УДК 004.82 3

И.А. Ходашинский, И.В. Горбунов, П.А. Дудин

Алгоритмы муравьиной и пчелиной колонии для обучения нечетких систем*

Для обучения нечетких систем предложены алгоритмы пчелиной колонии и три алгоритма муравьиной колонии: дискретный, непрерывный и прямой.
Ключевые слова: нечеткие системы, метаэвристики, алгоритм пчелиной колонии, алгоритм муравьиной колонии.

Постановка задачи

Пусть имеется объект исследования, заданный своей таблицей наблюдений. Проблема исследования обусловлена невозможностью построения аналитической модели изучаемого объекта, либо слишком большой сложностью такой модели, либо отсутствием достаточного опыта для построения экспертных систем, либо недостаточностью экспериментальных данных для статистического моделирования. Решением проблемы может быть переход от аналитических или статистических моделей к нечетким.

Рассматривается нечеткая система типа синглтон, i-е правило в которой имеет следующий вид:

$$\text{IF } x_1=A_{1i} \text{ AND } x_2=A_{2i} \text{ AND ... AND } x_n=A_{ni} \text{ THEN } y = r_i,$$

где A_{ij} – лингвистический терм, которым оценивается переменная x_i; r_i – действительное число, которым оценивается выход y.

Нечеткая система осуществляет отображение $f:\Re^n \to \Re$:

$$f(\mathbf{x})=\sum_{i=1}^{R}\mu_{A1i}(x_1)\cdot\mu_{A2i}(x_2)\cdot\ldots\cdot\mu_{Ani}(x_n)\cdot r_i \Big/ \sum_{i=1}^{R}\mu_{A1i}(x_1)\cdot\mu_{A2i}(x_2)\cdot\ldots\cdot\mu_{Ani}(x_n),$$

где $\mathbf{x}$ – входной вектор, R – число правил; n – количество входных переменных; μ_{Aij} – функция принадлежности, определяемая набором своих параметров, например: треугольная – тремя параметрами, трапециевидная – четырьмя, гауссова и параболическая – двумя.

Нечеткая система может быть представлена как

$$y = f(\boldsymbol{x}, \boldsymbol{\theta}),$$

где $\boldsymbol{\theta} = \|\theta_1,\ldots,\theta_N\|$ – вектор параметров; $N = n$ (число параметров, описывающих одну функцию принадлежности) (число термов, описывающих одну входную лингвистическую переменную, – является задаваемым параметром нечеткой системы); y – скалярный выход системы.

Пусть дано множество обучающих данных $\{(\boldsymbol{x}_p;\ t_p), p = 1,\ldots,m\}$, тогда среднеквадратическая функция ошибки, которую необходимо минимизировать, будет иметь вид

$$E(\theta)=\sum_{p=1}^{m}(t_p - f(\mathbf{x}_p,\theta))^2.$$

Для решения проблемы минимизации предлагается использовать алгоритмы муравьиной колонии, роящихся частиц и пчелиной колонии, а также построенные на их основе гибридные алгоритмы.

Алгоритмы муравьиной колонии

Дискретный алгоритм. Идея алгоритма основана на способности муравьев находить кратчайший путь между источником пищи и муравейником без использования визуальной информации [1]. Способность эта обусловлена выделением феромона во время их движения и использованием этого феромона в качестве маркера.

Алгоритм, предложенный в работе [1], реализует процедуру дискретной оптимизации, в то время как параметры $\boldsymbol{\theta}$ меняются непрерывно. Переход от непрерывной оптимизации к дискретной осуществляется посредством построения полного ориентированного графа поиска решения, количество вершин в котором определяется точностью нахождения

* Работа выполнена при финансовой поддержке РФФИ (проект № 09-07-99008).

значений параметров. Из каждой вершины выходят дуги с равномерно распределенными значениями нормированных параметров. Во все вершины графа равномерно распределяются муравьи. Цель муравья в задаче оценки параметров нечеткой системы – посетить столько вершин, сколькими параметрами задается функция принадлежности. Пометки дуг, по которым прошел муравей, будут являться найденным им решением [2].

Количество феромона, наносимого на дуги пропорционально качеству решения: чем меньше ошибка вывода нечеткой системы, выполненного на выбранных параметрах, тем больше феромона наносится на дуги.

