МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
ОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
им. Ф.М. ДОСТОЕВСКОГО

МАТЕМАТИЧЕСКОЕ И КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ

Сборник материалов научной конференции

(Омск, 18 октября 2013 г.)

Омск

2013

УДК 004+519
ББК 22.18я43+32.973я43
М 340

Рекомендовано к изданию
редакционно-издательским советом ОмГУ

Ответственные за выпуск:
канд. физ.-мат. наук, доцент *И.П. Бесценный*;
канд. техн. наук, доцент *Д.Н. Лавров*

М 340 Математическое и компьютерное моделирование : сборник материалов научной конференции (Омск, 18 октября 2013 г.). – Омск : Изд-во Ом. гос. ун-та, 2013. – 112 с.

ISBN 978-5-7779-1644-0

Включены материалы докладов, сделанных на научной конференции «Математическое и компьютерное моделирование», которая состоялась на факультете компьютерных наук ОмГУ 18 октября 2013 года.

УДК 004+519
ББК 22.18я43+32.973я43

ISBN 978-5-7779-1644-0

Содержание

УДК 519:159.947

А.К. Гуц

Омский государственный университет им. Ф.М. Достоевского

МОДЕЛЬ МАТЕРИАЛИЗАЦИИ МЫСЛЕЙ

Мысль человека i − это квантовая «частица», описываемая вектором $|\chi_i\rangle$, лежащим в абстрактном гильбертовом пространстве H, и являющаяся частицей окружения, внешней среды макроскопического объекта B, скажем, каменного шара. Состояние шара с центром масс в пространственной точке x − это вектор $|x\rangle$. Шар как квантовый объект может иметь разную пространственную локализацию, т. е. находится в разных местах пространства. Поэтому его состояние должно описываться в виде квантовой когерентной суперпозиции

$$\sum_{x\in \mathbb{R}^3} c_x |x\rangle, \tag{1}$$

а, точнее, с помощью матрицы плотности $\hat{\rho}$ квантовой системы $H_i \times H_B$, где H_i − гильбертово пространство состояний мыслей субъекта i, а H_B − гильбертово пространство состояний шара B. Удобно использовать так называемую редуцировнную матрицу плотности $\hat{\rho}_B(x,x',t)$.

Когерентная суперпозиция – это суперпозиция состояний, которые не могут быть реализованы одновременно с классической точки зрения. Важно отметить, что суперпозиция − это нелокальное состояние, в котором каменного шара, как локального элемента классической реальности, нет, не существует.

Желание субъекта i увидеть шар с центром в точке x − это трата психической энергии субъекта, означающая изменение, коррекцию мысли, помечаемую как успешную или неуспешную. Иначе говоря, имеющую измененное состояние $|\chi_i(x)\rangle$. Сам процесс материализации шара B представляет собой оператор $|\chi_i\rangle \to S|\chi_i(x)\rangle$.

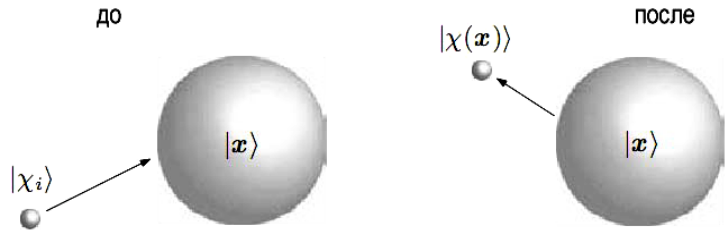


Рис. 1. Мысль о шаре удачна или неудачна (рис. из [1])

Субъект i', если и имеет такое же желание материализовать шар, все-таки, как следует допустить, видит шар с центром в точке x' $(x' \neq x)$.

Мысли большого числа субектов о материальном шаре в пространстве создают *Внешнюю среду* (*external environment*), взаимодействие с которой «квантового шара» (1) приводит к тому, что все субъекты видят шар локализованным (материализованным) в конкретной точке x_0

$$\sum_{x\in \mathrm{IR}^3} c_x \,|\, x\rangle \qquad \overset{\text{External environment}}{\underset{\Downarrow}{\rightarrow}} \qquad |\, x_0\rangle .$$

с вероятностью $|\, c_{x_0} \,|^2$. Шар появляется, материализуется как «плотное тело». Этот процесс называется декогеренцией квантовой суперпозиции (1). На языке матрицы плотности это происходит в том случае, когда $\hat{\rho}_B(x, x', t) \to 0$ при $t \to +\infty$. Мы имеем пространственную локализацию квантового объекта «шар».

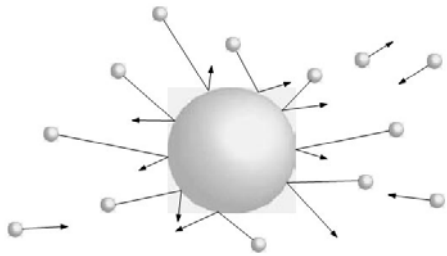


Рис. 2. Мысли о шаре множества субъектов создают внешюю среду (рис. из [1])

При ряде допущений для локализации шара с радиусом a получена формула [1]:

$$\hat{\rho}_B(x,x',t) = \hat{\rho}_B(x,x',0)e^{-\Lambda|x-x'|^2 t},$$

где

$$\Lambda \approx 10^{20} \frac{1}{cm^2 s}\left(\frac{a}{cm}\right)^6\left(\frac{T}{K}\right)^9.$$

Отсюда получаем, что шар радиуса 1 *м* материализуется за 10^{-50} *сек*. Таким образом, мысли людей о том, что в окружающем их пространстве появляется, скажем, каменный шар, действительно могут привести к тому, что из ничего вдруг материализуется реально каменный твердый большой шар? Воля и физика вполне допускают синтез для рационального мышления.

ЛИТЕРАТУРА

1. Schlosshauer M. Decoherence and the quantum-to-classical transition. Berlin-Heidelberg: Springer-Verlag, 2007. 192 p.

УДК 519.83+681.3.067

А.К. Гуц
Омский государственный университет им. Ф.М. Достоевского

ТЕОРИЯ ИГР, РАВНОВЕСИЯ НЭША И ЗАКОНОДАТЕЛЬСТВО В СФЕРЕ КОМПЬЮТЕРНЫХ ПРЕСТУПЛЕНИЙ

Если администратор компьютерного ресурса имеет противостояние с несколькими злоумышленниками, действующими разрозненно, то мы имеем бескоалиционную игру. Как известно, в таком случае возможна ситуация равновесия по Нэшу в смешанных стратегиях. Иначе говоря, и администратор и злоумышленники в таком случае какое-то время сохраняют неизменными свои стратегии поведения, и в силу этого каждая сторона имеет устраивающий ее максимальный выигрыш. Все довольны и никто не пытается разрушить установившуюся ситуацию.

Ясно, что подобная ситуация возникает в том случае, когда злоумышленник «пробил» защиту, обосновался в сети, но не наносит ущерб хозяину сети. Администратор обнаружил несанкционированное проникновение в сеть, но не знает, как избавиться от «гостя» или знает, но опасается, что его действия могут спровоцировать злоумышленника на нанесение серьезного ущерба сети. Каковы рекомендации следует дать администратору в данном случае?

Пожалуй, следует напомнить ему Уголовный кодекс РФ:

Статья 274. Нарушение правил эксплуатации средств хранения, обработки или передачи компьютерной информации и информационно-телекоммуни-кационных сетей

1. Нарушение правил эксплуатации средств хранения, обработки или передачи охраняемой компьютерной информации либо информационно-телекоммуникационных сетей и оконечного оборудования, а также правил доступа к информационно-телекоммуникационным сетям, повлекшее уничтожение, блокирование, модификацию либо копирование компьютерной информации, причинившее крупный ущерб, − наказывается штрафом в размере до пятисот тысяч рублей или в размере заработной платы или иного дохода осужденного за

период до восемнадцати месяцев, либо исправительными работами на срок от шести месяцев до одного года, либо ограничением свободы на срок до двух лет, либо принудительными работами на срок до двух лет, либо лишением свободы на тот же срок.

2. Деяние, предусмотренное частью первой настоящей статьи, если оно повлекло тяжкие последствия или создало угрозу их наступления, − наказывается принудительными работами на срок до пяти лет либо лишением свободы на тот же срок.

Как видим, «мирное сосуществование» с злоумышленником, − это, как минимум, нарушение правил доступа к информационно-телекоммуникационным сетям, действующих в организации или учреждении, создающее угрозу наступления тяжких последствий, что влечет привлечение администратора к уголовной ответственности.

УДК 004.9:631.4+519.83

А.К. Гуц, Л.А. Володченкова
Омский государственный университет им. Ф.М. Достоевского

ТЕОРЕТИКО-ИГРОВОЙ ПОДХОД ПРИ ПЛАНИРОВАНИИ МЕРОПРИЯТИЙ ПО ЗАЩИТЕ ЛЕСА

Защита лесных насаждений является важной задачей лесных управлений регионов. Если взглянуть на отношения Природы и Лесного управления с точки зрения теории игр, то убытки Лесного управление – это удачный ход Природы, а убыток – выигрыш игрока именуемого «Природа». В теории игр игра с «Природой» – это игра в условия неопределенности, т. е. отсутствия полной информации о стратегиях игрока «природа». У игрока «Природа» могут быть возможности сделать тот или иной «удачный» ход, который является не чем иным как убытком игрока «Лесное управление». Можно перечислить разные стратегии поведения обеих сторон, сводящиеся к тому, что они в соответствии с выбранной стратегией делают свои ходы, которые означают выигрыш «Лесного управления», когда лесозащитное мероприятие было своевременным и деньги потрачены не зря, либо проигрыш, когда «Природа», «выбрав» свою стратегию, сделала ход, который нанес убыток «Лесному управлению». Игру можно рассматривать как игру с нулевой суммой, когда выигрыш одного игрока равен проигрышу другого. Однако вполне допустимо рассмотреть проблемы организации защиты леса как биматричную игру. Стратегии игрока «Природа»: L_f – спонтанные лесные пожары; L_{np} – непатогенные факторы, которые вызывают ослабление состояния деревьев, но не приводят насаждения к гибели. В эту группу факторов отнесены: межвидовая и внутривидовая конкуренция, затенения, охлест, ошмыг и другие; L_p – размножение вредных организмов и болезни леса; L_a – антропогенные воздействия на лес и вытаптывание животными; L_w – неблагоприятные погодные условия и почвенно-климатические факторы. Стратегии игрока «Лесное управление»: LZ_f – противопожар-

ные мероприятия; LZ_{les} – защита лесов от вредных организмов и болезней; $LZ_{restore}$ – восстановление лесных ресурсов; LZ_{clear} – уход за лесами. Матрица выигрышей составляется с использованием цифр, планируемых расходов на противопожарные мероприятия.

Проведенные игры позволяют выявить ситуации равновесия Нэша и выявить оптимальные стратегии.

УДК 004.9:631.4

Л.А. Володченкова
Омский государственный университет им. Ф.М. Достоевского

УПРАВЛЕНИЕ КАТАСТРОФАМИ ЛЕСНЫХ ЭКОСИСТЕМ

Своевременное предупреждение наступающей экологической катастрофы лесного фитоценоза является важнейшей задачей любого лесного регионального управления. Для этого необходимо постоянно снабжать ответственных работников этих управлений данными о состоянии лесов. Это достигается, например, с помощью мониторинга состояния лесных экосистем.

В последнее время проблемы прогнозирования состояния лесных экосистем приобретают особую остроту в связи с расширением масштабов антропогенного влияния на лесные ландшафты. Процессы вымокания леса, лесные пожары по вине людей оказывают значительное воздействие на окружающую среду. В Омской области вымокание и пожары являются основными причинами гибели леса. Создание новых методов описания и прогнозирования лесных катастроф, таких как вымокание и пожары, в силу сказанного следует рассматривать как важнейшую задачу экологии.

Рассматривается следующая теоретико-катастрофическая мозаично-ярусная математическая модель лесного фитоценоза:

$$\frac{dx}{dt} = \frac{\partial}{\partial x} V(x,k,m,a,w),$$

$$V(x,k,m,a,w) = \frac{\alpha}{6}(x-x_0)^6 + k(x-x_0)^4 + m(x-x_0)^3 + a(x-x_0)^2 + w(x-x_0),$$

$$k = -c_k(CI - CI_0),\ \ m = c_m(\frac{s^2}{\mu} - 1),$$

$$a = -c_a(УАН - УАН_0),\ \ w = c_w(W - W_0),$$

где x – продукция фитомассы (т/га за год), CI – индекс конкуренции Вайса; s^2/μ – коэффициент дисперсии, являющийся показателем равномерности распределения деревьев в пространстве; если s^2/μ близко к нулю, то распределение регулярное, к единице – случайное, а

чем более единицы, – тем мозаичнее; $УАН$ – уровень антропогенной нагрузки на район, равный отношению степени антропогенного воздействия к биоклиматическому потенциалу, введенный П.В. Большаником и Н.О. Игенбаевой (он позволяет учитывать все типы вредного антропогенного воздействия на лесное насаждение); W – влажность почвы; c_k, c_m, c_a, c_w – постоянные коэффициенты, $\alpha = \alpha_1\alpha_2\alpha_3\alpha_4$ и α_j – доля фитомассы *j*-го яруса. Величины CI_0, $УАН_0$, W_0 – это критические значения факторов, обозначающие границы экологической устойчивости фитоценоза.

С помощью предложенной модели выводятся следующие бифуркационные соотношения:

$$\omega = \alpha x^5 + 4kx^3 + 3mx^2 + 2ax + w = 0$$
$$\Omega = 5\alpha x^4 + 12kx^2 + 6mx + 2a = 0. \qquad (1)$$

На соотношениях (1) предлагается основывать:

1) методику мониторинга лесной экосистемы, направленную на прогнозирование экологических катастроф и уровня деградации лесных экосистем;

2) методику оценки границ предельного антропогенного воздействия на лесной фитоценоз.

Порядок проведения мониторинга лесной экосистемы, направленного на прогнозирование экологических катастроф и уровня деградации лесных экосистем, предполагает выполнение следующих процедур:

– периодическое измерение для конкретной лесной экосистемы величин $\alpha, CI, s^2/\mu, УАН, W, x$;

– вычисление величин ω, Ω;

– если каждый раз наблюдается уменьшение ω и Ω, т. е. $\omega \to 0$ и $\Omega \to 0$, то это говорит о приближении экологического кризиса и об изменении уровня деградации лесной экосистемы.

Формулы (1) позволяют также оценить границы предельного антропогенного воздействия на лесную экосистему. Делается это следующим образом:

– периодически определяется величина $УАН$;

– найденное значение $УАН$ подставляется в формулы (1);

– если наблюдается уменьшение ω и Ω, то это говорит об усилении опасного антропогенного воздействия на лесную экосистему;

катастрофа наступит, если ω и Ω поменяют знак на противоположный. Значения величины $УАН$, при котором это наступает и являются предельно допустимыми.

Для реализации на практике высказанных предложений необходимо осуществлять мониторинг фитомассы ярусов, конкуренции, степени мозаичности леса, уровня антропогенной нагрузки на район, влажности почвы и годичной продукции фитомассы.

УДК 519.766.4

О.А. Кожуховская, А.Д. Кожуховский
Черкасский государственный технологический университет (Украина)

ВЕРОЯТНОСТНОЕ МОДЕЛИРОВАНИЕ ОПЕРАЦИОННЫХ РИСКОВ С ИСПОЛЬЗОВАНИЕМ БАЙЕСОВСКИХ СЕТЕЙ

Введение. Байесовские сети (БС) [1; 2] дают возможность отобразить у модели причинно-следственные связи между разными факторами риска и изменениями среды. В отличии от регрессионных моделей байесовские сети позволяют учитывать не только непосредственные зависимости уровня риска от факторов риска, но также и зависимости между факторами риска. Кроме этого, этот класс моделей предоставляет больше возможностей для формирования заключения на основе неполных данных. С математической точки зрения БС – ориентированный граф, где вершинам соответствуют факторы риска и изменение среды, а ребрам соответствуют обнаруженные или предусмотренные взаимосвязи. Сеть также описывается множеством условных распределений случайных величин, которые характеризуют факторы риска и переменные среды.

Постановка задачи. Допустим, что существуют n случайных величин $X_1, ..., X_n$. Совокупную вероятность значений этих величин можно выразить через произведение n условных вероятностей:

$$P(x_1, ..., x_n) = P(x_1) \prod_{j=2}^{n} P(x_j \mid x_1, ..., x_{j-1}). \quad (1)$$

Если допустить, что случайная величина X зависит не от всех предыдущих случайных величин (величин с меншими индексами), а только от их части, которую мы обозна чим как PA_j, тогда:

$$P(x_j \mid x_1, ..., x_{j-1}) = P(x_j \mid pa_j).$$

Отсюда вытекает, что формулу (1) можна переписать в следующем виде:

$$P(x_1, ..., x_n) = \prod_{j=1}^{n} P(x_j \mid pa_j). \quad (2)$$

Множество PA_j называют множеством отцовских величин для случайной величины X.

Преимуществом БС есть возможность одновременного использования экспертного оценивания (например, для оценивания структуры сети путем определения зависимостей между переменными), и математических методов для получения заключения по сети. За счет этого модель дает возможность связывать выборки статистических данных с экспертными знаниями. Заключение на основе БС может осуществляться путем распространения информации в любом направлении. БС используют для формирования вероятностного заключения – расчета условной вероятности получения значений для части случайных величин при условии известных значений других величин. Математически задание можна сформулировать как вычисление $P(y \mid x)$, где X – множество наблюдаемых значений, а Y – множество переменных, которые необходимо оценить.

Заключение можна осуществлять непосредственно с использованием формулы Байеса и операций маргинализации – вычисления сумм по реализациям всех переменных, кроме выбранных. В этом случае задание сводится к расчету условных вероятностей по формуле:

$$P(y \mid x) = \frac{\sum_s P(y,x,s)}{\sum_{y,s} P(y,x,s)}, \qquad (3)$$

где S – множество всех переменных за исключением X и Y.

Этот метод трудоемкий, то есть, задание формирования заключения с использованием БС есть NP – полным. Поэтому для БС предложено множество более эффективных алгоритмов формирования заключения [3; 4]. Рассмотрим один из этих алгоритмов.

