С. Толкачев

НЕЙРОННОЕ ПРОГРАММИРОВАНИЕ

диалоговых систем

Проектирование «умных» Web-страниц

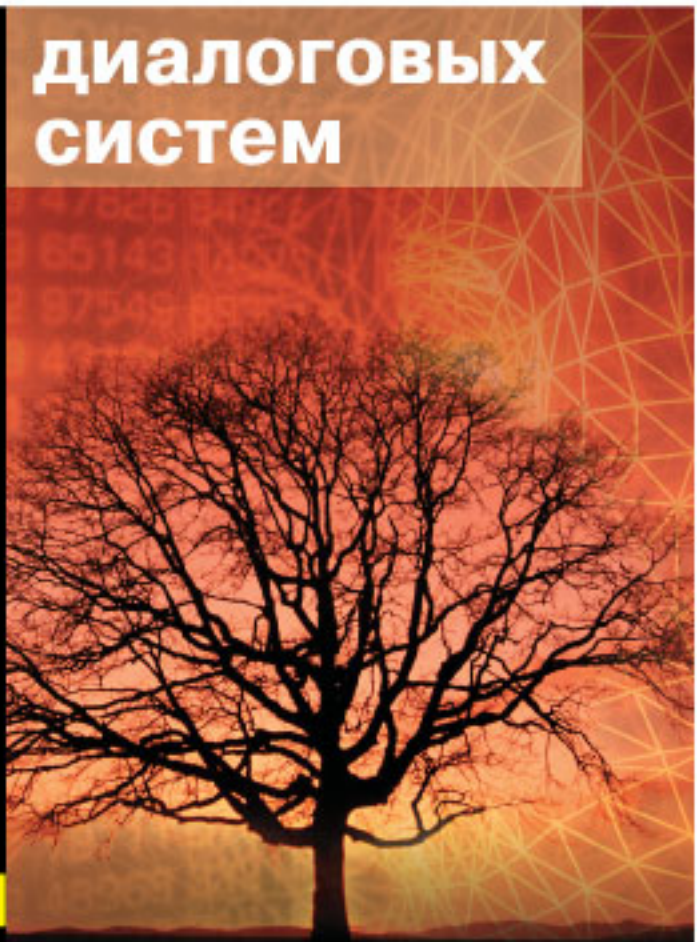
Усилители информации, нейроны и Интернет

Структуры и модели контекстных диалогов

Нейронные системы представления знаний

Интерактивный поиск в Интернет

Перцептивная лингвистика



WWW.ULIT.RU

Для программистов и разработчиков Web-приложений

...бывали случаи, когда из положительно дикого
брожения умов выходила со временем истина.

И. М. Сеченов. «Рефлексы головного мозга».

ВВЕДЕНИЕ

Способность выслушать и понять собеседника является одним из наиболее ценных достоинств человеческого общения. Задавая вопросы или просто обмениваясь фразами в процессе разговора, мы стремимся получать ответы, адекватные нашим внутренним потребностям. Нас в равной степени не удовлетворяют как сообщения, не приносящие информации, так и ответы, в которых ее количество значительно превосходит внутренние ограничения, установленные нами для конкретного разговора. Смысловое содержание ответа или *адекватная реакция* собеседника является одним из основных критериев, определяющим качество разговора и влияющим на изменение внутреннего состояния человека в процессе коммуникативного общения.

Если предположить, что у субъекта разговора существует некоторая целевая функция, которая определяет ожидаемое количество информации (*желаемый результат*), а количество информации, в поступающих к этому субъекту сообщениях, изобразить в виде некоторой траектории, то в зависимости от характера разговора эти траектории могут принимать самую разнообразную форму (рис. 1). Иногда необходимый результат может быть достигнут оптимальным образом (a). В иных случаях, задав тот или иной вопрос, вместо полезного ответа может быть получено большое количество избыточной информации, которая может увести далеко от первоначальной цели (b) и (c).

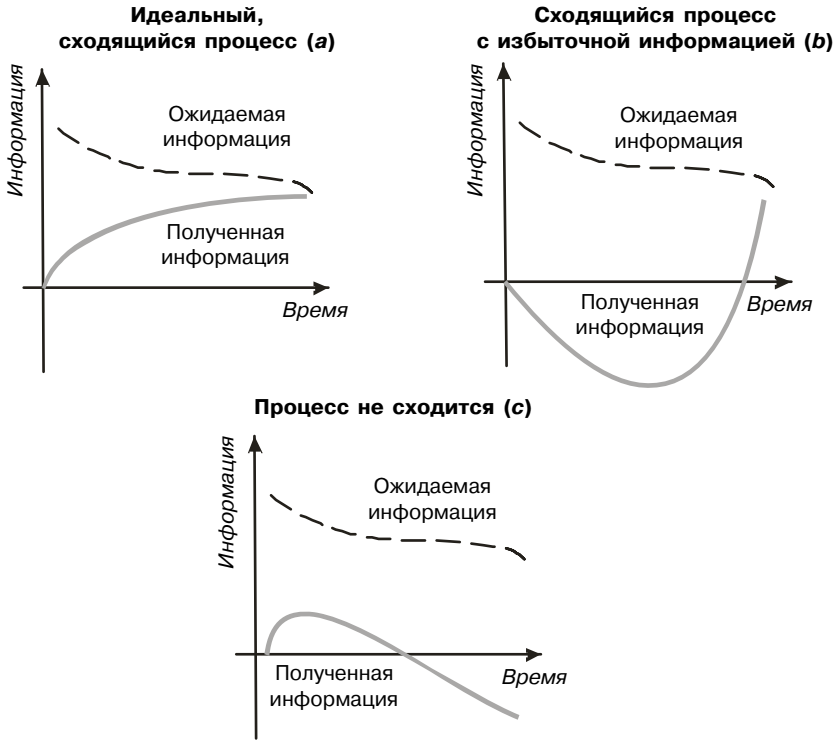


Рис. 1. Траектории развития процессов взаимодействия

Поток слов, поступающих к участнику разговора в процессе обмена сообщениями, вызывает возмущение сознания, которое можно представить себе как волнение поверхности воды в результате падения капель дождя. Распространение информационных волн в сознании, их интерференция друг с другом и взаимодействие с глубинными внутренними процессами образуют сложную систему, исследование и моделирование которой может быть достигнуто с применением методов и средств, аналогичных тем, которые применяются в физике и позволяют описывать поведение полей и частиц, основываясь на корпускулярно-волновых свойствах материи.

Если считать, что ожидаемое количество информации, которое мы стремимся получить в процессе взаимодействия, должно быть адекватно количеству информации, которое нам будет передано в ответе, мы можем сформулировать принцип информационной адекватности следующим образом:

$$I(A) \rightarrow I(Q),$$

где

I(A) — количество информации в ответе ***A***;

I(Q) — ожидаемое количество информации в ответ на заданный вопрос ***Q***.

В вычислительной технике широко применяются информационные характеристики различных видов устройств: памяти, процессоров, каналов связи и т. п., а из теории информации хорошо известны способы измерения информационных свойств потоков сообщений. Мы можем использовать аналогичные меры для оценки качества процессов взаимодействия и состояния участвующих в этом процессе систем. Однако если большинство информационных характеристик в вычислительной технике являются статическими, то процессы человеческого общения имеют ярко выраженный динамический характер. Моделирование таких процессов, определение их информационных характеристик, нахождение оптимальных траекторий взаимодействия и т. п. относятся к группе наиболее сложных и трудоемких задач в современном программировании. Среди основных факторов, определяющих их сложность можно выделить, в первую очередь, следующие:

- большое количество динамических параметров;
- постоянная адаптация и развитие внутренней структуры и функций;
- отсутствие четких критериев качества поведения.

Решение этих задач только лишь средствами традиционного программирования сопряжено с проблемами, корни которых лежат в логических основаниях алгоритмического моделирования. Программирование алгоритмических моделей, основу которого составляют последовательности пассивных логических конструкций, во многих случаях позволяет получить

приемлемый результат, но при этом *программа* и *результат* принципиально различаются и отделены друг от друга. В нейронных моделях используются активные элементы, обладающие внутренними динамическими свойствами и способные самостоятельно принимать и передавать сигналы. При этом исполнительные элементы и результаты представляют собой единое целое — динамическое пространство параллельных процессов и их состояний. Одним из способов создания таких моделей является нейронное программирование, под которым мы будем понимать методы организации и управления активными элементами в интерпретирующих системах.

Нейронное программирование обладает одним неоспоримым достоинством — природа в процессе эволюции уже построила огромное количество биологических прототипов, широкий спектр которых включает как нервные системы простейших многоклеточных, так и человеческий мозг, самую сложную из известных на сегодняшний день организованных систем, поэтому применение знаний биологии и нейрофизиологии может оказать существенную помощь при программировании искусственных нейронных систем.

Большой разброс в сложности моделируемых систем предполагает применение соответствующих технологий программирования. В такой же степени как и в алгоритмическом программировании методы построения больших операционных систем отличаются от методов написания небольших индивидуальных программ, в нейронном программировании процесс построения больших моделей существенно отличается от создания небольших по своему объему систем.

Технологии построения информационных систем относятся к области особых интересов в программировании. Среди многих работ в этом направлении можно выделить раннюю статью Э. Дейкстры "*The Structure of 'THE'-Multiprogramming System*" [7], в которой он сформулировал принципы построения программных систем, существенно отличающиеся от широко принятых тогда технологий программирования (см. например, Ф. Брукс "*Мифический человеко-месяц*" [3]). В этой работе Дейкстра рассматривает не только общие технологические принципы, но также вводит синхронизирующие примитивы — семафоры, при помощи которых ему удается решить проблему асинхронного распараллеливания процессов. Технология и новые структурно-функциональные решения в его работе взаимосвязаны, и это в совокупности позволяет ему получить решение, значительно опережающее известные в то время системы.

Представления Дейкстры о программировании, его элегантные теоретические и практические решения продолжают оставаться привлекательными для всех, кто связан с этой дисциплиной. В одном из своих последних интервью, голландской телекомпании *VPRO* в 2001 году, рассуждая о сути программирования, он говорит следующее:

> Существуют совершенно различные стили программирования. Я могу сравнить их с тем, как сочиняли музыку Моцарт и Бетховен. Когда Моцарт приступал к записи партитуры, композиция у него уже существовала в завершенном виде. Он записывал партитуру с первого раза набело. Бетховен был скептик и борец, который начинал писать до того, как он имел сложившуюся композицию... Для того, чтобы сочинять музыку, нужно уметь записывать ноты. Но быть композитором это вовсе не означает умение записывать ноты. Чтобы быть композитором, нужно чувствовать музыку.
>
> *http://www.cs.utexas.edu/users/EWD/videos/NoorderlichtVideo.html*

Дейкстра затрагивает особенную тему в программировании, связанную с индивидуальным творчеством и использует при этом аналогии с музыкой. Продолжая эту аналогию, можно сравнить необходимые для построения нейронных моделей интерпретирующие системы с различными музыкальными инструментами. Для исполнения больших и сложных музыкальных произведений в симфонических оркестрах

используются самые разнообразные группы инструментов. Для исполнения произведений камерной музыки, оказывается вполне достаточно одного. Среди инструментальных систем, предназначенных для индивидуального (камерного) программирования, особый интерес представляет система *HyperCard*. В этой системе удачно совмещены средства представления данных и система динамической интерпретации скриптов, что позволяет использовать ее как внешнюю оболочку-интерфейс к нейронным моделям. Архитектура системы *HyperCard* во многом послужила прототипом для веб-страниц, однако его возможности до сих пор во многих отношениях превосходят средства динамического программирования *DHTML*.

К сожалению, в 90-х годах система *HyperCard* была заморожена и в настоящее время практически прекратила свое существование. Причина в том, что руководство *Apple* не смогло понять и оценить ее стратегическое значение. В своем интервью, данном в 2003 году *CNET News*, Джон Скали, бывший в 80-х годах президентом компании *Apple*, рассказал об этом так:

> Если я обращусь назад, на то, что бы я хотел сделать по другому тогда, когда я был в Apple, я думаю что самая большая упущенная возможность — это система HyperCard. Она была создана в 1987 году первым программистом Apple — Билом Аткинсоном. Мы никак не могли понять, что же она из себя представляет на самом деле. Мы думали, что это была система для создания прототипов. Мы думали, что это была база данных. Был пример, когда она использовалась как интерфейс с протоколом TCP/IP для работы с суперкомпьютером Cray.
>
> У нас не хватило проницательности, чтобы понять, что все, что было внутри HyperCard, было на самом деле то, что позднее будет успешно разработано Тимом Бернес-Ли, с протоколами HTTP и HTML.
>
> *http://news.com.com/2008-7351-5085423.html*

В этом смысле между системой программирования *HyperCard* и нейронными методами моделирования есть

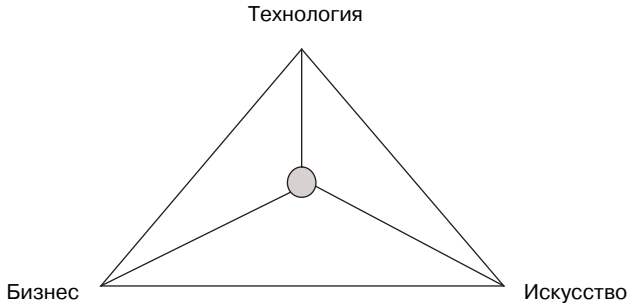
много общего. Нейронные модели в конце 60-х годов были определены как бесперспективные, и понадобились два десятилетия для того, чтобы интерес к этому направлению возродился вновь. Однако до сих пор этот интерес имеет скорее математический, чем программистский характер. Современные исследования свойств нейронных систем, в первую очередь, связаны с математическими задачами классификации и соответственно с нахождением оптимальных методов формирования весов межнейронных связей. Для программистов же, в первую очередь, интерес представляют динамические свойства нейронов, их уникальные способности к соединению, реконфигурации и размножению. Эти свойства нейронных моделей удивительно совпадают со свойствами Интернет, который является саморазвивающейся, децентрализованной системой и в которой происходит огромное количество параллельных процессов.

В основе Интернет лежит обмен сообщениями, которые могут быть представлены в самой разнообразной форме, что позволяет рассматривать его как большую интерактивную систему. При этом взаимодействие может происходить как с участием человека, так и при помощи различных ботов — искусственных представителей, способных автоматически просматривать содержимое веб-страниц, отвечать на вопросы посетителей, делать ставки на аукционах и т. п. Для построения таких ботов используются различные методы программирования, включая нейронное моделирование, которому в последнее время уделяется все большее внимание.

Результатом программирования — нейронного или алгоритмического — в конечном итоге является программный продукт, обладающий определенными коммерческими свойствами. По мере увеличения общего количества пользователей в Интернет, потребности рынка в высоко-технологичных системах, способных упростить процессы общения человека с компьютером, постоянно возрастают. Можно по-разному относиться к известному письму Билла Гейтса — "*An Open Letter to Hobbyists*", в котором он призывает к коммерческому

профессионализму в программировании (все программы, которые прилагаются к этой книге, относятся к категории *Open Software* и могут быть использованы свободно, в соответствии с общепринятыми нормами), однако тот факт, что на рынке к нейронным системам сегодня проявляется повышенный интерес, позволяет надеяться, что помимо профессионального любопытства, нейронное программирование сумеет привлечь коммерческий интерес разработчиков.

Судьба программного продукта зависит от множества разнообразных и зачастую противоречивых факторов. Творческие устремления и поиски интересных решений сталкиваются с коммерческими требованиями и технологическими ограничениями. Если представить процесс программирования, как постоянное нахождение упругого баланса между точками в пространстве, которые задают эти ограничения:



то искусство является одним из его важных составляющих. Искусство играет принципиальную роль в балансе между практическим смыслом и теоретическими ограничениями именно потому, что оно помогает находить компромиссы и соединять противоречивые взгляды в практике и теории [21]. Стремление к такому соединению, поиск различных подходов, методов и представлений, которые позволят решать все более широкий круг постоянно возникающих перед программистами задач, и является главной целью этой книги.

ЭНЕРГИЯ, ИНФОРМАЦИЯ И ЗНАНИЯ

В основе процессов преобразования как энергии так и информации лежат общие по своей сути принципы, что позволяет предположить, что практические конструкции энергетических машин и информационных систем должны обладать некоторыми подобными свойствами. Основываясь на этих аналогиях, мы попытаемся применить такие физические понятия как *работа*, *мощность*, *к. п. д.*, *принцип неопределенности* и другие к анализу информационных систем и надеемся, что накопленный в индустриальном мире опыт проектирования и эксплуатации механизмов и энергетических устройств поможет в проектировании информационных машин.

С развитием информационных технологий возникли такие виды деятельности как виртуальные предприятия, электронная торговля, дистанционное обучение, удаленная диагностика, информационно-поисковые сервисы и т. п., открывшие перед человечеством принципиально новые возможности, но одновременно с этим появился и ряд новых практических и теоретических проблем. Среди них — извлечение знаний из больших распределенных источников данных, удаление мусора из потоков сообщений, взаимодействие на естественных языках, интеграция разнородных каналов связи, и другие, которые в свою очередь вызвали необходимость пересмотра и расширения некоторых из методов и технологий современного программирования. Так например, увеличение доступных объемов данных во многих случаях не только

но и выполнять связанные с ним действия, существенно расширяет возможности интерпретации. Например, мы можем интерпретировать состояние группы нейронов как модель понятийного аппарата головного мозга, отвечающего за первичное восприятие слов или иных сигналов, поступающих из сенсорных групп. В ином случае мы можем рассматривать нейронные возбуждения как индикаторы экономических или производственных процессов — состояние запасов склада, загрузка оборудования, финансовые риски и пр. При этом интегральные характеристики, взаимовлияние и динамические свойства нейронных моделей позволяют существенно упростить представление данных, как для принятия решений в реальном времени, так и для применения математического аппарата, например, как методов оптимизации, прогнозирования, классификации и т. п.

Динамические свойства нейронных моделей позволяют упростить решение задач, связанных с лингвистическим анализом. Рассмотрим пример, в котором мы будем использовать группу взаимосвязанных нейронных слоев для решения задачи простейшего перевода слов из нескольких словарей, представляющих различные языки через промежуточный слой эквивалентных понятий.

