ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

9(205) 2013

ТЕОРЕТИЧЕСКИЙ И ПРИКЛАДНОЙ НАУЧНО-ТЕХНИЧЕСКИЙ ЖУРНАЛ

Издается с ноября 1995 г.

УЧРЕДИТЕЛЬ

Издательство “Новые технологии”

СОДЕРЖАНИЕ

Информация о журнале доступна по сети Internet по адресу **http://novtex.ru/IT.**
Журнал включен в систему Российского индекса научного цитирования.
Журнал входит в Перечень научных журналов, в которых по рекомендации ВАК РФ должны быть опубликованы научные результаты диссертаций на соискание ученой степени доктора и кандидата наук.

ОБЩИЕ ВОПРОСЫ

УДК 004.03 +530.1

И. М. Гуревич, канд. техн. наук, ст. науч. сотр.,
Институт проблем информатики РАН,
ООО "ГЕТНЕТ Консалтинг", гл. конструктор,
г. Москва,
e-mail: iggurevich@gmail.com

Информационные мировые константы

Возможности ученых и инженеров ограничены самой природой — конструкцией и свойствами атомов и молекул. В работе показано, что объем информации во Вселенной ограничен, объем информации в атомах, аминокислотах и азотистых основаниях, их дифференциальная информационная емкость, масса атома водорода, структура и разность энергий его базисных состояний накладывают фундаментальные ограничения на память и быстродействие вычислительных устройств, информационных систем. Природой также определены минимальные затраты, необходимые для записи и передачи информации. В работе приведены информационные мировые константы: информационная граница IB, постоянная памяти IM, постоянная быстродействия IS, постоянная информационных затрат IC.

Данные константы следует добавить в ряд мировых констант. Они определяют и ограничивают процессы формирования и развития естественных и искусственных объектов, в том числе определяют срок действия закона Мура и его аналогов.

Ключевые слова: *физические мировые константы, информационные мировые константы, фундаментальные ограничения, объем информации, память, быстродействие*

Введение

Физические мировые константы [1]: постоянная Планка h, гравитационная постоянная G, скорость света c, постоянная Больцмана k, ... — это числа, которые определяют характеристики и свойства Вселенной и всех физических систем, их формирование, развитие, взаимодействие, взаимопревращения ... Они также определили возможность возникновения жизни.

Уже не одно десятилетие развитие производства запоминающих устройств и микрочипов связано с постоянным увеличением плотности записи информации на эти носители. Может создаться впечатление, что завтра изобретут новую технологию и вновь увеличат плотность записи. И так будет всегда.

"Человек всегда стремился к большему, желая не просто повторить изобретения природы, но и превзойти их. До сих пор это ему не удавалось, и лишь с освоением нанотехнологий он может получить реальные шансы на воплощение своей давней "бредовой" мечты — присвоение функции Творца Вселенной, связанной с возможностью по своей воле создавать новый мир на основе биоорганомики, соединившей физику и молекулярную биологию" [2].

Увы, существуют фундаментальные ограничения, которые неминуемо прекратят этот рост и поставят земную цивилизацию перед выбором: либо прекратить запоминать новую информацию, либо навсегда стирать накопленную ранее. Как скоро наступит этот момент? Экспоненциальный рост объемов хранимой на нашей планете информации может привести к такому поворотному для человечества моменту неожиданно скоро. Полезно задуматься уже сегодня. Возможности ученых и инженеров ограничены самой природой — конструкцией и свойствами атомов и молекул — рядом информационных мировых констант: информационная граница IB, постоянная памяти IM, постоянная быстродействия IS, постоянная информационных затрат IC. В работах автора [3—8] показано, что объем информации во Вселенной ограничен, объем информации в атомах, аминокислотах и азотистых основаниях, их дифференциальная информационная емкость, масса атома водорода, структура и разность энергий его базисных состояний накладывают фундаментальные ограничения на память и быстродействие вычислительных устройств, информационных систем. Природой также определены минимальные затраты, необходимые для записи и передачи информации. Данные информационные ограничения определяют и ограничивают процессы формирования и развития естественных и искусственных объектов, в том числе определяют срок действия закона Мура и его аналогов.

1. Фундаментальные ограничения на объем информации в естественных системах

Впервые оценка объема информации в нашей Вселенной 10^{90} бит была дана в работе автора [3]. Объем информации во Вселенной ограничивает сложность естественных и искусственных систем. Данное ограничение (10^{90} бит) является наиболее сильным фундаментальным ограничением, накладываемым природой на сложность естественных и искусственных систем и является информационной мировой константой — *информационной границей* $IB = 10^{90}$ бит.

Эта же оценка была получена Lloyd Seth в 2001 г. [11]. В работах [4—10] получены оценки объема информации в элементарных частицах, атомах, молекулах, звездах, галактиках, ... Ниже приведены эти оценки.

- Причина и источник формирования информации — расширение Вселенной и ее исходная неоднородность. При расширении Вселенной изменяется ее фазовое состояние (симметрия) и кривизна пространства; формируются различные типы неоднородностей массы и энергии, в частности, возникают фундаментальные и элементарные частицы, галактические, звездные, планетные системы; формируется классическая информация, в том числе аминокислоты, азотистые основания, белки, ДНК, организмы, цивилизации.
- Наиболее подходящими для формирования и хранения информации структурными единицами материи являются фермионы, а для передачи информации — бозоны.
- При нарушении симметрии между слабым и электромагнитным взаимодействиями во Вселенной формируется 10^{90} бит информации.
- "Информационный" механизм формирования частиц в инфляционной Вселенной порождает число частиц, сравнимое с общепринятой оценкой числа частиц во Вселенной, порядка $10^{80}...10^{90}$.
- Степенное расширение Вселенной порождает неоднородности в объеме $\approx 10^{99}...10^{107}$ бит, из них в обычном веществе $\sim 10^{90}$ бит.
- Во Вселенной, в звездах содержится всего около 10^{80} бит информации.
- Если в ядрах галактик находятся черные дыры массой $\approx 10^{6}...10^{10}$ солнечных масс, то объем информации во Вселенной составляет $\approx 10^{99}...10^{107}$ бит.
- Минимально возможный объем информации во Вселенной с преобладанием вещества $\sim 1{,}7 \times \times 10^{79}$ бит, а с преобладанием излучения — $\sim 10^{91}$ бит.
- Максимально возможный объем информации во Вселенной $\approx 10^{120}$ бит.
- Рост объема информации при степенном расширении Вселенной $\propto \log_2 t$ (t — время с начала расширения).
- Уменьшение плотности информации при степенном расширении Вселенной $\propto \frac{\log t^2 t}{t^2}$.
- Рост объема информации при экспоненциальном расширении Вселенной $\propto \alpha t$.
- Уменьшение плотности информации при экспоненциальном расширении Вселенной $\propto \frac{t}{\mathbf{e}^{3\alpha t}}$, где α — показатель степени расширения.

Примечание. Приведенные оценки характеризуют нашу Вселенную. Однако, если считать, что наша Вселенная — лишь одна из бесконечного числа минивселенных Мультиверса и учитывать, что другие локальные вселенные в силу свойства всеобщности информации тоже содержат какое-то количество информации, то в Мультиверсе количество информации вряд ли конечно.

А. Д. Панов — сотрудник МГУ (Институт ядерной физики им. Скобельцына) в своих работах использует информационную мировую константу — *информационную границу* IB = 10^{90} бит в качестве ограничения на возможности искусственного интеллекта. Он отмечает: "Фактически, для вычисления поведения некоторых практически важных квантовых систем требуются такие мощности классического компьютера или такие объемы вычислений, которые нельзя реализовать не только практически, но и принципиально, так как для вычислений будет необходимо время, превышающее возраст Вселенной, или размер компьютера будет таков, что его нельзя будет уместить внутри космологического горизонта событий. Например, для того, чтобы с использованием квантового алгоритма Шора разложить на простые множители 1000-значное двоичное число (обычная задача для перспективных квантовых компьютеров), квантовому компьютеру требуется память всего в несколько тысяч квантовых ячеек памяти — кубитов, в то время как классическому компьютеру для представления состояния такого квантового компьютера потребуется память порядка 21 000 комплексных чисел — а это уже на много порядков больше объема информации, которая может быть записана во всем обычном веществе видимой части Вселенной 10^{90} бит). Поэтому реальные классические компьютерные симуляторы квантовых вычислений могут работать только с очень малоразмерными системами".

2. Фундаментальные ограничения на информационную емкость естественных и искусственных систем

Существует несколько типов материи с разной зависимостью объема информации (информационной емкости) от массы. Для обычного вещества $I = \beta M, f'(M) = \beta$. Дифференциальная информационная емкость обычного вещества не зависит от его массы. В настоящее время системы строят из обычного вещества. Для обычного вещества, содержащего в атоме, молекуле $I_{\text{at, mol}}$ бит информации, на 1 бит информации необходима масса $\beta = \frac{I_{\text{at, mol}}}{m_{\text{at, mol}}}$. Поэтому $I = \frac{I_{\text{at, mol}}}{m_{\text{at, mol}}} M, f'(M) = \frac{I_{\text{at, mol}}}{m_{\text{at, mol}}}$. Дифференциальная информационная емкость атомов разных элементов и, соответственно, молекул примерно одинакова.

Дифференциальная информационная емкость атомов, аминокислот, азотистых оснований определяет фундаментальные ограничения на информационную емкость устройств хранения данных [4—10].

Ограничение, накладываемое дифференциальной информационной емкостью неживой материи. Атомы это простейшие естественные средства хранения информации. На основе оценки дифференциальной

информационной емкости атомов — $\approx 10^{-28}$ бит/кг определяется нижняя граница G дифференциальной информационной емкости V искусственных устройств хранения данных $G \approx 10^{-28}$ бит/кг. Следовательно, информационная емкость I искусственных устройств хранения данных, построенных на базе атомов, не превосходит $10^{28}M$ бит, где M — масса устройства хранения данных, непосредственно используемая для хранения. Так как в настоящее время дифференциальная информационная емкость устройств хранения данных $\approx 10^{14}$ бит/кг, то для устройств, построенных на базе атомов, она может быть повышена не более, чем в $\sim 10^{14}$ раз.

Данное ограничение (10^{28} бит/кг) является наиболее сильным фундаментальным ограничением, накладываемым природой на информационную емкость естественных и искусственных систем и является информационной мировой константой — *постоянной памяти* $IM = 10^{28}$ бит/кг.

Ограничение, накладываемое характеристиками атома водорода. На один бит в атоме водорода, рассматриваемого как кубит (q-бит), природа тратит $m_{\text{H}} = 1{,}67 \cdot 10^{-27}$ кг. Следовательно, информационная емкость I искусственных устройств хранения данных, построенных на базе атомов водорода, используемых как q-биты, не превосходит $6 \cdot 10^{26}M$ бит, где M — масса устройства хранения данных, непосредственно используемая для хранения, — масса водорода.

Ограничение, накладываемое дифференциальной информационной емкостью живой материи. Белки, ДНК — простейшие естественные средства хранения информации, построенные природой из атомов. На основе оценки дифференциальной информационной емкости белков, ДНК — $\approx 10^{-25}$ кг/бит определяется нижняя граница IM дифференциальной информационной емкости искусственных устройств хранения данных и устройств, построенных на базе комбинаций атомов, $IM \approx 10^{25}$ бит/кг. Следовательно, информационная емкость искусственных устройств хранения данных, построенных на базе комбинаций атомов, не превосходит $10^{25}M$ бит, где M — масса устройства хранения данных, непосредственно используемая для хранения. Так как в настоящее время дифференциальная информационная емкость устройств хранения данных $\approx 10^{14}$ бит/кг, то для устройств, построенных на базе комбинаций атомов, она может быть повышена не более, чем в $\approx 10^{11}$ раз.

3. Фундаментальные ограничения на производительность естественных и искусственных систем

Разность энергий базисных состояний атома водорода, рассматриваемого как q-бит, накладывает фундаментальные ограничения на быстродействие вычислительных устройств. Согласно теореме Н. Марголиса и Л. Левитина [12] общее число элементарных действий, которые система может выполнить в секунду, ограничено энергией: $k_{op/s} = 2E/\hbar$, где E — превышение средней энергии системы над энергией нижнего состояния, или энергия активации $\hbar = \dfrac{h}{2\pi} = 1{,}0545 \cdot 10^{-34}$ с · Дж — уменьшенная постоянная Планка.

Число операций, выполняемых атомом водорода как q-битом, ограничено числом $k_{op/s} = 2E/\hbar \approx \approx 1{,}5 \cdot 10^{12}$ оп/с.

Производительность компьютера массой один килограмм, построенного из атомов водорода, не превышает 10^{39} (оп/с)/кг.

Данное ограничение является наиболее сильным фундаментальным ограничением, накладываемым природой на быстродействие вычислений естественных и искусственных систем, и является информационной мировой константой — *постоянной быстродействия* $IS = 10^{39}$ (оп/с)/кг.

4. Фундаментальные ограничения на характеристики компьютера массой один килограмм, построенного из атомов водорода

Память компьютера массой один килограмм, построенного из атомов водорода, не превышает $0{,}6 \cdot 10^{27}$ бит.

Производительность компьютера массой один килограмм, построенного из атомов водорода, не превышает 10^{39} оп/с.

5. Ограничения на характеристики компьютера нанобота

В 1986 г. американский инженер Эрик Дрекслер предложил использовать для производства наноустройств механические машины соответствующих (100—200 нм) размеров — нанороботы [2, 13, 14]. Эти роботы должны были собирать устройство непосредственно из атомов, поэтому они были названы ассемблерами — сборщиками. Сборщик оснащен манипуляторами длиной в несколько десятков нанометров, двигателем для перемещения манипуляторов и самого робота, включая упомянутые ранее редукторы и передачи, а также автономным источником энергии. Наноробот должен состоять из нескольких десятков тысяч деталей, а каждая деталь — из одной-двух сотен атомов. Важнейшим узлом наноробота был бортовой компьютер, который управлял работой всех механизмов, определял, какой атом или какую молекулу следовало захватить манипулятором и в какое место будущего устройства их поставить. Линейные размеры этого компьютера не должны были превышать 40—50 нм. Объем компьютера равен $V_C \approx 10^{-22}$ м^3. В таком компьютере можно разместить 10^8 атомов водорода. Следовательно, память компьютера наноробота, построенного из атомов водорода, не превышает 10^8 бит, а его производительность не превышает 10^{20} оп/с.

6. Фундаментальные ограничения на энергию хранения и передачи информации

К фундаментальным понятиям теории информации [15, стр. 20—25, 16] относится термодинамический предел для энергии переключения классического логического элемента $(P\tau)_{\min}$, определяемый как предельное значение минимальной работы $W_{\min} = \Delta E$, которую необходимо совершить над логическим элементом для того, чтобы термодинамически обратимым образом перевести его в состояние, отличающееся от исходного только на один бит информационной энтропии $(P\tau)_{\min} = \Delta E_{\min} = kT \cdot \ln 2 \approx \approx 3 \cdot 10^{-21}$ Дж/бит; где P — мощность; τ — длительность внешнего воздействия; E — энергия; k — постоянная Больцмана $k = 1{,}3806489 \cdot 10^{-23}$ Дж/К; T — абсолютная температура в градусах Кельвина (К).

Информационный процесс в логическом элементе можно охарактеризовать также энергией, называемой энергетической ценой одного бита.

При наличии шумов предельное значение энергетической цены $E = kT \cdot \ln 2 \approx 3 \cdot 10^{-21}$ Дж/бит.

Поскольку $E = mc^2$, где E — энергия, m — масса, c — скорость света, то к фундаментальным понятиям теории информации следует также отнести термодинамический предел переключения классического логического элемента в единицах массы $M_{\min}$, определяемый как предельное значение минимальной массы $M_{\min} = \Delta m$, которую необходимо передать логическому элементу для того, чтобы термодинамически обратимым образом перевести его в состояние, отличающееся от исходного только на один бит информационной энтропии.

В единицах массы термодинамический предел будет $M_{\min} = \dfrac{E_{\min}}{c^2} = \dfrac{(P\tau)_{\min}}{c^2} = \dfrac{k}{c^2} T$; $M_{\min} = 1{,}53619365 \cdot 10^{-40} T \approx 1{,}5 \cdot 10^{-40} T$ кг/бит.

Это оценка массы на запись, стирание, передачу одного бита классической информации. Данная оценка в $n \approx \dfrac{10^{-28}}{10^{-40}} \dfrac{1}{T} = 10^{12} T^{-1}$ раз больше приведенной в работах [15, 16] термодинамической оценки.

При $T = 300$ К термодинамический предел $M_{\min} = \Delta m \approx \dfrac{10^{-28}}{10^{-40}} \dfrac{1}{T} = 10^{12}(300)^{-1} \approx 10^{-13}$ кг/бит.

Это ограничение является наиболее сильным фундаментальным ограничением, накладываемым природой на затраты на запись и передачу информации в естественных и искусственных системах, и является информационной мировой константой — постоянной *информационных затрат* $IC = 10^{-13}$ кг/бит.

Объемы информации τ^i, проходящей через узлы сети, равны $\tau^i = l^i \lambda^i$, где l^i — длина сообщений, проходящих через узел (в битах); λ^i — интенсивность потока сообщений, проходящего через узел i (сооб/с).

Объем информации, проходящей через сеть в целом, равен

$$\tau = \sum_{i=1}^{n} \tau^i = \sum_{i=1}^{n} l^i \lambda^i.$$

Память компьютера массой 1 кг, построенного из атомов обычного вещества, не превышает 10^{28} бит, поэтому масса сети, в которой хранится τ бит информации, должна быть не менее

$$m_{\text{хр}} = \frac{\tau}{10^{28}} = \frac{\sum_{i=1}^{n} \tau^i}{10^{28}} = \frac{\sum_{i=1}^{n} l^i \lambda^i}{10^{28}} \text{ кг.}$$

Производительность компьютера массой 1 кг, не превышает 10^{39} оп/с, поэтому масса сети, в которой обрабатывается в единицу времени τ бит информации, должна быть не менее

$$m_{\text{обр}} = \frac{\tau}{10^{39}} = \frac{\sum_{i=1}^{n} \tau^i}{10^{39}} = \frac{\sum_{i=1}^{n} l^i \lambda^i}{10^{39}} \text{ кг.}$$

Поскольку $m_{\text{хр}} \gg m_{\text{обр}}$, масса сети классических и квантовых компьютеров определяется объемом хранимой в сети информации τ и не может быть менее $m_{\text{хр}} = \dfrac{\tau}{10^{28}} = \dfrac{\sum_{i=1}^{n} \tau^i}{10^{28}} = \dfrac{\sum_{i=1}^{n} l^i \lambda^i}{10^{28}}$ кг.

Поскольку при современных технологиях хранения данных устройство массой 1 кг может хранить не более 10^{14} бит, то масса сети классических и квантовых компьютеров определяется объемом хранимой в сети информации $\tau = \sum_{i=1}^{n} \tau^i = \sum_{i=1}^{n} l^i \lambda^i$ и не может быть менее $m_{\text{хр}} = \dfrac{\tau}{10^{14}} = \dfrac{\sum_{i=1}^{n} \tau^i}{10^{14}} = \dfrac{\sum_{i=1}^{n} l^i \lambda^i}{10^{14}}$ кг.

Электронная почта передает 204 166 667 писем за одну минуту или 3 402 778 писем в секунду. При средней длине письма 1000 бит объем передаваемой информации составляет 3,4 Тбит/с. При средней длине письма 10 000 бит объем передаваемой информации составляет 34 Тбит/с.

Аналитики подчеркивают, что объем хранящейся в Интернете информации удваивается приблизительно каждые полтора года. По оценкам IDC к 2012 г. суммарный объем контента Всемирной сети, формируемого за год, увеличится до 2500 экзабайт ($2{,}5 \cdot 10^{22}$ бит). При этом в 2006 г. в сети хранилось всего 161 млрд Гбайт данных.

При современных технологиях хранения масса хранения информации, формируемой за год в сети классических компьютеров (Интернет), использующей для вычислений квантовые компьютеры, не менее $2{,}5 \cdot 10^{22}$ бит/10^{14} (бит/кг) = $2{,}5 \cdot 10^{8}$ кг = $= 2{,}5 \cdot 10^{5}$ тонн. Абсолютный нижний предел массы хранения такой информации в 2012 г. при суммарном объеме контента Всемирной сети 2500 экза-

Оценки сети Интернет

Характеристики сети	Оценка сверху	Средняя оценка	Оценка снизу
Объем информации, формируемой за год, бит	3,00E+23	3,00E+21	6,90E+20
Абсолютный нижний предел массы хранимой информации, кг	3,00E-05	3,00E-07	6,90E-08
Масса хранимой информации при современных технологиях, кг	3,00E+09	3,00E+07	6,90E+06
Энергия, необходимая для записи и/или передачи информации (термодинамическая оценка), Дж	9,00E+02	9,00E+00	2,07E+00
Энергия, необходимая для записи и/или передачи информации (информационная оценка), Дж	9,00E+15	9,00E+13	2,07E+13

байтов ($2{,}5 \cdot 10^{22}$ бит) не менее $2{,}5 \cdot 10^{22}$ бит/10^{28} (бит/кг) = $2{,}5 \cdot 10^{-6}$ кг = 2,5 мг.

Минимальная энергия, необходимая для записи и/или передачи $2{,}5 \cdot 10^{22}$ бит, равна $E \approx 3 \cdot 10^{-21} \times \times 2{,}5 \cdot 10^{22} \approx 75$ Дж.

Оценки характеристик сети Интернет приведены в таблице.

7. Срок действия закона Мура

Закон Мура [17, 18] — эмпирическое наблюдение, изначально сделанное Гордоном Муром, согласно которому (в современной формулировке) число транзисторов, размещаемых на кристалле интегральной схемы, удваивается каждые 24 месяца. Часто цитируемый интервал в 18 месяцев связан с прогнозом Давида Хауса из *Intel*, по мнению которого *производительность* процессоров должна удваиваться каждые 18 месяцев вследствие роста числа транзисторов и быстродействия каждого из них. В 2007 г. Мур заявил, что закон, очевидно, скоро перестанет действовать ввиду атомарной природы вещества и ограничения скорости света.

Информационная емкость I искусственных устройств хранения данных, построенных на базе атомов, не превосходит $10^{28}M$ бит, где M — масса устройства хранения данных, непосредственно используемая для хранения. Так как в настоящее время дифференциальная информационная емкость устройств хранения данных $\sim 10^{14}$ бит/кг, то для устройств, построенных на базе атомов, она может быть повышена не более, чем в $\sim 10^{14}$ раз. Для оценки принимаем, что емкость памяти удваивается каждые 24 месяца, т. е. каждые два года. Тогда предельное значение емкости памяти $10^{28}M$ бит будет достигнуто через 46,5 лет. Всего через полвека.

Аналогичные оценки сроков действия законов могут быть получены, например для закона Крайдера, определяющего изменение стоимости хранения единицы информации на жестком диске; закона Баттерса, определяющего изменение пропускной способности сети; закона Нильсена, определяющего изменение полосы пропускания.

Заключение

Изложенное позволяет сделать следующие выводы.

1. В работе показано, что наряду с физическими мировыми константами существуют информационные мировые константы:
- информационная граница (*information border*, *limit*) $IB = 10^{90}$ бит;
- постоянная памяти (*constant memory*) $IM = 10^{28}$ бит/кг;
- постоянная быстродействия (*constant speed*, *performance*) $IS = 10^{39}$ (оп/с)/кг;
- постоянная информационных затрат (*information costs*) $IC = 10^{-13}$ кг/бит.

2. Данные константы следует добавить в ряд мировых констант: постоянная Планка h, гравитационная постоянная G, скорость света c, постоянная Больцмана k, ...

3. Информационные мировые константы определяют процессы формирования и развития естественных и искусственных объектов, в том числе определяют перспективы развития наномехатроники.

4. Достичь данных мировых констант при исследовании и создании информационных систем невозможно — это ограничения, которые необходимо учитывать, это пределы к которым необходимо стремиться.

Список литературы

1. **Физические** константы. Физический энциклопедический словарь. Т. 5. М.: Советская энциклопедия, 1966. С. 315—316.
2. **Теряев Е. Д., Филимонов Н. Б.** Наномехатроника: состояние, проблемы, перспективы // Мехатроника, автоматизация, управление. 2010. № 1. С. 2—14.
3. **Гуревич И. М.** Законы информатики — основа исследований и проектирования сложных систем связи и управления: Метод. пособие. М.: ЦООНТИ "Экос", 1989. 60 с.
4. **Гуревич И. М.** Законы информатики — основа строения и познания сложных систем. Изд. второе уточ. и доп. М.: Торус Пресс, 2007. 400 с.
5. **Гуревич И. М.** Информационные характеристики физических систем. Изд. второе уточ. и доп. Севастополь: Кипарис, 2010. 260 с.
6. **Гуревич И. М., Урсул А. Д.** Информация — всеобщее свойство материи: Характеристики, оценки, ограничения, следствия. М.: ЛИБРОКОМ, 2011. 312 с.
7. **Гуревич И.** Физическая информатика. LAP Lambert Academic Publishing, 2012. 288 с.
8. **Gurevich I.** Some works on physical informatics. LAP Lambert Academic Publishing, 2012.
9. **Гуревич И. М.** Фундаментальные ограничения на информационные характеристики систем // Труды 3-й мультиконференции по проблемам управления и 7-й научно-технической конференции "Мехатроника, автоматизация, управление" (МАУ-2010). 12—14 октября 2010 г. Санкт-Петербург. С. 144—147.
10. **Гуревич И. М.** Фундаментальные ограничения на емкость устройств хранения данных // Системы и средства информатики. 2010. Вып. 20, № 2. М.: ИПИ РАН. С. 240—254.
11. **Seth L.** Computational capacity of the universe. arxiv.org/abs/quant-ph/0110141 v1 24.
12. **Margolus N., Levitin L. B.** The maximum speed of dynamical evolution // Physica D. 1998. V. 120. P. 188—195.
13. **Дрекслер К. Э.** Машины создания. Грядущая эра нанотехнологии. URL: http://filosof.historic.ru/books/item/f00/s00/z0000328
14. **Эрлих Г. В.** Мифы нанотехнологий. 4 июня 2010. URL: http://www.nanometer.ru/2010/06/04/12756380321857_214259.html
15. **Валиев К. А., Кокин А. А.** Квантовые компьютеры: Надежда и реальность. Москва—Ижевск: Регулярная и хаотическая динамика. 2001.
16. **Валиев К. А.** Квантовые компьютеры и квантовые вычисления // УФН. 2005. Т. 175. № 1.
17. **Moore G. E.** Cramming more components onto integrated circuits // Electronics. 1965. Vol. 38, N 8.
18. **Закон** Мура. URL: http://ru.wikipedia.org/wiki/

УДК 004.822:514

В. В. Грибова, д-р техн. наук, зав. лаб.,
e-mail: gribova@iacp.dvo.ru,
А. С. Клещев, д-р физ.-мат. наук, гл. науч. сотр.,
Институт автоматики и процессов управления
ДВО РАН, г. Владивосток

Технология разработки интеллектуальных сервисов, ориентированных на декларативные предметные базы знаний

Часть 1. Информационные ресурсы

Описана технология разработки и сопровождения интеллектуальных систем как интеллектуальных сервисов. В основе предлагаемой технологии разработки интеллектуальных систем лежит принцип четкого разделения между декларативными знаниями (знаниями предметной области) и процедурными (о методе решения задачи). Представлен обзор двух основных технологий разработки интеллектуальных систем: основные принципы, достоинства и недостатки каждой технологии, дана постановка задачи, а также представлена двухуровневая модель информационных ресурсов.

Ключевые слова: *интеллектуальные системы, облачные технологии, семантические технологии, онтологии, базы знаний, мультиагентные системы*

Введение

Разработка интеллектуальных систем и систем обработки сложноструктурированной информации является трудоемким процессом, но еще более сложным и трудоемким является сопровождение таких систем. Поэтому исследование методов, средств, подходов к их разработке и сопровождению остается одной из актуальных задач.

Можно выделить две ключевые технологии разработки таких систем. *Первая технология* сводится к тому, что разработчику предлагается готовый решатель, и, в соответствии с представлением информации, на которое он рассчитан, он должен сформировать базу знаний, при этом часть знаний о решении задачи встроить в нее. Эту технологию поддерживают оболочки интеллектуальных систем. В качестве способа представления знаний используются базы правил. Первый опыт их применения показал, что базы знаний, основанные на правилах, исключительно сложно не только разрабатывать, но, прежде всего, сопровождать [1], поскольку эксперты не понимают такое представление. Поэтому следующий виток исследований был сконцентрирован на разработке подходов, стратегий, методов и процедур работы с экспертами, способам обработки полученных результатов, а также созданию программных средств, автоматизирующих процессы извлечения знаний из экспертов, специальных текстов и баз данных. Если вопрос приобретения знаний и формирования базы знаний удалось упростить и ориентировать на экспертов, то ее сопровождение — модификация, расширение знаний в процессе эксплуатации системы — осталось трудоемким процессом, для чего такие системы не предназначены, что было отмечено еще в работе [2], поскольку метод решения задачи частично содержится в базе знаний.

Вторая технология исходит от обратного принципа — при построении интеллектуальных систем наиболее сложным для создания и сопровождения компонентом является база знаний и любая другая сложноструктурированная информация. Поэтому она должна разрабатываться так, чтобы быть повторно используемой в различных интеллектуальных системах. Этот подход предполагает сначала формирование метаструктуры или онтологии, по которой может быть создан класс различных баз знаний либо другой сложноструктурированной информации, ей соответствующий. Решатель задачи основан на онтологии обрабатываемой информации и, таким образом, может быть повторно использован для различных интеллектуальных систем, имеющих ту же онтологию обрабатываемой информации. Основные преимущества такой технологии — процедурные знания отделены от декларативных, решатель задач является повторно используемым для класса задач с общей онтологией обрабатываемой информации.

Данная технология применена в работах [3, 4]. Однако подходы к ее реализации имеют существенные различия. В основе технологии "Protege" лежит объектно-ориентированный подход к формированию онтологий. В онтологии определяется иерархия классов для представления основных ее объектов, множество слотов для описания их свойств и отношений между ними, а также множество экземпляров классов. В процессе проектирования онтологий сначала формируются метапонятия верхнего уровня, затем по ним порождаются метапонятия следующего уровня и т. д. Решатель задач состоит из трех основных компонентов: описания алгоритма, включающего все шаги решения задачи, описания так называемой онтологии метода, которое включает спецификации форматов

входных и выходных данных, а также непосредственно программного кода, реализующего метод. Пользовательский интерфейс входит в метод решения задачи. В основе метода решения лежит утверждение, выдвинутое еще Чандрасекараном [5] о том, что в интеллектуальных системах можно выделить классы задач, которые они решают, и для каждого класса задач разработать метод ее решения, который будет повторно использоваться в различных системах (методы могут быть проблемно-зависимыми).

На основе описанной выше технологии в работах [6, 7] была предложена так называемая методология разработки интеллектуальных систем, называемая "Protege", поддерживаемая специализированным инструментарием, который по сути является CASE-системой. Разработка интеллектуальных систем на основе методологии "Protege" сводится к следующим шагам:

- разработка онтологии предметной области;
- генерация системы приобретения знаний на основе онтологии, с использованием которой формируется база знаний;
- выбор метода решения задачи;
- задание отображения между онтологией предметной области и онтологией метода (между онтологией, терминами и атрибутами базы знаний и входными/выходными данными).

Основная задача, которая решается с использованием Protege, — накопить базу онтологий (при этом онтология может содержать разный уровень детализации — как базу общих метапонятий — метаонтологию, так и сформированные на ее основе онтологии следующего, более низкого уровня детализации), базу знаний и методов, чтобы таким образом упростить разработку и сопровождение интеллектуальных систем.

Однако в данном подходе отсутствует четкое разделение между двумя системами понятий, или, как отмечено в работе [8], в "Protege" фактически игнорируется проблема общей организации концептуальной модели, в онтологию "подвешиваются понятия прикладного характера (***Пицца***, ***Страна***, ***Вино***) непосредственно, как подклассы понятия ***Нечто***" (thing). Такой подход "позволяет быстро и относительно просто сконструировать небольшую предметную онтологию", однако при этом отсутствует четкое разделение между базой знаний и онтологией. Возможность разработки общих для разных онтологий методов решения интеллектуальных задач также не представляется универсальным механизмом несмотря на отдельные удачные примеры, представленные в работе [9].

В работах [10, 11] также используются идеология полного разделения процедурных и декларативных знаний, принцип формирования базы знаний по онтологии. Существенным различием является четкое разделение между уровнями и, таким образом, разделение труда между инженерами по знаниям (формируют онтологию через интерфейс, ориентированный на метаязык) и экспертами (формируют базу знаний через сгенерированный по онтологии интерфейс). Однако в данной технологии не было предложено единых технологических принципов формирования решателей задач и пользовательских интерфейсов.

Целью данной работы является описание дальнейшего развития технологии — методов формирования каждого компонента интеллектуальной системы: базы знаний и данных, решателя задач и пользовательского интерфейса, а также платформы для их реализации.

Постановка задачи

Цель любой технологии программирования, в том числе технологии разработки интеллектуальных систем, состоит в снижении трудозатрат, при этом в технологическом цикле одну из ключевых ролей играет поддержка модифицирования или сопровождения интеллектуальной системы в процессе жизненного цикла как наиболее трудозатратной части по сравнению с разработкой (до 70 % трудозатрат). Известными способами достижения этой цели являются повторное использование компонентов, автоматизация проектирования, использование высокоуровневых языков программирования и специализированных инструментов поддержки разработки и сопровождения, ориентированных на группы разработчиков. Помимо указанных способов также можно выделить достаточно новую технологию облачных вычислений (Cloud Computing) [12, 13]. Основная ее идея — предоставление пользователям сервисов (в данном случае интеллектуальных) вместо предоставления им непосредственно версий программных систем для установки на их компьютерах. Преимущества использования технологии облачных вычислений широко обсуждаются в литературе [12—14]; дополнительное преимущество от использования данной технологии — значительное увеличение возможностей по управлению (сопровождению) программных (в том числе интеллектуальных) систем в процессе их жизненного цикла [15], поскольку все компоненты программной системы в процессе всего времени эксплуатации доступны для сопровождения разработчиками (либо другими специалистами, имеющими соответствующие полномочия) в соответствии с постоянно изменяющимися текущими требованиями пользователей, условиями эксплуатации, знаниями предметной области.

Технологию разработки интеллектуальной системы можно разбить на три основные части, соответствующие архитектурным компонентам интеллектуальной системы, — создание и сопровождение базы знаний, решателя задач и пользовательского интерфейса. Поскольку интеллектуальные системы имеют специфический архитектурный компонент — базу знаний, основной частью технологического процесса их создания и сопровождения являются разработка и модификация базы знаний.

При разработке интеллектуальных систем необходима такая технология, при которой эксперты могут формировать и сопровождать базы знаний любой степени сложности самостоятельно, без посредников. Задача разработки такой технологии может быть разбита на две подзадачи:

- отделение терминологии инженера по знаниям от терминологии эксперта, создание методов формирования и сопровождения баз знаний, ориентированных на экспертов, в понятной им терминологии;
- автоматизация разработки редакторов баз знаний, поскольку редакторы баз знаний являются сложными программными средствами; очевидно, что практически невозможно создавать редактор под формирование каждой базы знаний, поэтому необходимы методы и средства автоматизации разработки редакторов баз знаний, ориентированных на экспертов.

Следующей актуальной задачей является разработка универсальных методов создания решателей задач для интеллектуальных систем из повторно используемых компонентов в условиях полного отделения процедурных знаний (знаний о решении задач) от декларативных знаний (баз знаний) для снижения трудоемкости их проектирования и сопровождения.

Наконец, при разработке пользовательских интерфейсов для интеллектуальных систем необходимо разработать технологию их создания, удовлетворяющую следующим требованиям:

- реализация пользовательского интерфейса как web-интерфейса, т. е. поддерживающего взаимодействие пользователя с интеллектуальной системой через браузер;
- поддержка WIMP-интерфейса;
- адаптация пользовательского интерфейса к типам пользователей и характеристикам входных/выходных данных.

Первое и второе требования обусловлены необходимостью создания интеллектуальных систем как интернет-сервисов; при этом важно сохранение традиционного и привычного для большинства пользователей WIMP-интерфейса. Третье требование обусловлено стандартами юзабилити, которые прописывают набор правил организации удобного в использовании и дружественного интерфейса для различных категорий пользователей в зависимости от характеристик входных/выходных данных.

Таким образом, технология разработки интеллектуальных систем должна поддерживать проектирование, реализацию и сопровождение каждого компонента интеллектуальной системы в соответствии с требованиями, изложенными выше при условии согласованности между всеми этими компонентами.

Информационные ресурсы

Под информационными ресурсами в данной работе понимаются базы знаний, базы данных и вообще любая сложным образом организованная информация. Ниже рассматривается *двухуровневая модель* информационных ресурсов, предложенная в работе [10], в соответствии с которой информация порождается или модифицируется под управлением метаинформации. Таким образом, в соответствии с данной моделью любая информация связана с метаинформацией, по которой она порождается или модифицируется. Метаинформация информационного ресурса является представлением графовой грамматики (грамматики, порождающей размеченные графы) языка представления информации.

Метаинформация (МИ) представляет собой пару МИ = (GM, σM), где GM — граф понятий для описания информации (баз знаний и данных), возможно, содержащий циклы и петли, а σM — разметка этого графа. Граф GM есть тройка <Vertexes, Arcs, RootVertex>, где Vertexes — множество вершин графа, Arcs — множество дуг графа, RootVertex — начальная вершина графа. Множество вершин графа $\text{Vertexes} = \{\text{Vertex}_i\}_{i=1}^{vertexescount}$ состоит из двух непересекающихся подмножеств — терминальных вершин Terminal_Vertexes, из которых не выходит ни одной дуги, и нетерминальных вершин Neterminal_Vertexes, из которых выходит хотя бы одна дуга; $\text{Vertex}_i \in$ $\in$ Neterminal_Vertexes $\cup$ Terminal_Vertexes. Начальной вершиной графа RootVertex является нетерминальная вершина, RootVertex $\in$ Neterminal_ Vertexes.

Множество дуг графа Arcs есть множество пар $\{\text{Arc}_i\}_{i=0}^{arcscount}$, $\text{Arc}_i = <\text{VertexFrom}_i, \text{VertexTo}_i>$, где VertexFrom_i, VertexTo_i — вершины, из которых выходит и в которые входит дуга Arc_i соответственно, $\text{VertexFrom}_i \in$ Neterminal_Vertexes, $\text{VertexTo}_i \in$ Vertexes.

Средства разметки σM включают в себя разметку вершин σV и разметку дуг σA. Эта разметка позволяет при описании графа метаинформации задать ограничения на структуру и содержание графа информации.

Разметка вершин σV задается следующим образом: начальная вершина отображается в имя графа метаинформации, т. е. σV(RootVertex) = Имя Графа Метаинформации; нетерминальные вершины графа отображаются в пару: Имена понятий предметной области либо класса предметных областей, имеющих одну и ту же метаинформацию; Тип структурированного набора, т. е. σV(Neterminal_Vertexes) $\in$ $\in$ (Имена понятий предметной области × Тип структурированного набора). Назовем множество дуг графа, выходящих из одной вершины, *структурированным набором дуг*. Граф метаинформации имеет два возможных типа структурированных наборов: непустое множество и непустое множество альтернатив, т. е. Тип структурированного набора $\in$ {Непустое множество, Непустое множество альтернатив}.

Терминальные вершины могут иметь разметку двух типов — терминал, описывающий сорт, и терминал, описывающий значение, т. е. σV(Terminal_ Vertexes) $\in$ $\{\text{Терминал, описывающий Сорт}_i\}_{i=2}^{valuecount} \cup$ $\cup \{\text{Терминал, описывающий Значение}\}_{i=1}^{vertexescount}$. Терминал, описывающий сорт, есть пара Терминал,

описывающий Сорт$_i$ = <Имя$_i$, Значение$_i$>, где Имя$_i$ — строка символов конечной длины, Значение$_i$ принадлежит множеству имен базовых сортов, Значение$_i$ ∈ Имена Базовых сортов. Имена Базовых сортов = {Строка, Целое, Вещественное, Логическое, Дата, Дата-время, Бинарные данные}. Терминал, описывающий значение, есть тройка Терминал, описывающий Значение$_i$ = <Имя$_i$, Сорт$_i$, Значение$_i$>, где Имя$_i$ — строка символов конечной длины, Сорт$_i$ ∈ Имена Базовых сортов, Значение$_i$ ∈ Сорт$_i$. Если Сорт$_i$ = Строка, то значением является строка конечной длины; если Сорт$_i$ = Целое или Сорт$_i$ = = Вещественное, то значением является целое или вещественное число соответственно; если Сорт$_i$ = = Логическое, то значение принадлежит множеству логических значений {истина, ложь}, если Сорт$_i$ = = Дата или Сорт$_i$ = Дата и Время, то значение задается в формате <день>.<месяц>.<год><часы>:<минуты>:<секунды>, где <год> представлен четырьмя цифрами, остальные компоненты — двумя; если Сорт$_i$ = Бинарные данные, то значением является массив байтов конечной длины.

Разметка дуг σA задается следующим образом: σA(Arc$_i$) = Спецификатор дуги. Спецификатор дуги есть пара <Тип спецификатора, Факультативность>, т. е. Спецификатор дуги = <Тип спецификатора, Факультативность>, Тип спецификатора ∈ ∈ {копия, единственность, непустое множество}. Все спецификаторы могут использоваться в сочетании со спецификатором "факультативность", однако спецификатор "факультативность" может иметь лишь дуги, входящие в структурированный набор с типом "непустое множество".

Информация, как и метаинформация, представляет собой пару И = (GI, σI), где GI — граф (сеть) понятий, который, в отличие от графа метаинформации, не содержит циклов и петель, σI — разметка графа. Граф информации GI есть тройка <IVertexes, IArcs, IRootVertex>, где множество вершин графа (IVertexes), множество дуг графа (IArcs), корневая вершина (IRootVertex) описываются так же, как в графе метаинформации, средства разметки σI, в отличие от графа метаинформации, включают в себя только разметку вершин, т. е. σI ≡ σV. Все вершины графа информации отображаются в пару σV = (SM, RM), где SM — служебная метка; RM — метка соответствия между информацией и метаинформацией. Служебные метки могут быть двух типов — порожденные вершины (вершины, из которых выполнен шаг порождения) и непорожденные (вершины, из которых еще не выполнен шаг порождения). Метка соответствия RM — это Имя понятия предметной области, которому соответствует либо Имя понятия предметной области, заданное в метаинформации, либо некоторое имя, входящее в класс предметных областей, также заданный в метаинформации.

Соответствие между информацией и метаинформацией задается следующим образом. Каждой вершине в графе информации соответствует некоторая единственная вершина в графе метаинформации, называемая ее вершиной-прототипом (соответственно каждой вершине в графе метаинформации соответствует некоторое множество вершин в графе информации, причем множества вершин в графе информации, соответствующие различным вершинам в графе метаинформации, не пересекаются). Число вершин в графе информации определяется типом структурированного набора, заданного в разметке графа метаинформации. Так, если в разметке некоторой вершины-прототипа графа метаинформации структурированный набор задается типом "непустое множество", то в графе информации из вершины, соответствующей этой вершине-прототипу, выходит i дуг, где $0 \leqslant i \leqslant N$, N — число дуг, выходящих из вершины-прототипа в графе метаинформации; если в разметке вершины-прототипа структурированный набор задается типом "непустое множество альтернатив", то в графе информации из вершины, соответствующей вершине-прототипу, выходит единственная дуга.

Структура и разметка графа информации также зависит от типа спецификатора дуги метаинформации. Если в графе метаинформации дуга имеет Тип спецификатора = копия, то разметка вершины, в которую входит дуга со спецификатором данного типа, сохраняется в графе информации, т. е. Имя понятия предметной области в графе метаинформации совпадает с Именем понятия предметной области в графе информации; если в графе метаинформации дуга имеет Тип спецификатора = единственность, то этой дуге в графе информации соответствует только одна дуга, входящая в вершину, разметка которой соответствует Имени предметной области (у нетерминалов) или значению (у терминалов); если в графе метаинформации дуга имеет Тип спецификатора = непустое множество, то этой дуге в графе информации соответствует множество дуг, входящих в вершины, разметка которых соответствует некоторому Имени предметной области (у нетерминалов) или значению (у терминалов). Факультативность, указанная в разметке дуги графа метаинформации, означает, что в графе информации может отсутствовать понятие (термин) из метаинформации дуга, и, соответственно, вершина, в которую эта дуга входит.

