УДК 519.711

НЕЙРОНЕЧЕТКИЕ СИСТЕМЫ УПРАВЛЕНИЯ

М. В. Бураков,
канд. техн. наук, доцент
А. С. Коновалов,
д-р техн. наук, профессор
Санкт-Петербургский государственный университет аэрокосмического приборостроения (СПбГУАП)

Рассматриваются механизмы нейронечеткой технологии разработки интеллектуальных систем управления сложными объектами. Этот комплексный подход предполагает использование имитационного моделирования для извлечения знаний о процессе управления, нечетких технологий для представления и накопления знаний, нейросетевых механизмов для реализации закона управления, а также генетического алгоритма для настройки регулятора.

Введение

Нейронечеткие регуляторы (ННР) используют нечеткие правила и нейронный механизм реализации этих правил. Основное достоинство такого подхода заключается в том, что функционирование нейронной сети (НС) получает семантическую интерпретацию, а нечеткие правила реализуются в виде, допускающем обучение и параллельную обработку. Исследования ННР ведутся с середины 80-х годов прошлого века ([1–4] и др.).

ННР могут использоваться практически в тех же областях, что и «обычные» нечеткие регуляторы и НС, а именно — в системах распознавания, прогнозирования, диагностики и управления [5, 6]. Однако процесс конструирования ННР отличается тем, что он ориентирован на извлечение неизвестных знаний из данных (*data mining*), что необходимо для реализации фазы структурного обучения регулятора, предшествующей фазе параметрического обучения. Эта важная особенность может сделать ННР эффективным инструментом при работе с имитационными моделями (ИМ) сложных объектов.

Проблема конструирования регуляторов для сложных объектов с помощью ИМ рассмотрена в работах [7, 8]. Основная задача здесь заключается в переходе от ИМ к модели управления объектом путем выделения конечного набора управляющих правил из большого массива экспериментальной информации. Повсеместное использование компьютерного моделирования позволяет утверждать, что применение ННР на этом направлении является одним из важнейших и весьма перспективным.

Нечеткие контроллеры и моделирование

Нечеткие регуляторы, или нечеткие логические контроллеры (НЛК), являются наиболее ярким приложением теории нечетких множеств Л.Заде [9]. Базовое понятие этой теории предполагает, что любой элемент множества $x \subset X$ может соответствовать сразу нескольким нечетким подмножествам $A, B, \ldots Z$ с разной степенью принадлежности $\mu_A(x), \mu_B(x), \ldots, \mu_Z(x)$, принимающей значения в диапазоне [0, 1]. Это позволяет описывать качественные понятия, вводя в рассмотрение лингвистические переменные (ЛП).

В НЛК используются обычно ЛП «ошибка управления» (ε^*), «производная ошибки управления» ($(d\varepsilon(t)/dt)^*$) и «сигнал управления» (u^*). Эти ЛП принимают значения на множестве термов: «отрицательное большое» (ОБ), «положительное малое» (ПМ), «нулевое» (Н) и т. п. Каждый из термов является нечетким множеством. Механизмы работы НЛК детально описаны в обширной литературе (например, [10]). Здесь лишь подчеркнем, что нечеткие правила связывают наблюдаемую ситуацию и управление, которое должно в этой ситуации использоваться:

правило 1: если (ε^* = «Н») и ($(d\varepsilon(t)/dt)^*$ = «ОМ») то (u^* = «ПМ»);

правило 2: если (ε^* = «ОБ») и ($(d\varepsilon(t)/dt)^*$ = «ПМ») то (u^* = «Н»);

....

правило N: если (ε^* = «ПМ») и ($(d\varepsilon(t)/dt)^*$ = «ПС») то (u^* = «ОМ»).

Обычно правила НЛК формулируются с помощью человека-эксперта. Но этого часто бывает недостаточно для заполнения всей базы знаний. Здесь кроется главная трудность использования НЛК. Другая проблема заключается в отсутствии математического доказательства устойчивости НЛК. Эти проблемы могут ограничивать использование НЛК в некоторых областях и приложениях. Но, с другой стороны, условия устойчивости обычно доказываются для упрощенной (линейной) модели системы. Убедиться в устойчивости системы управления сложным объектом можно либо после проведения натурных экспериментов, либо после серии вычислительных экспериментов с использованием адекватной (нелинейной) математической модели объекта.

