Опубликовано в Научно-техническом вестнике СПбГУ ИТМО. 2008. Выпуск 53. Автоматное программирование, с. 295-308.

УДК 004.4’242

МЕТОД ОБУЧЕНИЯ СЛОЖНЫХ СИСТЕМ С БОЛЬШИМ ЧИСЛОМ ВХОДНЫХ ДАННЫХ И ВЫХОДНЫХ ВОЗДЕЙСТВИЙ

А.Л. Красс
(Санкт-Петербургский государственный университет информационных технологий, механики и оптики)

В работе предлагается метод, упрощающий процесс обучения класса сложных систем с большим числом входных данных и выходных воздействий. Он позволяет учитывать ограничения на поведение системы, что ведет к уменьшению размерности пространства поиска решений, удобным образом организовать разбиение задач системы на подзадачи, выполнению каждой из которых можно обучать систему отдельно от других и на основе различных методов машинного обучения. После этого полученные решения могут быть объединены с обеспечением достаточно сложные условия перехода между выполнением подзадач, в том числе вероятностные.
Ключевые слова: конечный автомат, машинное обучение

Введение

В области искусственного интеллекта существует множество задач обучения, в которых требуется учитывать большое число входных данных. Обучение таких систем – сложная задача. Оно обычно требует больших вычислительных затрат, а также тщательного и трудоемкого подбора настраиваемых параметров алгоритма обучения [1]. Наиболее существенны эти трудности для обучения без учителя [2].

В данной работе рассматриваются задачи, для которых применяется обучение без учителя. Для некоторых из них существуют небольшие подзадачи, для которых оптимальное или близкое к нему решение известно еще до обучения системы. В таких случаях можно сократить пространство поиска решений, зафиксировав решение известной подзадачи до обучения. Для распространенных методов это затруднительно, а если таких подзадач несколько, то могут возникнуть серьезные технические трудности. В данной работе предлагается метод, который позволяется избавиться от этих трудностей.

В работе вводится понятие автомата ограничений. Он представляет собой вероятностный автомат Мура-Мили [3] и организует совместную работу систем искусственного интеллекта, каждая из которых отвечает за выполнение своей подзадачи. С помощью этой структуры производится переключение между выполнением подзадач системы и контроль соблюдения ограничений на ее поведение. Примером ограничения может служить попытка робота пройти сквозь стену. Действия, которые непосредственно ведут к этой попытке, не имеют смысла и должны блокироваться автоматом ограничений.

Использование такой структуры дает возможность задавать каркас поведения системы, т.е. позволяет еще до обучения ограничить число допустимых стратегий системы, а также гарантировать, что нежелательные стратегии поведения не будут использованы, даже если они достаточно близки к оптимальной.

Автомат ограничений позволяет использовать решения подзадач, для которых существует оптимальное или достаточно близкое к нему решение еще до начала обучения системы. Во многих случаях это позволяет сократить количество входных данных, требующихся системе искусственного интеллекта для принятия решений, полностью перенеся ответственность за совершение некоторых действий в автомат ограничений. Это может существенно сократить пространство поиска решений и тем самым как уменьшить вычислительные затраты на процесс обучения системы, так и упростить подбор настраиваемых параметров алгоритма обучения.

Использование автомата ограничений может быть полезно в процессе применения генетических алгоритмов для генерации детерминированных конечных автоматов [4–7]. В этом случае при нарушении некоторого ограничения, к переходу, нарушившему его, имеет смысл применить оператор мутации. Таким образом, получается аналог условно-рефлекторной схемы [8] для обучения детерминированных конечных автоматов.

Методы обучения систем искусственного интеллекта

Наиболее часто системы искусственного интеллекта строят с помощью искусственных нейронных сетей или детерминированных конечных автоматов. В случае обучения без учителя единственно возможными вариантами обучения таких систем являются генетические алгоритмы [4, 9–13] и подкрепляющее обучение [14–18].

Для многих сложных систем выполняемую задачу часто можно разбить на некоторый набор подзадач. Для обучения таких систем имеет смысл использовать иерархическое подкрепляющее обучение [19–21]. Его основная идея состоит в том, чтобы сначала обучить систему выполнять наиболее простые подзадачи, а уже на их основе строить решения других подзадач и всей задачи в целом. Однако использование подкрепляющего обучения часто оказывается достаточно сложным или невозможным из-за того, что непонятно, как поощрять или наказывать за действия, которые привели к ситуации, в которой нужно поощрить или наказать систему. Очень часто непонятно, за что наказывать, если цель не достигнута по истечении времени, отведенного на выполнение задачи. В таких случаях иногда используют комбинацию генетических алгоритмов и подкрепляющего обучения [22–27]. Часто это позволяет добиться более высокой скорости обучения, чем при раздельном использовании этих методов.