Модель страхового шахрайства. Для построения модели и осуществления заключения с учетом невозможности использования стандартных методов разработан алгоритм, который состоит из таких шагов: 1 – подготовка данных; 2 – анализ эмпирических распределений; 3 – обнаружение припрятанных взаимосвязей; 4 – пояснение обнаруженных взаимосвязей; 5 – определение частицы шахрайства среди убытков с заданными параметрами; 6 – формирование критериев подозрительности; 7 – оценивание убытков по выборке.

При подготовке данных целесообразно формировать выборку по договорам с одинаковыми сроками действия и видами рисков, ко-

торые перекрываются. На втором этапе алгоритма строятся эмпирические распределения переменных, которые включены до байесовской сети на основании выборки, которая сформирована в результате выполнения первого этапа. Построенные распределения – это предмет для экспертного анализа, который направлен на выявление общих закономерностей.

На третьем этапе осуществляется выбор одной переменной при допущении об отсутствии шахрайства. Выполняется анализ условных распределений этой переменной при заданных значениях каждой из оставшихся переменных. Условные распределения сравниваются с безусловным распределением и, в случае выявления отличий, делается допущение о наличии припрятанных взаимосвязей.

Четвертый шаг предусматривает проведение экспертного анализа обнаруженных взаимосвязей и их сравнение с обобщенными сценариями шахрайства. В результате каждая из обнаруженных взаимосвязей может поясняться за счет реализации одного или нескольких сценариев шахрайства или за счет естественных взаимосвязей.

В рамках пятого этапа алгоритма для каждой обнаруженной зависимости определяется количество «обычных убытков» (которые возникли вследствие действия застрахованных рисков) и «убытков с признаками шахрайства» (которые связаны с перекручиванием страховальщиком информации о сроках или обстоятельствах их появления). Оценивание выполняется отдельно для каждой пары переменных, между которыми обнаружена припрятанная взаимосвязь. При оценивании сравниваются между собой условные и безусловные распределения, а количество случаев шахрайства устанавливается таким образом:

$$Q\{(F=1)\,|\,(A=a),(B=b)\} =$$
$$= Q\{A=a)\,|\,(B=b\} - Q\{A=a\}\,Q\{b=b\}/N,$$

где $Q\{(F=1)\,|\,(A=a),(B=b)\}$ – количество убытков с признаками шахрайства среди убытков, для которых параметр A принимает значение a и параметр B принимает значение b; $Q\{A=a\,|\,B=b\}$ – общее количество убытков; $Q\{A=a\}$ – количество убытков, для которых параметр A принимает значение a; $Q\{B=b\}/N$ – оценка вероятности того, что параметр B принимает значение b; $Q\{B=b\}$ – количество убытков с таким свойством; N – общее количество значений убытков у выборке). Это выражение основывается на допуще-

нии о том, что при отсутствии шахрайства переменные должны быть независимыми, а условное распределение совпадает с безусловным.

На шестом этапе алгоритма формируются критерии подозрительности на наличие шахрайства на основании оцененных убытков:

$$K_j : s_j \to [0; 1],\ j = 1,...,N;\ i = 1,...,T;\ s_j = (s_j^1, ..., s_j^m),$$

$$K_i = \left\{ k_{ji_1}^{i,l}, ..., k_{ji_{wi}}^{i,l},\ p^{i,l} \right\},\ l = 1,...,r_i,$$

где K_i – i-й критерий подозрительности; T – количество критериев; s_j – j-й элемент выходной выборки; N – количество элементов выборки; m – количество характеристик убытка (количество позиций в строке), которые включены в элемент-строку выборки; $j_{i_1},..., j_{i_{wi}}$ – номера позиций со строк s_j, на базе которых по критерию K_i определяется вероятность шахрайства; w_i – количество характеристик, которые используются i-м критерием; r_i – количество множеств значений характеристик, которые используются i-м критерием; $\left\{ k_{ji_1}^{i,l}, ..., k_{ji_{wi}}^{i,l},\ p^{i,l} \right\}$, – множество значений позиций и условной вероятности наличия шахрайства при условии равности этим значениям соответствующих позиций строки s_j. Значения критерия K_i на этом элементе выборки определяется так:

$$K_i(s_j) =$$

$$= \begin{cases} p^{i,l}, \textit{ если } \exists l \in \{1,...,r_i\} \textit{ такая, что } \forall t \in \{j_{i_1},...,j_{i_{wi}}\}: s_j^t = k_t^{i,l}; \\ 0, \quad \textit{в ином случае.} \end{cases} \quad (4)$$

Поскольку для некоторых видов убытков могут быть получены положительные значения для нескольких критериев, то совместная вероятность наличия шахрайства с даного убытка может быть оценена на седьмом этапе как максимальное из наличных значений критериев:

$$\hat{P}\left(PZ_j = Sh \mid s_j \right) = \max\left\{ K_1(s_j), ..., K_T(s_j) \right\},$$

где PZ_j – причина j-го убытка; $\hat{P}\left(PZ_j = Sh \mid s_j \right)$ – оценка вероятности наличия шахрайства у j- м убытке; если $Sh = 1$, то имело место шахрайство; $K_i(s_j)$ – значения i-го критерия для j-го убытка.

Величину, рассчитанную по выражению (4), можна трактовать как оценку математического ожидания индикаторной случайной величини, равной 1, если убыток s_j повязанный из шахрайством:

$$\hat{P}\left(PZ_j = Sh \mid s_j \right) = E\left[I_{Sh}(s_j) \right], \qquad (5)$$

где $I_{Sh}(s_j) = 1$ тогда и только тогда, когда убыток s_j повязанный с шахрайством; $I_{Sh}(s_j) = 0$ в других случаях. Если принять во внимание, что наступления отдельных убытков можна считать независимыми случайными событиями, то из (4) и (5) следует, что:

$$\hat{N}_{Sh} = \sum_{j=1}^{N} E\left[I_{Sh}(s_j) \right] = \sum_{j=1}^{N} \max\left\{ K_1(s_j), ..., K_T(s_j) \right\},$$

где $\hat{N}_{Sh}$ – оценка количества случав шахрайства у имеющейся выборке данных.

Для расчета величины потерь, повязанных с шахрайством, рассмотрим величину

$$V_{Sh}(s_j) = \begin{cases} V(s_j), \textit{ если убыток } s_j \textit{ повязанный с шахрайством}; \\ 0, \textit{ в ином случае}. \end{cases}$$

где $V(s_j)$ – величина выплаты по убытку j. Эту величину можна выразить через индикатор таким образом: $I_{Sh}(s_j): V_{Sh}(s_j) = I_{Sh}(s_j)\, V(s_j)$.

Если обозначить через $\hat{V}_{Sh}$ общий размер потерь, повязанных с шахрайством, то его можна оценить как математическое ожидание:

$$\hat{V}_{Sh} = \sum_{j=1}^{N} E\left[V_{Sh}(s_j) \right] = \sum_{j=1}^{N} E\left[I_{Sh}(s_j)\, V(s_j) \right] =$$

$$= \sum_{j=1}^{N} \max\left\{ K_1(s_j), ..., K_T(s_j) \right\} V(s_j).$$

Заключение по построенной модели выполнено на основании данных отрегулированных убытков, которые наданы страховой компанией (СК) (размер выборки после предыдущей обработки составил 2157 элементов). Обнаруженные зависимости поясняются реализацией трех обобщенных сценариев шахрайства: «Страхования поврежденной машины с дальнейшей фальсификацией убытка», «Страхования «задним» числом» и «Перекручивания обстоятельств события».

Так, анализ показал, что существует зависимость между переменными «Срок убытка» и «Прибыль». При условии наступления убытка в первые месяцы действия договора страхования вероятность того, что переменная «Прибыль» принимает значения $Z_r^{\ l}$ (например, 10000) і $Z_r^{\ h}$ (сверх 20000), будет высшей, чем без задавания условия на срок наступления убытка:

$$P(Zr \in \{Zr^l > Zr^h\} \mid t_z \in \{1,2,3,4\}) > \ P(Zr \in \{Zr^l > Zr^h\}),$$

где t_z – срок наступления убытка. Такое повышение вероятности может быть пояснено сценариями шахрайства «Страхования поврежденной машины с дальнейшей фальсификацией убытка» или «Страхования «задним числом».

Заключение. Построена модель у форме байесовской сети для описания операционных актуарных рисков, которая отображает причинно-следственные связи между факторами риска и потерями страховой компании. Эта модель имеет широкий спектр применения: – для анализа состояния внутренней среды страховой компании; – для анализа условий, в которых осуществляет свою деяльность компания; – для принятия управленческих решений и анализа возможных результатов от их реализации. БС применена также для определения вероятной причины потерь страховой компании, повязанных с операционными рисками.

Дальнейшие исследования будут направлены на усовершенствования методик моделирования и оценивания рисков у страховании, расширения номенклатуры математических моделей для их описания, а также сравнительному анализу существующих методов.

ЛИТЕРАТУРА

1. *Neil M., Fenton N. E., Tailor M.* Using bayesian networks to model expected and unexpected operational losses // Risk Analysis. 2005. P. 34–57.

2. *Згуровский М. З., Терентьев О. М., Бидюк П. И.* Методы построения байесовских сетей на основе оценочных функций // Кибернетика и системный анализ. 2008. № 2. С. 81–88.

3. *Cooper G. F.* The computational complexity of probabilistic inference using bayesian belief networks // Artificial Intelligence. 1990. № 42. P. 393–405.

4. *Guo H.*, *Hsu W.* A survey on algorithms for real-time bayesian network inference // Laboratory for Knowledge Discovery in Databases Department of Computing and Information Sciences, Kansas State University. 2002. 20 p.

УДК 519.688

И.Е. Евдокимов
НИЦ АО «Вертолёты России», НИУ МАИ, г. Москва

М.Д. Калугин
ИСП РАН, г. Москва

ИСПОЛЬЗОВАНИЕ ОТКРЫТОГО ПАКЕТА OPENFOAM И ОБЛАЧНЫХ ВЫЧИСЛИТЕЛЬНЫХ СЕРВИСОВ UNIHUB ДЛЯ РЕШЕНИЯ ЗАДАЧ МЕХАНИКИ СПЛОШНОЙ СРЕДЫ

Введение. Общие сведения о пакете OpenFOAM

OpenFOAM – свободно распространяемый инструментарий вычислительной гидродинамики для операций с полями (скалярными, векторными и тензорными). На сегодня является одним из «законченных» и известных приложений, предназначенных для FVM-вычислений.

Код OpenFOAM, разработан в Великобритании, в компании OpenCFD, Limited, и используется многими промышленными предприятиями более 12 лет. Свое название и идеологию построения код берет от предшественника FOAM (Field Operation And Manipulation), который является закрытым и продолжает развиваться параллельно с OpenFOAM. Первоначально программа предназначалась для прочностных расчетов и в результате многолетнего академического и промышленного развития на сегодняшний момент позволяет решать следующие задачи:

1. Прочностные расчеты;
2. Гидродинамика ньютоновских и неньютоновских вязких жидкостей как в несжимаемом, так и сжимаемом приближении с учётом конвективного теплообмена и действием сил гравитации. Для моделирования турбулентных течений возможно использование RANS-моделей, LES- и DNS-методов;
3. Возможно решение дозвуковых, околозвуковых и сверхзвуковых задач;
4. Задачи теплопроводности в твёрдом теле;
5. Многофазные задачи, в том числе с описанием химических реакций компонент потока;

6. Задачи, связанные с деформацией расчётной сетки;

7. Сопряжённые задачи;

8. Некоторые другие задачи, при математической постановке которых требуется решение дифференциальных уравнений в частных производных в условиях сложной геометрии среды;

9. Распараллеливание расчёта как в кластерных, так и многопроцессорных системах.

При этом пакет может совмещаться с другими разработанными открытыми и закрытыми программными комплексами для инженерных расчётов. Связь OpenFOAM с ними показана на рис. 1.

На рис. 2 а, б приведены результаты пробных расчётов в программном пакет OpenFOAM с использованием утилиты для построения сеток SnappyHexMesh (распространяется также, на условиях лицензии GPL) на сервере unihub.ru. Постпроцессинг задачи осуществлялся в свободной программе ParaView.

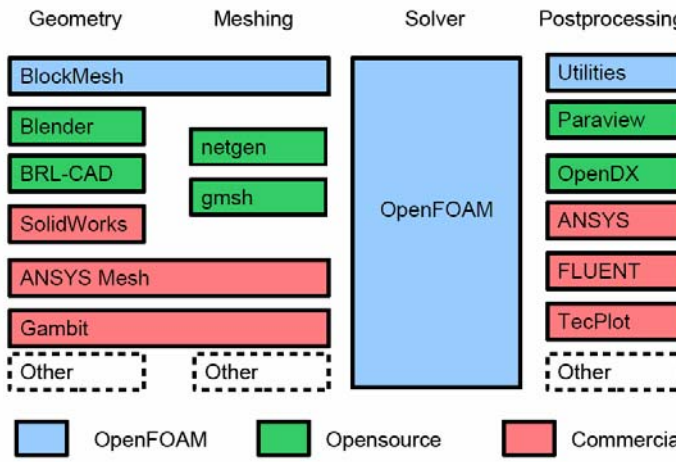
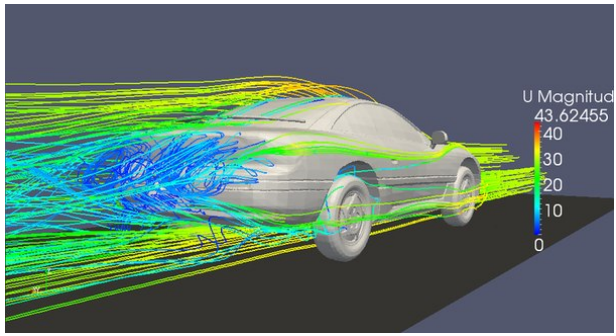
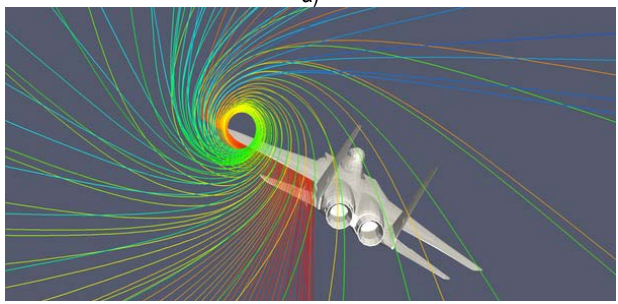


Рис. 1. Схема связи OpenFOAM с дополнительными пакетами для пре- и постпроцессинга



а)



б)

Рис. 2. Расчёт обтекания автомобиля марки Mitsubishi (а), пробный расчёт обтекания истебителя F-15 на малой скорости (б)

Использование OpenFOAM в рамках лаборатории unihub.ru

Схема работы с расчётным комплексом OpenFOAM может быть диверсифицирована по отношению к располагаемым вычислительным ресурсам, в соответствии с рисунком 3, в цикле вычислений могут быть использованы:

1. Собственные вычислительные мощности (начиная от мощного персонального компьютера – минимум 8 ядерный процессор с 16 Гб ОЗУ с мощной видеокартой для визуализации решений и значительным жёстким диском для хранения данных);

2. Мощности по запросу – использование облачных технологий (частные компании, например «Т-сервисы»);

3. Сотрудничество с академической средой, использование ресурсов Unihub.ru (программа «Университетский кластер») – в настоящее время является доступным и используемым. Тестовый расчёт –

обтекание модели F-15 вблизи Земли на скорости, близкой к посадочной, 25 млн элементов, обтекание автомобиля.

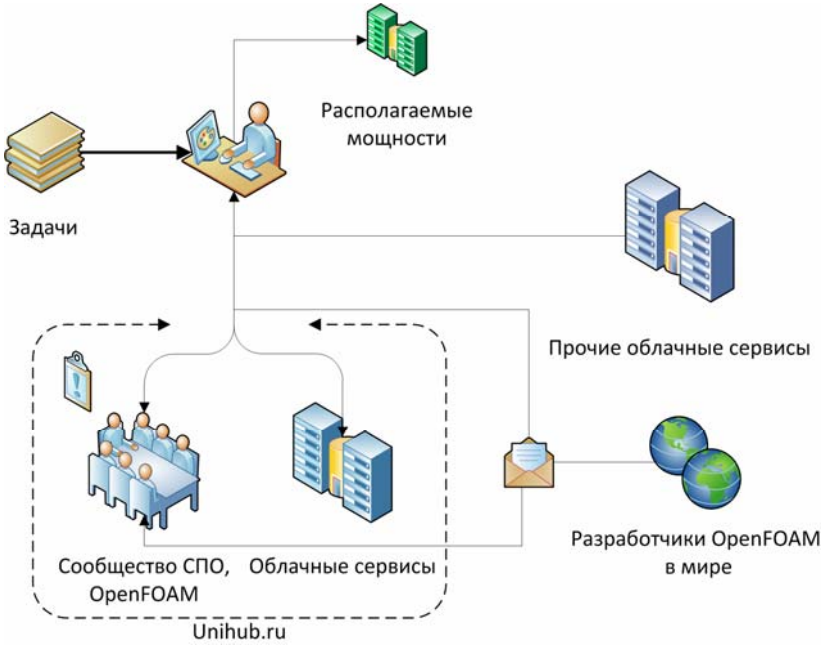


Рис. 3. Схема работы с применением OpenFOAM

Кроме того, использование открытого пакета позволяет сотрудничать в рамках общих целей с многочисленными разработчиками в мире, и небольшим сообществом разработчиков и пользователей OpenFOAM в России (ИСП РАН, Институт им. Курчатова, МГТУ им. Баумана, СПбГТУ, и т. д.).

Реализация расчётов в прикладной области

На рис. 4, 5 демонстрируются результаты расчётов аэродинамических характеристик плоских бесконечных профилей.

В результате проведённых расчётов демонстрируется хорошее совпадение с экспериментальными данными как в случае расчёта течения несжимаемого дозвукового потока, так и в случае сжимаемого трансзвукового потока.