НЛК присущи такие признанные достоинства, как малая чувствительность к внешним и параметрическим возмущениям в силу естественной адаптивности нечетких правил «в малом» и гибкость в силу того, что закон управления формируется из отдельных правил, так что каждое новое правило может дополнить описание закона управления в новой ситуации. Эти особенности открывают возможность использования компьютерного моделирования для формулирования правил управления объектом.

Несмотря на впечатляющую мощь современной вычислительной техники, использование ИМ для синтеза управления в реальном времени во многих случаях невозможно по ряду причин.

• Управляющее решение синтезируется после выполнения множества прогонов модели с варьируемыми параметрами. При этом нельзя гарантировать не только оптимальность полученного результата, но и саму возможность его получения за отведенное для принятия решения время.

• Для обеспечения адекватности модели объекту нужно решать проблему идентификации в реальном времени. В общем случае настройка модели также требует выполнения прогонов с изменяемыми параметрами.

Таким образом, можно констатировать, что обычно ИМ может лишь качественно соответствовать сложному объекту и не может использоваться в реальном времени. Однако при соблюдении качественного соответствия ИМ может использоваться в режиме *off line* для обучения интеллектуального регулятора. Аппаратом для обработки качественной информации служит теория нечетких множеств.

Методика формулирования нечетких правил по результатам компьютерного моделирования предполагает два этапа работы [3, 4].

На первом этапе многократно ставится задача управления при различных условиях. С помощью поисковых процедур синтезируется множество эталонных траекторий. Фиксируя вход и выход модели при движении по эталонной траектории в разные моменты времени, можно получить множество обучающих пар $<X, Y>$, где X — вектор входа модели (управление), Y — вектор выхода модели.

На втором этапе обучающие пары обрабатываются с целью формирования нечетких правил управления. Для этого нужно преобразовать обучающие пары в нечеткую форму. Здесь возможны два варианта:

1. Заранее описываются ЛП таким образом, что их термы равномерно распределяются по соответствующим базовым шкалам входных и выходных переменных.

2. Каждое «четкое» значение входа и выхода модели преобразуется в нечеткую форму, становясь центральной точкой функции принадлежности терма ЛП, в которой $\mu_A(x) = 1$.

Соответственно, можно предложить два алгоритма построения нечетких правил, один из которых основан на обработке значений принадлежности элементов обучающих пар к термам ЛП, а второй — на векторной классификации с помощью вспомогательной сети Кохонена.

Полученный итоговый набор нечетких правил является предварительным, с его помощью решается задача оценки структуры НС прямого распространения, параметры которой подстраиваются на следующем этапе работы.

Выбор структуры нейронечеткой системы

Формирование нечетких правил управления. В работах [7, 11] была сформулирована простая методика для генерации нечетких правил из вход—выходных данных, получаемых при моделировании в режиме *off line*. Похожая методика независимо предложена в работе [12].

Рассмотрим для простоты изложения систему с двумя входами x_1 и x_2 и одним выходом y. Пусть имеется множество обучающих данных, полученных при движении объекта по эталонным траекториям:

$$\left(x_1^1, x_2^1, y^1\right), \left(x_1^2, x_2^2, y^2\right), \ldots, \left(x_1^n, x_2^n, y^n\right).$$

Для получения нечетких правил выполняются следующие шаги.

Шаг 1. Входное и выходное пространства разделяются на нечеткие области — термы ЛП. Число термов каждой входной и выходной ЛП фиксировано, и они равномерно распределены в области определения каждой ЛП.

Для функций принадлежности термов вводится условие полноты ε [13] при $\varepsilon = 0{,}5$ или $\varepsilon = 1$ (рис. 1). Это условие означает, что для заданного значения x всегда существует терм T_i, такой что $\mu_{Ti}(x) \geq \varepsilon$. За счет этого условия нечеткая система выводов обеспечивает плавный переход от одного лингвистического значения к другому.

