0,057	2
		50	0,033	48	0,028	6	0,050	6
		400	0,069	22	0,025	16	0,054	4
	Отбор по плотности	10	0,033	24	0,030	4	0	4
		50	0,062	32	0,028	4	0,066	0
		400	0,000	100	0,284	0	0,154	0
	Турнирный отбор	10	0,721	0	0,289	0	0,165	0
		50	0,000	100	0,289	0	0,165	0
		400	0,000	100	0,251	0	0,163	0

следующих таблицах в колонке «Схема кодирования» цифра 1 соответствует традиционно-

му кодированию, 2 – префиксному, 3 – кодированию с наложением. Во всех случаях реализация простого элитизма осуществлялась путем сохранения лучшей особи. Анализ этих результатов позволяет сделать определенные выводы. Во-первых, целесообразно использование популяции в 50 особей: это экспериментально полученное оптимальное значение характерно для всех девяти проведенных экспериментов. Во-вторых, традиционный алгоритм ПЭГ (с кодированием путем обхода графа в ширину) достигает максимума своей производительности при использовании турнирного отбора. В-третьих, для ПЭГ с наложением оптимален отбор по плотности. В-четвертых, наилучшие результаты показывает применение префиксного ПЭГ в комбинации с оператором рулетки.

Итак, рассматривая операторы отбора вне зависимости от способа кодирования, можно сделать вывод о том, что оптимальным будет выбор оператора рулетки, как и было предложено автором традиционного алгоритма ПЭГ [1].

Оператор мутации

Влияние на динамику эволюции. В системах на основе генотипа и фенотипа пространство поиска отделено от пространства решений, что существенно улучшает производительность таких систем. Чем меньше ограничений накладывается на процедуру проецирования генотипа на фенотип и обратно, тем более эффективна система, так как практически любой оператор, включая мутацию, может быть использован для исследования пространства поиска.

Так, например, в системе *DGP* (*Developmental Genetic Programming* – ГП с развитием [4]) в результате трансляции генотипа в фенотип не всегда получается синтаксически правильное дерево, что приводит к дополнительным операциям по удалению непригодных особей. Поэтому мутация в *DGP* не превосходит по показателям операторы кроссовера.

В работе [5] для сравнения эффективности генетических операторов, применяемых в ПЭГ, были сопоставлены динамические данные процесса эволюции: графики значений ФФ лучшей особи с течением поколений (итераций) и графики средних значений ФФ популяции.

В ходе поиска модели, созданной формулой $y(x) = x^4 + x^3 + x^2 + x$, исследована динамика эволюционного процесса с различными значениями вероятности применения оператора мутации (p_{m1}, p_{m2}, p_{m3}, p_{m4}). Полученные значения ФФ приведены на рис. 1.

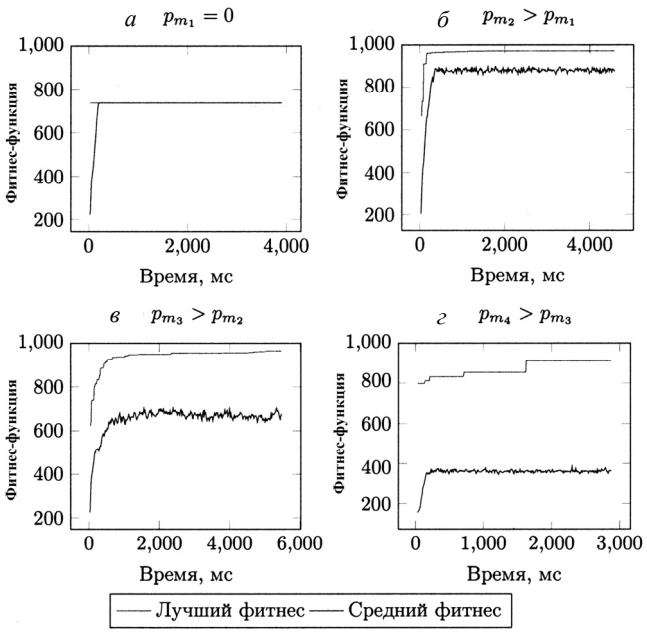

Рис. 1. Динамика эволюции при разных вероятностях мутации

На рис. 1,*а* видно, что график лучших значений ФФ практически совпадает с графиком средних значений ФФ, особенно в поздних поколениях. Такие популяции называются *умеренно инновационными* в силу небольшого различия между особями и медленного процесса эволюции. Вероятность успешного обнаружения решения задачи (отношение запусков, в ходе которых решение было обнаружено, к общему количеству запусков алгоритма), поставленной в эксперименте, составила 16 процентов.