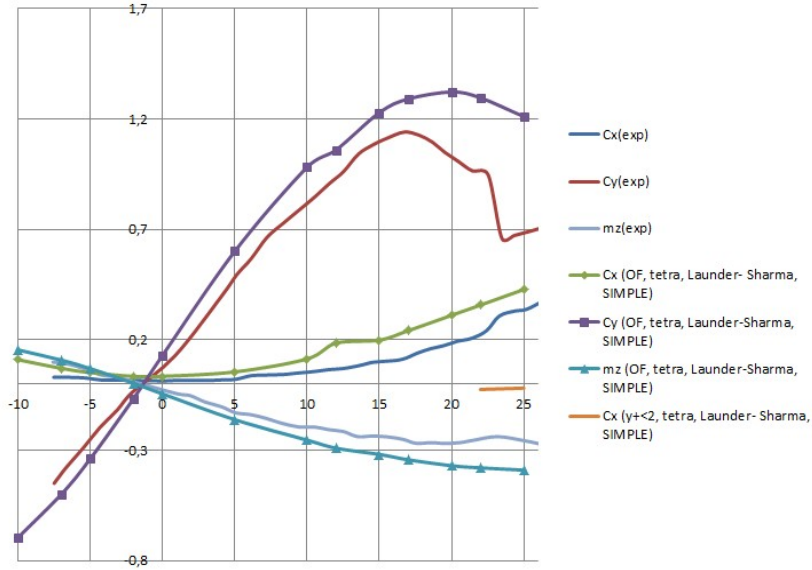


Рис. 4. Результаты расчёта характеристик профиля NACA23012 на модели турбулентности LowRe Launder-Sharma k-epsilon в зависимости от угла атаки

Так как в ряде задач расчёта течения плоского профиля экспериментальные данные могут быть недостаточно точными (вследствие пересчёта характеристик конечного крыла на бесконечный профиль), было проведено численное моделирование эксперимента по обтеканию крыла в трубе (рис. 6). Согласно полученным данным, расчётные результаты, полученные для реального крыла в трубе оказываются более точными по сравнению с расчётом бесконечного профиля. Данная задача носит более глубокий характер и требует привлечения значительных вычислительных ресурсов. Численная модель крыла в трубе насчитывала порядка 13 млн. элементов, и рассчитывалась на 48 ядрах кластера JSCC лаборатории unihub.ru.

Кроме задач общей аэродинамики, исследуется возможность использования пакета OpenFOAM для технологических расчётов производства композиционных материалов. В рамках данного пакета производится доработка решателей и разработка дополнительных библиотек для описания свойств смол. На данный момент реализована библиотека, позволяющая задавать температурно-зависимую вязкость. В решении данной задачи была продемонстрирована гибкость

пакета OpenFOAM, как математического программного обеспечения, позволяющего решать уравнения в частных производных. Пробные расчёты (рис. 7) показывают практическую возможность осуществления технологических расчётов, однако данная тема требует всесторонней и глубокой проработки вследствие специфических проблем, имеющих место при расчёте сильно вязких, ползучих сред или расчёте их совместного течения с газами, например, воздухом.

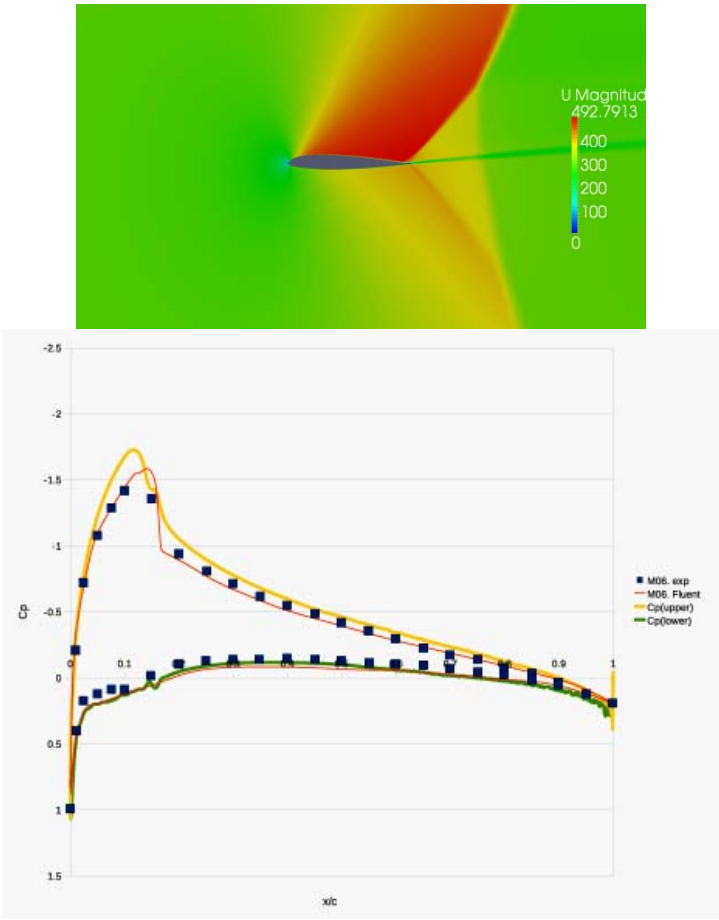


Рис. 5. Расчётное распределения скорости среды вблизи профиля при трансзвуковом обтекании (а), расчётное распределение давления по поверхности профиля (сравнение с экспериментом и пакетом Fluent, б)

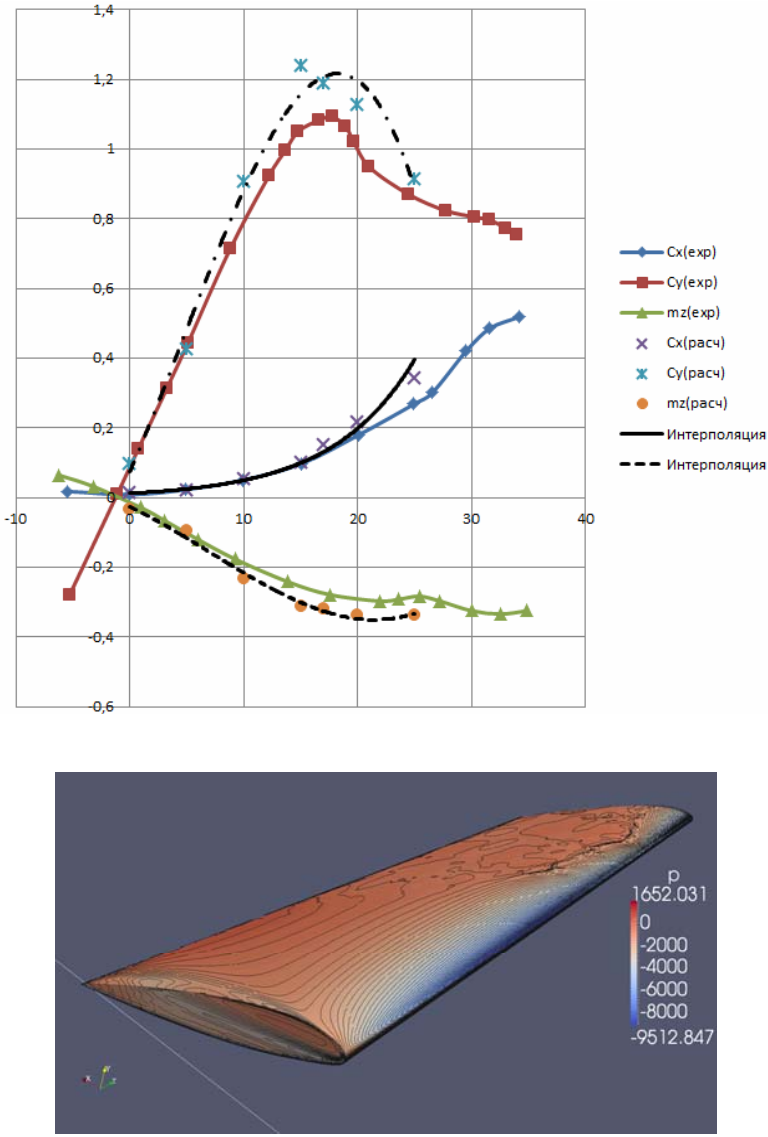


Рис. 6. Расчётные и экспериментальные характеристики профиля конечного удлинения (а), расчётное распределения давления по поверхности крыла (б)

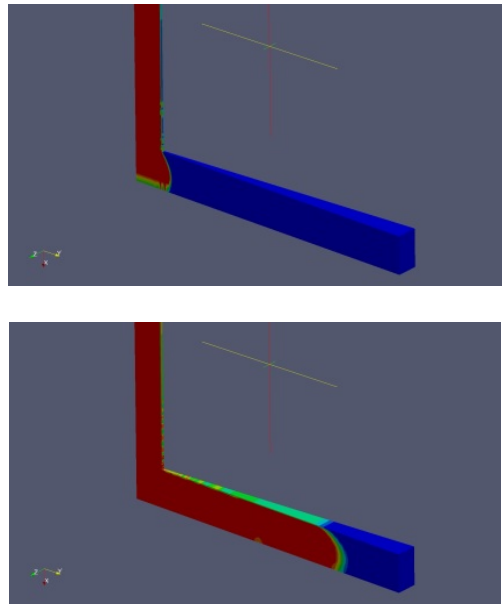


Рис. 7. Распределение смолы (красный цвет) в канале в различные моменты времени

УДК 004.414.2:004.04

А.В. Марков
Новосибирский государственный технический университет

ПРИМЕНЕНИЕ МАТРИЧНОГО ПРЕДСТАВЛЕНИЯ СЕТЕЙ ПЕТРИ К РАЗЛИЧНЫМ СИСТЕМАМ

Введение

Сетью Петри (P,T,I,O) [1] называют двудольный ориентированный граф, содержащий вершины двух типов – позиции и переходы. Вершины разного типа соединяются непосредственно. Позиции могут содержать метки (фишки, маркеры), способные перемещаться по сети. Событие – это срабатывание перехода, т. е. метка из входной позиции этого перехода перемещаются в выходную позицию [1].

В работах [2–4] приводилось различные анализы сетей: матричным представлением сетей, выделением языков сетей Петри [5], а также генерацией пространства состояний и построением дерева достижимости [2–4; 6].

Матричная форма представления сетей Петри $\left(P,T,D^-,D^+\right)$ является эквивалентом стандартной форме и позволяет дать определения в терминах векторов и матриц, отличие заключается лишь в появлении двух матриц D^- и D^+, представляющих входную и выходную функции [1].

Представление сетей Петри в виде матриц, предложенное в [1] и рассмотренное в [3–4], будет продемонстрировано на разных системах: система «Умный светофор» [8], классическая задача взаимодей-

---

[1] $P=\{p_1,p_2,\ldots,p_n\}$ – Конечное множество позиций, $n\geq 0$.

$T=\{t_1,t_2,\ldots,t_m\}$ – Конечное множество переходов, $m\geq 0.$

$I:T\to P^\infty$ является входной функцией – отображением из переходов в комплекты позиций.

$O:T\to P^\infty$ есть выходная функция – отображение из переходов в комплекты позиций.

ствия пользователя и банкомата [2], а также описание работы двухсимочных телефонов, рассмотренных в более ранних работах [7].

1. Описание работы двухсимочных телефонов

В настоящее время на рынке мобильной технике имеют своё место двухсимочные телефоны, укомплектованные одним или двумя радиомодулями (антенной для приёма и передачи сигнала).

При работе с одним радиомодулем в режиме ожидания активны обе SIM-карты. Во время звонка на одну или с одной из SIM, вторая SIM остаётся зарегистрированной в сети, но для звонящих переводится в состояние “занято”. Идентичная реакция возникает, когда с одной из SIM-карт подключаются к интернету. Исключением являются SMS, которые могут доставляться, как при разговоре, так и при активной интернет сессии. На рис. 1 показана сеть Петри, которая построена в программной среде CPN Tools и демонстрирующая логику работы двухсимочного телефона с одним радиомодулем.

Система (рис. 1, 7)], а именно сеть Петри [7], отображающая логику работы системы, была упрощена для более наглядного представления (рис. 1). Переделанная сеть из графического вида была преобразована в матричную форму (рис. 2).

2. Описание работы системы «Умный светофор»

Система «Умный светофор» – это часть рассматриваемого примера использования методики разработки программного обеспечения с применением диаграмм UML и аппарата сетей Петри.

Данная система, непосредственно, была изменена из [рис. 4.4, 8] в более упрощенную (рис. 3). Преобразование сети осуществлялось при помощи способа предложенного в [6], а именно создание массива меток с историей. Полученная сеть представлена в матричной форме на рис. 4.

3. Описание взаимодействия пользователя и банкомата

Банкомат – это устройство, автоматизирующее операции по выдаче и переводу денег, хранящихся в банке, лицу, которому они принадлежат [2]. Кредитная карта банка и соответствующей карте PIN-код идентифицируют каждого клиента.

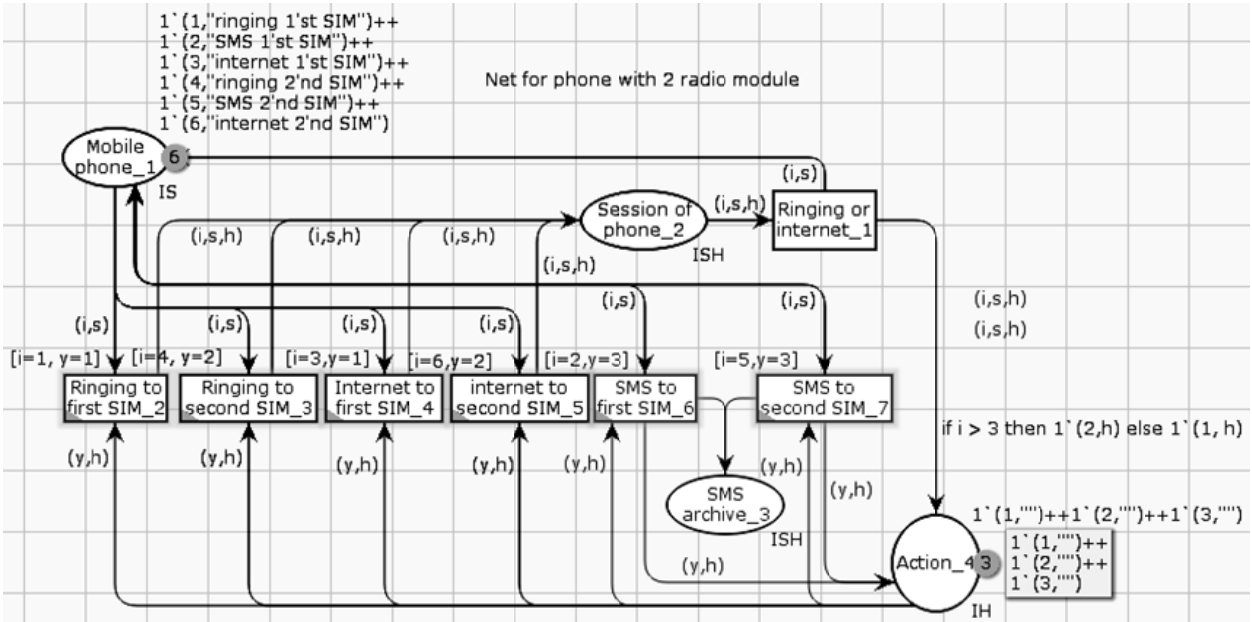


Рис. 1. Сеть Петри «Двухсимочный телефон»

Матрица D-	Матрица D+	Матрица D
0|1|0|0|	1|0|0|1|	1|-1|0|1|
1|0|0|1|	0|1|0|0|	-1|1|0|-1|
1|0|0|1|	0|1|0|0|	-1|1|0|-1|
1|0|0|1|	0|1|0|0|	-1|1|0|-1|
1|0|0|1|	0|1|0|0|	-1|1|0|-1|
0|0|0|1|	0|0|1|1|	0|0|1|0|
0|0|0|1|	0|0|1|1|	0|0|1|0|

Вектор начального состояния

6\|0\|0\|3\|

Рис. 2. Матричное представление сети Петри «Двухсимочный телефон»

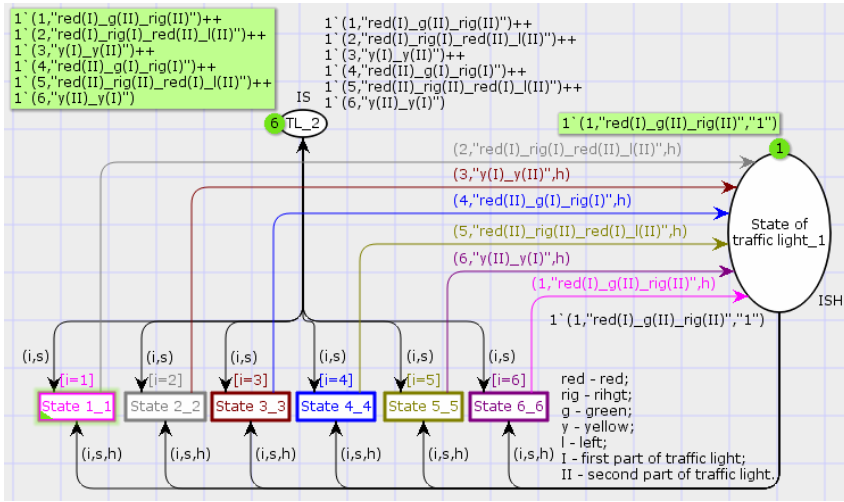


Рис. 3. Сеть Петри «Умный светофор»

Матрица D-	Матрица D+	Матрица D
1|1|	1|1|	0|0|
1|1|	1|1|	0|0|
1|1|	1|1|	0|0|
1|1|	1|1|	0|0|
1|1|	1|1|	0|0|
1|1|	1|1|	0|0|

Вектор начального состояния

6\|1\|

Рис. 4. Матричное представление сети Петри «Умный светофор»

Изначально от пользователя требуется вставить карту. После чего ввести личный PIN-код. При корректном вводе PIN-кода пользователю предлагается выполнить одну из следующих операций: просмотреть баланс счета, снятие наличных, пополнение баланса на мобильном номере (рис. 5).

Представим данную систему (рис. 5) и её матричную форму (рис. 6).