Шаг 2. Генерируются нечеткие правила. Сначала определяется степень принадлежности входных данных к отдельным термам ЛП. Например, пусть

$$\mu_{\text{ПМ}}(x_1^1) = 0{,}8;$$
$$\mu_{\text{ПС}}(x_2^1) = 0{,}2;$$
$$\mu_{\text{Н}}(y^1) = 0{,}6,$$

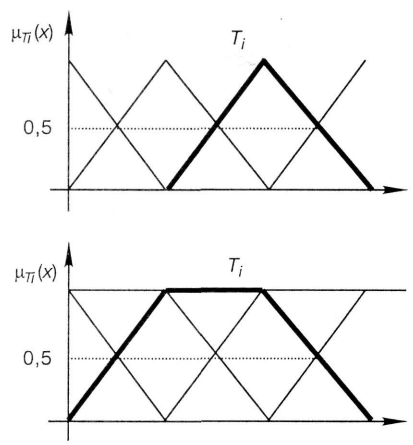
где $\mu_{\text{ПМ}}(x_1^1)$ — степень принадлежности входного значения x_1 к терму с названием «ПМ» соответствующей ЛП.

Таким образом, получается совокупность троек вида:

(0,8; 0,2; 0,6), (0,7; 0,5; 0,1) ...

Эта совокупность троек степеней принадлежности описывает все возможные варианты правил для каждой тройки входных данных.

Затем выбирается тройка с максимальным значением принадлежности по двум входным и



■ ***Рис. 1.*** *Описание термов лингвистических переменных*

одной выходной переменной, которая порождает правило. Например, правило с номером 1:

$$(x_1^1, x_2^1, y^1) \Rightarrow (\mu_{ПМ}(x_1) = 0{,}8;\ \mu_{ОС}(x_2) = 0{,}6;\ \mu_{ОБ}(y) = 0{,}9) \Rightarrow$$

Правило 1: если x_1 = ПМ и x_2 = ОС, то y = ОБ.

Шаг 3. Присвоение коэффициента определенности (КО) каждому правилу. Это позволяет выполнить сжатие базы правил, т. е. решить следующие задачи.

- во-первых, разрешить конфликты, когда одни и те же посылки в разных правилах порождают разные заключения;
- во-вторых, сократить общее число правил, так как учитываться могут только правила с максимальным КО.

Каждое правило активизируется с учетом своего КО, для вычисления которого можно использовать одну из двух формул (при приведенных выше данных):

$$КО = \mu_{ПМ}(x_1) \cdot \mu_{ОС}(x_2) \cdot \mu_{ОБ}(y) = 0{,}8 \cdot 0{,}6 \cdot 0{,}9 = 0{,}432;$$

$$КО = \min(\mu_{ПМ}(x_1) \cdot \mu_{ОС}(x_2) \cdot \mu_{ОБ}(y)) = \min(0{,}8 \cdot 0{,}6 \cdot 0{,}9) = 0{,}6.$$

Таким образом, если два или больше правил имеют одинаковые посылки и заключение, то используется правило, у которого больше КО.

Очевидно, этот метод может быть распространен и на системы со многими входами и многими выходами.

Использование сетей Кохонена. Для выделения нечетких правил из данных также можно использовать сети Кохонена — эффективный инструмент векторной классификации [4].

Будем считать, что U, V и W — нечеткие подмножества конечных множеств X, Y и Z мощностью соответственно n_x, n_y и n_z. Пусть имеется q наборов данных моделирования, описываемых тройками (A_i^*, B_i^* и C_i^*), принимающими значения на U, V и W.

Сеть Кохонена [14] имеет входной и выходной слой (рис. 2). Входной слой должен содержать $n_x + n_y + n_z$ входов, они соответствуют каждому элементу множеств X, Y и Z. Выходной слой должен содержать $n < q$ нейронов (точное число нейронов определяется по окончании обучения).