Предположим, что *слой 10000* мы будем использовать для хранения слов, принадлежащих русскому языку (*словарь* ***RU***), а *слой 100001*, соответственно, для слов английского языка (*словарь* ***EN***). Введем третий слой *10002*, в котором мы можем задать смысловые значения слов, или *понятия (словарь* ***M****)*. Допустим, что каждому слову из словарей ***RU*** и ***EN*** соответствует единственное слово в словаре ***M***, тогда мы можем построить простейшую многослойную нейронную сеть, связывающую слова и понятия (см. рис. 39). Использование дополнительного слоя ***M*** позволяет в дальнейшем перейти к более универсальному понятию — *смысл* (*meaning*), в тех случаях, когда представление слова необходимо представить в независимом от

конкретного языка виде. Например, нейрон N_1 в слое ***M*** может представлять смысловое значение слова *ГЕНЕРИРУЕТ* из словаря ***RU*** и слова *GENERATE* из словаря ***EN***.

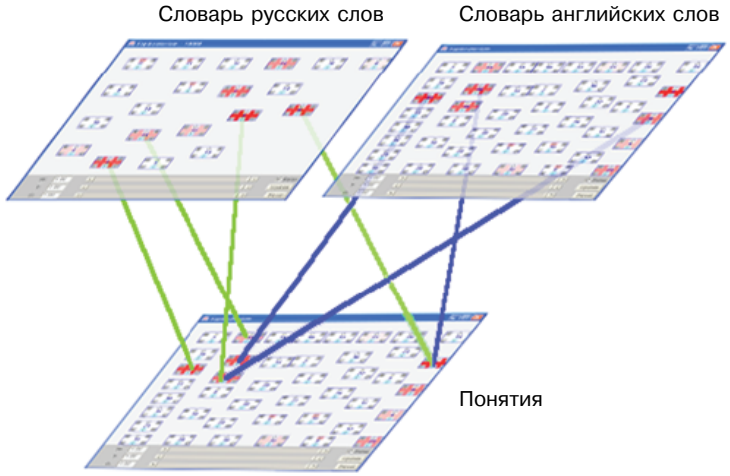


Рис. 39. Отображение слов на связанные с ними понятия
(Пример 3. НЭС http://www.nnod.com/np)

Использование смысловых значений, синхронизированных с входным потоком слов в живой речи, расширяет возможности лингвистических анализаторов, в которые можно встроить такой механизм. В этом примере используется поток слов из нескольких предложений, которые проверяются на принадлежность словарю, индексируются, и затем программа посылает сигнал возбуждения соответствующего нейрона. На рис. 40 показана схема взаимодействия между тремя слоями и программой выделения слов. Здесь предполагается, что соответствия между словами уже установлены, и веса этих соответствий одинаковы для всех слов.

В дальнейшем мы будем использовать множественные отношения между словами, и при этом весовые коэффициенты

этих отношений будут определяться в зависимости от их значимости и контекстов.

На этом примере можно проследить все основные особенности многослойных взаимосвязанных нейронных моделей. Возбуждения, развивающиеся на одном из нейронных слоев, могут переходить на другой слой, в котором они могут порождать новые потоки возбуждений, которые в свою очередь могут переходить на следующие слои и так далее. Взаимосвязные нейронные слои могут соединяться в очень сложные структурно-функциональные образования, поэтому приемы концептуального анализа и декомпозиции являются важными составляющими процесса проектирования, позволяя получить требуемый уровень детализации и представления. В некоторых случаях требования, предъявляемые к наглядности представления, могут существенно изменить внешний вид нейронной модели.

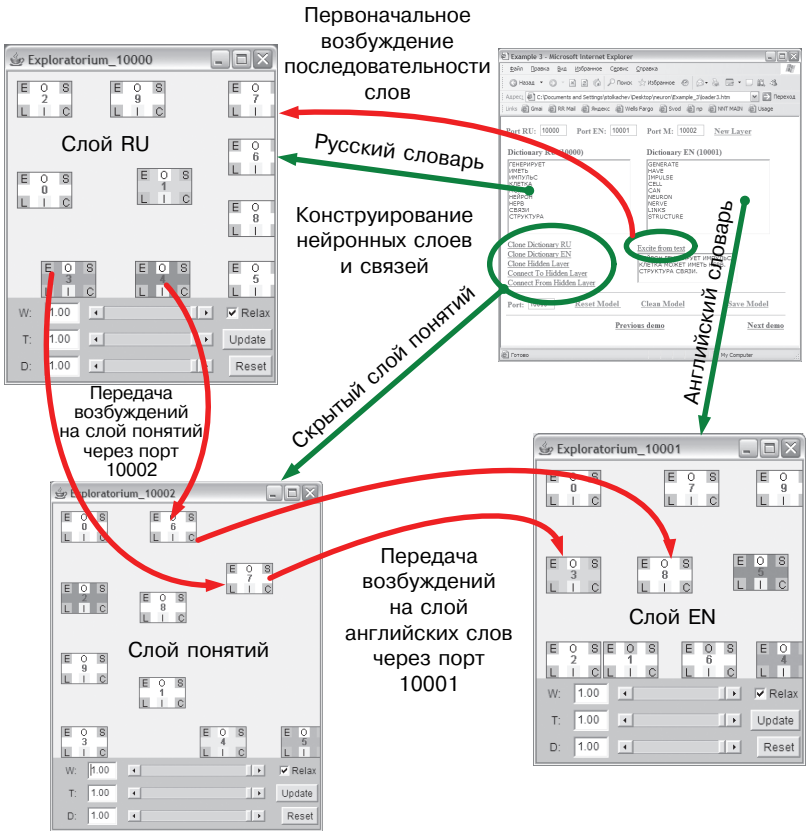


Рис. 40. Отображение слов на связанные с ними понятия
(Пример 3. НЭС http://www.nnod.com/np)

Слова являются элементарными единицами общения в человеческой речи, и для каждого слова существует множество известных значений, как правило, приведенных в различных словарях. Поступающие в нейронную модель последовательности слов, возбуждая соответствующие нейронные группы, приводят к изменению динамического пространства понятий отдельных слов и контекстных значений, образованных в процессе всей истории развития конкретной нейронной модели. В такой нейронной модели понимание можно определить как процесс нахождения наиболее оптимальной реакции в динамическом пространстве слов и контекстов. Внешнее проявление понимания всегда проявляется в виде реакции, поэтому уровень знаний системы можно, с точки зрения внешнего наблюдателя, свести к адекватности множества всевозможных реакций.

Так же, как для каждого отдельного слова, существует множество словарных значений, так и для различных словосочетаний существуют различные метазначения, а совместная совокупность наиболее вероятных значений определяет текущее состояние системы — понимание текущей фразы, которая в свою очередь, позволяет получить наиболее подходящую для этого понимания реакцию.

Очевидно, что глубина анализа словарных значений, как и значений словосочетаний, зависит от мощности нейронной модели. Чем больше вариантов значений в словаре системы и чем больше количество различных смысловых значений, связанных с различными словосочетаниями, тем глубже понимание и тем более качественным будет реакция системы

(рис. 41). Установление связей между значениями слов и словосочетаний связано с необходимостью иметь одновременный доступ к большому количеству различных информационных сечений нейронной модели. Визуализации и построению пользовательских графических интерфейсов, предназначенных для редактирования систем представления знаний, уделяется большое внимание особенно в последние

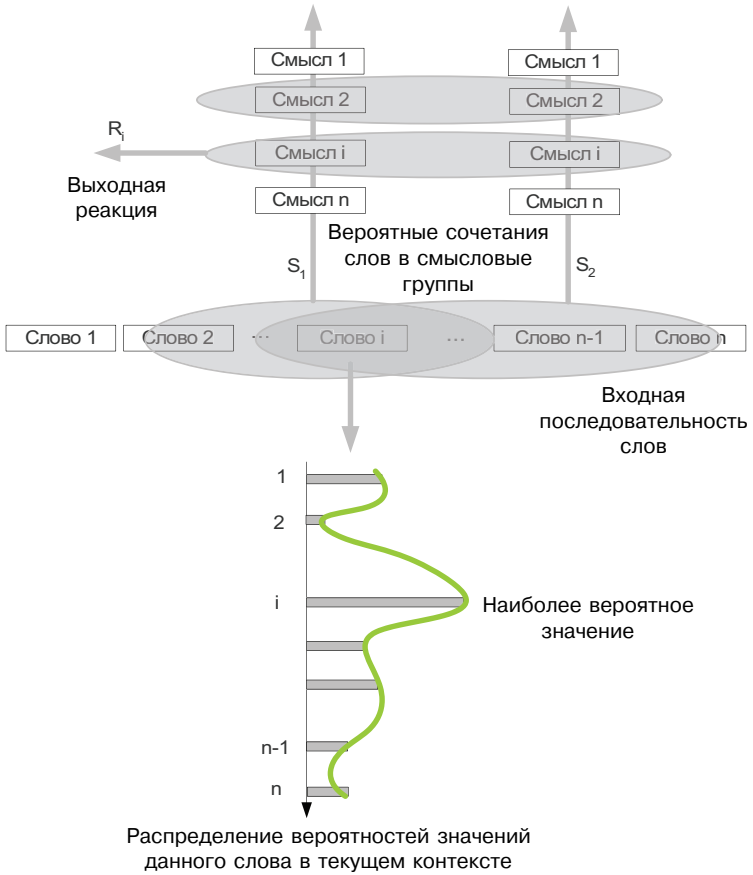


Рис. 41. Сочетание словарного смысла конкретного слова и смысла словосочетаний

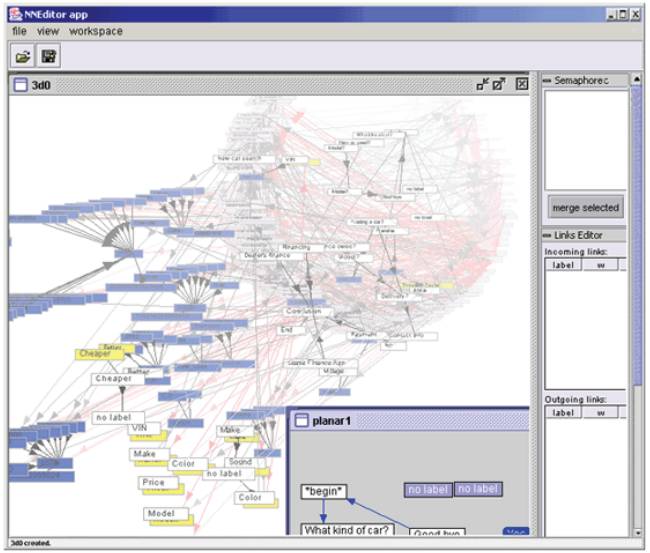


Рис. 42. Гравитационный редактор нейронных моделей
Copyright (c) 2005 NeMo Ltd. (http://kiberry.ru/nemo)

годы в связи с развитием мультимедийных методов представления графических данных.

На рис. 42 показан пример нейронного редактора *NN Editor* (автор А. Заборский), разработанный в компании *NeMo (http://kiberry.ru)* для моделирования диалогов с виртуальными агентами, размещенными на веб-страницах. В его основу положен принцип гравитации представлений [26], позволяющий выделять из большого количества взаимосвязанных нейронных структур наиболее близкие текущему пользовательскому представлению.

ПЕРЦЕПТИВНАЯ ЛИНГВИСТИКА

В середине прошлого века в физике и математике сформировались теоретические представления о границах возможностей получения точных или абсолютных знаний при проведении физических наблюдений и измерений свойств и характеристик объектов реального мира или в логических построениях и доказательствах, обосновывающих истинность или полноту каких-либо теорий. Помимо чисто теоретических результатов, представляющих интерес в первую очередь для физиков и математиков, в них содержатся универсальные методологические положения, которые могут быть использованы при решении прикладных инженерных, управленческих или экономических задач. Более того, эти результаты позволяют по новому понять определенные феномены в, казалось бы, очень далеких от формальных наук областях искусства — поэзии и живописи.

Принцип неопределенности Гейзенберга [12, 13] в физике гласит, что невозможно для частицы одновременно точно определить положение q и импульс p. Чем точнее мы определяем ее положение, тем меньше мы знаем об импульсе, и наоборот. Гейзенберг распространяет эти ограничения на основные динамические переменные, характеризующие физические системы — временные и пространственные координаты q и t; а также импульс и энергию p и E:

$$\Delta p \Delta q \sim h, \quad \Delta t \Delta E \sim h,$$

где h — постоянная Планка.

Можно считать, что методологический смысл принципа Гейзенберга заключается в том, что при проектировании систем мы должны постоянно учитывать эффект взаимосвязанных параметров, чтобы при увеличении точности определения одного параметра не потерять значение другого.

В математике в 1931 году Курт Гедель [11] доказал теоремы *о непротиворечивости и неполноте*, которые, если сформулировать их неформально, гласят, что любые непротиворечивые аксиоматические формализации в теории целых чисел содержат независимые высказывания, которые нельзя ни доказать, ни опровергнуть в рамках данной формализации и, если формализация в теории целых чисел непротиворечива, то она неполна. В программировании теоремы Геделя подтверждаются вполне конкретными практическими знаниями того, что в настоящее время не существует никаких способов доказательства корректности и результативности реальных программ, иначе как через их выполнение. Если считать, что эти теоремы верны применительно к методам построения формальных систем грамматического анализа естественных языков в математической лингвистике, мы никогда не сможем построить полную и непротиворечивую лингвистическую систему, способную в полном объеме понимать и интерпретировать живую речь на базе тех принципов, которые используются в их основаниях.

Можно найти множество подтверждений справедливости этих ограничений в программировании. Закон Брукса о том, что трудоемкость при разработке программ увеличивается в 10 раз по мере превращения простой программы в системный программный продукт, отражают тот факт, что стремление получить абсолютный и завершенный продукт неожиданно приводит к существенным затратам на, казалось бы, заключительном этапе интеграции. Брукс, анализируя изолированные программные системы на примере

операционной системы IBM OS/360, еще в шестидесятых годах показал, что не существует такого программного продукта, который можно было бы считать завершенным — постоянно изменяющаяся внешняя среда приводит к необходимости постоянно добавлять или модифицировать свойства программ.

Принцип неопределенности Гейзенберга и теоремы Геделя образуют конструктивную методологическую основу, используя которую проектировщики могут определить архитектурные ограничения и сконцентрироваться на направлениях, позволяющих получить практические результаты оптимальным путем. В задачах интерактивного взаимодействия значение таких ограничений трудно переоценить. Действительно, с одной стороны, существует постоянное стремление к повышению точности определения значений слов и понятий в классической лингвистике (языкознании), а с другой — аналогичное стремление к созданию полной формальной теории языков, в рамках которой можно было бы определить семантику и прагматику процесса общения в математической лингвистике.

При проектировании интерактивных систем с применением различных форм организации диалога программисту в одинаковой мере могут понадобиться знания формальных методов построения и использования грамматик, равно как и фонологии, лексики и семантики из теории языкознания. При этом для программиста, как и для прикладного специалиста в традиционных инженерных областях, принципиально важным условием успешного решения является нахождение «*золотой середины*» среди множества различных, зачастую противоречивых теоретических и прикладных знаний. Языкознание и математическая лингвистика в определенной степени напоминают физику и математику, поэтому аналогии между принципом Гейзенберга и теоремами Геделя нам нужны для того, чтобы при проектировании прикладных интерактивных систем мы могли использовать разумные ограничения и оптимальные

сочетания результатов исследований в языкознании вместе с формальными методами анализа и синтеза из математической лингвистики.

Так, при помощи разнообразных словарных систем (например, Wordnet [9]) мы можем создавать достаточно глубокие дефиниции и уточнения значений слов. При этом, словарные системы представляют лишь статические характеристики слов или словоформ в отрыве от их динамических связей, которые могут образовывать весьма сложный контекст, влияющий на комплексное восприятие и понимание в процессе живого общения. Принцип Гейзенберга применительно к лексическому анализу может быть сформулирован следующим образом: чем точнее мы определяем статическое значение конкретного слова в потоке живой речи, тем меньше мы знаем о его контекстном смысле. Точное статическое значение слова в динамическом окружения так же невозможно определить, как и установить точные координаты движущейся частицы:

$$\Delta m \Delta c \sim l,$$

где

- l — некоторая лингвистическая постоянная (шум);
- m — статическое значение слова;
- c — контекстный смысл.

Наличие вышеперечисленных ограничений позволяет существенно упростить моделирование интерактивных систем, однако при этом выбор и определение конкретных способов моделирования, степени формализации и глубины лексических представлений целиком зависят от индивидуального автора. И здесь можно вспомнить, что один из самых фундаментальных трудов по программированию называется — "*The Art of Computer Programming*". Монография «*Искусство программирования*» Дональда Кнута оказала существенное влияние на формирование программирования как научно-технической дисциплины. Кнут называет

программирование — искусством и считает, что программы могут содержать математические алгоритмы, быть оптимизированы при помощи математических методов, но при этом относиться к таким же произведениям искусства, как поэзия или музыка.

Искусство обладает удивительной способностью соединять несовместимое и создавать образы, передающие информацию, самым неожиданным способом. Неудивительно, что такая выразительная способность привлекала интерес многих математиков и представителей других точных наук. Среди этих исследований для нас особый интерес представляют работы Б. В. Раушенбаха по зрительному восприятию в живописи [35, 36], в которых он последовательно анализирует перцептивные и объективные модели пространства и их возможные представления в головном мозге человека. Рассматривая задачу художественного изображения объемного трехмерного мира на плоскости рисунка, Раушенбах приходит к неожиданным заключениям, которые могут оказаться весьма полезными при моделировании лингвистических процессов.

Зрительное восприятие основано с одной стороны, на хорошо изученных оптических преобразованиях в глазе, а с другой — в нем участвуют куда менее понятные процессы обработки зрительных образов в головном мозге. В физиологии, психологии, физике и математике можно найти множество разнообразных моделей, имеющих отношение к зрению, однако их объединение в одно целостное представление да сих пор является весьма сложной задачей. Математические модели зрительных образов рассматриваются в геометрии, где в зависимости от выбора аксиом можно получить геометрии Эвклида, Лобачевского, Минковского и других, в которых учитываются различные теоретические и объективные свойства реальности. Из психологии и физиологии хорошо известны факты о том, что внутреннее представление и объективный реальный образ могут существенно отличаться друг от друга. Человек восстанавливает трехмерное пространство, используя двухмерные зрительные сенсоры. Искажения, которые вносит головной мозг в зрительные представления, служат для более эффективного моделирования реальности, которое являясь индивидуальным, зависит от многих условий.