Постановка задачи

Пусть дан робот, получающий информацию от некоторого числа сенсоров, возможно, обладающий памятью о своих прошлых действиях или о состоянии среды, способный совершать определенное число действий.

Перед роботом стоит задача, состоящая из набора подзадач, которые он должен выполнить максимально эффективно в автономном режиме. Для оценки эффективности выполнения каждой подзадачи имеется оценочная функция. Существует некоторое число подзадач, для которых решение может быть однозначно построено еще до обучения системы. Для остальных подзадач стратегия поведения не определена, и поэтому требуется применять методы обучения без учителя (в том числе подкрепляющее обучение). Требуется построить и обучить систему, управляющую роботом, так, чтобы она обеспечивала выполнение поставленных перед роботом задач. Заметим, что постановку задачи можно было бы дать и в более общих терминах систем искусственного интеллекта, но для простоты она была сужена на системы, применяемые для управления автономными роботами.

Рассмотрим достоинства и недостатки существующих методов применительно к данной задаче, сгруппировав их по управляющей структуре.

Искусственная нейронная сеть

При использовании в качестве управляющей структуры искусственной нейронной сети для ее обучения следует применять генетические алгоритмы и частично подкрепляющее обучение, так как по постановке задачи доступно только обучение без учителя.

Существует проблема задания каркаса поведения системы. Например, если обучаемая система управляет роботом-собакой, то этот робот должен вести себя именно как собака, а не как кошка. Искусственная нейронная сеть пытается найти оптимальное решение. Поэтому за такие ограничения должна отвечать оценочная функция нейрон-

ной сети, которая используется генетическим алгоритмом. Такая модификация может существенно замедлить вычисление этой функции, а также замедлить процесс обучения в целом. При этом ее построение может быть нетривиальной задачей.

Если требуется учитывать память о прошлом состоянии системы или состояниях окружающей среды, то для их учета требуется ввести новые входы нейронной сети, что также ведет к усложнению процесса обучения системы.

При использовании искусственных нейронных сетей вероятностное поведение системы может быть организовано достаточно естественным образом, если каждому действию системы сопоставить выход нейронной сети и считать, что чем больше значение на соответствующем выходе, тем больше вероятность совершения данного действия.

Детерминированный конечный автомат

Автомат – естественная структура для учета информации о прошлом состоянии системы или состоянии окружающей среды.

Детерминированные конечные автоматы обучают с помощью генетических алгоритмов [5–7]. На переходах автомата, применяемого для управления роботом, обычно размещают условия, сформированные из полученных от сенсоров данных. Так как сенсоров обычно много, то их данные группируют в одно значение или производят кластеризацию. Для кластеризации обычно применяют классифицирующие нейронные сети [1]. При большом числе сенсоров достаточно существенна проблема выбора правильного разбиения на кластеры. Эта проблема должна решаться каждый раз эвристически до основного процесса обучения или же можно объединить классифицирующую структуру, например, нейронную сеть, и автомат [28], а потом применять генетический алгоритм к ним обоим, что крайне сильно увеличивает пространство поиска решений. При этом детерминированный конечный автомат достаточно естественно может задать каркас системы, если того требует задача. Однако вероятностное поведение с помощью такого автомата задать нельзя.

Вероятностный автомат

На вероятностный автомат [3] переносятся особенности обучения, изложенные в предыдущем разделе. Единственное существенное различие состоит в вероятностном поведении. Оно достигается за счет усложнения автомата, а это ведет к увеличению размерности пространства поиска решений, что может быть неприемлемо во многих случаях.

Выводы

На основании изложенного можно сделать вывод, что на данный момент не существует удовлетворительного метода, который объединял бы в себе возможность задания каркаса поведения системы, обучения подзадач различными методами на основе различных структур (искусственные нейронные сети, детерминированные автоматы Мура-Мили, вероятностные автоматы), а также имел возможность учитывать подзадачи, решение которых известно еще до процесса обучения, ограничения на поведение системы и обеспечивать вероятностное поведения системы.

Метод обучения сложных систем с большим числом входных данных и выходных воздействий

В настоящей работе предлагается метод обучения систем с большим числом входных и выходных воздействий. Он основывается на применении автомата ограничений. Это модификация вероятностного автомата Мура-Мили [3], который задает основу поведения системы, указывает, какие действия можно совершать из данного состояния системы и при каких условиях, отвечает за переключением между выполнени-

ем подзадач. К каждому состоянию этого автомата можно «прикрепить» систему искусственного интеллекта (например, искусственную нейронную сеть или автомат Мура-Мили [29]). Поведение системы определяется системой искусственного интеллекта, соответствующей текущему состоянию автомата ограничений. Если к текущему состоянию не прикреплена система искусственного интеллекта, то поведение определяется только автоматом ограничений. При этом состояниям, относящимся к одной подзадаче, обычно ставится в соответствие одна система искусственного интеллекта.