В случае, когда в графе информации все вершины помечены служебными метками "порожденные вершины", граф информации считается полностью порожденным в соответствии с метаинформацией, в противном случае порождение является частичным, т. е. считается, что вершина графа информации с меткой "непорожденная вершина" соответствует своей вершине-прототипу в графе информации.

Граф метаинформации можно рассматривать как порождающую графовую грамматику (грамматику, порождающую графы информации), определяющую язык, в терминах которого может быть порождено множество графов информации на этом языке. Поскольку грамматика является графовой,

то информация также представляется графом. В искусственном интеллекте граф такой структуры называется семантической сетью и предназначен для представления смысла некоторого высказывания (смысла текста, представленного в форме графа).

Проводя аналогию с программированием, можно заметить, что в предлагаемой модели, с одной стороны, вся информация представляется в едином унифицированном формате — в форме семантических сетей, в чем можно усмотреть прямое сходство с функциональным программированием, в котором все данные представлены также унифицировано — списками, с другой стороны, в традиционном программировании общепринятой практикой является разделение информации (входной, промежуточной и выходной) на множество разрозненных информационных единиц, соответствующих типам данных (программа имеет множество переменных различных типов). В связи с этим программист должен знать все неявные связи между переменными и выражать их через алгоритмы обработки.

Предлагаемая модель основана на обратном принципе: в программе число семантических сетей невелико, они имеют более сложную структуру, но эта структура явно определяет связи между всеми информационными единицами.

В качестве примера интеллектуальной системы, обрабатывающей такие структуры данных, можно рассмотреть экспертную систему медицинской диагностики. Входными данными для нее являются: база знаний, содержащая описание заболеваний некоторого раздела медицины; данные, содержащиеся в истории болезни пациента. Выходными данными является объяснение гипотезы о том, что пациент здоров, и о том, что пациент болен различными известными системе заболеваниями. Каждая из этих трех структур (база знаний, история болезни, объяснение) представлена семантической сетью.

Для компьютерного представления метаинформации и информации разработан метаязык (язык ИРУО), реализующий описанную выше модель, и универсальный редактор (редактор ИРУО) [10], имеющий два режима. Для формирования метаинформации на языке ИРУО используется режим инженера знаний. С помощью структурного редактора инженер знаний формирует на языке ИРУО размеченный граф метаинформации. По этому графу метаинформации редактор ИРУО в режиме эксперта генерирует интерфейс для эксперта (специалиста) предметной области. Информация, сформированная экспертом с помощью редактора ИРУО в режиме эксперта, также сохраняется на языке ИРУО. Таким образом, редактор по сути является интерпретатором метаинформации, представленной на языке ИРУО, который формирует информацию также на языке ИРУО. Такой подход обеспечил реализацию единственного редактора как метаинформации, так и информации для любой метаинформации, представленной на языке ИРУО.

Заключение

В работе рассмотрены две ключевые технологии к разработке интеллектуальных систем, их основные достоинства и недостатки. Первая технология основывается на предоставлении разработчику готового решателя задач, в соответствии с которым формируется база знаний, также частично содержащая метод решения задачи. Вторая технология основана на первоначальной разработке базы знаний, желательно повторно используемой, и разработке решателя задач, который основан на онтологии обрабатываемой информации. В рамках развития второй технологии рассмотрена постановка задачи и описан подход к проектированию информационных компонентов интеллектуальной системы — онтологии и базы знаний.

Работа выполнена при финансовой поддержке ДВО РАН в рамках Программы ОНИТ РАН (проект 12-I-ОНИТ-04) и РФФИ (проект 12-07-00179).

Список литературы

1. **Bachant J. McDermott J.** R1 revisited: four years in the trenches // The AI Magazine. 1984. Vol. 5, N 3. P. 21—32.

2. **Compton P., Horn K., Quinlan R., Lazarus L.** Maintaining an expert system // Proceedings of the fourth Australian Conference on Applications of Expert Systems. 1988. P. 110—129.

3. **Gennari J. H., Musen M. A., Fergerson R. W.** etc. The evolution of Protégé: An environment for knowledge-based systems development // International Journal of Human-Computer Studies. 2003. 58 (1). P. 89—123.

4. **Клещев А. С., Орлов В. А.** Компьютерные банки знаний. Многоцелевой банк знаний // Информационные технологии. 2006. № 2. С. 2—8.

5. **Chandrasekaran B.** Generic tasks in knowledge-based reasoning High-level building blocks for expert system design // IEEE Expert. 1986. 1 (1). P. 23—30.

6. **Gennari J. H., Musen M. A., Fergerson R. W., Grosso W. E., Noy N. F., Crubezy M., Eriksson H., Tu S. W.** The evolution of Protégé. An environment for knowledge-based systems development // International Journal of Human-ComputerStudies. 2003. 38 (1). P. 89—123

7. **Musen M.** Technology for building intelligent systems: from psychology to engineering // Proc. of the Nebraska Symposium on Motivation. 2007. P. 145—84.

8. **Пивоварова Л. М., Рубашкин В. Ш.** Компоненты онтологических систем и их реализация в современных проектах // ИНТЕРНЕТ И СОВРЕМЕННОЕ ОБЩЕСТВО: Тр. X Всероссийской объединенной конф. СПб.: Факультет филологии и искусств СПбГУ, 2007. С. 277—279.

9. **Crubezy M., Musen M. A.** Ontologies in supportof problem-solving // In S. Staab & R. Studer (Eds.) Handbook on ontologies. Berlin: Springer, 2004. P. 321—341.

10. **Клещев А. С., Орлов В. А.** Компьютерные банки знаний. Универсальный подход к решению проблемы редактирования информации // Информационные технологии. 2006. № 5. С. 25—31.

11. **Клещев А. С., Черняховская М.Ю., Грибова В. В.** и др. Мультидисциплинарная система управления информационными ресурсами различных уровней общности // Проблемы управления, 2006. № 4. С. 64—68.

12. **Gillam L.** Cloud Computing: Principles, Systems and Applications // Computer Communications and Networks. L.: Springer, 2010. 379 p.

13. **Табаков В. В.** Облачные вычисления — технологическая инновация // Сб. матер. Второй междунар. научно-практ. конф. "Проблемы развития инновационно-креативной экономики". URL: http://econference.ru/blog/conf06/227.html.

14. **Магнус Калькуль.** Ясное небо до самого горизонта: "облачные" вычисления и безопасность "из облака". URL: http://www.securelist.com/ru/analysis/204007652/Yasnoe_nebo_do_samogo_gorizonta_oblachnye_vychisleniya_i_bezopasnost_iz_oblaka.

15. **Грибова В. В., Клещев А. С., Шалфеева Е. А.** Управление интеллектуальными системами // Изв. РАН. Теория и системы управления, 2010. № 6. С. 122—137.

16. **Грибова В. В., Федорищев Л. А.** Интернет-комплекс для создания обучающих систем с виртуальной реальностью // Дистанционное и виртуальное обучение. 2012. № 7. С. 4—12.

17. **Грибова В. В., Клещев А. С., Крылов** и др. Проект IACPaaS — развиваемый комплекс для разработки, управления и использования интеллектуальных систем // Искусственный интеллект и принятие решений. 2011. № 1. С. 27—35.

УДК 004.89

И. Л. Букатова,
д-р физ.-мат. наук, вед. науч. сотр.,
Институт радиотехники и электроники
им. В. А. Котельникова РАН, г. Москва
e-mail: _bil2007@bk.ru

Развитие эволюционного моделирования в России: концепции, приложения, перспективы

Статья посвящена сорокалетней годовщине официального признания эволюционного моделирования в России как перспективного научно-практического направления. Определены причины опережающего развития парадигмы эволюционного моделирования в отечественных исследованиях.

Ключевые слова: *база знаний, вложенные системы, интеграция, интеллектуализация, интеллектуальная технология, целостно-эволюционные средства, эволюционные алгоритмы*

Случайностей ведь нет.
Что кажется подчас
Лишь случаем слепым,
То рождено
Источником глубоким.

Иоганн Шиллер

Введение

Наблюдая природу, осмысливая ее закономерности, в особенности, механизмы развития и естественной эволюции, человек стремился использовать полученные знания и опыт для удовлетворения своих потребностей. Взаимодействие человека с природой получило существенный стимул, когда у человека расширились его сознание и интеллект: сформировалось абстрактное восприятие окружающей среды. В результате появилась возможность осуществлять репродуктивную деятельность — сначала копируя и подражая, затем по аналогии, а затем и продуктивную — творческую.

Истоки эволюционных идей мы находим у античных философов, в трудах ученых и теологов средневековья и эпохи возрождения, а также у мыслителей-натуралистов более поздних веков. При этом рассматривались различные аспекты развития и вечного становления, проблемы наследственности и естественного отбора, развивались разнообразные концепции трансформирования и систематизирования законов эволюционного развития. Определенным этапом такого развития можно считать создание первого эволюционного учения Ж. Б. Ламарком, относящегося к началу XVIII века. Дальнейшее развитие идей эволюционизма, утверждение в биологии концепции вида, весь ход развития естествознания подготовили почву для появления учения Ч. Дарвина об изменяемости видов и отборе [1].

Дарвинизм как общее системное описание процесса естественной эволюции отражает эволюцию как происхождение видов, указывая на главные действующие факторы эволюции: наследственность, изменчивость и естественный отбор. Теория Дарвина была развита в различных аспектах его современниками.

Большой вклад в развитие теоретических основ эволюционного учения в более поздние, после дарвиновские периоды, внесли отечественные ученые: В. О. Ковалевский, И. М. Мечников, А. Н. Северцов, К. М. Завадский, В. И. Вернадский, И. И. Шмальгаузен и другие.

Так, например, в 1968 г. выходит в свет книга И. И. Шмальгаузена "Кибернетические вопросы биологии" [2], в которой (по мнению ее редакторов Р. Л. Берга и А. А. Ляпунова) автор "перевел теорию Дарвина на язык кибернетики" и осуществил на современном уровне синтез концепции Кювье, Дарвина и Вернадского. Конструктивный аспект эволюционных механизмов (не только в сфере органического мира), таким образом, уже был успешно апробирован отечественными исследователями.

Существует большое разнообразие эволюционных теорий, в которых эволюционные взгляды причудливо переплетаются между собой, иногда сочетаясь с антиэволюционными концепциями [1, 3]. Современные учения учитывают также законы генетики, изучающей изменение генетических систем в процессе исторического развития организмов [4, 5]. Следствием такой ситуации является то, что до настоящего времени не существует единой, общепризнанной теории эволюции. Такое положение объективно связано, в первую очередь, с исключительной сложностью самого процесса естественной эволюции. Для понимания механизмов эволюции и последующего их адекватного моделирования в целях решения практических задач посредством моделирования, отражающего процессы деятельностного, познавательного, осмысленного человеческого бытия, необходимо рассматривать множество концепций, дополняющих друг друга. Фактически мы имеем три компонента процесса развития цивилизации: действие, знание, понимание, которые определяют проблему накопления информации, формирования и использования знаний в процессе эволюции, формируют субъект — объектное представление процессов развития.

"Заменить процесс моделирования человека процессом моделирования его эволюции" — такова основная идея эволюционного моделирования, которая была предложена американскими авторами Л. Фогелем, А. Оуэнсом, М. Уолшем в книге "Искусственный интеллект и эволюционное моделирование" в 1965 г. [6].

В нашей стране этой идее предшествовали основополагающие работы советских ученых по моделированию эволюционных механизмов: А. Г. Ивахненко, Л. А. Растригина, Ю. М. Неймарка, Я. З. Цыпкина, которые в различных аспектах использовали аналоги мутаций и эволюции. Так, например, в рамках метода группового учета аргументов была успешно воспроизведена схема массовой селекции в целях генерации адекватных моделей сложного объекта (А. Г. Ивахненко [7], 1969); в стохастических процедурах поиска получили обобщение идеи активного целенаправленного использования случайности в качестве эволюционных механизмов оптимизации (Л. А. Растригин [8], 1968), идеи эволюции, включая моделирование трансформации гена и генома, использовались для адаптации популяции детерминированных и стохастических автоматов при оптимизации дискретных объектов

(Ю. И. Неймарк [9], 1967, М. Л. Цетлин [10], 1969); основные свойства сложных систем: обучение, самообучение, адаптация, самоадаптация, самоорганизация, являющиеся отражением соответствующих эволюционных процессов, были четко сформулированы, классифицированы и алгоритмически определены (Я. З. Цыпкин [11], 1970).

Аналогичные идеи моделирования эволюционных механизмов практически одновременно развивались далее в работах как отечественных, так и зарубежных ученых: генетические алгоритмы — J. H. Holland, 1975; эволюционные стратегии — I. Rechenberg, 1973, H. P. Schwefel, 1977, 1981; генетическое программирование — J. R. Koza, 1992, 1994; эволюционное программирование — L. J. Fogel, A. J. Owens, M. J. Walsh, 1966; эволюционные вычисления — D. B. Fogel, 1995 и др.

Перечисленные выше направления работ отражают труды международных конференций, проходивших с 1989 г. в Европе и США по тематике эволюционных стратегий, эволюционного программирования, генетических алгоритмов, генетического программирования, искусственной жизни, эволюционных вычислений. При этом нетрудно заметить изначально имевшиеся и ставшие со временем существенными методологические различия в отечественных и зарубежных разработках.

Напомним, что *эволюционные стратегии* — это алгоритмы, созданные в Германии для решения оптимизационных задач и основанные на принципах природной эволюции. *Эволюционное программирование* представляет собой подход, предложенный американскими учеными вначале в рамках теории конечных автоматов и обобщенный позднее для приложений к проблемам оптимизации. *Генетический алгоритм* (ГА) — алгоритм аналогий естественного отбора и генетических преобразований в процессе исторического развития индивида. *Генетическое программирование* — синтез на основе ГА компьютерной программы для задачи с неизвестным алгоритмом решения.

Все методы эволюционной обработки информации используют популяции потенциальных решений и реализуют принцип селекции и принцип преобразования наиболее приспособленных особей, однако они различаются способом представления особей и организацией процесса селекции.

Так, в наших исследованиях первоначальная парадигма эволюционного моделирования была преобразована в конструктивную идею *замены процесса моделирования сложного объекта процессом моделирования его эволюции*. Эта идея, в отличие от первоначальной идеи Л. Фогеля о создании средства искусственного интеллекта путем моделирования эволюции интеллекта человека, нашла убедительное практическое обоснование и быстрое воплощение с момента выхода русского перевода указанной выше книги в 1969 г.

На пути к эволюционной информатике

Сорокалетняя история разработок эволюционных и генетических алгоритмов, проводимых разными группами ученых России, Украины, Латвии и других государств бывшего СССР, предоставляет нам отличные возможности для конструктивного анализа основных идей, эффективных механизмов, очевидных достижений и дальнейших перспектив эволюционного моделирования.

К началу 80-х годов научные исследования в СССР по тематике эволюционного моделирования характеризуются следующим образом:

- эволюционное моделирование сформировалось как перспективное научное направление структурного синтеза моделей исследуемого объекта на основе имитации на ЭВМ эволюционных механизмов в системах обработки информации в условиях неопределенности и временных ограничений;
- показана высокая эффективность эволюционных средств (эволюционных вычислений и алгоритмов) при решении задач управления, распознавания, прогнозирования, идентификации, автоматизации проектирования вычислительных комплексов и автоматизации научного эксперимента;
- результаты разноплановых разработок в ведущих научных институтах и центрах СССР обсуждались на двух Всесоюзных семинарах, организованных ИРЭ АН СССР, и на секциях более 20 Всесоюзных конференций, посвященных различным системам обработки данных;
- по совокупной сравнительной оценке исследования по эволюционному моделированию в этот период времени в СССР существенно отличались от западных работ по эволюционной тематике. К сожалению, эти отечественные разработки в силу ряда объективных причин (ограничение на контакты, специфика разработок и областей приложений, процессы распада СССР) были слабо интегрированы в мировую науку и практически неизвестны зарубежным специалистам.

Активно выраженный интерес к генетическим и эволюционным алгоритмам в Западной Европе и США стал появляться в середине 80-х годов, когда увеличилось число конференций по ГА; появились конференции по эволюционным и мягким вычислениям; резко возросло число публикаций по тематике эволюционной обработки информации. В этот же период времени на основе проводимых в СССР исследований эволюционные технологии и средства уже были разработаны как инструментарий *нового научного направления — **эволюционной информатики***.

Эволюционная информатика — это совокупность алгоритмических, программных и аппаратных средств, основанных на имитации механизмов естественной эволюции в целях синтеза эволюционирующих структур обработки данных в условиях недостаточности информации, ее динамики и неопределенности.

Отечественный, отличный от западного, аспект развития эволюционного моделирования впервые был аргументирован в 1987 г. в статье [12]. При обсуждении эффективных механизмов и перспективных применений эволюционных алгоритмов авторы констатировали формирование новой области информатики — эволюционной информатики. Отмечалось, что методы и средства эволюционной информатики являются дальнейшим развитием идеи обучающихся вычислительных систем и проблемно-ориентированных спецпроцессоров [12—14]. Было показано, что отечественная элементная база оптоэлектроники уже в 80-е годы располагала возможностями для создания эволюционирующих спецпроцессоров. Обсуждалась эффективность применения эволюционных инструментальных средств при автоматизации научных исследований, обработке радиофизических измерений, а также в биофизике, медицине, экологии, глобальной экологии.

Координация научных исследований

К началу 90-х годов исследования по эволюционной информатике с развитием парадигмы в различных аспектах осуществлялись усилиями коллективов организаций, институтов и ведомств, возглавляемых известными учеными: А. Г. Ивахненко, Л. А. Растригиным, В. В. Нечаевым, И. Л. Букатовой, Д. Х. Имаевым, В. Д. Дмитриенко, Э. М. Куссулем, В. Ф. Иродовым, М. А. Раковым, В. Л. Кузнецовой, а также силами отдельных специалистов, внесших существенный вклад в практические приложения: С. Н. Гринченко, М. А. Семеновым, Д. А. Теркелем, В. Г. Редько, В. В. Корнеевым, А. М. Пинсоном, Е. Л. Петсоном, П. М. Брусиловским, В. А. Марковым, В. В. Казаневской, Г. С. Цветковым, А. Н. Антамошкиным, учениками и последователями академика А. Г. Ивахненко.

С учетом научной и практической важности описанного выше состояния работ в области исследований по эволюционной тематике по инициативе выдающихся ученых: академика АН СССР Ю. В. Гуляева, д-ра физ.-мат. наук В. Ф. Крапивина, проф. Л. А. Растригина и чл.-кор. АН СССР Я. З. Цыпкина при Президиуме АН СССР и Научном Совете по комплексной проблеме "Кибернетика" 10 апреля 1981 г. была создана подкомиссия по проблеме эволюционного моделирования (председатель д-р физ.-мат. нак И. Л. Букатова).

На подкомиссию были возложены задачи координации, обсуждения и поддержки перспективных исследований по данной тематике, организации секций, семинаров и конференций, издание сборников научных трудов и т. п. В разработке идеологии, в обсуждении перспективных направлений исследований и приложений, в идейной, организационной и материальной поддержке в рамках соответствующих проектов неоценимую конкретную помощь работе подкомиссии постоянно оказывали выдающиеся отечественные ученые академики АН СССР А. А. Красовский, Г. С. Поспелов, Д. А. Поспелов.

К концу 90-х годов число организаций, координируемых подкомиссией, возросло до 40. С их участием в ИРЭ АН СССР (в последующем ИРЭ РАН) ежегодно проводились Всесоюзные совещания и конференции по эволюционному моделированию, на которых, в том числе, разрабатывались и формировались перспективные направления исследований. В целях обсуждения теоретических и прикладных проблем эволюционного моделирования и его практического применения работали секции на многих, регулярно действовавших форумах:

- Международном симпозиуме "Инженерная экология", г. Москва, (с 1989 г.);
- Международной конференции по проблемам экоинформатики, г. Звенигород (1992 г.);
- Всесоюзном семинаре "Эволюционное моделирование и обработка данных радиофизического эксперимента", г. Звенигород (1983 г., 1984 г., 1987 г.);
- ежегодной Международной конференции "Новые информационные технологии", г. Ялта-Гурзуф, на которой по инициативе проф. Е. Л. Глориозова с 1991 г. успешно работала секция по эволюционному моделированию.

Работали секции и на других симпозиумах и семинарах, организованных известными учеными России, Украины, Латвии, Узбекистана, Армении.

Усилиями проф. Е. Л. Глориозова и проф. Э. Гудмана с 1993 г. начался активный обмен результатами многолетних отечественных и зарубежных исследований. Этот процесс особенно активизировался в 1996 г., когда с привлечением иностранных специалистов проф. Е. Л. Глориозов и проф. Э. Гудман организовали первую Международную конференцию и Дискуссионный научный клуб "Генетические алгоритмы и эволюционное моделирование" GA + ES'96 в г. Ялта-Гурзуф, Крым.

Кроме того, по инициативе проф. Э. Гудмана и акад. В. С. Бурцева в Москве в 1996 г. состоялась также первая Международная конференция "Эволюционные вычисления и их применения (EvCA'96)".

К сожалению, труды этих конференций [15, 16], проходивших в годы фактического распада научных школ бывшего СССР, отражают в основном западные исследования по эволюционным алгоритмам через призму идей ГА. В то же время в исследованиях тех лет, как и в современных разработках, на наш взгляд, отсутствует или недостаточно реализуется ряд основных идей и механизмов, обусловливающих эффективность средств эволюционной информатики. Даже спустя полтора десятилетия всесторонний анализ факторов этой эффективности представляет, на наш взгляд, определенный научно-прикладной интерес.

Основные концепции эволюционной информатики

Анализ эффективности средств эволюционной информатики предполагает их описание как совокупности, во-первых, основных процессов, моделирующих естественную эволюцию, и, во-вторых, как информационных технологий, реализующих получение знания в процессе моделирования. Заметим, что и то, и другое описания выступают в качестве *моделей*, отражающих наше понимание и наше представление об информационной технологии и естественной эволюции. Как следствие, актуален анализ развития парадигмы эволюционного моделирования как *интегральный результат исследований*, проведенных разными учеными. С этой целью рассматриваются концепции, отражающие постепенное развитие изначальных идей эволюционного моделирования по включению всех компонентов (принципов) естественной эволюции и развития эволюционных информационно-вычислительных технологий.

Заметим, что формирование таких концепций — весьма сложная задача, поскольку в настоящий момент известны десятки эволюционных учений, теорий и схем моделирования механизмов и процессов эволюции по аналогии с природными системами. Соответственно существует множество методов эволюционной обработки: генетические алгоритмы, эволюционное и генетическое программирование, эволюционные стратегии, эволюционный случайный поиск и др. Особенности этих методов естественным образом дополняют изложенные ниже концептуальные схемы и широко представлены в литературе. Поэтому в целях конструктивного восприятия данного материала они не выделены в описаниях основных концепций.

Дальнейшее изложение посвящено краткому обзору через призму развития, концепций наиболее значимых, но ныне мало доступных для читателя, направлений отечественных исследований по эволюционной информатике за период 1969—1996 гг. Такой обзор начнем с некоторых замечаний.

Замечание 1. *Любой эволюционный алгоритм циклично реализует известные процедуры естественной эволюции:* ***популяция → селекция → родители → скрещивание → потомки → отбор → замещение → обновленная популяция.***

Замечание 2. *Алгоритмы различаются изначально задаваемой информацией о задаче: либо используется ГЕНОТИП — представление пространства поиска в закодированном виде комплекса генов, составляющих хромосом, либо используется ФЕНОТИП — представление пространства решений задачи в привычном для восприятия виде.*

Обучающиеся, адаптивные и самоорганизующиеся эволюционные вычисления

Развитие первоначальной парадигмы эволюционного моделирования в отечественных исследованиях 1969—1992 гг. представляется на основе языка формализации [17, 18] в рамках *концепции обучающихся и адаптивных вычислений* следующим образом:

- в эволюционных схемах рассматривается эволюционирующий объект в виде структурированной модели (С-модели);
- предложена эволюционная схема формообразования, имитирующая факторы биологической эволюции на обобщенных и структурированных моделях (см. [18]);
- при моделировании эволюции расширен набор формализованных механизмов и эволюционных схем;
- разработаны и исследованы эволюционные схемы, включающие процессы обучения и адаптации;
- исследуются следующие эволюционные схемы: эволюционная стратегия поиска, генетический метод поиска, автоматная модель эволюции, эволюционный синтез структуры, видовая оптимизация.

Основные формализованные компоненты любой эволюционной схемы: ***цель, наследственная изменчивость, соревнование, отбор***.

Под ***целью*** подразумевается результат решения той или иной решаемой задачи оптимизации: одно- и многопараметрическая, одно- и многокритериальная.

Компонент ***наследственная изменчивость*** включает понятия: С-модель, класс С-моделей, структурный синтез С-моделей. Фиксация класса при эволюционном синтезе С-модели определяет простые элементы С-модели и список базовых, простых и составных изменений, не выводящих С-модель из заданного класса. При решении конкретных задач обработки сигналов (предсказание, распознавание, классификация, восстановление) исследованы следующие классы С-моделей: конечные автоматы, сети Петри, древовидные и многорядные модели, различные сети функциональных элементов, реализующие логические и функциональные схемы, функциональные ряды и т. п.

В компоненте ***соревнование*** указывается конкретная функция, по которой оценивается эффективность синтезированной модели.

Отбор содержит процедуру (или правило, функцию), по которой синтезированные модели пополняют популяцию.

Необходимое разнообразие эволюционных вычислений определяется, таким образом, *вариантами* исполнения механизмов эволюции: стратегиями процедур случайного поиска, списками режимов изменений, стохастическими реализациями режимов изменений С-моделей, процедурами обучения и адаптации, критериями отбора, классами С-моделей и т. д.

Отметим, что известная схема видовой оптимизации реализует процесс выживания вида за счет популяции с лучшей приспособительной функцией, т. е. моделирует параллельную эволюцию нескольких популяций. Для реализации модели параллельной эволюции в работе [18] определены параметры и факторы, влияющие на формирование конкретной ветви искусственной эволюции. Рассмотрены два типа эволюционных вычислений, различающихся процессами самоорганизации дарвиновских схем по указанным выше параметрам разнообразия.

Данные исследования формируют *концепцию самоорганизующихся вычислений* [18].

1. Эволюционное вычисление осуществляется одновременно по многим ветвям эволюции на основе эволюционной информационно-вычислительной технологии (ЭИВТ) путем имитации механизмов биологической эволюции, действующих на всех уровнях организации живого как "вложенных" системах [15, 17, 18].

2. Отдельная ветвь эволюции реализует процесс эволюционной идентификации структурированных моделей в соответствии с концепцией обучающихся и адаптивных вычислений. Самоорганизующиеся вычисления одновременно по многим ветвям имитируются разнообразием конкретного исполнения механизмов эволюции, указанных выше.

3. ЭИВТ является метатехнологией, так как включает в качестве компонент блоки пошагового и поэтапного управления, обеспечивающие одношаговые и многошаговые коррекции процессов эволюционных вычислений в целях повышения их эффективности [18].

Целостно-эволюционное приобретение знаний и интеллектуальные метатехнологии

Описание и анализ эволюционных средств как ЭИВТ приводит к необходимости анализа процесса приобретения (формирования и использования) знаний в совокупности вложенных систем "макросистема + (человек + И-технология) + И-технология" как целостной системы с позиций целостно-эволюционного подхода. Воспроизведение интеграции многошаговых эволюционных процессов в эволюционных схемах и соответствующие процедуры приобретения знаний составляют *концепцию целостно-эволюционного приобретения знаний*:

1. Знание понимается как фиксация существования объективных связей между предметами реального мира. Такая фиксация осуществляется в процессе приобретения знаний в виде познавательной модели (ПМ) на основе той или иной языковой конструкции. Наличие различных языковых конструкций обусловливает полиморфность ПМ.

2. Приобретение знаний трактуется как познавательный процесс, происходящий в структуре "вложенных" систем. Он включает следующие компоненты: знание реального мира (когнитивное воздействие макросистемы), сознание или ПМ (элементы интегрированных баз знаний), средства синтеза, анализа или интерпретации и средства использования ПМ (процессы интеллектуализации и целостно-эволюционной интеграции — продуктивное воздействие макросистемы).

3. Знания реального мира поступают в виде массива примеров (когнитивных воздействий макросистемы) об изучаемом объекте. При этом С-модель рассматривается как базовый элемент интегрированного знания, имеющий различные свойства, которые отражаются в понятиях: морфологических, инструментальных, процессуальных, целевых.

4. Процесс целостно-эволюционной интеграции знаний является открытой системой, для которой характерны эволюционные процессы системообразования: коррекция,

обучение, самообучение, адаптация, взаимоадаптация, самоорганизация. Интеграция эволюционных процессов осуществляется когнитивно-продуктивной технологией, состав которой актуализируется в соответствии с проблемной ситуацией, т. е. с когнитивным и продуктивным воздействием вложенных систем.

5. Формирование нового знания или приобретение некоторой совокупности базовых элементов системного свойства, осуществляется в процессе эволюционного структурного синтеза при формировании С-модели, а также за счет кооперации эволюционных процессов в соответствии с актуализированной когнитивно-продуктивной технологией.

Интеллектуализация (оснащение И-технологиями) различных систем в целях эффективного решения широкого спектра задач человеческой деятельности осуществляется в соответствии с концепцией целостно-эволюционного (ЦЭ) приобретения знаний (см. выше); процессами ЦЭ-интеграции и ЦЭ-интеллектуализации; средствами и метатехнологими целостно-эволюционной среды (ЦЭ-среды).

ЦЭ-интеграция настраивает технологические и технические средства на выполнение того или иного когнитивного процесса, *ЦЭ-интеллектуализация* выполняет конкретный когнитивный процесс и передает результат-аттрактор в интегрированную базу знаний данной вложенной системы.

В результате при создании И-технологии как когнитивно-продуктивной метатехнологии интегрируются следующие средства:

- программные модули, реализующие процессы и процедуры деятельности, а также технологии, решающие задачи деятельности;
- средства мониторинга текущей ситуации (идентификация текущей ситуации, идентификация режима актуализации, идентификация базовых парадигм);
- средства реализации парадигм когнитивного ресурса;
- средства реализации парадигм продуктивного ресурса;
- средства формирования базовых, когнитивной и продуктивной технологий;
- средства синтеза структур-стратегий базовых технологий (формирование структур-стратегий, формирование предысторий знаний и действий, оценка структур-стратегий);
- средства оптимизации структур-стратегий базовых технологий в различных режимах актуализации;
- средства реализации интеллектуальных функций (обучение, адаптация, самоорганизация), интегрированные в базовые технологии.

Таким образом, при синтезе метатехнологий ЦЭ-интеллектуализации осуществляется системная интеграция не только информационных потоков, но и знаний, теоретико-аналитических моделей, методик, процедур, механизмов обучения, адаптации, самоорганизации и других интеллектуальных функций (И-функций) и средств.

Элементная база и когнитивный эволюционный компьютер

Концепция элементной базы аппаратной поддержки эволюционных вычислений была впервые разработана совместно со специалистами по микроэлектронике в 80-е годы [12, 14]:

- разработано семейство базовых опто- и микроэлектронных универсальных логических элементов с ориентацией на интегральную технологию тех лет (сверхбольшие и сверхскоростные оптические интегральные схемы);
- существенным отличием этих элементов является наличие в них управления другими элементами посредством функционирования и переменных связей;
- данное семейство включает подсемейство известных нейроподобных элементов (пороговых, формальных нейронов), дополняя их свойствами связи;
- перспективность базовых элементов подтверждена разработкой схемотехнической модели эволюционного предсказывающего спецпроцессора и результатами моделирования [12, 17, 18].

В работе [19] впервые была выдвинута *концепция когнитивного эволюционного компьютера* как вычислительной системы, реализующей концепцию ЦЭ-интеллектуализации (и, как следствие, когнитивно-продуктивные метатехнологии) на основе нейроподобных элементов, соответствующих концепции элементной базы.

Когнитивный эволюционный компьютер принципиально отличается от традиционных средств искусственного интеллекта:

- он является обучающейся системой, ориентированной на неполную информацию;
- реализует параллелизм эволюционных механизмов на всех уровнях функционирования, начиная с уровня элементов;
- не требует предварительного распараллеливания, программирования или иной специальной разработки алгоритма решения задачи;
- работает со знанием в виде структурированных моделей, являясь в целом гибридной интеллектуальной системой.

Аппаратная реализация когнитивного эволюционного компьютера — это реализация когнитивно-эволюционной метатехнологии, она возможна в виде:

- нейроплаты или приставки к персональному компьютеру;
- эволюционного компьютера, например, на основе оптических СБИС;
- элементов на основе современных нанотехнологий, реализующих концепцию элементной базы средствами, отличными от оптоэлектронных, т. е. на микрометровых и субмикрометровых — (нано) уровнях.

Эффективные приложения и перспективы

При соответствующем развитии парадигмы эволюционного моделирования, в основу которого положены результаты работ названных выше специалистов, эффективно решаются следующие задачи:

- синтез помехоустойчивых радиотехнических систем и эффективных устройств многоканальной обработки информации в реальном времени;
- восстановление, классификация и прогнозирование параметров нестационарных процессов в условиях информативной неопределенности;
- обработка информации при априори неизвестной динамике параметров изучаемых систем, позволяющая автоматизировать процессы восстановления и классификации информации в прикладных радиотехнических системах и в системах глобального мониторинга;
- автоматическое генерирование дискретно-непрерывных описаний объекта, создание гибкой организации банков данных, формирование адаптивных баз знаний;

- анализ и проектирование сложных информационных систем типа многоканального устройства интеллектуальной обработки сигналов, гибридных интеллектуальных информационно-вычислительных систем;
- управление развитием крупномасштабных (глобальных) систем;
- создание эффективных интеллектуальных метатехнологий для решения актуальных задач биофизики, медицины, экологии, экологического мониторинга, социальных и социоприродных систем.

Высокая эффективность когнитивно-эволюционных метатехнологий в различных приложениях создает обоснованную ***перспективу*** средств эволюционной информатики и, как установлено, достигается в результате:

- реализации целостности и адаптивности процесса приобретения знаний;
- системной интеграции методологий, методов, моделей, средств деятельности по схеме ЦЭ-представлений;
- формирования корпоративных баз данных и знаний на основе многоуровневой (по совокупности вложенных систем) интеграции знаний и метазнаний;
- мониторинга и адекватного учета динамики внутренних и внешних факторов деятельности, обеспечивающих эффективное преодоление проблемы информативной неопределенности;
- автоматизации и многоуровневой интеллектуализации моделирования, анализа, прогноза, принятия управленческих решений путем оснащения И-функциями по схеме системной ЦЭ-интеграции.

Таким образом, изложенные выше результаты анализа показывают, что эволюционные модели, технологии и средства наиболее *перспективны* в тех прикладных областях, в которых:

- изучаются сложные взаимосвязи большого числа регулируемых, контролируемых параметров и измеряемых величин;
- априори отсутствуют адекватные функциональные зависимости (модели) между ними;
- наблюдается суперпозиция процессов с различными временными темпами изменчивости;
- невозможно четко указать пределы необходимой информации и указать степень детализации требуемых моделей;
- отсутствует информация для определения этих зависимостей в целях оптимального управления сложными и дорогостоящими научными исследованиями.

Заключение

В сравнении с традиционными интеллектуальными технологиями *когнитивно-эволюционная метатехнология* является интеллектуальным ядром (а не интеллектуализированным фрагментом) моделируемой системы. Она не только решает информационно-расчетные задачи на основе формирования знаний в отдельных предметных областях, но и интегрирует когнитивные и продуктивные знания всех вложенных компонентов объекта интеллектуализации. Она интегрирует комплекс интеллектуальных функций технологии, а не только автоматизирует ее отдельные интеллектуальные функции.

Одновременно когнитивно-эволюционная метатехнология характеризуется глобальной устойчивостью в достижении целей, которая обеспечивается на основе ЦЭ-интеграции вложенных систем и использует имитацию эволюционных механизмов на всех уровнях организации живого [20]. И это особенно актуально в современных условиях глобализации и тотальной информатизации реальных систем, что с учетом выше изложенных факторов эффективности определяет современные перспективы эволюционного моделирования [5, 21—27].

Список литературы

1. **Красилов В. А.** Эволюция и биостратиграфия. М.: Наука, 1977. 250 с.
2. **Шмальгаузен И. И.** Кибернетические вопросы биологии. Новосибирск: Наука, 1968. 224 с.
3. **Иорданский Н. Н.** Эволюция жизни. М.: Академия, 2001. 425 с.
4. **Курейчик В. М.** Генетические алгоритмы. Состояние. Проблемы. Перспективы // Изв. РАН. ТиСУ. 1999. № 1. С. 144—160.
5. **Рутковская Д., Пилиньский М., Рутковский Л.** Нейронные сети, генетические алгоритмы и нечеткие системы: Пер. с польск. И. Д. Рудинского. М.: Горячая линия — Телеком, 2006. 452 с.
6. **Фогель Л., Оуэнс А., Уолш М.** Искусственный интеллект и эволюционное моделирование. М.: Мир, 1969. 230 с.
7. **Ивахненко А. Г.** Самообучающиеся системы распознавания и автоматического управления. Киев: Техніка, 1969. 392 с.
8. **Растригин Л. А.** Статистические методы поиска. М.: Наука, 1968. 376 с.
9. **Неймарк Ю. И., Фуфаев Н. А.** Динамика неголономных систем. М.: Наука, 1967.
10. **Цетлин М. Л.** Исследования по теории автоматов и моделирование биологических систем. М.: Наука, 1969. 316 с.
11. **Цыпкин Я. З.** Основы теории обучающихся систем. М.: Наука, 1970. 251 с.
12. **Гуляев Ю. В., Крапивин В. Ф., Букатова И. Л.** На пути к эволюционной информатике // Вестник АН СССР, 1987. № 11. С. 53—61.
13. **Вопросы** эволюционного моделирования (сб. ст.). М.: ИРЭ АН СССР, 1982. 68 с.
14. **Букатова И. Л., Елинсон М. И., Шаров А. М.** Обучающиеся системы. М.: Знание, 1979. 50 с.
15. **Новые** информационные технологии в науке, образовании и бизнесе // Труды XXIII Междунар. конф. "IT + SE'96", Ялта-Гурзуф, 1996.
16. **First** International Conference on Evolutionary Computation and Its Applications. EvCA'96. Program and Proceedings. June 24—27, 1996. Moscow, Russia.
17. **Букатова И. Л.** Эволюционное моделирование и его приложения. М.: Наука, 1979. 232 с.
18. **Букатова И. Л., Михасев Ю. И., Шаров А. М.** Эвоинформатика: теория и практика эволюционного моделирования. М.: Наука, 1990. 212 с.
19. **Bukatova I. L. and Guliaev Yu. V.** From genetic algoritms to evolutionary computer. In Forrest, editor // Proc. of the Fifth International Conference on Genetic Algorithms, pages 614—617, Urbana-Champaign, IL, 17—21. July 1993. Morgan Kaufmann, San Mateo, CA.
20. **Гринченко С. Н.** Метаэволюция (систем неживой, живой и социально-технологической природы). М.: ИПИ РАН, 2007. 456 с.
21. **Нечаев В. В., Дарьин А. В.** Информационное общество и человек. Вып. 1. Моделирование процессов эволюции: ресурсный подход. М.: Междунар. изд-во "Информациология", 1999. 68 с.
22. **Нечаев В. В.** Морфологическая БАС в задачах эволюционного моделирования // Методы прикладной математики и системного программирования. Межвуз. сб. трудов. М.: МИРЭА, 1985. С. 188—193.
23. **Букатова И. Л., Макрусев В. В.** Теория целостно-эволюционной интеллектуализации социальных систем. М.: МИГКУ, 2004. 125 с.
24. **Букатова И. Л., Рощупкин О. М.** Интеллектуализация банковской деятельности: целостно-эволюционный подход. М.: Альянс, 2005. 242 с.
25. **Верлань А. Ф., Дмитриенко В. Д., Корсунов Н. И., Шорох В. А.** Эволюционные методы компьютерного моделирования. Киев: Наукова Думка, 1992. 256 с.
26. **Редько В. Г.** Эволюционная кибернетика. М.: Наука, 2001. 156 с.
27. **Букатова И. Л.** Когнитивно-эволюционные модели инновационного развития глобальных систем // Социология инноватики: социальные и культурные условия модернизации. Доклады и выступления 4-й междунар. конф. по социологии инноватики. 24—25 ноября 2011 г. М.: ФГБОУ ВПО РГАИС, 2012. С. 133—136.

УДК 004.421.2:519.8

Ю. А. Зак, д-р техн. наук, науч. консультант,
фирма "Smartroute", Германия, Аахен,
e-mail: yuriy_zack@hotmail.com

Приближенные методы решения Job-Shop-Problem — построение расписаний выполнения *n* заданий на *m* машинах

Предлагаются эффективные приближенные методы решения Job-Shop-Problem. На основе установленных свойств допустимых и оптимальных расписаний на первых шагах работы алгоритма устанавливается факт совместности исходной системы ограничений, вычисляются нижние границы критерия эффективности в оптимальном решении. Алгоритм приближенного решения задачи строится на основе построения на каждом шаге решения допустимого расписания выполнения работ на наиболее напряженной машине и корректировки наиболее ранних и допустимых наиболее поздних сроков начала выполнения каждой операции на других машинах, сроки выполнения на которых еще не назначены.

Ключевые слова: *Job-Shop-Problem, оценки оптимального решения, приближенные решения, граничные сроки начала выполнения операций, свойства допустимых и оптимальных расписаний*

Введение

Известная в литературе задача построения расписаний выполнения N заданий, каждое из которых состоит из множества J_i, $i = 1, ..., N$, операций, выполняемых в заданной для каждого из заданий последовательности друг за другом на m машинах, а также определения последовательностей технологических операций на каждой машине, заключается в следующем [1—5].

1. Каждое задание состоит из некоторого фиксированного числа технологических операций, выполняемых в строго заданной для каждого задания последовательности на некотором подмножестве машин.

2. Каждая технологическая операция любого из заданий может быть выполнена только на одной предназначенной для ее выполнения машине.

3. Заданы требуемые времена выполнения каждой технологической операции всех заданий, которые не допускают прерываний в процессе их выполнения.

4. Заданы директивные сроки наиболее раннего срока начала и наиболее позднего срока завершения выполнения каждого из заданий и времен возможного использования машин.

Необходимо построить оптимальные последовательности выполнения различных операций всех заданий на каждой рабочей станции, определить времена начала и завершения их выполнения, обеспечивающие выполнение всех технологических и ресурсных ограничений и минимизацию времени завершения всего комплекса работ (длины расписания), т. е. критерия оптимальности вида

$$F_{\max} = \max_{1 \leqslant i \leqslant n} T_i \to \min, \qquad (1)$$

где T_i — фактические сроки завершения выполнения i-го задания.

Теоретически существует до $(n!)^m$ различных допустимых расписаний, и, как показано [1, 3, 4], рассматриваемая задача относится к классу *NP*-сложных проблем. В литературе (см., например, [1—8]) предлагаются различные точные и приближенные методы решения рассматриваемой проблемы при отсутствии ограничений на сроки завершения выполнения заданий: решение задач математического программирования с непрерывными и целочисленными переменными; метод ветвей границ; метод динамического программирования и последовательного анализа вариантов и др. Эти методы позволяют получить точное решение лишь для задач относительно небольшой размерности и требуют весьма существенных объемов вычислений. Наибольшее применение для практических приложений находят приближенные методы решения этой задачи [1, 2, 4, 8]. Эти методы основаны на использовании различного рода эвристик, алгоритмов глобального случайного поиска, генетических алгоритмов и эволюционных стратегий. Все эти приближенные методы имеют целый ряд существенных недостатков, основные среди которых следующие:

— отсутствуют оценки точности полученного приближенного решения;

— они не приемлемы в условиях наличия ограничений на директивные сроки завершения выполнения заданий и время работы машин и не позволяют установить факт несовместности заданной системы ограничений уже на ранних этапах решения.

В работах автора [6, 7] на основе рассчитанных возможных наиболее ранних и допустимых наиболее поздних сроков начала выполнения каждой операции установлены свойства допустимых и оптимальных расписаний, на основе которых совместно Dr. rer. nat. S. Rotin (С. В. Ротиным) [7] разработаны

алгоритмы точного решения сформулированной задачи, которые реализованы Dr. rer. nat. S. Rotin в виде программы на C^{++} [11]. Эта программа позволяет решать тестовые и практические задачи не очень большой размерности.