На каждой итерации происходит увеличение количества феромона на каждой дуге и его испарение.

Увеличить точность вычисления параметров можно несколькими путями. Первое самое очевидное решение – это увеличение точности за счет увеличения количества вершин орграфа. Но увеличение размерности графа на порядок увеличивает время вычисления на два порядка.

Для устранения этой проблемы предлагается другой способ – поэтапное увеличение точности. Сначала алгоритм работает с точностью 0,1 и находит оптимальные значения параметров функции принадлежности. Затем найденные параметры ограничиваются слева и справа значением 0,1 и в этих интервалах берутся значения с шагом 0,01 в качестве пометок нового орграфа. И так далее в зависимости от требуемой точности. Это позволяет существенно уменьшить время вычислений.

Еще одним способом увеличения точности является применение градиентного алгоритма на втором этапе оптимизации после завершения работы алгоритма с заданной точностью вычисления [2].

Непрерывный алгоритм муравьиной колонии. В классическом алгоритме муравьиной колонии при выборе очередной дуги муравей руководствуется дискретным распределением вероятности. В случае непрерывного алгоритма выбор, который делает муравей, не ограничен конечным множеством, здесь дискретное распределение заменяется непрерывным, заданным своей функцией плотности [3].

В непрерывном алгоритме для описания используется функция плотности вероятности с гауссовым ядром. Под гауссовым ядром $G^i(x)$ понимается функция, основанная на взвешенной сумме нескольких одномерных гауссовых функций $g_l^i(x)$:

$$G^i(x)=\sum_{l=1}^{k}\omega_l g_l^i(x)=\sum_{l=1}^{k}\omega_l\frac{1}{\sigma_l^i\sqrt{2\pi}}e^{-\frac{(x-\mu_l^i)^2}{2{\sigma_l^i}^2}}.$$

Каждому i-му параметру функции принадлежности соответствует свое гауссово ядро, $i = 1,\ldots,N, N$ – число настраиваемых параметров нечеткой модели. Каждая функция $G^i(x)$ описывается тремя векторами: $\boldsymbol{\omega}$ – вектор весов, связанных с индивидуальными гауссовыми функциями; $\boldsymbol{\mu}_i$ – вектор математических ожиданий; $\boldsymbol{\sigma}^i$ – вектор среднеквадратичных отклонений. Количество элементов всех этих векторов равно числу функций Гаусса, составляющих гауссово ядро.

В непрерывном алгоритме вводится понятие архива решений. Архив решений представлен таблицей, в которой k строк. Каждая строка представляет собой найденное муравьем решение $\boldsymbol{s}_l=\| s_l^1, s_l^2,\ldots, s_l^N \|$, ошибку E и вес решения $\boldsymbol{\omega}_l$. Решения упорядочены в архиве согласно их качеству. Вес $\boldsymbol{\omega}_l$ решения s_l вычисляется согласно следующей формуле:

$$_l=\frac{1}{qk\sqrt{2\pi}}e^{-\frac{(l-1)^2}{2q^2k^2}},$$

где q – эмпирический параметр алгоритма.

Параметры гауссовой функции $g_l^i(x)$, определяются следующим образом:

$$\mu_l^i=s_l^i,\qquad \sigma_l^i=\xi\sum_{e=1}^{k}\frac{\left|s_e^i-s_l^i\right|}{k-1}.$$

Введение параметра $\xi > 0$ имеет эффект, подобный норме испарения феромона в дискретном алгоритме.

При добавлении нового решения в архив решений худшее решение удаляется из архива. Этот процесс аналогичен процессу испарения феромона.

Алгоритм

Шаг 1. Задать начальные параметры.

Шаг 2. Сгенерировать популяцию муравьев в колониях.

Шаг 3. Для всех архивов сгенерировать несколько случайных решений с последующим оцениванием и ранжированием.

Шаг 4. Найти значения вектора весов. Сделать текущим первого муравья первой колонии.

Шаг 5. Для текущего муравья текущей колонии вычислить номер l, используемой функции Гаусса. Определить параметры функций Гаусса. Сгенерировать N случайных величин $\{x_1, x_2, \dots, x_N\}$ на основе полученных функций.