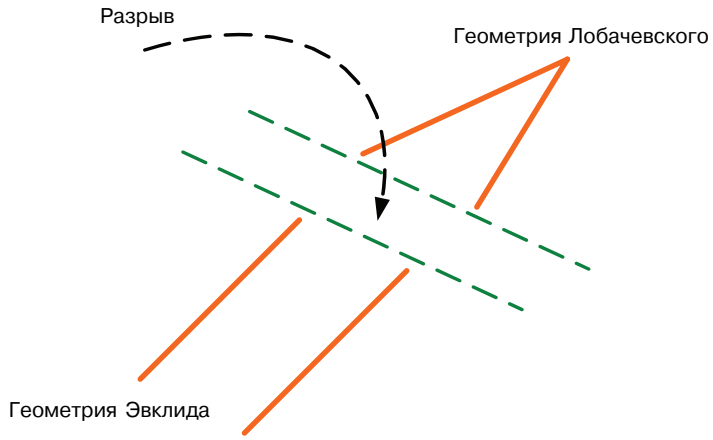


Рис. 43. Разрыв в двух геометрических представлениях в живописи

Если зрительное восприятие можно назвать прямой задачей, то художники решают обратную задачу отображения на двухмерной поверхности картины некоторых событий или свойств внешнего мира, которые в общем случае, помимо трехмерных геометрических свойств, могут включать динамику движения, время и целый ряд других факторов, совокупность которых создает *художественный образ*. Рассматривая процесс создания художественных изображений, Раушенбах анализирует свойства объективного и перцептивного пространств и приходит к интересному заключению: поскольку геометрические свойства реальных объектов и их перцептивных образов принципиально отличаются, художникам приходится постоянно находить

индивидуальные решения для создания иллюзии единства двух различных геометрических представлений. Анализируя свойства переднего и заднего планов картин, зрительного восприятия в целом, Раушенбах показывает наличие разрывов между двумя различными геометрическими моделями (рис. 43), которые художники, осознанно или интуитивно, соединяют, решая при этом проблему совмещения двух геометрий — Эвклида и Лобачевского.

По представлениям Раушенбаха существуют два основных способа передачи объективного пространства — чертеж, передающий объективную геометрию, и рисунок, передающий геометрию перцептивную. Таким образом, один и тот же предмет можно изобразить двумя способами — в чертеже и в рисунке, и при этом оба изображения будут верны, что и использовали художники, во все времена экспериментируя с их различными сочетаниями.

На протяжении многих веков художники, решая крайне сложные задачи моделирования многомерных образов, демонстрируют торжество здравого смысла над односторонними знаниями. Понимая парадоксальность определенных форм и методов, художники находят разнообразные искусственные приемы, позволяющие устранять разрывы и соединять несовместимые части в гармоничное целое. Искусство программирования в этом смысле можно в полной мере сравнить с искусством живописи. В информационных системах приходится преодолевать разрывы и сочетать методы и подходы в такой же степени противоречивые и казалось бы несовместимые как геометрии Эвклида и Лобачевского в живописи. При этом программистам приходится решать практически эквивалентную задачу отображения сложных многомерных динамических процессов в информационные представления, обладающие более низкой размерностью.

Введем в состав нашей информационной системы перцептивное пространство (рис. 44), в котором будет осуществляться моделирование представлений и знаний о внешнем мире. Для создания целостной системы, способной к осмысленному реагированию, нам необходимо будет объединить структурные компоненты математической лингвистики, словари и тезаурусы, классы и методы объектно-ориентированного программирования вместе с нейронными моделями.

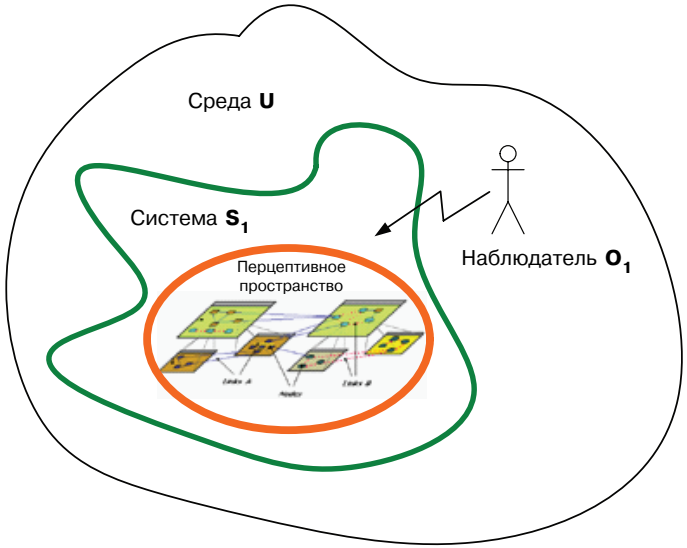


Рис. 44. Перцептивое пространство в интерактивной системе

Современные методы математической лингвистики с достаточной эффективностью позволяют анализировать и синтезировать сравнительно короткие фрагменты речи. С другой стороны нейронные модели и семантические сети позволяют создавать и поддерживать достаточно глубокие контекстные представления. Мы будем понимать под перцептивной лингвистикой объединение методов математической лингвистики и нейронных моделей, способное создавать и поддерживать перцептивные пространства в информационных системах.

Так же как проекции двухмерных зрительных образов позволяют воссоздать трехмерное изображение, а чертежные знаки и дополнительные детали (перспектива,

цвет, и т. п.) позволяют воссоздать полную картину реального объекта, нейронные проекции и связанные с ними параметрические таблицы и процедуры реагирования позволяют получить представление о многомерных свойствах информационных объектов, событий и процессов и их реакций.

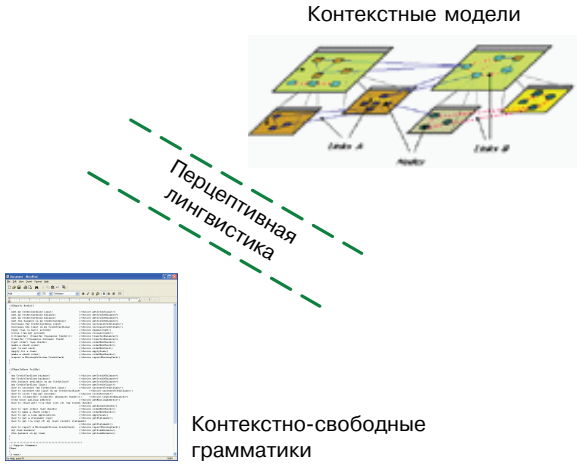


Рис. 45. Перцептивная лингвистика — объединение контекста и контекстно-свободных грамматик

Практическое воплощение перцептивной лингвистики в нашем случае представляет собой внутрисистемный интерфейс, позволяющий унифицировать поток сообщений/реакций и объединить контекстно-независимые грамматики с динамическим контекстом нейронных моделей. Например, мы можем использовать грамматики распознавания речи для анализа потока голосовых сообщений и при помощи нейронных моделей динамически расширять их в зависимости от конкретного контекста в процессе общения.

КОНТЕКСТЫ И ВРЕМЕННАЯ ЛОГИКА

В практике программирования хорошо известны примеры успешного применения нейронных моделей при решении задач классификации и распознавания образов. Однако искусственные нейроны могут быть не менее эффективно использованы для построения более широкого класса динамических систем, включая интерактивные и диалоговые системы. Как уже отмечалось выше, обмен сообщениями в процессе диалога приводит к изменению внутреннего состояния участвующих в этом обмене систем. При этом изменения могут происходить не только в системе, получающей сообщения, но и в системе, которая эти сообщения инициирует. Такие изменения, во всей их совокупности, образуют *контекст* конкретного взаимодействия. В общем случае в этом процессе может участвовать произвольное количество систем, однако мы ограничимся рассмотрением случая обмена сообщениями между двумя системами (рис. 46).

Очевидно, что синтез и анализ сообщений в этих системах должен происходить в зависимости от контекста, который, в свою очередь, может быть определен набором соответствующих параметров. В реальных системах контекст представляет собой сложную динамическую структуру. В него может входить информация о разнообразных характеристиках и свойствах обсуждаемой предметной области, а также об отношении к ним сторон, участвующих в обмене. Примерами контекста могут служить время, место, модель,

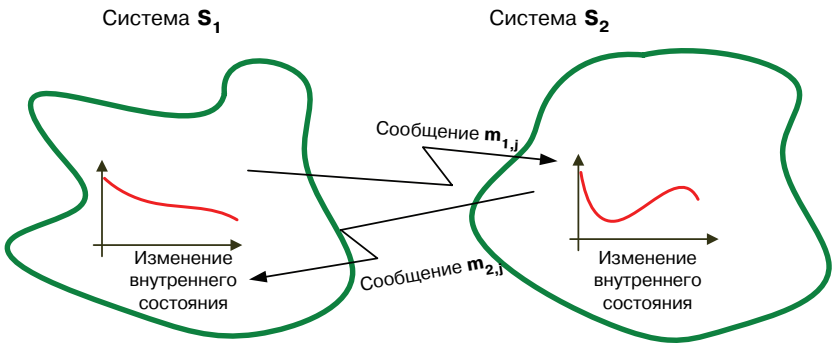


Рис. 46. Интерактивный обмен сообщениями между двумя системами

стоимость, в такой же степени, как и лингвистические свойства сообщения, такие, например, как принадлежность к той или иной лингвистической группе — вопрос, утверждение, приветствие и т. п.

Моделирование контекстов, отражающих смысл текущего разговора, является одним из ключевых факторов, определяющих эффективность интерактивного процесса. Рассмотрим некоторые особенности различных методов моделирования, которые могут оказаться полезными при практическом построении интерактивных систем.

Логическое моделирование. При логическом моделировании контекстов, например, в среде языков программирования, задача в конечном итоге сводится к нахождению множества логических функций, включающих переменные, характеризующие контекст взаимодействия. При изменении значений этих переменных, в результате вычисления логических функций может быть выполнена та или иная процедура P_i, отвечающая за реакцию соответствующего состояния системы. Если предположить, что существует n переменных, определяющих контекст системы S_i, тогда задачу вызова процедур P_i можно свести к нахождению и последующему решению логических функций:

$$\begin{aligned} &\textit{if } (f_1 (X_1, X_2, \ldots X_n)) \textit{ then } P_1 \\ &\textit{if } (f_2 (X_1, X_2, \ldots X_n)) \textit{ then } P_2 \\ &\ldots \\ &\textit{if } (f_m (X_1, X_2, \ldots X_n)) \textit{ then } P_m \end{aligned}$$

При использовании нечетких и временных логик для сравнительного небольшого и ограниченного набора переменных можно получить вполне приемлемые решения. Однако при увеличении количества переменных и взаимосвязей между ними происходит лавинообразное увеличение сложности, в результате чего общее решение на основе логического моделирования оказывается практически невозможным. В реальных системах количество переменных, влияющих на контекст, может насчитывать несколько тысяч, при том что количество связей между ними может быть такого же порядка. Наиболее сложным при программировании таких логических функций является открытый характер систем, что предполагает постоянное изменение как количества внутренних переменных, так и взаимосвязей между ними.

Аппаратное моделирование. При использовании аппаратных моделей, в распоряжении проектировщиков имеются различные функциональные и логические, в том числи и аналоговые, элементы — память, дешифраторы, задержки, преобразователи и т. п., в основе работы которых лежит последовательная логика. При проектировании аппаратных систем одним из наиболее критических факторов является время. Распространение сигналов, логика срабатывания функциональных элементов, переходные процессы при переключениях — все это имеет определенные временные характеристики, учет которых приводит к необходимости применения синхронизаций, стробирования, задержек и других способов, позволяющих получать логические решения в условиях реального времени.

Различия между аппаратным и логическим моделированием, связанные с фактором реального времени, имеют

фундаментальный характер. События в аппаратных системах имеют естественную асинхронную природу, в отличие от логического моделирования в программных средах, где практически всегда существует возможность упорядочить потоки событий при помощи подсистем планирования и управления очередями. Анализ временных диаграмм с целью их синхронизации является одним из основных методов проектирования в аппаратном моделировании. Для этого в процессе моделирования совмещают логические схемы и временные диаграммы. На рис. 47 приведен пример такого совмещения, где в качестве последовательного логического элемента выбран простейший двоичный счетчик.

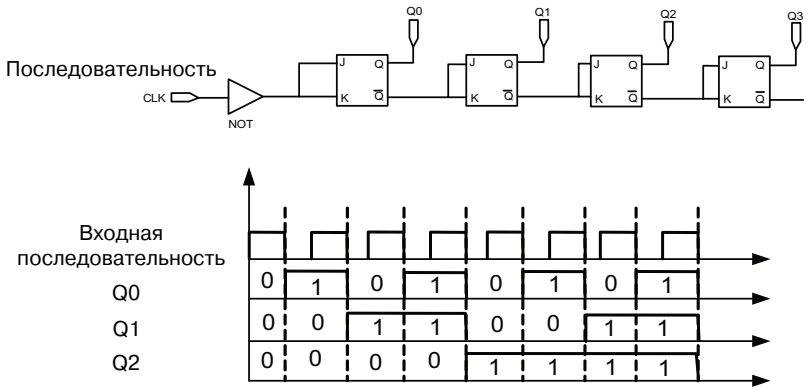


Рис. 47. Пример двоичного счетчика и его временная диаграмма (http://www.ets.ifmo.ru:8101/denisov/lec/lec4.htm)

Существует еще одна область человеческой деятельности, где формальные нотации и определенным образом связанные последовательности исполнений самым тесным образом синхронизированы в реальном времени. Речь идет о музыке, где используется своеобразная языковая формализация нотных записей, и, если так можно выразиться, технология проектирования музыкальных произведений, в общих

чертах, очень похожа на технологию аппаратно-программного проектирования логических устройств. Роль логики в музыке исполняет гармония, в которой определяются аксиоматические основания для законов объединения звуков и созвучий в последовательности, в рамках той или иной музыкальной теории. Как и в математической логике, в музыке существуют свои аксиоматики, языки, теории построения и интерпретации.

Если рассмотреть музыкальные нотации с точки зрения формальных языков, то можно считать, что ее алфавит состоит из множества нот определенной частоты и длительности. В отличие от классической операции конкатенации, которая определяется в теории формальных языков как операция соединения символов в последовательности, в музыкальных нотациях конкатенация является многомерной, то есть ноты могут быть соединены в цепочки во

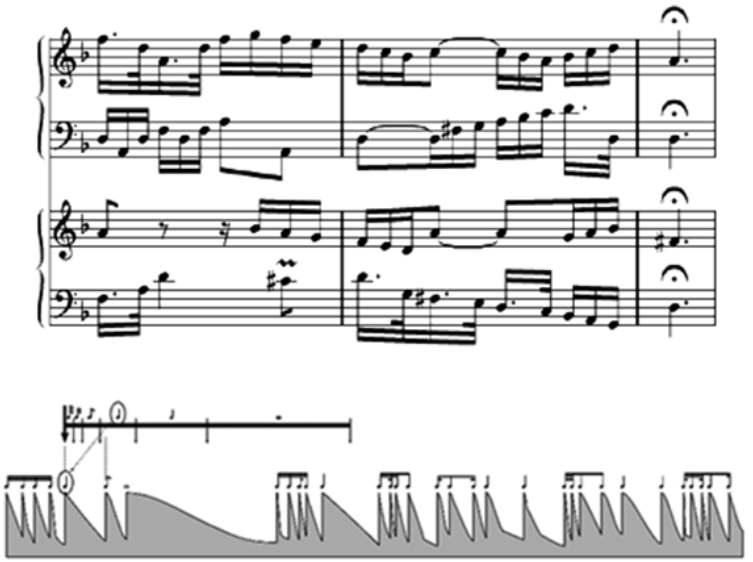


Рис. 48. Временное программирование в музыке
Copyright (c) 1999 James Ingram
(http://www.the-notation-of-time.de/Website1999-2004/devmus.htm)

многих направлениях. В целом же музыкальные произведения похожи на программы и их языковые конструкции могут быть интерпретированы параллельно и асинхронно.

Способы описания параллельных и асинхронных исполнений, применяемые в музыке, представляют особый интерес для нейронного моделирования. На рис. 48 показан пример музыкального произведения и временная диаграмма последовательности нот в форме амплитудно-временных характеристик из работы Джеймса Инграма "*The Notation of Time*" [14].

Исполнение музыкальных произведений можно рассматривать как множество развивающихся во времени параллельных процессов, порожденных на основании исходных нотных текстов. Каждый из этих процессов состоит из последовательности возбуждений и торможений некоторых музыкальных объектов, которые обладают в целом сравнительно небольшим динамическим набором хорошо формализуемых акустических свойств. Если считать, что каждый такой процесс последовательно возбуждает и затем определенным образом изменяет амплитуду звука в соответствии с партитурой, мы можем представить музыкальное исполнение в виде трехмерного пространства процессов, как это показано на рис. 49.