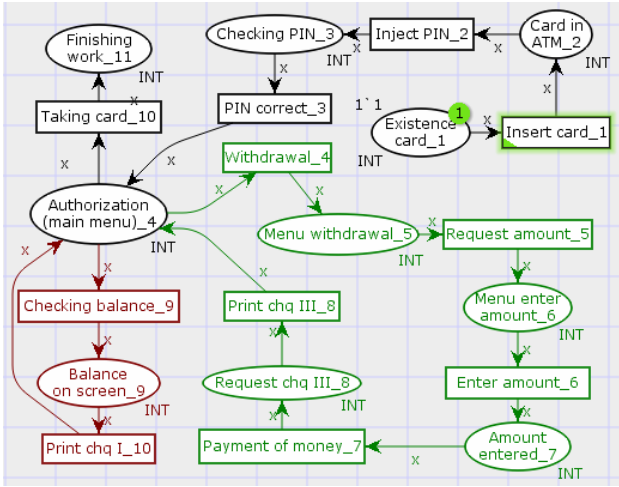


Рис. 5. Сеть Петри «Взаимодействие пользователя и банкомата»

Матрица D-

1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	0	0
0	0	0	0	0	1	0	0	0	0	0
0	0	0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	1	0	0	0
0	0	0	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	0	0
0	0	0	1	0	0	0	0	0	0	0

Матрица D+

0	1	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	0	0
0	0	0	0	0	1	0	0	0	0	0
0	0	0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	1	0	0	0
0	0	0	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	0	0
0	0	0	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	1

Матрица D

-1	1	0	0	0	0	0	0	0	0	0
0	-1	1	0	0	0	0	0	0	0	0
0	0	-1	1	0	0	0	0	0	0	0
0	0	0	-1	1	0	0	0	0	0	0
0	0	0	0	-1	1	0	0	0	0	0
0	0	0	0	0	-1	1	0	0	0	0
0	0	0	0	0	0	-1	1	0	0	0
0	0	0	1	0	0	0	-1	0	0	0
0	0	0	-1	0	0	0	1	0	0	0
0	0	0	1	0	0	0	0	0	-1	0
0	0	0	-1	0	0	0	0	0	0	1

Вектор начального состояния

1	0	0	0	0	0	0	0	0	0	0

Рис. 6. Матрица сети Петри «Взаимодействие пользователя и банкомата»

Заключение

Представление сетей в виде матриц впервые было введено в [1] и имеет ряд положительных свойств. Так, компьютерным системам легче воспринимать информацию, представленную в числовом, точнее в матричном виде, чем в иллюстрированном. Существенным недостатком является ручное преобразование из графического вида в матричную.

Стоит отметить, что алгоритм анализа с помощью матриц не является идеальным и требует доработок, которые связанны с возможностью задавания последовательности срабатывания переходов в векторе запуска, а также отсутствует запрет на срабатывания переходов, которые при определенных сценариях не задействованы в системе.

ЛИТЕРАТУРА

1. *Питерсон Дж.* Теория сетей Петри и моделирование: пер. с англ. М. : Мир, 1984.

2. *Марков А. В., Воевода А. А.* Анализ сетей Петри при помощи деревьев достижимости // Сборник научных трудов НГТУ. 2013. № 1(71). С. 78–95.

3. *Марков А. В.* Матричное представление сетей Петри // Сборник научных трудов НГТУ. 2013. № 2(71). С. 61–67.

4. *Марков А. В.* Анализ отдельных частей дерева достижимости сетей Петри // Сборник научных трудов НГТУ. 2013. № 3(72). – в печати.

5. *Воевода А. А.*, *Марков А. В.* О компактном представлении языков сетей Петри: сети с условиями и временные сети // Сборник научных трудов НГТУ. 2010. № 2(60). С. 77–83.

6. *Марков А. В.* Моделирование процесса поиска пути в лабиринте при помощи сетей Петри // Сборник научных трудов НГТУ. 2010. № 4(62) С.133–141.

7. *Марков А. В.*, *Прытков Д.* Описание работы двухсимочных мобильных телефонов с помощью сетей Петри. Камбиев метод // Сборник научных трудов НГТУ. 2010. № 3(65). С. 113–118.

8. *Марков А. В.* Автоматизация разработки программного обеспечения с использованием сетей Петри : маг. дис. Новосибирск : НГТУ, 2011.

УДК 519.24

Г.В. Трошина

Новосибирский государственный технический университет

ОЦЕНИВАНИЕ ВЕКТОРА СОСТОЯНИЯ ДИНАМИЧЕСКОГО ОБЪЕКТА С НЕТОЧНО ЗАДАННЫМИ ПАРАМЕТРАМИ

При решении задач идентификации в пространстве состояний часть переменных вектора состояния обычно оказываются неизмеримыми. Если имеется математическая модель системы, то можно вычислить состояние системы по наблюдаемым входам и выходам, например, [1–3]. Ниже используется общепринятая терминология. Восстановление вектора состояния $x(t)$ называется его оценкой $\hat{x}(t)$, а устройство, обеспечивающее получение оценки по измерениям управления $u(t)$ и вектора выхода $y(t)$ – наблюдателем. Пусть стационарный объект описывается системой уравнений

$$\dot{x}(t) = Ax(t) + Bu(t),\ y = Cx(t), \qquad (1)$$

где x – вектор состояния, u – вектор входных сигналов, y – вектор наблюдения, B – матрица управления, C – матрица наблюдения, A – матрица состояния. Вектор $x(t)$ можно аппроксимировать состоянием $\hat{x}(t)$ модели

$$\dot{\hat{x}}(t) = A\hat{x}(t) + Bu(t). \qquad (2)$$

Качество восстановления улучшается, если ввести в модель разность измеренного выхода и его оценки $y(t) - C\hat{x}(t)$ в виде обратной связи

$$\dot{\hat{x}}(t) = A\hat{x}(t) + Bu(t) + L(y(t) - C\hat{x}(t)), \qquad (3)$$

здесь L – некоторая матрица, обеспечивающая требуемый вид переходных процессов оценки вектора состояния $\hat{x}(t)$. Наблюдатель (3) восстанавливает все составляющие вектора состояния, поэтому он называется наблюдателем состояния полного порядка и порядок его равен порядку объекта. Синтез наблюдателя базируется на требованиях устойчивости и быстродействия наблюдателя. При этом следует

учитывать влияние погрешности задания параметров объекта на погрешность оценки вектора состояния [1; 4]. Если передаточная функция $W(s) = K/((\tau \cdot s + 1)(s + 1))$, где $K = 10$ и $\tau = 3$, то система (1) имеет параметры: $A = \begin{pmatrix} 0 & 1 \\ -1/3 & -4/3 \end{pmatrix}$, $B = \begin{pmatrix} 0 \\ 10/3 \end{pmatrix}$, $C = \begin{pmatrix} 1 & 0 \end{pmatrix}$ и $L = (18\frac{2}{3} \quad 74\frac{7}{9})^{\mathrm{T}}$.

MATLAB позволяет решать различные задачи, связанные с техническими вычислениями, при этом важная роль отводится специализированным группам программ, называемых toolboxes. Toolboxes применяются для обработки сигналов, систем контроля, нечеткой логики, вейвлетов и т. д. Simulink – это интерактивная система для моделирования нелинейных динамических систем. В пакете Simulink используются идеальные модели компонентов, т. е. не учитываются влияния помех и другие физические процессы. Поэтому, чтобы увидеть различия в переходных процессах исследуемых систем, можно задать различные начальные условия на интеграторах, например, на одном из интеграторов задано начальное условие равным 1. На рис. 1 представлено сравнение истинного значения вектора состояния и его оценки.

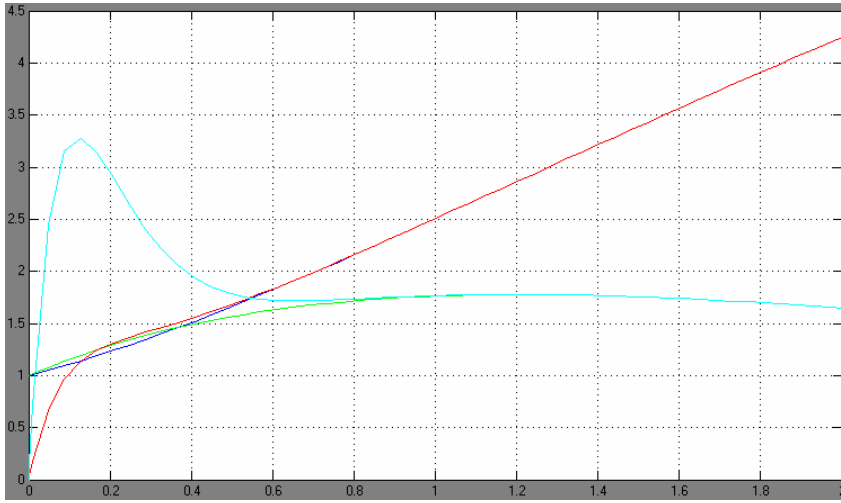


Рис. 1. Сравнение оценки и истинного значения вектора состояния

Ниже показан экран осциллографа, отображающий выходной сигнал объекта, заданного передаточной функцией и в пространстве состояний, а также выход наблюдателя. Введение начальных условий интегратора объекта позволяет увидеть различие между соответствующими компонентами векторов состояний объекта и наблюдателя в начале переходного процесса.

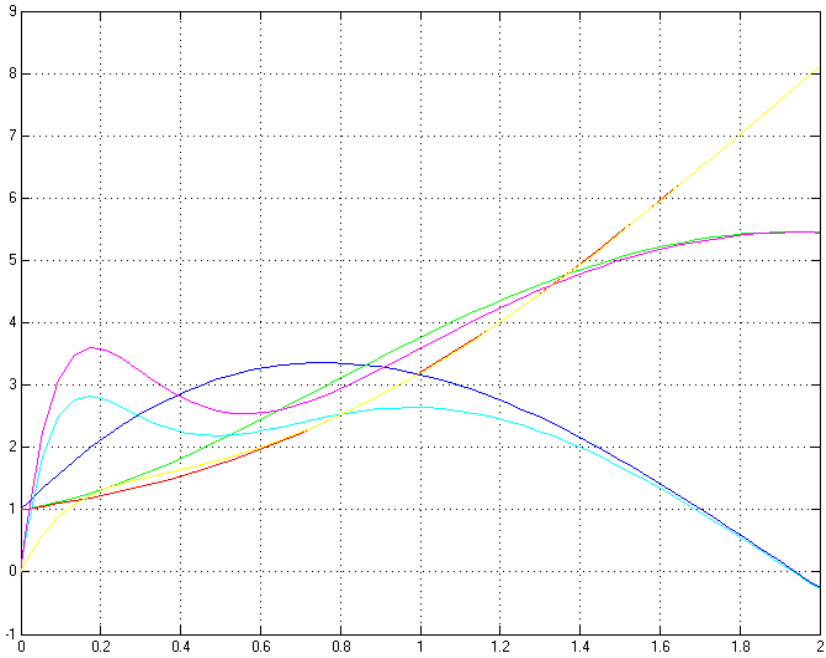


Рис. 2. Исследование работы наблюдателя

Изменяя коэффициенты передаточной функции и наблюдая за ее поведением на осциллографе, можно определить вид переходного процесса. Вычисление матрицы K регулятора осуществляется с помощью формулы Аккермана. Переходные процессы в исходной системе и системе с регулятором показаны на рис. 3.

Как видно из рис. 3, переходный процесс в системе с регулятором завершается примерно в 5 раз быстрее, чем в исходной системе.

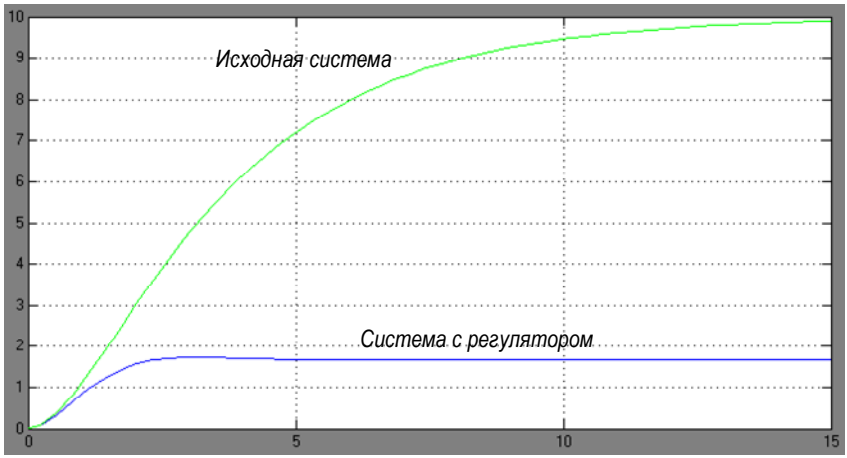


Рис. 3. Переходный процесс в системе с регулятором

Чтобы оценить влияние помех и параметров системы на оценку вектора состояния, необходимо наблюдать зависимость вектора состояния от скорости работы наблюдателя при определенных параметрах системы и помехах. Пусть исследуемый объект задан передаточной функцией $W(s) = \dfrac{K}{(\tau \cdot s + 1)(T \cdot s + 1)}$, где τ, K, T – параметры объекта. Для быстрого вычисления матрицы наблюдателя L выразим ее через параметры τ, K, T: Таким образом, матрица наблюдателя $L = \begin{pmatrix} l_1 \\ l_2 \end{pmatrix} = \begin{pmatrix} 2 \cdot a - \dfrac{T + \tau}{T \cdot \tau} \\ a^2 - \dfrac{1}{T \cdot \tau}\left(1 + \left(T + \tau\right) \cdot l_1\right) \end{pmatrix}$. Такая запись позволяет легко получать описание объекта в пространстве состояний, меняя параметры τ, K, T передаточной функции.

Природа помех в системе может быть различной. Для удобства моделирования примем в качестве помехи синусоидальный сигнал постоянной амплитуды, равной 5 % от установившегося значения выходного сигнала, и постоянной частоты 25 Гц.

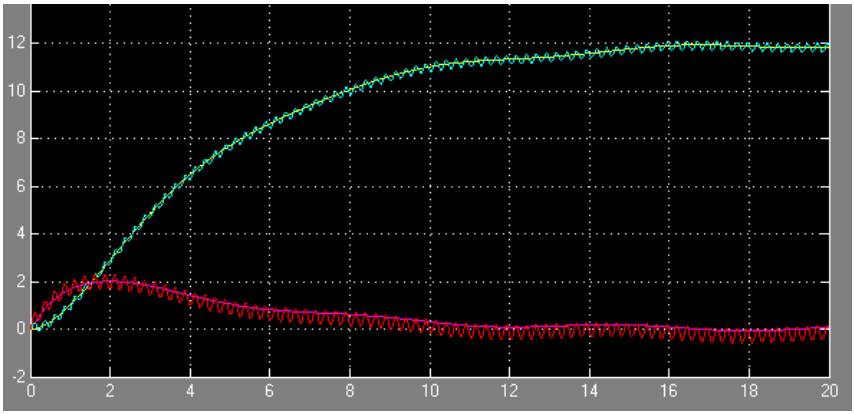


Рис. 4. Оценка вектора состояния при среднем быстродействии

На рис. 4 показано изменение оценки вектора состояния при значении $a = 5$.

Из рис. 4 можно заключить, что на оценку вектора состояния влияют как помеха, так и значения параметров. Однако, их влияние умеренно. Таким образом, для объектов, описываемых передаточной функцией вида $W(s) = \dfrac{K}{(\tau \cdot s + 1)(T \cdot s + 1)}$, оптимальный вид выходного сигнала оценки достигается при среднем значении быстродействии наблюдателя (в данном случае $a = 5$).

При расчете реальных систем низкое значение быстродействия наблюдателя негативно влияет на оценку вектора состояния ввиду существенного влияния параметров системы. Высокое быстродействие также нежелательно, поскольку при нем существенно усиливается влияние помех. Поэтому оптимальное значение быстродействия наблюдателя определяется в ходе анализа конкретной системы.

ЛИТЕРАТУРА

1. *Трошина Г. В.* Активная идентификация линейных динамических дискретных стационарных объектов во временной области : дис. …канд. техн. наук : Спец. 05.13.01; Новосиб. гос. техн. ун-т. Новосибирск, 2007. 171 с.

2. *Troshina G. V.* Algorithmic support for unknown parameters estimation of dynamic model on base of the Fisher information matrix // Proc. of the 8th Open German-Russian

Workshop «PATTERN RECOGNITION and IMAGE UNDERSTANDING». Nizhny Novgorod, 2011. P. 191–194.

3. *Troshina G. V.* Calculation of the Fisher information matrix on base of discrete-time systems in state-space description // Proc. of the 2-nd Indo-Russian Joint Workshop on Computational Intellegense and Modern Henristics in Automation and Robotics. Novosibirsk : NSTU, 2011. P. 102–105.

4. *Воевода А. А., Трошина Г. В.* Влияние неточного задания параметров объекта на оценку вектора состояния с использованием наблюдателя полного порядка // Сборник научных трудов НГТУ. Новосибирск, 2007. Вып. 4(50). С. 159–162.

УДК 004.056.53

Д.М. Бречка
Омский государственный университет им. Ф.М. Достоевского

ОЦЕНКА ЗАЩИЩЕННОСТИ КОМПЬЮТЕРНЫХ СИСТЕМ ОТ НЕСАНКЦИОНИРОВАННЫХ ДОСТУПОВ НА ОСНОВЕ МОДЕЛИ TAKE-GRANT*

В большинстве современных защищенных компьютерных систем для защиты информации от несанкционированных доступов используется механизм дискреционного разграничения доступа [1]. Однако наличие механизма еще не гарантирует защищенности информации, так как механизм может неправильно использоваться, или некорректно работать. Данное исследование посвящено разработке методов оценки корректности настройки механизмов дискреционного разделения доступа.

Оценивать корректность настройки дискреционной политики безопасности разумно на основе формальной модели безопасности. На сегодняшний день известен ряд формальных моделей безопасности [2]. Каждая модель имеет свои достоинства и недостатки, в связи с чем, возникает задача выбора наиболее подходящей модели для анализа безопасности компьютерных систем.