Шаг 6. Найти ошибку вывода нечеткой системы при параметрах $\{x_1, x_2, \dots, x_N\}$, если ошибка меньше текущей, то сохранить новые параметры.

Шаг 7. Добавить в архив решений новое решение, проранжировать архив, удалить из архива худшее решение.

Шаг 8. Если в текущей колонии имеется следующий муравей, то сделать его текущим и перейти к шагу 5, иначе перейти на шаг 9.

Шаг 9. Если имеется следующая колония, то сделать текущим первого муравья в этой колонии и перейти на шаг 5, иначе перейти на шаг 10.

Шаг 10. Если условие окончание работы алгоритма выполнено, то закончить, иначе сделать текущим первого муравья первой колонии и перейти к шагу 5.

Условием окончания работы алгоритма является достижение заданного числа итераций, либо получение ошибки, меньше заданной.

Прямой алгоритм муравьиной колонии. В прямом алгоритме [4] муравей отвечает за вычисление значений закрепленного за ним параметра, поэтому муравьев в алгоритме столько, сколько параметров нечеткой модели. Каждый i-й муравей создает свое решение, генерируя нормально распределенное действительное число $N(\mu_i, \sigma_i)$. В алгоритме используются два вида ферамонов: первый связан с центрами нормальных распределений $\boldsymbol{\mu} = \|\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_N\|$, второй – с разбросом $\boldsymbol{\sigma} = \|\sigma_1, \dots, \sigma_N\|$. Количество ферамона определяет значения параметров $\boldsymbol{\mu}$ и $\boldsymbol{\sigma}$. Для каждого параметра θ_j задан интервал изменения $[a_j, b_j]$, где b_j и a_j – верхняя и нижняя граница параметра θ_j.

В качестве начальных значений для параметров $\boldsymbol{\mu}$ используются заданные случайным образом или другим способом значения параметров $\boldsymbol{\theta}$. Начальные значения параметров $\boldsymbol{\sigma}$ вычисляются по следующей формуле:

$$\boldsymbol{\sigma}_i = \frac{b_i - a_i}{2}.$$

После того как муравьи нашли решения, определяется испарение ферамона. Для текущей t-й итерации испарение определяется следующим образом:

$$\boldsymbol{\mu}(t) = (1 - \rho)\, \boldsymbol{\mu}(t-1),$$
$$\boldsymbol{\sigma}(t) = (1 - \rho)\, \boldsymbol{\sigma}(t-1),$$

где ρ – эмпирический коэффициент испарения феромона, заданный на интервале $[0,1]$.

Далее происходит нанесение феромона:

$$\boldsymbol{\mu}(t) = \boldsymbol{\mu}(t) + \rho\boldsymbol{\theta}(t),$$
$$\boldsymbol{\sigma}(t) = \boldsymbol{\sigma}(t) + \rho|\boldsymbol{\theta}(t) - \boldsymbol{\mu}(t)|,$$

где $\boldsymbol{\theta}(t)$ – решение, найденное муравьиной колонией на текущей итерации, оно совпадает с глобальным лучшим решением.

Особенностью прямого алгоритма муравьиной колонии является включение в него простейшего локального поиска, состоящего из двух этапов: на первом этапе значение параметра θ_j увеличивается с определенным шагом до значения $\theta_j + d_j$, на втором этапе значение параметра уменьшается с определенным шагом до значения $\theta_j - d_j$. Значение d_j определяется по формуле

$$d_j = \sigma_j \,\mathrm{rand},$$

где rand – случайное равномерно распределенное число в интервале $[0, 1]$. Шаг вычисляется по следующей формуле:

$$st_j = d_j / K,$$

где K – целое число, отвечающее за вычисление значения шага.

В результате локального поиска определяется новый вектор параметров $\boldsymbol{\theta}$. Значения этих параметров передаются в нечеткую систему в качестве новых значений параметров. Вычисляются ошибка и лучшее решение текущего шага. Глобальное лучшее решение запоминается.

Для решения задачи оценки параметров необходима проверка изменения параметра θ_j на ограничения, накладываемые нечеткой системой. К таким ограничениям относятся покрытие всей области определения термов, описывающих входную переменную, и упорядоченность пиков функций принадлежности.