На рис. 1,*б* заметно, что форма графиков совпадает (без учета колебания графика средних значений ФФ), однако они не пересекаются, между ними наблюдается разрыв. Процентное соотношение успешных запусков составило 47 процентов. Такая модель эволюции – здоровая, но слабая.

С повышением вероятности мутации возрастает успешность алгоритма, достигая максимума при $p_{m3} = 0{,}1$, показанного на рис. 1,*в* и соответствующего здоровой и сильной модели.

Для этого случая характерен больший разрыв между графиками, однако тенденция к росту среднего значения ФФ сохраняется.

Наконец, последняя приведенная динамика со средним значением ФФ на постоянном низком уровне с небольшими колебаниями служит примером полностью хаотической системы, в которой, несмотря на элитизм, каждое поколение – по сути случайное.

Исследование операторов рекомбинации в качестве единственных источников, вносящих генетическое разнообразие, выявило процесс усреднения популяции – быстрое сокращение разрыва между графиками лучшего и среднего значений ФФ с последующим их перекрытием. Это означает, что геном всех особей популяции совпадает, и разнообразие устранено, что есть следствием преждевременной сходимости такого подхода к локальному оптимуму.

В начальной популяции, заполненной особями, созданными случайным образом, жизнеспособные решения – событие редкое, особенно при решении сложных задач. Удачным подходом в таком случае является начало процесса эволюции с одной или несколькими особями–основателями [6]. Эффект основателя, описанный Э. Майром, как создание новой популяции из особей–основателей, может быть инициирован дрейфом генов – изменением частоты вариантов генов вследствие случайных процессов и работы операторов отбора. Пример наиболее выраженного случая эффекта основателя – колонизация необитаемой области (создание новой популяции) одной особью.

Это означает, что при определенном значении вероятности мутации можно добиться максимальной эффективности работы всего алгоритма. Кроме того, оператор мутации уменьшает тенденцию популяции к гомогенизации (утрате разнообразия) и устраняет сильную зависимость эффективности алгоритма от размера популяции.

Влияние на производительность. Влияние наиболее вероятных значений среднего количества мутаций в хромосоме на работу алгоритмов с различными способами кодирования синтаксических деревьев показано в табл. 3.

Таблица 3. Эффективность алгоритма при различных вероятностях мутации

Схема кодирования	Число мутаций	*sin*		*rosen*		*sigmas*	
		e_b	r_f	e_b	r_f	e_b	r_f
1	0	0,032	3	0,037	2	0,054	3
	1	0,007	43	0,033	4	0,035	4
	2	0,026	38	0,001	7	0,056	4
	3	0,022	40	0,029	7	0,045	3
	5	0,057	15	0,005	7	0,050	1
2	0	0,086	0	0,058	0	0,056	1
	1	0,030	38	0,011	9	0,044	3
	2	0,009	39	0,029	11	0,039	2
	3	0,032	37	0,029	5	0,044	4
	5	0,063	18	0,005	12	0,043	2
3	0	0,032	3	0,038	2	0,050	2
	1	0,008	35	0,032	6	0,044	4
	2	0,009	39	0,028	10	0,050	3
	3	0,009	34	0,028	9	0,053	4
	5	0,053	14	0,033	7	0,063	0

Отметим общий для всех способов кодирования максимум эффективности при одной–двух мутациях на хромосому. В случае ПЭГ с наложениями максимум эффективности выделить значительно сложнее, однако все же прослеживается падение производительности при количестве мутаций более двух. Поэтому в дальнейших экспериментах будет использоваться значение в две мутации.