Рассмотрим общую *схему предлагаемого метода*. Для некоторых задач отдельные пункты могут быть опущены.

1. Определение входных данных.
2. Определение выходных воздействий.
3. Выделение подзадач.
4. Построение автомата ограничений.
5. Закрепление за состояниями автомата ограничений систем искусственного интеллекта.
6. Выбор оценочных функций для каждой подзадачи.
7. Раздельное обучение систем искусственного интеллекта, отвечающих независимым подзадачам.
8. Объединение решений.
9. Выбор глобальной оценочной функции и дообучение системы в целом (при необходимости).

Опишем каждый этап метода на примере имитации поведения муравья во время поиска и сбора пищи. Части метода, выходящие за рамки данного примера будут опущены и могут быть найдены в работе [30].

Постановка задачи

По ограниченному квадратному полю размером 30×30 клеток передвигается муравей (рис. 1). На поле в случайно выбранных клетках находятся 100 единиц еды. Муравей может передвигаться в любую из восьми соседних клеток, брать еду из клетки, на которой стоит. При смене позиции муравей оставляет в покинутой клетке некоторое количество феромона. Чем больше ходов сделал муравей с последнего момента, когда он брал еду, тем меньше феромона он оставляет. Если муравей кладет феромон в клетку, в которой уже находится феромон, то их количества суммируются.

Муравей обладает информацией о том, что находится в соседних восьми клетках и в клетке, на которой он стоит (рис. 2). Если муравей стоит на границе поля, то к нему поступает информация, что клетки за границей поля недостижимы.

Таким образом, муравей имеет 35 сенсоров: девять сенсоров передают информацию о наличии еды в текущей и восьми соседних клетках поля, девять сенсоров передают информацию о количестве феромонов в текущей и восьми соседних клетках поля, девять сенсоров передают информацию о наличии муравейника в текущей и восьми соседних клетках поля, восемь сенсоров сообщают о проходимости восьми соседних клеток поля (находятся ли они за его границей).

После того как муравей взял еду, он должен с ней вернуться в специально выделенную часть поля размером 2×2 клетки, отстоящую на одну клетку от верхнего левого угла, и выполнить на одной из клеток этой части поля действие – положить еду. После этого он может приступить к поиску новой еды.

В любой момент времени муравей может нести не более одной единицы еды.

Муравей может за один ход либо взять еду из текущей клетки поля, либо положить еду в муравейник, либо переместиться в соседнюю клетку поля.

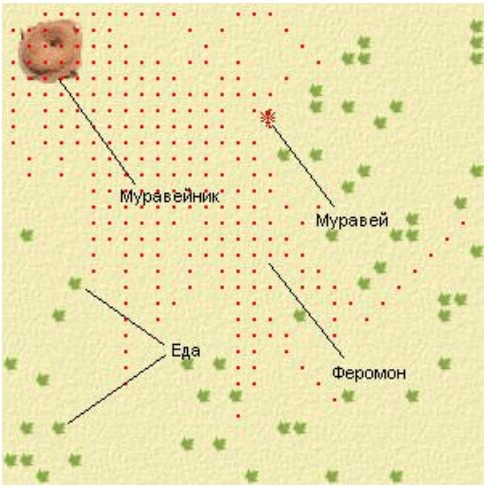

Рис. 1. Муравей, передвигающийся по полю

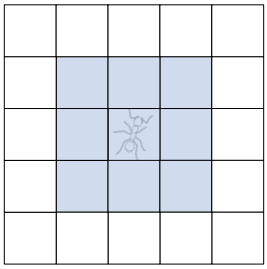

Рис. 2. Область обзора муравья

Муравей делает 1000 ходов. После этого количество еды, перенесенное в муравейник, фиксируется. Задача состоит в том, чтобы обучить муравья переносить как можно больше еды за 1000 ходов.

Определение входных данных

Входные данные генерируются 35 сенсорами:

- 9 сенсоров передают информацию о наличии еды в текущей и восьми соседних клетках поля;
- 9 сенсоров передают информацию о количестве феромона текущей и восьми соседних клетках поля;
- 9 сенсоров передают информацию о наличии муравейника в текущей и восьми соседних клетках поля;
- 8 сенсоров формируют информацию о проходимости восьми соседних клеток поля (находятся ли они за границей поля).

Определение выходных воздействий

Выходные воздействия – это действия муравья:

- взять еду из текущей клетки поля;
- положить еду в муравейник;
- переместиться на соседнюю клетку поля.

Выделение подзадач

На этом этапе требуется выбрать разбиение задачи системы на подзадачи. Для каждой подзадачи имеет смысл определить, какие входные данные ей потребуются и какие выходные воздействия система может генерировать.