С другой стороны, политика безопасности в конкретной компьютерной системе может не полностью соответствовать классической модели безопасности. В этом случае возникает проблема адаптации классической модели к политике, применяемой в конкретной системе.

Из сказанного можно сформулировать основные цели исследования:

1) Из существующих моделей безопасности выбрать наиболее подходящие для анализа компьютерных систем на предмет наличия несанкционированных доступов;

2) Разработать алгоритмы анализа безопасности компьютерной системы в рамках выбранной модели безопасности;

* *Работа выполнена при поддержке «ИнфоТеКС Академия».*

3) Адаптировать выбранные модели и разработанные алгоритмы к стандартным политикам безопасности конкретных компьютерных систем;

4) Разработать программный продукт для анализа настройки безопасности компьютерной системы.

В качестве формальной модели безопасности выбрана модель Take-Grant [3]. Данная модель является одной из наиболее хорошо развитых моделей, кроме того, Take-Grant обладает механизмами, позволяющими проводить анализ безопасности систем за полиномиальное время.

Для модели Take-Grant разработаны полиномиальные алгоритмы поиска основных структур для анализа безопасности: алгоритмы поиска tg-путей [4; 5], алгоритмы поиска островов [4; 5], алгоритмы поиска мостов [5; 6], алгоритмы поиска начальных и конечных пролетов мостов [5; 6].

В ходе работы был проведен анализ дискреционных политик безопасности защищенных операционных систем семейства Windows NT и Unix. Анализ показал принципиальную возможность применения модели Take-Grant в полном объеме операционных систем Windows NT для защиты субъектов и объектов системы от несанкционированного доступа. В свою очередь, анализ операционных системах семейства Unix показал, что применение модели Take-Grant в полном объеме в этих системах не возможно, так как отсутствует возможность реализовать некоторые базовые положения модели (например, не возможно реализовать команду Take). Однако, операционные системы обоих типов используют дискреционную политику для управления доступом к объектам файловой системы, а модель Take-Grant предоставляет удобные инструменты анализа доступа к объектам. Таким образом, существует возможность использования элементов модели Take-Grant для анализа доступа к объектам файловой системы Unix и Windows NT. На сегодняшний день разработан прототип программного продукта (для систем Unix и Windows NT) для выявления скрытых доступов к объектам файловой системы. Для разработки программного продукта использовался объектно-ориентированный подход к программированию, в качестве языка разработки выбран C#. Программный продукт собирает информацию из внутренних структур операционной системы и строит граф доступа. Далее анализ скрытых доступов производится с помощью данного графа. Проведена оценка

трудоемкости и тестирование работы программного продукта. Тестирование программного продукта показало его пригодность для использования в локальных системах с относительно небольшим количеством файлов.

Разработка программного продукта для оценки корректности настройки разделения доступа в операционных системах позволит находить ошибки администрирования компьютерной системы и своевременно их исправлять. Кроме того, такой анализ позволит выявить наиболее полномочных субъектов компьютерной системы. Задача защиты таких субъектов наиболее актуальна, так как успешно реализованная атака на них может нанести серьезный ущерб системе.

Наибольший эффект от анализа настройки разделения доступа будет достигнут в том случае, если анализу будет подвергаться не отдельный компьютер, а сетевой домен под управлением операционной системы. Управление компьютерной сетью на основе доменов широко применяется в организациях различного рода. Атака на сетевой домен может принести гораздо больше вреда предприятию, чем атака на отдельный компьютер пользователя.

На данном этапе реализация программы оценки настройки дискреционного разделения доступа выполнена в виде отдельного программного продукта для оценки защиты локального компьютера. В дальнейшем планируется создание Internet-сервиса, при помощи которого зарегистрированные пользователи могли бы оценить защищенность своей локальной машины или сетевого домена.

Разработка программы для оценки настройки политики безопасности операционных систем позволит отработать все основные моменты, связанные с реализацией подобных программ. В перспективе возможно расширение применения модели Take-Grant для систем других классов, таких как мобильные системы или системы управления базами данных.

ЛИТЕРАТУРА

1. DoD 5200.28 - std Department of Defense Trusted Computer System Evaluation Criteria. 1985.

2. *Девянин П. Н.* Модели безопасности компьютерных систем : учебное пособие для студентов высших учебных заведений. М. : Издательский центр «Академия», 2005. 144 с.

3. *Lipton R. J.* A Linear Time Algorithm for Deciding Subject Security // Journal of the ACM (Addison-Wesley). 1977. № 3. С. 455–464.

4. *Бречка Д. М.* Анализ возможности доступа в модели Take-Grant // В мире научных открытий. 2010. № 4 (10). Ч. 4. С. 11–13.

5. Проблемы обработки и защиты информации. Книга 1. Модели политик безопасности компьютерных систем / Белим С.В., Белим С.Ю., Бречка Д.М. и др.; под редакцией С.В. Белима. Омск : ООО Полиграфический центр КАН, 2010. 164 с.

6. *Бречка Д. М.* Алгоритм поиска мостов типа $\vec{t}^{*}$ и $\overleftarrow{t}^{*}$ в графе доступов для дискреционной модели безопасности Take-Grant // Математические структуры и моделирование. 2011. Вып. 23. С. 99–104.

УДК 004.934

О.А. Вишнякова, Д.Н. Лавров
Омский государственный университет им. Ф.М. Достоевского

ДЕТЕКТОР РЕЧЕВОЙ АКТИВНОСТИ ПО КРАТКОВРЕМЕННЫМ ХАРАКТЕРИСТИКАМ

Детектирование речевой активности (Voice Activity Detection) – это определение участков тишины в речевом сигнале, что является важной частью предобработки сигналов в задачах распознавания, сжатия, усиления речи, голосовой идентификации и ряда других. Соответственно, качественная классификация участков обрабатываемого сигнала существенно повысит точность решения последующих задач. К идеальному детектору можно предъявить ряд требований, как воспроизводимость, точность (малое количество ошибок первого и второго рода), помехоустойчивость, адаптивность, простота реализации, обработка в реальном времени [1].

На текущий момент предложены различные алгоритмы детектирования [2; 3]. Самым простым в реализации является анализ фрагментов по краткосрочной энергии и нулевым пересечениям, однако эти алгоритмы не устойчивы в зашумленном сигнале.

Важным критерием в выборе алгоритмы является априорные знания о шуме. На практике в ряде случаев возможно получить статистические характеристики среды и возможно применение статистических алгоритмов на основе отношения правдоподобия [4; 5]. Однако при отсутствии информации о шумовой составляющей возможно применения адаптивных признаков либо комбинации признаков.

Одним из возможных способов детектирования является адаптивный анализ на основании кратковременных признаков таких как кратковременная энергия, доминантная частота спектра и спектральная мера плотности [6]. Значения показателей рассчитываются для каждого участка сигнала длительность 10–20 мс.

1. Общее описания детектирования на основании кратковременных признаков.

Пусть на вход подан сигнал x, представляющий собой смесь полезного сигнала и аддитивного шума. Тогда кратковременную энергию E входного сигнала x длины N можно определить как:

$$E = \sum_{k=1}^{N} x_k \ .$$

Наряду с энергией, в спектральной области вычислима такая характеристика как доминантная частота амплитудного спектра. Таким образом, если $X(\omega)$ – Фурье-образ исходного сигнала *x*, $S(\omega)$ – АЧХ, то доминантной является частота, на которой достигается максимум амплитудного спектра:

$$F = \arg\max S(\omega)$$

Еще одной спектральной характеристикой сигнала является тональный коэффициент (Spectral Flatness Measure), который служит мерой близости сигнала к шуму и дает максимум равный 1 на белом шуме и стремится к нулю на чистом тоновом сигнале [7]. Определим его следующим соотношением:

$$SFM = \frac{\frac{1}{N}\sum_{n=0}^{N-1} \ln S(n)}{\frac{1}{N}\sum_{n=0}^{N-1} S(n)} = \frac{\sqrt[N]{\prod_{n=0}^{N-1} S(n)}}{\frac{1}{N}\sum_{n=0}^{N-1} S(n)} .$$

Таким образом, тональный коэффициент есть отношение геометрического и арифметического средних значений коэффициентов спектра на интервале. Принято использование логарифмическая форма показателя:

$$SFM_{db} = 10 \lg \frac{\frac{1}{N}\sum_{n=0}^{N-1} \ln S(n)}{\frac{1}{N}\sum_{n=0}^{N-1} S(n)} .$$

Решение о наличии в интервале входного сигнала речевой составляющей принимается по превышению хотя бы по двум и параметров пороговых значений, которые адаптивно подбираются по началу сигнала.

2. Описание алгоритма детектирования:

• Разбиение на фреймы по 10–15 мс;

• Определение кратковременной энергии *E(i)* для каждого фрейма;

• Вычисление АЧХ для каждого участка

$$S_i(\omega) = abs(FFT(x_i(t)))\,;$$

• Поиск наиболее доминантной компоненты спектра:

$$F(i) = \arg\max S_i(\omega)\,;$$

• Вычисление тонального коэффициента

$$SFM_{db}(i) = -10\lg\frac{\frac{1}{N}\sum_{n=0}^{N-1}\ln S(n)}{\frac{1}{N}\sum_{n=0}^{N-1}S(n)}\,;$$

• Определение минимальных значений параметров на первых 5–19 фреймах и установка адаптивных пороговых значений (предполагается, что в начале сигнала обычно встречается пауза как минимум в 0.5 сек);

• Принятие решения на основании комбинации решений.

На рис. 1, 2 представлен исходный сигнал и значения краткосрочных параметров на каждом из интервалов. Пики приходятся на вокализированные участки сигнала.

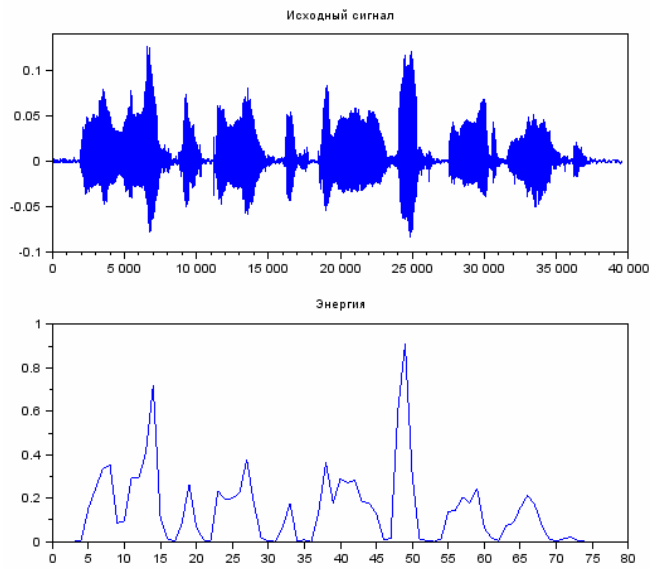


Рис. 1. Исходных сигнал и значения краткосрочной энергии на интервалах

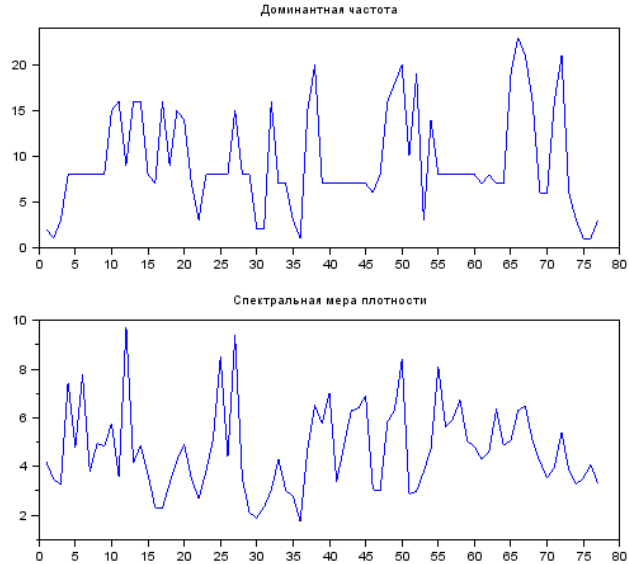


Рис. 2. Значения доминантной частоты, SFM на интервалах

ЛИТЕРАТУРА

1. *Savoji M. H.* A robust algorithm for accurate end pointing of speech // Speech Communication. 1989. PP. 45–60.

2. *Ramírez J., Górriz J. M., Segura J. C.* Voice Activity Detection. Fundamentals and Speech Recognition System Robustness // Robust Speech Recognition and understanding. 2007. P.460.

3. *Benyassine A., Shlomot E., Su H. Y., Massaloux D., Lamblin C., Petit J. P.* ITU-T Recommendation G.729 Annex B: a silence compression scheme for use with G.729 optimized for V.70 digital simultaneous voice and data applications // IEEE Communications Magazine. N35. 1997. PP. 64-73.

4. *Аграновский А. В.* Теоретические аспекты алгоритмов обработки и классификации речевых сигналов. М. : Радио и связь, 2004.

5. *Sohn J.* A statistical model-based voice activity detection // Signal Processing Letters. V.6. № 1. 1999.

6. *Moattar H., Homayounpour M.* A simple but efficient real-time voice activity detection algorithm // 17th European Signal Processing Conference (EUSIPCO 2009).

7. *Dubnov Sh.* Generalization of Spectral Flatness Measure for Non-Gaussian Linear Processes. Signal Processing Letters. 2004. № 11. V. 8. P. 698–701.

УДК 004.622

Д.Н. Лавров, А.В. Альтергот,
О.А. Вишнякова, В.П. Долгополов
Омский государственный университет им. Ф.М. Достоевского

СПЕЦИФИКАТОР БИНАРНЫХ ДАННЫХ ДЛЯ XML-ФОРМАТА ОБМЕНА БИОМЕТРИЧЕСКИМИ ДАННЫМИ

Альтернативой JSON-формата [1] является XML-формат, который в настоящее время разрабатывается нашим коллективом для обмена данными между подсистемами системы сбора биометрических данных. Принято решение сами биометрические образцы хранить в бинарном виде в кодировке BASE64. Формат бинарных данных описывается тэгом <format>, содержащим спецификатор формата.

Формат спецификатора

Структура спецификатора описывается строкой

$$\underbrace{N_1 x \ldots x N_i x \ldots x N_R}_{\text{размерность}} : \underbrace{T_1 M_1 \ldots T_L M_L}_{\substack{\text{формат ячейки,}\\ \text{описание типов полей}\\ \text{и максимальных значений}}} : \underbrace{F_s}_{\substack{\text{частота}\\ \text{дискретизации}\\ \text{в Гц, lpi, dpi, кад./с}}} ,$$

где R – размерность данных;

$_i$ – точек на соответствующей i-й оси;

L – число полей в ячейке;

T_jM_j – тип ячейки (T_j) и максимальное значение (M_j), которое может быть опущено, что будет означать, что оно совпадет с максимальным, определяемым типом; $T_j \in \{b,s,i,l,B,S,I,L,F,D\}$;

B – байт без знака (8 бит),

S – слово без знака (16 бит),

I – двойное слово без знака (32бита),

L – четверное слово без знака (64 бита), в нижнем регистре те же размерности со знаком (соответствуют в точности типам java:

b – byte,

s – short,
i – int,
l – long),

F – соответствует java float (32 бита),

D – соответствует java double (64 бита).

F_s – частота дискретизации в герцах (для звука, сигналов акселерометров и графического планшета), разрешение при печати в точках на дюйм (для изображений), в кадрах в секунду для видео потока;

: – разделитель между группами однотипных полей полями;

x – разделитель внутри группы (xx – значение по умолчанию в соответствии с типом).

Примеры формат-строк

Изображение размером 320x200 в формате RGB с разрешением 300 dpi:

`320x200:BBB:300` – сокращенная форма;

`320x200:B255B255B255:300` – полная форма.

Изображение размером 320x200 в формате CMYK с разрешением 600 dpi:

`320x200:BBBB:600`

Моно звуковая дорожка с частотой дискретизации 48кГц и 32-битной оцифровкой (4 байта), 1000 отсчётов:

`1000:I:48000` – нецентрированные или

`1000:i:48000` – центрированные относительно нуля отсчёты.

Стерео, звуковая дорожка с частотой дискретизации 22 кГц и 16-битной оцифровкой, 32000 отсчётов:

`32000:II:22000`

Данные графического планшета A6 (4,13x5,83 дюйма) с частотой дискретизации по времени 50Гц и разрешением по горизонтали и вертикали 2540 dpi (lpi) и 512 уровнями давления, 2000 отсчётов:

`2000:FFS512:xxx:50,` если разрешение сразу переводится во float

или, если не переводится, – используем формат без сокращений (4,13*2540=10490,2 точек; 5,83*2540= 14808,2):

`2000:S10491S14809S512:50`

Пример формат-строки для видеопотока в 1000 кадров размером 320x200 в формате RGB и частотой 30 кад. / сек.

```
320x200x1000:BBI:30
```

Порядок байт

Порядок байт для записи чисел будем использовать от старших байт к младшим. Этот порядок не совпадает с принятыми соглашениями для архитектуры процессоров Intel, но является более естественным и, кроме того, BIG_ENDIAN используется по умолчанию в Java. Преобразования из BIG_ENDIAN в LITTLE_ENDIAN в языке Java не представляет сложности[2].

Регулярное выражение для проверки формат-строки

Разработано регулярное выражение для проверки и разбора формат-строки:

```
((?:[1-9][0-9]*)(?:x[1-9][0-9]*)*):((?:
[B,b,I,i,L,l,S,s,F,D]
   (?:[1-9][0-9]*)?(?:\{\w+?\})?)+):([1-
   9][0-9]*(?:\{\w+\})?)
```

Первая вторая и третья группы (в java метод group()) выдают группы символов между разделителями «:» для последующего разбора

Созданы JUnit тесты для проверки корректности её работы.