Модификации

Наиболее существенным недостатком алгоритма ПЭГ есть его склонность к преждевременной сходимости к локальному оптимуму, потому любые техники, направленные на решение этой проблемы, приводят к существенному росту производительности ПЭГ, что выражается в сокращении времени сходимости, улучшении средней приспособленности решений и повышению вероятности успешного обнаружения оптимального решения.

Для усиления способности алгоритма к поиску в [7] предложен следующий алгоритм динамической мутации *DM–GEP*. При максимальном количестве поколений T процесс эволюции разбивается на три фазы:

• Начальная фаза: поколения от первого до $T_1, 0 < T_1 < T$. Вероятность мутации p_m возрастает каждые два поколения со значения 0,022 до 0,44 с шагом 0,001.

• Метафаза: поколения от T_1 до T_2,$T_1 < T_2 < T$. Вероятность мутации p_m возрастает каждые пять поколений от значения 0,022 до 0,66 с шагом 0,002.

• Анафаза: поколения от T_2 до T. Вероятность мутации p_m уменьшается каждые десять поколений от значения 0,066 до 0,022 с шагом 0,001.

В работе [8] предложено использовать динамическую установку вероятности мутации как самого разрушительного оператора, делая ее индивидуальной для каждой особи и зависимой от ФФ особи. Чем выше ФФ особи, тем более она приспособлена, тем больше внимания требуется уделять локальному поиску, тем меньшая вероятность мутации устанавливается.

$$p_m(I) = \left(1 - f(I)/1000\right) * \left(p_{m_{\max}} - p_{m_{\min}}\right) + p_{m_{\min}},$$

где 1000 – максимальное значение фитнеса; $p_{m_{\min}} = 0,044$ и $p_{m_{\max}} = 0,1$ – нижний и верхний пределы вероятности мутаций соответственно.

Производительность ПЭГ существенно зависит от заданных вероятностей применения генетических операторов. Снизить влияние этих значений на работу алгоритма можно с помощью подхода выбора нового значения, принятого в методе симуляции отжига [9], суть которого состоит в использовании зависящего от времени (номера итерации алгоритма) параметра T, называемого температурой. Чем выше температура в данный момент времени, тем более вероятна замена исходной особи новой, полученной путем применения генетического оператора. Температура системы уменьшается с каждой последующей итерацией, способствуя поиску окрестностей глобального оптимума на первой фазе алгоритма и приводя затем к уточнению его местоположения. Скорость понижения температуры управляется некоторым параметром *a*.

При таком подходе на каждой итерации алгоритма ПЭГ ко всем родительским особям *old* последовательно применяются генетические операторы, каждый из которых порождает новую дочернюю особь *new*. Дочерняя особь занимает место родительской в популяции при выполнении следующего условия:

$$\min\left(1, e^{-\left[\frac{f(old)-f(new)}{T_i}\right]}\right) > random[0,1],$$

где $f(old)$, $f(new)$ – ФФ родительской и дочерней особей соответственно; $random[0, 1]$ – случайная величина в диапазоне [0, 1]; T_i – температура на i-й итерации алгоритма. Данная модификация позволила несколько улучшить эффективность ПЭГ.

Операторы рекомбинации

Эти операторы перемещают последовательные участки генома в пределах хромосомы. Три типа таких участков накладывают различные ограничения на операторы.

Оператор копирования со сдвигом копирует последовательность символов генома в любую позицию головы гена, кроме первой. Ограничение на первую позицию обусловлено тем, что перемещаемая последовательность может начинаться с терминального символа, помещение которого в корень приведет к вырожденному дереву из одного элемента. Ген–источник копируемой последовательности остается неизменным. Элементы головы гена–приемника, начиная с позиции, в которую будет скопирована целевая последовательность, сдвигаются для освобождения места, а вышедшие за пределы головы элементы отбрасываются.

Оператор копирования со сдвигом в корень отличается от предыдущего тем, что целевая последовательность начинается с элемента–функционала и копируется в корень гена–приемника. Оба оператора копирования вносят значительные изменения и поэтому наравне с мутацией отлично подходят для внесения генетического разнообразия, предотвращая застревание в локальном оптимуме и ускоряя поиск хороших решений.