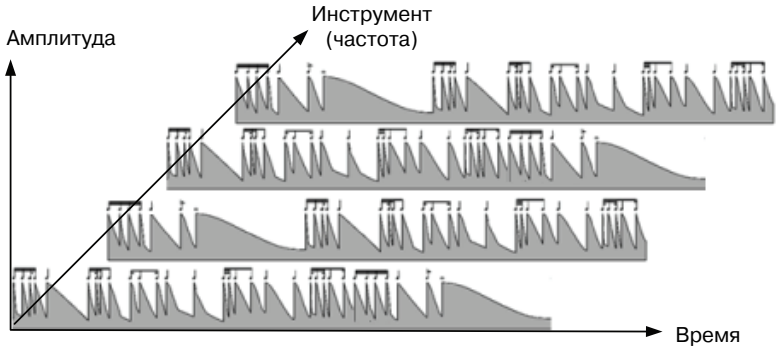
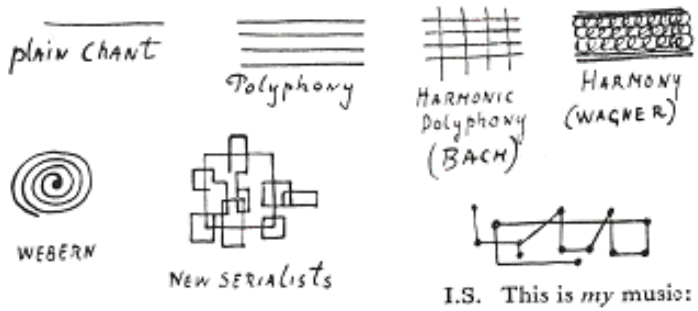


Рис. 49. Амплитудно-частотное представление исполнения музыкального произведения

Здесь будет уместно привести в качестве примера рисунок И. Стравинского из его книги *"Conversations With Igor Stravinsky"* [25], характеризующий его представление о гармонии и способах развития музыкальных тем как процессов, развивающихся во времени и пространстве.



Представления Стравинского о разнообразных формах построения музыкальных произведений могут быть использованы при организации диалога между человеком и информационной системой. Опираясь на его представления о динамике развития музыкальных произведений, можно получить множество различных архитектурных решений организации диалогов — простые линейные разговоры, параллельные линейные непересекающиеся темы, линейно-связанные темы, и т. п.

В нотах и их музыкальных интерпретациях сочетаются дискретные, формально-логические описания, которые задает автор и аналоговое параллельное исполнение, интерпретирующими эти партитуры музыкантами. Если сравнить рис. 47, где представлена амплитудно-частотная характеристика музыкального исполнения и рис. 24, на котором показано развитие возбуждений в нейронах, то можно увидеть, насколько эти две модели похожи. Нейроны обладают очень близкими к нотам свойствами и характеристиками, а их способность к возбуждению и торможению позволяет говорить об возможности применения

некоторых из формальных свойств музыкальных моделей в нейронном программировании.

Сравнительно простой нотный язык позволяют создавать необыкновенное разнообразие музыкальных композиций — от незатейливых мелодий до чрезвычайно сложных симфонических произведений, при этом в нем можно выделить несколько характерных отличий от свойств языков программирования.

1. Нотный алфавит включает в себя элементы, обладающие временными свойствами.
2. Конкатенации элементов алфавита в цепочки могут иметь несколько направлений.
3. Пространство, где осуществляется исполнение нотных записей, является многомерным.
4. Форма записи (нотный стан) обладает возможностью создавать большое количество параллельных процессов и синхронизировать их исполнение.

В нотном языке, как и любом другом, можно создавать конструкции, обладающие определенными свойствами, и проводить их анализ, применяя различные формальные или неформальные методы. Для нейронных моделей должен существовать не менее эффективный способ описания, временная логика и язык, в котором можно создавать временные конструкции, применяя которые можно проводить анализ и синтез нейронных структур. Фактическое подобие нейронных и музыкальных моделей позволяет воспользоваться некоторыми музыкальными приемами и методами, применительно к формальным методам описания и интерпретации нейронных систем.

По сравнению с музыкальными системами, нейронные модели обладают еще более высоким уровнем динамизма — в нейронных моделях сама «*партитура*» может развиваться и модифицировать свои свойства во времени. Способность нейронов к клонированию и динамическому установлению связей с другими нейронами позволяет говорить о программировании программ в полном смысле этого понятия.

Рассмотрим в качестве примера клонирование некоторых основных типовых нейронных структур. В вычислительной технике в качестве простейшего запоминающего элемента используется триггер — устройство из двух базовых логических элементов, способное сохранять одно из n устойчивых состояний и переключаться в зависимости от входного сигнала. Для построения триггера применяются обратные связи, соединяющие попарно выходы логических элементов, с входами, отвечающими за сброс данного элемента в нулевое состояние. На рис. 50 показаны простейший двоичный *R-S* (*Set-Reset*) триггер и его двоичный нейронный эквивалент.

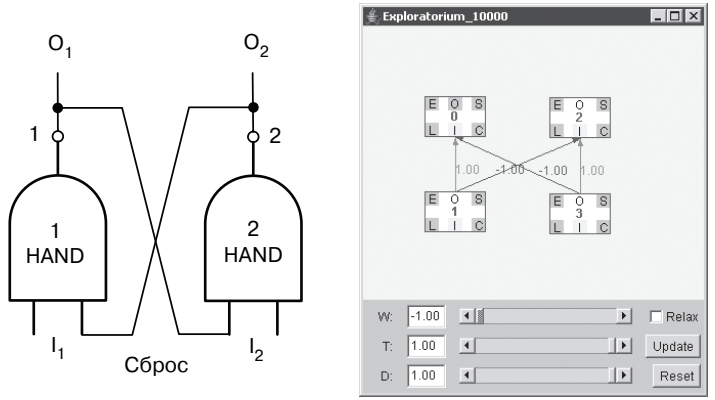


Рис. 50. Электронный триггер и его нейронный аналог

Для построения нейронного эквивалента двоичного триггера понадобятся четыре нейрона, соединенные попарно, причем прямые связи имеют веса: $w_{0,1}$ и $w_{2,3}$, равные **+1**, а обратные — $w_{0,3}$ и $w_{2,1}$, равные **−1**. При таком соединении, в случае возбуждения нейрона ***0***, что соответствует единичному сигналу на входе I_1 (предполагается, что пороговое значение нейрона равно или меньше уровня входного возбуждения), нейрон ***0*** распространит возбуждение через все свои выходы. При этом нейрон ***1*** получит

положительный единичный входной сигнал, который переведет его в возбужденное состояние, а нейрон ***3*** получит отрицательный единичный сигнал, который сбросит его возбуждение, если этот нейрон был возбужден перед этим. Точно так же работает нейрон ***2***. Таким образом, эта схема полностью повторяет работу триггера.

В отличие от абстрактного логического элемента, нейронный триггер имеет дополнительный параметр — время переключения, управляя которым можно получать различные временные задержки. Замкнув пару триггеров в кольцо, можно получить сетевую петлю — модель бесконечного цикла. Интерпретация петель, или бесконечные циклы, в нейронных моделях имеют практический смысл. Поскольку в них присутствует время, бесконечный цикл может быть использован как синхронизатор различных процессов и событий, поскольку большинство операций в нейронном программировании зависит от времени, в отличие от оператора в языках программирования, где большинство операций должны быть инвариантны ко времени.

Рассмотрим подробнее процесс создания нейронных компонент. Двоичный нейронный триггер легко расширить и получить многовходовой запоминающий элемент для ***n*** состояний. На рис. 51 показано решение для нейронного триггера, имеющего шесть состояний. Используя *НЭС*, мы можем достаточно легко построить двоичный нейронный триггер, клонируя протонейрон и последовательно соединяя пары нейронов в полученной группе, задавая веса связей соответственно **+1** и **−1**. Однако если мы захотим построить нейронный триггер, имеющий 100 состояний, ручное программирование вряд ли можно считать целесообразным. На первый взгляд может показаться, что эту задачу достаточно просто решить традиционными программными методами. Действительно, если существует программа-эмулятор, в которой поддерживаются нейронные модели, то очевидно, что внутри этой програмы возможно создать подпрограмму или метод, способные построить такой триггер, например следующим образом:


```
Number_of_States = 100;
for (i := 1; i <= Number_of_States*2; i++)
{
    make_New_Neuron(i);
}
for (i := 1; i <= Number_of_States; i++)
{
    for (j := Number_of_States+1; j <= Number_of_States*2;
i++)
{
            w = -1;
            if (j == i + Number_of_States) w = 1;
            make_New_Nerve(i, j, w);
    }
}
```

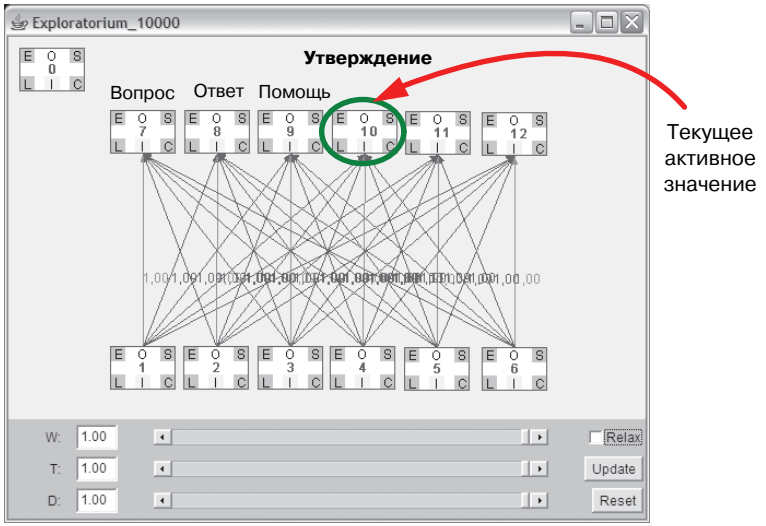


Рис. 51. Система из шести нейронных триггеров

Выполнение такой программы возможно либо непосредственно в теле нейронной оболочки, либо нам понадобится интерфейс, похожий на SQL/ODBC. В любом случае, если

программирование регулярных нейронных структур можно свести к достаточно простым программам, сложные нерегулярные структуры приведут к существенному усложнению алгоритмов и собственно программирования.

Используя механизм клонирования и временных задержек, это же решение может быть достигнуто иным способом.

Если соединить вместе временную логику клонирования и гармонические законы музыки, то программисты смогли бы создавать последовательности, управляющие развитием нейронных систем, и при этом проверку качества решения, отладку и выявление ошибок можно было бы осуществлять в той же форме, как это делают композитор и дирижер. Можно предположить, насколько эстетичнее был бы такой процесс программирования!

КЛОНИРОВАНИЕ

Способность нейронов к клонированию открывает возможности для создания сложных многосвязных структур по аналогии с тем, как это происходит с синтезом белков в биологических организмах. Перед тем, как перейти к анализу программных методов клонирования, рассмотрим, как происходит этот процесс в биологических системах. Можно отнести к самым выдающимся в истории человечества открытие Дж. Уотсоном и Ф. Криком [29] двойной спиральной структуры соли дезоксирибонуклеиновой кислоты (*ДНК*), которая лежит в основе воспроизведения биологических клеток. На рис. 52 показано структурное (*а*), молекулярное (*б*) и информационное (*в*) представления этой молекулы. Какими бы сложными ни были структура и организация цепи *ДНК*, с точки зрения программирования она представляет собой последовательность кодов, которые могут быть интерпретированы исполнительным механизмом — процессором, роль которого в клетке выполняют рибосомы.

Если считать, что рибосома — это исполнительный процессор, а молекула *РНК* — программа, то синтез белков является ничем иным, как параллельным исполнением последовательности кодов, записанных на носителе — ленте *РНК*, полученной путем копирования части другого носителя — ленты *ДНК*. Все остальные детали, связанные с процессом непосредственного химического синтеза, для программистов не представляют особого интереса, так же как и собственно

природа исполнения команд в любом процессоре — будь это биологическая рибосома или кремниевый *Intel 8080*.

На рис. 53 показан процесс синтеза белков из последовательностей троек нуклеотидов — кодонов, отвечающих за присоединение соответствующих аминокислот, из которых в конце концов и формируются белки. Количество кодонов можно считать соизмеримым с количеством кодов в сложных программах, таких например, как операционная система. Единственное отличие между современным процессором и внутриклеточными процессами — это скорость интерпретации. Можно считать, что в основе деления клеток лежит процесс интерпретации достаточно простых кодов, образующих очень длинные последовательности. При этом в биологических системах важную роль играют внешние факторы — ферменты, электро-магнитные поля и другие, которые можно представить как дополнительные параметры, влияющие на исполнение.

Если сравнить процесс синтеза белка в клетке и вычислительную систему Тьюринга [28], можно увидеть, что между ними существует много общих структурных-функцио-

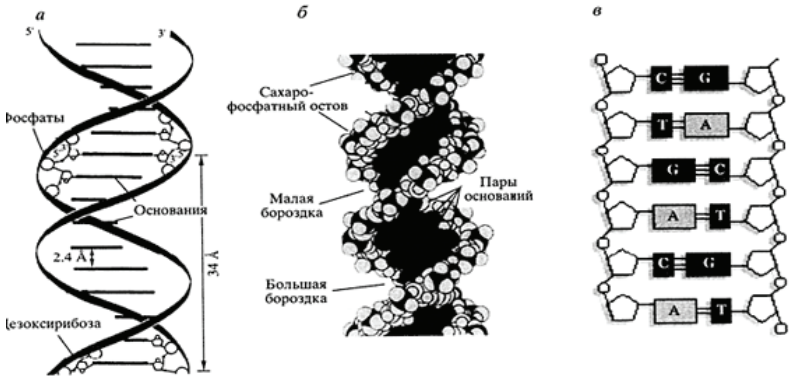


Рис. 52. Три представления ДНК
Copyright (c) 2003 Е.Д. Свердлов (http://www.ibmh.msk.su/vivovoco/VV/JOURNAL/VRAN/DNA/NA.HTM#zero)

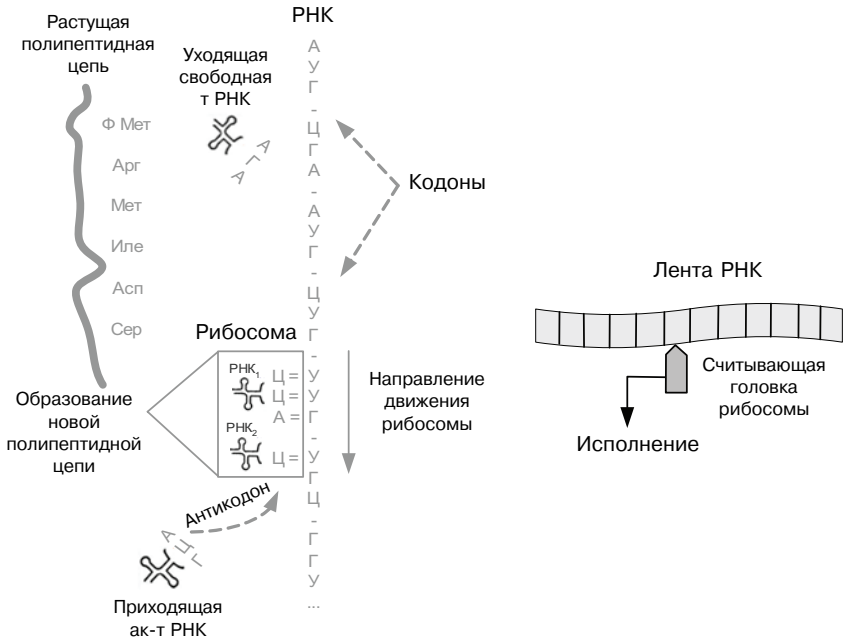


Рис. 53. Биологическая схема синтеза белка и ее программное представление
Copyright (c) Большая Советская Энциклопедия (трансляция в биологии) (http://encycl.yandex.ru/art.xml?art=bse/00080/15000.htm&encpage=bse)

нальных свойств. Система Тьюринга (рис. 54) включает в себя вычислительную машину и ленту, состоящую из множества ячеек, в которых могут быть записаны символы. В каждый момент времени вычислительная машина обрабатывает одну ячейку ленты, содержащую один символ — $S(i)$ и характеризуется состоянием q_i из конечного множества состояний q_n. Поведение машины определяется набором команд, которые определяются парой q_i, $S(i)$. В зависимости от конфигурации, в которой она находится, машина может сдвигать ленту вправо или влево, записывать или считывать символы в ячейки и изменять свое состояние (конфигурацию). Эта машина является элементарной системой

программирования, которую А. Тьюринг использовал в первую очередь для того, чтобы уточнить понятие алгоритма и вычислимости. Несмотря на чисто теоретический характер его работы и примитивность базовых операций, мы можем использовать модель машины Тьюринга для построения практического механизма клонирования искусственых нейронов.

Расширим внутренний механизм сенсорного реагирования нейрона, добавив к каждому сенсору входную очередь, способную хранить внутри себя последовательности временных задержек, которые будут определять время, по истечении которого, данный сенсор нейрона будет возбужден (рис. 55). Будем представлять входные сигналы как пару — время задержки и амплитуда (t_i, a_i). При этом, амплитуда входного сигнала должна быть нормирована от **−1** до **+1**, а время задержки должно быть задано в виде целого числа в миллисекундах. Например, мы можем загрузить в магазин, связанный с сенсором возбуждения, последовательность (*10 000, 0,5*), (*5000, 1*), (*2000, −1*). В этом случае нейрон возбу-

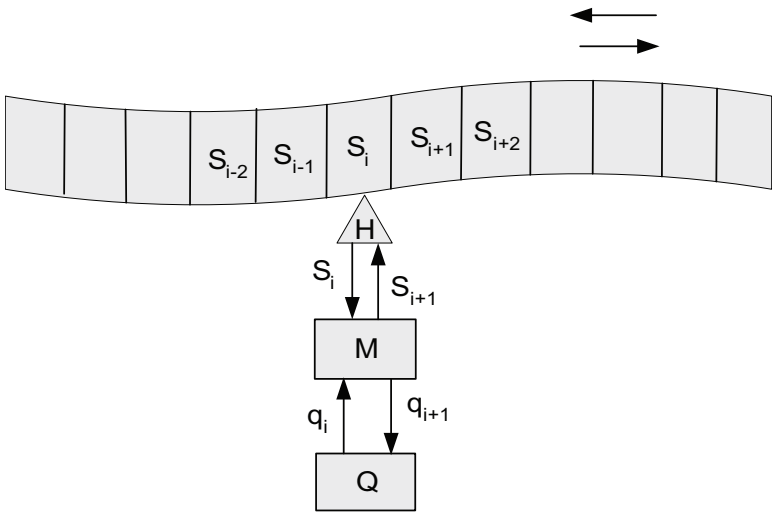


Рис. 54. Схема работы системы Тьюринга

дится последовательно три раза, через *10* секунд с уровнем возбуждения *0,5*, затем через *5* секунд с уровнем возбуждения *1* и еще через *2* секунды с уровнем возбуждения *−1*.