Критика

Присутствует изменчивость единиц измерения Fs: то это dpi, то Гц, то кадры в секунду. Для улучшения читаемости и лучшего понимания контекста, решено ввести комментарии в формат строку (?:\{\w+\})?).Таким образом, формат-строки с комментариями теперь выглядят так:

```
1000:B:100{Hz}
123:L{x}L{y}L{p}L{t}:60{Hz}
1000:L{x}L{y}L{p}L{t}:60{Hz}
```

[2] Примеры смены порядка байт можно найти в статье: http://javainception.ru/index.php/sistemavvodavivodajava/sistemavvodavivodajava/o-poryadke-baytov.html

```
320x200:B255{Red}B255{Green}B255{Blue}B255{null}
:300{dpi}
```

Заключение

В результате длительных дискуссий в коллективе разработчиков выработан формат, который удовлетворяет всех разработчиков проекта. Спецификация формат-строки представлена в данной статье. Разработано регулярное выражение для быстрой проверки корректности формат-строки и её анализа. Созданы JUnit-тесты для модульного тестирования использования этого регулярного выражения.

ЛИТЕРАТУРА

1. *Вишнякова О. А. Лавров Д. Н.* Формат обмена данными в системе сбора и обработки биометрических образцов // Международная научно-практическая конференция «Информационные ресурсы в образовании». Нижневартовск : НГГУ, 17–19 апреля 2013. С. 146–148.

УДК 519.725:004.94

В.В. Коробицын, Д.П. Монастыренко, М.Н. Московцев
Омский государственный университет им. Ф.М. Достоевского,
ОАО «ОНИИП», г. Омск

ОПРЕДЕЛЕНИЕ ОПТИМАЛЬНОГО КОЛИЧЕСТВА ИТЕРАЦИЙ ДЕКОДИРОВАНИЯ В CUDA-РЕАЛИЗАЦИИ ТУРБО-ДЕКОДЕРА

В системах передачи информации важную роль играют методы помехозащищенного кодирования, обеспечивающих обнаружение и исправление ошибок, возникающих при передаче по каналу связи. Предложенные в 1993 году итеративно декодируемые коды или турбо-коды [1] получили широкое распространение в системах передачи информации. Главным преимуществом турбо-кодов является существенное снижение вероятности ошибки при декодировании для малых отношений сигнал-шум на бит (до 0,7 dB).

Алгоритм кодирования турбокода достаточно быстрый и не требует больших вычислительных затрат. В то время как алгоритм декодирования требует интенсивных вычислений, связанных с вычислением мягкого выхода, который вычисляется итеративно. Существуют реализации алгоритма декодирования турбо-кода для персональных компьютеров (например, реализация М. Валма и Р. Морелос-Сарагосы [2]). Однако данная реализация скорее учебная, чем промышленная. Средняя производительность декодирования турбо-кода со скоростью 1/3, памятью 3 и 5 итерациями декодирования составляет около 50 Кбит/с на процессоре с тактовой частотой 2 ГГц. После некоторой оптимизации данная реализация может быть пригодна для декодирования сигнала приходящего с модема на скорости до 56 Кбит/с. В случае, когда требуется декодировать сигнал с более высокой скоростью передачи или несколько каналов одновременно, необходимо применять более высокоскоростное аппаратное обеспечение и параллельное программирование.

В качестве аппаратного обеспечения можно применить высокоскоростные графические карты или вычислительные модули Tesla. В качестве платформы для разработки программ можно применить технологию CUDA.

Цель выполняемого проекта – исследование возможности использования программно-аппаратной архитектуры CUDA для реализации быстрых декодеров турбо-кода.

1. Описание эксперимента

В эксперименте были использованы следующие показатели турбо-кодера:

– количество состояний 4, 8, 16;

– скорость кода 1/3 (без выкалывания);

– количество итераций от 1 до 32;

– размер блока от 32, 2048 и 32768 бит;

– соотношение сигнал/шум от 0,2 до 3,0 дБ.

В каждом эксперименте определялось количество ошибочно декодированных бит информации, объем декодированной информации и время, затраченное сервером на базе Tesla для декодирования. Условием завершения прогона является: получения более 1000 ошибок или декодировано более 30 000 000 бит.

Весь объем декодируемой информации разбивается на блоки заданного размера и отправляются на GPU порциями по 256 блоков одновременно (оптимальный размер порции будет определен далее). На основе собираемых данных об эксперименте определяются следующие показатели:

– вероятность ошибочного декодирования одного бита информации;

– скорость декодирования (бит/с).

Поставленная задача эксперимента – определить оптимальные параметры турбо-кодера для обеспечения малой вероятности ошибки и высокой скорости декодирования в зависимости от соотношения сигнал/шум.

2. Оценка оптимального количества итерации

Для оценки оптимального количества итераций, необходимого для декодирования было введено понятие унифицированной стоимости работы декодера C, определяемое следующим соотношением:

$$C = \text{Perf} / (k * \text{Prob}),$$

где Perf – скорость декодирования (объем декодируемой информации в единицу времени), Prob – вероятность ошибки декодирования, k – масштабный коэффициент (в эксперименте равен 1 000 000). Введен-

ный показатель дает возможность определить оптимальные параметры декодера. Чем выше С, тем лучше декодер. Используя этот показатель, получены оптимальные значения количества итераций для декодера с 8 состояниями (табл. 1). Функция зависимости показателя С от количества итераций (рис. 1) имеет явно глобальный максимум, позволяющий однозначно определить количество итераций.

Таблица 1

Оптимальные значения количества итераций при разных соотношениях сигнал/шум для кодера (8 состояний, длина блока – 32768 бит)

№	*Интервал соотношения сигнал/шум (дБ)*	*Оптимальное число итераций*
1	менее 0,5	более 12
2	0,5–0,8	7
3	0,8–1,4	5
4	1,4–2,9	3
5	более 2,9	2

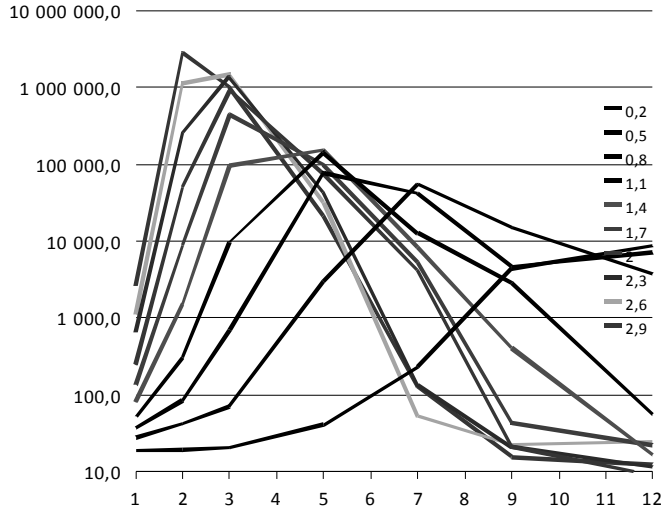


Рис. 1. Графики зависимости показателя стоимости С от количества итераций (декодер с 8 состояниями и длиной блока — 32768 бит, разные линии соответствуют разным соотношениям сигнал/шум)

При использовании оптимальных параметров декодера получается линейное уменьшение (в логарифмической шкале) вероятности ошибки декодирования от соотношения сигнал/шум (рис. 2) и линейный рост скорости декодирования (рис. 3).

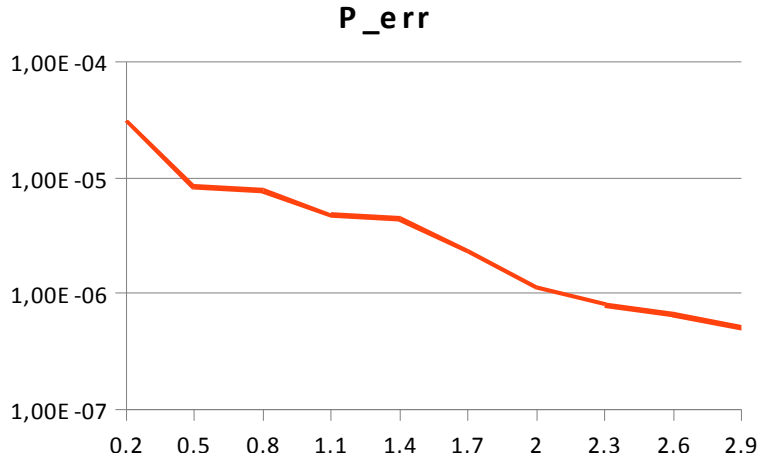


Рис. 2. Зависимость вероятности ошибочного декодирования от соотношения сигнал/шум (ось вероятности в логарифмической шкале)

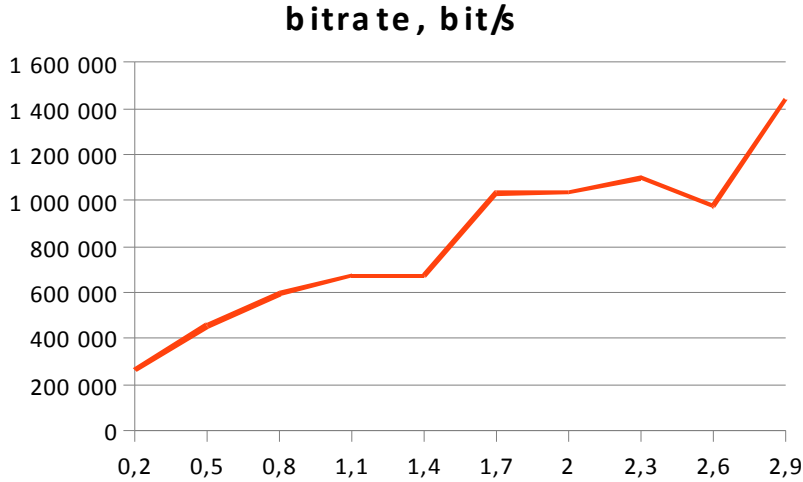


Рис. 3. Зависимость скорости декодирования от соотношения сигнал/шум при выборе оптимального количества итераций (скорость измеряется в бит/с)

Выводы и заключение

Полученные результаты эксперимента показывают возможность построения турбо-декодера с оптимальным (переменным) количеством итераций в зависимости от среды передачи информации, выраженной соотношением сигнал/шум.

После проведения ряда экспериментов было установлено:

1) кодер с 4 состояниями малоэффективен, поскольку дает большую вероятность ошибки декодирования, чем кодеры с 8 и 16 состояниями. 4-декодер может быть использован только при соотношении сигнал/шум более 2,5 дБ;

2) кодеры с 8 и 16 состояниями равнозначны по эффективности;

3) увеличение длины блока позволяет повысить эффективность декодера (чем больше размер блока, тем надежнее работает декодер, и как следствие уменьшается время декодирования). Наиболее эффективный размер блока оказался равен 32768 бит;

4) для оценки оптимального количества итераций, можно воспользоваться оценкой унифицированной стоимости работы декодера C. Функция зависимости показателя C от количества итераций имеет глобальный максимум, позволяющий однозначно определить оптимальное количество итераций.

ЛИТЕРАТУРА

1. *Berrou C., Glavieux A.* Near Optimum Error Correcting Coding And Decoding: Turbo-Codes // IEEE Transactions on Communications. V. 44. № 10. October 1996.

2. *Морелос-Сарагоса Р.* Искусство помехоустойчивого кодирования. Методы, алгоритмы, применение. М. : Техносфера, 2005. 320 с.

УДК 004.93'1

А.Г. Казанцева
Омский государственный университет им. Ф.М. Достоевского

АРХИТЕКТУРА ПРИЛОЖЕНИЯ ДЛЯ СБОРА БИОМЕТРИЧЕСКИХ ОБРАЗЦОВ ПОХОДКИ ЧЕЛОВЕКА

В настоящее время достаточно активно ведутся исследования в области биометрической идентификации и аутентификации. В частности, большой интерес ученых вызывает такой сравнительно новый тип биометрии, как походка человека.

Различают три подхода к сбору данных о походке: машинное зрение, носимые сенсоры и сенсоры в полу [1]. Из этих подходов наиболее разработанным является подход с машинным зрением. Другие два подхода менее распространены, в связи с чем для них не существует общедоступных баз данных образцов. Это затрудняет как разработку, т.к. каждый исследователь вынужден самостоятельно собирать образцы, так и тестирование биометрических алгоритмов и систем, т.к. для корректного сравнения производительности предложенных методов необходимо проверять их на одном наборе данных. Поскольку наши научные интересы лежат в области распознавания походки на основе носимых сенсоров, в дальнейшем будем обсуждать именно это направление.

Рассмотрим существующие на данный момент закрытые базы данных, используемые в исследованиях по изучению походки.

База данных Trung et al [2] содержит данные 736 субъектов (382 мужчины и 354 женщины) и является крупнейшей из известных нам БД образцов походки, полученных с помощью носимых сенсоров (акселерометр и гироскоп, расположенные сзади на поясе). Она имеет хорошее распределение по возрасту (от 2 до 78 лет, количество детей сопоставимо с количеством взрослых) и полу. В общей сложности БД содержит 1472 записи (по 2 на субъекта).

В своих исследованиях Gafurov et al [3–10] использовали набор из нескольких БД:

1) БД из 100 субъектов (70 мужчин, 30 женщин). Сенсор на поясе, справа. Всего 760 записей, из них: 400 – обычные, 360 – попытки имитации чужой походки.

2) БД из 50 субъектов (33 мужчины, 17 женщин). Сенсор в кармане. Всего 200 записей.

3) БД из 30 субъектов (все мужчины). Сенсор на правой щиколотке. 16 записей на субъекта (по 4 записи на каждый из 4-х типов обуви), всего 480 записей.

4) БД из 30 субъектов (23 мужчины, 7 женщин). 4 записи на субъекта (2 – правая рука, другие 2 – левая), всего 120 записей.

5) БД из 21 субъекта (12 мужчин, 9 женщин). Сенсор на щиколотке. Всего 42 записи.

Для записи походки использовался акселерометр. Все эксперименты проводились в условиях ходьбы по ровной поверхности и с нормальной скоростью. Суммарное количество записей в БД Gafurov et al: 1602.

БД Kobayashi et al [11] в общей сложности содержит 2331 запись походки 58 субъектов, в среднем по 40 на человека. Распределение по полу и возрасту неизвестно. Во время записи сенсор (акселерометр) находился в руке.

Остальные известные нам БД и того меньше: около 10–30 субъектов.

Стоит отметить, что все вышеперечисленные БД обладают хотя бы одним из следующих недостатков:

- Недостаточное количество субъектов.
- Недостаточное количество записей на одного субъекта.
- Неравномерное распределение по полу.
- Неравномерное распределение по возрасту.
- Отсутствие разнесенных по времени образцов.
- Отсутствие записей, учитывающих такие потенциально способные повлиять на походку или ее распознавание факторы, как скорость ходьбы/бег, обувь, тип поверхности, расположение датчика на теле, переносимый субъектом груз.
- Отсутствие записей попыток фальсификации походки.

Кроме того, ни одной из этих баз нет в открытом доступе.

Таким образом, становится понятно, что существует реальная необходимость в сборе большой, качественной, желательно общедоступной БД образцов походки, которая была бы лишена всех перечислен-

ных выше недостатков. Так как сбор БД такого размера вручную (при помощи ПО, специально для этого не предназначенного) довольно затруднителен, было принято решение автоматизировать этот процесс.

Рис. 1. Интерфейс приложения AccelRecorder

В рамках данной задачи было разработано приложение AccelRecorder, интерфейс которого приведен на рис. 1. Это приложение написано на языке Java и поддерживает сбор биометрических образцов с помощью носимых сенсоров. В качестве носимого сенсора выступает устройство SunSPOT [12] компании Sun Microsystems, оснащенное трехосевым MEMS-акселерометром. Для связи с ним используются библиотеки из SunSPOT SDK.

Интерфейс приложения выполнен в виде так называемого «мастера» (wizard). В его основе лежит API для создания wizard'ов CJWizards [13]. На первом шаге мастера оператор вводит информацию о субъекте, на втором – настройки записи, на третьем происходит непосредственно запись образцов. Для снижения вероятности опечаток при заполнении информации о субъекте в приложении реализована несложная проверка правильности ввода на основе библиотеки Hibernate Validator [14] и регулярных выражений.

UML диаграмма пользовательского интерфейса приведена на рис. 2. Опишем ее основные элементы. Класс *RecorderWizard* отвечает за запуск и перезапуск интерфейса мастера. Класс *AccelRecorderPageFactory* реализует интерфейс *PageFactory* из библиотеки CJWizards и задает порядок шагов. Классы *SubjectPage*, *SettingsPage* и *SamplesPage* представляют собой непосредственно страницы мастера (*WizardPage*). Класс *ValidatedTextField* представля-

ет собой текстовое поле с проверкой ввода. Также на рис. 2 можно видеть основные зависимости от сторонних библиотек.

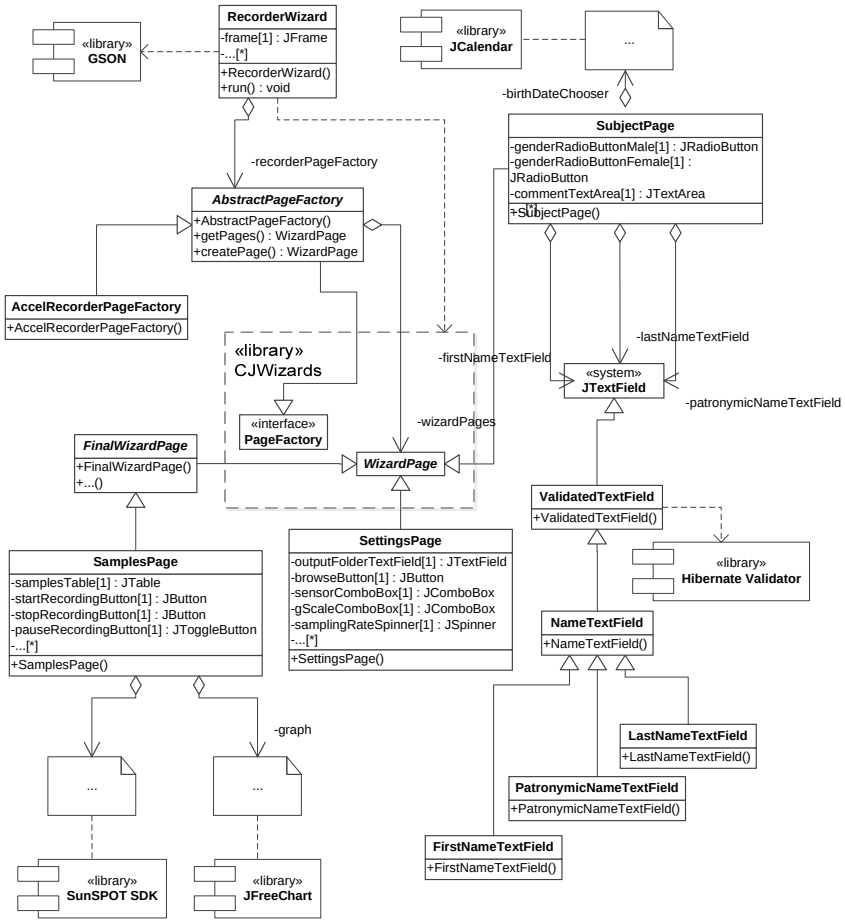


Рис. 2. UML диаграмма интерфейсной части приложения
(некоторые второстепенные элементы и связи опущены для наглядности)

Для хранения данных используются классы предметной модели, представленные на рис. 3.