В данной работе предлагаются эффективные приближенные методы решения задачи. На основе установленных свойств допустимых и оптимальных расписаний на первых шагах работы алгоритма устанавливается факт совместности исходной системы ограничений, вычисляются нижние границы критерия эффективности в оптимальном решении, выбирается машина, являющаяся "узким местом" в выполнении данного комплекса работ. На основе построения на каждом шаге решения допустимого расписания выполнения работ на наиболее напряженной машине и корректировки наиболее ранних и допустимых наиболее поздних сроков начала выполнения каждой операции на других машинах, сроки выполнения на которых еще не назначены, строится алгоритм приближенного решения задачи. В процессе работы алгоритма многократно решается известная в литературе задача построения оптимального расписания на одной машине в условиях наличия ограничений на сроки начала и завершения выполнения операций, эффективные методы решения которой изложены в работах автора [8—10].

1. Математическая модель задачи

На m рабочих станциях (машинах) $k = 1, ..., m$, должно быть выполнено n заданий. Каждое из заданий состоит из множества $\overline{J}_i$ последовательно следующих друг за другом операций $O_{ijk}, j = 1, ..., \overline{J}_i$. В дальнейшем каждая операция O_{ijk} однозначно определяется мультииндексом $\alpha = (i, j, k)$.

Введем следующие обозначения:

$i = 1, ..., n$ — индексы подлежащих выполнению заданий; $j = 1, ..., m$ — индексы операций; $k = 1, ..., m$ — индексы используемых рабочих станций (машин); $A = \{\alpha = (i, j, k): j \in \overline{J}_i, i = 1, ..., n, k = 1, ..., m\}$ — множество мультииндексов операций всех заданий; $k = \mu(i, j)$ — индекс рабочей станции, на которой должна выполняться операция O_{ijk}; $t(i, j, k)$ — время выполнения операции O_{ijk}; $x(i, j, k)$ и $\sigma(i, j, k) = x(i, j, k) + t(i, j, k)$ — фактическое (установленное расписанием выполнения работ) время начала и завершения выполнения операции O_{ijk}; $\theta(i, j, k)$ и $\tau(i, j, k)$ — соответственно допустимые наиболее раннее время начала и наиболее позднее время завершения выполнения операции O_{ijk}; $T_i = \sigma(i, \overline{J}_i)$ — фактическое время завершения выполнения задания i, $i = 1, ..., n$; b_i и B_i — соответственно допустимые наиболее ранние сроки начала и наиболее поздние сроки завершения выполнения заданий $i = 1, ..., n$; h_k и H_k — соответственно допустимые наиболее ранние сроки начала и наиболее поздние сроки окончания работы k-й машины, $k = 1, ..., m$; $F_{\max} = \max\limits_{1 \leqslant i \leqslant n} T_i$ — фактическое время завершения выполнения всех заданий (длина расписания); F_k^s — время завершения выполнения всех операций на k-й машине на s-м шаге итеративного процесса; F^s — выбранная длина расписания (значение критерия оптимальности) на s-м шаге итеративного процесса.

Требуется найти значения неизвестных переменных $x(\alpha)$, $\alpha \in A$, которые обеспечили бы значение глобального минимума критерия оптимальности $F_{\max}$.

Если условиями задачи не заданы значения величин b_i или B_i, то полагаем их равными $b_i = 0$, $B_i^s = F^s$. Если не заданы значения h_k и H_k, то полагаем их равными $h_k = 0$, $H_k = \infty$. Начальные значения возможных наиболее ранних и допустимых самых поздних времен начала выполнения всех операций i-го задания могут быть определены из следующих соотношений:

$$\theta(i, 1, k) = \max(b_i, h_k), \ \theta(i, j + 1, q) = \max[\theta(i, j, k) + t(i, j, k) + 1, h_q], \ j = 1, ..., J_i - 1; \quad (2)$$

$$\tau(i, J_i, k) = \min(F_i^s; B_i; H_k) - t(i, J_i, k) + 1,$$
$$\tau(i, j - 1, q) = \min[\tau(i, j, k); H_q] - t(i, j - 1, k) + 1. \quad (3)$$

Фактическое время начала выполнения операции O_{ijk} должно быть выбрано в пределах следующих ограничений:

$$\theta(i, j, k) \leqslant x(i, j, k) \leqslant \tau(i, j, k). \quad (4)$$

В дальнейших разделах работы устанавливаются свойства допустимых расписаний, на основе которых становится возможным исключить из рассмотрения подмножество расписаний, заведомо не содержащих допустимых планов, а также установить некоторые частичные порядки выполнения операций различных заданий на машинах.

2. Свойства допустимых и оптимальных расписаний

Утверждение 1. Если в результате осуществления преобразований для какой-либо операции O_{ijk}, выполняемой на k-й машине, справедливо неравенство

$$\theta(i, j, k) > \tau(i, j, k), \quad (5)$$

то не существует допустимых расписаний выполнения всех операций на машине j за время, меньшее или равное F_k^s.

Утверждение 2. Если для двух операций O_{ljk} и O_{prk}, выполняемых на одной и той же k-й машине, справедливо одно из системы неравенств

$$\begin{aligned} &\theta(l, j, k) + t(l, j, k) + 1 > \tau(p, r, k), \\ &\theta(p, r, k) + t(p, r, k) + 1 > \tau(l, j, k), \end{aligned} \quad (6)$$

то не существует допустимых расписаний выполнения всех операций на машине k за время, меньшее или равное F_k^s.

Утверждение 3. Если для двух операций O_{ljk} и O_{prk}, выполняемых на одной и той же k-й машине, справедливо хотя бы одно из пары условий

$$\tau(p, r, k) < \theta(l, j, k),$$
$$\theta(p, r, k) + t(p, r, k) + 1 \leqslant \tau(l, j, k), \qquad (7)$$

то в допустимых расписаниях, удовлетворяющих системе ограничений (2)—(4), операция O_{prk} должна выполняться раньше операции O_{ljk}, т. е.

$$x(p, r, k) + t(p, r, k) + 1 \leqslant x(l, j, k).$$

Следствия утверждения 3.

$$\overline{\theta}(l, j, k) =$$
$$= \max\{(\theta(p, r, k) + t(p, r, k) + 1), \theta(l, j, k)\}, \qquad (8)$$

$$\overline{\tau}(p, r, k) = \min\{[\tau(l, j, k) - t(p, r, k)], \tau(p, r, k)\}. \qquad (9)$$

Утверждение 4. Если для нескольких операций $O_{i_p j_p k}$, $p = 1, ..., P$, выполняемых на одной и той же k-й машине, установлено, что $x(i_p, r_p, k) < x(l, j, k)$, $p = 1, ..., P$, то справедливы соотношения

$$\theta(l, j, k) \geqslant \min\left\{ \max_{1 \leqslant p \leqslant P} [\theta(i_p, r_p, k) + t(i_p, r_p, k)]; \right.$$
$$\left. \min_{1 \leqslant p \leqslant P} \theta(i_p, r_p, k) + \sum_{p=1}^{P} t(i_p, r_p, k) \right\} + 1. \qquad (10)$$

Доказательство утверждения следует из условий (6)—(10), а также из того, что время начала выполнения операции O_{ljk} не может быть ранее времени завершения всех операций $O_{i_p j_p k}$, $p = 1, ..., P$, предшествующих ей, плюс наиболее ранний срок начала выполнения этого подмножества операций.

Утверждение 5. Время выполнения расписания не может быть меньше величины

$$F = \max\left\{ \max_{1 \leqslant i \leqslant n} \left[\max(b_i, h_k) + \sum_{j \in \overline{J}_i} t(i, j, k) \right]; \right.$$
$$\left. \max_{1 \leqslant k \leqslant m} \left[h_k + \sum_{k=1}^{R_k} t(i, j, k) \right] \right\}, \qquad (11)$$

где R_k — суммарное число операций всех заданий, выполняемых на k-й машине.

Формула (11) отражает тот факт, что длительность выполнения расписания не может быть меньше, чем время завершения выполнения всех операций каждого задания или времени работы какой-либо машины.

3. Алгоритмы определения допустимых интервалов времен начала выполнения операций

При установленном значении максимальной длины расписания F^s в условиях заданной системы ограничений на сроки выполнения заданий и допустимые времена работы машин для каждого из индексов k рассмотрим следующий алгоритм вычислений.

Алгоритм 1 (s, k). Значения счетчиков z_{A1} и z_{B1} положим равными нулю.

Шаг 1. Для каждой пары технологических операций O_{ljk} и O_{prk}, выполняемых на k-й машине, проверяется выполнение условий (5), (6). Если выполняется хотя одно из этих неравенств, то устанавливаем значения $z_{B1} := 2$,

$z_{A1} := 2$ и $\Delta^s := \min\{\theta(l, j, k) - \tau(l, j, k)\}$, либо

$$\Delta^s := \min\{\theta(l, j, k) + t(l, j, k) - \tau(p, r, k), \theta(p, r, k) + t(p, r, k) - \tau(l, j, k)\}, \qquad (12)$$

и алгоритм завершает работу, выполнив необходимые вычисления, предусмотренные в шаге 5. В противном случае переходим к шагу 2.

Шаг 2. Рассматриваем все пары операций, выполняемых на k-й машине. Если для какой-то пары операций O_{ljk} и O_{prk} выполняются условия (7), то полагаем $z_{A1} := 1$, $z_{B1} := 1$, корректируем соответствующие граничные значения времен начала выполнения операций, выполнив преобразования (8), (9). Переходим к шагу 3.

Шаг 3. Если для нескольких операций $O_{i_p j_p k}$, $p = 1, ..., P$, выполняемых на k-й машине, установлено, что они должны выполняться после операции O_{ljk}, т. е. $x(i_p, r_p, k) < x(l, j, k)$, то полагаем $z_{A1} := 1$, $z_{B1} := 1$, и выполняем пересчет граничных значений времен начала выполнения этих операций по формулам (10). Переходим к шагу 4.

Шаг 4. Если $z_{A1} := 0$, то алгоритм выполнения преобразований для k-й машины завершает свою работу. Если $z_{A1} := 1$, то полагаем $z_{A1} := 0$, и переходим к выполнению шага 1.

Шаг 5. Положив $F^{s+1} = F^s + \Delta^s$, $s := (s + 1)$, выполняем пересчет значений $\theta(i, j, k)$ и $\tau(i, j, k)$ по формулам (3), (4), после чего, положив $k = 1$, снова выполняем алгоритм **1(s,k)**.

Если алгоритм **1(k)** завершил свою работу со значением признака $z_{B1} := 1$, то выполняем алгоритм **2(s)**.

Алгоритм 2(s). Значение счетчика z_{B1} положим равным нулю.

Выполняем для всех заданий $i = 1, ..., m$ пересчет граничных значений времен начала выполнения операций по формулам

$$\overline{\theta}(j, j + r, k) = \max\{(\theta(i, j + r - 1, k) + t(i, j + r - 1, k)), \theta(i, j + r, k)\}, r = 1, ..., (J_i - j); \qquad (13)$$

$$\overline{\tau}(j, j - p, k) = \max\{(\tau(i, j - p + 1, k) - t(i, j - p + 1, k)), \theta(i, j - p, k)\};$$
$$p = (j - 1), (j - 2), ..., 1. \qquad (14)$$

Для каждой операции O_{ijk} проверяем выполнение неравенства (6). Если справедливо неравенство (6), то полагаем $z_{B1} = 2$ и вычисляем значение

$$\Delta^s = \theta(i, j, k) - \tau(i, j, k), \qquad (15)$$

и завершаем работу алгоритма. Если неравенство (6) не выполняется, то выполняем преобразования (13), (14) для операций всех заданий.

Если алгоритм **2(s)** завершил работу со значением признака $z_{B1} = 0$, то полагаем $k := (k + 1)$. Если

$k \leqslant m$, то с новым значением k вновь выполняем алгоритм **1(s, k)**. Если $z_{B1} = 2$, то положив $F^{s+1} = F^s + \Delta^s$, $s = (s + 1)$, выполняем пересчет значений $\theta(i, j, k)$ и $\tau(i, j, k)$ по формулам (2), (3), после чего, положив $k = 1$, снова выполняем алгоритм **1(s, k)**.

4. Построение допустимых расписаний выполнения работ на одной машине

Сформулированная выше Job-Shop-Problem предусматривает решение в качестве составной решение задачи построения допустимых расписаний выполнения работ на одной машине, которая в данном случае должна рассматриваться в следующей формулировке.

Пусть $\overline{I}_k$ — подмножество операций, которые должны быть выполнены на k-й машине. Заданы времена выполнения операций $t(i, j, k)$, $(i, j, k) \in \overline{I}_k$, ограничения на начальные сроки их выполнения $x(i, j, k) \in [\theta(i, j, k), \tau(i, j, k)]$, а также на времена завершения их выполнения операций $\sigma(i, j, k) \leqslant \leqslant \tau(i, j, k) + t(i, j, k)$. Ни одна из операций не может прерываться в процессе ее выполнения и одновременно на машине не может выполняться более одной операции. Необходимо построить расписания выполнения работ, удовлетворяющие всем сформулированным требованиям.

В работах автора [9, 10] подробно описаны различные алгоритмы решения этой задачи. Для приближенного решения Job-Shop-Problem используется алгоритм, в процессе которого на первом шаге полиномиальным алгоритмом строится расписание выполнения работ на одной машине, допускающее прерывание работ в процессе их выполнения (обозначим такое расписание Ψ_1) [9], т. е. построение некоторой последовательности выполнения операций $\overline{W}_k(\Psi_1)$, и определяются времена начала и завершения выполнения каждой операции этой последовательности. В дальнейшем это расписание Ψ_1 преобразуется в расписание работ, т. е. в допустимое расписание Ψ_2, в котором разрывы во времени выполнения операций отсутствуют. Второй этап решения задачи представляет собой ветвящийся процесс последовательного преобразования расписания Ψ_1 в различные расписания $\Psi_1^1(\overline{t})$ и $\Psi_1^2(\overline{t})$, каждое из которых некоторым образом устраняет процесс прерывания выполнения какой-то операции до тех пор, пока не будет построено допустимое расписание Ψ_2 без прерываний времен выполнения операций.

Пусть построено допустимое расписание $\Psi_2(k)$ выполнения операций на k-й машине. Обозначим: $V\{\Psi_2(k)\} = \{(\nu_1, k), ..., (\nu_p, k), ..., (\nu_{N_k}, k)\}$ — последовательность выполнения операций в $\Psi_2(k)$; $x(\nu_p, k)$, $\sigma(\nu_p, k)$, $d(u_p, k)$ — соответственно фактические времена начала и завершения операций, а также граничные сроки завершения выполнения операций,

$$\eta(\nu_p, k) = \tau(\nu_p, k) - \theta(\nu_p, k); \qquad (16)$$

$\lambda(\nu_p, k)$ — суммарное число временных интервалов $t \in [\sigma(\nu_p, k); x(\nu_{p+1}, k)]$, в течение которых на k-й машине не выполняется никаких операций.

Тогда допустимые наиболее ранние и поздние сроки начала выполнения операций на k-й машине могут быть рассчитаны по следующим формулам:

$$\theta(\nu_p, k) = x(\nu_p, k),\ p = 1, ..., N_k; \qquad (17)$$

$$\tau(\nu_{N_k}, k) = \theta(\nu_{N_k}, k) + [d(\nu_{N_k}, k) - \sigma(\nu_{N_k}, k)]; \qquad (18)$$

$$\tau(\nu_{N_k-1}, k) = \theta(\nu_{N_k-1}, k) + \min\{[d(\nu_{N_k-1}, k) - \sigma(\nu_{N_k-1}, k) + \lambda(\nu_{N_k-1}, k)]; \eta(\nu_{N_k}, k)\}; \qquad (19)$$

$$\tau(\nu_{p-1}, k) = \theta(\nu_{p-1}, k) + \min\{[d(\nu_{p-1}, k) - \sigma(\nu_{p-1}, k) + \lambda(\nu_{p-1}, k)]; \eta(\nu_p, k)\}$$
$$\forall p = (N_k - 2), ..., 2. \qquad (20)$$

5. Алгоритмы решения Job-Shop-Problem

Алгоритм решения задачи представляет собой ветвящийся процесс и основан на расчете наиболее ранних $\theta(i, j, k)$ и допустимых наиболее поздних времен начала выполнения всех операций каждого из заданий $\tau(i, j, k)$. В процессе анализа и построения допустимых последовательностей выполнения операций на каждой машине значения $\theta(i, j, k)$ корректируются в сторону увеличения, а $\tau(i, j, k)$ — в сторону уменьшения. На основе установленных выше свойств допустимых решений и нижней границы длины оптимального расписания на каждом этапе решения проверяется перспективность дальнейших продолжений и отсев не перспективных расписаний. Как только будут построены допустимые последовательности выполнения всех операций на каждой машине и вычислены соответствующие им граничные значения $\theta(i, j, k)$ и $\tau(i, j, k)$ для расписания установленной длины, процесс построения расписания останавливается, и по простым формулам вычисляются времена начала и завершения выполнения каждой операции всех заданий, т. е. значения $x(i, j, k)$ и $\sigma(i, j, k)$. В процессе таких преобразований уже на первых шагах устанавливается факт несовместности системы ограничений и определение номеров заданий и машин, временные ресурсы которых должны быть расширены.

Все множество заданий $\tilde{I} = \{i = 1, ..., n\}$ разделим на два подмножества: $\tilde{I}^1$ — подмножество заданий, для которых заданы ограничения на сроки завершения заданий, и $\tilde{I}^2$ — подмножество заданий, для которых такие ограничения отсутствуют.

Обозначим $\tilde{K}_1^s$ и $\tilde{K}_2^s$ — соответственно подмножества машин, для которых на s-м шаге алгоритма установлена и не установлена строгая последовательность выполнения всех операций. $\tilde{K}_1^s \cap \tilde{K}_2^s = \varnothing$,

$\tilde{K}_1^s \cup \tilde{K}_2^s = \tilde{K} = \{1, ..., m\}$. Если $k \in \tilde{K}_1^s$, то построена *такая* последовательность выполнения операций

$$\tilde{V}_k^s = \{(\nu_1^s, k), (\nu_2^s, k), ..., (\nu_{p-1}^s, k), (\nu_p^s, k), ..., (\nu_{N_k}^s, k)\}, \quad (21)$$

в которой для каждой пары рядом стоящих операций (ν_{p-1}^s, k) и (ν_p^s, k), $p = 2, 3, ..., N_k$, выполняются соотношения

$$\theta(\nu_p^s, k) \leqslant \theta(\nu_{p-1}^s, k) + t(\nu_{p-1}^s, k) \leqslant \tau(\nu_p^s, k), \theta(\nu_p^s, k) + t(\nu_p^s, k) < \tau(\nu_{p-1}^s, k). \quad (22)$$

Алгоритм решения сформулированной задачи Job-Shop-Poblem в условиях наличия граничных значений b_i и (или) B_i заключается в следующем.

Шаг 1. Определяем минимальные времена, необходимые для выполнения каждого из заданий и работы машин:

$$Y_i = b_i + \sum_{j \in \bar{J}_i} t(i, j, k), \; i = 1, ..., n;$$

$$Z_k = h_k + \sum_{i=1}^{n} \sum_{j \in \bar{J}_i} t(i, j, k), \; k = 1, ..., m. \quad (23)$$

Если ограничения на значения b_i и (или) B_i не заданы, то полагаем соответственно $b_i = 0$, $B_i = \pi_i Y_i$, где $\pi_i > 1$ и, например, π_i равно 1.5, 2.0, 2.5. Если ограничения на значения h_k и (или) H_k не заданы, то полагаем соответственно $h_k = 0$, $H_k = \pi_k H_k$, где $\pi_k > 1$ и, например, π_k равно 2.5, 3.0.

Шаг 2. Рассчитываем значение $F^0 = \max\{\max_{1 \leqslant i \leqslant n} Y_i; \max_{1 \leqslant k \leqslant m} Z_k\}$.

Шаг 3. По формулам (2), (3) рассчитываем значения $\theta(i, j, k)$ и $\tau(i, j, k)$. Если хотя бы для одной из операций подмножества заданий $\tilde{I}^1$ не выполняется неравенство (5), то система ограничений задачи является несовместной и алгоритм завершает свою работу. Если одно из неравенств (5) не выполняется для некоторого задания $i_1 \in \tilde{I}^2$, то соответствующим образом увеличиваем значение π_{i_1} и переходим к выполнению шага 1. Если выполняются неравенства (5) для всех (i, j, k), то переходим к выполнению шага 4.

Шаг 4. Полагаем $s = 0$, вычисляем значение F^s, и для каждого значения $k = 1, ..., m$ выполняем последовательно алгоритмы **1(s, k)** и, если он завершил свою работу со значением $z_{B1} := 1$, выполняем алгоритм **2(s, k)**. Если для некоторого значения k_1 алгоритм **1(s, k_1)** завершил работу на шаге 1 со значением Δ^s, то в случае $\Delta^s = \theta(l, j, k_1) - \tau(l, j, k_1)$ и при этом $l \in \tilde{I}^1$, то система ограничений задачи является несовместной и алгоритм завершает работу. Если $l \in \tilde{I}^2$, то на шаге 5 алгоритма корректируем значение $F^{s+1} = F^s + \Delta^s$, $s := (s + 1)$, и вновь переходим к выполнению шага 1.

Если для всех значений $k = 1, ..., m$ последовательное выполнение алгоритмов **1(s, k)** и **2(s, k)** успешно завершилось и при этом выполняются неравенства (5), то переходим к выполнению шага 5, определив значения $\theta(l, j, k)$ и $\tau(l, j, k)$, которые вычислены на предыдущих шагах алгоритма.

Шаг 5. Выбираем машину μ, для которой длительность выполнения всех назначенных на ней операций максимальна:

$$Z_\mu = \max_{1 \leqslant k \leqslant m} \{\min_{1 \leqslant i \leqslant n} \min_{j \in \bar{J}_i} \theta(i, j, k) + \sum_{i=1}^{n} \sum_{j \in \bar{J}_i} t(i, j, k) | k \in \tilde{K}_1^s\}. \quad (24)$$

Если $\tilde{K}_1^s = \varnothing$, то переходим к шагу 8. В противном случае для данной μ-й машины алгоритмом **3k** строим последовательность выполнения всех операций $\tilde{W}_\mu(\Psi_1)$, допускающую разрывы во времени выполнения отдельных операций. Если в построенной последовательности для всех операций выполняются условия $\sigma(i, j, \mu) \leqslant d(i, j, \mu)$, то переходим к шагу 6. Если для некоторой операции (i, j, μ) выполняется неравенство $\sigma(i, j, \mu) > d(i, j, \mu)$, то в случае, если $i \in \tilde{I}^2$, увеличиваем граничное значение минимального значения длины расписания и времени завершения выполнения всех операций этого задания, т. е. положив

$$F^{s+1} = B_i^{s+1} \geqslant B_i^s + [d(i, j, \mu) - \sigma(i, j, \mu)], \; s := (s + 1), \quad (25)$$

переходим к шагу 1 алгоритма.

Если неравенство $\sigma(i, j, \mu) > d(i, j, \mu)$ выполняется для некоторой операции, индекс которой $i \in \tilde{I}^1$, то система ограничений задачи является несовместной и с соответствующим сообщением алгоритм завершает свою работу.

Шаг 6. Для данной μ-й машины алгоритмом построения последовательности выполнения работ на одной машине [7] (алгоритм **4μ**) преобразовываем построенную последовательность $\tilde{W}_\mu(\Psi_1)$ в расписание $\Psi_{2\mu}$, не допускающее разрывов во времени выполнения операций. Если алгоритмом **4μ** построена допустимая последовательность выполнения операций на данной машине без прерывания выполнения операций, то переходим к шагу 7. Если в процессе работы алгоритма **4μ** будет установлено, что $\sigma(i, j, \mu) > d(i, j, \mu)$, $i \in \tilde{I}^1$, то система ограничений задачи является несовместной, и с соответствующим сообщением алгоритм завершает свою работу. Если $\sigma(i, j, \mu) > d(i, j, \mu)$ и $i \in \tilde{I}^2$, то выполнив вычисления (23), переходим к шагу 1 алгоритма.

Шаг 7. Выполняем для машины $k = \mu$ пересчет значений $\theta(i, j, \mu)$ и $\tau(i, j, \mu)$ по формулам (17)—(20). Корректируем значения этих показателей для других машин по формулам (2), (3). Полагаем $\tilde{K}_1^s = \tilde{K}_1^s \cup \mu$, $\tilde{K}_2^s = \tilde{K}_2^s / \mu$. Если $\tilde{K}_1^s = \varnothing$, то переходим к шагу 8.

Шаг 8. Определены строгие порядки выполнения всех операций на каждой машине и вычислены для всех операций каждого из заданий значения $\theta^s(i, j, k)$ и $\tau^s(i, j, k)$, удовлетворяющие всем ограничениям задачи. Для получения значений $x(i, j, k)$ и $\sigma(i, j, k)$ каждой операции в последовательности $i = 1, ..., n$ полагаем

$$x(i, j, k) = \theta^s(i, j, k),\ \sigma(i, j, k) = x(i, j, k) + t(i, j, k),\quad j \in \widetilde{J}_i. \qquad (26)$$

После выполнения вычислений (25) для каждого задания алгоритмом **1(s, k)** на каждой машине (при необходимости) корректируем значения $\theta^s(r, j, k)$ и $\tau^s(r, j, k)$ для других заданий $r = i + 1, ..., n$ и выполняем вычисления (26) для очередного r-го задания.

Таблица 1

№ заданий	Машины, выполняющие операции заданий			Времена выполнения операций		
	1	2	3	1	2	3
1	1	2	3	4	6	8
2	2	3	1	2	10	5
3	2	1	3	7	3	6
4	1	3	2	8	4	9
5	2	1	3	6	5	8

Таблица 2

Допустимые времена работы	Номера машин		
	1	2	3
Начало	0	3	0
Завершение	40	45	52

Таблица 3

Допустимые времена работы	Номера заданий				
	1	2	3	4	5
Начало	5	0	4	2	0
Завершение	30	38	46	36	47

Таблица 4

№ заданий	Индексы операций	Допустимые самые ранние и наиболее поздние времена начала выполнения операций							
		В соответствии с ограничениями задачи		После определения допустимых времен выполнения на 3-й машине		После определения допустимых времен выполнения на 2-й машине		После определения допустимых времен выполнения на 1-й машине	
		θ	τ	θ	τ	θ	τ	θ	τ
1	(1,1,1)	5	12	5	12	5	11	10	11
	(1,2,2)	9	16	9	16	12	15	12	12
	(1,3,3)	15	22	15	22	19	22	19	19
2	(2,1,2)	3	21	3	6	3	6	3	3
	(2,2,3)	5	23	5	8	5	8	5	5
	(2,3,1)	15	33	15	33	15	33	17	29
3	(3,1,2)	4	30	4	23	5	8	5	8
	(3,2,1)	11	32	11	30	12	30	14	26
	(3,3,3)	14	40	27	33	27	33	27	33
4	(4,1,1)	2	15	2	9	2	9	2	3
	(4,2,3)	10	23	15	17	15	17	15	17
	(4,3,2)	14	27	19	27	24	27	24	27
5	(5,1,2)	3	28	3	28	18	21	18	21
	(5,2,1)	9	34	9	34	24	34	24	34
	(5,3,3)	14	39	33	39	33	39	33	39

6. Иллюстративный пример

На трех машинах должно быть выполнено пять заданий. Каждое из заданий состоит из трех операций. Последовательности машин, на которых выполняются операции каждого из заданий, и времена выполнения этих операций приведены в табл. 1.

Таблица 5

№ заданий	Индексы операций	Времена начала и завершения выполнения операций	
		$x(i, j, k)$	$\sigma(i, j, k)$
1	(1,1,1)	10	13
	(1,2,2)	12	17
	(1,3,3)	19	26
2	(2,1,2)	3	5
	(2,2,3)	6	15
	(2,3,1)	18	22
3	(3,1,2)	5	12
	(3,2,1)	15	17
	(3,3,3)	27	32
4	(4,1,1)	2	9
	(4,2,3)	15	18
	(4,3,2)	24	32
5	(5,1,2)	18	23
	(5,2,1)	24	32
	(5,3,3)	34	41

Таблица 6

№ машин	Индексы операций	Времена начала и завершения выполнения операций	
		$x(i, j, k)$	$\sigma(i, j, k)$
1	(4,1,1)	2	9
	(1,1,1)	10	13
	(3,2,1)	15	17
	(2,3,1)	18	22
	(5,2,1)	24	32
2	(2,1,2)	3	4
	(3,1,2)	5	11
	(1,2,2)	12	17
	(5,1,2)	18	23
	(4,3,2)	24	32
3	(2,2,3)	2	14
	(4,2,3)	15	18
	(1,3,3)	19	26
	(3,3,3)	27	32
	(5,3,3)	33	41

В табл. 2 приведены допустимые времена работы машин, а в табл. 3 — граничные сроки начала и завершения каждого из заданий.

В табл. 4 приведены рассчитанные в процессе работы алгоритма значения наиболее ранних и допустимых наиболее поздних сроков выполнения всех операций каждого из заданий на всех машинах.

Результаты решения примера сведены в табл. 5 и 6. В табл. 5 приведены времена начала $x(i, j, k)$ и завершения выполнения всех операций каждого из заданий $\sigma(i, j, k)$, а в табл. 6 — последовательности выполнения операций, а также времена начала и завершения их выполнения на каждой машине.

Как показали результаты вычислительных экспериментов, описанными выше алгоритмами с достаточно высокой точностью (3...8 % от оптимального значения) и сравнительно небольшими объемами вычислений можно решать практические задачи календарного планирования производства размерностью $n \leqslant 35$, $m \leqslant 12$.

Список литературы

1. **Танаев В. С., Шкурба В. В.** Введение в теорию расписаний. М.: Физматгиз. Наука, 1975. 256 с.
2. **Domschke W., Scholl A., Voß S.** Produktionsplanung. Ablauforganisatorische Aspekte. Berlin, Heidelberg: Springer Verlag, 2005. 456 s.
3. **Brucker P., Jurisch B.** A new lower bound for the job-shop scheduling problem // Eur. J. Oper. Res. 1993. N 64 (2). P. 156—167.
4. **Brucker P.** Scheduling Algorithms. Berlin, Heidelberg und New York: Springer-Verlag, 1998.
5. **Balas E., Lancia G., Serafini P., Vazacopoulos A.** Job-shop scheduling with deadlines // J. Combinatorial Optimization. 1998. N 1 (4). S. 324—353.
6. **Зак Ю. А.** Некоторые свойства задач теории расписаний // Автоматика и телемеханика. 1978. № 1. С. 123—132.
7. **Zack Yu., Rotin S.** Mathematisches Modell und Algorithmen der Termin- und Reihefolgeplanung — Im Internet, 2004, 29 s. URL: http://www.optimorum.de/doc/J_II_Cmax.pdf
8. **Зак Ю. А.** Прикладные задачи теории расписаний и маршрутизации перевозок. М.: URSS, 2012, 394 с.
9. **Зак Ю. А.** Свойства допустимых и оптимальных расписаний выполнения работ на одной машине // Проблемы управления. 2012. № 5. С. 54—61.
10. **Зак Ю. А.** Построение допустимых и оптимальных расписаний выполнения работ на одной машине. Кибернетика и системный анализ. 2012. № 1. С. 62—82.
11. **Free** Distribution. URL: http://www.optimorum.de/form1.htm

---

УДК 004:658.512.2

В. М. Третьяков, д-р техн. наук, проф.,
ФГБОУ ВПО "Ковровская государственная технологическая академия им. В. А. Дегтярева",
e-mail: tretykov.kovrov@list.ru

Элементы теории паттернов для моделирования изделий машиностроения

Рассмотрены элементы теории паттернов, используемые при моделировании изделий машиностроения, и объекты моделирования. Для визуального представления паттерновых сетей и их компонентов используется графический формализм теории графов.

Ключевые слова: *моделирование, паттерновые сети, проектирование изделий, семейство изделий*

Паттерновые сети являются инструментом анализа и проектирования компьютерных информационных систем [1], технологических процессов [2], изделий машиностроения и их семейств [3—7]. Цель статьи — рассмотрение элементов теории паттернов, используемых при моделировании изделий машиностроения, имеющих составные части.

Большинство окружающих нас объектов представляют собой системы, построенные из соединенных между собой составных частей. Составные части снабжены "органами" для создания соединений, поэтому *образующая*, являющаяся основным непроизводным элементом теории паттернов У. Гренандера [8], может служить моделью любых составных частей. *Связи* образующей соответствуют "органам" составных частей, служащим для соединения.

Связи и узлы сопряжения

Термин *связь* образующей, используемый в теории паттернов, неточно отражает суть явления. Речь идет не о соединении чего-то с чем-то, а об "органе", который имеет образующая, для создания соединения. Эту "неточность" использованного термина отметил в примечании на с. 13 первого тома книги [8] редактор перевода Ю. И. Журавлев. Вместо термина *связи* он предложил термин *узлы*. Далее вместо термина *связь* применяется термин *узел сопряжения* (или просто *узел*). Этот термин также используется автором для обозначения фрагмента конструкции составной части, служащего для соединения с другим объектом [3—7].

В машиностроении *узел сопряжения — это совокупность поверхностей (или их фрагментов, линий, точек) составной части, через которые она обменивается потоками энергии, вещества и сигналов с другим объектом* [7]. В него входят участки поверхностей, которые в данный момент времени непосредственно осуществляют обмен потоками. Для передачи потока поверхности узлов соединены друг с другом. Узлы могут соединяться только попарно, так как, если две поверхности соприкасаются, то третья взаимодействовать с ними не может. Понятие *узел сопряжения* является обобщением таких широко применяемых в технике понятий, как ус-

тановочные и присоединительные места, посадочное отверстие, привалочная плоскость и т. п.

Узлы сопряжения обладают определенными свойствами, признаками, параметрами. Множество признаков узла сопряжения включает описание входящих в него поверхностей, например, формы и взаимного расположения, размеров, отклонения формы и расположения, чистоты обработки поверхностей, материалов. Совокупность признаков π^{β} определяет *исполнение узла*. Узлы сопряжения, отличающиеся признаками, имеют разное исполнение.

Природа передаваемых потоков позволяет разделить множество узлов сопряжения составной части на виды. К одному виду будем относить те из них, которые передают потоки одной и той же физической природы. Электрическую энергию передает один вид узлов, механическую энергию — другой вид и т. д. Распределение потоков по узлам сопряжения составной части определяется выполняемой функцией. Каждый вид узлов сопряжения может быть разделен на подвиды. Число подвидов равно максимальному числу узлов данного вида у составной части. Далее будем считать, что все узлы составной части передают потоки одной физической природы.

Свойства передаваемых потоков определяют параметры узлов сопряжения. От приложенных сил зависят размеры резьбы в резьбовых соединениях, сила тока определяет размеры контактов, соединяющих электронные компоненты и т. п.

Узел сопряжения составной части — это базовое элементарное конструкторское понятие. Введение его позволяет использовать модели, построенные на основе теории паттернов, для описания строения любого технического устройства, разделенного на части. *Составным частям соответствуют образующие, сложным конструкциям — паттерновые сети. Узлам сопряжения* составных частей ставятся в соответствие *узлы* образующих, а *соединениям — связки*, образованные их узлами.

Графические схемы образующих и паттерновых сетей

Графический формализм, используемый в работе [8] для изображения образующих, не очень удачен. Он получен разрезанием ребер графа, смежных одной вершине. "Обрезки" ребер снабжены треугольниками для визуализации соединений (рис. 1).

В работе [3] и последующих работах автор использовал графы для изображения структурных моделей, отражающих узлы сопряжения изделий. Вершина графа соответствует узлу сопряжения. Ребро показывает соединение двух узлов (рис. 1). Дуга графа — это ребро, учитывающее направление передачи потока. Вершины графа, изображающие узлы, принадлежащие одной образующей, заключены в оболочку. Оболочка показывает, что узлы — неотъемлемая часть образующей, и облегчает "чтение" модели.

На рис. 2 показаны структурные модели фрагментов технических устройств: шарикоподшипника, соединения пайкой, резьбового соединения. Составные части обозначены цифрами. Индекс в обозначении образующей g_i равен номеру составной части, моделью которой она является. Узлы сопряжения обозначены символом β_{ij}. Первый индекс указывает номер образующей, которой он принадлежит, второй индекс — номер узла в структуре ее узлов. В нижней части рис. 2 отмечены поверхности деталей, входящие в узлы β_{12} и β_{21} составных частей 1 и 2.

Использование графов делает схему паттерновой сети более лаконичной и компактной. Но возникает небольшая проблема: как различать входные и выходные узлы на схеме образующей. Ее можно решить, используя показатели узлов.

В большинстве изделий отнесение узла сопряжения к входным или выходным узлам зависит от направления передаваемого потока. На рис. 3 показаны два варианта использования одного и того же клинового механизма. Здесь *I* — входной, а *O* — выходной потоки механической энергии. Образующие g_1, g_2 и g_3 соответствуют клиньям *1*, *2* и корпусу *3*. Взаимодействующие фрагменты поверхностей, относящиеся к обозначенным узлам, выделе-

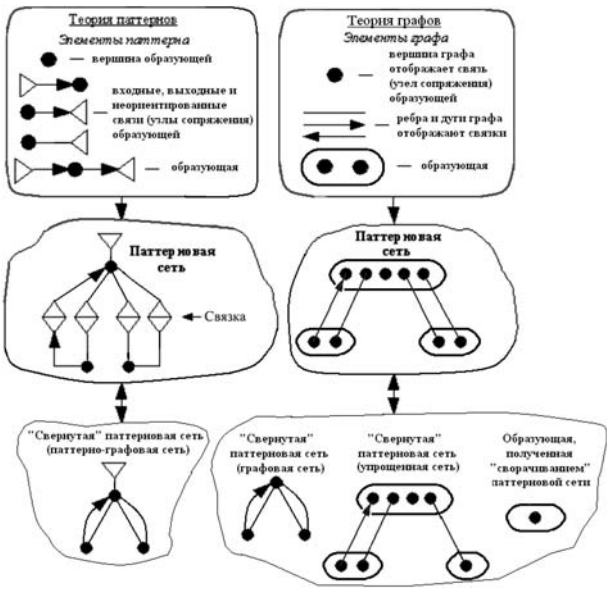


Рис. 1. Варианты графического представления образующих и паттерновых сетей

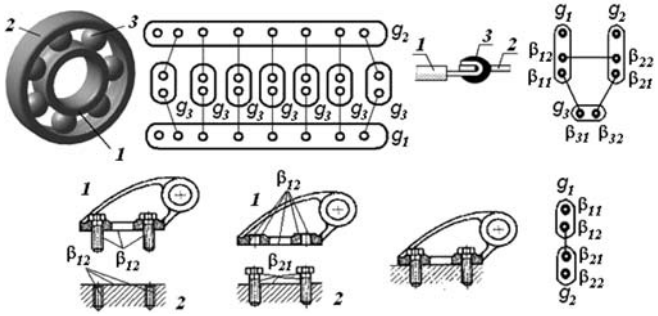


Рис. 2. Фрагменты сборочных единиц изделий и их структурные модели

ны серым цветом. На варианте слева звено 1 является входным, а звено 2 — выходным, а на варианте справа — наоборот. В структурных моделях это учитывается направлением дуг, поэтому двум вариантам использования звена 1 соответствует одна и та же образующая g_1.

Паттерновую сеть можно "свертывать" (см. рис. 1), получая паттерно-графовую, графовую, упрощенную паттерновую сеть или образующую. Упрощенная паттерновая сеть получается слиянием узлов образующей, передающих в одном направлении одному и тому же объекту одинаковые по физической природе потоки. Возможна обратная операция развертывания графовой и паттерно-графовой сети в паттерновую сеть, а также разделение узла сопряжения на несколько узлов.

Показатели узлов образующих и отношение согласования

В машиностроении используют разные соединения составных частей (резьбовые, сварные и т. п.). Они отличаются типом, видом и параметрами. Для идентификации соединений составных частей в изделиях и соответствующих им связок в структурных моделях удобно использовать числа. В изделии имеем соединение под номером K, в паттерновой сети ему соответствует "связка K" (см. рис. 3).

Все исполнения двух соединенных узлов могут быть разделены на два вида: охватывающее (например, β_{31} на рис. 3) и охватываемое β_{12}. В некоторых соединениях деление на охватывающее и охватываемое исполнения можно провести условно. Например, исполнение одного узла считать охватывающим (β_{21} на рис. 3), а соединенного с ним — охватываемым (β_{13}).

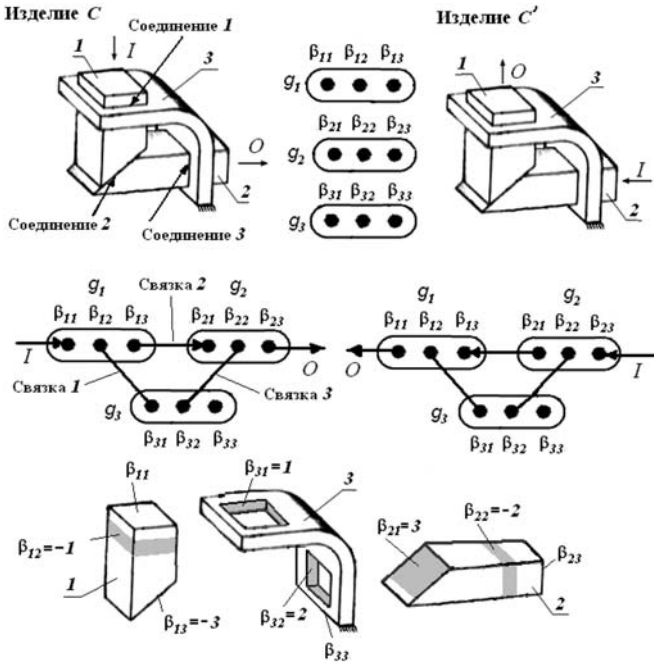


Рис. 3. Два варианта использования обратимого механизма и его структурные модели

Число K, обозначающее соединение, можно применить для идентификации создающих его узлов сопряжения. Охватывающее и охватываемое исполнения узла будем учитывать знаками плюс и минус. Например, для узла, имеющего охватывающее исполнение, можно использовать знак "+" ($\beta_{31} = 1$ на рис. 3), а для узла с охватываемым исполнением — знак "−" ($\beta_{12} = -1$). Узлы сопряжения, имеющие одинаковое обозначение (номер и знак совпадают), идентичны и соединяться не могут. Узлы, обозначения которых отличаются только знаком, позволяют образовывать соединения.

Обозначения узлов сопряжения, учитывающие их исполнение, будем использовать как показатели узлов образующих. Это позволит единообразно записывать отношения согласования (ρ-соединено) узлов образующих. Для этого используются алгебраические уравнения вида $\beta_{ij} + \beta_{kl} = 0$ (уравнения сборки), где β_{ij} и β_{kl} — обозначения узлов сопряжения, образующих связку [4]. Например, для связок на рис. 3 отношение согласования будет иметь вид $\beta_{12} + \beta_{31} = 0$; $\beta_{13} + \beta_{21} = 0$; $\beta_{22} + \beta_{32} = 0$. Связка существует, если признаки узлов сопряжения, вошедших в уравнение, отличаются только знаком: $\beta_{12} + \beta_{31} = (-1) + 1 = 0$.

Предложенный вариант показателя узлов и отношения согласования позволили решить задачу синтеза изделий, не имеющих избыточного разнообразия составных частей [4]. Она была сведена к задаче определения показателей узлов совокупности образующих, используемых в конечном множестве паттерновых сетей. Значения показателей определяются решением системы уравнений, включающей уравнения сборки и уравнения эквивалентности узлов образующих вида: $\beta_{mn} = \beta_{pq}$.

Введенный показатель позволяет идентифицировать входные и выходные узлы образующей. Будем различать входы и выходы по знаку показателя. Пусть узел является выходным, если его показатель имеет знак "минус". Узел с положительным значением показателя — входной. Вид отношения согласования ρ-соединено сохраняется.

Узел сопряжения не только обеспечивает соединение образующих, но и передает потоки. Поэтому, кроме показателя исполнения β_{ij} и показателя π^{β}_{ij}, характеризующего признаки поверхностей узла, необходим показатель p^{β}_{ij}, описывающий передаваемый поток (его вид и параметры).