Каждый вход соединен со всеми выходами. Начальные значения весов присваиваются случайно. Введем обозначения: a_{ij}, b_{ij}, e_{ij} — значение веса между входом x_j, y_j, z_j соответственно и выходом i; A_i, B_i, E_i — векторы, составленные из a_{ij}, b_{ij}, e_{ij}.

Пусть имеются входные наблюдения (A^*, B^* и C^*). Для каждого выходного нейрона необходимо вычислить три значения сходства:

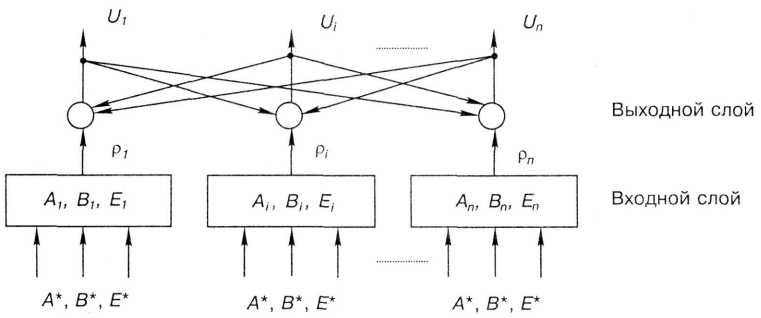
1) $\rho_i^a = S((A^*(x_1), a_{i1}), (A^*(x_2), a_{i2}), \ldots, (A^*(x_{nx}), a_{inx}))$;

2) $\rho_i^b = S((B^*(y_1), b_{i1}), (B^*(y_2), b_{i2}), \ldots, (B^*(y_{ny}), b_{iny}))$;

3) $\rho_i^e = S((E^*(z_1), e_{i1}), (E^*(z_2), e_{i2}), \ldots, (E^*(z_{nz}), e_{inz}))$.

Возможная формула для измерения сходства:

$$S((\alpha_1, \beta_1), (\alpha_2, \beta_2), \ldots, (\alpha_n, \beta_n)) = 1 - \frac{1}{n}\sum_{i=1}^{n} |\alpha_i - \beta_i|.$$



■ ***Рис. 2.*** *Сеть Кохонена в режиме кластеризации*

На следующем шаге для вычисления окончательного выходного сигнала нейрона вводится переменная:

$$\rho_i = \min\left[\rho_i^a, \rho_i^b, \rho_i^e\right].$$

В выходном слое действует механизм латерального возбуждения-торможения (см. рис. 2). В результате соревнования происходит возбуждение или не возбуждение отдельных нейронов. Если нейрон i побеждает в соревновании и возбуждается, то он (и его ближайшие соседи) модифицируется в направлении входного вектора к (A^*, B^*, C^*). Изменение весов нейрона i выполняется по формулам:

$$A_i' = A_i + \alpha(A^* - A_i);$$

$$B_i' = B_i + \alpha(B^* - B_i);$$

$$E_i' = E_i + \alpha(E^* - E_i).$$

Нейроны, близкие к i, также подстраиваются, но в меньшей степени. Так веса победивших нейронов приближаются к входным значениям.

Как показано в [14], в результате повторения этого процесса со всеми обучающими данными обеспечивается их кластеризация весами нейронов A_i, B_i и E_i в соответствии с обучающими данными. Один кластер относится к потенциальному правилу, которое можно вставить в базу правил. Каждый кластер представляется тройкой (A, B, E), которая формирует основу правила:

если (V есть A) и (U есть B) то (W есть E)

Веса связей описывают функции принадлежности термов ЛП, относящиеся к посылкам и заключениям продукционных правил. Таким образом, можно считать, что здесь все правила имеют КО = 1.

Архитектура нейронечеткого регулятора

Архитектура *HyFIS* (*Hybrid neural Fuzzy Inference System*) представляет собой многослойную сеть прямого распространения [15]. Топология *HyFIS* показана на рис. 3.

Нейроны входного слоя получают входные состояния, а нейроны выходного слоя вырабатывают управляющие решения. Промежуточные слои описывают функции принадлежности и правила, что позволяет получить яс-

ное представление о механизме работы такой сети.