Класс *GaitSample* описывает непосредственно образец походки. Большую часть своих атрибутов он наследует от абстрактного класса *Sample*: *subject* – объект класса *Subject*, содержащий информацию о субъекте; *sensor* – объект класса *Sensor*, содержащий информацию об оборудовании, с помощью которого производилась запись; *timestamp* – метка времени создания образца; *sampleData* – трехмерный массив, содержащий показания акселерометра; *time* – массив временных меток для *sampleData* (в мсек); *sampleHash* – уникальный идентификатор образца, полученный путем хеширования массива *sampleData* по алгоритму SHA-256; *comment* – необязательный комментарий. Помимо унаследованных, *GaitSample* имеет также следующие атрибуты: *gScale* – уровень чувствительности акселерометра (в единицах ускорения свободного падения g) и *samplingRate* – частота дискретизации (в Hz).

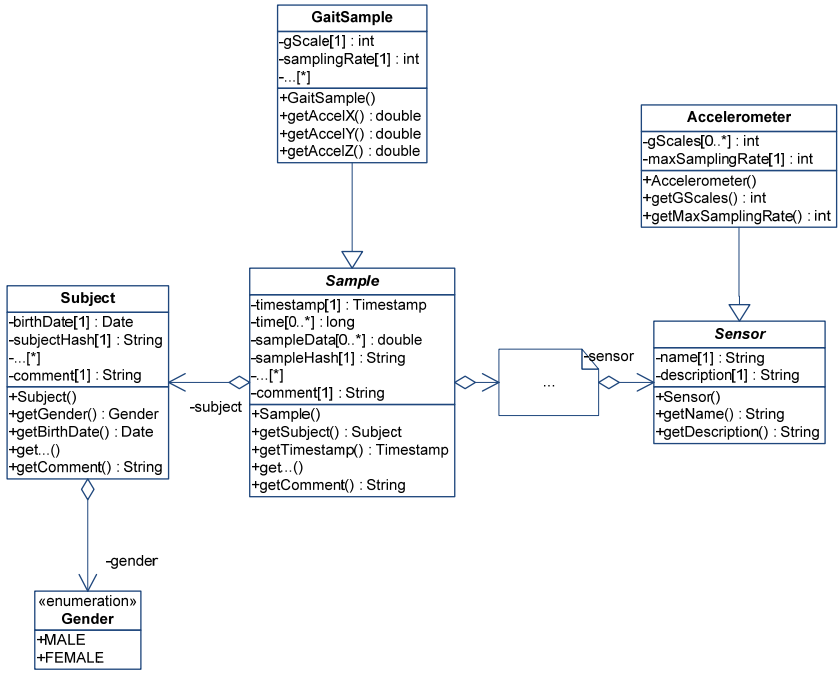


Рис. 3. UML диаграмма модели данных (private методы, константы и некоторые другие второстепенные элементы опущены для наглядности)

Класс *Subject* описывает субъекта, походка которого записывалась. Каждый объект класса *Subject* содержит следующую информацию: дата рождения (*birthDate*), пол (*gender*), необязательный комментарий (*comment*) и уникальный идентификатор субъекта (*subjectHash*). *subjectHash* получается из ФИО и даты рождения субъекта в формате «дд.мм.гггг» путем их конкатенации и последующего хеширования по алгоритму SHA-256. В явном виде ФИО субъекта не сохраняется. Класс *Accelerometer* является подклассом абстрактного класса *Sensor* и содержит: наименование акселерометра (*name*), его описание (*description*), поддерживаемые акселерометром настройки чувствительности (*gScales*) и максимальную поддерживаемую частоту дискретизации (*maxSamplingRate*).

Полученные объекты класса *GaitSample* сохраняются на диске в формате JSON при помощи библиотеки GSON [15].

На момент написания статьи приложение находится на стадии alpha версии. Реализована большая часть функционала, заявленного в [16].

Приложение было протестировано в условиях реального сбора данных: с 4 по 29 апреля 2013 года с его помощью был собран фрагмент БД образцов походки размером 180 человек, ~ 2500 записей. По результатам тестирования сделаны выводы о направлениях дальнейшего развития приложения.

ЛИТЕРАТУРА

1. *Gafurov D.* A survey of biometric gait recognition: Approaches, security and challenges // Annual Norwegian Computer Science Conference. 2007. P. 19–21.

2. *Trung N. T., Makihara Y., Nagahara H., Mukaigawa Y.* Performance evaluation of gait recognition using the largest inertial sensor-based gait database // 2012 5th IAPR International Conference on Biometrics (ICB), IEEE Biometrics Compendium. March 29 2012–April 1 2012. P. 360–366.

3. *Gafurov D.* Security analysis of impostor attempts with respect to gender in gait biometrics // First IEEE International Conference on Biometrics: Theory, Applications, and Systems, 2007. BTAS 2007. 27–29 September 2007. P. 1–6.

4. *Gafurov D., Snekkenes E., Bours P.* Spoof attacks on gait authentication system // IEEE Transactions on Information Forensics and Security. September 2007. Volume 2, Issue 3. P. 491–502.

5. *Gafurov D., Snekkenes E.* Towards understanding the uniqueness of gait biometric // 8th IEEE International Conference on Automatic Face & Gesture Recognition, 2008. FG '08. 17–19 September 2008. P. 1–8.

6. *Gafurov D., Snekkenes E.* Arm swing as a weak biometric for unobtrusive user authentication // IIHMSP '08 International Conference on Intelligent Information Hiding and Multimedia Signal Processing, 2008. 15–17 August 2008. P. 1080–1087.

7. *Gafurov D., Snekkenes E.* Gait recognition using wearable motion recording sensors // EURASIP Journal on Advances in Signal Processing. January 2009. Volume 2009, Article No. 7.

8. *Gafurov D., Snekkenes E., Bours P.* Improved gait recognition performance using cycle matching // 2010 IEEE 24th International Conference on Advanced Information Networking and Applications Workshops (WAINA). 20–23 April 2010. P. 836–841.

9. *Gafurov D., Bours P.* Improved hip-based individual recognition using wearable motion recording sensor // Security Technology, Disaster Recovery and Business Continuity. 2010. P. 179–186.

10. *Gafurov D., Bours P., Snekkenes E.* User authentication based on foot motion // Signal, Image and Video Processing, Volume 5, Issue 4. November 2011. P. 457–467.

11. *Kobayashi T., Hasida K., Otsu N.* Rotation invariant feature extraction from 3-d acceleration signals // 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). 22–27 May 2011. P. 3684–3687.

12. SunSPOTWorld. URL: http://www.sunspotworld.com/ (дата обращения: 27.09.2013).

13. Cjwizard – An API for creating step-by-step wizards with Java. URL: https://code.google.com/p/cjwizard/ (дата обращения: 27.09.2013).

14. Hibernate Validator – Bean Validation reference implementation. URL: http://www.hibernate.org/subprojects/validator.html (дата обращения: 27.09.2013).

15. google-gson – A Java library to convert JSON to Java objects and vice-versa. URL: http://code.google.com/p/google-gson/ (дата обращения: 27.09.2013).

16. *Казанцева А. Г.* Разработка приложения для сбора биометрических образцов походки человека // Информационные ресурсы в образовании : матер. Междунар. науч.-практ. конф.(г. Нижневартовск, 17–19 апреля 2013). Нижневартовск : НВГУ, 2013. С. 161–162.

УДК 004.942

А.В. Таран

Омский государственный университет им. Ф.М. Достоевского

ИСПОЛЬЗОВАНИЕ МЕТОДА ЭМПИРИЧЕСКОЙ МОДОВОЙ ДЕКОМПОЗИЦИИ ДЛЯ ОЧИСТКИ СИГНАЛА ОТ ПОМЕХ

Метод эмпирической модовой декомпозиции (EMD) – метод разложения сигналов на функции, которые называются «эмпирическими» модами. Метод представляет собой адаптивную итерационную вычислительную процедуру разложения исходных данных (непрерывных или дискретных сигналов) на внутренние колебания (эмпирические моды).

Традиционные методы анализа данных предназначены в основном для линейных и стационарных сигналов и систем, и только в последние десятилетия активно развиваются методы анализа нелинейных, но стационарных систем, и линейных, но нестационарных данных. Между тем, большинство естественных материальных процессов, реальных физических систем и соответствующих этим процессам и системам данных в той или иной мере являются нелинейными и нестационарными.

EMD – адаптивная методика разложения, в которой любой сложный сигнал может быть расщеплен на определенное число высокочастотных и низкочастотных составляющих посредством процесса, названного «отсеиванием».

Процесс отсеивания расщепляет первоначальный сигнал $y(t)$ на модовые функции IMFs. Сущность метода EMD заключается в последовательном вычислении функций эмпирических мод $c_j(t)$ и остатков $r_j(t) = r_{j-1}(t) - c_j(t)$, где $j = 1,2,3,...,n$ при $r_0(t) = y(t)$.

Результатом разложения будет представление сигнала в виде суммы модовых функций и конечного остатка:

$$y(t) = \sum_{j=1}^{n} c_j(t) + r_n(t) \qquad (1)$$

где n – количество эмпирических мод, которое устанавливается в ходе вычислений.

Рассмотрим пример работы алгоритма. На рисунке 1 представлены (снизу вверх): исходный сигнал, IMF[0], ... , IMF[n], остаток.

Взята сумма гармоники, нестационарного и нелинейного сигнала:

$$y(t) = 2\sin(2\pi x) + 2x\sin(\pi x/4) + 4\arcsin(\sin(x)).$$

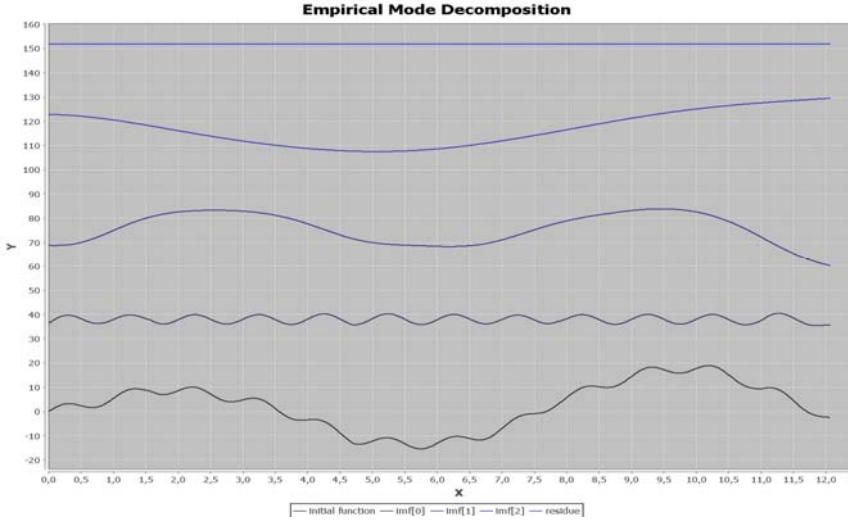


Рис. 1. Сумма гармоники, нестационарного и нелинейного сигнала

В повседневной жизни очень часто приходится сталкиваться с сигналами с помехами. Это связано с некачественной передачей сигналов, записью сигнала и т. д. Под помехой понимается любое воздействие на сигнал, которое ухудшает достоверность воспроизведения передаваемых сообщений.

В наиболее простом случае мы имеем дело с аддитивными помехами:

$$u(t) = s(t) + w(t),$$

где $u(t)$ – полученный сигнал, $s(t)$ – полезный сигнал, $w(t)$ – помеха.

Пусть имеется такой зашумленный сигнал. Если предположить, что $w(t)$ – случайный шум, мы можем его очистить с помощью метода EMD.

Вспомним формулу (1):

$$y(t) = \sum_{j=1}^{n} c_j(t) + r_n(t),$$

где $c_j(t)$ – эмпирические моды, $r_n(t)$ – остаток.

Если разбить сумму в правой части на две:

$$y(t) = \sum_{j=1}^{s} c_j(t) + \sum_{j=s+1}^{n} c_j(t) + r_n(t) \qquad (2)$$

и предположить, что первые s мод описывают помехи в сигнале, то можно их вычесть из исходной функции, получив таким образом очищенный сигнал. Такое предположение можно сделать, если присутствующий в сигнале шум является высокочастотным. Кроме того, шумы, сопровождающие полезную информацию в сигнале, полностью удовлетворяют определениям функций IMF [1]. Полученный базис в (1) является ортогональным [2], рассматриваемый шум является случайным, поэтому он должен выделиться из сигнала в несколько первых мод и некоторое количество последующих.

Данная методика не является строгой, поскольку шум может откладывать свой отпечаток на все моды. При отбрасывании нескольких первых мод может оставаться некоторая погрешность.

Рассмотрим пример очистки от помех. В примере в качестве помехи взят «белый шум», имеющий нормальное распределение от -1 до 1. На рисунке представлены (сверху вниз) зашумленный, очищенный, исходный сигнал.

Взята сумма гармоники, нестационарного, нелинейного сигнала, белого шума:

$$y(t) = x\sin(2\pi x) + 10\cos(\pi x/4) + 4\arcsin(\sin(x)) + \text{WhiteNoise}.$$

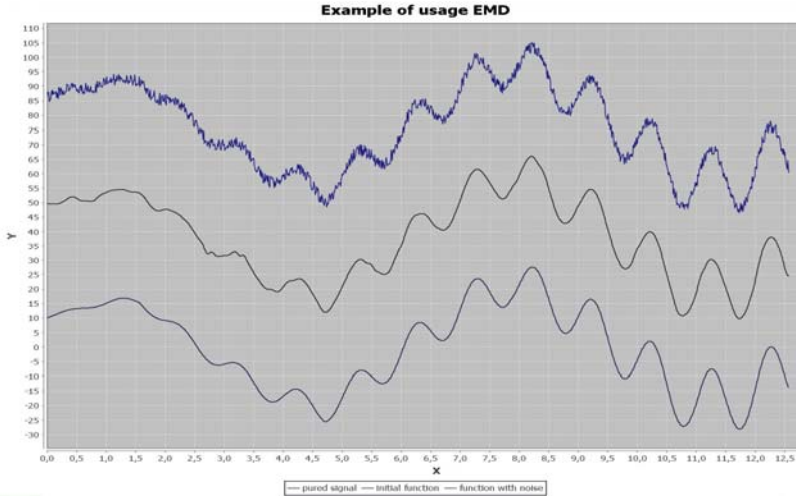


Рис. 2. Сумма гармоники, нестационарного, нелинейного сигнала, белого шума

ЛИТЕРАТУРА

1. *Давыдов В. А., Давыдов А. В.* Очистка геофизических данных от шумов с использованием преобразования Гильберта-Хуанга // Актуальные инновационные исследования: наука и практика. 2010. № 1. URL: http://www.actualresearch.ru (дата обращения: 26.09.2013).

2. *Norden E. Huang, Samuel S.P. Shen.* The Hilbert-Huang transform and its applications - World Scientific Publishing Co. Pte. Ltd. 5 Toh Tuck. Link, Singapore 596224.

УДК 004.04, 004.6, 005, 519.688, 519.711.2, 519.876.2

О.Л. Епанчинцева, Т.А. Погромская

Омский государственный университет им. Ф.М. Достоевского

МОДЕЛИ БИЗНЕС-ПРОЦЕССОВ ВЗАИМОДЕЙСТВИЯ ПРИЕМНЫХ КОМИССИЙ ВУЗОВ С ФИС ЕГЭ И ПРИЕМА

Новый Закон об образовании и другие нормативные документы, регламентирующие прием в вузы, требуют от приемных комиссий вузов всё большей открытости и прозрачности данных, а также обязательного взаимодействия с Федеральной информационной системой обеспечения проведения единого государственного экзамена и приема граждан в образовательные учреждения среднего профессионального образования и образовательные учреждения высшего профессионального образования и региональных информационных систем обеспечения проведения единого государственного экзамена (ФИС ЕГЭ и приема).

Приемные комиссии вузов имеют информационные системы (ИС) (собственные или на основе готовых решений), автоматизирующие в большей или меньшей степени бизнес-процессы приема абитуриентов. До сих пор нет типового решения комплексной организации и проведения приемной кампании и передачи сведений в ФИС ЕГЭ и приема. Поэтому актуальной задачей становится подготовка и передача данных, обрабатываемых информационными системами приемных комиссий вузов, в ФИС ЕГЭ и приема.

Для успешного решения данной задачи с помощью методологии ARIS были построены модели подготовки и передачи данных из информационно-аналитической системы «Абитуриент» [1] Омского государственного университета им. Ф.М. Достоевского (ОмГУ) в ФИС ЕГЭ и приема.

На построенных моделях реализована информационно-аналитическая система «Абитуриент-ФИС», которая располагается в защищенном сегменте сети вуза, автоматизирует и существенно упрощает процедуру загрузки данных в ФИС ЕГЭ и приема, и которая прошла успешную эксплуатацию в Омском государственном университете

им. Ф.М. Достоевского и Горно-Алтайском государственном университете в ходе приемной кампании 2013 года.

Система «Абитуриент-ФИС» реализована разработчиками управления информатизации ОмГУ на платформе Java и, помимо собственно загрузки данных в ФИС ЕГЭ и приема, позволяет выявить ошибки загрузки (при их наличии).

