Ю.Р. ЦОЙ, В.Г. СПИЦЫН
Томский политехнический университет
qai@mail.ru, spitsyn@ce.cctpu.edu.ru

ПРИМЕНЕНИЕ ГЕНЕТИЧЕСКОГО АЛГОРИТМА ДЛЯ РЕШЕНИЯ ЗАДАЧИ АДАПТИВНОГО НЕЙРОУПРАВЛЕНИЯ

Аннотация

В работе описывается разработанный генетический алгоритм для одновременной настройки весов и структуры искусственной нейронной сети (NEvA). Отличительными особенностями разработанного подхода являются гибкий и эффективный эволюционный поиск, сбалансированность структур получаемых ИНС, а также возможность удаления малоинформативных входных признаков. Рассматривается возможность применения разработанного алгоритма для решения задачи нейроуправления. Проводится сравнение полученных результатов с результатами работы существующих алгоритмов.

Введение

Генетический алгоритм (ГА) является методом оптимизации, использующим эволюционные принципы для поиска решения [1]. Благодаря механизмам наследственности, изменчивости и естественного отбора, ГА обладают богатыми способностями к адаптации. Искусственные нейронные сети (ИНС) часто используются для решения задач классификации, аппроксимации и управления [2].

Использование генетических алгоритмов для настройки ИНС называется нейроэволюцией или нейрогенезисом, а сами алгоритмы соответственно – нейроэволюционными [3]. Задача одновременной настройки весов и структуры нейросети формально не решена, и поэтому является сложной переборной проблемой. В данной работе описывается разработанный алгоритм NEvA – NeuroEvolutionary Algorithm, особенностями которого являются гибкий эволюционный поиск, сбалансированность структур получаемых ИНС, а также возможность удаления малоинформативных входных признаков.

Описание алгоритма

В разработанном алгоритме рассматривается популяция нейронных сетей. Генетическая информация каждой особи содержит информацию о связях сети и их весах. Пример генотипа приведен на рис.1. Для записи

информации о связи используются данные об индексах начального и конечного нейронов и значение веса. Данный способ соответствует разновидности прямого метода кодирования информации о структуре сети [4].

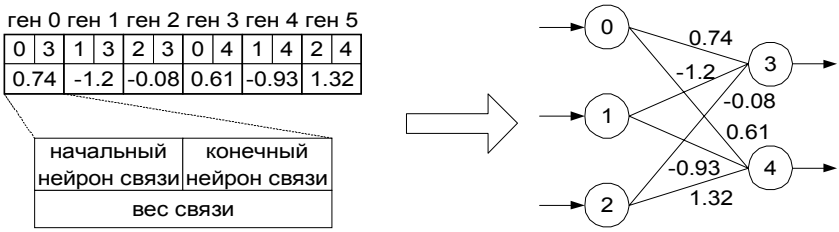

Рис.1. Представление ИНС в геноме особи

При оценке особей составлялись соответствующие им нейронные сети, и вычислялась ошибка выхода по формуле:

$$E = \frac{1}{2N_S}\sum_{i}^{N_S}\sum_{j}^{N_O}(o_{ij} - y_{ij})^2, \qquad (1)$$

где o_{ij} и y_{ij} – соответственно реальное и требуемое значения сигнала j-го выходного нейрона на i-м обучающем наборе данных, N_S – число наборов в обучающей выборке, N_O – число выходов сети.

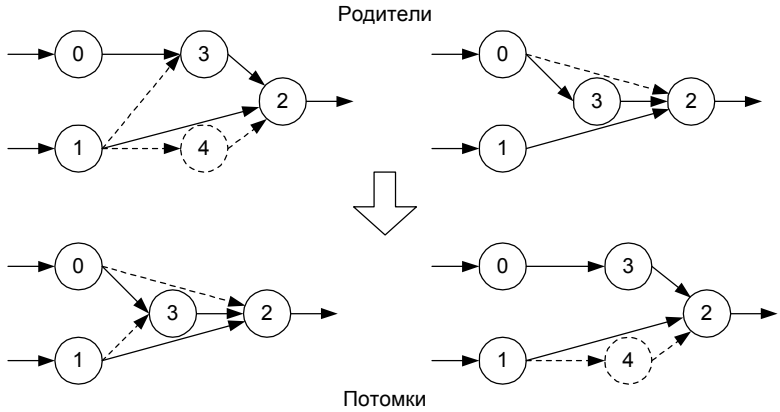

Рис.2. Пример скрещивания

Для скрещивания отбирались особи, ошибки которых меньше средней. В процессе репродукции получаются два потомка, которые заимствуют общие нейроны и связи обоих родителей, при этом веса родительских

особей рекомбинируют с использованием 2-точечного кроссовера. Различные нейроны и связи «разыгрываются» между потомками случайным образом. Пример скрещивания представлен на рис. 2. Сплошными линиями обозначены общие нейроны и связи, пунктирными – «разыгрываемые».