Нейроны слоя 1 просто передают входные («четкие») сигналы в последующий слой. Каждый нейрон слоя 1 связан только с теми нейронами слоя 2, которые описывают соответствующие термы ЛП.

Нейроны слоя 2 описывают термы входных ЛП. Для входного значения вычисляются функции принадлежности к термам соответствующей ЛП. Изначально веса связей между 1-м и 2-м слоем единичные, и функции принадлежности распределены равномерно в пространстве весов. Если существуют какие-то экспертные знания, то их можно использовать при инициализации.

Нейроны слоя 2 используют гауссовские функции принадлежности, два параметра которых — центр (c) и ширина (σ) — настраиваются при обучении:

$$y_j^2 = \mu_{T_j}(x_i) = e^{-((x_i - c_j)^2/\sigma^2)}.$$

На рис. 3 ЛП описываются с помощью термов «большой» (Б), «средний» (С), «маленький» (М). Во многих случаях нужно учитывать знак входной переменной, тогда используются термы «положительный большой» (ПБ), «отрицательный средний» (ОС) и т. д.

Выходом нейронов второго слоя являются функции принадлежности входного значения к термам ЛП.

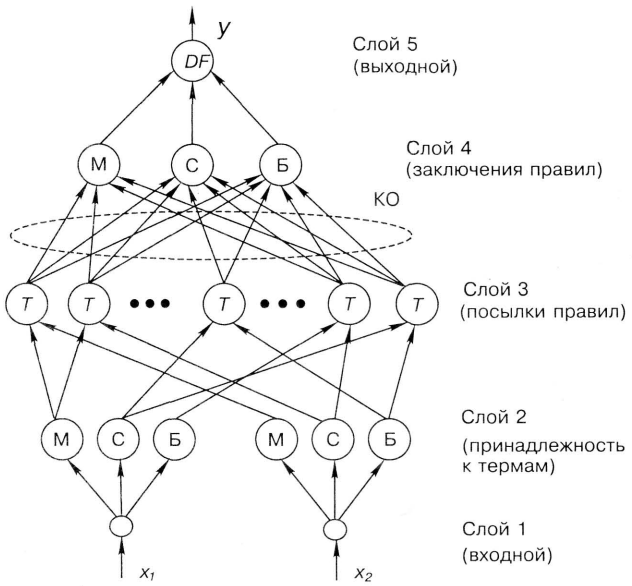
Каждый нейрон слоя 3 описывает посылки нечетких правил. Веса связей полагаются единичными. Нейрон этого слоя выполняет операцию *AND* (или вычисляет другую *T*-норму). Совокупность нейронов этого слоя формирует нечеткую базу правил. Функционирование каждого нейрона слоя 3 можно описать так:

$$y_j^3 = \min_{i \in I_j} (y_i^2),$$

где I_j – множество нейронов слоя 2, связанных с j-м нейроном слоя 3, y_i^2 — выход нейрона i слоя 2.

Веса связей между слоями 3 и 4 описывают КО правила.

Нейроны слоя 4 описывают возможные заключения правил. Каждый нейрон реализует функцию *OR*, учитывая совместное действие правил с одним и тем же термом заключения. Так описывается область восприятия выходной переменной. Уровень активизации нейрона слоя 4 описывает степень



■ ***Рис. 3.*** *Нейронная сеть HyFIS*

соответствия его функции принадлежности всем нечетким правилам вместе. Функционирование нейрона слоя 4 можно описать так:

$$y_k^4 = \max_{j \in I_k} (y_j^3 w_{kj}),$$

где I_k – множество нейронов слоя 3, связанных с k-м нейроном слоя 4, y_k^4 — выход нейрона k слоя 4.

Связи w_{kj} между слоями 3 и 4 действуют как механизм вывода, позволяющий избежать состязания правил. Каждое правило активизируется с некоторой степенью, представленной квадратом веса связи.