Можно выделить две подзадачи: найти и взять еду, вернуться в муравейник и положить еду. Первую будем назвать задачей *A*, вторую – задачей *B*.Для выполнения каждой из этих задач муравей должен знать о наличии еды в текущей и восьми соседних клетках поля, о количестве феромонов в текущей и восьми соседних клетках поля, о проходимости восьми соседних клеток поля (находятся ли они за границей поля). Муравей должен иметь возможность перемещаться в любую из восьми соседних клеток поля, брать еду во время выполнения задачи *A* и класть еду в муравейник во время выполнения задачи *B*.

Построение автомата ограничений

Автомат ограничений строят в виде модификации вероятностного автомата Мура-Мили. Модификация заключается в том, что к каждому его состоянию может быть прикреплена система искусственного интеллекта, и переходы автомата могут содержать дополнительную информацию, говорящую насколько значим и желателен соответствующий переход. Например, имеет смысл помечать переходы, которые происходят по достижению системой какой-либо локальной цели. Эта информация может быть использована во время обучения подзадач и системы в целом.

Автомат ограничений должен отражать то, какие действия и при каких условиях система может совершать, находясь в определенных состояниях, и управлять переключением между выполнением подзадач (рис. 3). Если по одному событию возможны несколько переходов с условиями, которые могут быть верны одновременно, то этим переходам должны быть присвоены вероятности, которые в дальнейшем можно будет подобрать оптимальным образом в процессе обучения системы.

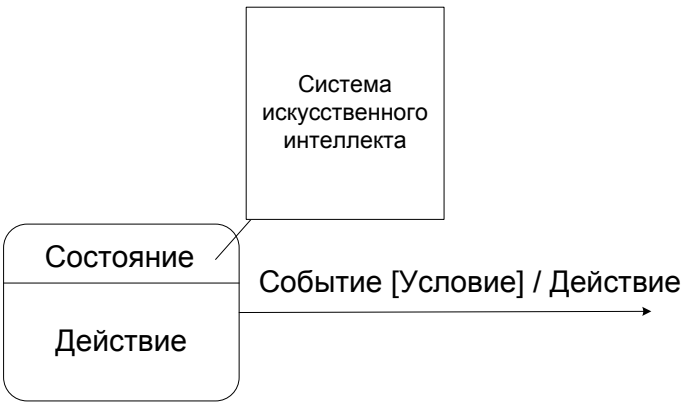

Рис. 3. Состояние автомата ограничений

Когда система хочет сделать ход (рис. 4), она получает из текущего состояния автомата ограничений соответствующую систему искусственного интеллекта, которая должна выбрать выходное воздействие системы или выдать вектор вероятностей выходных воздействий, на основе данных от сенсоров и памяти. Выходное воздействие является событием для автомата ограничений. Если при этом если не существует ребра из текущего состояния автомата ограничений по данному событию с выполняющимся условием перехода, то выходное воздействие не допускается. Если выбранное системой искусственного интеллекта выходное воздействие (или ни одно выходное воздействие из вектора вероятностей выходных воздействий с ненулевой вероятностью) не допус-

кается автоматом, то происходит перебор всех переходов из данного состояния автомата, пока не будет найден допустимый. Он и будет выполнен.

Рис. 4. Основной цикл работы системы под управлением автомата ограничений

Состоянию автомата ограничений может не ставиться в соответствие система искусственного интеллекта. В этом случае переход выбирается только на основании соблюдений условий перехода. Таким образом, автомат ограничений можно использовать как основную структуру управления системой, включая разного рода внутренние функции, например, диагностику неисправностей аппаратных компонентов системы. Это позволяет упростить использование автомата ограничений для задания многофункциональных систем.

Автомат ограничений строится на основе выбранного разбиения на подзадачи, а его переходы задают ограничения на поведение системы. Автомат ограничений строится так, чтобы каждое его состояние соответствовало некоторому состоянию системы. К каждому состоянию автомата ограничений добавляются переходы, которые возможны из этого состояния, тем самым будут заданы ограничения на поведение системы.

Построим автомат ограничений для рассматриваемой задачи. Автомат ограничений будем строить в виде автомата Мили [29], так как в данной задаче не требуется организовывать вероятностные переходы между выполнением подзадач.

Достаточно естественно выделить два состояния: *A* и *B*. В состоянии *A* муравей не несет еды, в состоянии *B* – несет. Опишем, каким условиям должны удовлетворять переходы из состояния *A:*

- если муравей в состоянии *A* стоит на еде, то он должен ее взять (это несколько спорный вопрос, но для простоты было выбрано именно это решение) и перейти в состояние *B;*
- если муравей в состоянии *A* видит еду, но не стоит на ней, то он должен идти в клетку, ее содержащую (если он видит несколько клеток с едой, то выбирается произвольная);
- если муравей не видит еды, то он может идти в любую сторону, если соответствующая клетка проходима;
- муравей не может класть еду в муравейник (так как у него не может быть еды в этом состоянии автомата).