Состав и структура паттерновой сети, преобразование подобия

Полученная в результате соединения образующих паттерновая сеть (далее просто сборка) c характеризуется составом $\{g_1, g_2, ..., g_n\}$, внутренней структурой σ_c (соединителем) и структурой внешних узлов сопряжения $R^e(c)$, служащих для соединения с другими объектами (см. укрупненную модель сборки c' на рис. 4). Соединитель σ можно

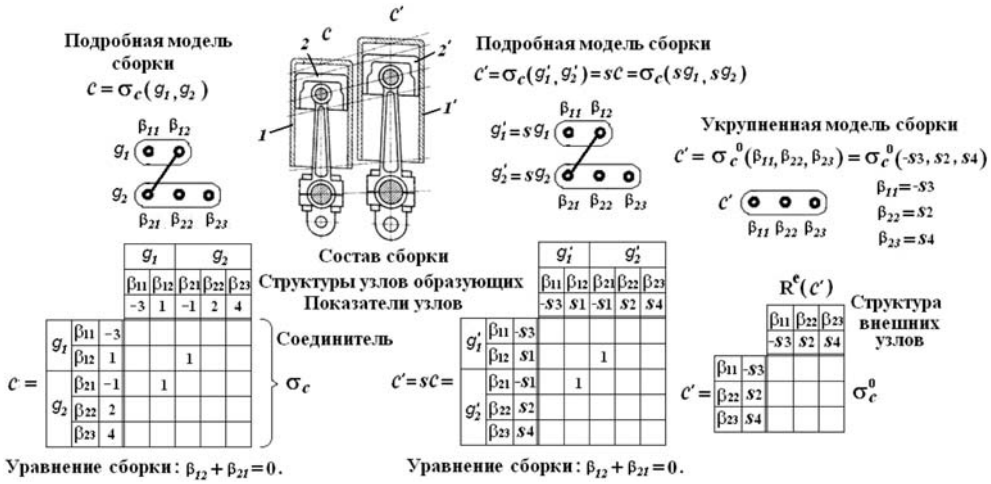


Рис. 4. Размерно-подобный ряд изделий и их структурные модели с матрицами смежности узлов сопряжения образующих

представить в виде к-дольного графа связей узлов сопряжения или матрицы смежности (рис. 4).

Для учета в моделях возможных исполнений составных частей, входящих в размерно-подобные ряды, используется преобразование подобия s (рис. 4). Двум размерно-подобным парам составных частей *1*, *1′* и *2*, *2′* ставятся в соответствие две пары образующих, связанных преобразованием подобия: $g_1' = sg_1$, $g_2' = sg_2$. На сборки распространяется преобразование подобия: *состав*$(sc) = \{sg_1, sg_2, ..., sg_n\}$, *структура*$(sc) =$ *структура*$(c) = \sigma_c$.

Преобразование подобия s влияет на признаки узлов, но не влияет на структуру узлов сопряжения, индекс класса образующих и отношение согласования. Так же как и для сборки C для связки, образованной узлами β_{12} и β_{21}, сборки C' (см. рис. 4) отношение согласования выполняется: $\beta_{12} + \beta_{21} = s1 + (-s1) = s(1 - 1) = 0$.

Алгебраическое описание образующих и паттерновых сетей

Для описания остовов образующих и паттерновых сетей удобно использовать алгебраическую форму [8] $c = \sigma_c(g_1, g_2, ..., g_n)$ (см. рис. 4). Образующая тоже может быть представлена в алгебраической форме подобно сборке: $g_{ij} = \sigma^0_{ij}(\beta_{i1}, \beta_{i2}, ..., \beta_{in})$ [5]. Ее составляющими являются узлы сопряжения. Соединитель σ^0_{ij} представляет собой граф, не содержащий ребер (ноль-граф). Пример ноль-графа σ^0_c для укрупненной модели сборки c' показан на рис. 4. Он является подграфом графа σ_c, содержащим только изолированные вершины. Переход от подробной модели сборки c' к ее укрупненной модели, учитывающей только внешние узлы, можно представить в виде:

$$c' = \sigma_c(g_1', g_2') = \sigma_c(\sigma_1^0(\beta_{11}, \beta_{12}), \sigma_2^0(\beta_{21}, \beta_{22}, \beta_{23})) =$$

$$= \sigma_c(\beta_{11}, \beta_{12}, \beta_{21}, \beta_{22}, \beta_{23}) = \sigma_c^0(\beta_{11}, \beta_{22}, \beta_{23}).$$

Выводы

Паттерновые сети позволяют описывать формально и наглядно структурные и параметрические свойства изделий машиностроения. С помощью паттерновых сетей можно с разной степенью подробности описывать строение составных частей изделий вплоть до учета всех их поверхностей [7]. Рассмотренные структурные модели, построенные на основе теории паттернов, были использованы [3—7] при проектировании семейств изделий машиностроения и при решении задачи уменьшения номенклатуры составных частей изделия.

Термин *связь* образующей неточно отражает наличие у нее атрибута, служащего для создания соединения. Для его обозначения лучше использовать термин *узел сопряжения* образующей.

Использование термина *модульный* (*модульное мышление*, *модульная сеть* и т. п.) не отражает то новое, что внесла теория паттернов в создаваемые модели объектов окружающего мира. Образующая — модель любой составной части, но такой же моделью может служить и вершина графа. Новое в модели — описание *узлов сопряжения* составной части.

Список литературы

1. **Шуткин Л. В.** Новое мышление компьютерного мира: применение сетей данных // НТИ. Сер. 2. Информ. Процессы и системы. 1998. № 6. С. 23—27.
2. **Смирнов В. А., Семенов А. М.** Модульный принцип моделирования ремонта ПС // Мир транспорта. 2012. № 3.
3. **Третьяков В. М.** Групповое проектирование технических устройств. Разработка элементной базы // Автоматизация и современные технологии. 1997. № 9. С. 10—21.
4. **Третьяков В. М.** Математические модели для определения номенклатуры компонентов элементной базы семейства машин // Проблемы машиностроения и надежности машин. 1999. № 2. С. 8—13.
5. **Третьяков В. М.** Семейство изделий и его элементная база // Автоматизация и современные технологии. 2001. № 1. С. 28—34.
6. **Третьяков В. М.** Основы проектирования семейства изделий // Справочник. Инженерный журнал. Приложение. 2004. № 6. 24 с.
7. **Третьяков В. М.** Использование понятия "узел сопряжения" при конструировании изделий машиностроения // Вестник машиностроения. 2010. № 12. С. 24—30.
8. **Гренандер У.** Лекции по теории образов / Пер. с англ. под ред. Ю. И. Журавлева. М.: Мир. Т. 1. 1979. 384 с.; Т. 2. 1981. 448 с.; Т. 3. 1983. 432 с.

УДК 004.94

В. Н. Гридин[1],
д-р техн. наук, проф.,
В. И. Анисимов[1],
д-р техн. наук, проф., гл. науч. сотр.,
e-mail: vianisimov@inbox.ru,
А. И. Ларистов[2], канд. техн. наук, доц.,
e-mail: ailaristov@inbox.ru,
Аль-Шами Мохаммед[2], аспирант
[1] Центр информационных технологий в проектировании, г. Одинцово, Моск. обл.
[2] СПб ГЭТУ ("ЛЭТИ")

Модель предметной области для базы данных схемных компонентов схемотехнической САПР

Рассматриваются вопросы организации информационного обеспечения Web-ориентированной схемотехнической САПР на основе технологий баз данных. Предлагается модель предметной области в виде обобщенной ER-диаграммы, учитывающей иерархию наследования моделей схемных компонентов. Дополнительно в ER-диаграмме отображаются математические зависимости между атрибутами моделей различных уровней. Показывается преимущество механизма хранимых процедур для учета формульных зависимостей на уровне базы данных. На основе разработанных ER-диаграмм предметной области предлагается реализация реляционной базы данных моделей компонентов в среде универсальной СУБД Oracle 9i.

Ключевые слова: *база данных моделей компонентов, ER-диаграммы, хранимые процедуры, Web-ориентированная САПР, СУБД Oracle 9i*

Введение

Одним из перспективных направлений развития автоматизированного проектирования в настоящее время является использование Интернет-технологий для организации дистанционного доступа к проектирующим подсистемам и информационным ресурсам САПР [1—3]. Применение подобных технологий позволяет организовать коллективную работу над общим проектом распределенного коллектива пользователей и обеспечить удаленный доступ к централизованным проектным данным. Возможным решением данной задачи является адаптация функционирования существующих промышленных САПР на основе реинжиниринга архитектуры системы [3] с разнесением готовых модулей системы между клиентом и сервером так, чтобы добиться оптимальной производительности в условиях низкоскоростных каналов сети Интернет и лимитированных ресурсов Web-серверов. При таком подходе возникает необходимость радикальной перестройки информационного обеспечения САПР на основе технологий баз данных, так как информационный фонд в традиционных архитектурах САПР построен с использованием файлов операционной системы и библиотечных файлов и не ориентирован на удаленный доступ через Web-интерфейс.

Основной формой организации информационного обеспечения в наиболее популярных схемотехнических САПР [4, 5] является библиотечная форма, представляющая собой набор текстовых файлов в специальных форматах, содержащих информацию о схемных компонентах. Для формирования и редактирования библиотек компонентов используются специализированные диалоговые редакторы. Достоинством библиотечной организации данных является простота реализации и возможность визуального контроля информации. В то же время подобной организации присущи и существенные недостатки:

- отсутствие процедур подбора и поиска компонентов по совокупности критериев;
- незащищенность информации от несанкционированного доступа;
- отсутствие эффективных средств удаленного доступа к данным;
- отсутствие разграничения прав пользователей на модификацию и удаление информации;
- трудности централизованного копирования и восстановления данных.

Кроме того, на содержательном уровне в библиотеках отсутствует информация об эксплуатационных и предельных параметрах схемных компонентов (нормативно-справочная информация), нет информации о математическом описании моделей компонентов и информации об эквивалентных схемах моделей. Неполная информация затрудняет процесс подбора схемных компонентов для проекта и заставляет инженера-схемотехника пользоваться дополнительными информационными источниками.

Для устранения рассмотренных недостатков предлагается использовать в процессе проектирования Web-ориентированную базу данных схемных компонентов (БДСК), содержащую полный объем информации и размещенную на Web-сервере информационных ресурсов САПР [6].

Использование технологий баз данных в САПР

Процесс создания информационного обеспечения САПР на основе технологий баз данных включает этап разработки структурной модели предметной области (инфологической модели данных). Этот этап предполагает выявление перечня физических объектов, их свойств и установление логических связей между ними. Для схемотехнических САПР основной составляющей информационного обеспечения являются данные о моделях радиоэлектронных компонентов, входящих в состав проектируемой схемы.

Радиоэлектронные компоненты различаются по своему функциональному назначению и конструктивному исполнению. В зависимости от функционального назначения можно выделить группы однородных компонентов определенного вида (резисторы, конденсаторы, биполярные транзисторы, полевые транзисторы, операционные усилители и т. д.). В свою очередь, компоненты внутри вида можно сгруппировать по типам. Компоненты определенного типа характеризуются одинаковой технологией изготовления, близкими значениями электрических параметров и геометрических размеров (например, резисторы типа МЛТ-0,5, транзисторы типа КТ3102Б и т. д.). В базе данных, как правило, хранится информация о среднестатистическом экземпляре компонента каждого типа. Следует отметить, что в рассмотренной классификации компонентов возможно выделение *подвидов* радиоэлектронных компонентов по некоторому общему характерному признаку или свойству (например, высокочастотные биполярные транзисторы, маломощные биполярные транзисторы и т. д.).

В процессе разработки БДСК для построения семантической модели будем использовать одну из наиболее популярных семантических моделей данных — модель "Сущность-Связь" (часто ее называют кратко ER-моделью от Entity-Relationship) [7]. На различных вариантах ER-модели основано большинство современных подходов к проектированию баз данных (главным образом, реляционных). Моделирование предметной области базируется на использовании графических диаграмм, включающих небольшое число разнородных элементов. Простота и наглядность представления концептуальных схем баз данных в ER-модели привели к ее широкому распространению в CASE-системах, поддерживающих автоматизированное проектирование реляционных баз данных. Одна из наиболее популярных и развитых среди множества разновидностей ER-моделей применяется в CASE-системе компании Oracle [8].

Основными понятиями ER-модели являются сущность, связь и атрибут. Сущность — это реальный или представляемый объект, информация о котором должна сохраняться и быть доступной. В диаграммах ER-модели сущность представляется в виде прямоугольника, содержащего имя сущности. При этом имя сущности — это имя типа, а не некоторого конкретного экземпляра этого типа. В качестве примера может быть рассмотрена сущность БИПОЛЯРНЫЙ ТРАНЗИСТОР с экземплярами ("транзистор К611Б", "транзистор 1Т905А", "транзистор 2Т798А"). Этот пример показывает, что в базе данных будут содержаться однотипные структуры данных (экземпляры сущности), описывающие биполярные транзисторы.

Иерархия наследования моделей компонентов

В сложных технических системах объекты предметной области могут иметь несколько уровней представления. Представления о проектируемой системе при использовании системного подхода к проектированию расчленяют на иерархические уровни. На верхнем уровне используют наименее детализированное представление, отражающее только самые общие черты и особенности проектируемой системы. На следующих уровнях степень подробности описания возрастает, при этом рассматривают уже отдельные блоки системы, но с учетом воздействий на каждый из них его соседей. На каждом из уровней представления объект проектирования может иметь различное семантическое описание, что приведет к расщеплению сущностей ER-модели на несколько иерархических уровней. При этом возникает иерархия наследования (или иерархия категорий), что представляет собой особый тип объединения сущностей, которые имеют общие характеристики.

Таким образом, сущность может быть расщеплена на два или более взаимно исключающих подтипов сущности, каждый из которых включает общие атрибуты и/или связи. Эти общие атрибуты и/или связи явно определяются один раз на более высоком уровне. В принципе, подтипизация может продолжаться на более низких уровнях, но опыт использования ER-модели при проектировании баз данных показывает, что в большинстве случаев оказывается достаточно двух-трех уровней.

Например, различные схемные компоненты (биполярный транзистор, полевой транзистор, операционный усилитель, интегральная схема) имеют общие характеристики: число выводов, тип корпуса, условия применения и т. д. Из их общих свойств можно сформировать обобщенную сущность (родовой предок) "Компонент" (рис. 1), чтобы представить информацию, общую для всех типов компонентов. Специфическая для каждого типа ра-

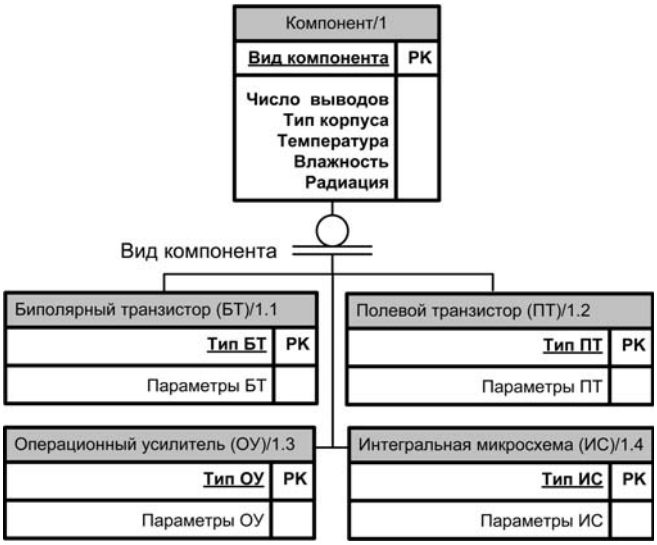


Рис. 1. Иерархия наследования сущностей

диоэлектронного компонента (РК) информация может быть расположена в категориальных сущностях (потомках) "Биполярный транзистор" (БТ), "Полевой транзистор" (ПТ), "Операционный усилитель" (ОУ), "Интегральная микросхема" (ИС).

Обычно иерархии наследования создаются, когда несколько сущностей имеют общие по смыслу атрибуты, либо когда сущности имеют общие по смыслу связи, либо когда это диктуется бизнес-правилами. При этом обобщенная сущность получает наименование супертипа сущности, а категориальные сущности определяются как подтипы сущности. Для каждой категории можно указать дискриминатор — атрибут родового предка, который показывает, как отличить одну категориальную сущность от другой (атрибут Вид компонента на рис. 1).

Дальнейшая подтипизация сущностей возможна на уровне более детального рассмотрения вида компонента. В зависимости от этапа проектирования электронной схемы схемный компонент может быть представлен семантически различными данными. В этом случае возможно существование нескольких типов сущностей, имеющих различные структуры данных и относящихся к одному и тому же физическому объекту. На этапе схемотехнического проектирования радиоэлектронный компонент замещается математической моделью, характеризуемой определенной эквивалентной электрической схемой, системой уравнений и значениями параметров. При этом для одного и того же вида компонента можно сформировать ряд моделей разной степени точности, различающихся степенью учета тех или иных особенностей физических процессов, происходящих в реальном компоненте. Кроме того, для различных видов анализа (статический режим, частотная область, временная область) используются различные математические модели. Таким образом, возникает супертип сущности "Модель компонента" и подтипы сущности "Модель_1", "Модель_2" и т. д. Между экземплярами рассматриваемых сущностей возможно существование логических связей, которые отображаются на ER-диаграмме в виде линий и различаются по типу: "один к одному", "один ко многим" и "многие ко многим".

Обобщенная ER-диаграмма для базы данных модельных компонентов

Для сложных технических объектов помимо логических связей между экземплярами сущностей возможно наличие математических зависимостей, в которых в качестве параметров и переменных выступают атрибуты сущностей. Данная особенность не учитывается в "классической" ER-диаграмме, что снижает ее полноту и достоверность. Для отображения математических зависимостей дополним ER-диаграмму специальным графическим символом, используемым в блок-схемах программ и алгоритмов для обозначения типового процесса. На рис. 2 приведена обобщенная ER-диаграмма иерархии моделей для биполярного транзистора. Аналогичные ER-диаграммы построены и для других видов схемных компонентов.

При преобразовании ER-диаграммы в реляционную (более точно, в SQL-ориентированную) схему базы данных необходимо учитывать наличие супертипов и подтипов сущностей. Если в концептуальной схеме (ER-диаграмме) присутствуют подтипы,

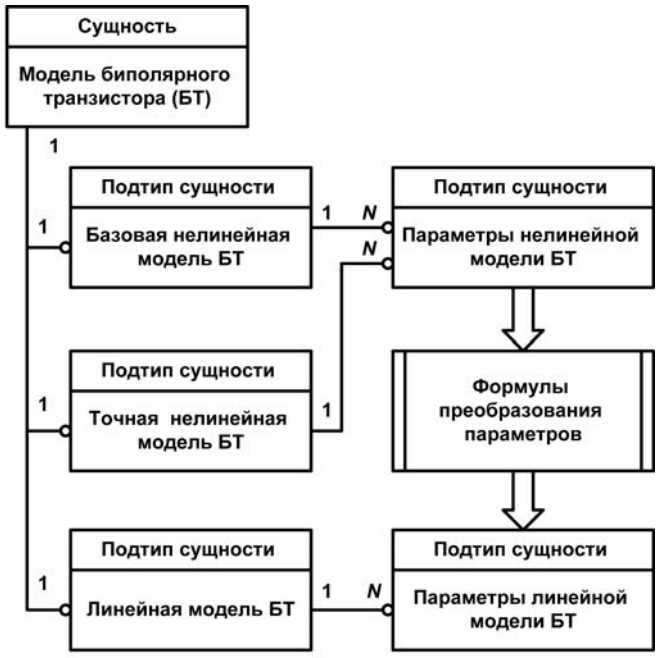


Рис. 2. Обобщенная ER-диаграмма моделей биполярного транзистора

то возможны два способа их представления в реляционной схеме:

1) объединить все подтипы сущности в одной таблице;

2) для каждого подтипа сущности образовать отдельную таблицу.

При применении первого способа таблица создается для максимального супертипа (типа сущности, не являющегося подтипом), а для подтипов могут создаваться представления. Таблица содержит полный набор столбцов, соответствующий всем атрибутам (и связям) каждого подтипа. В таблицу добавляется, по крайней мере, один столбец, содержащий КОД ТИПА; он становится частью первичного ключа. Для каждой строки таблицы значение этого столбца определяет подтип сущности, экземпляру которого соответствует строка. Столбцы этой строки, которые соответствуют атрибутам и связям, отсутствующим в данном подтипе сущности, должны содержать неопределенные значения.

При использовании второго способа для каждого подтипа первого уровня супертип воссоздается с помощью представления UNION (из всех таблиц подтипов выбираются общие столбцы — столбцы супертипа).

На выбор способа перехода от ER-диаграммы к реляционной модели данных (схеме данных) для разрабатываемой БДСК влияют следующие факторы:

- высокая вложенность подтипов сущностей;
- большое разнообразие связей между сущностями (в модели данных присутствуют все три типа связей "1-1", "1-*N*", "*N*-*N*");
- возможность добавления новых видов схемных компонентов и проектов в процессе эксплуатации баз данных.

Учет этих факторов приводит к необходимости использования второго способа при построении реляционной модели данных. В соответствии с данным способом каждому подтипу сущности соответствует своя таблица, что значительно упрощает логику работы приложения и дает возможность расширения базы данных.

Математические зависимости между экземплярами сущностей могут быть учтены также двумя способами. Первый способ предполагает реализацию соответствующих формул на уровне программных модулей системы. Такой подход используется в большинстве современных САПР и отличается простотой выполнения [4, 5]. Недостатком данного способа является закрытость соответствующего программного кода и невозможность его модификации для учета дополнительных эффектов и зависимостей, вносимых в модели пользователем САПР.

Более гибким подходом является предлагаемая авторами реализация формульных зависимостей с помощью механизма хранимых процедур. В этом случае все математические вычисления проводятся непосредственно в базе данных. Код хранимых процедур содержит инструкции и операторы расширенного подмножества языка SQL и легко трансформируется в соответствии с требованиями пользователя. При таком способе возможна дополнительная обработка результатов моделирования, например, построение различных графических зависимостей (вольт-амперных характеристик и др.), иллюстрирующих поведение компонента в проектируемой схеме.

Заключение

В статье предложен новый подход к организации информационного обеспечения Web-ориентированных САПР, основанный на замене традиционно используемых библиотечных файлов интегрированной базой данных схемных компонентов. В результате анализа предметной области разработана обобщенная иерархическая ER-диаграмма, учитывающая математические зависимости между атрибутами разных по уровню моделей компонентов. На основе предложенных моделей данных была реализована реляционная база данных модельных компонентов в среде универсальной СУБД Oracle 9i, что позволяет эффективно использовать язык PL/SQL, содержащий расширенный набор математических функций и типов данных двойной точности, для программирования хранимых процедур. Применение технологий баз данных обеспечивает быстрый доступ инженера-схемотехника к полной информации о схемном компоненте, позволяет выполнить подбор схемных компонентов по совокупности параметров и организовать процедуры по защите и копированию данных в схемотехнических САПР.

Список литературы

1. **Анисимов В. И., Гридин В. Н.** Методы построения систем автоматизированного проектирования на основе Internet-технологий и компактной обработки разреженных матриц // Информационные технологии в проектировании и производстве. 2009. № 1.

2. **Гридин В. Н., Анисимов В. И., Ларистов Д. А.** Архитектура схемотехнических САПР со встроенным браузером // Автоматизация в промышленности. 2009. № 11.

3. **Дмитревич Г. Д., Ларистов А. И., Мохсен А. А.** Архитектура WEB-ориентированных САПР // Информационно-управляющие системы. 2010. № 5 (48). С. 20—23.

4. **Хайнеман Р.** Визуальное моделирование электронных схем в PSPICE: Пер. с нем. М.: ДМК Пресс, 2008. 336 с.

5. **Разевиг В. Д.** Система сквозного проектирования электронных устройств Design Center 8.0. М.: Солон, 1999. 698 с.

6. **Ларистов А. И., Лячек Ю. Т., Абу Сара М. Р.** Интегрированные базы данных в программных системах проектирования электронных схем // Информационно-управляющие системы. 2009. № 3 (40). С. 69—71.

7. **Дейт К.** Введение в системы баз данных. К.: Диалектика, 1999. 320 с.

8. **Кайт Т.** Oracle для профессионалов. Кн. 1. Архитектура и основные особенности. 3-е изд., перераб. и доп.: Пер. с англ. СПб.: ООО "ДиаСофтЮП", 2005. 656 с.

УДК 004.056

Е. С. Новикова, канд. техн. наук, ст. науч. сотр.,
И. В. Котенко, д-р техн. наук, проф., зав. лаб.,
e-mail: ivkote@comsec.spb.ru
Санкт-Петербургский институт
информатики и автоматизации РАН

Проектирование компонента визуализации для автоматизированной системы управления информационной безопасностью

Представлены результаты проектирования подсистемы визуализации автоматизированной системы управления информационной безопасностью. Исследованы механизмы визуализации в современных системах управления информационной безопасностью. Описана архитектура подсистемы визуализации, позволяющая расширять графический функционал системы и использовать различные технологии для реализации графических элементов. Приведено описание реализации системы, иллюстрирующее предложенный подход.

Ключевые слова: *автоматизированные системы управления информационной безопасностью, визуализация событий безопасности, архитектура компонента визуализации*

Введение

Сложность современных информационных инфраструктур, необходимость поддержки различных платформ, рост числа приложений и клиентов обусловливает необходимость применения автоматизированных систем управления безопасностью в информационных системах (*Security Information and Event Management*, *SIEM*). Важной составной частью SIEM-системы является ее подсистема визуализации, с помощью которой пользователь может получить результаты работы системы, определить правила ее функционирования и управлять ресурсами системы. Кроме того, поскольку SIEM-система предоставляет доступ ко всей информации, полученной от датчиков безопасности, которая в большинстве случае имеет текстовый формат, то графическое представление этих данных значительно повышает эффективность работы оператора системы, позволяя ему решать задачи в заданные временные рамки.

В проекте MASSIF [1] Седьмой рамочной программы Евросоюза разрабатывается SIEM-система нового поколения, которая позволит не только эффективно выявлять различные события безопасности, но и моделировать возможные сценарии развития атак и прогнозировать возможные будущие шаги нарушителя при его оперативном обнаружении. В настоящей работе представлены результаты проектирования компонента визуализации SIEM-системы. Исследуются принципы проектирования графического интерфейса и механизмы визуализации в современных SIEM-системах, описывается архитектура компонента визуализации SIEM-системы и интерфейсы взаимодействия различных структурных компонентов системы, приводится описание реализации предложенного подхода на примере компонента моделирования атак и оценки уровня защищенности сети.

1. Механизмы визуализации

Задачей SIEM-систем является формирование системного видения на информационную безопасность контролируемой системы [2]. Получая на вход данные от различных датчиков безопасности (системы обнаружения/предотвращения вторжений, сканеры уязвимости, роутеры, и т. д.), SIEM-системы нормализуют данные, приводя к единому формату и анализируя их, своевременно выявляют события безопасности.

Общая структура SIEM-системы представлена на рис. 1.

Компонент визуализации предоставляет пользователю возможность взаимодействовать с различными функциональными компонентами SIEM-системы, одновременно объединяя их в единую систему. Он позволяет пользователю отслеживать сетевую активность в режиме реального времени, анализировать события, хранящиеся в репозитории, управлять инцидентами безопасности и ресурсами сети, реагировать на события, формировать отчет-

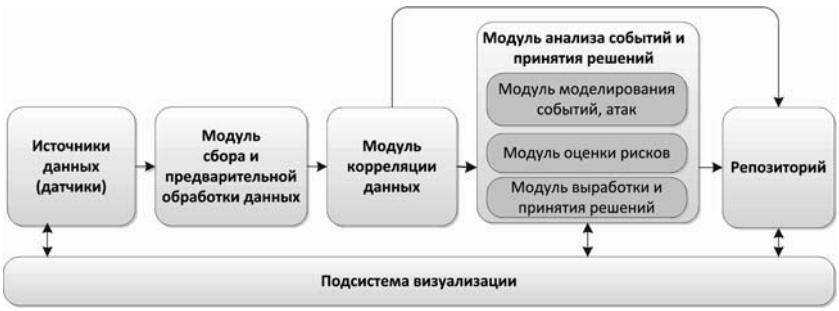


Рис. 1. Общая структура SIEM-системы

ную документацию, конфигурировать датчики безопасности и т. п.

Обычно информация в SIEM-системах группируется с помощью панелей управления (*dashboards*), предназначенных для решения определенных задач защиты информации. В зависимости от характера решаемой задачи различают *стратегические*, *аналитические* и *оперативные* панели управления. Стратегические панели управления используются для представления обобщенной информации о состоянии безопасности контролируемой системы, в то время как оперативные панели позволяют пользователю получить детальные данные о конкретном элементе информационной системы. SIEM-системы имеют достаточно разнообразный набор панелей управления [3—6], однако можно выделить ряд типовых панелей, реализованных в большинстве SIEM-систем и предназначенных для решения наиболее часто возникающих задач защиты информации: "Угрозы", "Сетевая активность", "Ресурсы системы", "Риски", "Отчеты". Кроме того, пользователь может гибко настраивать существующие панели управления для решения конкретной задачи, добавляя и конфигурируя необходимые ему графики и таблицы [3, 4, 6].

На рис. 2 (см. четвертую сторону обложки) показана главная панель управления системы OSSIM, содержащая информацию об уровне риска, числе разрешенных инцидентов безопасности и общем числе контролируемых событий безопасности, числе событий безопасности, зарегистрированных SIEM-системой, и событий, обнаруженных другими датчиками безопасности. Таким образом, с ее помощью пользователь может оценить эффективность как функционирования самой SIEM-системы, так и работы системных администраторов безопасности.

Следует отметить, что современные SIEM-системы реализуют схожий графический функционал [3, 4]. Наиболее часто используются стандартные двухмерные модели представления данных (линейные графики, диаграммы с накоплением, гистограммы, секторные диаграммы). С их помощью выявляются наиболее часто используемые сетевые службы, наиболее уязвимые хосты и приложения, число попыток повышения прав, число неудачных таких попыток и т. д. Использование этих графических моделей представления данных объясняется, во-первых, их простотой, и, во-вторых, тем, что они позволяют пользователю быстро выявить наиболее критические объекты.

Для представления данных порядкового типа, таких как метрики безопасности, критичность событий безопасности, часто используют пиктограммы, позволяющие оценить текущее значение параметра в контексте его возможных значений.

Топология компьютерных сетей и информационные потоки отображаются в виде графов, вершины которых соответствуют узлам сети, а грани — связям или информационным потокам между ними [3—5]. Модели визуализации, основанные на графах, также используют для отображения различных правил безопасности, так как они в большинстве случаев легко могут быть представлены в виде направленного графа.

Во многих SIEM-системах реализована поддержка географических карт для отображения географического расположения хостов контролируемой сети, поскольку такое представление позволяет пользователю определить границы ответственности подразделений организации, в отличие от других панелей управления, которые формируют общее видение сетевого статуса и уровня рисков всей системы в целом.

Немногие SIEM-системы реализуют более сложные графические модели представления данных, которые позволяют выявлять скрытые зависимости между данными, и в конечном итоге формулировать правила обнаружения новых информационных угроз, такие как карты деревьев [7], графы на параллельных координатах [8], матричные представления и т. д. Таким образом, очевидна необходимость реализации в SIEM-системах более сложных средств отображения информации, стимулирующих процесс визуального анализа данных [9]. Эта задача может быть решена, если при проектировании системы предусмотреть возможность динамического расширения графического функционала системы, путем добавления новых или усовершенствованных реализованных моделей визуализации.

2. Подходы к построению систем визуализации

В настоящее время наиболее популярной парадигмой для разработки масштабируемых распределенных приложений является сервис-ориентированная архитектура. Многие современные системы визуализации поддерживают возможность удаленной визуализации данных [10—12]. Например, в работе [13] для графической интерпретации данных использована технология "толстого клиента", в рамках которой все вычисления, включая рендеринг данных, осуществляются на стороне клиента, данные загружаются в базу данных через соответствующие веб-интерфейсы. Авторы работы [14] предложили перенести все вычисления, необходимые для визуализации, на сторону сервера и разработали трехуровневую архитектуру, состоящую из уровня клиента, задающего интерфейс пользователя, уровня управляющих веб-сервисов, который предоставляет определенный интерфейс системе визуализации, и наконец, уровня графических компонентов, который определяет основную функциональность для визуального анализа. Н. Холмберг и др. [15] рассматривают возможность интеграции различных технологий визуализации в одном веб-приложении. Кроме того, они приводят краткий обзор существующих 2D- и 3D-технологий визуализации. Б. Греттарсон и др. [12] исследуют проблему масштабируемости в существующих интерактивных

веб-приложениях, предоставляющих различные функции визуализации сетей. Авторы разработали инструмент WiGis, реализующий интерактивную визуализацию графов, и приводят результаты экспериментов по оценке его эффективности путем визуализации графов, состоящих из сотен тысяч узлов.

Большинство систем, предоставляющих веб-сервисы визуализаций данных [10, 11], реализуют механизм подключаемых компонентов, позволяющий легко расширять графическую функциональность.

3. Архитектура подсистемы визуализации

Для разработки архитектуры визуализации авторами настоящей статьи был адаптирован подход, предложенный в работе [14]. Использование принципов сервис-ориентированной архитектуры обеспечивает требование функциональной расширяемости компонента: при добавлении нового графического элемента нет необходимости заново переписывать приложение. Кроме того, данный подход хорошо согласуется моделью графических систем "данные → отображение → вид → управление" [16].

Архитектура подсистемы визуализации состоит из трех основных компонентов: *пользовательский интерфейс*, *управляющие сервисы* и *графические элементы*.

На рис. 3 представлено схематичное изображение предложенной архитектуры. Стрелками обозначены потоки данных между ее элементами. Выделение пользовательского интерфейса в отдельный компонент позволяет поддерживать разработку разнообразных видов графических интерфейсов, начиная от простой командной строки, заканчивая сложным многооконным интерфейсом с различными панелями управления.

Управляющие сервисы обеспечивают подключение и регистрацию функциональных компонентов и графических элементов, поэтому условно их можно разделить на две группы: контроллер графических элементов и контроллер SIEM-сервисов.

Контроллер графических элементов предоставляет стандартный интерфейс по работе с потоками

Рис. 3. Архитектура подсистемы визуализации

визуализации, поддерживающий создание и остановку графического потока, который реализуется на уровне графических элементов.

Контроллер SIEM-сервисов обеспечивает управление функциональными модулями.

Графические элементы представляют собой библиотеку графических примитивов — графов, лепестковых диаграмм, гистограмм, карт деревьев, географических карт и т. д., и осуществляют обработку входных данных, их отображение и взаимодействие пользователя непосредственно с входными данными.

Важным моментом является организация взаимодействия между функциональными и графическими элементами. Необходимо реализовать достаточно гибкую связь между компонентами, обеспечив таким образом их относительную независимость друг от друга. Для этого предлагается использовать следующий сценарий взаимодействия. Когда какому-либо функциональному модулю необходимо визуализировать данные, он запрашивает у контроллера графических элементов перечень доступных графических элементов и выбирает наиболее подходящий элемент, оценивая его свойства. Контроллер графических элементов создает экземпляр элемента и возвращает его функциональному модулю, который, в свою очередь, передает ему данные для отображения. Графический элемент соответствующим образом обрабатывает данные и предоставляет уже готовый экран с визуализацией и элементами ее управления (масштабирование, управление точкой обзора и т. д.). Таким образом, для реализации данного подхода необходимо, во-первых, выработать общий формат обмена данными и, во-вторых, определить интерфейс графического объекта, позволяющий передать данные для их графической интерпретации и получить результат визуализации.

Благодаря такому решению любое изменение во внутренней логике какого-либо элемента системы (функционального или графического) не затронет другие элементы, даже связанные с ним. Кроме того, такой подход позволит разрабатывать и тестировать компоненты независимо, что позволит повысить качество самого приложения. При реализации графических элементов могут быть использованы различные технологии визуализации, например, Java3D, Flash, SVG и т. д.

4. Реализация подсистемы визуализации

Предложенный подход был реализован в виде программного компонента на языке Java. Динамическое подключение модулей осуществляется с помощью программного каркаса Apache Felix [17], который реализует спецификацию динамической плагинной шины для приложений Java [18]. В рамках описанной нами архитектуры каркас Apache Felix исполняет роль управляющих сервисов, обеспечивая динамическое подключение модулей и их взаимодействие с помощью стандартных функций: *install()*, *start()*, *stop()*, *uninstall()*, *getServiceReferences()* и т. д.

Функциональные компоненты SIEM-системы рассматриваются как независимые компоненты, каждому из которых в подсистеме визуализации может соответствовать несколько панелей управления. Для того чтобы иметь возможность подключать разработанные для них графические интерфейсы пользователя, был определен интерфейс *AbstractSiemComponent*, позволяющий подсистеме визуализации получить доступ к графическим элементам компонента и элементам управления (меню, контекстное меню, панель инструментов).

При запуске компонента, реализующего данный интерфейс, в главном окне подсистемы визуализации появляются элементы и панели управления, соответствующие данному компоненту. Графические компоненты реализуют интерфейс *AbstractGraphicalComponent*, определяющий методы передачи данных, создания графического представления и получения списка доступных действий пользователя с изображением. Оба интерфейса *AbstractSiemComponent* и *AbstractGraphicalComponent* наследуют интерфейс *AbstractPlugin*, который позволяет подсистеме получать сведения о подключаемом компоненте (название, версия, краткое описание, иконка, тип подключаемого элемента) и передавать ему информацию об исполняемой среде.

UML-диаграмма интерфейсов приведена на рис. 4.

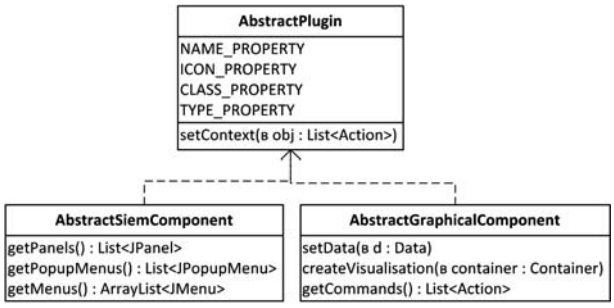


Рис. 4. UML-диаграмма интерфейсов подсистемы визуализации

Авторами был разработан графический интерфейс для компонента моделирования атак и оценки уровня защищенности компьютерной сети (рис. 5, см. четвертую сторону обложки).

Главное окно компонента разделено на четыре функциональных вида. Главный вид *C* представляет исследуемую сеть. Пользователь может добавлять и удалять узлы компьютерной сети. Пиктограммы оборудования являются настраиваемыми, и пользователь может настроить вид по своему желанию. Фон пиктограммы используется для представления значения метрики безопасности, выбираемой пользователем из предопределенного списка. Вид *A* отражает иерархическую структуру компьютерной сети. Используя редактор свойств узлов сети *B*, пользователь может сконфигурировать каждый узел сети и саму сеть в целом. Он может задать как значения заранее определенных свойств, таких как IP-адрес, тип хоста (веб-сервер, файловый сервер, роутер и т. д.), установленное программное и аппаратное обеспечение, так и определить собственные свойства хостов.

Панель управления *D* используется для отображения значений метрик защищенности: уровень защищенности анализируемой сети, уровень рисков, уровень достоверности информации. Такое расположение информации об исследуемой сети позволяет оценить ее состояние в целом, и пользователь имеет возможность проанализировать результаты оценки защищенности сети в контексте исходной информации, которая доступна в разнообразных видах, расположенных на одной панели управления.

Для представления отчета о выявленных уязвимостях используют стандартные секторные диаграммы. Этот элемент представления является подключаемым, и если компонент моделирования атак находит его в перечне доступных графических сервисов, то на его панели управления появляется дополнительный элемент управления, позволяющий получить отчет в виде секторных диаграмм (рис. 6, см. четвертую сторону обложки).

Красной рамкой выделены элементы (пункт меню и вкладка Network Constructor), добавленные в подсистему визуализации, и кнопка вызова отчета об уязвимостях в виде секторных диаграмм, добавленная на панель инструментов компонента.

Заключение

В настоящей работе представлены результаты проектирования компонента визуализации для автоматизированных систем управления информационной безопасностью. Для решения поставленной задачи были исследованы механизмы визуализации, использующиеся в современных системах управления информационной безопасностью, рассмотрены различные подходы к проектированию систем визуализации. Предложена архитектура компонента визуализации, учитывающая результаты проведенного исследования и позволяющая легко расширять функциональность приложения. Предложенный подход был реализован в виде подсистемы визуализации, позволяющей динамически подключать модули. На примере компонента моделирования атак и оценки защищенности компьютерной сети было показано взаимодействие с подключаемыми графическими элементами.

В дальнейшем планируется расширение функциональности компонента визуализации, в том числе за счет реализации новых форм визуализации данных.

Работа выполняется при финансовой поддержке РФФИ, программы фундаментальных исследований ОНИТ РАН (проект № 2.2), государственного контракта 11.519.11.4008 и при частичной финансовой поддержке, осуществляемой в рамках проекта Евросоюза MASSIF.

Список литературы

1. **MASSIF** Website. URL: http://www.massif-project.eu/ (дата обращения: 26.12.2012).
2. **Miller D., Harris S., Harper A., VanDyke S., Blask C.** Security Information and Event Management (SIEM) Implementation. New York: McGraw Hill Professional, 2010. 464 p.
3. **Arcsight SIEM** website. URL: http://www.ndm.net/siem/main/arcsight-siem (дата обращения: 26.12.2012).
4. **QRadar SIEM** website. URL: http://q1labs.com/products/qradar-siem.aspx. (дата обращения: 26.12.2012).
5. **Tivoli** Security Information and Event Manager websight. URL: http://www-01.ibm.com/software/tivoli/products/security-info-event-mgr/index.html. (дата обращения: 26.12.2012).
6. **OSSIM SIEM** website. http://communities.alienvault.com/ (дата обращения: 26.12.2012).
7. **Shneiderman B., Plaisant C.** Treemaps for space-constrained visualization of hierarchies. URL: http://www.cs.umd.edu/hcil/treemap-history/index.shtml (дата обращения: 26.12.2012).
8. **Inselberg A., Dimsdale B.** Parallel coordinates: a tool for visualizing multidimensional geometry // Proc. of the 1-st Conference on Visualization. IEEE Computer Society Press, 1990. P. 361—378.
9. **Новикова Е. С., Котенко И. В.** Анализ механизмов визуализации для обеспечения защиты информации в компьютерных сетях // Труды СПИИРАН. Вып. 23. СПб.: Наука, 2012. С. 7—30.
10. **ManyEyes** visualisation services. URL: http://www-958.ibm.com/software/analytics/manyeyes/ (дата обращения: 26.12.2012).
11. **Touchgraph** navigator. URL: http://www.touchgraph.com, (дата обращения: 26.12.2012).
12. **Gretarsson B., Bostandjievy S., O'Donovanz J., Hoellererx T.** WiGis: A Framework for Scalable Web-based Interactive Graph Visualizations // Graph Drawing (GD 2009). LNCS, 2009. Vol. 5849. P. 119—134.
13. **Ananthuni R., Karki B. B., Bollig E. F., da Silva C. R. S., Erlebacher G.** A Web-Based Visualization and Reposition Scheme for Scientific Data // Proc. of the Conf. on Modeling Simulation and Visualization Methods. CSREA Press, 2006. P. 311—317.
14. **Wood J., Brodlie K. W., Seo J., Duke D. J., Walton J.** A web services architecture for visualization // Proc. IEEE 4th International Conference on eScience. IEEE Computer Society Press, 2008. P. 1—7.
15. **Holmberg N., Wuensche B., Tempero E.** A Framework for Interactive Web-Based Visualization // Proc. of the 7th Australasian User Interface Conference (AUIC '06). Australian Computer Society, 2006. Vol. 50. P. 137—144.
16. **Chi E. H.** A Taxonomy of Visualization Techniques Using the Data State Reference Model // IEEE Symposium on Information Visualization, 2000. P. 69—75.
17. **Apache** Felix framework. URL: http://felix.apache.org/site/index.html (дата обращения: 26.12.2012).
18. **OSGi** technology. URL: http://www.osgi.org/Main/HomePage (дата обращения: 26.12.2012).

УДК 512.772.7

С. И. Алешников, канд. техн. наук, доц., зав. каф.,
e-mail: elliptec@mail.ru,
М. В. Алешникова, ст. препод.,
e-mail: aleshnikova_m_v@mail.ru,
А. А. Горбачёв, канд. техн. наук, доц.,
e-mail: terjer@mail.ru,
Балтийский федеральный университет
им. И. Канта, г. Калининград

Протокол доверенного шифрования на основе модифицированного алгоритма вычисления спаривания Вейля на алгебраических кривых для облачных вычислений

Предлагается протокол доверенного шифрования через прокси-сервер на основе модифицированного алгоритма Миллера, вычисляющего спаривание Вейля на гиперэллиптических кривых рода 2, в частности, на суперсингулярных кривых с высокой степенью кручения над простым полем. Получены оценки эффективности модифицированного алгоритма.