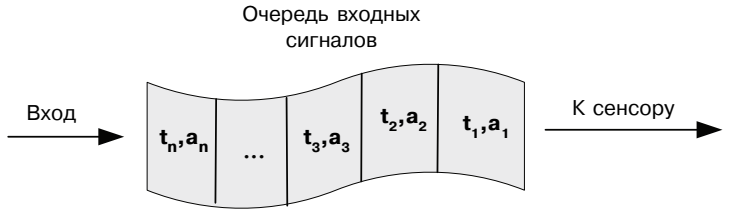


Рис. 55. Очередь входных сигналов

Очевидно, что добавление в модель нейрона очередей входных сигналов, обладающих временными характеристиками, предполагает наличие механизма внутренней и внешней синхронизации. Синхронизация является одним из фундаментальных свойств реальных систем, будь то биологические, социальные или технические устройства. Очереди сигналов, связанные с сенсорами внутри нейрона, позволяют перейти к построению более общей модели клонирования, способной к воспроизведению сложных структур, на основе сравнительно простых последовательностей кодов, которые могут быть представлены в виде ленты, по аналогии с биологическими и математическими лентами, рассмотренными выше.

Допустим, что в момент возбуждения сенсора *Клон* (см. рис. 33) нейрон, прежде чем выполнить операцию клонирования, постарается найти *подходящую* для него ленту, которую мы будем называть *Neuro_DNA* и, если такая лента существует, выполнить клонирование в соответствии с последовательностью кодов, записанных на этой ленте. Под такой лентой для нейрона будем использовать файл, который может находиться в определенном месте и иметь название: ***i.dna***, где ***i*** — номер нейрона. Ниже приведен фрагмент метода клонирования нейрона, где показано, как происходит чтение файла с данными для клонирования:

```
try
{
    String file_Name = this.id + ".dna";
    FileInputStream fstream = null;
    fstream = new FileInputStream(file_Name);
    BufferedReader in
    = new BufferedReader(new InputStreamReader(fstream));
    String theInput = "";
    while ((theInput = in.readLine()) != null)
    {
            ...
            ...
            this.cloneNode(v_S_DELAY, v_L_DELAY, v_C_DELAY);
      }
}catch (IOException e)
{
    this.cloneNode(null, null, null);
}
```

	Сенсор S	Сенсор L	Сенсор C
Нейрон 1	$(t_1,a_1),(t_2,a_2),...(t_n,a_n)$	$(t_1,a_1),(t_2,a_2),...(t_n,a_n)$	$(t_1,a_1),(t_2,a_2),...(t_n,a_n)$
Нейрон 2	$(t_1,a_1),(t_2,a_2),...(t_n,a_n)$	$(t_1,a_1),(t_2,a_2),...(t_n,a_n)$	$(t_1,a_1),(t_2,a_2),...(t_n,a_n)$
Нейрон 3	$(t_1,a_1),(t_2,a_2),...(t_n,a_n)$	$(t_1,a_1),(t_2,a_2),...(t_n,a_n)$	$(t_1,a_1),(t_2,a_2),...(t_n,a_n)$
...			
Нейрон N	$(t_1,a_1),(t_2,a_2),...(t_n,a_n)$	$(t_1,a_1),(t_2,a_2),...(t_n,a_n)$	$(t_1,a_1),(t_2,a_2),...(t_n,a_n)$

Рис. 56. Структура файла Neuro_DNA

Структура файла представляет собой последовательность строк (последовательность символов, ограниченная *CRLF* в *MS Windows*), каждая из которых в свою очередь состоит из трех последовательностей пар, разделенных *LF* (рис. 56).

Алгоритм клонирования нейронов чрезвычайно прост: прото-нейрон, инициированный для выполнения этой операции, начинает читать принадлежащий ему файл *Neuro_DNA*, по мере чтения очередной строки создает клон и загружает в его сенсорные очереди соответствующие последовательности пар задержек и амплитуд сигналов. *Neuro_DNA* можно также представить как линейную последовательность пар, разделенную на подпоследовательности двумя типами специальных символов — маркеров. Один из этих маркеров — ***NN*** (*New Neuron*) инициирует создание нового нейрона. Другие маркеры — ***QSS***, ***QLS*** и ***QCS*** (*Que to Select/Link/Clon Sensors*) определяют сенсор во вновь созданном нейроне, в очередь которого будет записана следующая за этим маркером группа пар.

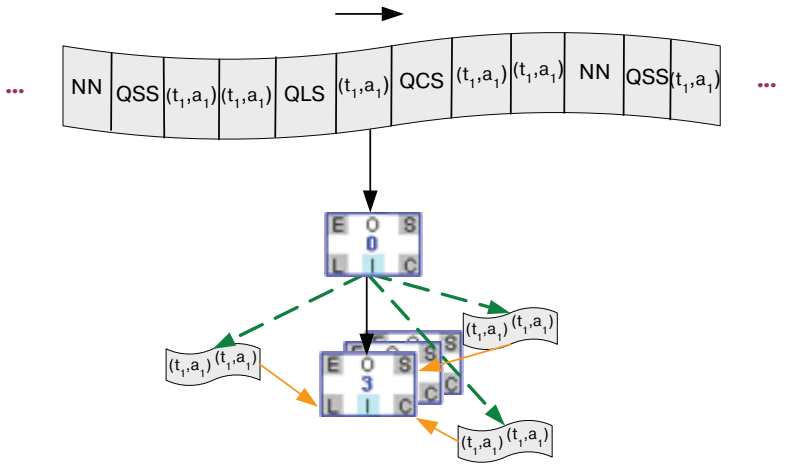


Рис. 57. Клонирование нейронов протонейроном в результате линейного считывания ленты Neuro_DNA

Конкретный пример файла *Neuro_DNA* для протонейрона ***0***, позволяющий создать нейронный триггер для шести состояний (см. рис. 51), приведен в табл. 3.

Если представить его в табличном виде в формате *Excel* этот файл имеет 12 строк и 3 столбца и в соответствии с этим протонейрон, выполняя операцию клонирования, построит 12 новых нейронов и поместит в сенсорные очереди каждого из них соответствующие последовательности пар задержек и амплитуд. Процесс клонирования, соответствующий структуре связей между 12 нейронами, приведен в Примере 4 *НЭС*. В этом примере если возбудить сенсор ***C*** (*Clone*) нейрона ***0***, возникнет группа, между элементами которой будут установлены связи в соответствии с временными задержками и весами, заданными в файле ***0.dna***.

Клонирование и одновременное создание связей между нейронами можно считать аналогом программирования, однако механизмы, которые лежат в основе нейронных процессов,

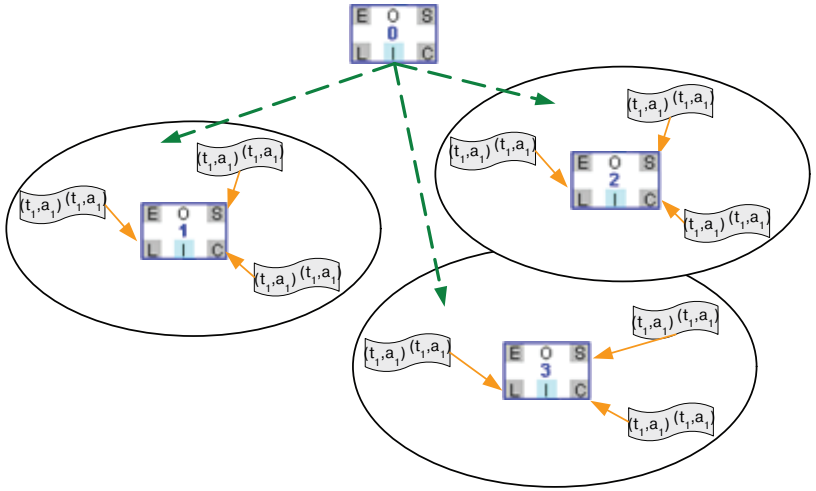


Рис. 58. Размножение нейронов с заданными временными параметрами

Таблица 3

Нейрон	Сенсор S	Сенсор L	Сенсор C
1	(0,0)	(1500,1),(5500,-1)	(0,0)
2	(0,0)	(10500,1),(5500,-1)	(0,0)
3	(0,0)	(20500,1),(5500,-1)	(0,0)
4	(0,0)	(30500,1),(5500,-1)	(0,0)
5	(0,0)	(40500,1),(5500,-1)	(0,0)
6	(0,0)	(50500,1),(5500,-1)	(0,0)
7	(1000,1),(14000,1),(10000,1),(10000,1),(10000,1)(10000,1)	(0,0)	(0,0)
8	(5000,1),(5000,1),(15000,1),(10000,1),(10000,1),(10000,1)	(0,0)	(0,0)
9	(5000,1),(10000,1),(5000,1),(15000,1),(10000,1),(10000,1)	(0,0)	(0,0)
10	(5000,1),(10000,1),(10000,1),(5000,1),(15000,1),(10000,1)	(0,0)	(0,0)
11	(5000,1),(10000,1),(10000,1),(10000,1),(5000,1),(15000,1)	(0,0)	(0,0)
12	(5000,1),(10000,1),(10000,1),(10000,1),(10000,1),(5000,1)	(0,0)	(0,0)

одинаковы как для создания функциональных структур, так и для их исполнения. В нейронных моделях при развитии и модификации внутренних свойств используется тот же механизм возбуждения, что и при реакции на внешние сигналы. Если в вычислительных системах программа и компьютер имеют принципиально различную природу, в нейронных системах создание функциональных структур и их интерпретация осуществляются одинаковым образом. При этом нейронная система является высоко параллельным образованием, после размножения каждый нейрон начинает свой собственный процесс развития.

На рис. 59 показана временная диаграмма образования соединений между нейронами, образующими триггер для шести состояний. Красным цветом показаны возбуждения, в результате которых будут построены положительные связи, а голубым — последовательности возбуждений, которые приведут к созданию отрицательных связей. Период нахождения нейрона в состоянии *«Выбран»*, в которое нейрон переходит при возбуждении сенсора ***S***, здесь принят равным *5* сек или *5000* мсек. В данном примере используются относительные временные интервалы, и для согласования возможных потерь в процессе интерпретации (аналогичные переходным процессам в реальных физических схемах) мы будем использовать компенсирующие временные поправки, которые для данного примера равны *500* мсек.

Предполагается, что время клонирования всех нейронов пренебрежимо мало, и в некоторый момент, все они запускают внутренние процессы обработки очередей событий, созданных протонейроном в момент их создания. Через *1000* мсек после рождения нейрон *7* перейдет в состояние *«Выбран»*, в котором он будет находиться *5000* мсек. Через *1500* мсек, нейрон *1* возбудит сенсор ***L*** (*Построить связь*) и в этот момент будет образована положительная связь между нейроном *1* и нейроном *7*. Через *5000* мсек возбудятся нейроны *8–12* и через *7000* (*1500* + *5500*) мсек снова возбудится нейрон *1*, в результате чего будут установлены отрицательные связи между нейроном *1* и нейронами *8–12*, и так далее до тех пор, пока не будут образованы все связи этого триггера.

Период времени, в течение которого нейрон имеет возбужденное состояние, может быть выбран за основу для определения базовой частоты синхронизации в модели в целом. В дальнейшем мы будем использовать минимально возможное время, в течение которого нейрон обладает способностью к соединению — τ, как специальную величину, которая зависит от конкретной вычислительной системы и может быть вычислена эмпирически. Например, в нашем случае $\tau = 10000$ мсек.

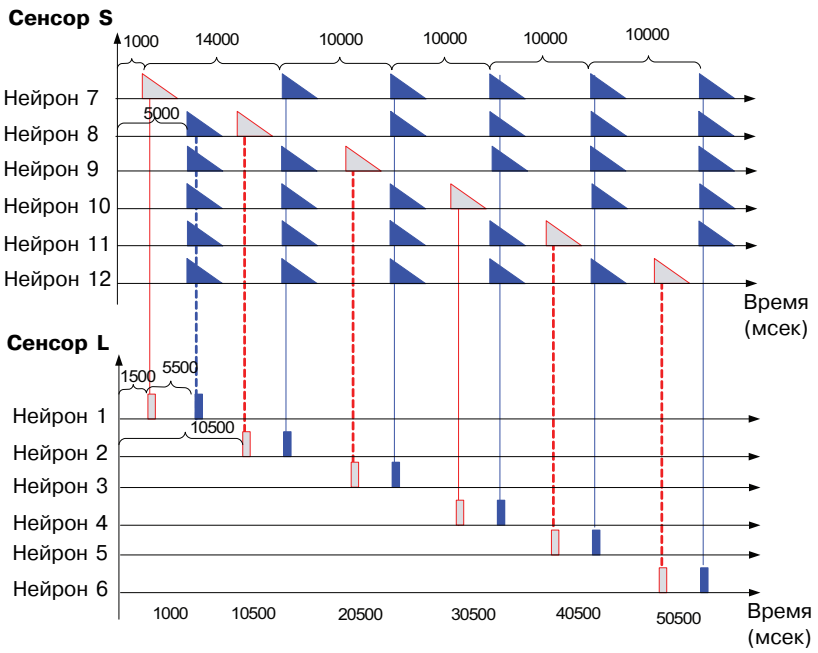


Рис. 59. Временная диаграмма построения нейронного триггера для шести состояний

Таким образом, поведение каждого нейрона может быть представлено в виде последовательности событий, которая может быть синхронизирована с другими нейронами как при помощи относительных временных задержек, так и путем введения глобальных стробирующих последовательностей. Мы будем в дальнейшем использовать комбинированные методы синхронизации, по возможности не усложняя структуру искусственного нейрона без необходимости.

СЕЛЕКТИВНЫЙ КОНТЕКСТ

Какими бы развитыми не были формы общения между человеком и интерактивными системами, без их информационного наполнения предметными данными такие системы имеют весьма ограниченную область применения. Нейронные модели позволяют создавать системы, способные поддерживать сложные динамические контексты. Примером таких систем являются диалоговые и поисковые системы, в которых фразы или поисковые запросы в значительной степени зависят от контекста. Наибольший практический интерес для моделирования виртуальных представителей представляет объединение поисковых методов с диалоговыми формами. Рассмотрим в качестве примера разговор *Покупателя* и *Продавца* в гипотетическом виртуальном магазине:

Покупатель: *Я бы хотел приобрести цифровую камеру.*

Продавец: *Вот, например, несколько камер, с различными качествами. Какая из них Вам больше нравится?*

Покупатель: *Вот эта камера меня устраивает, но нет ли у вас подобной за меньшую стоимость?*

Продавец: *Вот несколько более дешевых моделей.*

Покупатель: *Я бы хотел похожую, только от Sony.*

Продавец: *Пожалуйста, вот эти камеры произведены компанией Sony.*

Покупатель: *Это то, что меня интересует, но мне нужна камера с лучшим увеличением.*

Продавец: *У камеры Sony MVC CD и разрешение прекрасное, и запись осуществляется непосредственно на цифровой диск.*

Продавец: *Да, все хорошо, только тяжеловата. Нет ли похожей, но получше...*
и т. д.

В этом примере можно найти все основные проблемы, с которыми сталкиваются разработчики интерактивных справочных и коммерческих систем в интернете. Чем более образован и информирован покупатель или пользователь, чем шире спектр свойств продукции — тем сложнее процесс выбора, тем больше времени требуется для продавца или специалиста по обслуживанию для удовлетворения запроса и соответственно, тем дороже стоимость обслуживания. Современные формальные методы анализа и синтеза естественных языков не в состоянии предложить решения, позволяющие экономически эффективно конструировать подобные диалоги, в первую очередь потому, что практически каждая фраза в них является контекстно зависимой, с одной стороны, а с другой — данные о продукции должны быть получены из конкретной, как правило, реляционной базы данных, для которой запросы должны быть сформулированы в классической логической форме.

Необходимость соединять данные из реляционных баз вместе с лингвистическими конструкциями, зависящими от контекста, побуждает нас искать новые синтетические методы построения интерактивных систем. Рассмотрим структуру и потоки данных в гипотетической системе, которая способна вести вышеприведенный диалог.

Предположим, что простейшая структура базы данных "*Cameras*" для цифровых камер может быть задана следующими таблицами:

Model	*— название камеры (LS443, EasyShare DX4330...)*
Make	*— производитель (Sony, Canon, Fuji...)*
Price	*— цена ($400, $1200,...)*
Description	*— описание*
ZOOM	*— коэффициент увеличения (2x, 4x, 12x...)*
Matrix	*— матрица (2MP, 3MP, 4MP...)*
SCU	*— идентификационный номер*

Можно предположить, что *Продавец*, знакомый с *SQL*, мог бы, используя терминал доступа к базе данных, в процессе вышеприведенного разговора посылать запросы и передавать *Покупателю* результаты. Очевидно, что для такого типа структур баз данных может существовать сравнительно небольшое число вариантов запросов, которые могут удовлетворить большинство покупателей, если заданы конкретные значения переменных, характеризующие текущий контекст данного разговора. Например:

```
SELECT * FROM Cameras WHERE Price < $800 AND Price > $300

SELECT * FROM Cameras WHERE Price < $800 AND Price > $300
    AND MAKE = 'Sony'

SELECT * FROM Cameras WHERE Price < $500 AND Price > $200
    AND MAKE = 'Sony' AND ZOOM > 3
...
```

Если *Продавец* будет помнить о производителе, ценовом диапазоне и других характеристиках, которые интересуют *Покупателя*, он всегда сможет сформулировать следующий запрос к базе данных. Однако для того, чтобы ответить на такие, например, вопросы как *«Нет ли похожей, но получше...»*,

нам придется расширить структуру базы данных и добавить в нее лингвистические (*Fuzzy*) значения [32].

Добавим в структуру нашей базы данных для каждой переменной, которая имеет числовое выражение, ее нормированное представление. Так например, для переменной *Цена* (*Price*), создадим нормированное значение *Price_Norm*, которое может быть вычислено следующим образом:

$$Price_Norm_i = (Price_i - Min_Price) / (Max_Price - Min_Price),$$

где

$Price_Norm_i$ — нормированное значение ($0 > Price_Norm > 1$),
$Price_i$ — значение цены для i-ого объекта,
Max_Price — максимальная цена среди множества всех цен,
Min_Price — минимальная цена среди множества всех цен.