Слой 5. Здесь вырабатывается выходная (выходные) переменные системы, т. е. выполняется операция дефаззификации (*DF*) — вычисления точного выходного значения. Для этого может применяться метод центра тяжести или центра области (ЦО). Схема дефаззификации ЦО предполагает расчет выходного сигнала по формуле:

$$y_l^5 = \frac{\sum_{k \in I_l} y_k^4 \sigma_{lk} c_{lk}}{\sum_{k \in I_k} y_k^4 \sigma_k},$$

где I_l – множество нейронов слоя 4, связанных с k-м нейроном слоя 5; σ_{lk} и c_{lk} – ширина и центральная точка функции принадлежности k-го нейрона слоя 4.

Веса связей нейронов слоев 4 и 5 единичные. Обучаются только веса между слоями 3 и 4.

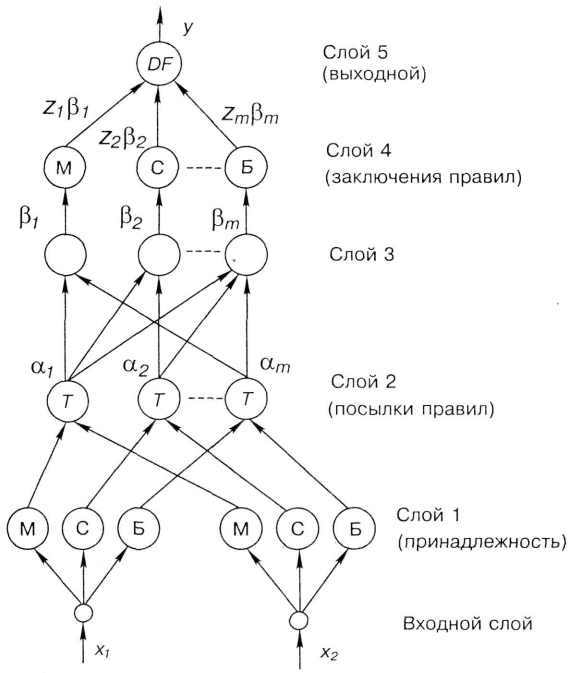
Таким образом, архитектуру *HyFIS* выгодно использовать при процедуре выделения нечетких правил.

Архитектура *ANFIS* (*Adaptive Neuro-Fuzzy Inference System*) описана в работе [16]. Структура *ANFIS* представлена на рис. 4.

Функционирование *ANFIS* несколько отличается от *HyFIS*. Рассмотрим работу *ANFIS* подробнее.

- Нейроны входного слоя и слоя 1 работают аналогично *HyFIS*.
- Слой 2 реализует нечеткую *T*-норму, на его выходе появляется величина α_i, являющаяся результатом применения нечеткой операции *AND* по отношению к посылкам соответствующего правила.
- Нейроны слоя 3 вычисляют значения:

$$\beta_i = \frac{\alpha_i}{\alpha_1 + \alpha_2 + \ldots + \alpha_m}.$$



■ ***Рис. 4.*** *Структура сети ANFIS*

• В слое 4 выполняется операция вычисления величины:

$$\beta_i z_i = \beta_i F_i^{-1}(\alpha_i),$$

где F – функция принадлежности, описывающая терм управления z_i.

• Слой 5 выполняет операцию дефаззификации:

$$z_0 = \beta_1 z_1 + \beta_2 z_2 + \ldots + \beta_m z_m.$$

Настройка сети *ANFIS* также предполагает корректировку функций принадлежности заключений нечетких правил F_i. Очевидно, что эту структуру выгодно использовать при процедуре выделения нечетких правил на основании имитационного моделирования.

Параметрическая настройка регулятора

В процессе параметрической настройки регулятора требуется минимизировать ошибку управления по всем обучающим парам, а также проверить работу регулятора в тех ситуациях, которые не входили в обучающую выборку. При работе с нелинейными ИМ для этого необходимо использовать генетический алгоритм (ГА). ГА основан на фундаментальных представлениях о механизме эволюции как процессе выживания наиболее приспособленных особей и гибели наименее приспособленных [17].

При использовании ГА настраиваемые параметры регулятора (КО правил или параметры функций принадлежности) кодируются двоичными цепочками конечной длины — хромосомами. Совокупность хромосом образует популяцию — множество точек многомерного пространства, в котором происходит поиск решения. Каждая хромосома оценивается путем моделирования переходного процесса (рис. 5).