Ключевые слова: *билинейные спаривания, спаривание Вейля, спаривание Тэйта, алгебраические кривые гиперэллиптические кривые, якобиан, генерация ключей, шифрование, расшифрование*

В настоящее время билинейные спаривания превратились в главный инструмент построения эффективных криптосистем с открытым ключом на эллиптических и гиперэллиптических кривых. При этом основной проблемой реализации таких систем является вычисление спаривания. Уже разработано довольно много алгоритмов таких вычислений.

Весьма быстрые алгоритмы вычисления спариваний Вейля и Тейта на эллиптической кривой основаны на алгоритме Миллера [8]. Много эффективных алгоритмов для вычисления спаривания на эллиптических кривых представлено в [3]. В работе [5] впервые описан алгоритм для вычисления спаривания на кривых рода $\geqslant 2$, дающий укорочение цикла счета по сравнению с алгоритмом Миллера. В [2] идеи о сокращении цикла счета распространяются на суперсингулярные абелевы многообразия. Позднее были предложены и другие виды спариваний: η_T-спаривание для обыкновенных эллиптических кривых, *ate*-спаривание и скрученное *ate*-спаривание, оптимизированное *ate*-спаривание, ate_i-спаривание, спаривание *R-ate*, оптимальное спаривание и др.

Гиперэллиптическое *ate*-спаривание [6] обладает двумя хорошими свойствами. Во-первых, длина цикла в алгоритме Миллера для вычисления спаривающей функции в g раз короче, чем у соответствующего спаривания Тейта, где g — род основной кривой C. Во-вторых, спаривание автоматически редуцируется, т. е. заключительное возведение в степень, необходимое в спаривании Тейта, не проводится. Кроме того, при использовании кручения

высокой степени для гиперэллиптического спаривания последнее может иметь преимущества перед случаем эллиптических кривых.

Так как спаривание Тейта может быть вычислено более эффективно, чем спаривание Вейля, исследователи рассматривают главным образом спаривание Тейта. В работе [7] показано, что даже обеспечивая высокий уровень секретности, такой как при длине ключа 192 или 256 бит, вычисление спаривания Вейля может иногда происходить быстрее, нежели спаривания Тейта. В частности, в [11] установлено, что для некоторых гиперэллиптических кривых с высокой степенью кручения вычисление спаривания Вейля может быть более быстрым, чем спаривания Тейта, *ate* и ate_i.

Недавние достижения в криптографии на основе спариваний представлены в [9].

В настоящей работе предложен протокол доверенного шифрования через прокси-сервер на основе модифицированного алгоритма Миллера, вычисляющего спаривание Вейля на гиперэллиптических кривых рода 2, в частности, на суперсингулярных кривых с высокой степенью кручения над простым полем. Получены оценки эффективности модифицированного алгоритма.

В схеме облачных вычислений участвуют пользователи *A*, *B* и прокси-сервер. Пользователь *A* желает делегировать пользователю *B* через прокси-сервер право расшифровывать некоторые из получаемых *A* сообщений. Это — так называемое доверенное (рас)шифрование. Он шифрует отправляемое сообщение и доверяет прокси-серверу дошифровать его. Прокси-сервер его дошифровывает и пересылает пользователю *B*. Тот расшифровывает полученное зашифрованное сообщение. Однако основная проблема состоит в том, что необходимо, чтобы прокси-сервер дошифровал пересылаемое сообщение именно ключом дошифрования, предназначенным пользователю *B* и никому другому.

Исходные данные. Пусть G_1 — аддитивная, G_2 — мультипликативная циклические группы простого порядка l и $e : G_1 \times G_1 \to G_2$ — билинейное отображение (спаривание Вейля). Пространство исходных сообщений есть G_2, пространство зашифрованных сообщений есть $G_1 \times G_2$, H_1 — хэш-функция, действующая из {0, 1}* в G_1, где {0, 1}* — множество бинарных последовательностей произвольной длины, и H_2 — хэш-функция из G_2 в G_1. Выбрана случайная образующая ζ группы G_1. Обозначаем $\mathrm{F}_l^{\times}$ мультипликативную группу простого поля Φ_l. Случайно выбирается целое $s \in \mathrm{F}_l^{\times}$ в качестве "master key" и публикуется $s \cdot \zeta$.

Начальное формирование ключей. Пользователь *A* с идентификационными данными ID_A запрашивает в центре сертификации приватный секретный ключ. Центр сертификации вычисляет $h_A = H_1(ID_A)$, $D_A = s \cdot h_A$ и отсылает D_A пользователю *A*. Те же самые операции выполняются для пользователя *B*.

Генерация секретного значения. Пользователь *A* случайно выбирает целое число $x_A \in \mathrm{F}_p^{\times}$ (по модулю l). То же самое делает *B*.

Генерация секретных ключей. Пользователь *A* вычисляет секретный ключ $sk_A = x_A \cdot D_A = s \cdot x_A \cdot h_A$ и сохраняет его в секрете. То же самое действие выполняет пользователь *B*. Для доверенной расшифровки *A* также выбирает случайное целое число t и также сохраняет его в секрете.

Генерация открытых ключей. Пользователь *A* вычисляет свой открытый ключ $pk_A = (h_A; s \cdot x_A \cdot \zeta)$, публикует его, и любой, кто хочет послать сообщение к *A*, может использовать его для шифрования. То же самое делает *B*. Заметим, что открытый ключ формируется без дополнительного обращения в центр сертификации.

Прямое шифрование. Чтобы зашифровать сообщение $m \in G_2$, которое может расшифровать только он сам, пользователь *A* случайно выбирает целое число r и вычисляет $c = C_A(m) = (r \cdot \zeta;\ m \cdot e(h_A;\ s \cdot x_A \cdot \zeta)^r)$. Для доверенной расшифровки пользователем *B* пользователь *A* также случайно выбирает целые числа r и вычисляет $c' = C'_A(m) = (pk_B(t \cdot r);\ r \cdot \zeta;\ m \cdot e(h_A;\ s \cdot x_A \cdot \zeta)^r)$, где $pk_B(t \cdot r) = C_B(t \cdot r)$ — результат шифрования $t \cdot r$ с помощью открытого ключа pk_B пользователя *B*.

Прямое расшифрование. Для расшифрования $C_A(m) = (u,\ v)$ с помощью секретного ключа sk_A пользователь *A* вычисляет

$$v/e(sk_A;\ u) = m \cdot e(h_A;\ s \cdot x_A \cdot g)^r / e(s \cdot x_A \cdot h_A;\ r \cdot \zeta) = m.$$

Генерация ключей для доверенного дошифрования. Если пользователь *A* хочет делегировать право расшифрования пользователю *B*, то он случайно выбирает $x \in G_2$ и вычисляет доверенный ключ дошифрования

$$rk_{A \to B} = (-s \cdot x_A \cdot h_A + t \cdot H_2(x);\ C_B(x)),$$

который затем посылает прокси-серверу.

Прокси-дошифрование. Для дошифрования шифрованного текста $C'_A(m)$ с помощью ключа дошифрования $rk_{A \to B}$ прокси-сервер вычисляет

$$c'' = m \cdot e(h_A;\ s \cdot x_A \cdot \zeta)^r \cdot e(-s \cdot x_A \cdot h_A + t \cdot H_2(x);\ r \cdot \zeta) = m \cdot e(t \cdot H_2(x);\ r \cdot \zeta)$$

и затем посылает $(pk_B(t \cdot r),\ c'',\ C_B(x))$ пользователю *B*. Если прокси-сервер пошлет данную тройку другому пользователю, то этот другой пользователь, не зная секретного ключа *B*, не сможет восстановить $t \cdot r$. Правда, такая конструкция требует дополнительных ресурсов пользователю *A* для вычисления $pk_B(t \cdot r)$, но меньших, чем прямое шифрование данных m открытым ключом pk_B. Тем самым обеспечивается бóльшая безопасность доверенного шифрования, чем, например, в схеме, изложенной в работе [10].

Доверенное расшифрование. После получения $(pk_B(t \cdot r);\ c'';\ C_B(x))$ пользователь *B* с помощью

своего секретного ключа получает $t \cdot r$, затем вычисляет $t \cdot r \cdot \zeta$, расшифровывает $C_B(x)$, чтобы получить x, и, наконец, получает сообщение $c''/e(H_2(x), t \cdot r \cdot \zeta) = m$.

Безопасность данной схемы шифрования основана на трудности решения билинейной проблемы Диффи — Хеллмана. Возможно обобщение на многосерверное доверенное шифрование.

В качестве группы G_1 используется подгруппа кручения якобиана гиперэллиптической кривой, в качестве G_2 — группа корней в конечном поле. Рассмотрим ситуацию подробнее.

Пусть $\overline{\mathrm{F}}_q$ — алгебраическое замыкание конечного поля Φ_q с числом элементов q. Гиперэллиптическая кривая C рода $g \geqslant 1$ над Φ_q есть несингулярная плоская кривая, определяемая уравнением $y^2 + h(x)y = f(x)$, где $f(x)$ — унитарный многочлен степени $2g + 1$, $h(x)$ — многочлен степени, не превосходящей g. Множество $C(\mathrm{F}_{q^k})$ точек кривой C с координатами в расширении F_{q^k} степени k (которая далее уточняется) для $g \geqslant 2$ не образует группы, однако C можно вложить в абелево многообразие J_C размерности g, называемое якобианом C. Это многообразие изоморфно группе Pic^0_C классов дивизоров степени нуль, называемой группой Пикара. Якобиан кривой C, определенной над F_{q^k}, обозначается $J_C(\mathrm{F}_{q^k})$, он изоморфен группе классов дивизоров степени нуль $\mathrm{Pic}^0_C(\mathrm{F}_{q^k})$ на C, определенных над F_{q^k}.

Зафиксируем подгруппу некоторого порядка l группы $J_C(\Phi_q)$. Говорят, что эта подгруппа имеет степень вложения k, если порядок l делит $q^k - 1$, но не делит $q^i - 1$ для всех $i : 0 < i < k$. Это означает, что группа корней μ_l степени l из единицы имеет максимальный порядок l и содержится в F_{q^k}. В описанном выше протоколе $G_2 = \mu_l$. Для криптографических приложений l должно быть большим простым числом, делящим порядок $\#J_C(\Phi_q)$ якобиана $J_C(\Phi_q)$, и наибольший общий делитель $\mathrm{НОД}(l, q) = 1$. Это в точности соответствует выбору простого порядка l групп, используемых в вышеописанном протоколе.

Пусть $J_C(\mathrm{F}_{q^k})[l]$ есть l-я группа кручения. Именно эта группа принимается за G_1. Будем обозначать $\mathrm{F}_{q^k}(C)$ поле F_{q^k}-рациональных функций на кривой C. Для $\overline{D}_1 \in J_C(\mathrm{F}_{q^k})[l]$, $\overline{D}_2 \in J_C(\mathrm{F}_{q^k})$, где D_1 — дивизор, представляющий $\overline{D}_1$, D_2 — дивизор, представляющий $\overline{D}_2$, и носители этих дивизоров дизъюнктны (т. е. $\mathrm{supp}(D_1) \cap \mathrm{supp}(D_2) = \varnothing$), существует рациональная функция $f_{l, D_1} \in \mathrm{F}_{q^k}(C)$ такая, что дивизор функции $\mathrm{div}(f_{l, D_1}) = lD_1$. Полагаем

$$f_{l, D_1}(D_2) = \prod_{P \in C(\overline{\mathrm{F}}_q)} f_{l, D_1}(P)^{\mathrm{ord}_P(D_2)}.$$

Главной вычислительной проблемой является построение такой рациональной функции f_{l, D_1} и вычисление ее значения $f_{l, D_1}(D_2)$. Пусть G_{iD_1, jD_1} — рациональная функция такая, что

$$\mathrm{div}(G_{iD_1, jD_1}) = iD_1 + jD_1 - (iD_1 \oplus jD_1),$$

где $\oplus$ — групповой закон в J_C и $(iD_1 \oplus jD_1)$ — редуцированный дивизор. Модифицированный алгоритм Миллера для гиперэллиптических кривых строит рациональную функцию f_{l, D_1} на основе следующей рекуррентной формулы:

$$f_{i+j, D_1} = f_{i, D_1} f_{j, D_1} G_{iD_1, jD_1}.$$

Спаривание Вейля допускает модификацию, использующую так называемое кручение. Пусть C — кривая рода g, определенная над полем K. Кривая C' над K, изоморфная C, называется кручением C. Более того, кривая C' называется кручением степени d кривой C, если существует изоморфизм $\phi : C' \to C$, определенный над K_d, где K_d — расширение K степени d и d минимально. Для всякой кривой C, определенной над K, множество ее кручений $\mathrm{Twist}(C/K)$ есть множество кривых C'/K, изоморфных C над K.

Для построения кручения данной кривой C над полем K необходимо вычислить группу автоморфизмов $\mathrm{Aut}(C)$ кривой C. Для гиперэллиптических кривых рода 2 над конечными полями возможными редуцированными группами автоморфизмов являются:

$$C_2,\ C_{10},\ C_2 \times S_3,\ V_4,\ D_8,\ D_{12},\ 2D_{12},$$
$$\widetilde{S}_4,\ \widetilde{S}_5,\ M_{32} \text{ или } M_{160},$$

где C_n — циклическая группа порядка n; V_4 — четверная группа Клейна; D_n — группа диэдра порядка n; M_n — группа порядка n, возникающая из точных последовательностей [4]; $2D_{12}$, $\widetilde{S}_4$ и $\widetilde{S}_4$ есть 2-накрытия группы диэдра D_{12} и симметрических групп S_4 и S_5 соответственно.

Степень d кручения кривой C зависит от порядка элемента из $\mathrm{Aut}(C)$, т. е. если C' — кручение C степени d, то группа $\mathrm{Aut}(C)$ должна содержать элемент порядка d. Однако если характеристика поля K есть $\mathrm{char}(K) > 5$, то $\mathrm{Aut}(C)$ всегда есть подгруппа одной из перечисленных групп. В таком случае для

$\mathrm{char}(K) > 5$ возможны лишь случаи $d = 1, 2, 3, 4, 5, 6, 8, 10$. Например, для конечного поля $K = \Phi_q$, где q — большое простое число, $q \equiv 1 \pmod 8$, кривая C_1: $y^2 = x^5 + ax$ имеет группу автоморфизмов $C_8 \subset \widetilde{S}_4$. Ее кручением степени $d = 8$ является кривая C_1' с уравнением $y^2 = x^5 + a\lambda x$, соответствующий изоморфизм есть

$$(x, y) \mapsto (\lambda^{1/4}x, \lambda^{5/8}y),$$

где $\lambda \in \Phi_q$ и не является z-степенным вычетом в Φ_q для $z \in \{1, 2, 4, 8\}$.

Спаривание Вейля есть невырожденное билинейное отображение

$$e_w(\cdot, \cdot) : J_C(\mathrm{F}_{q^k})[l] \times J_C(\mathrm{F}_{q^k})[l] \to \mu l,$$

где $e_w(\overline{D}_1, \overline{D}_2) = (-1)^r f_{l, D_1}(D_2)/f_{l, D_2}(D_1)$. Это спаривание не требует заключительного возведения в степень, однако нуждается в двух итерациях цикла Миллера.

В работе [1] найдены уравнения оптимальных кривых рода 3, не являющихся гиперэллиптическими, а именно:

$$\begin{cases} z^2 = \alpha_0 + \alpha_1 x + \alpha_2 x^2 + \beta y; \\ y^2 = x^3 + ax + b; \end{cases}$$

$$\begin{cases} z^2 = \alpha_0 + \alpha_1 x + \alpha_2 x^2 + (\beta_0 + \beta_1 x) y; \\ y^2 = x^3 + ax + b; \end{cases}$$

$$\begin{cases} z^2 = \alpha_0 + \alpha_1 x + \alpha_2 x^2 + \alpha_3 x^3 + (\beta_0 + \beta_1 x) y; \\ y^2 = x^3 + ax + b, \end{cases}$$

где $\alpha_0, \alpha_1, \alpha_2, \alpha_3, \beta_0, \beta_1, a, b$ — коэффициенты из Φ_q. Нижележащая эллиптическая кривая E задана уравнением $y^2 = x^3 + ax + b$. Группа автоморфизмов $\mathrm{Aut}_{\mathrm{F}_q}(X)$ кривой X над полем Φ_q изоморфна группе диэдра порядка 6. Предложено обобщение спаривания Вейля на указанные кривые и разработан соответствующий алгоритм его вычисления.

Список литературы

1. **Alekseenko E., Aleshnikov S., Markin N., Zaytsev A.** Optimal curves over finite fields with discriminant —19 // J. Finite Fields ant Their Applications. 2011. Vol. 17, Issue 4 (July 2011). P. 350—358.
2. **Barreto P.S.L.M., Galbraith S.** et al. Efficient Pairing Computation on Supersingular Abelian Varieties // In Designs, Codes and Cryptography. 2007. Vol. 42 (3).
3. **Blake I. F., Seroussi G., Smart N. P.** Advances in Elliptic Curve Cryptography. Cambridge University Press, 2005.
4. **Cardona G., Nart E., Pujol'as J.** Curves of genus two over fields of even characteristic // Math. Z. 250. 2005. N 1. P. 177—201.
5. **Duursma I. M., Lee H.-S.** Tate Pairing Implementation for Hyperelliptic Curves $y^2 = x^p - x + d$ // ASIACRYPT. Springer. 2003. LNCS 2894. P. 111—123.
6. **Granger R., Hess F.** et al. Ate Pairing on Hyperelliptic Curves // Advance in Cryptology — EUROCRYPT'2007. Springer-Verlag, 2007. LNCS 4515. P. 430—447.
7. **Koblitz N., Menezes A.** Pairing-based Cryptography at High Security Levels. In Cryptography and Coding. Springer-Verlag, 2005. LNCS 3796. P. 235—249.
8. **Miller V. S.** Short Programs for Functions on Curves (Manuscript). 1986. URL: http://tcs.uj.edu.pl/~mistar/pdf/Miller1986Short Programs.pdf http://crypto.stanford.edu/miller/
9. **Pairing-Based** Cryptography — Pairing 2010 // 4th International Conference. Yamanaka Hot Spring, Japan, 2010. LNCS 6487.
10. **Wu X., Xu L., Zhang X.** CL-PRE: a Certificateless Proxy Re-Encryption Scheme for Cloud-Based Data Sharing // 11th annual ACM workshop on Digital rights management, 2011.
11. **Zhang F.** Twisted Ate Pairing on Hyperelliptic Curves and Applications. Cryptology ePrint Archive, Report 2008/274. 2008.

ИНФОРМАЦИЯ

26—29 ноября 2013 г.

Переславль-Залесский, ИПС имени А. К. Айламазяна РАН

Национальный Суперкомпьютерный Форум (НСКФ-2013)

Форум посвящен вопросам создания и практики применения суперкомпьютерных технологий, проводится при поддержке Отделения нанотехнологий и информационных технологий Российской академии наук.

НСКФ-2013 приглашает к участию:

- разработчиков суперЭВМ;
- разработчиков системного и прикладного программного обеспечения для суперЭВМ;
- разработчиков грид-технологий и грид-сервисов;
- представителей российских суперкомпьютерных центров;
- аспирантов и молодых ученых, работающих в суперкомпьютерной отрасли;
- потребителей суперкомпьютерных технологий.

В рамках Национального Суперкомпьютерного Форума проводятся следующие мероприятия:

- научно-практическая конференция;
- выставка;
- серия мастер-классов и тренингов;
- серия круглых столов;
- пресс-тур.

Рабочие языки Форума: русский и английский. Синхронный перевод не обеспечивается. Мероприятия пресс-тура проводятся на русском языке.

Подробная информация: http://www.nscf.ru/

УДК 621.391

С. В. Дворников[1], д-р техн. наук, доц., проф.,
e-mail: practicdsv@yandex.ru,
С. С. Дворников[2], студент,
В. В. Борисов[3], нач. сектора,
Д. С. Бабенко[1], курсант,
А. Г. Москалец[4], науч. сотр.,
А. А. Устинов[1], д-р техн. наук, проф., зам. нач. каф.,
А. П. Чихонадских[4], канд. техн. наук,
ст. науч. сотр., нач. научного центра
[1]Военная академия связи, г. Санкт-Петербург
[2]Санкт-Петербургский государственный политехнический университет
[3]НПФ ТИРС
[4]ГосНИИПП

Демодуляция сигналов с относительной фазовой манипуляцией с адаптивным порогом принятия решения

Предлагаются результаты аналитических исследований и данные компьютерного эксперимента по обоснованию выбора порога принятия решения при демодуляции сигналов с относительной фазовой манипуляцией. Для этих целей доказывается целесообразность учета апостериорных вероятностей достоверности демодулированных символов. Представляются основные этапы алгоритма, реализующего разработанный научный подход.

Ключевые слова: *демодуляция сигналов, адаптивный порог принятия решения, сигналы с относительной фазовой манипуляцией*

Введение

В общем случае демодуляция сигналов с относительной фазовой манипуляцией (ОФМ) сводится к процедурам сравнения фазы принятого символа с опорной величиной, в качестве которой выступает значение фазы символа, принятого на предыдущем временном интервале [1]. Реализация указанных процедур предусматривает дифференциальное кодирование информации разностью фаз между двумя последовательными импульсами. Для передачи i-го сообщения ($i = 1, ..., M$) фаза текущего сигнала должна быть смещена на $\varphi_i = 2\pi i/M$ радиан относительно фазы предыдущего. В этом случае детектор вычисляет координаты поступающего символа путем его корреляции с локально генерируемыми сигналами $\sqrt{2/T}\cos\omega_0 t$ и $\sqrt{2/T}\sin\omega_0 t$. Затем измеряет угол между вектором текущего и предыдущего принятых сигналов и на основании этого принимает решение о значении демодулированного символа.

Принципиально схемы демодуляции сигналов фазовой манипуляции и ОФМ отличаются тем, что в первом случае принятое излучение сравнивается с идеальным, в то время как во втором — обрабатываются два зашумленных сигнала. Следовательно, ОФМ дает вдвое больший шум, поэтому при ее использовании следует ожидать вдвое большую вероятность ошибки по отношению к фазовой манипуляции. Однако основное преимущество схемы ОФМ, заключающееся в простоте ее реализации, обеспечивает ей широкое применение в радиосистемах. Именно поэтому настоящая статья посвящена поискам практических решений, направленных на снижение ошибки демодуляции сигналов ОФМ в условиях шумов высокой интенсивности.

Обоснование порога принятия решения

Согласно [2] процесс демодуляции сигналов ОФМ сводится к задаче однозначного отображения принятой реализации S_k, где $k = 0,1$, в пространстве ее допустимых значений $V_k = \{V_0, V_1\}$. Причем $S_0 \in V_0$, а $S_1 \in V_1$. Тогда граница, разделяющая пространства V_1 и V_0, определяется величиной различия векторов S_1 и S_0:

$$\Delta l = \frac{1}{d}\ln\frac{p(0)}{p(1)}, \qquad (1)$$

где d — разность векторов S_1 и S_0, а $p(1)$ и $p(0)$ — априорные вероятности передачи сигналов S_1 и S_0 соответственно.

Следовательно, порог принятия решения по отнесению принятой реализации S_k к одной из областей V_1 и V_0 полностью определяется априорными вероятностями передачи сигналов по каналу связи.

Предположим, что демодулятор способен сохранять результаты обработки N элементов. В этом случае апостериорная частость проявления сигналов S_1 и S_0 определяется как

$$\tilde{p}(1) = \frac{n_1}{N} \text{ и } \tilde{p}(0) = \frac{n_0}{N}, \qquad (2)$$

где n_1 и n_0 — число решений о приеме S_1 и S_0.

Очевидно, что при $N \to \infty$ результат (2) можно рассматривать как апостериорные вероятности появления сигналов S_1 и S_0 на выходе канала связи.

Тогда, если априорные и апостериорные вероятности сигналов на входе и выходе совпадают, можно говорить об отсутствии ошибок демодуляции.

Для двоичного канала связи с двумя символами {0, 1} на входе и выходе апостериорные вероятности их проявления будут определяться как

$$\tilde{p}(0) = p(0)p(0/0) + p(1)p(0/1)$$
$$\text{и } \tilde{p}(1) = p(1)p(1/1) + p(0)p(1/0), \quad (3)$$

где $p(0/0)$ — условная вероятность принятия сигнала S_0 при передаче сигнала S_0; $p(0/1)$ — условная вероятность принятия сигнала S_0 при передаче сигнала S_1; $p(1/1)$ — условная вероятность принятия сигнала S_1 при передаче сигнала S_1; $p(1/0)$ — условная вероятность принятия сигнала S_1 при передаче сигнала S_0.

В случае симметричного канала связи, когда выполняются равенства

$$p(0/0) = p(1/1) = p,\ p(0/1) = p(1/0) = 1 - p, \quad (4)$$

имеем:

$$\tilde{p}(0) = p(0)p + p(1)(1 - p),$$
$$\tilde{p}(1) = p(1)p + p(0)(1 - p). \quad (5)$$

Из условия (5) следует, что в случае равновероятных сигналов ($p(1) = p(0)$) даже при наличии ошибок в канале связи (выполняются равенства (4)) априорные вероятности передачи сигналов равны апостериорным вероятностям их демодуляции, т. е. $p(1) = \tilde{p}(1)$ и $p(0) = \tilde{p}(0)$.

Следовательно, для симметричного канала связи с равновероятностными сигналами совпадение априорных и апостериорных вероятностей не свидетельствует об отсутствии ошибок.

Таким образом, можно заключить, что граница принятия решения между областями V_1 и V_2 зависит не только от $p(1)$ и $p(0)$, но и от $\tilde{p}(1)$ и $\tilde{p}(0)$.

Следовательно, решение задачи демодуляции сигналов ОФМ сводится к нахождению границы разделения сигналов S_1 и S_2. Согласно [2] вероятность ошибки для двоичного канала определяется соотношением

$$p_{\text{ош}} = p(1)p(0/1) + p(0)p(1/0). \quad (6)$$

Однако в выражении (6) отсутствуют величины $\tilde{p}(1)$ и $\tilde{p}(0)$. В связи с этим определим обратный канал связи, в котором апостериорные вероятности приема сигналов будут выступать в роли априорных, и наоборот. Вероятность ошибки в обратном канале связи определим в виде соотношения:

$$\tilde{p}_{\text{ош}} = \tilde{p}(1)p(1/0) + \tilde{p}(0)p(0/1). \quad (7)$$

Становится очевидным, что минимизация ошибки в обратном канале связи однозначно приведет к минимизации ошибки в прямом, и наоборот.

В таком случае ошибку демодуляции можно рассматривать с позиций суммарных ошибок прямого и обратного каналов связи:

$$p_{\text{д}} = p_{\text{ош}} + \tilde{p}_{\text{ош}} =$$
$$= (p(1) + \tilde{p}(0))p(0/1) + (p(0) + \tilde{p}(1))p(1/0). \quad (8)$$

Для минимизации выражения (8) определим, что условные вероятности $p(0/1)$ и $p(1/0)$ представляют вероятности попадания случайного вектора X в полупространства V_0 и V_1. Тогда, в соответствии с [3], условные вероятности $p(0/1)$ и $p(1/0)$ можно трактовать как значения функций распределения случайной величины X, плотности которых в областях V_0 и V_1 будут соответственно определяться выражениями

$$q_1 = \frac{dp(0/1)}{dV} \text{ и } q_0 = \frac{dp(1/0)}{dV}. \quad (9)$$

Тогда условные вероятности можно представить как

$$p(0/1) = \int_{V_0} q_1 dV,\ p(1/0) = \int_{V_1} q_0 dV. \quad (10)$$

С учетом равенств (10) ошибка демодуляции (8) будет определяться как

$$p_{\text{д}} = (p(1) + \tilde{p}(0)) \int_{V_0} q_1 dV + (p(0) + \tilde{p}(1)) \int_{V_1} q_0 dV. \quad (11)$$

Поскольку $\int_V q_1 dV = \int_V q_0 dV = 1$, где $V = V_1 + V_0$, то равенство (10) перепишем в виде

$$\int_{V_0} q_1 dV = 1 - \int_{V_1} q_1 dV,\ \int_{V_1} q_0 dV = 1 - \int_{V_0} q_0 dV. \quad (12)$$

Используя равенства (12), выражения для суммарной ошибки (11) окончательно запишем следующим образом:

$$p_{\text{д}} = (p(1) + \tilde{p}(0)) - \int_{V_1} ((p(1) + \tilde{p}(0))q_1 -$$
$$- (p(0) + \tilde{p}(1))q_0)dV = (p(0) + \tilde{p}(1)) - \int_{V_0} ((p(0) +$$
$$+ \tilde{p}(1))q_0 - (p(1) + \tilde{p}(0))q_1)dV. \quad (13)$$

Из выражения (13) следует, что для минимизации $p_{\text{д}}$ необходимо максимизировать входящие в его состав интегралы. Это, в свою очередь, приводит к тому, что области V_1 и V_0 следует выбирать таким образом, чтобы подынтегральные выражения были положительными. Следовательно, необходимо, чтобы одновременно выполнялись два взаимоисключающих неравенства

$$(p(1) + \tilde{p}(0))q_1 > (p(0) + \tilde{p}(1))q_0$$
$$\text{и } (p(0) + \tilde{p}(1))q_0 > (p(1) + \tilde{p}(0))q_1. \quad (14)$$

Для рассматриваемого случая наилучшая ситуация состоит в обеспечении равенства

$$(p(1) + \tilde{p}(0))q_1 = (p(0) + \tilde{p}(1))q_0. \quad (15)$$

Тогда условием выбора границы между областями V_1 и V_0 является обеспечение соотношения

$$\frac{q_1}{q_2} = \frac{p(2) + \tilde{p}(1)}{p(1) + \tilde{p}(2)}. \quad (16)$$

Применительно к сигналам ОФМ введем параметры r_1 и r_0, характеризующие расстояния от сигналов S_1 и S_0 до принятого сигнального вектора X. Тогда в случае нормального распределения многомерных плотностей вероятностей при их единичной дисперсии имеем:

$$q(r) = \frac{1}{\sqrt{2\pi}} \exp(-r^2/2). \quad (17)$$

В этом случае отношение (16) можно представить в виде

$$\frac{q_1}{q_0} = \exp\frac{\left|r_0^2 - r_1^2\right|}{2}, \; |r_0^2 - r_1^2| = 2\ln\frac{p(0) + \tilde{p}(1)}{p(1) + \tilde{p}(0)}. \quad (18)$$

Равенство (18) представляет уравнение гиперплоскости, перпендикулярной вектору $\Delta S = |S_0 - S_1|$ и пересекающей его в точке, отстоящей от середины на величину Δl, которая в конечном итоге будет определена как

$$\Delta l = \frac{1}{d} \ln\frac{p(0) + \tilde{p}(1)}{p(1) + \tilde{p}(0)}. \quad (19)$$

Таким образом, граница между областями принятия решений при демодуляции сдвигается от середины между векторами S_1 и S_0 на величину, определяемую выражением (19). Особенность указанного выражения в том, что оно определяется не только априорными вероятностями передачи сигналов, но и зависит от апостериорных вероятностей их приема. Поскольку априорное задание вероятностей передачи сигналов, как правило, не представляется возможным, то принимают гипотезу о выполнении равенства

$$p(1) = p(0) = 0{,}5. \quad (20)$$

В то же время использование данной гипотезы во многих случаях оправдано, в частности, в системах с рандомизацией информации или ее криптографической защитой. В таких системах вероятность появления логических символов "1" и "0" на выходе демодулятора приближается к 0,5.

В этом случае границы между собственными областями будут определяться сдвигами [3]:

а) для идеального приемника "без памяти":

$$\Delta l = \frac{1}{d} \ln\frac{0{,}5}{0{,}5} = 0;$$

б) для идеального приемника "с памятью":

$$\Delta l = \frac{1}{d} \ln\frac{1 + \tilde{p}(1)}{1 + 2\tilde{p}(0)}.$$

Предположим, что демодулятор имеет возможность хранить в памяти N сигналов на своих входе и выходе. В этом случае решающее устройство признает за переданный сигнал тот, который удовлетворяет условию

$$\begin{cases} X_1 \equiv S_0, \text{ если } X_1/S_0 \geqslant -\Delta l; \\ X_1 \equiv S_1, \text{ если } X_1/S_1 < -\Delta l, \end{cases} \quad (21)$$

где $\Delta l = 0 \; \forall i = \overline{1, N}$.

Далее, на основе принятых в соответствии с правилом (21) сигналов вычисляются апостериорные частости $\tilde{p}(1)$ и $\tilde{p}(0)$, а для каждого из N хранящихся в памяти сигналов повторно выносится решение:

$$\begin{cases} X_1 \equiv S_0, \text{ если } X_1/S_0 \geqslant -\Delta l; \\ X_1 \equiv S_1, \text{ если } X_1/S_1 < -\Delta l, \end{cases} \quad (22)$$

где $\Delta l = \frac{1}{d} \ln\frac{1 + \tilde{p}(1)}{1 + 2\tilde{p}(0)} \; \forall i = \overline{1, N}$.

Именно решение (22) выдается получателю.

После демодуляции множества из N сигналов указанный процесс повторяется. Причем решающая функция (22), т. е. граница между собственными областями, должна корректироваться для каждой новой серии по результатам значений $\tilde{p}(1)$ и $\tilde{p}(0)$, полученным на предыдущем этапе демодуляции.

Алгоритм демодуляции ОФМ-сигналов

Поскольку основным критерием выбора порога принятия решения является статистика появления "единиц" и "нулей" в выходной последовательности демодулятора, то очевидно, что судить о воздействии помех на канал передачи можно лишь в том случае, если априорно известно их соотношение в передающей последовательности. Поэтому рассмотренный подход предпочтителен для источников, использующих рандомизирующие устройства. К таковым следует отнести перемежители и криптопреобразователи, которые обеспечивают равновероятностное появление "единиц" и "нулей", обеспечивая так называемое свойство "баланса". Таким образом, нарушение "баланса" в принимаемой последовательности на выходе демодулятора будет свидетельствовать не об изменении свойств источника, а о воздействии помехи на канал связи.

Согласно [4], свойство "баланса" будет проявляться на выборке не менее 200 символов, что удовлетворяет требованиям вычисления статистических оценок. Следовательно, контроль соблюдения свойства "баланса" в принимаемой последовательности позволяет обнаруживать ошибки. А изменение порога принятия решения о принятом символе при демодуляции сигнала (в случае нарушения свойства "баланса") позволяет исправлять эти ошибки.

В связи с этим основные этапы алгоритма демодуляции сигналов ОФМ с адаптивным порогом принятия решения будут следующими.

Этап 1. Принимают аналоговый сигнал ОФМ $S(t)$, фильтруют и выравнивают его амплитуду, получая восстановленный сигнал $S_c(t)$.

Этап 2. Генерируют опорное колебание $S_0(t)$ с частотой сигнала $S_c(t)$.

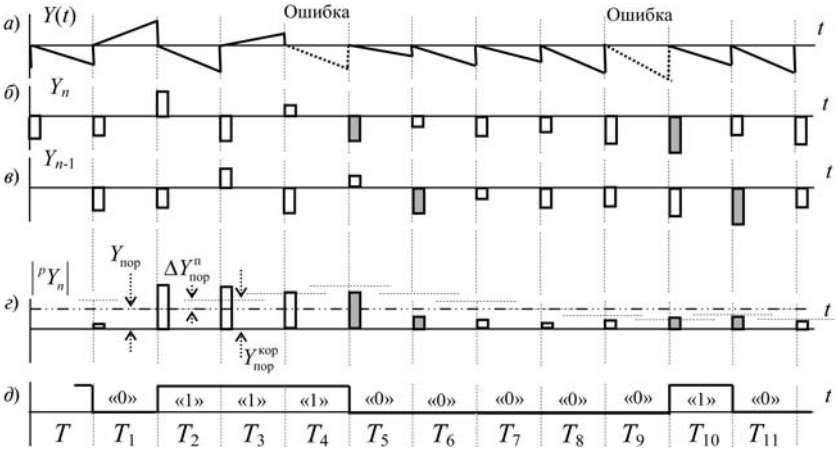
Этап 3. Вычисляют корреляционную функцию $Y(t) = S_c(t)S_0(t)$ между опорным колебанием $S_0(t)$ и восстановленным сигналом $S_c(t)$.

Этап 4. Интегрируют корреляционную функцию $Y(t)$ последовательно на временных интерва-

лах длительностью T, по окончании которых фиксируют ее значение Y_n. Здесь $n = 1, 2, ...$ — текущее значение временного интервала, равное длительности информационного символа.

Этап 5. Вычисляют модуль разницы $|^p Y_n|$ значений корреляционных функций Y_n и Y_{n-1} соответственно на n-м и на $(n-1)$-м временных интервалах T.

Этап 6. Полученное значение модуля разницы $|^p Y_n|$ сравнивают с пороговым значением корреляционной функции $Y_{\text{пор}}$ и при выполнении условия $|^p Y_n| > Y_{\text{пор}}$ присваивают принятому информационному элементу значение "единицы", в противном случае — "нуля".



Принцип демодуляции ОФМ-сигналов в условиях помех, вносимых каналом, с адаптацией порога принятия решения

Особенностью предложенного алгоритма является уточнение порогового значения корреляционной функции $Y_{\text{пор}}$ на длительности каждого демодулированного символа. Указанное значение $Y_{\text{пор}}$ уточняется следующим образом.

Предварительно формируют L-элементную последовательность с равным числом единичных и нулевых элементов в ней. Затем демодулированный на n-м временном интервале T информационный символ записывают первым элементом в L-элементную последовательность, сдвигая все ее элементы на один бит при сохранении общей длины. Соответственно, с поступлением очередного элемента последний элемент последовательности автоматически аннулируется. После этого корректируют пороговое значение $Y_{\text{пор}}$, для чего подсчитывают число "единиц" в измененной L-элементной последовательности. Затем рассчитывают отклонение порогового значения корреляционной функции $\Delta Y^{\text{п}}_{\text{пор}}$ от предварительно заданной величины $Y_{\text{пор}}$ по формуле

$$\Delta Y^{\text{п}}_{\text{пор}} = \frac{1}{4 Y_{\text{пор}}} \ln \frac{0{,}5L + k(1)}{1{,}5L - k(1)}, \qquad (23)$$

где $k(1)$ — число "единиц" в L — элементной последовательности.

И уже на основании результата вычисления (23) рассчитывают значение $Y^{\text{кор}}_{\text{пор}}$ путем алгебраического сложения по формуле

$$Y^{\text{кор}}_{\text{пор}} = Y_{\text{пор}} + \Delta Y^{\text{п}}_{\text{пор}}. \qquad (24)$$

И затем повторяют все этапы предложенного алгоритма, используя на этапе 6 вместо значения $Y_{\text{пор}}$ уточненную величину $Y^{\text{кор}}_{\text{пор}}$.

В качестве примера на рисунке представлены эпюры, поясняющие принцип демодуляции сигнала ОФМ в условиях помех в канале связи, приводящих к ошибкам при адаптивно изменяющемся пороговом значении корреляционной функции $Y^{\text{кор}}_{\text{пор}}$ [5].

Так, на рисунке *а* в корреляционной функции $Y(t)$ пунктирной линией показаны ошибки, которые возникли в результате помех в канале связи. Ошибки вычисления функции $Y(t)$, соответственно, приводят к ошибкам в Y_n и Y_{n-1}.

Ошибочные символы в Y_n и Y_{n-1} на рисунках *б* и *в* отмечены серым цветом. На рисунке *г* изображены пороговые уровни $|^p Y_n|$, получаемые на каждом интервале демодулированного символа. Здесь же показаны значения $\Delta Y^{\text{п}}_{\text{пор}}$, рассчитанные относительно $Y_{\text{пор}}$, в соответствии с выражением (23), и величина $Y^{\text{кор}}_{\text{пор}}$, получаемая по результатам суммы (22).

Анализ представленных результатов показывает следующее. Из-за канальных помех в демодулированной последовательности возникают четыре ошибочных символа на интервале T_5, T_6, T_{10} и T_{11} (см. рисунки *г* и *д*). Следует отметить, что применение в рассмотренной ситуации в качестве порога величины $Y_{\text{пор}}$ не исправляет ни одной из указанных ошибок. В то же время использование уточненной величины $Y^{\text{кор}}_{\text{пор}}$ ведет к тому, что три из четырех ошибок будет исправлено (на рисунке *г* пороговые значения $Y^{\text{кор}}_{\text{пор}}$ показаны линиями тонкого пунктира). Это подтверждает теоретические предположения.

В таблице представлены данные имитационного моделирования. Их анализ показывает, что наилучшие результаты обеспечиваются при 8 % ошибок в демодулированных символах. В этом случае применение адаптивного порога позволяет свести число ошибок до 3 %, что в совокупности с использованием помехоустойчивого кодирования, например кода Рида — Соломона, обеспечит уровень ошибок в канале порядка 10^{-7}.

Результаты моделирования при длине регистра $L = 200$ символов

Число ошибок	4	8	12	16	20	24
Число ошибок после порога $Y_{\text{пор}}$	4	8	12	16	20	24
Число ошибок после порога $Y^{\text{кор}}_{\text{пор}}$	2	4	5	6	10	16
Дисперсия ошибок после порога $Y^{\text{кор}}_{\text{пор}}$	±1	±1	±1	±2	±2	±2

Заключение

Теоретические результаты и данные имитационного моделирования указывают на правомерность выбранного направления по повышению эффективности демодуляции сигналов ОФМ. Между тем, рассмотренный подход требует дальнейшего изучения, в частности, для обоснования числового значения $\Delta Y^{\text{п}}_{\text{пор}}$ на каждом этапе уточнения порогового уровня.

Полученные результаты могут использоваться при модернизации модемов, использующих сигналы ОФМ, в частности, при передаче цифрового телевидения. Поскольку в стандарте DVB-T2 применяется временное перемежение при минимальной длине пакета в 16 200 символов, то на такой выборке будут выполняться все обоснованные в статье условия.

Список литературы

1. **Возенкрафт Дж., Джекобс И.** Теоретические основы техники связи. М.: Мир, 1969.
2. **Зюко А. Г., Фалько А. И., Панфилов И. П.** и др. Помехоустойчивость и эффективность систем передачи информации. М.: Радио и связь, 1985.
3. **Комарович В. Ф., Устинов А. А., Лобышев А. И.** Адаптивное различение двух сигналов в условиях воздействия окрашенных помех на основе анализа статистики выходной последовательности приемника // Информация и космос. 2007. № 4. С. 5—9.
4. **Мелентьев О. Г.** Теоретические аспекты передачи данных по каналам с группирующимися ошибками / Под ред. В. П. Шувалова. М.: Горячая линия — Телеком, 2007. 232 с.
5. **Дворников С. В., Дворников С. С., Устинов А. А.** и др. Способ демодуляции сигналов с относительной фазовой модуляцией на основе адаптивного порога. Патент РФ № 2454014 от 20.06.2012 г.

УДК 621.39

А. М. Чуднов, д-р техн. наук, проф.,
e-mail: Chudnow@yandex.ru,
А. В. Овчинников, адъюнкт,
e-mail: Ovchinnicow@yandex.ru
Военная академия связи им. С. М. Буденного,
г. Санкт-Петербург

Оптимизация порога стирания при приеме псевдослучайных сигналов

Рассмотрена задача оптимизации порога стирания при приеме псевдослучайных сигналов. Работа направлена на повышение эффективности функционирования системы передачи информации. Оптимизация порога стирания осуществляется с позиции максимизации потенциальной скорости передачи информации при различных вариантах ее оценки.

Ключевые слова: *передача информации, зона стирания, физический уровень, код Хемминга, код Голея*

Введение

Для передачи информации в условиях преднамеренных помех широкое применение нашли псевдослучайные сигналы с расширением, используемым для передачи полосы частот [1—4]. Как правило, для приема таких сигналов применяют корреляционный приемник Котельникова, реализующий преобразование принимаемого сигнала, при приеме которого выполняется сравнение с эталонными сигналами и преобразование его в ближайший эталонный сигнал [3—5]. Хорошо изучены методы анализа помехоустойчивости систем передачи информации с псевдослучайными сигналами, формируемыми на основе различных методов расширения спектра, проведен анализ показателей эффективности систем передачи информации (СПИ) с такими сигналами [1—4]. Однако вопросы повышения эффективности функционирования СПИ в условиях преднамеренных помех изучены недостаточно. Известные верхние границы помехоустойчивости СПИ с псевдослучайными сигналами [1] существенно (на 6 дБ) превышают соответствующие значения, достигаемые в известных и применяемых на практике системах.