Для преодоления локальных минимумов в алгоритме используется обновление параметров **σ**. С этой целью введен параметр конвергенции, вычисляемый по следующей формуле:

$$cf = \frac{\sum_{j=1}^{N} \frac{2\sigma_j}{b_j - a_j}}{N}.$$

Когда алгоритм приближается к локальному минимуму, коэффициент конвергенции cf приближается к 0. Как только коэффициент конвергенции становится меньше критического значения cf_r, то вектор **σ** возвращается в начальное состояние.

Условием окончания работы алгоритма может быть выполнение заданного числа итераций или достижение ошибки меньше заданной.

Алгоритм пчелиной колонии

Идея алгоритма взята из модели поведения пчел при поиске мест, где можно добыть как можно больше нектара. Сначала из улья вылетает в случайном направлении какое-то количество пчел-разведчиков, которые пытаются отыскать участки, где есть нектар. Через некоторое время пчелы возвращаются в улей и особым образом сообщают остальным, где и сколько они нашли нектара. После этого на найденные участки отправляются рабочие пчелы – фуражиры. Чем больше на данном участке предполагается найти нектара, тем больше фуражиров летит в этом направлении, а разведчики улетают искать другие участки, после чего процесс повторяется [5].

Алгоритм

Вход: Начальное количество разведчиков S, массив фуражиров W, максимальное число итераций $IterM$, начальная температура $T0$, значение α.

Выход: Оптимальные параметры антецедентов нечеткой системы.

Шаг 1. $Iter$:=1, BS:=S, T:= $T0$, W:= ∅.

Шаг 2. Создание случайных BS векторов-решений по каждому из разведчиков, вычисление для каждого вектора ошибки решения.

Шаг 3. Определение лучшего решения *best* в $\{BS \cup W\}$.

Шаг 4. ЦИКЛ по j от 1 до BS ВЫПОЛНИТЬ

Если exp(-|ошибка.j − ошибка.*best* |/ T) > *rand*,

то включить j-е решение в массив фуражиров W.

Шаг 5. Формирование новых решений NW на базе W.

ЦИКЛ по j от 1 до $|W|$ ВЫПОЛНИТЬ

$$NW.\mathbf{x} := W.\mathbf{x} \pm \text{rand}\ (W.\mathbf{x} - best)$$

Шаг 6. Формирование новых решений NB на базе *best*.

ЦИКЛ по j от 1 до $|W|$ ВЫПОЛНИТЬ

$$NB.\mathbf{x} := best \pm \text{rand}\ (W.\mathbf{x} - best)$$

Шаг 7. Вычисление нормированной ошибки решений для всех фуражиров

F :=NW+NB+W

$$co.j := \text{ошибка}.j \ / \ \Sigma_i \text{ошибка}.i$$

Шаг 8. Формирование W. В него войдут решения из F,

$$co.j \begin{cases} \le 0{,}01, \text{ то решение } j \text{ включается } 4 \text{ раза}, \\ \le 0{,}1, \text{ то решение } j \text{ включается } 2 \text{ раза}, \\ \le 0{,}3, \text{ то решение } j \text{ включается } 1 \text{ раз}/ \end{cases}$$

Шаг 9. Формирование разведчиков BS:= $|F| - |W|$.

Шаг 10. Если превышено количество итерации $IterM$ или достигнута требуемая точность, то ВЫХОД, иначе { $Iter$:= $Iter$ +1, T:= αT, переход на *Шаг* 2}.

Эксперимент, обсуждение результатов

Суть эксперимента заключалась в аппроксимации нечеткой системой следующей тестовой функции: $f(x_1, x_2) = x_1 * \sin(x_2)$. Входные переменные описаны пятью нечеткими термами. База содержит 25 нечетких правил. Термы заданы треугольными ФП. Критерием аппроксимации является средняя абсолютная ошибка вывода.

Алгоритмы, рассмотренные в статье, принадлежат к классу методов, основанных на популяции. К этому же классу относится и генетический алгоритм, сравним эти алгоритмы:

– алгоритмы работают с множеством потенциальных решений (популяцией), основываясь на принципе сотрудничества между индивидами популяции;

– алгоритм на основе колоний прост в реализации и не требует больших вычислительных затрат; здесь отсутствуют специфические генетические операторы кроссовера и мутации.