При управлении динамическим процессом общая ошибка регулирования $\varepsilon(t)$ должна подсчитываться на достаточно большом количестве точек переходного процесса (N), тогда пригодность хромосомы можно оценить по формуле:

$$P = \frac{1}{1 + \sum_{i=1}^{N} \varepsilon(t_i)}.$$

На основании этих оценок «хорошие» хромосомы популяции размножаются с помощью генетического оператора копирования. Остальные генетические операторы — скрещивания и мутации — позволяют исследовать все пространство поиска решения [18, 19].

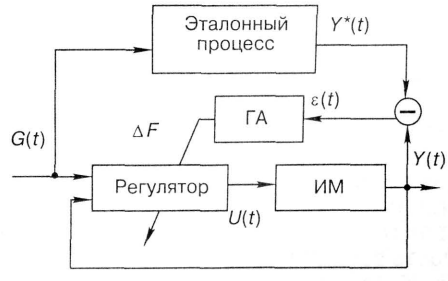
Успех параметрической настройки зависит от того, насколько верно была выбрана структура регулятора. В сложных случаях процесс обучения может включать несколько шагов, с чередованием структурной и параметрической настройки регулятора.

Практические приложения

Описанная методика может быть весьма эффективна в задачах управления сложными динамическими объектами. Рассмотрим два примера.

1. Задача управления подводным исследовательским буксируемым комплексом (БК) [20, 21]. БК представляет собой систему судно—трос—подводный аппарат (ПА). Траектория движения ПА определяющим образом зависит от динамики судна и троса длиной несколько километров. Нелинейная ИМ БК позволяет получить лишь относительно малое количество прогонов в ускоренном времени. Но в режиме *off line* может быть накоплено большое количество данных моделирования. В результате структурирования этой информации выделяется ограниченное число нечетких управляющих правил, параметры которых корректируются с учетом оперативной информации [20, 21].

2. Задача управления посадкой самолета вертикального взлета и посадки [22]. Используя нелинейную ИМ объекта управления, можно с помощью поисковых процедур синтезировать в режиме *off line* большое количество удовлетворительных траекторий движения объекта. После выполнения кластеризации пространства входов и выходов получается ограниченный набор нечетких правил,



■ ***Рис. 5.*** *Настройка регулятора с помощью ГА*

задающий структуру управляющей НС, параметры которой потом подвергаются коррекции. Для обеспечения адаптации в реальном времени возможно взаимодействие конкурирующих структур [22].

Заключение

И система нечетких правил, и искусственная НС являются универсальными аппроксиматорами — с их помощью можно описать любую нелинейную функцию. Однако только соединение этих двух подходов в рамках ННР позволяет одновременно использовать способности НС к обучению и удобства представления и анализа нечетких продукционных правил. Конструирование ННР включает два этапа, на каждом из которых необходимо использование ИМ объекта.

На первом этапе происходит структурный синтез регулятора — определение количества продукционных правил, достаточного для описания закона управления. Как было показано в статье, здесь возможны два подхода: либо предварительное описание термов ЛП, либо поиск этого описания в результате кластеризации. Соответственно, при первом варианте выгодно использовать архитектуру *HyFIS*, а при втором варианте — архитектуру *ANFIS*.

На втором этапе происходит параметрическая (точная) настройка — уточнение тех или иных параметров НС, реализующей нечеткую продукционную систему. Для архитектуры *HyFIS* — это коэффициенты определенности правил, а для архитектуры *ANFIS* — параметры функций принадлежности. Здесь выгодно использовать ГА — мощный инструмент поиска глобального экстремума целевой функции.

Описанная методика может использоваться при разработке систем управления многими реальными сложными объектами.