Настоящая работа направлена на изучение вопросов повышения эффективности функционирования СПИ с псевдослучайными сигналами за счет использования в приемнике сигналов режима стирания ненадежно принятых символов с последующим исправлением стертых и ошибочно принятых символов внешним кодом. С учетом использования на более высоком уровне архитектуры СПИ — алгоритмов кодирования/декодирования оптимизация порога стирания осуществляется с позиции максимизации потенциальной скорости передачи информации при различных вариантах ее оценки. Обоснованием используемой модели дискретного канала связи является возможность симметрирования и обеспечения независимости ошибок в кодовых блоках известными способами псевдослучайных преобразований данных при их передаче и приеме.

Постановка задачи

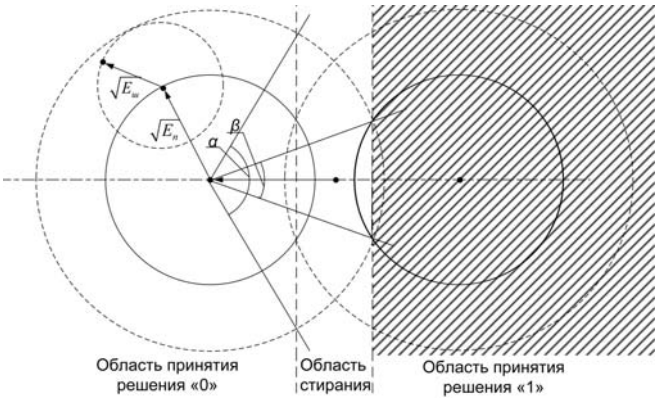
Рассматривается ε-оптимальная СПИ, в которой передаваемые двоичные сигналы при передаче и приеме обрабатываются взаимообратными *n*-мерными псевдослучайными изометрическими преобразованиями, нормализующими воздействующие на сигнал помехи [1]. Как известно [1—4], зависи-

мость вероятности ошибочного приема сигналов от отношения мощности сигнала к суммарной мощности помехи $\mathcal{P}_с/\mathcal{P}_п$ при достаточно больших значениях n определяется выражением:

$$p_{ош} = 1 - \mathcal{F}(h), \qquad (1)$$

где $\mathcal{F}(\cdot)$ — интеграл вероятностей, $h = \sqrt{n\mathcal{P}_с/\mathcal{P}_п}$.

Особенностью изучаемой в работе ситуации является то, что в соответствии с постановкой задачи в приемнике СПИ предусмотрена зона стирания ненадежных символов (так называемое "мягкое" решающее правило приемника), которая задается параметром ρ — порогом стирания. Принцип работы приемника с "мягким" решающим правилом иллюстрируется рисунком, где $E_п$ — энергия помехи, равная произведению мощности помехи $\mathcal{P}_п$ на период сигнала $T_с$; $E_ш$ — энергия шума, равная спектральной плотности сигнала. На рисунке видны механизмы появления ошибок и формирования приемником сигнала "стирание". Пусть s_i — переданный сигнал; v — воздействующая в канале помеха; $u = s_i + v$ — сигнал на входе приемника, $r(u, s_i)$ — оператор сигнала на входе приемника. При фиксированном пороге стирания ρ приемник по принятому сигналу u выдает на уровень кодирования СПИ решение о переданном символе i при условии $r(u, s_i) < ρ$, в противном случае сигнал u попадает в зону стирания $D(ρ) = [u : r(u, s_i) \geqslant ρ]$ и на выход выдается сигнал "стирание". Кроме того, в работе рассматривается вариант и оценивается эффективность решающего правила, при котором порог стирания выбирается так, чтобы на длине приема кодового блока было фиксированное число стираний m. Оценка эффективности использования "мягкого" решения для конкретных кодов осуществляется по показателям, характеризующим достоверность передачи информации. Для определения потенциальной эффективности приемника с "мягким" решением используются показатели, характеризующие скорость передачи информации.



Принцип работы приемника с "мягким" решающим правилом ($E_ш$ — энергия шума; $E_п$ — энергия помехи)

Оценка выигрыша "мягкого" решения с фиксированным порогом стирания

При фиксированном пороге стирания вероятности ошибки и стирания соответственно будут равны

$$p_{ош} = 1 - \mathcal{F}((1 + ρ)h); \qquad (2)$$

$$p_{ст} = \mathcal{F}((1 + ρ)h) - \mathcal{F}((1 - ρ)h). \qquad (3)$$

Достаточную для использования в инженерных расчетах точность дает приближение формулы (3) для $ρ < 0{,}3$, тогда оценка $p_{ст}$ будет иметь вид:

$$p_{ст} \approx \sqrt{\frac{2}{\pi}}\, ρh \exp(h^2/2). \qquad (4)$$

Рассмотрим вопрос определения оптимального порога стирания $ρ_0$ и соответствующего этому порогу показателя эффективности $Q(p_{ош}, p_{ст})$ СПИ из условия:

$$Q(p_{ош}, p_{ст}) \Rightarrow \max_{ρ}, \qquad (5)$$

где для СПИ с определенными кодами в качестве зависимостей $Q(p_{ош}, p_{ст})$ рассматривается вероятность $P_{пр}(n, d)$ правильного декодирования кода с длиной блока n и кодовым расстоянием d, а при определении потенциальной эффективности приемника с "мягким" решением используется показатель скорости передачи информации $R(p_{ош}, p_{ст})$ при $n \to \infty$.

Полагается, что распределение вероятностей на множестве последовательностей символов длины n подчиняется закону независимых испытаний, что может быть обеспечено путем перемежения символов кодовых блоков. При этом вероятность появления в блоке (длины n) i стираний и j ошибок равна

$$P(i, j|ρ) = \frac{n!}{(n - i - j)!\,i!\,j!}\, p_{ст}^{\,i}\, p_{ош}^{\,j}\, (1 - p_{ст} - p_{ст})^{n - i - j}. \qquad (6)$$

При использовании кода с параметрами (n, k, d) (k — число информационных символов) вероятность правильного декодирования кодового блока определяется выражением

$$P_{пр}(n, d|ρ) = \sum_{i + 2j < d} P(i, j|ρ). \qquad (7)$$

Рассчитанные для примера по формулам (6), (7) значения вероятности ошибочного декодирования $P_{ош}(n, d|ρ) = 1 - P_{пр}(n, d|ρ)$ (8,4,4)-кода Хемминга ($P_{ош1}$) и (24, 12, 8)-кода Голея ($P_{ош2}$) приведены в таблице.

Результаты расчетов показывают, что использование "мягкого" решения в приемнике СПИ в рассмотренных случаях повышает эффективность ее функционирования. При этом оптимальный порог стирания составляет $ρ = 0{,}15...0{,}20$, а энергетический выигрыш достигает 1,2...1,5 дБ.

Вместе с тем, при использовании длинных кодов эффективность решающего правила с фиксированным порогом стирания ненадежно принятых символов снижается. Асимптотическая оценка эффективности "мягкого" решающего правила в системах с ис-

h	ρ	$p_{ст}$	$p_{ош}$	$P_{ош1}$	$P_{ош2}$
2	0	0	$2{,}3 \cdot 10^{-2}$	$1{,}3 \cdot 10^{-2}$	$1{,}9 \cdot 10^{-3}$
2	0,2	$4{,}7 \cdot 10^{-2}$	$8{,}1 \cdot 10^{-3}$	$4{,}5 \cdot 10^{-3}$	$5{,}9 \cdot 10^{-4}$
3	0	0	$1{,}3 \cdot 10^{-3}$	$4{,}8 \cdot 10^{-5}$	$3{,}0 \cdot 10^{-8}$
3	0,2	$7{,}9 \cdot 10^{-3}$	$1{,}5 \cdot 10^{-4}$	$2{,}5 \cdot 10^{-6}$	$3{,}4 \cdot 10^{-10}$
4	0	0	$3{,}0 \cdot 10^{-5}$	$2{,}5 \cdot 10^{-8}$	—
4	0,2	$6{,}6 \cdot 10^{-4}$	$7{,}5 \cdot 10^{-7}$	$8{,}3 \cdot 10^{-11}$	—

правлением ошибок и стираний может быть получена на основе асимптотических границ скорости передачи кодов: для верхней границы Хемминга

$$R \leqslant R_H = p\log p + (1 - p)\log(1 - p); \qquad (8)$$

для нижней границы Гилберта

$$R \geqslant R_G = 2p\log(2p) + (1 - 2p)\log(1 - 2p). \qquad (9)$$

Для определения в соответствии с этими границами скорости передачи информации системой с исправлением ошибок и стираний нужно положить

$$R(p_{ош}, p_{ст}) = \begin{cases} R \text{ при } p < 0, \\ 0 \text{ при } p \geqslant 0, \end{cases} \qquad (10)$$

где

$$p = 2p_{ош} + p_{ст}. \qquad (11)$$

Из выражений (7)—(10) видно, что максимум величины $R(p_{ош}, p_{ст})$ достигается при условии минимизации эквивалентной вероятности ошибки (11). Заменяя $p_{ош}$, $p_{ст}$ их значениями из (2), (3), в соответствии с (11) получим оптимизационную задачу:

$$\mathcal{F}((1 - \rho)h) + \mathcal{F}((1 + \rho)h)) \Rightarrow \max_{\rho}, \qquad (12)$$

которая, как нетрудно установить, имеет единственное решение: $\rho = 0$.

К аналогичному результату можно прийти на основе оптимизации пропускной способности дискретного канала связи со стиранием и ошибками:

$$C(p_{ош}, p_{ст}) = (1 - p_{ст})(1 - \log(1 - p_{ст})) + \\ + (1 - p_{ош} - p_{ст})\log(1 - p_{ош} - p_{ст}) + p_{ош}\log p_{ош}.$$

Так, проведенная в работе вычислительными методами оптимизация пропускной способности канала $C(p_{ош}, p_{ст})$ показала неэффективность "мягкого" решающего правила приемника в СПИ, реализующих скорость передачи, близкую к пропускной способности канала связи.

Оценка выигрыша "мягкого" решения с фиксированным числом стираний

При использовании "мягкого" решающего правила с фиксированным числом стираний приемник при приеме из канала физического уровня последовательности из n сигналов устанавливает такой порог стирания, при котором стертыми будут ровно i символов, что обеспечит декодирующему устройству режим работы с заданным (оптимальным) числом стираний. Таким образом, например, для кода Хемминга (8, 4, 4) можно реализовать следующие (s, t)-режимы декодирования (s — число исправляемых стираний, t — число исправляемых ошибок, доминируемые режимы исключены): (1,1), (3,0), а для ((24, 12, 8)-кода Голея: (1,3), (3,2), (5,1), (7,0).

Пусть $D^n(\rho) = D(\rho) \times D(\rho) \times \ldots D(\rho)$ (n раз), $\mathbf{u} = (u_1, u_2, \ldots, u_n)$ — последовательность принятых сигналов длины n; $I(u, \rho) = \mathrm{Ind}(u \in D(\rho))$ — индикатор вхождения сигнала $\mathbf{u}$ в зону стирания:

$$\rho(\mathbf{u}, m) = \mathrm{argmin}(\rho | \sum_i I(u_1, \rho) = m) \qquad (13)$$

— минимальное значение порога ρ, при котором число попаданий сигналов из последовательности $\mathbf{u}$ в зону стирания равно m.

Таким образом, при поступлении на вход приемника последовательности сигналов $\mathbf{u}$ в приемнике устанавливается порог стирания, равный $\rho(\mathbf{u}, m)$. При этом очевидно в кодовом блоке число стираний равно m, а вероятность j ошибок определяется формулой (2) при $\rho = \rho(\mathbf{u}, m)$.

Анализ выигрыша от использования "мягкого" решения методами машинного моделирования показал, что для рассмотренных выше кодов оптимальное число стираний $m = 1$, соответственно режимы декодирования (s, t) будут (1,1) для (8, 4, 4)-кода Хемминга и (3,2) для ((24, 12, 8)-кода Голея. Дополнительный (по отношению к предыдущему случаю) выигрыш составляет 0,4...0,6 дБ.

Вместе с тем, анализ соотношения (13) показывает, что при $n \to \infty$ выполняется условие $m \to \infty$, причем $\rho(\mathbf{u}, m) \to \rho$, т. е. в асимптотическом случае значение порога фиксируется на уровне, определенном для СПИ с заданным порогом стирания.

Заключение

Предложенная методика позволяет анализировать СПИ, в приемнике которой на физическом уровне предусмотрено стирание ненадежно принятых символов, выдаваемых на канальный уровень системы. На примерах анализа (8, 4, 4)-кода Хемминга и (24, 12, 8)-кода Голея определены оптимальные пороги стирания и оценена эффективность алгоритма приема с "мягким" решением. Установленное значение оптимального порога стирания составило $\rho = 0{,}15...0{,}20$, а оптимальное число стираний при их фиксации приемником определяется на основе решения рассмотренной оптимизационной задачи.

Вместе с тем, следует отметить, что эффективность использования "мягких" решений в системах с длинными кодовыми блоками снижается и асимптотически при $n \to \infty$ стремится к нулю.

Список литературы

1. **Чуднов А. М.** Теоретико-игровые задачи синтеза алгоритмов формирования и приема сигналов // Проблемы передачи информации. 1991. № 3. С. 57—65.
2. **Omura J. K., Levitt B. K.** Coded Error Probability Evaluation for Antijam Communication Systems // IEEE Trans. Commun. 1994. Vol. COM-30, no. 5. P. 896—903.
3. **Варакин Я. Е.** Системы связи с шумоподобными сигналами. М.: Радио и связь, 1990. 256 с.
4. **Simon M. K., Omura J. K., Scholtz R. A., Levitt B. K.** Spread Spectrum Communications Handbook, Revised Edition. New York: McGraw-Hill, Inc., 1994.
5. **Коржик В. И., Финк Л. М., Щелкунов К. Н.** Расчет помехоустойчивости систем передачи дискретных сообщений. М.: Радио и связь, 1981. 232 с.

УДК 004.93'11

П. А. Козловский, аспирант,
С. О. Старков, д-р физ.-мат. наук,
ИАТЭ НИЯУ "МИФИ", Обнинск,
e-mail: starkov@iate.obninsk.ru,
А. А. Тельных, кандидат физ.-мат. наук,
Институт прикладной физики РАН,
Нижний Новгород

Поиск нечетких дубликатов на основе бинарных шаблонов

Представлен алгоритм для поиска нечетких дубликатов на основе бинарных шаблонов. В предлагаемом методе проводится иерархическое разбиение изображения на части и подсчет бинарных характеристик данных частей. С помощью использования интегрального изображения достигается высокая скорость построения сигнатуры изображения, а побитовое сравнение сигнатур позволяет сделать вывод о близости сравниваемых изображений. Для оценки данного алгоритма приводится сравнение по скорости и качеству с алгоритмами Colorgrid и Surf.

Ключевые слова: *поиск нечетких дубликатов изображений, алгоритмы нечеткого сравнения, интегральное изображение*

Введение

Проблема поиска нечетких дубликатов изображений в последнее время становится очень актуальной. С увеличением количества цифровых изображений и возможностей для хранения информации представляется важным поиск в некотором смысле похожих изображений. Для решения этой задачи предлагается множество методов.

"Похожесть" изображений довольно сложно формализовать. Действительно, различные изображения разные люди могут посчитать как сходными, так и нет. Более того, это часто зависит от контекста.

Введем два понятия основных определений похожих изображений, или же *нечетких дубликатов*.

1. Если одно изображение можно получить из другого применением любой комбинации следующих преобразований: изменения контраста; малого масштабирования изображения; поворота на угол, кратный 90°; добавления малого шума, применения сжатия с потерей качества, то такие изображения считаются нечеткими дубликатами.

2. Если одно изображение можно получить из другого применением любой комбинации следующих преобразований: всех преобразований первого определения, малого изменения точки съемки (ракурса), поворота на произвольный угол в плоскости изображения, то такие изображения считаются нечеткими дубликатами.

Понятие малости может быть численно введено для каждого из преобразований, исходя из того, что нечеткие дубликаты должны без труда идентифицироваться человеком как одинаковые, полученные одно из другого.

Алгоритмы, направленные на решение задачи поиска нечетких дубликатов в первом определении, могут быть также устойчивы к добавлению рамок, водяных знаков и других локальных изменений в изображении. Обычно такие алгоритмы выделяют некие интегральные характеристики изображения.

Второе определение более широкое и полностью включает в себя первое. Оно может использоваться для поиска похожих кадров в видеопоследовательностях, поиска частично перекрывающихся изображений (построение панорам) и в других сходных задачах.

В данной работе предложен новый алгоритм, придерживающийся первого определения нечетких дубликатов, который назовем *алгоритм на основе бинарных шаблонов*. В этом алгоритме использовано иерархическое разбиение изображений для их нечеткого сравнения. Для оценки производительности этого алгоритма мы сравним его с реализацией алгоритма *Colorgrid*, рассмотренного, например, в работе [1]. Кроме того, мы сравним его с популярным алгоритмом *Surf* [2], придерживающимся второго определения нечетких дубликатов.

Как правило, в любом алгоритме поиска нечетких дубликатов для каждого изображения выделяется некая *сигнатура*, т. е. относительно небольшая структура данных, связанная с данным изображением. По степени близости таких сигнатур и выносится суждение о том, являются ли изображения нечеткими дубликатами.

Использование сигнатур связано с тем, что брать для каждого сравнения исходное изображение, которое может быть весьма большим, неэффективно. Таким образом, выделение сигнатуры изображения является предобработкой перед поиском нечетких дубликатов.

Алгоритмы Surf и Colorgrid

Алгоритм Surf дает возможность для каждого объекта на изображении выделить "интересные" точки, позволяющие дать нам описание характерных черт объекта. Это описание, будучи получено из одного изображения, может быть потом использовано для нахождения того же объекта на других изображениях.

Сигнатурой изображения при использовании алгоритма Surf будем называть выделенный им набор ключевых точек и их дескрипторы. Подробное опи-

сание алгоритма нахождения ключевых точек и построения их дескрипторов можно найти в работе [2].

С помощью дескрипторов точки в двух наборах можно сопоставить, чтобы определить, дескрипторы каких точек близки. Такие пары точек можно считать "совпадающими" на двух изображениях.

Зная число совпавших пар точек, можно использовать различные подходы. В практической части был использован простой подход — число найденных пар точек должно быть не меньше 5 % от минимального числа точек в наборах. Альтернативой этому методу сравнения двух наборов ключевых точек может служить алгоритм RANSAC [3].

Основная проблема алгоритма Surf заключается в его низкой производительности. Нахождение ключевых точек в различных масштабах с последующим подсчетом их дескрипторов и сравнение таких наборов является вычислительно сложной задачей, выполнение которой может занимать секунды. Такой подход может быть необходим в некоторых задачах (таких, как распознавание объекта на изображении или склеивание панорам), но для задачи обнаружения нечетких дубликатов в первом определении можно использовать более простые, а значит быстрые алгоритмы. Как пример таких алгоритмов, рассмотрим алгоритм Colorgrid.

Алгоритм Colorgrid более близок к алгоритму бинарных шаблонов как по вычислительной сложности, так и по определению нечетких дубликатов. Общий ход получения сигнатуры изображения алгоритмом Colorgrid состоит из следующих шагов.

1. Создается уменьшенная копия изображения (в практической части бралось изображение размером 64 × 64 пикселя, размер можно регулировать). Изображение рассматривается в градациях серого, таким образом, информация отдельных цветовых каналов не учитывается.

2. Выполняется нормализация: интенсивность распределяется по всей шкале 0...255, таким образом, изображение становится более контрастным: самые темные участки будут абсолютно черными, а самые светлые — абсолютно белыми.

3. К полученному изображению применяется размытие Гаусса.

Полученные уменьшенные копии изображений (представляющие собой сигнатуры изображений по этому методу) сравнивали попиксельно, суммировали взятую по модулю разницу яркости каждой пары пикселей с одинаковыми координатами первого и второго изображений. Чтобы получить значение в процентах, найденное значение делили на максимально возможное, равное 256 × 64 × 64.

У алгоритма Colorgrid могут возникать проблемы при попытках различить определенные изображения, в особенности изображения с однородным фоном, на которых присутствуют различные, но сравнительно малые объекты в одних и тех же частях изображения. Например, изображение человека и изображение шкафа, сфотографированных на чистом белом фоне, с помощью алгоритма Colorgrid различить может быть весьма затруднительно, если каждый объект занимает меньше половины площади изображения. При построении сигнатуры оба объекта превратятся в кучки черных пикселей, и несовпадений за счет разной структуры объектов заметно не будет. За счет иерархического разбиения изображения с этим недостатком можно справиться с помощью алгоритма на основе бинарных шаблонов. Более подробная информация об алгоритме Colorgrid содержится в работе [1].

Алгоритм на основе бинарных шаблонов

Основная идея алгоритма состоит в том, что изображение иерархически разбивается на участки, для каждого из которых подсчитывается некоторый *код участка изображения*. Совокупность таких кодов для нескольких уровней иерархии и составляет сигнатуру изображения для этого алгоритма. Чем больше рассматриваемый уровень иерархии, тем более мелкие детали изображения рассматриваются, но тем больше влияние шума или искажений, которыми и отличаются нечеткие дубликаты.

Код участка изображения, используемый в данном алгоритме, представляет собой 9-битовый код, описывающий прямоугольный участок некоторого изображения, причем стороны этого прямоугольного участка параллельны сторонам самого изображения. Вычислим бинарный код для некоторого участка F с левым верхним углом в координатах (X_F, Y_F), шириной W_F и высотой H_F (рис. 1).

Среднюю яркость фрагмента F обозначим $<I_F>$. Разобьем фрагмент изображения F на девять равных частей f_0, ..., f_8 (рис. 1). Подсчитаем среднюю яркость $<I_{f_n}>$ каждой из частей фрагмента F.

Имея в распоряжении яркость всего фрагмента $<F_N>$ и яркость каждого из девяти участков $<f_n>$, входящих в данный фрагмент, будем кодировать

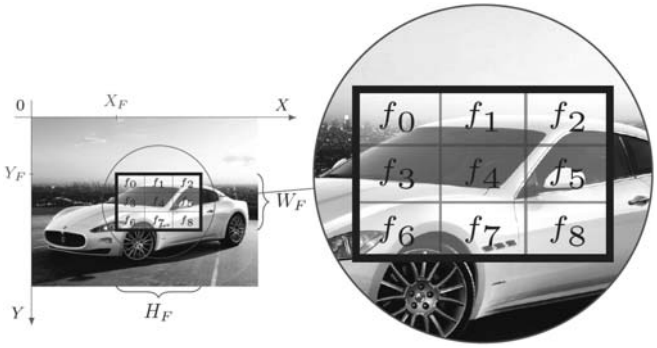


Рис. 1. Разбиение фрагмента на участки

каждый участок одним битом $c(f_n)$, получаемым следующим образом:

$$c(f_n) = \begin{cases} 1, \text{ если } < I_{f_n} > \text{ меньше } < I_F >, \\ 0, \text{ в противном случае.} \end{cases}$$

Таким образом, бинарный код фрагмента F состоит из девяти битов $c(f_n)$, соответствующих девяти частям этого фрагмента изображения, и может быть получен по следующей формуле:

$$BCode(F) = \sum_{n=0}^{8} c(f_n) \cdot 2^{8-n},$$

где $BCode(F)$ и есть бинарный код фрагмента F, и он может принимать значения от 1 до 511.

Данный бинарный код основан на *Census*-трансформации, описанной в работе [4]. Однако в нем имеются некоторые отличия:

- сравнение ведется со средней яркостью всей области и в преобразование включен центральный пиксель; соответственно, для кодирования области используются 9 бит, а не 8, как это сделано в *Census*-трансформации;
- используемое преобразование принципиально нелокально, в него вовлечены большое число пикселей и применяется оно сразу целиком к области изображения произвольного размера.

Преимущество нелокального подхода заключается в том, что с использованием интегрального изображения (оно более подробно описано в работе [5]) можно вычислять яркость участков изображения за четыре арифметических действия. Это позволяет выделять сигнатуру изображения чрезвычайно быстро.

Сигнатура всего изображения строится иерархически. Иерархическое кодовое представление любого изображения строится посредством рекурсивного разбиения изображения на прямоугольные фрагменты (по четыре фрагмента на каждом шаге рекурсии) и кодирования каждого полученного фрагмента посредством бинарного кода. В результате, каждый уровень L состоит из $N = 4^L$ бинарных кодов, а сигнатура представляет собой совокупность из нескольких уровней, бинарные коды которых записаны в виде последовательности, и являющейся сигнатурой.

Чтобы решить изначальную задачу — найти нечеткие дубликаты, нам необходимо ввести функцию похожести между сигнатурами изображений. Значение такой функции, близкое к единице, означает, что изображения, вероятно, являются нечеткими дубликатами, тогда как близкое к нулю значение указывает, что они совсем не похожи.

Введем расстояние между двумя бинарными кодами. За это расстояние возьмем *расстояние Хемминга* — число позиций, в которых соответствующие цифры двух двоичных слов одинаковой длины различны.

На каждом уровне L дерева сигнатуры содержится $N_L = 4^L$ бинарных кодов. Функция похожести между двумя сигнатурами на уровне L равна

$$S(C_1, C_2, L) = \frac{9N_L - \sum_{k=0}^{N_L - 1} d(C_1(L, k), C_2(L, k))}{9N_L},$$

где C_1 и C_2 — сравниваемые сигнатуры; L — уровень сравнения; $C(L, k)$ — k-й бинарный код уровня L сигнатуры C; d — функция расстояния между двумя бинарными кодами, т. е. расстояние Хемминга.

Таким образом, функция похожести между сигнатурами на уровне L равна отношению совпавших бит во всех бинарных кодах уровня L к общему числу бит в бинарных кодах этого уровня.

В качестве функции похожести между двумя сигнатурами возьмем наименьшее из всех значений функции похожести по каждому из доступных уровней:

$$S(C_1, C_2) = \min_{L_0 \ldots L_{\max}} (S(C_1, C_2, L)),$$

где $L_{\max}$ — максимальный из уровней сигнатур, который имеется как в сигнатуре C_1, так и в сигнатуре C_2.

Отметим, что можно изменить функцию похожести так, чтобы получить инвариантность к зеркальным отображениям по горизонтали и вертикали и поворотам на углы, кратные 90°.

Результаты сравнения и выводы

Для сравнения алгоритма на основе бинарных шаблонов и алгоритма Surf, реализация Surf была взята из открытой библиотеки JOpenSurf. Реализация алгоритма на основе бинарных шаблонов и алгоритма Colorgrid была собственной. При учете времени операций время загрузки изображения с жесткого диска и декодирования jpeg во внимание не принималось.

Набор изображений состоял из 199 широкоформатных изображений и 199 копий, уменьшенных до ширины 640 пикселей добавлением 15 % шума, результаты по числу ошибок приведены в таблице.

Ошибок второго рода у алгоритма Surf не было. Однако есть одна ошибка по нахождению ложного дубликата. Эта ошибка связана с изображениями, в которых имеются большие белые поля, тогда как содержательная часть изображения не так велика. Алгоритм на основе бинарных шаблонов справился

Число ошибок алгоритмов на первом тестовом наборе

Ошибки	Алгоритм на основе бинарных шаблонов	Surf
Число ошибок первого рода	4	1
Число ошибок второго рода	36	0

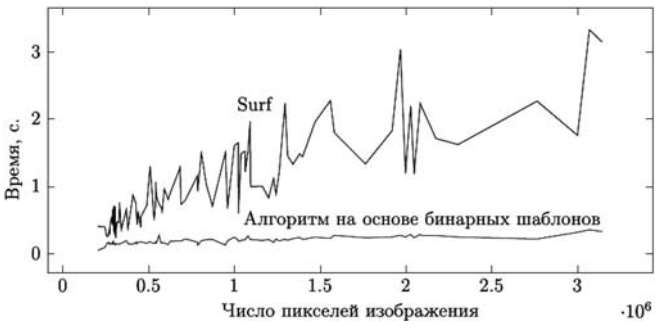


Рис. 2. Зависимость времени извлечения сигнатуры от числа пикселей изображения

с задачей в целом неплохо, хотя имеются ошибки в обе стороны, но их количество несущественно.

Зависимость времени извлечения соответствующих свойств от числа пикселей на изображении представлена на рис. 2. Из него следует, что извлечение сигнатуры для алгоритма на основе бинарных шаблонов в несколько раз быстрее, чем для Surf, и на данной выборке не превысило полсекунды. Также немаловажным является среднее время сравнения дескрипторов двух изображений. Для алгоритма на основе бинарных шаблонов оно составило в среднем 0,001179 с, тогда как для Surf — 0,352718 с. Таким образом, сравнение двух сигнатур для алгоритма на основе бинарных шаблонов быстрее на два порядка, что неудивительно, так как для сравнения изображений с помощью Surf требуются многократные ресурсозатратные операции по поиску близких векторов (их можно оптимизировать, например, с помощью *kd*-деревьев [6], однако это неслишком эффективно ввиду большой размерности векторов).

Для сравнения алгоритма на основе бинарных шаблонов и алгоритма Colorgrid было взято 350 изображений в формате *jpeg* довольно большого разрешения (порядка 1000 пикселей в ширину). Их масштабировали до некоторого размера, а потом сжимали с помощью *jpeg* с разной степенью сжатия.

В первой серии экспериментов качество *jpeg*-сжатия было выставлено в 0 (максимальное сжатие, максимальная потеря качества). В каждом из четырех тестов первой серии использовали такой тип сжатия, а размеры изображений были соответственно 640, 320, 210 и 105 пикселей в ширину.

Вторая и третья серии экспериментов были такими же, как и первая, но качество *jpeg* у них было соответственно 5 (среднее) и 12 (лучшее качество, слабое сжатие).

Таким образом, всего было 12 тестов (3 серии по 4 теста). В каждом тесте участвовало 700 изображений (350 оригинальных и 350 отмасштабированных в соответствующий размер с соответствующим качеством сжатия). В каждом тесте каждое изображение из тестового набора сравнивалось с каждым и подсчитывалось расстояние между ними по обоим алгоритмам. Для обоих алгоритмов вычислялись время выделения сигнатур и их сравнения, а также качественные характеристики.

Чтобы выделить качественные характеристики, использовали три подхода. В первом минимизировалось общее число ошибок как первого, так и второго рода. Находилась такая граница для функции расстояния между сигнатурами, при установлении которой суммарная ошибка первого и второго рода у данного алгоритма была минимальна на данном тесте. Соответственно, подсчитывалось число ошибок в обе стороны при данном лучшем "проходном балле".

Во втором подходе оценка качества проводилась по-другому: ошибки первого рода считались в 100 раз хуже, чем ошибки второго рода.

При третьем подходе, наоборот, ошибки второго рода считались в 100 раз хуже, чем первого.

По скорости реализация алгоритма Colorgrid несколько быстрее алгоритма на основе бинарных шаблонов. Это связано с тем, что он оперирует уменьшенной копией изображения, тогда как алгоритм на основе бинарных шаблонов строит интегральное изображения для оригинала. Однако это отличие не так существенно, так как оба алгоритма достаточно быстры и время их работы по сравнению с загрузкой изображения с диска и декодированию формата *jpeg* пренебрежимо мало.

Теперь проанализируем качество. На рис. 3 дано число ошибок каждого из алгоритмов при минимизации общего числа ошибок. По данным видно, что общее число ошибок алгоритмов на данной

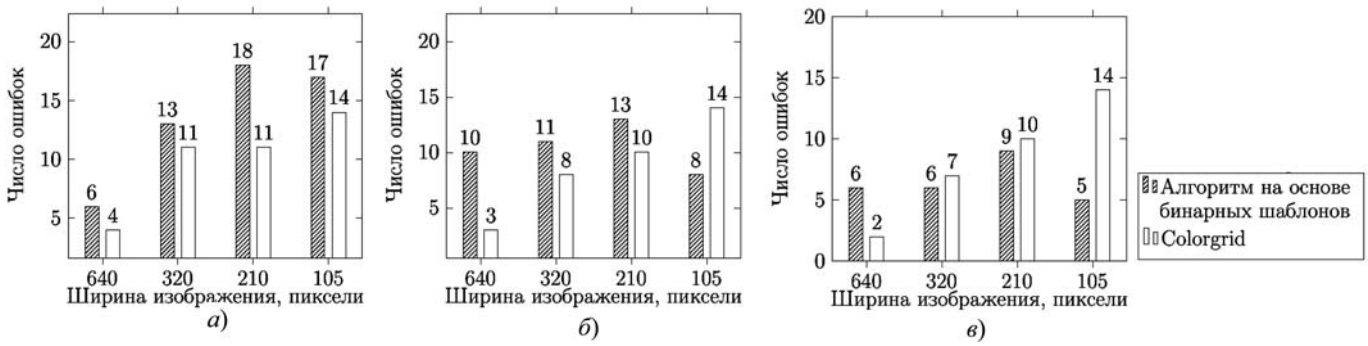


Рис. 3. Число ошибок при равнозначности ошибок первого и второго рода:
а — первая серия: максимальное сжатие; *б* — вторая серия: среднее сжатие; *в* — третья серия: минимальное сжатие

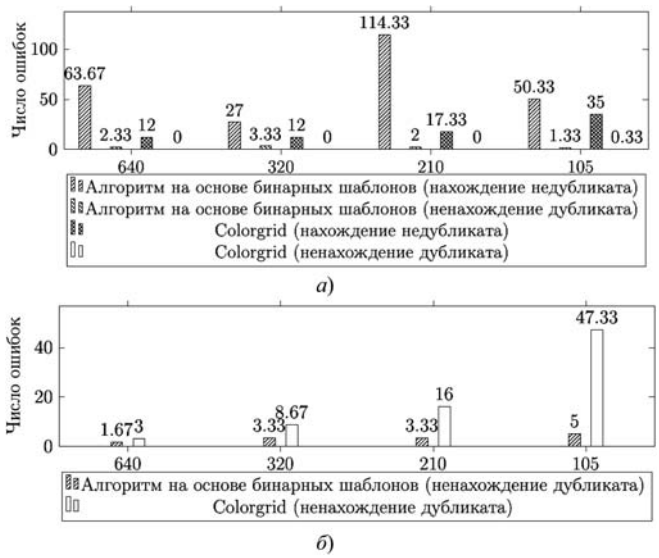


Рис. 4. Усредненные данные при минимизации разных ошибок: *а* — минимизация ошибки ненахождения дубликатов; *б* — минимизация ложного детектирования дубликатов

выборке сравнимо и колеблется в зависимости от размера и степени сжатия. В целом, число ошибок у алгоритма Colorgrid в этом случае незначительно ниже.

На рис. 4, *а* изображены гистограммы числа ошибок при условии, что ложноотрицательные в 100 раз менее предпочтительны, чем ложноположительные. Данные усреднены по всем сериям. Из диаграммы видно, что с этой задачей лучше справляется алгоритм Colorgrid. С его помощью можно практически избежать ложноотрицательных срабатываний. Алгоритм на основе бинарных шаблонов в этом смысле ему уступает.

На рис. 4, *б* изображены соотношения числа ошибок алгоритмов в случае, если ложноположительная ошибка считается в 100 раз хуже ложноотрицательной. В использованных алгоритмах на данной выборке при этом число ложноположительных ошибок было нулевым, так что мы приведем лишь число ложноотрицательных. Здесь алгоритм на основе бинарных шаблонов показывает существенно лучшие результаты, чем алгоритм Colorgrid, с помощью которого затруднительно различить некоторые изображения, особенно с однородным фоном и различными объектами в центрах обоих изображений.

Заключение

Проведенный сравнительный анализ позволяет сделать следующие выводы.

Алгоритм Surf в целом показывает весьма интересные результаты в задачах поиска нечетких дубликатов. Изменения масштаба, поворот, смена освещения и ракурса — он в той или иной степени устойчив ко всем этим преобразованиям. Однако за такую универсальность приходится платить. Во-первых, большое время занимает извлечение особенностей изображения, во-вторых, их сравнение тоже является нетривиальной задачей. Кроме того, велик размер хранимых данных для одного изображения. В задачах, где все эти возможности не нужны, Surf является слишком громоздким. Здесь можно применять более эффективные и быстрые методы, такие как алгоритм на основе бинарных шаблонов или алгоритм Colorgrid. Если сравнивать последние два метода, то алгоритм Colorgrid имеет преимущество в случае, когда мы хотим минимизировать число пропущенных дубликатов. Этот метод хорошо подходит для того, чтобы при поиске дубликата конкретного изображения отсекать некоторое множество изображений, не являющихся дубликатами, и оставлять меньший набор для более полного анализа.

В том случае, когда мы хотим находить исключительно дубликаты и минимум "мусора", алгоритм на основе бинарных шаблонов дает лучшие результаты и является более предпочтительным.

Авторы благодарны Ю. Д. Калафати за полезные обсуждения.

Статья написана в рамках выполнения НИР по Государственному контракту № 14.514.11.4022 Минобрнауки.

Список литературы:

1. **Кисель Я.** Алгоритм поиска нечетких дубликатов в коллекции изображений // Труды РОМИП 2007—2008 (Российский семинар по оценке методов информационного поиска). 2008. С. 170—173.

2. **Bay H., Tuytelaars T., Van-Gool L.** SURF: Speeded up robust features. Computer Vision and Image Understanding. 2008. N 3. С. 346—359.

3. **Fischler M., Bolles R.** Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography // Commun. ACM. 1981 г. N 24. P. 381—395.

4. **Zabih R., Woodfill J.** Non-Parametric Local Transforms for Computing Visual Correspondence // Proc. Third European Conf. Computer Vision. 1994. P. 150—158.

5. **Viola P., Jones M.** Robust Real-time Object Detection // International Journal of Computer Vision, 2002.

6. **Silpa-Anan C., Hartley R.** Optimised KD-trees for fast image descriptor matching // Proc. of IEEE Conference on Computer Vision and Pattern Recognition. 2008. P. 1—8.

ДИСКУССИОННЫЙ КЛУБ

УДК 004.41

В. П. Воеводин, д-р физ.-мат. наук, гл. науч. сотр.,
e-mail: valery.voevodin@ihep.ru,
ГНЦ РФ Институт физики высоких энергий,
г. Протвино

Структура понятия надежности вычислительной системы

Надежность является обобщающим понятием, вокруг которого развивается множество связанных с ней теорий и исследований. С позиции программных систем определяется структура понятия надежности вычислительной системы и круг относящихся к надежности вопросов, которые могут потребовать рассмотрения и решения при проектировании больших информационно-вычислительных и управляющих программно-аппаратных комплексов. Указываются некоторые имеющие отношение к надежности вычислительных систем методики и технологии. Отмечаются связи основных структурных элементов надежности вычислительной системы с другими предметными областями.

Ключевые слова: *вычислительная система, программная система, надежность, безотказность, безопасность, сопровождаемость, защищенность*

Введение

Разработка и реализация технических систем осуществляется в целевом пространстве — стоимость, производительность и надежность. Часто возникает проблема размерности надежности вычислительных систем, которая менее понятна, чем размерность стоимости и производительности. Надежность в широком смысле является комплексной характеристикой рассматриваемого объекта, а для потребителя она является одним из основных показателей качества. Оценка надежности осуществляется множеством единичных/обобщенных количественных показателей, зависящим как от особенностей рассматриваемого объекта, так и от условий его эксплуатации, например, специальные показатели устойчивости к внешним воздействиям — помехозащищенность, виброустойчивость, термостойкость, влагонепроницаемость и т. д. Можно встретить разные общие определения надежности, которые не имеют принципиальных различий [1—3].

Вопросы надежности рассматриваются и в регламентирующих документах в отдельных предметных областях в разделах, относящихся к требованиям и управлению качеством, где определения ориентированы на объекты конкретной предметной области, например ГОСТ Р 51814.2—2001. Указанные определения надежности акцентируются на технических аппаратных объектах, а ее понятие охватывает безотказность, долговечность, ремонтопригодность и сохраняемость как базовые показатели надежности со своими размерностями, например, среднее время наработки на отказ, средний срок службы, среднее время восстановления, средний срок сохраняемости. Обзор современных систем стандартов, методов анализа и количественных оценок, относящихся к надежности, приводится в работе [4].

Вычислительная система и ее надежность

Общее определение надежности удовлетворяло информационно-вычислительным и управляющим системам (ВС) до тех пор, пока программное обеспечение (ПО) было второстепенной частью ВС. Рост значимости ПО привел к постепенному переосмысливанию и изменению самого понятия надежности ВС. Программные компоненты интеллектуально активны, т. е. способны анализировать текущее состояние системы и по результатам анализа переводить ее в новое состояние, обеспечивая широкий диапазон возможных применений.

При разработке больших программных систем понятие надежности адаптировалось с учетом особенностей именно ПО. Например, отказы ПО в основном есть результат дефектов проектирования; надежность ПО не является функцией времени; изменения условий внешней среды влияют только на входные данные, а не на программы; прогноз надежности ПО в первую очередь базируется на анализе человеческого фактора.

Расширение сферы применения ВС в разнообразных предметных областях, неразделимость ПО и аппаратуры в некоторых случаях (микропрограммные реализации, микроконтроллеры) приводят к появлению новых аспектов в вопросе определения понятия надежности ВС как программно-аппаратного комплекса. Одна и та же аппаратура, но с различающимся ПО может быть использована в совершенно разных предметных областях с принципиально различными требованиями к надежности ВС.

Функциональность ВС выражается посредством поставляемого внешней среде множества *услуг* или системных *сервисов*. Основные определения и таксономия надежности с ориентацией на поставку корректных услуг в современных вычислениях и коммуникациях приведены в [5].

В общем виде надежность ВС определяется через выполнение своей функциональности следующим образом:

- *качественная* часть — возможность поставлять услуги, которым можно доверять (система должна оправдывать доверие к себе);
- *количественная* часть — возможность избегать неприемлемой для ее внешней среды частоты и серьезности отказов (критерий оценки надежности в предоставлении множества услуг).

Невыполнение данных условий означает нарушение требований надежности и отказ в предоставлении корректного сервиса.

Появление в современном мире таких терминов, как кибервойна, кибертерроризм, киберпреступность делает необходимым понимание структуры понятия надежности именно ВС и разработку средств обеспечения должного уровня надежности. Понятие надежности по сущности является звеном, связывающим ВС с решением ряда проблем в различных предметных областях. Это понятие предоставляет также удобные средства для соотнесения разных, порой противоречивых интересов внутри единого концептуального каркаса.

Вычислительная система и ее среда

Любая система противопоставляется среде, с которой взаимодействует, а ВС, содержащая вычислительные средства, взаимодействует со средой посредством обмена информацией. Поставка корректных услуг вычислительной системой означает *своевременный* обмен *корректной* информацией со средой, что обеспечивает надежный коммуникационный/вычислительный/управляющий процесс.

ВС поставляет услугу в форме своевременного появления корректной информации на границе со средой. Сумма всей информации на границе определяет *внешнее состояние* ВС в конкретный момент времени. Корректность последовательности внешних состояний системы определяет уровень доверия поставляемому сервису. Таким образом, отказ системы в предоставлении сервиса есть некорректное внешнее состояние системы по информации или по времени ее появления (существенное требование систем реального времени).

Любая система состоит из взаимодействующих подсистем, которые, в свою очередь, состоят из своих подсистем и так далее до атомарных элементов. Следовательно, ВС имеет множество внутренних границ, появление информации на которых в определенные моменты времени говорит о корректности функционирования соответствующих подсистем и элементов. Информация на внутренних границах определяет *внутреннее состояние* ВС или состояние ее подсистем. Совокупность внешнего и внутреннего состояний формирует *полное состояние* ВС.

Поскольку обмен информацией между ВС и средой осуществляется в обоих направлениях, то возможны *обоюдные* воздействия. Поэтому при обсуждении надежности рассматриваются различные виды возможного воздействия как среды на ВС, так и ВС на среду.

Структурные элементы надежности

Надежность ВС есть интегральная концепция, включающая в себя совокупность трех групп характеристик:

- группа атрибутов (показателей) — определяет пути оценки надежности;
- группа угроз — определяет формы того, что может отрицательно воздействовать на надежность ВС;
- группа средств — задает направления к повышению надежности ВС.