Оператор перестановки генов вносит изменения в результат вычисления декодированного дерева только при условии использования некоммутативной связующей функции в мультигенной хромосоме, такой как «ЕСЛИ, ТО».

Однако преобразующая сила (способность вносить генетическое разнообразие) данного оператора наиболее проявляется в связке с опе-

раторами кроссовера. Например, возможно появление дублируемых генов – явление, выполняющее ощутимую роль в биологии и эволюции.

Все три вида рекомбинации осуществляют обмен генетическим материалом между двумя родительскими хромосомами, порождая две новые особи. Вероятность того, что будет применен один из трех описанных далее операторов, обычно задают величиной 0,7.

В операторе одноточечной рекомбинации две особи скрещиваются вокруг линии, проходящей через случайным образом выбранную позицию хромосомы.

В большинстве случаев полученное потомство проявляет характеристики своих родителей, что делает одноточечную рекомбинацию наиболее часто используемым в ПЭГ оператором, наравне с мутацией.

Отличие двухточечной рекомбинации состоит в том, что обмен участками генома происходит между двумя точками. Поскольку преобразующая сила двухточечной рекомбинации выше, чем у одноточечной, она применяется чаще для решения сложных задач с использованием мультигенных хромосом.

В генной рекомбинации проводится замена генов, занимающих в обеих хромосомах позицию, определяемую случайным образом.

Следует отметить, что использование рекомбинации и/или операторов перемещения как единственного вида применяемых операторов сводится к перемешиванию существующих участков генома и не обеспечивает создание нового генетического материала. Поэтому для решения сложных задач в таком случае может понадобиться популяция большого размера.

Числовые константы в ПЭГ

Создание числовых констант с плавающей запятой необходимо для выполнения символьной регрессии. В стандартном алгоритме ПЭГ для обозначения константы в геноме используется терминальный символ «?», а начальный набор констант представлен доменом D_C [10]. Каждый ген располагает собственным массивом, содержащим используемые им константы и заполняемым на основе домена D_C при генерации начальной популяции. Численное значение константным символам назначается только при экспрессии генов. Для внесения генетического разнообразия был добавлен специальный оператор мутации констант.

Начальный набор возможных числовых констант может быть задан вручную: как перечислением, так и заданием диапазона (успешное решение задач таким методом прямого управления константами возможно только при наличии априорных знаний о решении, позволяющих задать корректный диапазон). Обычно используется метод инициализации значений констант случайными числами.

Динамические константы

В работе [11] предлагается при старте алгоритма ПЭГ на этапе создания новой популяции начальные значения констант в хромосомах отбирать из множества A ={0,1316, 0,2128, 0,3441, 0,5571, 0,9015, 1,4588, 2,3605, 3,8195, 6,1804, 10,0007}, полученного таким образом, чтобы соотношение соседних элементов было золотым сечением. В данной модификации используется следующий оператор мутации каждой константы C_j в геноме выбранной особи:

- Из фиксированного глобального массива первичных констант A случайным образом выбирается константа V.
- Случайным образом выбирается оператор Op из множества {+, —, /, *}.
- $C_j \leftarrow Op\left(C_j, V\right)$.

Плавная и случайная мутация констант

Исследования операторов мутации, основанных на техниках плавной и случайной мутации, описаны в работе [12]. Был задан оператор одноточечной мутации константы: при начальной конфигурации ПЭГ задается отсортированный список возможных констант. Этот оператор изменяет одну символ–константу в гене: при случайной мутации на замену выбирается любой другой элемент списка, при плавной – какой-либо из соседних текущему. Данная операция применяется к каждому гену хромосомы. Локальный поиск должен проводиться в сторону оптимального решения, что означает применение результата мутации только тогда, когда он повышает значение ФФ особи.

Всего было исследовано пять подходов к мутации:

- Плавная мутация лучшей особи в конце расчета поколения.
- Случайная мутация лучшей особи в конце расчета поколения.
- Плавная мутация каждой особи в первые α процента поколений в конце расчета поколения.
- Случайная мутация каждой особи в первые α процента поколений в конце расчета поколения.
- Интервальная случайная мутация: первые g поколений случайной мутации подвергаются все особи популяции в конце расчета поколения.