Такие значения могут быть получены для каждой переменной, имеющей числовой тип, включая дату, стоимость, вес и т. п. Пример базы данных, в которой вычислены нормированные значения для нескольких переменных, определяющих свойства цифровых камер, приведен в табл. 4.

Как видно из этого примера, мы можем использовать размытые значения для таких переменных, как *Цена*, *Коэффициент увеличения*, *Матрица* и т. п. На рис. 60 показана последовательность нескольких записей из базы данных "*Cameras*" в виде карт. Карта представляет собой графический интерфейс к базе данных и содержит поля, позволяющие отображать и модифицировать записи в базе данных. Кроме этого, карта позволяет отобразить нормированную интегральную характеристику каждой конкретной камеры. Таким образом, карта позволяет соединить вместе точные лингвистические характеристики, которые могут быть заданы ключевыми словами — *Sony*, *CD400* и размытые приближенные значения, позволяющие ввести лингвистические переменные — *больше*, *среднее*, *лучше* и т. п., для которых могут быть заданы области в соответствующих группах нормированных значений.

Таблица 4. Фрагмент базы данных "*Cameras*"

Model	Make	Price	Mtrx	ZOOM	Price_N	ZOOM_N	Mem_N	SCU
LS443	Kodak	$429,99	4	3	0,31	0,29	0,69	41778439289
EasyShare DX4330	Kodak	$296,99	2	3	0,20	0,29	0,20	41771580506
EasyShare LS420	Kodak	$270,99	2,1	2	0,17	0,14	0,22	41778963470
EasyShare CX4230	Kodak	$198,49	2	3	0,11	0,29	0,20	41778843710
Pocket	Logitech	$95,49	1,3	1	0,02	0,00	0,02	97855014672
DiMAGE F100	Minolta	$417,49	4	3	0,30	0,29	0,69	43325992964
DiMAGE 7i	Minolta	$999,00	5	7	0,82	0,86	0,94	43325992988
DiMAGE S304	Minolta	$573,49	3,3	4	0,44	0,43	0,52	43325992247
DiMAGE 7Hi	Minolta	$1196,99	5	7	1,00	0,86	0,94	43325993374
DiMAGE X	Minolta	$389,49	2,1	3	0,28	0,29	0,22	43325992780
Coolpix 4500	Nikon	$631,99	4	4	0,49	0,43	0,69	18208255030
Coolpix 5700	Nikon	$1047,99	5	8	0,87	1,00	0,94	18208255047
Coolpix 885	Nikon	$469,49	3,2	3	0,35	0,29	0,50	18208255054
MVC-CD400	Sony	$890,99	2	3	0,73	0,29	0,20	27242606487
MVC-CD250	Sony	$595,99	2,1	3	0,46	0,29	0,22	27242606524
MVC-CD300	Sony	$858,99	3	3	0,70	0,29	0,45	27242589223
MVC-CD200	Sony	$578,99	2,1	3	0,45	0,29	0,22	27242589247
CyberShot DSC-S75	Sony	$494,99	3,3	3	0,37	0,29	0,52	27242589278
CyberShot DSC-P2	Sony	$390,99	2	3	0,28	0,29	0,20	27242607354

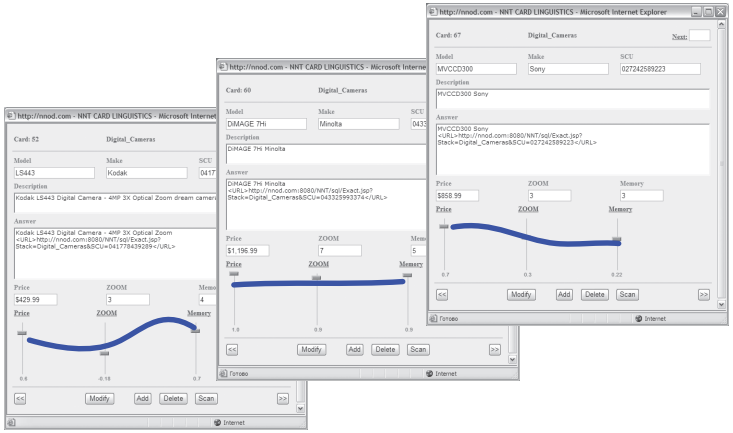


Рис. 60. Последовательность карт, содержащих характеристики цифровых камер

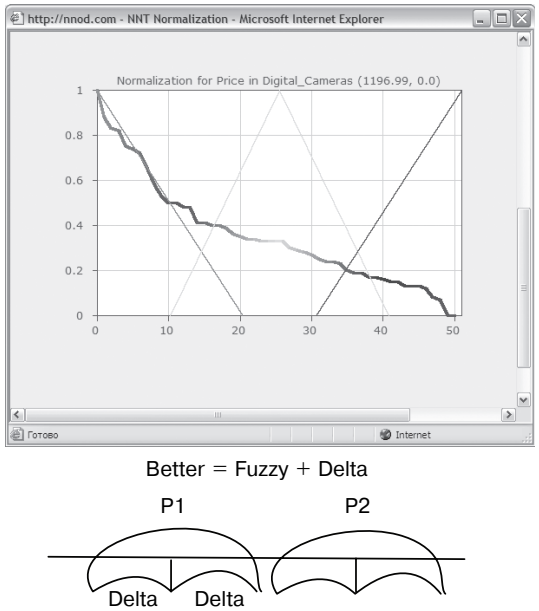


Рис. 61. Точные и размытые запросы к базе данных

Сочетание точных и размытых характеристик в рамках общей модели данных позволяет существенно расширить возможности интерфейса между пользователем и информационными системами. Реляционные базы данных, традиционно используемые для хранения и поиска данных о разнообразных продуктах, могут быть также использованы для хранения размытых характеристик. Если считать, что точные характеристики могут быть заданы группой ключевых значений $\boldsymbol{K_1, K_2, \ldots K_i, \ldots K_n}$, а размытые значения могут быть заданы группой нормированных значений $\boldsymbol{F_1, F_2, \ldots F_i, \ldots F_n}$, то запросы в базу данных для нахождения требуемой записи могут быть объединены вместе и представлены в общем виде как запрос:

SELECT * ***FROM*** *table*
WHERE KEY_i ***IS LIKE*** $\%K_i\%$ ... ***AND*** ...
$(FUZZY_i < F_i + delta_i$ ***AND*** $FUZZY_i > F_i - delta_i)$ ***AND*** ...,

где

KEY$_i$ — в таблице базы название поля, содержащего ключевые характеристики;
$\boldsymbol{K_i}$ — значение характеристики (точное);
FUZZY$_i$ — в таблице базы название поля, содержащего ключевые характеристики;
$\boldsymbol{F_i}$ — значение характеристики (размытое);
delta$_i$ — диапазон области в размытых характеристиках.

Такого вида универсальный запрос удовлетворяет широкому кругу потенциальных вопросов, примеры которых были приведены выше. Диапазон в области размытых характеристик (***delta***) задает границы примерной области значений, которая может удовлетворять конкретный вопрос пользователя. Эта величина выбирается исходя из практических соображений и в наших примерах равна 0,1, что составляет 10% от положительного диапазона размытых значений (рис. 61).

Таким образом, в каждый момент времени существует множество конкретных характеристических значений, которые задают представление системы о том, что подразумевает *покупатель* и на основании которых можно выбрать из базы данных группу записей, удовлетворяющих этим представлениям. Мы будем называть такую группу *селективный контекст*. В процессе взаимодействия между человеком и информационной системой *селективный контекст* будет изменяться, но мы будем считать, что всегда возможно выбрать из базы данных группу записей (которая может быть и пустой), соответствующую текущему *селективному контексту*.

Введем для каждого ключевого $K_1, K_2, \ldots K_i, \ldots K_n$, и размытого $F_1, F_2, \ldots F_i, \ldots F_n$ значения в нейронной модели соответствующий нейрон N_j, возбуждение которого, определяет: будет ли включено данное характеристическое значение в универсальный *SQL* запрос. Более того, для размытых характеристик уровень возбуждения нейрона может соответствовать величине F_i размытого значения, и в этом случае:

$$F_i = E(N_j),$$

где

- N_j — нейрон, соответствующий размытому нормированному значению F_i,
- $E(N_j)$ — уровень возбуждения нейрона, соответствующего данной размытой характеристике.

Для поддержания *селективного контекста* можно использовать нейронные триггеры, которые позволяют хранить активные динамические состояния, отражающие текущее состояние процесса взаимодействия. С их помощью можно создавать как традиционные статические двоичные запоминающие элементы, так и динамические элементы, возбуждение которых может изменяться во времени в соответствии с внутренним механизмом разряда.

На рис. 62 показан пример структуры нейронных слоев, содержащих две группы — контекстные триггеры для поддержания текущих значений одной таблицы базы данных — точных характеристик (в данном примере — *Производитель/Make*) и единичных нейронов, хранящих текущие значения для размытых характеристик (*Price_N*, *Zoom_N* и т. д.).

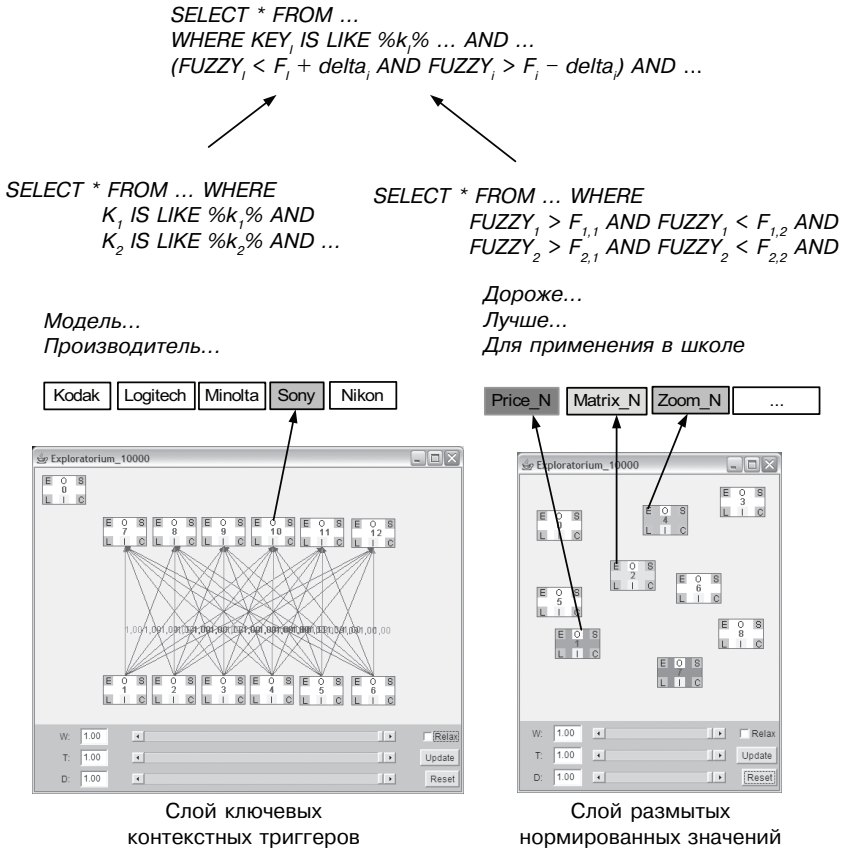


Рис. 62. Нейронные слои, поддерживающие точные и размытые запросы к базе данных

ОБРАТНЫЕ СВЯЗИ В ИНТЕРАКТИВНЫХ ПРОЦЕССАХ

В конечном итоге, эффективность интерактивной системы определяется информативностью сообщений, которые эта система способна генерировать в процессе диалога. Поскольку в диалоге участвуют как минимум две системы, для того чтобы диалог между ними имел смысл, каждая система должна оценивать качество создаваемых сообщений и на основании этого осуществлять соответствующие корректировки своего поведения. Природа нейронных моделей дает нам возможность сделать допущение, что хотя бы и в первом приближении, но две системы (человек — компьютер), участвующие в диалоге, воспринимают сообщения на основе одинаковых принципов. Такое допущение позволяет разработать механизм, способный получить в компьютере приближенную оценку связанных с данным разговором информационных процессов, происходящих внутри человека.

Как уже отмечалось выше, *информация есть мера, характеризующая изменения, которые происходят внутри системы в результате получения сообщения.* Такое, на первый взгляд, очень общее определение, оставляет нам большую свободу в выборе методов нахождения этих изменений. Например, для вычисления количества информации можно использовать традиционные способы, основанные на определении информации как меры изменения неопределенности системы. Нейроны представляют собой вычислительные элементы, которые идеально подходят для регистрации и хранения таких изменений. Результат внешнего воздействия в таком случае может быть вычислен как разница между возбуждениями нейронов в определенной области модели до и после этого воздействия.

Предположим, что система S_1 инициирует процесс взаимодействия с системой S_2, и при этом, в системе S_1 существует функция, которая позволяет некоторым способом определять границы ожидаемого результата. Значение этой функции может быть представлено как изменение некоторой величины, имеющей информационный смысл, и выражено в битах. Будем считать, что ожидаемый результат равен *1 биту*, если предполагаемый ответ имеет два возможных равновероятных значения. Во многих случаях такой ответ может быть сведен к одному из значений пары *да/нет*. Если системе S_1 удается сформулировать вопрос в такой форме, что система S_2 способна ответить на него однозначно из набора, состоящего из пары предполагаемых ответов, то фактически при получении такого ответа процесс взаимодействия можно считать завершенным. Система S_1 получает ожидаемое количество информации, и цель взаимодействия при этом достигнута заданием единственного вопроса. Однако в реальной жизни такие ситуации встречаются достаточно редко. В большинстве случаев, ожидаемый результат не сводится к простому двоичному ответу, и с другой стороны, системе S_1 не всегда удается сформулировать вопрос в такой форме, что система S_2 может однозначно на него ответить.

В процессе живого общения люди постоянно уточняют и корректируют цели и соответственно форму и содержание разговора. Такое поведение вполне согласовывается с поведением кибернетических систем с обратными связями. Обратные связи в той или иной форме присущи практически всем сложным техническим, биологическим и социальным системам [30]. Применительно к экономике и финансовым рынкам обратные связи носят исключительно информационный характер, и при этом они оказывают влияние на

процессы принятия решений, которые во многом похожи на процессы, происходящие в интерактивных информационных системах при выборе адекватных реакций. Добавление механизма обратных связей в системы интерактивного взаимодействия открывает возможности для применения хорошо развитых кибернетических и математических методов анализа при исследовании и оптимизации их поведения. Однако применительно к нейронным моделям нужно учитывать, что их сложность значительно превосходит сложность систем, которые традиционно рассматриваются в качестве объектов управления.

Обратные связи в диалогах непосредственно влияют на его контекст. Интеграция контекстных полей, которые можно построить используя нейронные модели, вместе с механизмом обратных связей, позволяющим корректировать развитие диалога, качественно изменяет процесс взаимодействия человека и информационных систем.

На рис. 63 показана характеристика процесса общения в системе S_1. Ожидаемый результат может быть представлен как некоторая целевая функция, которая задает предполагаемое значение получаемой информации. Это значение для некоторых случаев может быть известно с достаточно высокой точностью. Например, если систему S_1 интересует факт наличия какого-либо продукта или наличие определенного свойства у объекта, то в этом случае ожидаемое количество информации будет равно $E = 1$ *бит*. Вопрос, который система S_1 может задать системе S_2, в этом случае выглядит следующим образом:

Система S_1*: У камеры Nikon Coolpix 5700 увеличение равно 8x?*
Система S_2*: Да.*

Такого типа вопросы, в которых предполагаемый ответ сводится к выбору из минимального набора возможных вариантов, называются закрытыми. Вопросы, в которых предполагаемый ответ может иметь большую вариативность, называются открытыми.

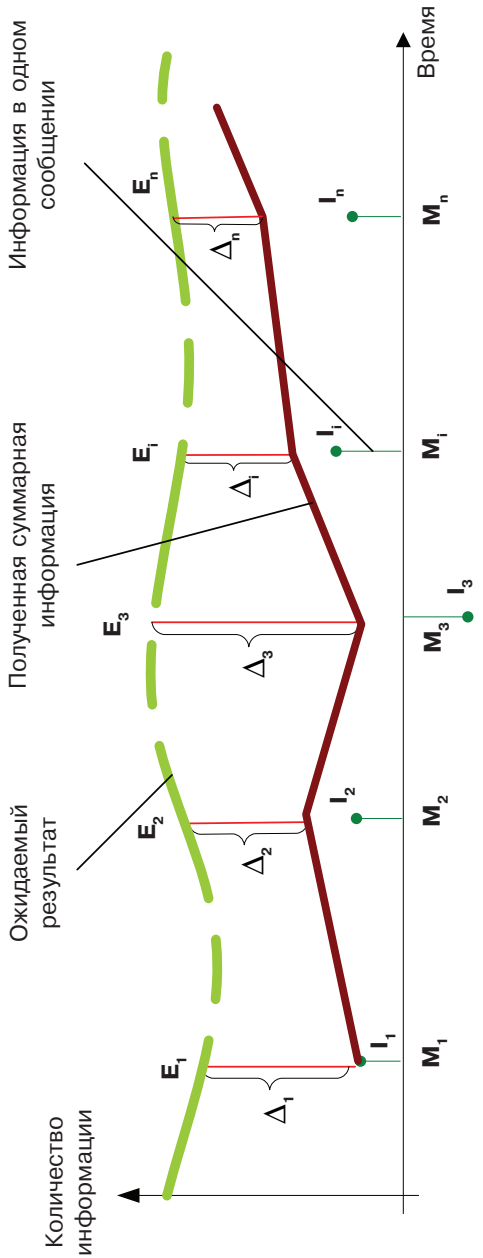


Рис. 63. Изменение информационного состояния системы в процессе обмена сообщениями

Очевидно, что если вопрос будет сформулирован в несколько ином виде, ожидаемое количество информации изменится:

***Система S_1**: Какое увеличение у камеры Nikon Coolpix 5700?*
***Система S_2**: 8x.*

В этом случае, ожидаемое количество информации можно оценить приблизительно в *4 бита* (если предположить, что существует 16 вариантов увеличения, то используя простейшую формулу вычисления количества информации:

$$E = \log_2(P_{\text{до сообщения}} / P_{\text{после сообщения}}),$$
когда $P_{\text{до сообщения}} = 1$ *и* $P_{\text{после сообщения}} = 1/16,$
$E = 4$ *бита*.