Литература

1. **Gupta M. M., Rao D. H.** On the principles of fuzzy neural networks // Fuzzy Sets and Systems. — 1984. — Vol. 61. — P. 1–18.
2. **Kosko B.** Neural network and fuzzy and fuzzy systems. Englewood Cliffs, NJ: Prentice-Hall, 1991.
3. **Keller J. M., Yager R. R., & Tahani H.** Neural network implementation of fuzzy logic // Fuzzy sets and systems. — 1992. — № 45. — P. 1–12.
4. **Ronald R. Yager.** Modeling and Formulating Fuzzy Knowledge Bases Using Neural Networks // Neural Networks. —1994. — Vol. 7. — № 8. — P. 1273–1283.
5. **Ayoubi M., Isermann R.** Neuro-fuzzy systems for diagnosis // Fuzzy sets and systems. — 1997. — № 89. — P. 289–307.
6. **Jang J. S. R., Sun C. T., Mizutani E.** Neuro-Fuzzy and Soft Computing. A Computational Approach to Learning and Machine Intelligence. Prentice-Hall International, 1997. — 613 p.
7. **Бураков М. В., Попов О. С.** Совместное использование имитационного моделирования и экспертных процедур для управления динамическими объектами // «Известия ВУЗов. Приборостроение». — 1994. — № 5–6. — С. 11–13.
8. **Бураков М. В., Коновалов А. С., Попов О. С.** Адаптивное управление сложными объектами // Оборонная техника. — 1998. — № 6–7. — С. 56–61.
9. **Mamdani E. H.** Application of fuzzy algorithms for control of simple dinamic plant. — IEEE Proc., 1974. — V. 121. — № 12. — P. 1585–1588.
10. **Бураков М. В., Попов О. С.** Интеллектуальные системы управления. Учебное пособие. — СПб.: ГААП, 1997. — 108 с.
11. **Бураков М. В., Попов О. С.** Формирование базы знаний управляющей экспертной системы. Рук. деп. в ВИНИТИ 19.05.93. — № 1329–В93. — 21 с.
12. **Wang I. X., Mendel J. M.** Generating fuzzy rules by learning from examples. IEEE Transactions on System, Man and Cybernetics — 1992. — 22(6). — P.1414–1427
13. **Lee C. C.** Fuzzy logic in control systems: fuzzy logic controller — part 1 and 2. IEEE Transaction on System, Man and Cybernetics. — 1990. — 20(2). — P. 404–435.
14. **Kohonen T.** Self-organization and associative memory. Berlin, Springer-Verlag, 1984.
15. **Kim J., Kasabov N.** HyFIS: adaptive neuro-fuzzy inference systems and their application to nonlinear dynamical systems // Neural networks, 12(1999). — P. 1301–1319.
16. **Jang J. S. R.** ANFIS: Adaptive network based fuzzy inference systems. IEEE Trans. on Systems, Man, and Cybernetics. 23(03): 665–685. May 1993.
17. **Goldberg D. E.** Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley, Reading, MA. 1989.
18. **Бураков М. В.** Синтез нейронного регулятора // Изв. Академии наук. Теория и системы управления. — 1999. — № 3. — С.140–145.
19. **Burakov M. V., Konovalov A. S.** Peculiarities of Genetic Algorithm Usage when Synthesizing Neural and Fuzzy Regulators // Advanced Computer Systems. Editors J. Soldek, J. Pejas. The Kluwer International Series in Engineering and Computer Science. P.139–148. KLUWER ACADEMIC PUBLISHERS, Boston/ Dordrecht/ London, 2002.
20. **Burakov M. V.** Synthesis Of The Fuzzy Controller // Proceeding of the first Inter. Conf. on Computer's Methods in Control Systems «CMCS' 97 »,Szczecin, Poland, December 11–12, 1997. — P. 77–82.
21. **Popov O. S., Burakov M. V.** Principles of Construction and Structure of an Automated Control System by an Underwater Towed Complex for Ocean Researchers // Proc. of the Second Int. Conf. on Marine Technology «ODRA 97», Szczecin, Poland, 13–15 may 1997. — P. 465–472.
22. **Burakov M., Konovalov A.** Development of dynamic object's neural controller by means of decomposition // Int. Conf. Of Advanced Computer systems «ACS'2000», 23-25 October 2000, Szchecin, Poland. — P.307–311.