Опишем, каким условиям должны удовлетворять переходы из состояния *B:*

- если муравей в состоянии *B* стоит на клетке муравейника, то он должен положить еду и перейти в состояние *A*;
- если муравей в состоянии *B* видит клетку муравейника, то он должен идти в нее (если он видит несколько клеток муравейника, то выбирает произвольную);
- если муравей не видит клетку муравейника, то может идти в любую сторону, при условии, что соответствующая клетка проходима;
- муравей не может взять еду (так как он уже несет еду).

Вышеописанный автомат приведен на рис. 5, 6.

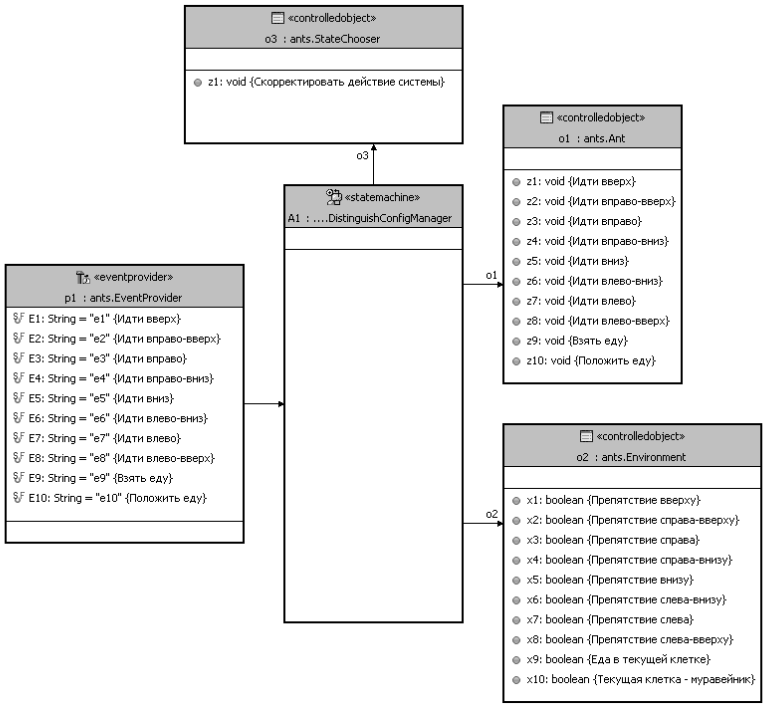

Рис. 5. Диаграмма классов для автомата, задающего ограничения на поведение муравья

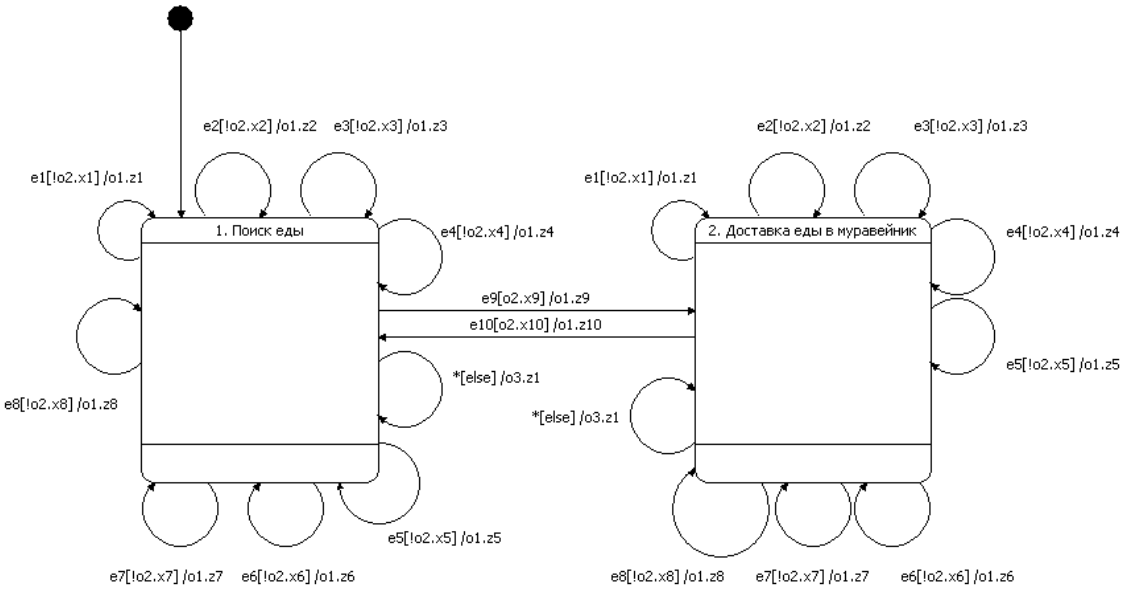

Рис. 6. Автомат ограничений для муравья

Закрепление за состояниями автомата ограничений систем искусственного интеллекта

«Прикрепим» к каждому состоянию автомата однослойную нейронную сеть. Такие нейронные сети просты в обучении и позволяют естественным образом обеспечивать вероятностное поведение системы. Для этого будем интерпретировать выходы нейронной сети как вектор вероятностей действий системы, где каждому действию соответствует один выход.