Если в ответ система S_2 помимо простого ответа добавит еще сведения о цене, наличие этой камеры на складе, и т. п., то в этом случае полученная информация будет значительно больше ожидаемых *4 бит*. Например, система S_2 может ответить на этот же вопрос следующим образом:

***Система S_1**: Какое увеличение у камеры Nikon Coolpix 5700?*
***Система S_2**: 8x, и ее стоимость у нас $1047,99. Но я Вам порекомендую DiMAGE 7Hi Minolta за $1196,99.*

В данном случае, мы можем считать, что эта информация может оказаться полезной для покупателя, однако количество полученной информации, очевидно, значительно больше, чем количество ожидаемой информации. Ответ может содержать дополнительные данные, которые с точки зрения системы S_1 могут рассматриваться как информационный шум. Например:

***Система S_2**: Это очень хорошая камера, и это разрешение 8x, позволяет делать чудесные снимки. У нас сейчас распродажа и ее стоимость всего $1047,99. Но я Вам порекомендую другую камеру. У меня есть приятель, и он без ума от нее — DiMAGE 7Hi Minolta всего за $1196,в таблице базыв таблице базы99.*

Со своей стороны система S_1 также может вносить в вопрос информационный мусор:

***Система S_1**: Вы знаете, я слышал, что у всех камер Coolpix хорошее разрешение. А самая лучшая из них — это 5700. Вы мне не подскажете, какое у нее разрешение?*

Безусловно взаимодействие человека с интерактивными информационными системами накладывает определенные рамки на форму общения, однако общие принципы информационных оценок процессов обмена информации остаются неизменными. Сообщения, которыми обмениваются системы, будут содержать избыточную информацию, и они также будут содержать информационный шум. Очевидно также, что во многих случаях нам не удастся получить точные информационные характеристики этих сообщений, однако мы должны стремиться к минимизации избыточной информации при генерации ответов. При этом нужно учитывать, что система S_2 может иметь свое представление об ожидаемой информации в системе S_1, и это позволяет ей строить различные траектории развития диалога и в зависимости от этого предоставлять для системы S_1 дополнительную информацию.

В первоначальном состоянии система S_1 обладает некоторой информацией I_0 об интересующем ее событии. После получения сообщения M_1 система S_1 приобретает некоторую информацию I_1, и переходит в новое состояние, в котором она может послать системе S_2 новый вопрос, в ответ на который она может получить новую дополнительную информацию I_2. В общем случае полученная информация может привести к тому, что общее представление системы S_1 об инте-

ресующем событии может уменьшиться. В каждый момент времени существует расстояние Δ_i между текущим информационным представлением системы S_1 об интересующем ее событии и целевой функцией, задающей ограничение на количество получаемой информации.

Таким образом, принцип обратной связи применительно к системам интерактивного взаимодействия может быть сформулирован как стремление минимизировать расстояние Δ_i в каждой точке процесса общения, как системой S_1, инициирующей вопросы, так и системой S_2, отвечающей на эти вопросы и одновременно способной задавать дополнительные вопросы со своей стороны. Симметричность систем S_1 и S_2 позволяет рассматривать эти системы как равноправные. Действительно, систему S_1 может интересовать нечто в системе S_2, а в свою очередь система S_2 может иметь интерес к информации, которой обладает система S_1.

Поскольку система S_1 и система S_2 могут иметь свои собственные различные цели, и при этом должны моделировать и контролировать как свое собственное поведение, так и поведение партнера по диалогу, задача достижения эффективного интерактивного взаимодействия сводится к поиску оптимальной траектории диалога в условиях существования двух целевых функций, задающих оптимальные информационные коридоры для систем S_1 и S_2.

СОЗНАНИЕ И ПОДСОЗНАНИЕ

Последовательности сообщений, которыми могут обмениваться две системы образуют траектории, которые можно представить как последовательности переходов в некотором пространстве всех возможных состояний P, в которых может находиться данная система. В каждый текущий момент времени состояние P_i, в котором находится данная система мы будем называть активным или сознательным состоянием. Для каждого возможного состояния P_i можно создать соответствующую карту (страницу) C_i, которая может определять реакцию системы. Карта C_i является контейнером данных и программ и может быть загружена в интерпретатор текущего состояния (*активное поле*).

Аналогом пространства P является Интернет, состоящий из множества страниц, которые становятся активными только в тот момент, когда они загружаются в программу просмотра (*веб-браузер*). Для пользователей Интернет, в каждый момент времени в текущем окне *веб-браузера* активной является одна страница, из которой возможны гипертекстовые переходы по разным направлениям. Используя такую аналогию, будем считать, что на каждом шаге интерактивного обмена, внутри каждой системы существует объект, аналогичный по своим свойствам текущей странице веб-браузера в Интернете и обладающий способностью генерировать ответ и осуществлять динамический переход к последующему объекту. Нейронные модели при этом служат

механизмом, обеспечивающим, с одной стороны, сохранение контекста и смыслового содержания процесса обмена сообщениями, необходимого для генерации ответа, а с другой, позволяют осуществить переход к следующему объекту, отражающему новое состояние системы.

Карты состояний в общем случае могут представлять собой сложные структурные объединения программ и данных, хранимые в формате *DHTML/XML*. Для редактирования и просмотра содержимого карт может использоваться веб-браузер, при помощи которого автор может выполнять все основные операции редактирования и модификации их содержимого. Различные наборы карт приведены в качестве примера в разделе «Демонстрация работы лингвистической нейронной среды» *НЭС*. Получаемые в результате *XML*-описания могут быть сохранены в любой подходящей системе хранения данных, к которой может иметь доступ интерпретатор текущего состояния. В отличие от веб-браузера, интерпретатор текущих состояний выполняет действия, в основном связанные с изменением внутреннего состояния всей системы. Он взаимодействует с нейронной моделью, определяет условия перехода в новое состояние и выбора следующей карты, загружает программы для динамического исполнения, принимает входные сообщения и передает выходные сообщения в соответствующие каналы ввода/вывода. В целом, его функции являются достаточно ограниченными, что позволяет рассматривать его как «жесткий» или встраиваемый компонент системы. С другой стороны, форматы карт, содержащиеся в них программы и данные, могут быть как угодно сложными, что делает систему открытой для модификации и расширения.

Интерпретатор текущего состояния (рис. 64) можно рассматривать как аналог человеческого сознания. В каждый момент времени в активном поле интерпретатора содержится текущее полученное сообщение и готовится ответ на это сообщение. В это поле сходятся входные и выходные нейронные слои, и оно имеет доступ ко всему множеству нейро-

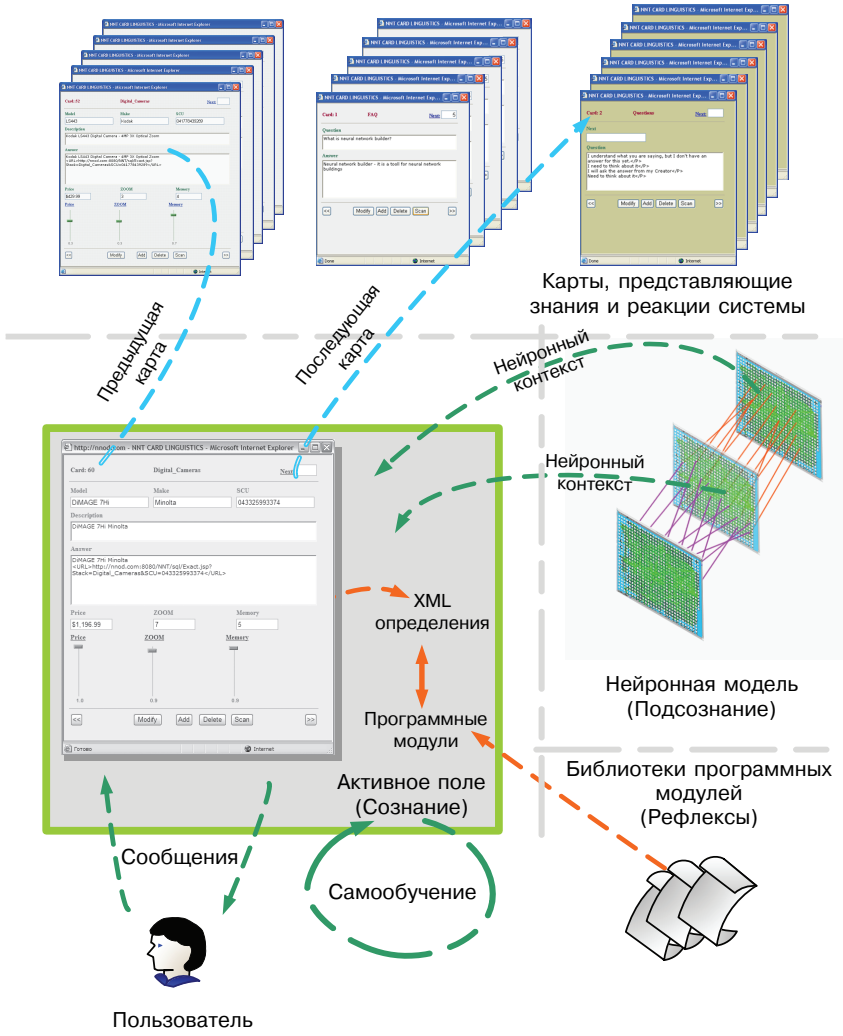


Рис. 64. Последовательность переходов между объектами пространства состояний

нов в рамках активной нейронной модели. В свою очередь нейронные модели можно считать аналогом подсознания. В них происходят массивные параллельные процессы, которые практически невозможно свести к представлениям, которые могут быть выражены в виде аналитических выражений в привычной форме.

Соединение нейронных моделей с активным полем, в котором возможны интерпретации различных программ, открывает возможности для интеграции алгоритмических и нейронных методов в единый комплекс. Выполнение программ, интерпретируемых (скрипты) или предварительно откомпилированных, и динамически загружаемых модулей (*Java*-классы или *DLL*-объекты) можно сравнить с рефлекторными знаниями. Карты и *XML* выполняют роль особого соединительного механизма в интеграции алгоритмических и нейронных методов. *XML* является универсальным и открытым языком, для которого можно сравнительно легко создавать различные интерпретирующие системы, а карты обладают интуитивным пользовательским интерфейсом, позволяющим внешним пользователям поддерживать и развивать интегрированную систему.

Алгоритмические методы позволяют подключить к нейронным моделям широкий набор решателей-агентов для анализа речи, принятия решений, традиционных численных и программных задач и т. п. Интеграция этих двух различных способов моделирования, основанных на принципиально различных аксиоматиках, возможна лишь путем их искусственного объединения, и именно для этих целей и служит активное поле.

Такая интеграция позволяет, например, эффективно решать задачи, связанные с пониманием различных форм речи. В восприятии устной и письменной речи есть много общего, но в тоже время они принципиально отличаются по динамическим характеристикам, по времени реакции и по грамматическим свойствам. В процессе разговора участники обмениваются сообщениями, соответствующими определенной теме. В том случае, когда один из участников задает вопрос, от второго участника предполагается получение ответа, и при этом время, в течение которого этот ответ должен быть получен, ограничено достаточно жестко. Реакция на полученные сообщения происходит в реальном времени, и поток слов, определяющий данный разговор, как правило, не может быть повторен сначала.

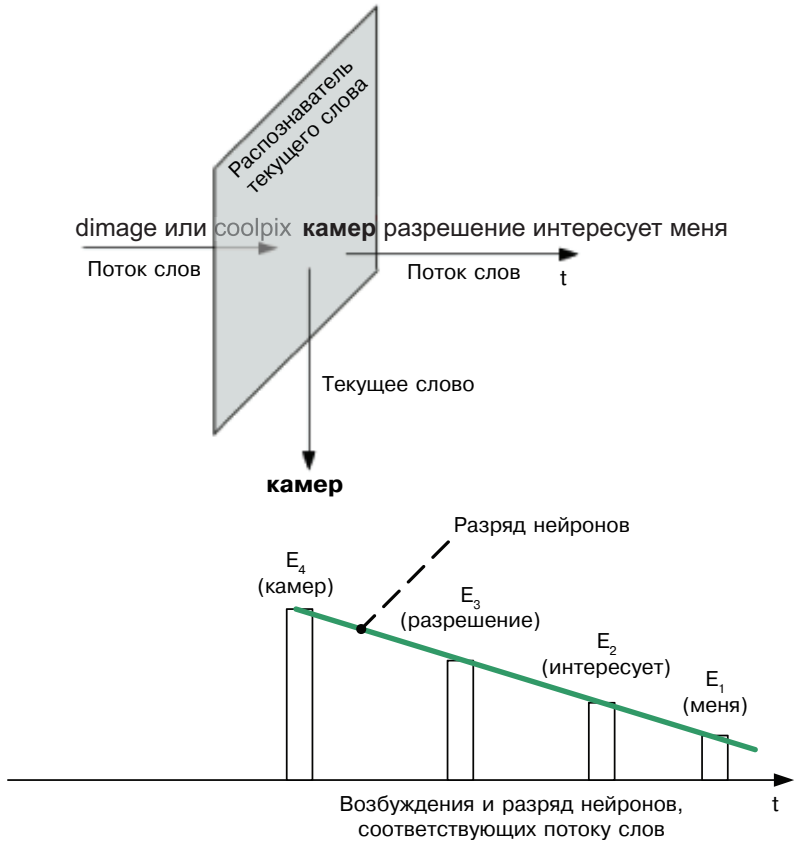


Рис. 65. Последовательность слов в потоке речи и соответствующие возбуждения нейронов

По характеру взаимодействия и времени реакции, обмен сообщениями можно свести к трем основным формам.

- ***Живая речь***. Сообщение не является целостным и не доступно в текстовом виде. Отсутствуют четкие границы, разделяющие фразы. Время ответной реакции критично. Контекст зависит от времени.
- ***Чат***. Время реакции ограничено, сообщение доступно для анализа. Контекст конкретного сообщения может зависеть от предыдущих сообщений. Фразы достаточно разделены, хотя встречаются нарушения общепринятых грамматических правил.
- ***Текстовые сообщения, электронная почта***. Все сообщение представляет законченную последовательность предложений. Время реакции не критично.

Чаты или обмен сообщениями в текстовом виде занимают промежуточное положение между динамичным устным разговором и статичным процессом чтения текстов. В чатах, так же как и в условиях устной речи, существует ограниченное время реакции на сообщения, однако сообщения в чате представляют собой, как правило, вполне законченные грамматические конструкции.

В устной речи поток слов не образует законченные фразы, имеющие четко выраженные грамматические формы. Предложения могут быть выделены в этом потоке весьма условным образом, в отличие от чатов или обычных статических текстов. Разделение грамматических конструкций в устной речи значительно сложнее, чем в письменной. Помимо таких факторов, как интонация, специальные обороты речи и паузы, время являются важным атрибутом, позволяющим определить границы предложений и установить контекст, связывающий отдельные, иногда достаточно отстоящие друг от друга, слова по смыслу. На рис. 66 показана структура связей между активным слоем, в котором хранятся текущие слова, имеющие отношение к конкретному

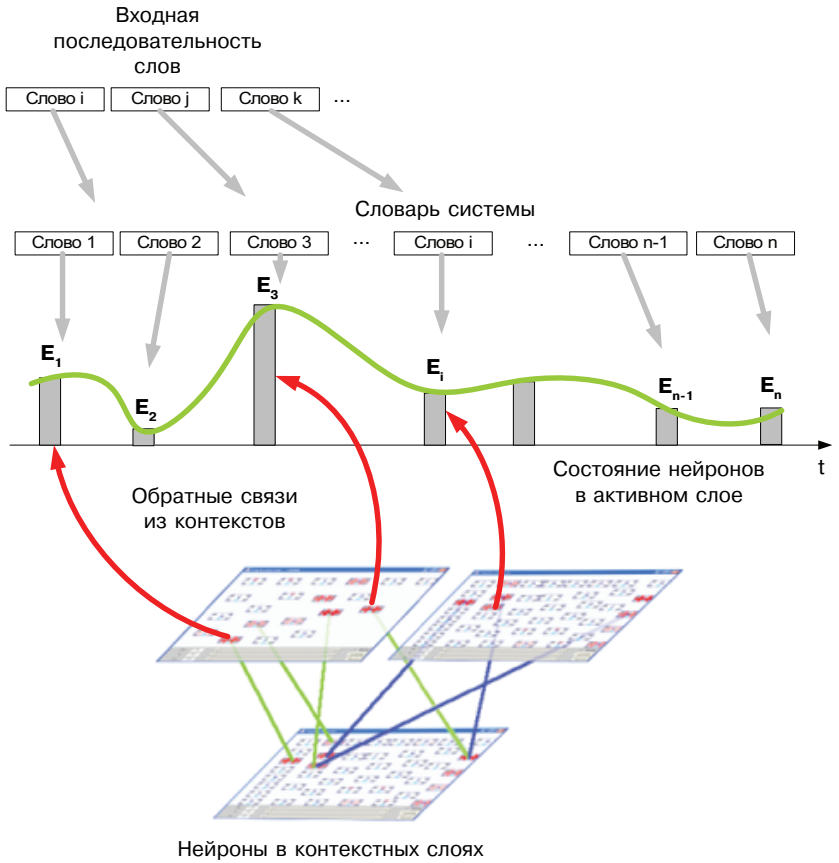


Рис. 66. Последовательность слов и обратные связи из контекстных слоев

разговору и контекстными слоями, позволяющая поддерживать активными те слова или значения, которые имеют отношение к текущему моменту. В процессе разговора слова активного слоя получают изначальное возбуждение, которое со временем угасает, если из контекстных слоев это слово не получит дополнительное возбуждение, подтверждающее важность этого слова на текущий момент.

Нейронные системы, прекрасно приспособленные для отображения большого количества динамических процессов, идеально подходят для моделирования динамики изменения смысла устной речи. При этом контекст разговора, представляя механизм обратной связи, может влиять как на слова или значения, хранимые в активном слое, так и на связанные значения внутри контекстных слоев.