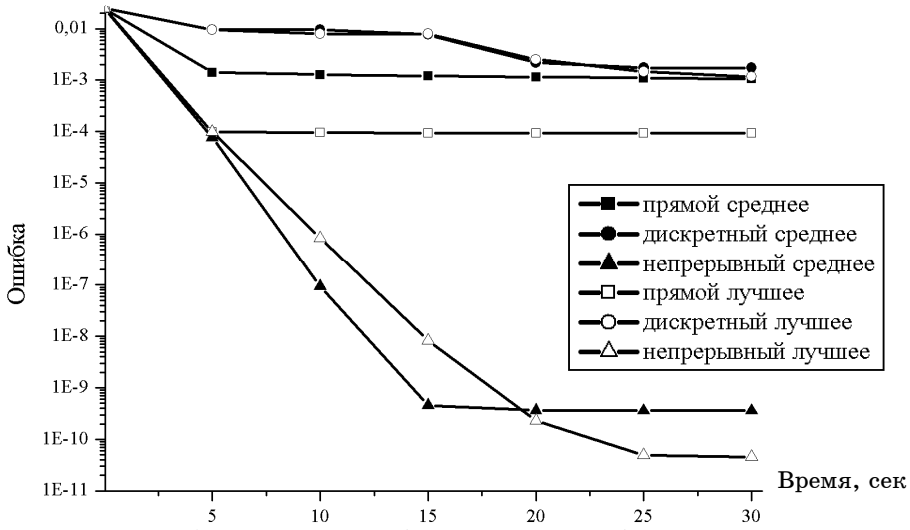

Рис. 1. Зависимость ошибки от времени работы алгоритма

Результаты эксперимента (рис. 1) показали, что эффективность непрерывного алгоритма муравьиной колонии выше по сравнению с остальными алгоритмами. Эффективность и скорость прямого алгоритма немного выше дискретного алгоритма муравьиной колонии и сопоставима с алгоритмом пчелиной колонии.

Литература

1. Dorigo M. Ant System: Optimization by Colony of Cooperating Agents / M. Dorigo, V. Maniezzo, A. Colorni // IEEE Transaction Systems, Man and Cybernetics. – Part B. – 1996. – Vol. 26. – P. 29–41.

2. Ходашинский И.А. Параметрическая идентификация нечетких моделей на основе гибридного алгоритма муравьиной колонии / И.А. Ходашинский, П.А. Дудин // Автометрия. – 2008. – Т. 44, № 5. – С. 24–35.

3. Socha K. Ant Colony Optimization for Continuous Domains /K. Socha, M. Dorigo // Technical Report TR/IRIDIA/2005-037. – Bruxelles: Universite Libre de Bruxelles, 2005. – 34 p.

4. Kong M. Application of ACO in Continuous Domain / M. Kong, P. Tian, L. Jiao et al. (Eds.): ICNC 2006, LNCS 4222. – Part II. – Berlin: Springer-Verlag, 2006. – P. 126–135.

5. Karaboga D. On the performance of artificial bee colony (ABC) algorithm / D. Karaboga, B. Basturk // Applied Soft Computing. – 2008. – Vol. 8. – P. 687–697.

Ходашинский Илья Александрович
Д-р. техн. наук, профессор каф. автоматизации обработки информации (АОИ) ТУСУРа
Тел.: (382-2) 70-15-93
Эл. почта: hodashn@rambler.ru

Дудин Павел Анатольевич
Аспирант каф. АОИ ТУСУРа

Горбунов Иван Викторович
Студент каф. АОИ ТУСУРа
Эл. почта: avla@inbox.ru

I.A. Hodashinsky, I.V. Gorbunov, P.A. Dudin

Bee and ant colony algorithms for fuzzy systems identification

In this article, four techniques based on colony paradigms for fuzzy models identification are considered: discrete ant colony algorithm, continuous ant colony algorithm, direct ant colony algorithm, and bee colony algorithm.

Keywords: fuzzy systems, metaheuristics, ant colony optimization, bee colony algorithm.