Как следует из предыдущего раздела, если в зону видимости муравья попадает еда, то его поведение полностью управляется автоматом ограничений, поэтому в нейронную сеть, прикрепленную к состоянию *A*, добавлять входы, отвечающие за информацию о еде не имеет никакого смысла. То же касается и выхода отвечающего за действие взять еду. Аналогичным образом автомат не позволит муравью выйти за границы поля. Поэтому можно исключить из нейронной сети и входы, отвечающие за проходимость клеток. Полученная нейронная сеть приведена на рис. 7.

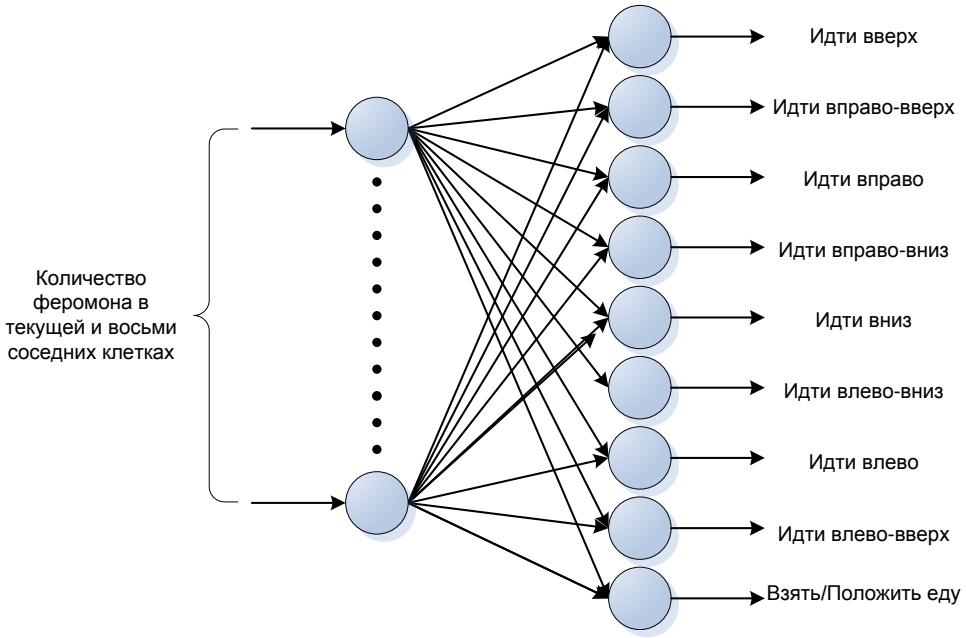

Рис. 7. Нейронная сеть, используемая для получения вероятностей совершения действий муравьем при использовании автомата ограничений

Отметим, что знание о наличии муравейника в текущей и восьми соседних клетках поля может быть полезным, так как муравейник находится практически в левом верхнем углу поля. Поэтому муравей может определить свою позицию на поле, если видит одну из клеток муравейника. В данном случае эта информация не используется, чтобы сохранить чистоту экперимента при сравнении решения, полученного с помощью предлагаемого метода, с решением данной задачи классическими методами, где при использовании этой информации существенно бы увеличилось пространство поиска решений, что еще больше усложнило бы нахождение решения.

Для выполнения задачи *B* муравей должен знать о наличии муравейника в текущей и восьми соседних клетках поля может быть полезным и о количестве феромонов в текущей и восьми соседних клетках поля. Так же он должен обладать информацией о проходимости восьми соседних клеток поля (находятся ли они за границей поля). Муравей должен иметь возможность перемещаться в любую из восьми соседних клеток поля и класть еду в муравейник.

Как можно видеть из предыдущего раздела, если в зону видимости муравья попадает муравейник, то его поведение полностью обеспечивается автоматом ограничений. Поэтому в нейронную сеть, прикрепленную к состоянию *A*, добавлять входы, отвечающие за информацию о муравейнике, не имеет никакого смысла. Это же относится и к выходу, отвечающему за действие «Положить еду». Аналогичным образом автомат не позволит муравью выйти за границы поля. Поэтому можно исключить из нейронной сети и входы, отвечающие за проходимость клеток. Полученная нейронная сеть приведена на рис. 7. Она имеет точно такую же структуру, что и нейронная сеть, использующаяся в состоянии *A*.