В ***группу атрибутов*** надежности ВС входят следующие показатели:

- *готовность* — готовность системы предоставлять корректный сервис;
- *безотказность* — непрерывность в предоставлении корректного сервиса;
- *безопасность* — отсутствие недопустимых последствий для среды в результате функционирования системы;
- *конфиденциальность* — отсутствие несанкционированного раскрытия информации в ВС;
- *целостность* — отсутствие несанкционированных изменений в состоянии ВС;
- *сопровождаемость* — расширяет понятие ремонтопригодности и подразумевает возможность системы переносить модификации, восстановления и замещения аппаратных/программных платформ;
- *защищенность* — обобщенный показатель, который предполагает одновременное существование *готовности*, *конфиденциальности* и *целостности*.

К ***группе угроз*** надежности относятся следующие формы воздействия:

- первичные — *дефекты*, *сбои*;
- вторичные — *рассогласования*;
- конечные — *отказы*.

Рис. 1. Структура надежности ВС

Группу средств повышения надежности составляют направления:

- *предвидения* дефектов;
- *предотвращения* дефектов;
- *устранения* дефектов;
- *устойчивости* к дефектам.

Средства надежности предназначены для уменьшения числа видимых со стороны среды отказов в системном сервисе. Традиционно отказы ВС фиксируются во времени в целях *понимания* изменения их частоты и *оценки* эффективности средств надежности.

В результате определяется структурное дерево надежности вычислительной системы, которое представлено на рис. 1.

Атрибуты надежности

Готовность — часть показателя защищенности, единичный количественный показатель, акцентируется на предположении того, что возможны чередования периодов предоставления сервиса и простоев ВС, измеряется вероятностью предоставления корректных услуг в конкретный момент в заданном интервале времени.

Безотказность — количественный единичный показатель, который говорит о способности системы непрерывно предоставлять корректный сервис на протяжении некоторого времени, измеряется вероятностью безотказной работы системы в течение определенного интервала времени.

Безопасность — количественный единичный показатель, измеряется вероятностью того, что в заданный интервал времени не произойдет отказа системы, который может оказать воздействие на среду с катастрофическими последствиями.

Безопасность для аппаратной среды. Если среда ВС включает в себя исполнительные устройства и механизмы, то отказ в предоставлении сервиса (некорректное/несвоевременное изменение внешнего состояния системы) через эти устройства может привести к непредусмотренным изменениям среды с нежелательными последствиями. Пользуясь терминологией модели "Швейцарский сыр", можно сказать, что открываются "отверстия" от ВС в область промышленной безопасности для обсуждения вопросов аварий и их последствий. Здесь задачи надежности решаются через процедуры оценки рисков и управления рисками для среды.

Безопасность для пользователя. Если средой ВС является человек, то возможны отрицательные воздействия на него. Эти вопросы пока недостаточно проработаны и урегулированы, нет четкого определения безопасности и рисков воздействия ВС на человека. Но они тесно связаны с понятием надежности, и существует потребность в разработке средств обеспечения безопасности воздействия ВС на человека.

Есть два направления воздействия на человека. Во-первых, погружение пользователя в виртуальную реальность, оказывающую влияние на мироощущение и восприятия

человека. Во-вторых, смысловая составляющая хранимой и предоставляемой ему информации независимо от возраста и уровня образования — проблемы информационного воздействия на отдельную личность и группы людей.

Конфиденциальность — часть показателя защищенности, единичный количественный показатель меры, до которой система может гарантировать, что неавторизованный пользователь не сможет разобраться в защищенной информации.

Целостность — часть показателя защищенности, единичный количественный показатель, измеряется вероятностью того, что дефекты и атаки (внешние угрозы вторжения) не приведут к искажению состояния системы, т. е. вероятностью отсутствия несанкционированных изменений в полном состоянии системы.

Сопровождаемость — комплексный показатель системы, заключающийся в измерении возможности ВС переносить модификации, восстановления после отказов и замещения базовых платформ, она обеспечивается процессом *сопровождения* и включает следующие показатели:

- *модификации* — подстройка системы к изменениям среды, обновление и добавление системных функций;
- *восстановления* — устранение проявленных дефектов и последствий, обнаружение и устранение "дремлющих" дефектов;
- *замещения* — возможность замены аппаратной и системной платформы ВС при минимальных простоях сервиса; важнейшая характеристика для систем управления объектами, время жизни которых (десятки лет среда практически не изменяется) значительно превышает время жизни базовых подсистем ВС.

Защищенность — комплексный показатель готовности системы на предоставление корректного сервиса авторизованным пользователям при отсутствии несанкционированного доступа к информации или несанкционированного управления состоянием системы. Данный атрибут надежности тесно связан с задачами обеспечения *информационной безопасности* ВС, где оцениваются риски информационным ресурсам и потенциальные последствия. Значимость данного показателя растет с расширением глобального информационного пространства.

Угрозы надежности

Первичные угрозы. *Дефекты* — это внутренние изъяны ВС, которые могут быть внесены на любом этапе цикла жизни ВС как результат преднамеренных действий или недостаточной компетенции. Дефекты могут воздействовать на систему во время использования и привести к неприемлемо деградировавшей производительности или полному отказу поставлять определенный сервис. Возможные дефекты классифицируются по разным критериям, они могут зародиться внутри системы или возникнуть вне ее границ и распространиться в нее посредством взаимодействия. Дефект может быть явной или предполагаемой причиной рассогласования и называется *активизированным*, когда вызывает рассогласование, в противном случае он *дремлющий*. Зачастую дефект сначала вызывает рассогласование в состоянии сервиса отдельного компонента, являющегося частью внутреннего состояния ВС, поэтому нет немедленного воздействия на внешнее состояние.

Сбой — кратковременная самоустраняющаяся утрата работоспособности элемента, характерен для аппаратных средств.

Вторичные угрозы. *Рассогласование* — отклонение текущего внешнего состояния какой-либо подсистемы от корректного состояния по информации или времени ее появления. *Рассогласование* есть часть полного состояния ВС, которая в последующем может привести к отказу в сервисе. Рассогласование *обнаруженное*, если о его присутствии говорит *сообщение об ошибке* или *сигнал ошибки*. Рассогласования, которые присутствуют, но не обнаружены, являются *скрытыми*.

Конечные угрозы. *Отказ* — это переход от корректного внешнего состояния к некорректному по информации или времени, т. е. к невыполнению системой своей функции. Обратный переход есть *восстановление* функциональности ВС.

Мода отказа сервиса есть способ проявления отклонений во внешнем состоянии ВС. Моды отказа упорядочиваются по *уровням серьезности*, с которыми ассоциируется максимальная приемлемая вероятность появления отказа.

Процесс развития отказа сервиса ВС:

- по какой-то причине происходит активизация дефекта в компоненте;
- это вызывает рассогласование на внешней границе данного компонента;
- рассогласование может привести к отказу в сервисе этого компонента;
- отказ может активизировать дефекты в компонентах, с которыми взаимодействует данный компонент;
- процесс может распространяться до внешней границы, до отказа сервиса ВС.

Средства повышения надежности

Предвидение дефектов. *Предвидение* дефектов входит в решение задачи *прогнозирования отказов*, где широко используется метод FMEA (Potential Failure Mode and Effects Analysis) — индуктивный анализ потенциальных отказов системы с классификацией по вероятности появления и серьезности последствий.

Существуют методы прогнозирования аппаратных постепенных отказов на основе дрейфа значений определенных параметров, например, в связи с износом. Прогнозирование сбоев и внезапных аппаратных отказов может базироваться на оценках, использующих предыдущий опыт работы с подобными системами, и анализе логики схожих механизмов отказов.

Дефекты ПО должны быть выявлены и устранены до сдачи в эксплуатацию. *Предвидение программных* дефектов осуществляется на стадии тестирования с полным покрытием кода в целях выявления наличия дефектов в программной системе.

Предотвращение дефектов. *Предотвращение* дефектов — действия по выявлению и предупреждению возможности возникновения дефектов в ВС. Широко распространены такие средства, как классическая триада идентификация — аутентификация с одним или более факторами — авторизация, криптография, журнализация, аудиторское слежение, защитные экраны на внешних и внутренних границах ВС, антивирусы, системы предотвращения проникновения и др.

Устранение дефектов. *Устранение* дефектов — обсуждаются пути уменьшения числа и уровня серьезности дефектов, числа сбоев в работе критичных по безопасности системах. Устранение проводится во время разработки и во время использования. Цель *устранения* совместно с *предвидением* заключается в достижении уверенности в том, что спецификации надежности и функциональности адекватны, а система полностью им отвечает.

В фазе использования ремонт аппаратуры является частью средств устранения дефекта. Устранение дефектов в ПО осуществляется на стадии отладки по результатам тестирования, выявившего наличие дефектов.

Устойчивость к дефектам (отказоустойчивость). *Устойчивость* к дефектам предполагает разработку способов избежать отказов сервиса ВС в присутствии дефектов и сбоев. Нарушение надежности встречается, когда система отказывает чаще или серьезнее, чем приемлемо для потребителя. Как правило, задача решается посредством включения в систему преднамеренной аппаратной и/или программной избыточности, предназначенной для обнаружения рассогласований на внутренних границах системы и предотвращения их распространения до отказа системного сервиса.

Восстановление от рассогласований в отказоустойчивых системах может быть достигнуто *откатом*: вперед — после обнаружения рассогласования система фиксирует свое состояние и корректирует его так, чтобы продолжить работу; назад — система возвращает себя в корректное более раннее состояние и продолжает работу из этого состояния.

В ПО для обнаружения рассогласований используются контроль корректности состояния, сторожевые таймеры, таймауты и т. д. Избежать отказов позволяют специальные техники программирования: многовариантное программирование с голосованием, оборонительное программирование с попыткой управления всеми возможными состояниями, программирование с контролем электромагнитной совместимости, программирование с контролем иммунитета и др.

Заключение

Рассмотренные выше определения по большей части достаточно обобщены в целях покрытия всего диапазона ВС — от единичных микроконтроллеров до больших распределенных информационно-вычислительных и управляющих систем с потребителями, оперативным и обслуживающим персоналом.

В разных обстоятельствах акцентирование внимания и концентрация усилий осуществляются на различные свойства планируемых сервисов ВС, например:

- на достижение необходимого отклика в реальном времени;
- на вероятность получения требуемого результата;
- на возможность избежать отказов, которые могут нанести вред среде;
- на степень предотвращения преднамеренных вторжений.

Понятие надежности снабжает разработчиков ВС средствами подхода к решению задачи обеспечения подходящего баланса разных свойств и используемых техник. Связь базовых атрибутов надежности с внешними воздействиями представлена на рис. 2.

Помимо рассмотренных атрибутов могут быть определены вторичные, которые уточняют или специализируют первичные. Некоторые структурные элементы тесно связаны, например, восстановление (часть сопровождения) и отказоустойчивость. Отказоустойчивость отличается от сопровождения тем, что последнее предполагает вклад со стороны внешних для ВС агентов.

Варианты акцентирования на различные атрибуты надежности напрямую влияют на баланс техник, применяемых для достижения целевой надежности ВС. Неизбежен конфликт некоторых атрибутов, что приводит к поиску компромиссов. Безопасность и защищенность фокусируются на уклонении от специфических отказов и предполагают анализ и оценку соответствующих рисков.

Показатель замещения в атрибуте сопровождаемости влияет на ряд проектных решений и направлен на максимизацию возможного времени эксплуатации больших ВС без значимых дополнительных простоев, на минимизацию затрат при замене физически и морально устаревших базовых платформ. Во времена отсутствия общепризнанных стандартов и недостаточной поддержки совместимости и переносимости основное решение заключалось в приобретении достаточного запаса аппаратных элементов и эксплуатации ВС без кардинальных обновлений прикладного информационного и программного обеспечения. В этом случае срок эксплуатации ВС зависел от показателя сохраняемости резервных аппаратных средств. Появление обратно совместимых аппаратных платформ, операционных сред с общепризнанными стандартами интерфейсов программирования приложений, обеспечивающих переносимость на уровне исходного или исполняемого кода (например, POSIX, LSB), средств виртуализации определило и подходы к решению проблем замещения аппаратных и программных платформ.

Рис. 2. Базовые показатели надежности ВС и внешние воздействия

К примеру, первая система управления бустерного синхротрона ИФВЭ была создана на базе машин ЕС-1010, произведенных в 1977—1978 гг. и эксплуатировавшихся без замещения платформ до 1997 г. благодаря значительному резерву всех аппаратных узлов. В следующей системе управления с 1993 г. (начало разработки) по 2012 г. выполнено несколько замещений платформ. Одни и те же данные и прикладные программы неоднократно переносились, начиная с процессоров Intel 486, Alpha, Motorola 68K с операционными системами Red Hat Linux, DEC Unix, LynxOS и до эксплуатируемых сейчас в системе управления процессоров от Pentium 4 до Xeon с системами проекта Fedora (версии от 3 до 16). Некоторый опыт проведения замещений без остановки объекта управления изложен в [6].

На практике оценка надежности осуществляется анализом частоты и серьезности отказов, а также длительности простоев из-за неудовлетворительных значений атрибутов надежности, которые важны для данного применения. Такой анализ выполняется непосредственно по имевшим место отказам и простоям или косвенно через оценку соответствующих рисков.

Список литературы

1. **ГОСТ** 27.002—89 Надежность в технике. Основные понятия. Термины и определения.

2. **Большая** советская энциклопедия. М.: Советская энциклопедия. Т. 17. 1974.

3. **Большой** энциклопедический словарь. М.: Большая Российская энциклопедия. 2002.

4. **Струков А. В.** Анализ международных и российских стандартов в области надежности, риска и безопасности. URL: http://szma.com/standarts_analysis.pdf

5. **Avizienis A., Laprie J.-C., Randell B.** Dependability and its threats: a taxonomy // IFIP 18th World Computer Congress, Topical Sessions, Toulouse, France. 2004. P. 91—120.

6. **Kim L., Klimenkov E., Komarov V., Kuznetsov V., Voevodin V.** Experience with the new control system of IHEP accelerators complex // Proceedings of the ICALEPCS'2005, Switzerland, Geneva, 2005.

УДК 004.3

А. Б. Барский, д-р техн. наук, проф.,
e-mail: arkbarsk@mail.ru,
Нгуен Ван Лой, аспирант,
e-mail: loimiit@gmail.com,
МГУПС (МИИТ)

Информационно-справочная система "Многосерверная база данных с циркулирующими сегментами" на логической нейронной сети

На основе знаний, полученных при моделировании многосерверной базы данных с циркулирующими сегментами, строится информационно-справочная система. В основе системы лежит аппарат логических нейронных сетей. Система допускает неограниченное развитие, модификацию, использование нечетких данных.

Ключевые слова: *многосерверная база данных, циркуляция сегментов, интенсивность потока запросов, среднее время выполнения запроса, база знаний, логическая нейронная сеть*

Введение

В работе [1] на базе основных понятий [2, 3] о многосерверных базах данных с циркулирующими сегментами проведены исследования с помощью аналитической модели. Эти исследования носят предварительный характер. Их цель — показать, что высоко интенсивные потоки запросов к базе данных могут обслуживаться в приемлемое время, если базу данных превратить в многоканальную систему массового обслуживания. Каналами обслуживания являются серверы, к которым жестко подключены рабочие станции пользователей. Сегменты базы данных циркулируют между серверами, обеспечивая доступность всем пользователям. Установлено, что, несмотря на потери времени на ожидание "своего" сегмента, такая система становится эффективной именно при больших интенсивностях потока запросов. Она в целом, за счет разделения потока запросов, позволяет перенести критическую точку с пресловутым отношением $\lambda/\mu = 1$ (λ — интенсивность обращения к базе данных, μ — максимальная интенсивность обслуживания запросов к базе данных) на многократно большие значения интенсивности потока запросов, т. е. значительно увеличить пропускную способность системы и вероятность выполнения запросов.

Этот вывод подтвердился при проведении имитационного детерминированного моделирования со случайными сценариями следования запросов разной интенсивности.

Результаты моделирования образуют базу знаний[1], работа с которой особенно необходима на этапе проектирования системы. При отсутствии информации можно обратиться к модели. Однако требования оперативности исследования разных вариантов принимаемых решений, интерполяция этих решений внутри исследуемой области на основе логического вывода, обращение к нечетким данным требуют создания информационно-справочной системы. Такая система должна осуществлять быструю ассоциативную выборку по предполагаемым значениям параметров, легко развиваться на основе опыта эксплуатации и моделирования, легко подвергаться модификации и уточнению решений.

1. Имитационная модель многосерверной базы данных с циркулирующими сегментами

Целью моделирования является нахождение такого *оптимального соотношения между параметрами* многосерверной базы данных с циркулирующими сегментами, при котором минимизируется среднее время выполнения запроса:

$$T_{\text{вып}} \to \min. \qquad (1)$$

Дополнительный критерий качества характеризует пропускную способность системы — серверы плюс реализуемая на них база данных — с позиций вероятности $P(\lambda)$ выполнения запроса как функцию интенсивности потока:

$$P(\lambda) \to \max. \qquad (2)$$

Актуальность этого критерия существенно связана с успешно преодолеваемыми ограничениями по производительности сервера и скорости передачи данных. Несомненно, выполнение этого критерия должно быть проиллюстрировано при моделировании.

Конкретно задача моделирования заключается в нахождении среднего времени $T_{\text{вып}}$ выполнения *простых* запросов к базе данных (БД) как функции варьируемых параметров системы. Основными задаваемыми параметрами модели являются:

λ — интенсивность обращения к БД. Представляет собой интегральную характеристику, отражающую равную интенсивность потоков запросов пользователей, поступающих на каждый сервер;

n — число серверов $i = 0, ..., n - 1$, участвующих в хранении и обработке циркулирующих сегментов с частотой формирования запросов λ/n;

$t_{\text{вып}}$ — среднее "чистое" время выполнения единичного запроса к БД ($\mu = 1/t_{\text{вып}}$ — максимальная интенсивность обслуживания запросов к БД);

m — число независимых сегментов $j = 0, ..., m - 1$, на которое разбивается БД, с равновероятным обращением по каждому запросу;

[1] База знаний отличается от базы данных наличием процедуры логического вывода.

τ — длительность такта, определяемая временем однократного выполнения цикла обработки буфера запросов и временем ротации сегментов в процессе циркуляции.

На рис. 1 (см. третью сторону обложки) представлена схема информационного взаимодействия всех объектов в составе модели многосерверной БД с циркулирующими сегментами.

Главными элементами этой схемы, детально воспроизведенной для *i*-го сервера, являются очередь запросов ($ОЗ_i$), ожидающих обслуживания, и буфер сегментов ($БС_i$). Очередь запросов пополняется вновь поступающими запросами: реально — от пользователей рабочих станций, подключенных к серверу, при моделировании — от блока формирования потока запросов. В цикле обзора $ОЗ_i$ запросы, для которых в $БС_i$ есть необходимые сегменты БД, условно выполняются и исключаются. Их время выполнения фиксируется для усреднения.

Для плотной загрузки $ОЗ_i$ и сокращения поиска свободных регистров используется список свободных регистров ($ССР_i$) (на схеме не показан). Новый запрос записывается в $ОЗ_i$ по адресу из "головы" этого списка. Регистр, занимаемый выполненным запросом, дополняет $ССР_i$.

Представленная схема определяет алгоритм имитационной модели, который делится на две взаимодействующие части:

- алгоритм работы серверов, динамически пополняющих состав запросов, имитирующих их выполнение на поступивших сегментах и уничтожение выполненных запросов;
- алгоритм ротации сегментов.

Запросы, формируемые на входе сервера, могут сразу же анализироваться с точки зрения присутствия в памяти сервера требуемых сегментов, т. е. возможности немедленного выполнения.

Для приема сегментов от смежного сервера используется буфер-приемник, обеспечивающий промежуточное хранение поступивших сегментов. Этот буфер позволяет избежать уничтожения еще не отправленных сегментов при организации их циркуляции. С помощью этого буфера реализуется конвейерная обработка сегментов, так как его заполнение совместимо с обработкой сегментов, уже находящихся в буфере сегментов.

2. Некоторые результаты моделирования

Как и в работе [1], исследуем диапазон изменения интегрированной λ-характеристики (λ = 10, 20, ..., 160), рассчитанной для одного такта. Для сравнения с графиком *II* на рис. 2 [1] выберем значения τ = 0,1 (обусловлено максимальными возможностями ротации сегментов), n = 8, $t_{вып}$ = 0,02 (μ = 50; относительно одного такта μ = 5), n = 20.

Для этих значений определим ограниченный объем B буфера с учетом допустимого числа D запросов, выполняемых в сбалансированной системе за такт. Не вдаваясь в обоснование, положим B = 10, так как D = 5.

Найдем с помощью модели среднее значение времени выполнения и вероятность выполнения запросов к БД, обусловленную отказом в обслуживании запросов в связи с переполнением буфера.

Расчеты представлены в табл. 1.

Графики представлены на рис. 2 (кривая *I*) и 3.

Анализ кривой *I* на рис. 2 показывает, что более явно наблюдается экспоненциальный рост значения $T_{вып}$ с ростом λ. Это весьма просто объяснить:

1) возрастает объем вспомогательных работ по совокупному обслуживанию буфера запросов, что невозможно учесть при "теоретической" интерпретации процесса;

2) играет роль все то же отношение $\lambda/(n\mu)$, которое более быстро достигает единицы и начинает ее превышать. При этом среднее время выполнения не устремляется к бесконечности лишь потому, что возрастает интенсивность отказов в обслуживании запросов.

Это показывает анализ графика на рис. 3.

В целом, учитывая, что время выполнения запроса на рис. 2 задается в тактах, следует признать, что это же время, найденное в результате модели-

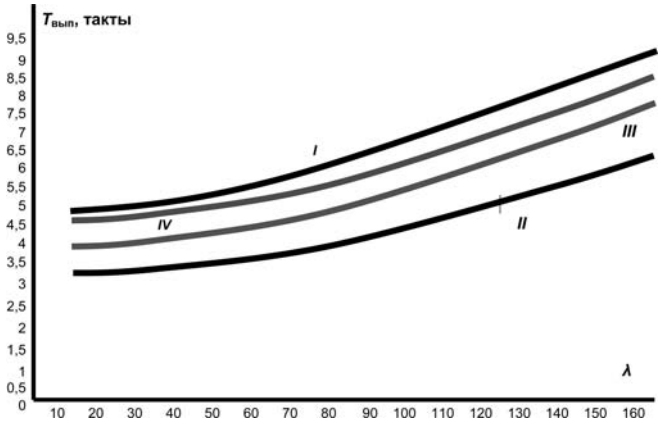


Рис. 2. Зависимость среднего времени выполнения запросов к БД от интенсивности их потока

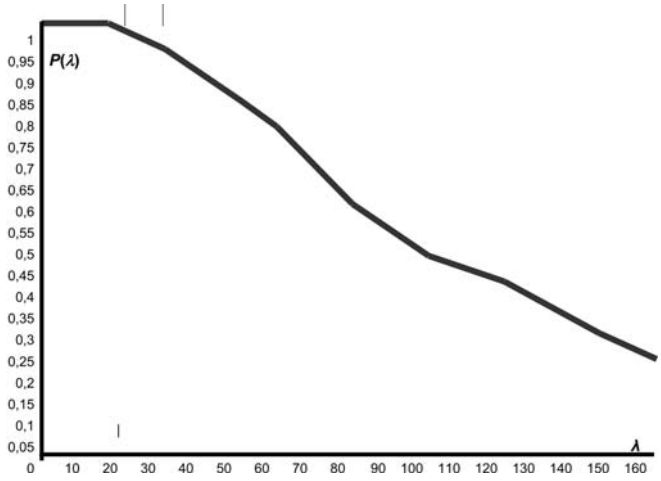


Рис. 3. Зависимость вероятности выполнения запросов к БД от интенсивности их потока для n = 8, m = 20, $t_{вып}$ = 0,02, τ = 0,1

Таблица 1

Обработка варианта $\tau = 0{,}1$, $n = 8$, $t_{вып} = 0{,}02$, $m = 20$

λ	10	20	30	50	60	80	90	100	120	140	160
$T_{вып}$, с	0,44	0,45	0,47	0,49	0,51	0,53	0,55	0,58	0,61	0,65	0,68
$P(\lambda)$	1	1	0,95	0,9	0,82	0,6	0,52	0,48	0,42	0,31	0,25

Таблица 2

Обработка варианта $\tau = 0{,}05$, $n = 8$, $t_{вып} = 0{,}005$, $m = 20$

λ	10	20	30	50	60	80	90	100	120	140	160
$T_{вып}$, с	0,16	0,17	0,18	0,2	0,21	0,23	0,25	0,255	0,27	0,3	0,33

Таблица 3

Обработка варианта $\tau = 0{,}05$, $n = 10$, $t_{вып} = 0{,}005$, $m = 24$

λ	10	20	30	50	60	80	90	100	120	140	160
$T_{вып}$, с	0,2	0,21	0,22	0,24	0,25	0,27	0,283	0,292	0,32	0,345	0,37

рования, не сильно отличается от "теоретического". Учитывая значительно меньшую сложность расчета кривой *II* на рис. 2 [1] с помощью аналитической модели, по результату имитационного моделирования можно выполнить калибровку, позволяющую приблизить график *I* на рис. 2 к кривой *II*, представленной в работе [1].

Резко сократим длительность такта, следующую из возможности быстрого обмена сегментами между серверами. Исследуем вариант $\tau = 0{,}05$, $t_{вып} = 0{,}005$, $n = 8$, $m = 20$, не учитывая ограничение ни на объем буфера, ни на производительность сервера. Таким образом, будем считать, что вероятность выполнения запросов равна единице.

Результаты моделирования представлены в табл. 2.

График представлен на рис. 2 кривой *II*.

Рассмотрим вариант $\tau = 0{,}05$, $t_{вып} = 0{,}005$, $n = 10$, $m = 24$. Результаты моделирования представлены в табл. 3.

График представлен на рис. 2 кривой *III*.

Наконец, кривая *IV* на этом же рисунке соответствует варианту $\tau = 0{,}05$, $t_{вып} = 0{,}005$, $n = 12$, $m = 24$.

Отметим, что увеличение числа m сегментов БД с увеличением числа n серверов способствует более частой передаче сегментов следующему серверу без задержки в случае их невостребованности на сервере, на который они поступили. При этом уменьшается время оборота сегментов и, следовательно, время их ожидания запросами. Это снижает среднее время выполнения запроса.

Увеличение числа серверов, хотя и увеличивает среднее время выполнения запроса, служит росту пропускной способности системы. Оно позволяет обслуживать большие потоки запросов, подключать большое число пользователей (клиентов) за счет организации многоканального обслуживания базы данных.

3. Построение информационно-справочной системы

Привлекательность метода логических нейронных сетей — в достижении высокой реальной производительности при решении задач (по сравнению с традиционными "алгоритмическими" методами), в возможности широкого распараллеливания (подобно мозговым процессам), в возможности создания простого и универсального интерфейса пользователя, допускающего экспериментирование, модификацию, смену специализации и т. д.

Следует отметить замечательную возможность нейросети — *табличную аппроксимацию функции многих переменных*, снабженную процедурой интерполяции для нахождения приближенного значения векторной функции по произвольному значению вектора-аргумента.

Информационной основой построения *обученной* нейронной сети (рис. 4) для строящейся информационно-справочной системы являются таблицы вида 1, 2, 3, построенные в процессе моделирования. Здесь используются семь таблиц. Можно дополнить каждую таблицу данными для недостающих значений λ (с шагом $\Delta\lambda = 10$), для чего необходимо сделать дополнительные расчеты.

Простейшая функция активации имеет вид

$$V := \frac{1}{k}\sum_i V_i;\ V := \mathit{if}\ V \geqslant h\ \mathit{then} V\ \mathit{else}\ 0,$$

(k — *число активных входов нейрона*). (3)

Здесь V_i — величина возбуждения рецептора, подаваемая на вход нейрона. Далее будет видно, что в нашем случае $k = 6$.

Максимально возбудившийся нейрон указывает на решение. Однако компоненты решения являются действительными числами, поэтому возможно усреднение каждой компоненты по нескольким возбудившимся нейронам. Так как вектор-результат содержит только две компоненты, из которых вторая — вероятность — не всегда актуальна (из-за практической возможности формирования большого буфера), то будем считать, что решением является вектор, состоящий из двух компонент — времени выполнения запроса и вероятности его выполнения.

Чтобы не загромождать рисунок, направление синапсических связей указано во вставках, а решения обозначены на "телах" нейронов парой искомых чисел.

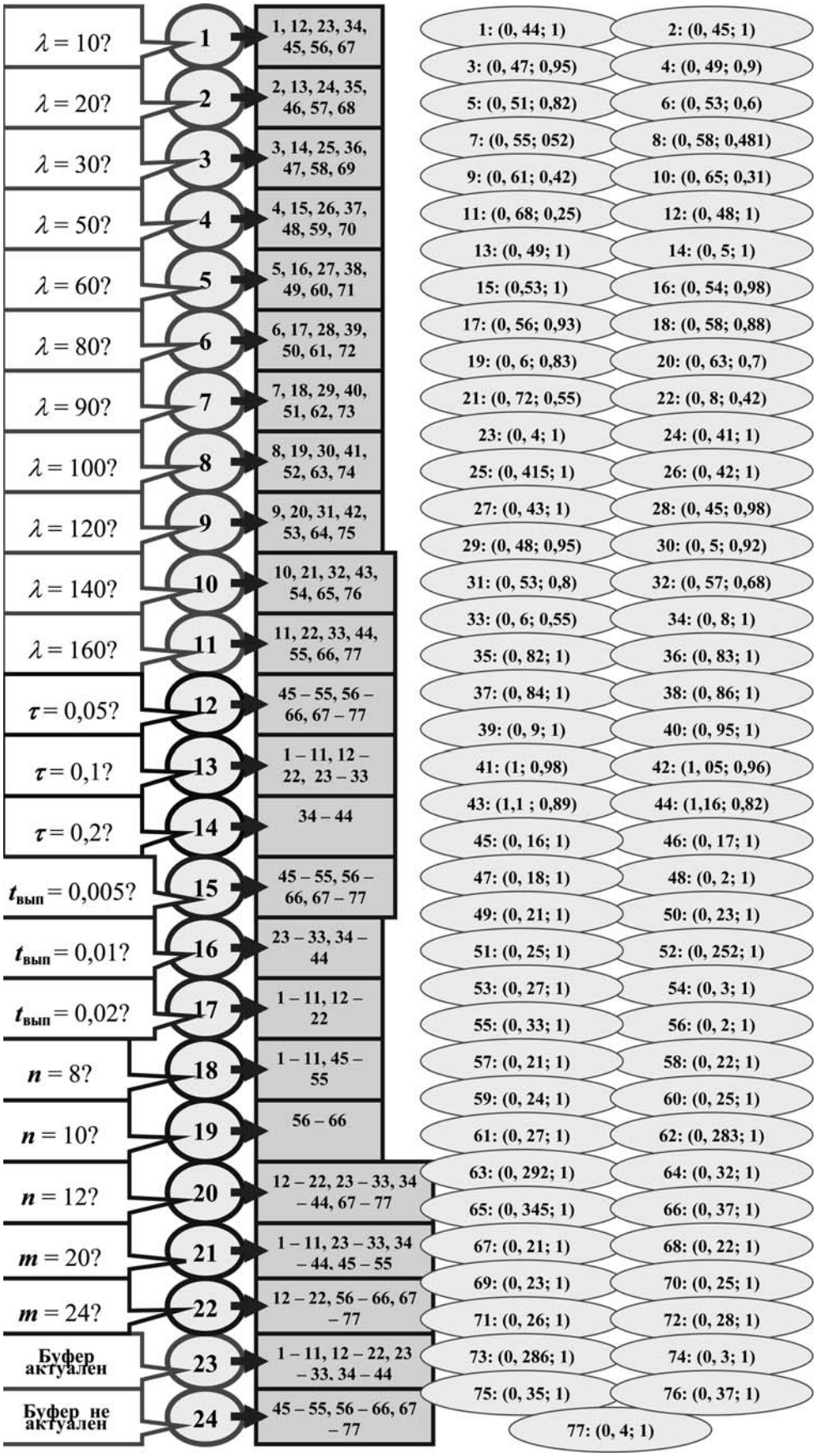


Рис. 4. Логическая нейронная сеть в основе информационно-справочной системы

Таблица 4

Структура матрицы следования для логической нейронной сети, реализующей информационно-справочную систему

Нейроны	Факторы															
	...	$\lambda = 140$	$\lambda = 160$	$\tau = 0,05$	$\tau = 0,1$	$\tau = 0,2$	$t_{вып} = 0,005$	$t_{вып} = 0,01$	$t_{вып} = 0,02$	$n = 8$	$n = 10$	$n = 12$	$m = 20$	$m = 24$	Буфер ограничен	Буфер неограничен
Пример		0,7	0,3	0,5	0,5		0,6	0,4			0,5	0,5		1		1
							. . . . .									
10		1			1				1	1			1		1	
11			1		1				1	1			1		1	
							. . . . .									
32		1			1			1				1	1		1	
33			1		1			1				1	1		1	
							. . . . .									
76		1		1			1					1		1		1
77			1	1			1					1		1		1

Напоминаем [3], что в рабочем режиме каждый рецептор возбуждается автоматически или "вручную" в соответствии с ответом на вопрос: "Какова достоверность того, что, например, значение λ, с которым обратился пользователь, совпадает с тем значением λ, за которым закреплен данный рецептор?"

Поскольку исследования многопроцессорных БД далеки от завершения, веса всех синапсических связей приняты равными единице. На данном этапе исследований мы не можем с достаточной достоверностью установить, какой из рассмотренных факторов и в какой степени влияет на получаемый результат. Это может быть выявлено по длительному опыту эксплуатации.

Изображение логической нейронной сети на рис. 4 приведено для наглядности. Обработка нейросети основана на ее представлении в виде матрицы следования [3].

Большое число используемых нейронов и связей не позволяет полностью представить всю матрицу следования. В табл. 4 показана такая матрица в сокращенном виде — для демонстрации метода ассоциативных вычислений. Она легко обрабатывается с помощью EXEL

Обработаем гипотезу о предполагаемых параметрах (значениях факторов) создающейся многосерверной базы данных с циркулирующими сегментами. Пусть $\lambda = 146$, $\tau = 0,075$, $t_{вып} = 0,007$, $n = 11$, $m = 12$, размер буфера не является актуальным. Возбуждение рецепторов указано в строке "Пример".

"Возбудим" рецепторы, разделив "единицу" обратно пропорционально расстоянию до двух близких рецепторов, закрепленных за каждым фактором: $V_{10} = 0,7$, $V_{11} = 0,3$, $V_{12} = V_{13} = 0,5$, $V_{15} = 0,6$, $V_{16} = 0,4$, $V_{19} = V_{20} = 0,5$, $V_{22} = 1$, $V_{24} = 1$.

Считаем функцию активации всех нейронов 1—77 скалярным умножением каждой его строки в матрице на строку "Пример". Полученные результаты делим на 6 (каждый нейрон имеет 6 входных связей) и сравниваем с порогом h.

В общем случае порог выбирается экспериментально, чтобы отсечь бесперспективные, но перегружающие расчеты, малые значения возбуждения. В данном случае представляется целесообразным принять $h = 0,6$. Под рассматриваемый пример выбраны строки матрицы следования, представленные в табл. 4.

Полученные значения возбуждения нейронов:

$$V_{10} = V_{11} = V_{32} = V_{33} = 0,\ V_{76} = 0,71,\ V_{77} = 0,65.$$

По другим, не приведенным в таблице строкам, возбуждения равны нулю.

Находим $t_{вып} = \dfrac{0,37 \cdot 0,71 + 0,4 \cdot 0,65}{0,71 + 0,65} = 0,38.$

Заключение

- На основе имитационного моделирования построен ряд таблиц и графиков, позволяющих определить зависимость среднего времени выполнения запросов и вероятности их выполнения от таких характеристик, как интенсивность потока запросов, длительность такта системы, "чистое" среднее время выполнения запроса, число серверов в системе, число сегментов БД, актуальность ограничения объема буфера.
- Объединение полученных при моделировании результатов в единую *базу знаний*, являющуюся информационной основой *системы принятия решений*, позволило построить *логическую нейронную сеть*, реализующую информационно-справочную систему "Многосерверная база данных с циркулирующими сегментами". Процедура ассоциативной выборки из базы знаний проводится с помощью *функции активации* нейрона на основе интерполяции и усреднения близких решений. Система обеспечивает высокую скорость доступа, расширяемость и модифицируемость.

Список литературы

1. **Барский А. Б., Нгуен Ван Лой.** Оценка среднего времени выполнения запроса к многосерверной базе данных с циркулирующими сегментами // Информационные технологии. 2013. № 3. С. 48—51.
2. **Барский А. Б.** Grid-вычисления. Организация, методы, планирование. Saarbrücken: LAP LAMBERT Academic Publishing. 2012.
3. **Барский А. Б.** Нейронные сети логического вывода. Курс лекций. Saarbrücken: LAP LAMBERT Academic Publishing. 2011.

УДК 004.032.26

А. И. Галушкин,
д-р техн. наук, проф., зам. зав. каф.,
Московский физико-технический институт
(государственный университет),
e-mail: neurocomputer@yandex.ru

Нейросетевые технологии в перспективных суперЭВМ. Концепция развития высокопроизводительных вычислений на базе супернейрокомпьютеров (2012—2020 гг.)

Представлена история развития масштабируемых суперЭВМ с двухслойной архитектурой и их оценка с точки зрения эффективности моделирования сложных нейронных сетей. Отмечается ведущая роль в настоящее время суперЭВМ на базе графических процессоров для моделирования нейронных сетей. Представлена концепция развития супернейрокомпьютеров — суперЭВМ на базе нейросетевых технологий.

Ключевые слова: *нейросетевые технологии, супернейрокомпьютеры, двухслойные архитектуры, масштабируемость.*

Данная работа посвящена краткому изложению развития масштабируемых суперЭВМ с двухслойной архитектурой. Эта история схематично представлена на рис. 1 (см. третью сторону обложки), где отмечено, что первыми образцами ЭВМ с двухслойной архитектурой были ЭВМ с матричными сопроцессорами и суперЭВМ типа STARAN.

В соответствии с профессиональными интересами автора большинство типов высокопроизводительных вычислительных систем исследовали тестированием моделями сложных нейронных сетей. Именно этот тест был главным при оценке производительности ЭВМ той или иной архитектуры. Количественным критерием производительности являлись и являются в настоящее время следующие три показателя:

- число эмулируемых нейронов;
- количество эмулируемых связей между нейронами;
- скорость переключения связей в секунду.

Транспьютерные системы [1] стали предвестником многих современных суперЭВМ как кластерных, так и гибридных. Транспьютерные системы кластерного типа оказались неэффективными для моделирования сложных нейронных сетей, однако еще в конце 80-х — начале 90-х годов прошлого столетия эффективными для решения этой задачи были предвестники современных гибридных суперЭВМ, а именно двухслойные транспьютерные архитектуры (рис. 2) с периферийными [1]:

— сигнальными процессорами JMSA100;

— процессорами обработки изображений JMSA110;

— процессорами i860.

Гибридные суперЭВМ на базе процессоров nVidia являются, по сути дела, гармоничным развитием этой архитектуры на более совершенном технологическом уровне. Если отметить два основных свойства современных суперЭВМ — масштабируемость и двухслойность архитектур, то, с нашей точки зрения, для перехода на экзафлопный уровень вычислений необходима реализация, по крайней мере, еще одного шага — изменения логического базиса алгоритмов решения задач и, соответственно, элементной базы.

В работе [2] отмечено следующее: "Белый дом заявил о нежелании втягиваться в гонку вооружений в вопросе создания все более быстрых компьютеров, и год назад в своем докладе предупредил, что акцент на скорость "может отвлечь ресурсы от фундаментальных исследований, направленных на развитие принципиально новых подходов, которые помогли бы нам, в конечном итоге, обогнать другие страны"".

Основными предпосылками принципиального изменения архитектуры вычислительных систем при переходе к системам экзафлопной производительности являются следующие:

- необходимость резкого повышения надежности за счет отказа от фон Неймановской архитектуры вычислительных систем из элементов булевской логики И, ИЛИ, НЕ, когда в системе из значительного числа этих элементов происходит катастрофический отказ при отказе любого элемента;

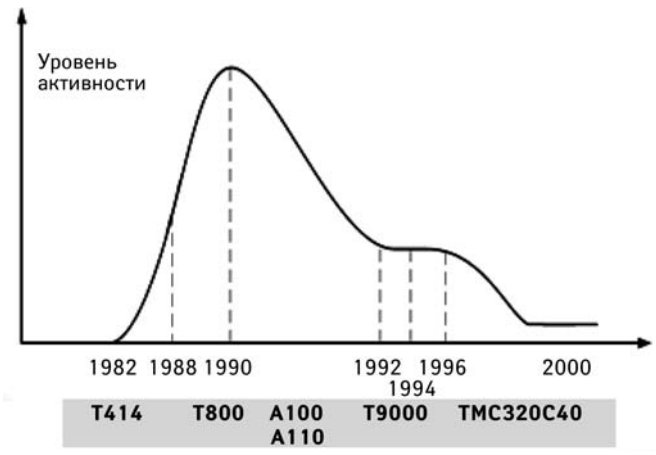


Рис. 2. Активность разработок и применения транспьютерных систем

- необходимость резкого снижения энергопотребления за счет отказа от существующего позиционного метода представления информации и перехода к другим, обеспечивающим снижение энергопотребления и, как следствие, дополнительное повышение надежности;
- необходимость резкого повышения однородности схемотехники элементов вычислительных систем, что должно привести к повышению эффективности при том резком увеличении интеграции элементов, которое будет иметь место при переходе к экзафлопной производительности.

Детальный анализ документа [3] показал необходимость разработки подобного документа, ориентированного на использование при разработке экзафлопных суперЭВМ нейросетевых технологий [4].

К разработке документа [4] "Экзафлопные нейросетевые технологии. Концепция развития технологий высокопроизводительных вычислений на базе супернейрокомпьютеров (2012—2020 гг.)" было привлечено более сотни ведущих российских ученых в области нейросетевых технологий, в основном работающих в российских университетах, с некоторым, к сожалению незначительным, участием ученых из РАН.

Научно-техническими заделами по созданию отечественных супернейрокомпьютеров экзафлопной производительности являются заделы по следующим направлениям:

- *Теория нейронных сетей* как методика синтеза структур из нейронов различного вида, алгоритмов адаптации весовых коэффициентов в этих структурах в процессе решения различных задач.
- *Нейроматематика* как раздел вычислительной математики, связанный с решением в нейросетевом логическом базисе различных сложных формализуемых и неформализуемых задач.
- *Нейроуправление* как раздел теории управления, связанный с применением нейрокомпьютеров в качестве систем идентификации сложных динамических систем и нейрокомпьютеров.
- *Нейрокомпьютеры и нейрочипы*.

В работе [5] представлен систематизированный материал аннотаций работ российских авторов в области развития и применения нейросетевых технологий за 1982—2010 годы, а именно:

- диссертационные работы (кандидатские и докторские);
- монографии;
- научно-технические отчеты фонда ВНТИЦ.

Анализ этих материалов показывает значительный потенциал ученых в России в области развития и применения нейросетевых технологий.

Известны ведущие в мире основные проекты супернейрокомпьютеров экзафлопной производительности:

- Irvine Sensors;
- проект Ливерморской лаборатории;
- проект Института продвинутых архитектур (Сандиа, Оак Ридж);
- проект Интел, центр суперкомпьютерных вычислений Сан-Диего, DARPA;
- проект eLiza — самоуправляемой автономной компьютерной системы (IBM);
- SyNAPSE (Systems Neuromorphic Adaptive Plastic Scalable Electronics);
- Neurogrid (Стэнфордский университет);
- SpiNNaker (Spiking Neural Network Architecture) — группы английских университетов;
- BICA (Biologically Inspired Computer Architecture);
- BlueBrain.

Основными прикладными задачами, стимулирующими развитие суперЭВМ, в том числе супернейрокомпьютеров, являются следующие:

- биоинформатика;
- криптография;
- ядерные исследования;
- газо-, аэро-, гидродинамика;
- обработка космических изображений;
- обработка больших массивов текстовой информации;
- моделирование мозга.

В работе [6] был представлен предварительный рабочий материал к Концепции развития технологий высокопроизводительных вычислений на базе супернейрокомпьютеров. В работах [4] и [6] отмечены следующие задачи и направления.

1. Задачи, требующие разработки и применения нейросетевых технологий.

1.1. Нейросетевые технологии в хемо- и биоинформатике.

1.2. Нейросетевые технологии в криптографии.

1.3. Нейросетевые технологии в ядерной энергетике.

1.4. Нейросетевые технологии в медицине.

1.5. Нейросетевые технологии в системах добычи и обработки полезных ископаемых.