Применение подхода, при котором мутации подвергается только лучшая особь поколения, никак не повлияло на эффективность алгоритма. Из этого можно сделать вывод, что лучшая особь, полученная стандартным алгоритмом ПЭГ, по определению достаточно хороша и не требует модификаций. Это подтверждает также решающую роль эволюционного процесса ПЭГ (выполнение генетических модификаций и отбор) в поиске оптимальных решений.

Применение мутации к каждой особи популяции значительно повышает значение ФФ популяции, особенно в первых поколениях. Однако с течением времени процентное соотношение особей, улучшенных после мутации, уменьшается. Это происходит вследствие того, что спустя определенное количество поколений формируются группы субоптимальных решений, константы которых уже должным образом настроены, поэтому на данном этапе более значимы операторы, приводящие к большим изменениям.

Применение методов, использующих мутации констант к каждой особи, приводит к росту времени выполнения алгоритма, однако получение более качественных решений не позволяет напрямую сравнить эффективность таких модифицированных алгоритмов со стандартным ПЭГ. Кроме того, применение упомянутых трех техник к различным задачам не позволило выявить лучшую из них.

Плавно-динамические константы

Недостаток описанных операторов мутации численных констант это отсутствие тонкой подстройки коэффициентов особей: степень изменения ничем не ограничена и отсутствуют гарантии поступательного движения в сторону оптимума. Для устранения этого недостатка [13] была построена процедура осуществления направленного процесса эволюции, основанная на комбинации динамического подхода работы с константами ПЭГ и метода иммунных сетей. Создание нового терминала–константы выполняется согласно динамическому подходу: новому узлу (при создании начальной популяции либо после изменения типа узла на терминал–константу оператором мутации) задается значение одно из элементов массива «золотых сечений» A. Оператор плавно-динамической мутации констант изменяет значение в пределах ±10 процентов от текущего:

$$V = V \times \left(1 + random\left(-0{,}1 \ldots 0{,}1\right)\right).$$

Результаты эксперимента, подтверждающие преимущество такого подхода в сравнении с неограниченной мутацией, приведены в табл. 4.

Таблица 4. Эффективность алгоритма при плавной и динамической мутации констант

Вид мутации констант	*sin*		*rosen*		*sigmas*	
	e_b	r_f	e_b	r_f	e_b	r_f
Динамическая	0,030	11	0,028	2	0,058	3
Плавно-динамическая	0,032	35	0,028	10	0,059	2

Фитнес-функция

Для повышения эффективности отбора в [7, 14] предложено введение порогового значения ФФ. Особи, ФФ которых не достигает этого порогового значения, не допускаются к репродукции. Установка порогового значения требует соблюдения баланса между ускорением сходимости и уменьшением разнообразия, что приводит к преждевременной сходимости. Поэтому целесообразно применять динамический порог, обновляющийся на каждой итерации и представляющий собой среднее значение ФФ популяции, умноженное на коэффициент масштабирования. В большинстве случаев в зависимости от задачи используется значение коэффициента в диапазоне [0,15; 1,5], с опти-

мальным значением 1,25. Давление на процесс эволюции, оказываемое порогом отбора, положительно влияет на поиск оптимального решения, существенно сокращая время поиска.

В схему расчета ФФ в работе [15] добавлен коэффициент давления отбора, отражающий количественную степень влияния возрастания среднеквадратичной ошибки решения на падение его приспособленности

$$f(i,g)=1000/(1+k*err(i,g)),$$

где i – индекс особи в популяции; g – поколение; $err(i,g)$ – среднеквадратичная ошибка i-го решения; k – коэффициент давления отбора; 1000 – максимальное значение ФФ.

На рис. 2 приведен график зависимости ФФ от погрешности (*MSE*) для различных значений k, а в табл. 5 – влияние значения этого коэффициента на эффективность модификаций алгоритмов.