Таким образом, за рассмотренными состояниями закреплены нейронные сети одинаковой структуры. Они имеют девять входов, отвечающих за передачу информации о количестве феромона в текущей и соседних клетках, и девять выходов (рис. 7). Эти нейронные сети отвечают только за передвижение муравья, основываясь на феромонах, которые лежат в зоне видимости муравья. Остальные функции будет выполнять автомат ограничений.

Выбор оценочных функций для каждой отдельной задачи

Для задачи сбора еды оценочная функция будет выглядеть как усредненное количество еды, собранное за некоторое число запусков на случайно сгенерированных полях. Для задачи возвращения в муравейник оценочная функция – усредненное количество еды, положенное в муравейник за некоторое число запусков на полях, полученных в процессе выполнения первой подзадачи.

Раздельное обучение систем искусственного интеллекта, отвечающих независимым подзадачам

Теперь можно обучить систему выполнять каждую подзадачу отдельно. Так как было принято решение построить конечное решение только для подзадачи поиска и сбора пищи, то рассмотрим этот и последующие разделы в рамках этой задачи.

Данной подзадаче соответствует только одно состояние автомата ограничений. Чтобы иметь возможность обучить систему выполнять эту задачу, необходимо все переходы, которые ведут в состояния, не относящиеся к поставленной задаче, перевести в соответствующее состояние внутри группы. При этом во многих случаях требуется указать на этом ребре действие, которое бы приводило состояние системы в соответствие с поставленной задачей.

В данном случае есть только один переход, который ведет вовне группы. Это переход по действию «Взять еду». Переведем его в то же состояние, а на переходе к действию «Взять еду» добавим перемещение в левый верхний угол муравейника и действие «Положить еду». Тем самым будет выполнена имитация решения второй подзачи.

Для обучения нейронной сети используются генетические алгоритмы. Достаточная скорость и качество обучения достигаются при размере популяции в 100 особей, двухточечным кроссовером и отбором с применением элитизма. Подробнее процесс обучения рассмотрен в работе [30].

Анализ полученных результатов

После 100 итераций обучение перестало давать сколь-нибудь существенное увеличение функции приспособленности. Муравей был обучен собирать в среднем 48.8 единиц еды за 1000 ходов.

Данная подзадача была решена с помощью обучения искусственной нейронной сети генетическим алгоритмом без применения предложенного метода. Потребовалось использовать однослойную нейронную сеть из 26 входов и 9 выходов. Лучший результат был показан при схожих параметрах генетического алгоритма. За 500 итераций для популяции в 300 особей удалось научить муравья собирать столько же еды, что и при применении предлагаемого в данной работе метода. Но, как можно видеть, это потребовало в 15 раз больше вычислительных затрат.

Учитывая простоту задачи, можно сделать вывод, что для более сложных задач выигрыш при применении данного метода будет еще более существенен.

Заключение

В области искусственного интеллекта много задач обучения, в которых требуется учитывать большое число входных и выходных данных. Обучение таких систем – сложная задача. Оно обычно требует больших вычислительных затрат, а также тщательного и трудоемкого подбора настраиваемых параметров алгоритма обучения.

Если доступно только обучение без учителя, то распространенные методы [1, 2, 4–7, 9–28] из-за большой размерности пространства поиска решений могут требовать недопустимо высоких вычислительных затрат для обучения системы. Однако если из задач, которые должна выполнять система, можно выделить подзадачи, для которых оптимальное или близкое к нему решение известно еще до обучения системы, то во многих случаях можно сократить пространство поиска решений, применив предлагаемый в данной работе метод.

В результате сравнения результатов обучения искусственной нейронной сети генетическим алгоритмом с применением предложенного методом и без него можно сделать вывод, что данный подход может быть полезен для решения задач описанного класса, так как из-за уменьшения размерности пространства поиска решений возрастает скорость обучения. По этой же причине метод требует менее тщательного подбора настраиваемых параметров обучения.

Данный метод дает возможность задавать каркас поведения системы, т.е. еще до обучения ограничить число допустимых стратегий системы, а также гарантировать, что нежелательные стратегии поведения не будут использованы, даже если они достаточно близки к оптимальной.

Литература

1. De Nardi R., Togelius J., Holland O., Hollland L., Simon M. Evolution of Neural Networks for Helicopter Control: Why Modularity Matters / Proceedings of the IEEE Congress on Evolutionary Computation, 2006.
2. Барский А.Б. Нейронные сети: распознавание, управление, принятие решений. – М.: Финансы и статистика, 2004.
3. Bukharaev R. G. Probabilistic automata // Journal of Mathematical Sciences. – Springer New York. – 1980. – V. 13. – № 3.
4. Chambers L. D. Practical Handbook of Genetic Algorithms, Volume 3, Chapter 26, 6 – Algorithms to Improve the Convergence of a Genetic Algorithm with a Finite State Machine Genome. CRC Press, 1999.
5. Царев Ф.Н., Шалыто А.А. О построении автоматов с минимальным числом состояний для задачи об «умном муравье» / Сборник докладов X международной конференции по мягким вычислениям и измерениям. – СПбГЭТУ "ЛЭТИ". – Т. 2. – 2007. – С. 88–91.