Механизм мутации предусматривает добавление и удаление нейронов и связей, а также изменение случайной связи на произвольную величину. Выбор типа мутации производится в зависимости от количества нейронов в сети и связей между ними в соответствии со следующими эвристиками:

$$f_C = \frac{N_C}{2^{FB-1}\left[N_N\left(N_N-1\right)-N_I\left(N_I-1\right)-\left(1-FB\right)N_O\left(N_O-1\right)\right]}, \qquad (2)$$

$$f_N = \frac{N_I + N_O}{N_N}, \qquad (3)$$

где N_C – количество связей в сети, N_I, N_O, N_N – соответственно количество входных, выходных нейронов и общее число нейронов в сети, FB – флаг, обозначающий, разрешено появление обратных связей (FB=1) или нет (FB=0). Необходимо отметить, что связи от скрытых нейронов к выходным могут появляться в любом случае. Коэффициент f_C необходим для оценивания степени «связанности» нейронов в сети, т.е. чем большее число возможных связей реализовано, тем больше значение коэффициента. Использование второго коэффициента f_N основано на эвристическом предположении, что чем больше количество входных и выходных сигналов нейронной сети, тем, возможно, более сложная структура необходима для решения задачи.

Для выбора вида мутации использовались коэффициенты, вычисленные по формулам (2) и (3) и возведенные в квадрат, обозначим их как F_C и F_N. Схематично алгоритм выбора вида мутации представлен на рис. 3. Здесь СВ – случайная величина, N_H – количество скрытых нейронов в мутирующей сети.

Условно весь алгоритм можно разделить на две «ветви» по первому условному переходу:

1. Ветвь увеличения f_C (усложнение структуры сети).
2. Ветвь уменьшения f_C (упрощение структуры сети).

Поскольку удаление нейрона может привести как к уменьшению, так и к увеличению f_C, в зависимости от количества связей нейрона, то этот вариант присутствует в обоих ветвях. Таким образом, основным фактором для регуляции структуры получаемых ИНС является степень их «связанности».

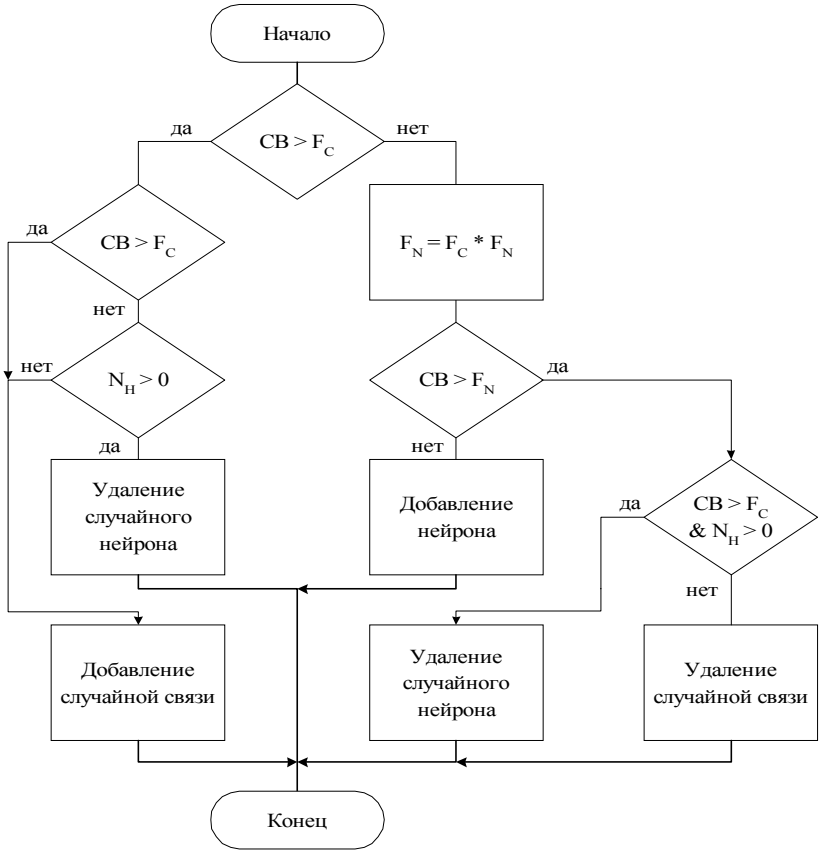

Рис.3. Схема выбора вида мутации

Умножение F_N на F_C необходимо для того, чтобы изменять количество нейронов адекватно топологии сети, т.к. при добавлении (удалении) нейронов необходима информация о целесообразности изменений. Эта информация может быть косвенно получена, исходя из значения характеристики f_C.