1.6. Нейросетевые технологии в управлении энергетическими системами.

1.7. Нейросетевые технологии в управлении транспортными системами.

1.8. Нейросетевые технологии в биометрии.

1.9. Нейросетевые технологии в обработке статических и видеоизображений.

1.10. Нейросетевые технологии в обработке сигналов.

1.11. Нейросетевые технологии в обработке больших массивов текстовой информации.

1.12. Нейросетевые технологии в разработках интерфейсов мозг—компьютер.

1.13. Нейросетевые технологии в авиации.

1.14. Нейросетевые технологии в космической технике.

1.15. Нейросетевые технологии в роботостроении.

1.16. Нейросетевые технологии в моделировании сложных технических систем.

1.17. Нейросетевые технологии в распределенных вычислениях и моделировании распределенных систем.

1.18. Нейросетевые технологии в экологии.

1.19. Нейросетевые модели экономических, социальных, политических процессов.

2. Направления фундаментальных исследований в области нейросетевых технологий.

2.1. Теория нейронных сетей.

2.2. Нейросетевые технологии в решении сложных математических задач (нейроматематика).

2.3. Нейроуправление.

2.4. Разработка моделей разделов мозга для современных и перспективных суперЭВМ.

3. Направления, связанные с созданием супернейрокомпьютеров.

3.1. Нейрокомпьютеры.

3.2. Основные предпосылки предлагаемого направления развития работ по архитектуре перспективных экзафлопных суперЭВМ с применением нейросетевых технологий.

3.3. Нейрочипы:

цифровые нейрочипы;
аналоговые и аналого-цифровые нейрочипы;
супернейрокомпьютер фирмы Irvine Sensors;
мемристоры;
клеточные нейрочипы;
нейрочипы с частотно-импульсной модуляцией сигналов;
специализированные аналоговые и аналого-цифровые нейрочипы.

3.4. Перспективные технологии в реализации супернейрокомпьютеров:

многослойные системы;
оптические нейрокомпьютеры;
одноэлектронные системы;
молекулярные и ДНК-вычисления;
квантовые нейрокомпьютеры.

3.5. Специализированные и периферийные системы:

ассоциативная память;
решатели систем линейных алгебраических уравнений;
специализированный процессор обработки изображений.

3.6. Разработка системного программного обеспечения:

операционная система;
система ввода/вывода;
система управления;
внешняя среда;
модель программирования;
компиляторы;
библиотеки;
средства отладки.

3.7. Перспективные свойства супернейрокомпьютеров:

ассоциативная память;
эффективное распараллеливание алгоритмов решения задач;
реконфигурируемость;
масштабируемость;
надежность;
визуализация;
облачные вычисления;
тестирование программного обеспечения;
методика оценки производительности нейрокомпьютеров.

3.8. Создание и развитие супернейрокомпьютерных центров.

Необходимо отметить объективные причины формирования представленной "Концепции по развитию технологий высокопроизводительных вычислений на базе супернейрокомпьютеров" (2012—2020 гг.).

1. В концепции "Экзафлопные технологии" [3] отсутствует даже упоминание о нейросетевых технологиях.

2. В авторском коллективе концепции "Экзафлопные технологии" [3] отсутствуют специалисты по нейросетевым технологиям.

3. Список цитируемой литературы в концепции "Экзафлопные технологии" [3] говорит о слабости, если не сказать точнее о нулевой информированности ее разработчиков об активном движении зарубежных и отечественных ученых в направлении разработок сверхвысокопроизводительной вычислительной техники с применением нейросетевых технологий.

Предлагаемая концепция [4] является в той или иной степени развитием концепции "Экзафлопные технологии" [3] по следующим причинам:

1. Любой из разрабатываемых нейросетевых пакетов прикладных программ разрабатывается, в том числе и для перспективной реализации гибридной экзафлопной суперЭВМ на базе GPU.

2. Разрабатываемый по предлагаемой концепции пакет программ распределения нагрузки между процессорами с применением нейросетевых технологий ориентирован в основном и, в первую очередь, на суперЭВМ, разрабатываемые по концепции "Экзафлопные технологии" [3].

3. Обе концепции ставят перед собой одну и ту же цель — создать высокопроизводительные вычислительные системы, по отношению производительности к стоимости (энергопотреблению) значительно превосходящие существующие.

4. Обе концепции нельзя рассматривать независимо (научно, организационно, финансово) друг от друга. Обе концепции существуют параллельно друг с другом уже несколько десятилетий. Концепция "Экзафлопные технологии" в первые годы своего существования практически сразу начала трактоваться как "универсальная". Предлагаемая концепция в течение нескольких десятилетий трактовалась как "специализированная", эффективная для

решения узкого класса сложных задач. "Универсальность" предлагаемой концепции стала активно формироваться в начале 90-х годов прошлого столетия и развивается в настоящее время.

Список литературы

1. **Галушкин А. И.** Транспьютерные системы — начало становления в России ЭВМ с массовым параллелизмом // Информатизация и связь. № 1. 2009.
2. **Высокопроизводительные** вычисления (мировой рынок) // TADVISER. 24.11.2011.
3. **Экзафлопные** технологии. Концепция по развитию технологий высокопроизводительных вычислений на базе супер-ЭВМ экзафлопного класса (2012—2020 гг.). 2012 г.
4. **Экзафлопные** нейросетевые технологии. Концепция развития технологий высокопроизводительных вычислений на базе супернейрокомпьютеров (2012—2020 гг.). 2012 г.
5. **Галушкин А. И., Симоров С. Н.** Нейросетевые технологии в России (1982—2010 гг.) // М.: Горячая линия-Телеком, 2012.
6. **Галушкин А. И.** Стратегия развития современных супернейрокомпьютеров на пути к экзафлопным вычислениям // Приложение к журналу "Информационные технологии". 2012. № 3. 32 с.

УДК 004.032.26

В. И. Горбаченко, д-р техн. наук, проф.,
e-mail: gorvi@mail.ru,
М. В. Жуков, аспирант,
e-mail: maxim.zh@gmail.com,
Пензенский государственный университет

Обучение сети радиальных базисных функций методом доверительных областей для решения уравнения Пуассона

Разработан алгоритм обучения сети радиальных базисных функций с помощью метода доверительных областей. На примере решения уравнения Пуассона показано, как алгоритм можно использовать для решения задач математической физики. Экспериментальные исследования показали, что предложенный алгоритм по точности превосходит известные алгоритмы.

Ключевые слова: *сети радиальных базисных функций, обучение сетей радиальных базисных функций, метод доверительных областей, уравнение Пуассона*

Введение

Сеть радиальных базисных функций (*Radial Basis Function Network*, RBF-сеть) [1] — это сеть, состоящая из двух слоев. Первый слой осуществляет нелинейное преобразование входного вектора **x**. Второй слой выполняет линейное суммирование. В качестве функции преобразования используются радиальные базисные функции (Radial Basis Function, RBF-функции), так, в данной работе применяется функция Гауссиан:

$$\varphi(\mathbf{x}) = \mathbf{e}^{\frac{-\|\mathbf{x}-\mathbf{c}\|^2}{2a^2}},$$

где **x** — точка пространства; **c** — центр функции; a — ширина; $\|\cdot\|$ — евклидова норма.

Одна из важнейших областей применения RBF-сетей — решение задач аппроксимации [1]. В частности, RBF-сети применяют для аппроксимации решений задач математической физики. Преимущество данного подхода перед конечно-разностным методом, методом конечных элементов и другими методами заключается в его бессеточной природе, что позволяет в значительной степени упростить решение задач со сложной геометрией, а также избежать экспоненциального роста вычислительной сложности при увеличении размерности задачи [2—4]. Кроме того, как показали экспериментальные исследования, проведенные в работе [5], RBF-сети значительно лучше аппроксимируют решения дифференциальных уравнений, чем многослойные нейронные сети.

Решение уравнения Пуассона с помощью RBF-сети

Решим с помощью RBF-сети двухмерное уравнение Пуассона

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f(x, y),\ (x, y) \in \Omega, \qquad (1)$$

$$u = p(x, y),\ (x, y) \in \partial\Omega, \qquad (2)$$

где Ω — область решения; $\partial\Omega$ — граница области; f и p — известные функции.

Представим RBF-сеть как аппроксиматор функции решения:

$$u(\mathbf{x}) = \sum_{k=1}^{m} w_k \mathbf{e}^{\frac{-\|\mathbf{x}-\mathbf{c}_k\|^2}{2a_k^2}},$$

где m — число RBF-функций (скрытых нейронов); w_k — вес, связывающий выходной нейрон с k-м нейроном скрытого слоя. Решение (1)—(2) происходит в процессе обучения RBF-сети [6, 7], т. е. настройки весов, центров и ширины нейронов, таким образом, чтобы функционал ошибки, представляю-

щий собой сумму квадратов невязок в контрольных точках, принимал минимальное значение:

$$I(\mathbf{w}, \mathbf{c}, \mathbf{a}) = \sum_{i=1}^{N} \left[\frac{\partial^2 u(x_i, y_i)}{\partial x^2} + \frac{\partial^2 u(x_i, y_i)}{\partial y^2} - f(x_i, y_i) \right]^2 + \lambda \sum_{i=N+1}^{N+K} [u(x_i, y_i) - p(x_i, y_i)]^2, \quad (3)$$

где λ — штрафной множитель; N и K — число внутренних и граничных контрольных точек соответственно. Таким образом, задача обучения может рассматриваться как задача многокритериальной минимизации функционала (3). В работе [3] приведен обзор методов обучения сети. Все они основаны на одном из двух подходов: либо обучаются только веса, а центрам и ширине RBF-функций задаются фиксированные значения [8]; либо поочередно с помощью градиентных методов обучаются и центры, и ширина, и веса [4, 6, 7]. Методы, в основу которых положен первый подход, позволяют быстро находить оптимальные значения весов, однако требуют тщательного подбора ширины RBF-функций, предназначены для решения только линейных дифференциальных уравнений, вследствие большого числа используемых RBF-функций возникает эффект переобучения сети. Сети, обученные с помощью методов, основанных на втором подходе, в значительно меньшей степени подвержены эффекту переобучения, позволяют решать нелинейные дифференциальные уравнения, тем не менее обучение происходит довольно медленно, а качество решения сильно зависит от начальных параметров сети. В данной работе используется альтернативный, третий подход, согласно которому центры, ширина и веса обучаются одновременно. Для этого используется метод доверительных областей (*Trust Region Method*, TR-метод) [9].

Метод доверительных областей

TR-метод позволяет решать задачи вида

$$\text{minimize } f(\mathbf{x}), \quad (4)$$

где $f(\mathbf{x})$ — целевая, дважды дифференцируемая функция. Метод относится к группе итеративных методов. На каждой итерации для заданной области B_k, называемой доверительной областью, и начальной точки $\mathbf{x}_k \in B_k$ строится модель $m_k(\mathbf{x})$, аппроксимирующая функцию $f(\mathbf{x})$, причем $m_k(\mathbf{x}_k) = f(\mathbf{x}_k)$. Доверительная область B_k задается множеством точек

$$B_k = \{\mathbf{x} \in R^n \mid \|\mathbf{x} - \mathbf{x}_k\| \leqslant \Delta_k\},$$

где Δ_k — радиус доверительной области, $\|\cdot\|$ — произвольная норма вектора, наилучшим способом соответствующая решаемой задаче. Ключевыми особенностями метода являются: возможность одновременной оптимизации большого числа параметров; высокие показатели эффективности и сходимости даже для плохо обусловленных задач; возможность за счет использования аппроксимационных моделей преодолевать локальные минимумы; возможность минимизировать выпуклые функции, т. е. функции с отрицательно определенной матрицей Гессе; и, наконец, при использовании в роли аппроксимирующей функции разложения в ряд Тейлора второго порядка задача (4) сводится к задаче минимизации квадратичного функционала [9]. Все это делает TR-метод идеальным для решения задачи минимизации функционала (3), которая часто имеет и большое число оптимизируемых параметров, и множество локальных минимумов, и плохо обусловленную матрицу Гессе.

Тогда задача (4) сводится к задаче минимизации m_k в области B_k. Полученный в результате ее решения шаг $\mathbf{s}_k$, такой, что $\|\mathbf{s}_k\| \leqslant \Delta_k$ и m_k принимает минимальное значение в точке $\mathbf{x}_k + \mathbf{s}_k$ для заданной области B_k, становится пробным шагом, минимизирующим f. Если значение $f(\mathbf{x}_k + \mathbf{s}_k) - f(\mathbf{x}_k)$ сопоставимо с $m_k(\mathbf{x}_k + \mathbf{s}_k) - m_k(\mathbf{x}_k)$, т. е. уменьшение, предсказанное моделью, подтверждено целевой функцией, то пробная точка становится начальной точкой для следующей итерации. В противном случае точка $\mathbf{x}_k + \mathbf{s}_k$ отклоняется, а доверительная область сужается в надежде на то, что модель будет точнее аппроксимировать целевую функцию в меньшей области. Ниже представлено формальное описание метода [9]:

Шаг 1. Инициализация. Задаются начальное положение $\mathbf{x}_0$, радиус доверительной области Δ_0, параметры μ_1, μ_2, γ_1, γ_2 такие, что $0 < \mu_1 \leqslant \mu_2 < 1$ и $0 < \gamma_1 \leqslant \gamma_2 < 1$, $k = 0$.

Шаг 2. Построение модели. Выбирается норма, строится модель m_k функции f в области B_k.

Шаг 3. Минимизация модели. Вычисляется направление $\mathbf{s}_k$, минимизирующее значение функции m_k в области B_k.

Шаг 4. Проверка точности модели. Вычисляется

$$p_k = \frac{f(\mathbf{x}_k) - f(\mathbf{x}_k + \mathbf{s}_k)}{m_k(\mathbf{x}_k) - m_k(\mathbf{x}_k + \mathbf{s}_k)}.$$

Если $p_k \geqslant \mu_1$, то $\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{s}_k$, иначе $\mathbf{x}_{k+1} = \mathbf{x}_k$.

Шаг 5. Изменение радиуса доверительной области.

$$\Delta_{k+1} = \begin{cases} [\Delta_k; \infty), & \text{если } p_k \geqslant \mu_2, \\ [\gamma_2 \Delta_k; \Delta_k], & \text{если } p_k \in [\mu_1; \mu_2), \\ [\gamma_1 \Delta_k; \gamma_2 \Delta_k], & \text{если } p_k < \mu_1, \end{cases}$$

$k = k + 1$. Перейти к шагу 2.

Представленный алгоритм носит обобщенный характер, многие детали которого не описаны. В частности, нет никаких указаний на то, как выбирать модель m_k и норму $\|\cdot\|$.

На практике [9] в качестве нормы выбирают евклидову норму, а аппроксимацию целевой функции осуществляют с помощью квадратичного функционала:

$$q(\mathbf{x}_k + \mathbf{s}) = f(\mathbf{x}_k) + \langle \mathbf{g}, \mathbf{s} \rangle + \frac{1}{2} \langle \mathbf{s}, \mathbf{Hs} \rangle,$$

где q — разложение функции f в ряд Тейлора второго порядка в окрестности точки $\mathbf{x}_k$; $\mathbf{g} = \nabla_x f(\mathbf{x}_k)$ — вектор градиента функции f в точке $\mathbf{x}_k$; $\mathbf{H}$ — симметричная матрица, представляющая собой матрицу Гессе функции f, т. е. $\mathbf{H} = \nabla_{xx} f(\mathbf{x}_k)$; $\langle \cdot \rangle$ — скалярное произведение векторов. Тогда на каждой итерации необходимо решить задачу

$$\text{minimize } q(\mathbf{x}_k + \mathbf{s}),$$
$$\text{причем, } \|\mathbf{s}\| \leqslant \Delta, \qquad (5)$$

где $\Delta = \Delta_k$. Заметим, что при использовании TR-метода необязательно вычислять точное решение (5), что требует определенных вычислительных затрат, но чем точнее решена задача (5), тем меньше итераций потребуется для решения задачи (4). То есть при выборе требуемой точности решения (5) нужно маневрировать между сложностью ее решения и сложностью вычисления f, $\mathbf{g}$ и $\mathbf{H}$.

Метод Стайхауга

На первый взгляд поиск решения задачи (5) не представляет никакой сложности, в частности, его можно найти с помощью метода сопряженных градиентов (*Conjugate Gradient Method*, CG-метода) [10]. Однако для сходимости данного метода необходимо, чтобы матрица $\mathbf{H}$ была положительно определенной, другими словами, функция m_k была вогнутой. К тому же, в задаче (5) на решение наложено ограничение $\|\mathbf{s}\| \leqslant \Delta$, т. е. решение не должно выходить за границу доверительной области. Одним из методов, учитывающих эти особенности, является метод Стайхауга [11]. В его основу положен метод сопряженных градиентов с предобусловливанием (*Preconditioned Conjugate Gradient Method*, PCG-метод) [10], т. е. CG-метод, примененный не к системе $\mathbf{Hx} = \mathbf{g}$, а к ее предобусловленному аналогу $\tilde{\mathbf{H}}\,\tilde{\mathbf{x}} = \tilde{\mathbf{g}}$, где $\tilde{\mathbf{H}} = \mathbf{C}^{-1}\mathbf{H}\mathbf{C}^{-1}$; $\tilde{\mathbf{x}} = \mathbf{C}\mathbf{x}$; $\tilde{\mathbf{g}} = \mathbf{C}^{-1}\mathbf{g}$; а $\mathbf{C}$ — симметричная положительно определенная матрица, подобранная таким образом, чтобы $\tilde{\mathbf{H}}$ оказалась хорошо обусловленной матрицей. Вычислительную сложность PCG-метода можно значительно уменьшить, правильно подобрав $\mathbf{C}$ и заменив предобусловливатель на $\mathbf{M} = \mathbf{C}^2$, причем матрица $\mathbf{M}$ должна быть легко обратима. В данной работе в качестве предобусловливателя используется диагональная матрица $\mathbf{M}$, состоящая из диагональных элементов матрицы $\mathbf{H}$.

PCG-методу свойственны те же ограничения, что и CG-методу, поэтому, если $\mathbf{H} < 0$, т. е. в процессе вычисления нового сопряженного направления $\mathbf{p}_k$, получится, что $\mathbf{p}_k\mathbf{H}\mathbf{p}_k < 0$, Стайхауг предложил в качестве шага $\mathbf{s}$ использовать вектор $\mathbf{s}_k + \tau\mathbf{p}_k$, где $\mathbf{s}_k$ — шаг, полученный на k-й итерации, τ является решением уравнения

$$\|\mathbf{s}_k + \tau\mathbf{p}_k\|_{\mathbf{M}} = \Delta. \qquad (6)$$

Заметим, что в уравнении (6) используется не евклидова норма как в (5), а $\mathbf{M}$ — норма (например, если в качестве $\|\cdot\|$ выбрана евклидова норма, то $\|\mathbf{x}\|_{\mathbf{M}} = \sqrt{\langle \mathbf{x}, \mathbf{M}\mathbf{x}\rangle}$). Это связано с тем, что при применении предобусловливателя $\mathbf{M}$ к (5) происходит преобразование доверительной области.

В случае если $\mathbf{H} > 0$ и в результате применения PCG-метода на k-м шаге получен вектор $\mathbf{p}_k$, выходящий за границу доверительной области, то шаг $\mathbf{s}$ также вычисляется по формуле (6). В обоих случаях τ является положительным корнем квадратного уравнения (6) и вычисляется по формуле

$$\tau = \frac{-\langle \mathbf{s}_k, \mathbf{M}\mathbf{p}_k\rangle + \sqrt{\langle \mathbf{s}_k, \mathbf{M}\mathbf{p}_k\rangle^2 + \|\mathbf{p}_k\|_{\mathbf{M}}^2(\Delta^2 - \|\mathbf{s}_k\|_{\mathbf{M}}^2)}}{\|\mathbf{p}_k\|_{\mathbf{M}}^2}. \qquad (7)$$

Ниже представлено формальное описание метода Стайхауга.

Шаг 1. Инициализация. $k = 0$, $\mathbf{r}_k = -\mathbf{g}$, $\mathbf{s}_k = 0$, $\mathbf{z}_k = \mathbf{M}^{-1}\mathbf{r}_k$, $\mathbf{p}_k = \mathbf{z}_k$.

Шаг 2. $\gamma = \langle \mathbf{p}_k, \mathbf{H}\mathbf{p}_k\rangle$.

Шаг 3. $\mathbf{H} < 0$. Если $\gamma \leqslant 0$, то вычислить τ по формуле (7), $\mathbf{s} = \mathbf{s}_k + \tau\mathbf{p}_k$. Конец.

Шаг 4. $\alpha = \langle \mathbf{r}_k, \mathbf{z}_k\rangle/\gamma$.

Шаг 5. Пересечена граница. Если $\|\mathbf{s}_k + \alpha\mathbf{p}_k\|_{\mathbf{M}} \geqslant \Delta$, то вычислить τ по формуле (7), $\mathbf{s} = \mathbf{s}_k + \tau\mathbf{p}_k$. Конец.

Шаг 6. Переход. $\mathbf{s}_{k+1} = \mathbf{s}_k + \alpha\mathbf{p}_k$; $\mathbf{r}_{k+1} = \mathbf{r}_k - \alpha\mathbf{H}\mathbf{p}_k$; $\mathbf{z}_{k+1} = \mathbf{M}^{-1}\mathbf{r}_{k+1}$; $\beta = \langle \mathbf{r}_{k+1}, \mathbf{z}_{k+1}\rangle/\langle \mathbf{r}_k, \mathbf{z}_k\rangle$; $\mathbf{p}_{k+1} = \mathbf{z}_{k+1} + \beta\mathbf{p}_k$; $k = k+1$.

Критериями останова на шаге 6 могут служить те же критерии, что применяются в CG-методе, например, $\frac{\|\mathbf{r}_{k+1}\|}{\|\mathbf{g}\|} < \xi$, где ξ — заранее заданное число.

Обучение RBF-сети с помощью TR-метода

Перейдем теперь непосредственно к задаче минимизации функционала (3). Чтобы избавиться от необходимости вывода формул расчета частных производных по весам, центрам и ширине, необходимых для вычисления вектора градиента ∇I и матрицы Гессе $\mathbf{H}(I)$, и сделать описание методики применения TR-метода для минимизации функционала ошибки более общим, вместо частных производных будем использовать разностные частные производные. Учитывая это и тот факт, что в качестве RBF-функции используется Гауссиан, функционал (3) можно записать в виде

$$I(\mathbf{p}) = \sum_{i=1}^{N}\left[\sum_{k}^{m}\left[\frac{u_k(x_i + 2h, y_i) - 2u_k(x_i + h, y_i) + u_k(x_i, y_i)}{h^2} + \frac{u_k(x_i, y_i + 2h) - 2u_k(x_i, y_i + h) + u_k(x_i, y_i)}{h^2}\right] - f(x_i, y_i)\right]^2 + \lambda \sum_{i=N+1}^{N+K}\left[\sum_{k}^{m}[u_k(x_i, y_i)] - p(x_i, y_i)\right]^2, \qquad (8)$$

где $u_k = w_k \mathbf{e}^{\frac{-((x_i - c_k^x)^2 + (y_i - c_k^y)^2)}{2a_k^2}}$; $\mathbf{p} = (w_1, ..., w_m, c_1^x, ..., c_m^x, c_1^y, ..., c_m^y, a_1, ..., a_m)$; h — шаг разностной производной.

Более того, поскольку функционал (8) — квадратичный функционал и задача (5) не требует точного решения, вычислительную сложность расчета $\nabla I(\mathbf{p})$ и $\mathbf{H}(\mathbf{p})$ можно значительно уменьшить, воспользовавшись формулами, используемыми в методе Гаусса—Ньютона для минимизации квадратичного функционала, $\nabla I(\mathbf{p}) = [\mathbf{J}(\mathbf{p})]^{\mathrm{T}}\mathbf{r}(\mathbf{p})$, $\mathbf{H}(\mathbf{p}) \approx \mathbf{G}(\mathbf{p}) = [\mathbf{J}(\mathbf{p})]^{\mathrm{T}}\mathbf{J}(\mathbf{p})$, где $\mathbf{r}(\mathbf{p}) = [r_i]_{i=1}^{N+M}$ — вектор ошибок, а $\mathbf{J}(\mathbf{p}) = (J_{ij})_{i=1, j=1}^{N+M, 4m}$ — матрица Якоби. Компоненты r_i и J_{ij} вычисляются по следующим формулам:

$$r_i = \begin{cases} \sum\limits_k^m \xi_k(\mathbf{p}, x_i, y_i) - f(x_i, y_i), \ 1 \leqslant i \leqslant N, \\ \sqrt{\lambda} \sum\limits_k^m \psi_k(\mathbf{p}, x_i, y_i) - p(x_i, y_i), \ N < i \leqslant N; \end{cases}$$

$$J_{ij} = \begin{cases} \frac{1}{h}[\xi_{j(\mathrm{mod}\ m)}(p_1, ..., p_j + h, ..., p_{4m}, x_i, y_i) - \\ - \xi(p_1, ..., p_j, ..., p_{4m}, x_i, y_i)], \ 1 \leqslant i \leqslant N, \\ \frac{\sqrt{\lambda}}{h}[\psi_{j(\mathrm{mod}\ m)}(p_1, ..., p_j + h, ..., p_{4m}, x_i, y_i) - \\ - \psi_{j(\mathrm{mod}\ m)}(p_1, ..., p_j, ..., p_{4m}, x_i, y_i)], \\ N < i \leqslant N + K, \end{cases}$$

где

$$\xi_k(\mathbf{p}, x_i, y_i) = \frac{u_k(x_i + 2h, y_i) - 2u_k(x_i + h, y_i) + u_k(x_i, y_i)}{h^2} + \frac{u_k(x_i, y_i + 2h) - 2u_k(x_i, y_i + h) + u_k(x_i, y_i)}{h^2},$$

$$\psi_k(\mathbf{p}, x_i, y_i) = u_k(x_i, y_i).$$

Экспериментальные исследования

Решение уравнения Пуассона (1)—(2) с помощью RBF-сети, обученной TR-методом, проводилось для $f(x, y) = \sin(\pi x)\sin(\pi y)$, $p(x, y) = 0$ и квадратной области решения, ограниченной точками (0; 0) и (1; 1) [6]. Данная задача имеет аналитическое решение $u_a(x, y) = -\frac{1}{2\pi^2}\sin(\pi x)\sin(\pi y)$. Погрешность решения рассчитывалась по формуле относительной средней квадратичной погрешности $\sqrt{\sum\limits_{i=1}^{N+K}(u_i - u_{ai})^2} \Big/ \sqrt{\sum\limits_{i=1}^{N+K} u_{ai}^2}$, где u_i — численное решение в точке i; u_{ai} — аналитическое решение.

Для обучения сети использовались 144 контрольные точки, 100 из которых располагалось в области решения Ω, 44 — в области $\partial\Omega$. Экспериментальные исследования были проведены для трех различных конфигураций RBF-сети.

В первом случае RBF-сеть состояла из 16 нейронов, центры которых в начальный момент времени были случайным образом расположены в квадратной области, ограниченной точками (−0,2; −0,2) и (1,2; 1,2), ширина нейронов выбиралась случайным образом из интервала [0,3; 0,6], начальные значения весов RBF-функций также выбирались случайным образом из интервала [−0,05; 0,05]. На рис. 1 и 2 представлены начальная и конечная (конфигурация RBF-сети после обучения) конфигурации RBF-сети соответственно. На рисунках точками обозначены центры RBF-функций, закрашенными вокруг них областями — веса RBF-функ-

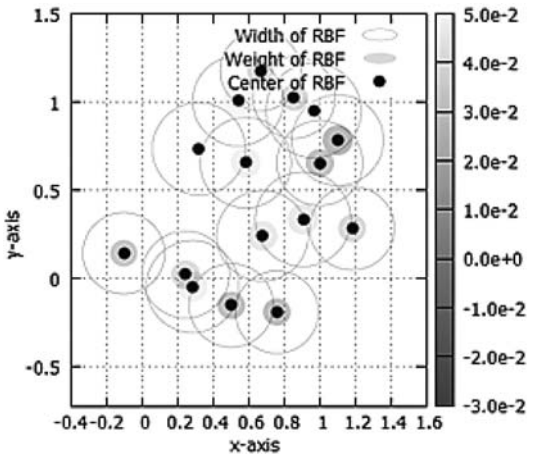


Рис. 1. Эксперимент № 1. Начальная конфигурация сети

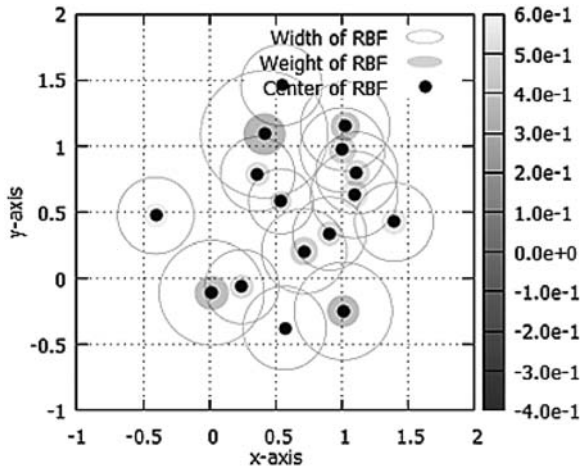


Рис. 2. Эксперимент № 1. Конфигурация сети после обучения

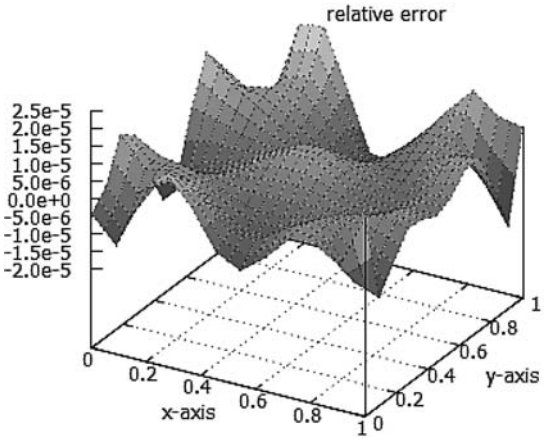


Рис. 3. Эксперимент № 1. Погрешность решения

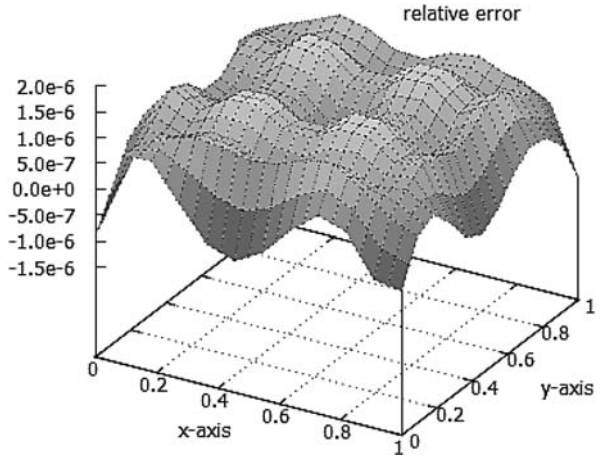


Рис. 6. Эксперимент № 2. Погрешность решения

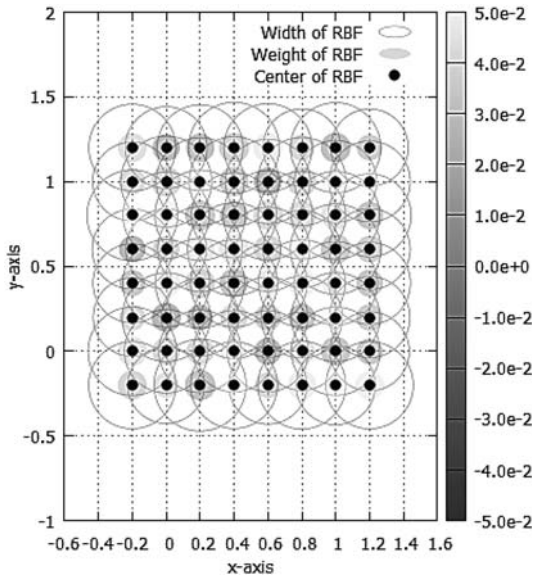


Рис. 4. Эксперимент № 2. Начальная конфигурация сети

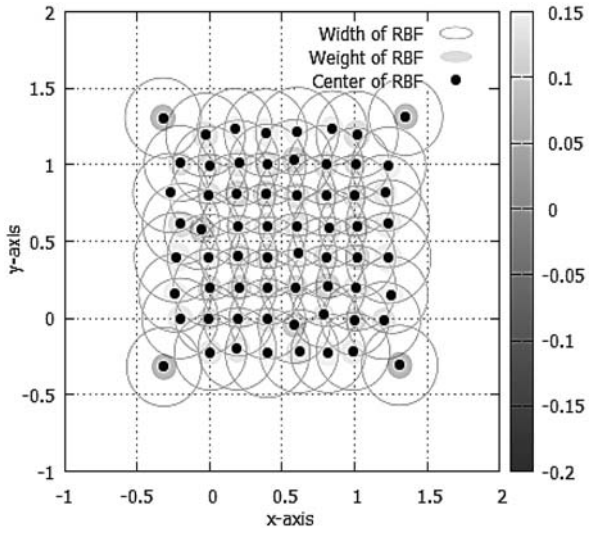


Рис. 5. Эксперимент № 2. Конфигурация сети после обучения

ций (цвет области зависит от веса RBF-функции, радиус области равен 1/3 ширины RBF-функции), серыми окружностями — ширина RBF-функций (радиус окружности равен ширине RBF-функции). По рисункам видно, что в процессе обучения центры RBF-функций сместились к диагоналям квадрата, ограниченного точками (0; 0) и (1; 1), а вблизи вершин этого квадрата разместились функции, имеющие наибольший вес. Обучение сети завершилось на 12-й итерации со значением средней квадратичной погрешности решения, равным $2{,}9 \cdot 10^{-4}$. График погрешности представлен на рис. 3.

Во втором эксперименте RBF-сеть состояла из 64 нейронов, центры которых располагались в узлах равномерной квадратной сетки, ограниченной точками (-0,2; −0,2) и (1,2; 1,2), ширина нейронов была постоянной и равной 0,5, начальные значения весов RBF-функций были равны 0. Начальная и конечная конфигурации сети представлены на рис. 4 и 5, по ним видно, что наиболее интересные изменения произошли с центрами RBF-функций: они образовали некое подобие окружностей вокруг точки (0,5; 0,5), а функции, расположенные в вершинах квадрата, ограниченного точками (−0,2; −0,2) и (1,2; 1,2), были вытолкнуты за его границы. За 25 итераций была достигнута погрешность решения, равная $4 \cdot 10^{-5}$. График погрешности представлен на рис. 6.

В последнем случае конфигурация RBF-сети была такой же, как и во втором случае, однако в обучении участвовали только веса RBF-функций. Обучение сети заняло три итерации. Погрешность составила $5{,}6 \cdot 10^{-5}$.

Во всех трех случаях значение относительной средней квадратичной погрешности в 20, а то и в 100 раз меньше значения погрешности, достигнутого в работе [6] — $5 \cdot 10^{-3}$. Более того, полученное в эксперименте № 2 значение погрешности в 10 раз меньше значения, полученного в работе [7] ($5 \cdot 10^{-4}$) с помощью метода сопряженных градиентов минимизации квадратичного функционала ошибки, что

объясняется способностью TR-метода сглаживать минимизируемый функционал ошибки.

Заключение

Как и следовало ожидать, чем больше нейронов в сети, тем точнее она аппроксимирует искомое решение, однако в этом случае возрастает риск переобучения сети, процесс обучения такой сети требует значительно больше времени (для обучения сети из эксперимента № 1 потребовалась в 2 раза меньше итераций, чем для обучения сети из эксперимента № 2, кроме того, в эксперименте № 2 каждая итерация заняла в 3 раза больше времени, чем в эксперименте № 1).

В третьем случае фактически решалась система линейных алгебраических уравнений (СЛАУ). Действительно, если зафиксировать центры и ширину RBF-функций, задача минимизации (3) сводится к минимизации квадратичного функционала с линейными функциями невязки, который путем дифференцирования можно преобразовать в СЛАУ. В результате возникает вопрос, зачем вообще обучать центры и ширину RBF-функций, если их можно зафиксировать и решить СЛАУ. У такого подхода есть несколько проблем, во-первых, он применим только к линейным дифференциальным уравнениям с простыми граничными условиями, во-вторых, требует тщательного подбора ширины, и наконец, не позволяет избавиться от эффекта переобучения. Приведен же он здесь для того, чтобы показать, что обучая центры, ширину и веса RBF-сети с помощью TR-метода (эксперимент № 2), можно достичь такой же точности решения, как и при сведении его к решению СЛАУ (эксперимент № 3).

Список литературы

1. **Хайкин С.** Нейронные сети: полный курс. М.: Вильямс, 2006. 1104 с.
2. **Васильев А. Н., Тархов Д. А.** Нейросетевое моделирование: Принципы. Алгоритмы. Приложения. СПб.: Изд-во Политехнического университета, 2009. 527 с.
3. **Kumar M., Yadav N.** Multilayer perceptrons and radial basis function neural network methods for the solution of differential equations: A survey // Computers and mathematics with applications. November 2011. N 62. P. 3796—3811.
4. **Горбаченко В. И., Артюхина Е. В.** Бессеточные методы и их реализация на радиально-базисных нейронных сетях // Нейрокомпьютеры: разработка, применение. 2010. № 11. С. 4—10.
5. **Choi B., Lee J.** Comparison of generalizing ability on solving differential equations using backpropagation and reformulated radial basis function networks // Neurocomputing. 2009. V. 73. P. 115—118.
6. **Jianyu L., Siwei L., Yingjiana Q., Yapinga H.** Numerical solution of elliptic partial differential equation using radial basis function neural networks // Neural Networks. 2003. 16 (5/6). P. 729—734.
7. **Горбаченко В. И., Артюхина Е. В., Артюхин В. В.** Радиально-базисные нейронные сети для решения краевых задач бессеточными методами // Нейроинформатика 2010: Сборник научных трудов XII Всероссийской научно-технической конференции. В 2-х ч. Ч. 2. М.: НИЯУ МИФИ, 2010. С. 237—247.
8. **Mai-Duy N., Tran-Cong T.** Numerical solution of differential equations using multiquadric radial basis function networks / // Neural Networks. 2001. V. 14. P. 185—199.
9. **Conn A. R., Gould N. M., Toint P. L.** Trust regions methods — MPS-SIAM series on optimization, 2000. 959 p.
10. **Дэннис Дж., Шнабель Р.** Численные методы безусловной оптимизации и решения нелинейных уравнений. М.: Мир, 1988. 440 с.
11. **Staihaug T.** The conjugate gradient method and trust region in large scale optimization // Society for industrial and applied mathematics, 1983.

CONTENTS

The article is dedicated to the 40th anniversary of the official recognition of evolutionary modeling as a promising scientific and practical direction in Russia. In this article the reasons of accelerated development of evolutionary modeling paradigm in Russian research are being defined.

Keywords: knowledge base, nested systems, integration, intellectualization, intellectual technology, integrity-evolutionary tools, evolutionary algorithms

Offers efficient approximate methods for solving Job-Shop-Problem. On the basis of the established properties of admissible and optimal schedules in the first steps of the algorithm is established fact that the consistency of the original system constraints, are computed lower bounds criterion of efficiency in the optimal solution. Algorithms for the solution of the problem is based on the construction of solutions at each step a feasible schedule of work for the most intense car and adjusting the earliest and the most acceptable of late start of each operation on the other machines, deadlines that have not yet been assigned.

Keywords: Job-Shop-Problem, the optimal solution, the approximate solutions, boundary dates of commencement of operations, properties of admissible and optimal schedules

This paper presents a consideration of some patterns theory elements used for engineering products modeling. Simulation objects have also been considered here. Graphic formalism of graph theory is used to visually represent pattern networks and their components.

Keywords: modeling, pattern networks, engineering products design, product family

The questions of the organization of information web-oriented schematic CAD database technologies. Domain model is proposed as a generalized ER-diagram, taking into account the inheritance hierarchy of models of circuit components. Additionally, the ER-diagram display mathematical relationships between the attributes of models at different levels. Shows the advantage of a mechanism to account for the stored procedures definable level dependencies of the database. Based on the developed ER-diagrams domain a proposed relational database models of circuit components (BDMK) in the medium of universal database Oracle 9i.

Keywords: database component models, ER-diagram, stored procedures, web-oriented CAD, database Oracle 9i

This paper presents the results of the design of the SIEM visualisation subsystem. Different visualisation techniques implemented in SIEM-systems are discussed. The architecture of the visualisation subsystem that allows easy extension and integration of different visualisation technologies is suggested. The proposed approach was implemented as software component. It demonstrates integration issues of the attack modelling component and graphical elements.

Keywords: security event and information management, security visualisation techniques, visualization component architecture

In this work is showed the protocol of the trusted encryption through Proxy Server on the basis of the modified Miller algorithm, calculating Weil pairing on a genus 2 hyperelliptic curves, in particular, on a supersingular curves with a high degree twist over a prime fields. The estimations of efficiency of the modified algorithm are received.

Keywords: bilinear pairings, Weil pairing, Tate pairing, algebraic curves, hyperelliptic curves, Jacobian, key generation, encryption, decryption

Results of analytical researches and data of computer experiment on reasons for a choice of a threshold of decision-making in case of demodulation of signals with differential phase shift keying are offered. For these purposes feasibility of the accounting of posterior probabilities about the demodulated characters is proved. The main stages of the algorithm implementing the developed scientific approach are represented.

Keywords: demodulation of signals, the adaptive threshold of decision-making, signals with the differential phase shift keying

The paper contains optimization erasure threshold when receiving pseudorandom signals. The work is aimed at improving the efficiency of information transfer. Optimizing the erasure threshold is from the perspective of maximizing the potential speed transmission with different variants of its assessment.

Keywords: communication, erase area, the physical layer, Hamming code, Golay code

Article presents an algorithm for near duplicate search based on binary templates. In this method image is hierarchically divided into subparts and binary characteristics of these parts are calculated. By using integral image time needed to build image signature is short. Whether images are similar or not is concluded by comparing image signatures in a bitwise way. To evaluate this algorithm, its time and quality parameters are compared with Colorgrid and Surf algorithms.

Keywords: near duplicate image search, fuzzy image comparison, integral image

The dependability is a generic notion with a set of theories and investigations around. The paper defines the structure of a dependability notion and related questions may arise during design phase of computer systems with information, communication or control services. Certain associated with dependability techniques and methods are mentioned. Some relations of the computer system dependability notion structural components with other technical or human areas are noted.

Keywords: computer system, software system, dependability, reliability, safety, maintainability, security

On the base of knowledge obtained in a multi-server database with circulating segments modeling the information system is build. In base of it system the apparatus of a logical neural network is lies. The system allows unlimited development, modification, use of fuzzy data.

Keywords: a multi-server database with circulating segments, the intensity of the flow of requests, the average query execution time, base of knowledge, logical neural network

The history of development of scaled super computers with two-layer architecture and their assessment from the point of view of efficiency of modeling of difficult neural networks is presented. The leading role by the supercomputer on the basis of graphic processors in a problem of modeling of neural networks is noted now. The concept of development of superneurocomputers — the super computer on the basis of neural network technologies is presented.

Keywords: neural network technologies, superneurocomputers, two-layer architecture, scalability

The learning algorithm of radial basis function network using trust region method was developed. There is shown in the article, how Poisson's equation can be solved with this algorithm. In accordance with the experimental results our algorithm is more accurate the earlier used algorithms.

Keywords: radial basis function network, radial basis function network learning, trust region method, poisson's equation

Адрес редакции:

107076, Москва, Стромынский пер., 4

Телефон редакции журнала **(499) 269-5510**

E-mail: it@novtex.ru

Дизайнер *Т.Н. Погорелова*. Технический редактор *Е. В. Конова*.

Корректор *Т. В. Пчелкина*.

Сдано в набор 05.07.2013. Подписано в печать 28.08.2013. Формат 60×88 1/8. Бумага офсетная.

Усл. печ. л. 8,86. Заказ IT913. Цена договорная.

Журнал зарегистрирован в Министерстве Российской Федерации по делам печати, телерадиовещания и средств массовых коммуникаций.

Свидетельство о регистрации ПИ № 77-15565 от 02 июня 2003 г.

Оригинал-макет ООО "Адвансед солюшнз". Отпечатано в ООО "Адвансед солюшнз".

105120, г. Москва, ул. Нижняя Сыромятническая, д. 5/7, стр. 2, офис 2.