6. Petrovic P. Evolving automatons for distributed behavior arbitration. Technical Report. Norwegian University of Science and Technology. 2005.
7. Petrovic P. Simulated evolution of distributed FSA behaviour-based arbitration / The Eighth Scandinavian Conference on Artificial Intelligence (SCAI'03). – 2003.
8. Цетлин М.Л. Исследования по теории автоматов и моделированию биологических систем. – М.: Наука, 1969.
9. Hussain T. Methods of Combining Neural Networks and Genetic Algorithms. http://citeseer.ist.psu.edu/hussain97methods.html
10. Filippidis A., Jain L. C., Martin N. M. Using genetic algorithms and neural networks for surface land mine detection //IEEE Transactions on Signal Processing. – 1999. – 47(1) – 176–186.
11. Koehn P. Combining genetic algorithms and neural networks: The encoding problem. Master's thesis. University of Erlangen and The University of Tennessee. Knoxville. 1994. – Режим доступа: http://citeseer.ist.psu.edu/article/koehn94combining.html
12. Mitchell M. An Introduction to Genetic Algorithms. First MIT Press, 1998.
13. Miller B., Goldberg M. Genetic algorithms, tournament selection, and the effects of noise // Complex Systems. – 1995. – V. 9. – I. 3. – P. 193–212.
14. Sutton R. S., Barto A. G. Reinforcement Learning: An Introduction. – The MIT Press, Cambridge. MA, 1998.
15. Kaelbling L. P., Littman M. L., Moore A. W. Reinforcement learning: A Survey // Journal of Artificial Intelligence Research. – 1996. – 4. – P. 237–285.
16. Stanley K., Miikkulainen R. Efficient Reinforcement Learning Through Evolving Neural Network Topologies / GECCO 2002, pp. 569–577.
17. Armstrong W. W., Coghlan B., Gorodnichy D. O. Reinforcement learning for robot navigation / International Joint Conference on Neural Networks (IJCNN'99). Proceedings. – Washington DC. – 1999.
18. Coulom R. Reinforcement Learning Using Neural Networks, with Applications to Motor Control. Institut National Polytechnique de Grenoble, 2002. – Режим доступа: http://citeseer.ist.psu.edu/coulom02reinforcement.html
19. Kaiser M., Dillmann R. Hierarchical learning of efficient skill application for autonomous robots / International Symposium on Intelligent Robotics Systems. – Pisa. Italy. – 1995.
20. Soni V., Singh S. Reinforcement Learning of Hierarchical Skills on the Sony Aibo Robot /In the International Conference on Developmental Learning (ICDL '06). – Bloomington. USA, 2006.
21. Diuk C., Strehl A. L., Littman M. L. A hierarchical approach to efficient reinforcement learning in deterministic domains / Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems. – Hakodate. Japan. – 2006.
22. Teller A., Veloso M. Internal Reinforcement in a Connectionist Genetic Programming Approach // Artificial Intelligence. – 2000. – V. 120. – № 2. – P. 165–198.
23. Whitley D., Dominic S., Das R. Genetic Reinforcement Learning with Multilayer Neural Networks / Proceedings of the Fourth International Conference on Genetic Algorithms. – P. 562–569. – Morgan Kaufmann.
24. Smith J. E. Coevolving Memetic Algorithms: A Review and Progress Report. // IEEE Transactions on Systems Man and Cybernetics – Part B. – 2007. – 37. – P. 6–17.
25. Ong Y. S., Keane A. J. Meta-Lamarckian learning in memetic algorithms // IEEE Transactions on Evolutionary Computation. – 2004. – 8. – P. 99–110.
26. Ong Y. S., Lim M. H., Zhu N., Wong K. W. Classification of Adaptive Memetic Algorithms: A Comparative Study" // IEEE Transactions on Systems Man and Cybernetics – Part B. – 2006. – 36. – P. 141–152.

27. Царев Ф.Н., Шалыто А.А. Применение генетического программирования для построения мультиагентной системы одного класса / Международная научно-техническая мультиконференция «Проблемы информационно-компьютерных технологий и мехатроники». Материалы международной научно-технической конференции «Многопроцессорные вычислительные и управляющие системы» (МВУС`2007). – Таганрог: НИИМВС, 2007. – Т.2. – С. 46–51.
28. Брауэр В. Введение в теорию конечных автоматов. – М.: Радио и связь, 1987.
29. Красс А. Метод обучения сложных систем с большим числом сенсоров и актуаторов. Бакалаврская работа / СПбГУ ИТМО, факультет «Информационные технологии и программирование», кафедра «Компьютерные технологии». – 2008.