Такой подход, с одной стороны, не ограничивает «сверху» количество скрытых нейронов, а с другой – препятствует быстрому увеличению чис-

ла нейронов в сети, т.к. добавление каждого нового нейрона в сеть будет происходить с меньшей вероятностью.

В начальной популяции все особи представляют сети без скрытых нейронов, все входные нейроны которых соединены с выходными. Веса сетей являются случайными величинами из диапазона [-0,5; 0,5]. В процессе эволюции сети растут и усложняются. Такой подход аналогичен подходу, использованному в алгоритме NEAT [5], и представляется более рациональным, чем развитие популяции сетей с изначально случайной архитектурой, поскольку позволяет более последовательно оценивать нейросетевые структуры. Использовались нейроны без смещений с лог-сигмоидной функцией активации.

Описание задачи и результаты работы

Рассмотрим возможность использования представленного алгоритма для решения задачи адаптивного управления. В качестве задачи была выбрана проблема «Перевернутый маятник» («Inverted pendulum») [6]. Существует несколько разновидностей этой проблемы. В самой простой имеется тележка, на которой установлен шест. В начальный момент времени он отклоняется от положения равновесия, и задача заключается в том, чтобы, перемещая тележку вперед и назад с определенным фиксированным усилием, удержать шест. В более сложном варианте воздействие на тележку может непрерывно изменяться в некоторых пределах (как правило [-10N; 10N]). Удаление входов с информацией о скорости движения маятников и тележки является дальнейшим усложнением проблемы. Также существует постановка задачи для двумерной среды, когда тележка может перемещаться по двум независимым направлениям [7]. Подробное описание математической модели задачи и приведено в [8], условия проведения экспериментов – в [5].

Были использованы следующие варианты задачи:

1. один инвертированный маятник, сила воздействия на тележку фиксированная и равна -10Н или 10Н. Эксперименты проводились для сетей с одним выходом с лог-сигмоидной функцией активации.
2. два инвертированных маятника с полной информацией об их движении (не требует поиска сети с обратными связями).

Каждый алгоритм запускался по 50 раз. Результаты экспериментов приведены в табл.1. Данные для алгоритмов GENITOR, SANE и ESP взяты из [8]. Предлагаемый алгоритм обозначен как NEvA. Использована популяция из 16 особей, вероятность мутации равнялась 0,2, вероятность

кроссовера – 0,7. Аббревиатура СКО в табл. 1 означает среднеквадратичное отклонение результатов работы алгоритмов.

Табл.1. Результаты решения проблемы с 1 маятником

Алгоритм	Количество попыток				Число неудач
	Среднее	Лучшее	Худшее	СКО	
GENITOR	1846	272	7052	1396	0
SANE	535	70	1910	329	0
ESP	285	11	1326	277	0
NEvA	451	37	3872	720	0

По полученным средним результатам разработанный алгоритм проигрывает только алгоритму ESP. Стоит отметить, что алгоритм ESP не является нейроэволюционным в строгом смысле, т.к. не настраивает топологию ИНС (в этом и следующем эксперименте использовались полносвязные сети). Информация о структуре использованных для решения сетей приведены в табл. 2. Данные для алгоритмов GENITOR, SANE, ESP приведены на основании работ [8, 9]. “+/-” для алгоритма SANE означает, что количество скрытых нейронов и связей фиксировано, но алгоритм настраивает их расположение.

Табл.2. Данные о сетях для проблемы с 1 маятником

Алгоритм	Среднее число скрытых нейронов	Среднее число связей	Настройка структуры сети
GENITOR	5	35	-
SANE	8	40	+/-
ESP	5	30	-
NEvA	0.58	5.22	+

В силу того, что во всех алгоритмах кроме NEvA, топология нейронных сетей предлагалась исследователями, число нейронов и число связей сравнительно велико. Рассматриваемый алгоритм настраивает структуру ИНС в процессе работы, что дает достаточно ощутимый выигрыш.