Характер взаимодействия влияет на структуру речи человека вне зависимости от того, является его собеседником другой человек или компьютер. В одной из ранних работ на эту тему [6] рассматриваются изменения в структуре речи человека в зависимости от способа коммуникации. Авторы рассматривают разговоры между двумя людьми в прямом общении, по телефону и через компьютерный терминал (аналог современного чата). Анализируя ситуационные разговоры между людьми, в статье приводится интересное заключение по поводу использования прямых контекстных оборотов — слов *ЭТОТ*, *ЭТО*, *ТОТ* и т. п. В чате такие контекстные указатели используются значительно чаще, чем в телефонных разговорах. Очевидно, что возможность видеть предыдущие сообщения позволяет человеку быстро определить смысл контекстных связей и тем самым сократить время на передачу повторных слов или словосочетаний, за счет использования коротких контекстных указателей.

Во время устного живого разговора, у человека нет возможности вернуться к предыдущим сообщениям, однако компьютер способен хранить всю последовательность поступивших слов, и для того, чтобы поддерживать общение в привычной для человека форме, компьютерные системы должны обладать способностью *«забывать»* предыдущие сообщения аналогично тому, как это делает человек. Такая способность к «забыванию» в нейронных моделях моделируется за счет разряда нейрона. Пример, демонстрирующий возбуждение и разряд нейронов в промежуточном слое в результате воздействия последовательности слов, показан на рис. 38.

В том случае, когда партнером человека в устном общении выступает компьютер, предварительное распознавание осуществляется, как правило, акустической системой распознавания голоса (*Automatic Voice Recognition — AVR*), которая затем передает последовательности слов в лингвистическую систему распознавания речи (*Automatic Speech Recognition — ASR).* Большинство систем *AVR* использует грамматики, позволяющие распознать слова в контексте определенного предложения. Применение грамматик значительно повышает качество распознавания за счет добавления вероятностной характеристики слова в последовательности. Задача распознавания речи заключается в поиске такого сочетания акустических моделей вместе с лингвистическими, при котором обеспечивается наиболее эффективное определение вероятностей словосочетаний в последовательностях акустических форм слов и соответствующих им словарей. В тех случаях, когда грамматики хорошо заданы, современные акустические распознаватели позволяют получить исключительно высокую точность распознавания.

Очевидно, что с расширением предметной области разговора происходит увеличение сложности грамматик, и именно в этом заключается проблема современных систем распознавания речи. Практически невозможно создать универсальную грамматику, способную обеспечить распознавание последовательностей слов для всевозможных вариантов человеческих сообщений, даже для сравнительно узких предметных областей. Среди различных способов, которые могут быть использованы при решении этой проблемы, можно выделить статистические модели (*Statistical Learning Models — SLM*), способные самообучаться в процессе предварительной тренировки (см., например, [5]), и методы адаптивного управления, позволяющие выделять закономерные последовательности и обучаться с помощью хорошо сбалансированных алгоритмов и критериев [33].

Один из путей порождения акустических грамматик лежит в применении нейронных моделей, способных устанавливать взаимосвязи в последовательностях слов (или элементов

их представляющих). На рис. 67 показан результат обучения словарного слоя нейронной модели на последовательности предложений, взятых из протоколов чатов, таких, например, как:

What's Longhorn
Who are you
Tel me about your company
Tell me about your technology
How long have you been in business
what's your phone number
what's his phone number
...

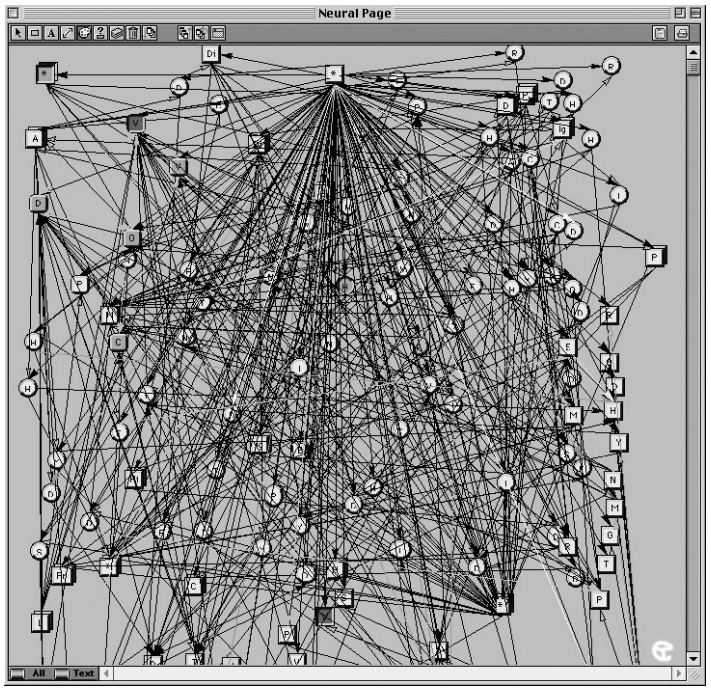


Рис. 67. Статистические зависимости между словами после обучения (для наглядности в словах показаны только начальные буквы)

Обучение нейронной модели заключается в построении первоначального словаря и затем в установлении связей между словами предложений и последующей корректировке этих связей по мере обучения на последующих фразах. Существует множество алгоритмов обучения нейронных сетей, которые могут быть легко применены к нашей модели.

ГЛОССАРИЙ

Адекватное — соответствующее коммуникативным установкам субъекта в процессе общения или восприятия. В философии этот термин служит для определения верности или, иными словами, качества восприятия. В этом смысле истина является абсолютом адекватности мышления бытию.

Бит — единица измерения информации. Соответствует изменению состояния системы в результате получения простейшего сообщения о событии, вероятность которого равна 0,5.

Гармония — стройная и соразмерная связь компонентов в единое целое. В музыке гармония основана на законах и правилах объединении звуков в созвучия. В отличие от мелодии, гармония определяет «вертикальное» согласование звуков.

Данное — сообщение, записанное в любой форме и на любом носителе.

Информация — мера изменения состояния системы в результате получения сообщения из внешней среды. Измеряется в битах. Информация часто используется как синоним таких понятий, как данные, знания, факты и др.

Интерпретатор — программа, которая может выполнять другие программы. В отличие от компилятора, который только подготавливает программы для их дальнейшего исполнения в иной среде, интерпретатор немедленно приступает к выполнению последовательности команд.

Клонирование — производство идентичной копии на основании исходного образца.

Лексема — слово, рассматриваемое как единица словарного состава языка в совокупности всех его конкретных грамматических форм, а также всех возможных значений (смысловых вариантов). В одну лексему объединяются разные парадигматические формы одного слова (например, «словарь, словарём, словарю» и т. п.) и разные смысловые варианты слова, зависящие от контекста, в котором оно употребляется (например, «соль» в смысле названия вещества и в значении того, что придает остроту или интерес какому-либо высказыванию, мысли и т. п.).

Лексикология — раздел языкознания, в котором исследуется словарный состав языка, его лексика, взаимодействие между значениями слов и понятиями. Понятия чаще всего интернациональны, тогда как значения слов национальны. Изучает закономерности функционирования и развития словарного состава языка.

Лингвистика — наука о языке. Объектом лингвистики является строение, функционирование и историческое развитие языка, язык во всем объеме его свойств и функций.

Математическая лингвистика — математическая дисциплина, разрабатывающая формальный аппарат для описания строения естественных и некоторых искусственных языков. Включает теории и способы описания синтаксических структур, формальных грамматик и аналитических моделей языка.

Мелодия — линейная последовательность событий. Мелодия представляет горизонтальное согласование звуков. Эти последовательности содержат повторяющиеся образы звуковых объектов (длительность и тон).

Мера — функция, которая устанавливает соответствие между количественной и качественной характеристиками явлений, объектов и процессов. Например, она позволяет связать такие характеристики, как «объем», «вероятность» «скорость» и числовые величины из некоторого множества значений.

Нейрон — специфичная клетка, которая осуществляет восприятие, обработку и передачу возбуждений от рецепторов к другим нейронам.

Нерв — соединение между нейронами, по которому передаются возбуждения. Аксон — единственный отросток, по которому импульсы передаются к другим клеткам. Синапсы — контакты, по которым импульсы поступают внутрь нейрона.

Обратная связь — обратное воздействие результатов процесса на его протекание или управляемого процесса на управляющий орган. Различают положительную и отрицательную обратные связи. Положительная — если результаты процесса усиливают его, отрицательная — когда результаты процесса ослабляют его действие.

Парадигма — система форм одного слова, отражающая видоизменения слова по присущим ему грамматическим категориям, например, по роду, числу и падежу для существительных, лицу, времени, виду и прочее для глаголов; схема изменения слова по грамматическим категориям; образец типа склонения или спряжения.

Переменная — одно из основных понятий математики, логики и программирования. В высшей математике под переменной понимается некоторая «величина», которая может «изменяться», принимая в процессе этого изменения различные «значения». В формализованных языках математической логики переменной называются символы фиксированного вида, могущие при определенных условиях заменяться выражениями. В программировании под переменной понимается тройка — идентификатор, область хранения и связанное с ними значение.

Полиморфизм — в физике, минералогии, химии способность некоторых веществ существовать в состояниях с различной атомной кристаллической структурой. Различие в структуре обусловливает и различие в свойствах полиморфных модификаций данного вещества.

Понятие — в формальной логике это элементарная единица мыслительной деятельности, обладающая известной

целостностью и устойчивостью и взятая в отвлечении от словесного выражения этой деятельности. Понятие — это то, что выражается (или обозначается) любой самостоятельной частью речи, а если перейти от масштабов языка в целом к «микроуровню», — то членом предложения.

Предложение — основная единица связной речи, характеризуемая определенными смысловыми и структурными признаками. По отношению говорящего к высказываемой им мысли предложения делятся на три группы: повествовательные, восклицательные и вопросительные.

Процесс — последовательная смена состояний системы в результате как внутренних, так и внешних действий.

Рибосомы — внутриклеточные частицы, осуществляющие биосинтез белка; рибосомы обнаружены в клетках всех без исключения живых организмов: бактерий, растений и животных; каждая клетка содержит тысячи или десятки тысяч рибосом. Белок всех организмов состоит из 20 видов аминокислот. Каждый белок характеризуется определенным ассортиментом и количественным соотношением аминокислот.

Синхронизация — процедура согласования процессов обработки или передачи данных. Синхронизация осуществляется на физическом уровне: посредством тактирования, задающего единый стандарт дискретного времени для управления процессом передачи сигналов.

Система — совокупность объектов, которые могут быть выделены из среды при помощи границы. Система может энергетически или информационно взаимодействовать с другими системами.

Слово — центральная единица языка. В любом языке есть слово, означающее «слово». Основные свойства слова: фонетические и семантические, воспроизводимость, синтаксическая самостоятельность, валентность. Слова делятся на знаменательные (обозначающие некоторое понятие) и служебные, которые служат для связи слов между собой.

Словоформа — слово, рассматриваемое как представитель определенной лексемы и определенной грамматической

формы. Совокупность всех словоформ данной лексемы образует ее парадигму.

Сообщение — объект, при помощи которого происходит взаимодействие. Сообщение всегда имеет физическую природу и, в отличие от всех остальных явлений, приводит к изменению состояния системы, которая это сообщение получила.

Состояние — набор параметров и их значений, характеризующих поведение системы во времени.

Среда — включает в себя все системы и все объекты, которые обладают физическими и информационными свойствами. Можно рассматривать среду, как универсальную систему, которая окружает любую, входящую в нее систему.

Энергия — общая количественная мера движения и взаимодействия всех видов материи. В классической физике энергия системы меняется непрерывно и может принимать любые значения. В квантовой теории энергия микросистемы может принимать только дискретные значения.

БИБЛИОГРАФИЯ

1. **Aristotle.** The Basic Works of Aristotle. — New York : The Modern Library, 2001.
2. **Berners-Lee, T.** Information Management: A Proposal. — CERN, 1989. May. (www.w3.org/history/1989/proposal.htm)
3. **Brooks, F.** The Mythical Man Month. — Boston : Addison Wesley, 1995.
4. **Bush, V.** As We May Think // Atlantic Monthly. — 1945. Vol. 176. № 1. P. 101–108.
5. **Cherkassky, V., Mulier, F.** Learning from Data. — New York : John Wiley & Sons, 1998.
6. **Cohen, P. R., Fertig, S., Starr, K.** Dependencies of discourse structure on the modality of communication: telephone vs. teletype // Proceedings of the 20th conference on Association for Computational Linguistics. — Toronto, Ontario, Canada. — 1982. June 16-18.
7. **Dijkstra, E. W.** The Structure of the THE-Multiprogramming System // Communications of the ACM. — 1968. Vol. 11. № 5. P. 341–346.
8. **Einstein, A.** Relativity. The Special and the General Theory. — New York : Three Rivers Press, 1961.
9. **Fellbaum, C.** WordNet: An Electronic Lexical Database. — Cambridge, MA : MIT Press, 1998.
10. **Feynman, R. P.** Feynman Lectures on Computation. — Cambridge, Mass. : Perseus Publishing, 1999.

11. **Godel, K**. On Formally Undecidable Propositions of Principia Mathematica and Related Systems. — New York : Dover Publications, Inc., 1992.
12. **Heisenberg, W.** Physics and Philosophy. The Revolution in Modern Science. — New York : Prometheus Books, 1999.
13. **Heisenberg, W.** The Physical Principles of the Quantum Theory. — Leipzig, Dover Publications, Inc., 1949.
14. **Ingram, J.** The Notation of Time // Contact Magazine. — London, 1985.
15. **Maass, W., Natschlager, T., Markram, H.** Computational models for generic cortical microcircuits // Computational Neuroscience: A Comprehensive Approach / In J. Feng, editor. — Boca Raton : Chapman & Hall/CRC, 2004. P. 575–605.
16. **Maass, W., Natschlager, T., Markram, H.** The "liquid computer": A novel strategy for real-time computing on time series // Special Issue on Foundations of Information Processing of TELEMATIK. — 2002. Vol. 8. № 1. P. 39–43.
17. **Martin, J.** Cybercorp: The New Business Revolution. — 1996. — ISBN 0-814-40351-4.
18. **McCulloch, W., Pitts, W.** A logical calculus of the ideas immanent in nervous activity // Bulletin of Mathematical Biophysics. — 1943. Vol. 7. P. 115–133.
19. **Neumann, J.** The Computer and the Brain. — 2-nd edition. — New Haven, Conn. : Yale University Press, 2000.
20. **Neumann, J.** First Draft of a Report on the EDVAC : Contract 670-ORD4926, Philadelphia, PA. — Morse School of Electrical Engineering. University of Pennsylvania. — 30 June 1945.
21. **Pink, D. H.** A Whole New Mind. — New York : The Berkley Publishing Group, 2006.

22. **Shannon, C.** A Mathematical Theory of Communication // The Bell System Technical Journal. — 1948. Vol. 27. July, October. P. 379–423, 623–656.
23. **Shannon, C.** The Bandwagon // IRE Trans. Information Theory. — 1956. Vol. 2.
24. **Stay, J. F.** HIPO and integrated program design // IBM Systems Journal. — 1976. Vol. 15. № 2. P. 143.
25. **Stravinsky, I., Craft, R.** Conversations with Igor Stravinsky. — Berkeley : University of California Press, 1980.
26. **Tecuci, Gh., Boicu, M., Marcu, D.** Training and Using Disciple Agents: A Case Study in the Military Center of Gravity Analysis Domain // AI Magazine. — 2002. Vol. 23. № 4. P. 51-68.
27. **Teller, A., Veloso, M.** Neural programming and an internal reinforcement policy. In First International Conference Simulated Evolution and Learning. — Springer-Verlag, 1996.
28. **Turing, A. M.** Computable Numbers, With an Application to the Entscheidungsproblem // Proceedings of the London Mathematical Society. — 1936. Vol. 42. № 2. P. 230–265.
29. **Watson, J. D., Crick, F. H. C.** Genetical implications of the structure of deoxyribonucleic acid // Nature 171. — 1953. May.
30. **Wiener, N.** Cybernetics. — 2-nd edition. — Cambridge, Mass. : MIT Press, 1961.
31. **Winograd, T., Flores, F.** Understanding Computers and Cognition: A New Foundation for Design. — Addison-Wesley, 1987.
32. **Заде, Л.** Понятие лингвистической переменной и его применение к принятию приближенных решений / Л. Заде. — М. : Мир, 1976.
33. **Жданов, А. А.** Бионический метод автономного адаптивного управления // От моделей поведения к искусственному интеллекту / Под ред. В. Г. Редько. — М. : УРСС, 2006.

34. **Колмогоров, А. Н.** Теория информации и теория алгоритмов / А. Н. Колмогоров. — М. : Наука, 1987.
35. **Раушенбах, Б. В.** Геометрия картины и зрительное восприятие / Б. В. Раушенбах. — СПб. : Азбука-классика, 2002.
36. **Раушенбах, Б. В**. Пространственные построения в древнерусской живописи / Б. В. Раушенбах. — М. : Наука, 1975.
37. **Сеченов, И. М.** Рефлексы головного мозга // Сеченов И. М., Павлов И. П., Введенский Н. Е. Физиология нервной системы : Избранные труды : Вып. 1 / Под общ. ред. К. М. Быкова — М. : Гос. изд-во мед. лит., 1952.
38. **Шеннон, К.** Работы по теории информации и кибернетике / К. Шеннон. — М. : Изд-во ин. лит., 1963.

ОГЛАВЛЕНИЕ

Введение ... 5
Энергия, информация и знания ... 13
Усилители, нейроны и Интернет ... 20
Программирование реакций ... 27
Среда нейронного программирования ... 32
Взаимодействующие системы ... 37
Интерпретации и измерения ... 48
Вычислительные ассоциативные нейроны ... 58
Конструктивный нейрон ... 65
Сетевая оболочка и базовые протоколы ... 81
Нейроны, слова и смысл ... 90
Перцептивная лингвистика ... 100
Контексты и временная логика ... 109
Клонирование ... 121
Селективный контекст ... 133
Обратные связи в интерактивных процессах ... 142
Сознание и подсознание ... 149
Типовые конструктивные элементы ... 160
Заключение ... 172
Приложение ... 175
Глоссарий ... 182
Библиография ... 187