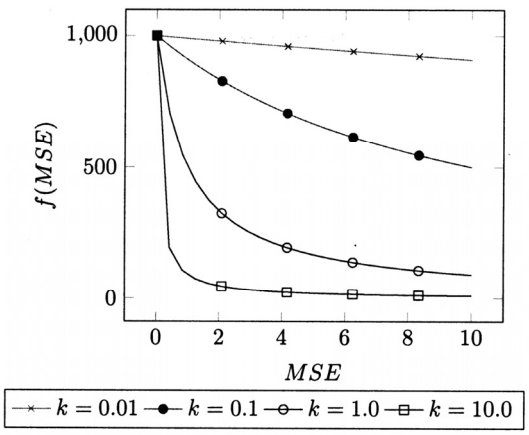

Рис. 2. Графики ФФ для различных значений k

Анализ результатов экспериментов по символьной регрессии на различных наборах данных позволил определить оптимальное значение коэффициента давления отбора $k = 10{,}0$. Данная величина отличается от значения, принятого в исходном алгоритме ПЭГ ($k = 100$).

Частичный подсчет фитнес-функции

Главным недостатком как эволюционных алгоритмов в целом, так и ПЭГ в частности есть низкая скорость их работы (в сравнении со скоростью построения искусственных нейронных сетей, регрессионных моделей). Любое ускорение алгоритма ПЭГ позволяет улучшить качество получаемых моделей, так как дает возможность проведения расчета большего количества поколений и выполнения большего числа независимых запусков за эквивалентное время.

Таблица 5. Эффективность алгоритма при различных коэффициентах давления k

Схема кодирования	k	*sin*		*rosen*		*sigmas*	
		e_b	r_f	e_b	r_f	e_b	r_f
1	0,01	0,080	1	0,039	1	0,044	1
	0,1	0,085	0	0,080	0	0,085	0
	1	0,054	10	0,035	6	0,059	2
	10	0,019	48	0,033	8	0,069	0
	100	0,040	60	0,029	12	0,041	10
2	0,01	0,083	0	0,030	1	0,065	0
	0,1	0,081	0	0,075	0	0,079	0
	1	0,073	6	0,036	2	0,074	0
	10	0,011	52	0,006	12	0,049	6
	100	0,008	58	0,024	6	0,044	4
3	0,01	0,043	3	0,035	1	0,066	0
	0,1	0,072	4	0,039	2	0,069	0
	1	0,056	10	0,028	6	0,085	0
	10	0,022	42	0,029	6	0,048	6
	100	0,019	50	0,037	4	0,039	14

Следует отметить, что наиболее ресурсоемкой частью алгоритма есть вычисление ФФ, так как на вход модели подаются все имеющиеся данные, которые затем обрабатываются в соответствии с программой (формулой, правилом), закодированной моделью. В работе [16] удалось получить универсальный метод ускорения процедуры расчета фитнеса, позволивший значительно улучшить оба показателя эффективности (e_b и r_f) и основанный на следующих соображениях.

В формулу расчета ФФ, как правило, входит СКО модели, а само значение ФФ служит для сравнения эффективности моделей между собой. Как оказалось, для такой оценки не требуется вычисление СКО по полному набору подаваемых на вход данных. Если на определенном этапе текущее значение СКО (либо соответствующая ФФ) превысит некоторый порог *threshold*, дальнейшие расчеты могут быть прерваны. Такой подход позволяет избежать лишних затрат на точное вычисление ФФ плохо приспособленных особей, заменяя его приближенным оценочным значением.

В этом случае весь набор входных данных *DataIn* во избежание последовательностей соседних точек перемешивается и разбивается на G пакетов D_{g_k}:

$$Data\ In = \bigcup_{j=1,G} D_{g_j}$$

После обработки каждого пакета проводится оценка ФФ, и на ее основании выносится решение о прекращении вычислений особи. Условием прекращения служит сравнение значения ФФ по вычисленным первым K группам с пороговым значением *threshold,* которое динамически изменяется в процессе эволюции и поэтому задается некоторой долей (например, 0,7) от среднего значения ФФ популяции предыдущего поколения:

$$f(i,g+1) = f_z\left(i, 0,7\frac{1}{N}\sum_{j=1}^{N} f(i,g)\right),$$

$$f_p(i,K) = f_{MSE}\left(i, \bigcup_{j=1,K} D_{g_j}\right), K \le G,$$

где $f(i,g+1)$, $f(i,g)$ – возвращаемое значение ФФ i-й особи рассчитываемого и предыдущего поколений; $f_z(i,threshold)$ – функция взвешенного значения ФФ; N – размер популяции; K – количество обработанных групп входных данных. Для внесения различия между особями, расчет которых был прерван в различные моменты времени (пороговое значение ошибки было превышено при разном количестве обработанных групп входных данных), в формулу вводится коэффициент штрафа K/G:

$$f_z(i,threshold) =$$

$$= \begin{cases} f_p(i,G) & f_p(i,G) \ge threshold; \\ f_p(i,K)\dfrac{K}{G} & f_p(i,K) < threshold. \end{cases}$$

Таким образом, плохие решения будут использовать меньше ресурсов, в то время как лучшие особи будут вычислены наиболее точно. В табл. 6 приведены результаты сравнительного анализа производительности исходных алгоритмов с модифицированными частичным подсчетом СКО.

Таблица 6. Эффективность алгоритма с различными фитнес-функциями

Схема кодирования	ФФ	*sin*		*rosen*		*sigmas*	
		e_b	r_f	e_b	r_f	e_b	r_f
1	СКО	0,365	0	0,141	0	0,152	0
	Частич. СКО	0,000	100	0,289	0	0,165	0
	R^2	14,858	0	4,361	0	0,093	0
2	СКО	0,000	100	0,275	0	0,138	0
	Частичю СКО	0,000	100	0,285	0	0,165	0
	R^2	20,775	0	30,153	0	0,765	0
3	СКО	0,000	100	0,203	0	0,165	0
	Частич. СКО	0,000	100	0,173	0	0,130	0
	R^2	0,000	100	4,120	0	10608,88	0

Заключение. В ходе анализа существующих методов и средств, направленных на ускорение получения модели алгоритмом ПЭГ и повышение ее качества, а также проведенных экспериментов выявлен ряд ограничений производительности традиционного алгоритма, таких как длительность вычисления ФФ, отсутствие тонкой подстройки числовых констант, влияние размера хромосомы на скорость сходимости, проблемы с поиском сложных моделей. Поэтому дальнейшие исследования должны быть направлены на создание методов и подходов, способствующих преодолению данных ограничений.

1. *Ferreira Candida*. Gene Expression Programming: a New Adaptive Algorithm for Solving Problems // Complex Systems. – 2001. – **13,** № 2. – P. 87–129.
2. *Руденко О.Г.*, *Мирошниченко С.В.*, *Бессонов А.А.* Программирование с экспрессией генов: способы кодирования и создания синтаксических деревьев // УСиМ. – 2015. – № 3. – С. 82–92.
3. *Evolutionary* Algorithm Based on Overlapped Gene Expression. / Jing Peng, Chang-jie Tang, Jing Zhang et al. // ICNC (3). – T. 3612 H3 Lecture Notes in Comp. Sci. – Springer, 2005. – P. 194–204.
4. *A field* guide to genetic programming / R. Poli, W.B. Langdon, N.F. McPhee et al. – Lulu Enterprises, 2008. – 250 p.
5. *Ferreira C.* Mutation, Transposition, and Recombination: An Analysis of the Evolutionary Dynamics // 4th Int. Workshop on Frontiers in Evolutionary Algorithms. – North Carolina, USA, 2002. – http://www.gene- expression-programming.com/webpapers/ferreira-fea02.pdf
6. *Ferreira C.* Analyzing the Founder Effect in Simulated Evolutionary Processes Using Gene Expression Programming // Soft Comp. Syst. Design, Management and Applications. – IOS Press, 2002. – P. 153–162.
7. *Teodorescu L.*, *Huang Z.* Enhanced Gene Expression Programming for signal-background discrimination in