Результаты решения задачи с двумя маятниками и информация об использованных сетях представлены соответственно в табл. 3 и 4. Результаты для эволюционного программирования взяты из работы [10], для алгоритмов SANE, ESP из [8], для алгоритма NEAT из публикации [5]. Скорость работы алгоритма NEvA сопоставима с результатами алгоритма

NEAT. При этом полученные сети имеют меньшее количество скрытых нейронов: от 0 до 2 против от 0 до 4 у NEAT [5]. Количество связей для NEAT дано предположительно и посчитано на основе информации о числе скрытых нейронов. В скобках в результатах работы алгоритма NEvA указано среднее количество нейронов и связей по итогам 50 запусков.

Табл.3. Результаты решения проблемы с 2 маятниками

Алгоритм	Среднее количество попыток	Размер популяции	Число неудач
Эволюционное программирование	307200	2048	0
SANE	12600	200	0
ESP	3800	200	0
NEAT	3578	150	0
NEvA	3777	59	0

Табл.4. Данные о сетях для проблемы с 2 маятниками

Алгоритм	Среднее число скрытых нейронов	Среднее число связей	Число неудач
ESP	5	35	0
NEAT	0 – 4	6 – 15	0
NEvA	0 – 2 (1,4)	4 – 16 (7.24)	0

Одна из особенностей разработанного алгоритма заключается в том, что в результате мутации возможно удаление межнейронных связей. Таким образом, можно удалять малоинформативные входы. Например, для успешного балансирования маятников необязательно иметь информацию о скорости передвижения тележки или о ее координатах. В результате возможны нейросетевые решения с 1 изолированным входом из 2, на которые подается информация о тележке. Такие решения были найдены:

- в 2 случаях для балансирования одного маятника для сетей с одним выходом;
- в 8 случаях для задачи с двумя маятниками, при этом 1 раз была получена сеть с 2-мя изолированными входами с информацией о скорости перемещения тележки.

Результаты предварительных исследований по способности к удалению малоинформативных входных сигналов показывают, что для сети с одним выходом и 100 входами, из которых информативным является только 1, в результате обучения возможно удаление от 0 до 8 неинформа-

тивных входов (в среднем 2). Данные получены по 10 запускам для популяции из 50 особей с ограничением эволюции в 3000 поколений и вероятностью мутации 0,25.

Заключение

Предлагаемый алгоритм NEvA успешно справился с поставленной задачей. Результаты работы сопоставимы с результатами известных алгоритмов ESP и NEAT.

В дальнейшем планируется рассмотреть самый сложный вариант проблемы «Инвертированный маятник», когда информация о скоростях движения шестов и тележки не подается на входы управляющей нейронной сети. При этом для решения задачи необходимо наличие обратных связей в структуре нейронной сети.

Список литературы

1. Holland J. Adaptation in Natural and Artificial Systems. – The University of Michigan Press, Ann Arbor, 1975.
2. Круглов В.В., Борисов В.В. Искусственные нейронные сети: Теория и практика. – 2-е изд. стереотип. – М.: Горячая линия-Телеком, 2002. – 382 с.: ил.
3. Whitley D. Genetic Algorithms and Neural Networks, Genetic Algorithms in Engineering and Computer Science. / Eds. Winter, Periaux, Galan and Cuesta. – John Wiley, 1995. – P. 203-216.
4. Balakrishan K., Honavar V. Evolutionary Design of Neural Architectures – A Preliminary Taxonomy and Guide to Literature. – Iowa State University, CS TR #95-01, 1995.
5. Stanley K., Miikkulainen R. Evolving Neural Topologies through Augmenting Topologies // Evolutionary Computation. The MIT Press. – 2002 – №10(2). – P. 99-127.
6. Wieland, A. Evolving neural network controllers for unstable systems. // Proceedings of the International Joint Conference on Neural Networks. – Piscataway, New Jersey, 1991. – P. 667–673.
7. Faustino Gomez and Risto Miikkulainen. 2-D Pole Balancing Benchmark with Recurrent Evolutionary Networks. // Proceedings of the International Conference on Artificial Neural Networks (ICANN–98). – Skovde, Sweden, 1998.
8. Gomez F., Miikkulainen R. Incremental evolution of complex general behavior. // Adaptive Behavior. –1997. – №5. – P. 317–342.
9. Moriarty D.E., Miikkulainen R. Efficient Reinforcement Learning through Symbiotic Evolution. // Machine Learning. – 1996. – № 22. – P. 11-33.
10. Saravanan N., Fogel D. B. Evolving neural control systems. // IEEE Expert. – 1995. – P. 23-27.