particle physics // Proc. of XII Advanced Comp. and Analysis Techniques in Physics Research. – 2008.
8. *Tang C.*, *Duan L.*, *Peng J.* The Strategies to Improve Performance of Function Mining By Gene Expression Programming: Genetic Modifying, Overlapped Gene, Backtracking and Adaptive Mutation // Proc. of the 17th Data Engin. Workshop. – 2006.
9. *Siwei J.*, *Zhihua C.*, *Dan Z.* Gene expression programming based on simulated annealing // Proc. Int. Conf. on Wireless Communications, Networking and Mobile Computing. – T. 2. – IEEE, 2005. – P. 1264–1267.
10 *Function* finding and a creation of numerical constants in gene expression programming / C. Ferreria, J.M. Benitez, O. Cordon et al. // Advances in Soft Computing, Engineering Design and Manufacturing. – Berlin: Springer-Verlag, 2003. – P. 257–266.
11. *Peng J. Tang C.*, *Yang D.* Function Finding and Constants Creation Method in Evolutionary Algorithm Based on Overlapped Gene Expression // Proc. of the Third Int. Conf. on Natural Computation, **04**. – ICNC '07. – Washington, DC, USA: IEEE Comp. Society, 2007. – P. 18–22.
12. *Investigation* of Constant Creation Techniques in the Context of Gene Expression Programming / X. Li, C. Zhou, P.C. Nelson et al. // Late Breaking Papers at the 2004 Genetic and Evolutionary Computation Conf. – Seattle, Washington, USA, 2004. – http://www.cs.bham.ac.uk/~wbl/biblio/gecco2004/ /LBP023.pdf
13. *Мирошниченко С.В.* Символьная регрессия с помощью программирования с экспрессией генов // Сб. материалов форума: 17-й Международный молодежный форум «Радиоэлектроника и молодежь в XXI веке». – Харьков: ХНУРЭ, 2014. – С.184–185.
14. *Руденко О.Г.*, *Мирошниченко С.В.* Построение моделей нелинейных объектов на основе программирования с экспрессией генов // Вычислительный интеллект (результаты, проблемы, перспективы): Материалы 2-й Междунар. науч.-техн. конф. – Черкассы: Маклаут, 2013. – С. 118–119.
15. *Lopes H.S.*, *Weinert W.R.* EGIPSYS: an Enhanced Gene Expression Programming Approach for Symbolic Regression Problems // Int. J. of Applied Mathematics and Comp. Sci. – 2004. – **14**, N 3. – P.375–384.
16. *Руденко О.Г.*, *Мирошниченко С.В.* Сжатие изображений на основе модифицированного алгоритма программирования с экспрессией генов // Материалы первой международной научно-технической конференции: Проблемы информатизации. – Черкассы: ЧДТУ; Киев: ДУТ; Тольятти: ТДУ; Полтава: ПНТУ, 2013. –31 с.

Поступила 15.06.2015
Тел. для справок: +38 057 705-0862, 702-1354 (Харьков)
E-mail: *o.bezsonov@gmail.com*, *S.Miroshnichenko@gmail.com*

UDC 519.71

Rudenko O.G., Miroshnichenko S.V., Bezsonov A.A.

Gene Expression Programming: Genetic Operators

The existing methods and tools to accelerate the process of obtaining the model and improve its quality by gene expression programming (GEP) algorithm are analyzed. The following operators are utilized by GEP: replications, mutations, shifted copy, copy with a shift in root, genes' rearrangement, one- and two-point recombination, genetic recombination. The aim of this study is to investigate the influence of genetic operators' selection on the properties of the GEP algorithm.

The main drawback of both evolutionary algorithms in general, and GEP algorithm in particular, is a low speed of operation (compared with the constructing speed of artificial neural networks, regression models, etc.). Any acceleration of GEP algorithm may improve the quality of the model, as it allows the calculation of the greater number of generations and performing a larger number of independent to be run for the equivalent time.

To compare the genetic operators the following models are used: sinusoid (sin), Rosenbrock function (rosen) and the sum of the four sigmoid (sigmas). The experiments identified a number of performance limitations of the traditional algorithm, such as long duration of the FF calculating, lack of the numerical constants' fine-tuning, the impact of the chromosome's size on the rate of convergence, problems with the search of the complex models. It is recommended to direct the further research on the development of the methods and approaches that contribute to overcome the identified constraints.

●