Укрупненная событийная модель взаимодействия приемной комиссии вуза с ФИС ЕГЭ и приема

ЛИТЕРАТУРА

1. А.с. 2003611045 РФ, Роспатент. Информационно-аналитическая система «Абитуриент» (ИАС «Абитуриент») / Горнева И.С., Епанчинцева О.Л., Захаров А.М., Картешкина Е.В., Костюшина Е.А., Погромская Т.А., Рапаева И.А., Сергеева Т.И. (RU). №2003610757: Заяв. 07.04.2003; Опубл. 30.04.2003, Бюл. №3(44). С. 98.

УДК 311.312

М.С. Атепалихин
Тюменский государственный университет

Б.Ю. Кассал
Омский государственный педагогический университет

С.В. Белим
Омский государственный университет им. Ф.М. Достоевского

ВЫЯВЛЕНИЕ ВЗАИМОСВЯЗИ МЕЖДУ БИОЛОГИЧЕСКИМИ ВИДАМИ НА ОСНОВЕ АНАЛИЗА АССОЦИАТИВНЫХ ПРАВИЛ

Основные методы численного исследования биоценозов основаны на выявлении статистических парных корреляций численности видов, в том числе [1–3]. В данной области практически не используются методы биоинформатики и искусственного интеллекта. В данной работе предлагается использование метода ассоциативных правил для выявления взаимосвязей между биологическими видами, обитающими на одной территории. Ранее данный метод в задачах биоинформатики применялся для анализа последовательностей в геномах [4–6].

Метод ассоциативных правил является одним из подходов для построения систем искусственного интеллекта и получил широкое распространение в выявлении закономерностей [7]. Следует учитывать, что метод ассоциативных правил позволяет выявлять зависимости между величинами, которые более вероятны, чем простое угадывание, но не дает никаких объяснений наличию таких зависимостей. Более того, возможно проявление ошибочных закономерностей вследствие статистических погрешностей, которые не имеют отношения к действительности. Таким образом, данный подход следует рассматривать как метод поддержки принятия решения, но не «принятие решения» в явном виде.

Будем считать, что у нас задано конечное множество биологических видов V. Пусть имеются данные о присутствии биологических видов на различных территориях, которые заданы в виде таблицы, строками являются виды, столбцами территориальные единицы, а в

ячейках присутствие (+) либо отсутствие (–) представителей данного вида на данной территориальной единице. Для использования метода поиска ассоциативных правил необходимо выделить транзакции. В данном случае в качестве транзакций могут быть использованы столбцы таблицы. Обозначим транзакции через T1, T2, …,TN. Далее рассматриваем всевозможные подмножества множества V (то есть наборы видов) с целью выяснения, какие из них образуют наиболее вероятные биоценозы. Следует отметить, что если общее количество видов в множестве V равно m, то таких наборов видов будет 2^m. Для каждого такого набора, который будем обозначать F, рассчитываем величину, называемую *поддержкой* (*support*):

$$Supp(F) = D(F)/N,$$

где $D(F)$ – количество транзакций, содержащих набор F, N – общее количество транзакций. Поддержка показывает, насколько данное ассоциативное правило является обоснованным, то есть насколько часто оно встречается в записях. Сама по себе эта характеристика не может рассматриваться как критерий для доверия или недоверия ассоциативному правилу. Поддержка всего лишь показывает на распространенность явлений, описываемых ассоциативным правилом. Безусловно, начинать изучение необходимо с ассоциативных правил с высокой поддержкой, так как они позволяют выявить наиболее распространенные явления. Но следует помнить, что ассоциативные правила с низкой поддержкой также могут принести много интересной информации и помочь выявить неочевидные закономерности.

После того, как рассчитана поддержка для всех наборов видов, может быть выбрана одна из двух стратегий:

1. Упорядочиваем наборы по убыванию поддержки и рассматриваем список сверху вниз. Такой подход наиболее оправдан и позволяет получать максимальное количество информации. Однако он требует априорных знаний о том, что в статистических данных малое количество ошибок. Даже низкий процент случайных погрешностей оказывает заметное влияние на формирование ассоциативных правил с низкой поддержкой.

2. Оставляем только наборы, поддержка которых не меньше некоторой минимальной величины Supp_min. Такой подход значительно сокращает дальнейшие вычисления, однако требует решения не очевидной задачи о выборе Supp_min. Однако такой подход необходим, если сбор данных допускает наличие статистических ошибок. Выбор

Supp_min позволяет исключить влияние случайных погрешностей на конечный результат.

На следующем шаге необходимо сформировать ассоциативные правила вида:

$$\textit{если } \{v_1,v_2,...,v_{i-1},v_{i+1},...,v_k\}, \textit{ то } v_i$$

для всех видов от v_1 до v_k. В дальнейшем, для краткости, будем использовать запись

$$\{v_1,v_2,...,v_{i-1},v_{i+1},...,v_k\} => v_i.$$

Для каждого из ассоциативных правил необходимо рассчитать величину, называемую достоверностью:

$$conf(F,v_i) = D(F) / D(v_i),\ (i = 1,...,k),$$

где $D(F)$ – количество транзакций, содержащих набор F, $D(v_i)$ – количество транзакций, содержащих вид v_i. Достоверность лежит в интервале от 0 до 1. Достоверность показывает, с какой вероятностью присутствие биологического вида v_i вытекает из присутствия остальных биологических видов, входящих в набор F. Чем она больше, тем больше вероятность, что вид попал в набор не случайно. Однако следует помнить, что возможные случайные совпадения даже для ассоциативных правил с высокой поддержкой. Некоторые ассоциативные правила могут быть следствием не природных закономерностей, а методики сбора и обработки информации.

Для ассоциативных правил существует еще ряд различных характеристик, однако мы пока ограничимся этими двумя, так как смысл остальных не так очевиден в применении к задачам поиска биоценозов.

Отдельный интерес представляют отрицательные ассоциативные связи, показывающие несовместимость биологических видов на одной территории. Отрицательная ассоциативная связь записывается в виде:

$$\textit{если } \{v_1,v_2,...,v_{i-1},v_{i+1},...,v_k\}, \textit{ то } v_i\ (\{v_1,v_2,...,v_{i-1},v_{i+1},...,v_k\} => \textit{not } v_i).$$

Трактовать отрицательную ассоциативную связь надо как вероятность отсутствия на данной территории вида v_i в случае присутствия видов из набора $\{v_1,v_2,...,v_{i-1},v_{i+1},...,v_k\}$. Для отрицательной ассоциативной связи также необходим расчет поддержки и достоверности.

Таким образом, анализ статистических данных, проведенный на основе методов поиска ассоциативных правил, позволяет выявлять новые закономерности сосуществования и взаимодействия биологических видов. В частности, удается установить положительные и отрицательные ассоциативные связи, которые находят объяснение на

основе экологического анализа. Так, наличие положительных ассоциативных связей свидетельствует о возможности совместного существования видов с различными экологическими нишами. Отрицательные ассоциативные связи свидетельствуют о конкуренции за ресурсы или наличия хищничества. Тем не менее, стоит помнить, что анализ статистических данных с целью выявления ассоциативных правил выполняет роль поддержки принятия решений, а окончательный результат может быть получен только исходя из экологического анализа выявленных закономерностей.

ЛИТЕРАТУРА

1. *Bray J. R., Curtis J. T.* An ordination of upland forest communities of southern Wisconsin // Ecological monographs. 1957. Vol. 27. P. 325–349.

2. *Jaccard P.* Étude comparative de la distribution florale dans une portion des Alpes et des Jura // Bulletin del la Société Vaudoise des Sciences Naturelles. 1901. № 37. P. 547–579.

3. *Sorensen T.* A method of establishing groups of equal amplitude in plant sociology based on similarity of species content // Kgl. danske vid. selskab. biol. krifter, 1948. № 4. P. 232–244.

4. *Hwang S., Kuznetsov I. B.* Bioinformatics Study of Functional Associations Observed in Multiple Sources of Human Genome Data // The Open Applied Informatics Journal. 2007. 1, 1–10.

5. Gabriela Czibula, Maria-iuliana Bocicor and istvan gergely czibula, promoter sequences prediction Using Relational Association Rule Mining// Evolutionary Bioinformatics 2012:8 181–196.

6. Hojung Nam1, KiYoung Lee2 and Doheon Lee*1 Identification of temporal association rules from time-series microarray data sets // BMC Bioinformatics 2009, 10(Suppl 3):S6.

7. *Agrawal R., Imielinski T., Swami A.* Mining Associations between Sets of Items in Massive Databases. Proc. ACM SIGMOD, 1993. P. 207–216.

8. *Кассал Б. Ю.* Гидробионты Средне-Иртышского района // Труды Зоологической Комиссии. Ежегодник. Вып. 3 : сб. науч. тр. / под ред. Б.Ю. Кассала. Омск : Издатель-Полиграфист, 2006. 155 с. С. 30–42.

9. *Кассал Б. Ю.* Животные Омской области: биологическое многообразие : монография. Омск : Изд-во АМФОРА, 2010. 574 с.

УДК 004.056

С.В. Белим, Н.Ф. Богаченко
Омский государственный университет им. Ф.М. Достоевского

ИССЛЕДОВАНИЕ СТРУКТУРЫ СПИСКОВ ДОСТУПОВ В СИСТЕМАХ С ИЕРАРХИЕЙ ОБЪЕКТОВ

Дискреционное разделение доступа является минимальным требованием к политике безопасности компьютерной системы для того, чтобы она считалась защищенной [1; 2]. Дискреционный подход к разграничению доступа реализован в большинстве современных операционных систем и систем управления базами данных [3–6].

В традиционной модели дискреционного разграничения доступа можно выделить два существенных несоответствия реальным системам. Прежде всего, объекты компьютерной системы, как правило, объединены в файловую систему средствами операционных систем. При этом файловая система имеет иерархическую логическую структуру, как правило, древовидную. Вследствие чего доступ к отдельному объекту может осуществляться двояко: либо напрямую, либо по ссылкам от других объектов. Для некоторых операционных систем, например семейства Windows, для двух разных способов обращения к файлу разрешительная подсистема может принять различные решения. Кроме этого, у каждого объекта свой список разрешенных доступов, который принято называть ACL (Access Control List). При этом ACL организован в виде структуры данных «список», а не структуры данных «одномерный массив», как это предполагается в обычной матрице доступов. В итоге в одном ACL может быть как разрешение на доступ, так и запрет на тот же доступ. Причем актуальной будет только одна запись, которая расположена в списке раньше. Классическая матрица доступов не допускает двух противоречивых записей. В связи с этим, актуальным является развитие модели дискреционного разделения доступа с учетом иерархии объектов и организацией данных о доступе в виде структуры, отличной от массива.

Важно выявить при каких условиях система сводится к классической матрице доступов. Данный вопрос является весьма важным в силу того, что основные результаты о безопасности систем с дискре-

ционным разграничением доступа получены исходя из предположения об использовании матрицы доступов. Выполнимость этих результатов применительно к системам с другим представлением данных о правах доступа не очевидна.

Системы, сводимые к матрице доступов, будем называть *матричными.* Таким образом, для каждой из дискреционных систем безопасности актуален вопрос, является ли она матричной. Более сложной является задача выявления и доказательства *нематричности* тех или иных систем. Если нематричные системы существуют, то возникает сложная проблема проверки истинности утверждений о безопасности, доказанных для матричных систем, применительно к нематричным системам.

Главное свойство подсистемы безопасности состоит в том, что решение о предоставлении либо отказе в доступе должно приниматься однозначно. Разделим все системы на два класса по ресурсам, используемым подсистемой безопасности для разрешения доступа:

1. *Локальные системы*, в которых принятие решения основывается только на анализе ACL.

2. *Нелокальные системы*, в которых при принятии решения используются кроме ACL объекта, также переменные окружения, такие как глобальные параметры, текущее состояние системы, режим работы системы и т.д.

Лемма 1 (о столбцах матрицы доступов). *Для локальных систем организация ACL в виде списка эквивалентна организации ACL в виде массива.*

Лемма 2. *Для нелокальных систем организация ACL в виде списка не сводится к ACL в виде массива.*

Далее остановимся на локальных системах. Рассмотрим структуру данных, обеспечивающую доступ к отдельным ACL. Она наследуется от структуры, отражающей логическую организацию объектов в системе. На сегодняшний день наиболее распространенной логической структурой объектов является дерево. Поэтому ACL в таких системах также объединены в дерево. Соответственно важным является вопрос наследования прав доступа с учетом иерархии в рамках дерева. Будем различать два крайних подхода:

1. ACL потомка никак не связан с ACL предка. В этом случае будем говорить, что система *без наследования*.

2. ACL потомка полностью включает в себя ACL предка. В этом случае будем говорить, что система *с полным наследованием*.

Лемма 3 (о строке матрицы доступов). *В системах с полным наследованием древовидная структура организации ACL может быть сведена к одномерному массиву.*

Теорема 1. *Локальные системы с полным наследованием являются матричными.*

В системах, которые не являются системами «с полным наследованием», ситуация сильно осложняется. Чаще всего результат запроса на доступ существенно зависит от способа доступа к объекту. В качестве примера приведем известную уязвимость операционных систем семейства Windows. Для одного и того же пользователя возможен запрет на доступ к папке, но разрешение на чтение файла внутри этой папки. Если пользователь движется последовательно по файловому дереву, то он не сможет прочитать файл, так как у него отсутствует доступ к папке. Но если пользователь из консоли введет прямой путь к файлу, то он сможет его прочитать. Таким образом, разрешение или запрет на доступ зависят от формы запроса.

Лемма 4. *Системы, в которых доступ к объектам возможен только последовательно по дереву, эквивалентны системам с полным наследованием.*

Лемма 5. *Системы с возможностью доступа не последовательно по дереву и отсутствием наследования не являются матричными.*

Рассмотрим теперь системы, в которых права доступа к объекту зависят только от начальной точки пути доступа по дереву логической организации объектов. Такие системы могут быть описаны в рамках общего подхода с помощью введения набора составных прав доступа вида (r, A) = «получение права r из точки файловой системы A». Легко понять, что такой набор прав будет конечным. Также легко показать, что по отношению к новому набору составных прав доступа, системы, зависящие от пути доступа, будут матричными. Такие системы будем называть *квазиматричными*.

В дальнейшем планируется исследовать матричность операционных систем семейств Unix, Linux и Windows, а также системы управления базами данных Oracle.

ЛИТЕРАТУРА

1. Гостехкомиссия России. Руководящий документ: Средства вычислительной техники. Защита от несанкционированного доступа к информации. Показатели защищенности от несанкционированного доступа к информации. М. : ГТК 1992.

2. Гостехкомиссия России. Руководящий документ: Защита от несанкционированного доступа к информации. Термины и определения. М. : ГТК 1992.

3. *Немет Э., Снайдер Г., Хейн Т.* Unix и Linux. Руководство системного администратора. М. : Вильямс, 2012. 1312 с.

4. *Руссинович М., Соломон Д.* Внутреннее устройство Microsoft Windows. СПб. : Питер, 2005. 992 с.

5. *Тьеро М., Ньюмен А.* Oracle. Руководство по безопасности. М. : ЛОРИ, 2004. 560 с.

6. Linux с улучшенной безопасностью. Руководство Пользователя. Документация к Fedora Linux. URL: https://docs.fedoraproject.org/ru-RU/Fedora/13/html/Security-Enhanced_Linux/index.html (дата обращения: 25.09.2013).

УДК 004.056

С.В. Белим, Н.Ф. Богаченко
Омский государственный университет им. Ф.М. Достоевского

ВИДЫ АССОЦИАТИВНЫХ ПРАВИЛ МЕЖДУ ПРАВАМИ ДОСТУПА ДИСКРЕЦИОННОЙ ПОЛИТИКИ БЕЗОПАСНОСТИ

Согласно требованиям к защите информации, дискреционное разграничение доступа является основной реализацией разграничительной политики доступа к конфиденциальной информации [1; 2] и реализовано в большинстве современных операционных систем [3–5]. Вместе с тем, во многих операционных системах распространена ситуация, в которой между правами доступа к объектам существуют ассоциативные правила вида:

«если r_1, то r_2»,

означающие, что если субъект обладает правом доступа r_1 к некоторому объекту, то он обладает или может получить право r_2 на этот же объект.

В качестве примера можно привести иерархию прав в операционных системах семейства Windows. Например, «общее право на чтение объекта» влечет за собой «специальное право чтения данных в объекте» и «стандартное право чтения DACL».

Следует отметить, что существует два вида ассоциативных правил между правами. Первый вид подразумевает возможность получения права при первом же запросе на доступ к объекту. Такие ассоциативные правила назовем *монооперационными*, так как получение права осуществляется за одну операцию. Второй вид ассоциативных правил требует выполнение субъектом нескольких операций для получения второго права. Такие ассоциативные правила назовем *мультиоперационными*.

В дальнейшем монооперационное ассоциативное правило, связывающее права доступа r_1 и r_2 к некоторому объекту, будем обозначать как $r_1 \rightarrow r_2$. Аналогично, мультиоперационное ассоциативное правило обозначим двойной стрелкой: $r_1 \Rightarrow r_2$.

Примерами моноопераційних правил являються все иерархические отношения между правами в операционных системах Windows.

В качестве примера мультиоперационных ассоциативных правил можно привести следующую ситуацию, также имеющую место в операционных системах семейства Windows. Пусть r_1 = «право на запись в DACL», а r_2 = «право на чтение файла», тогда существует мультиоперационное ассоциативное правило $r_1 \Rightarrow r_2$, так как пользователь может осуществить «запись в DACL права на чтение файла» на первом шаге и «чтение файла» на втором шаге.

Как легко понять, существование моноопераційних ассоциативных правил, как правило, является следствием обобщения простых правил в составные для удобства администрирования безопасности системы. Операцию расщепления составного права на более простые принято называть декомпозицией прав. Учитывая операции обобщения и декомпозиции прав можно построить *ориентированный граф моноопераційних ассоциативных правил*. Очевидно, что для построения такого орграфа не следует учитывать ассоциативные правила, получаемые по транзитивности. Принято считать, что орграф моноопераційних ассоциативных правил является деревом. Но данный вопрос требует дополнительного исследования применительно к той или иной операционной системе.

Используя описанный выше орграф, несложно доказать, что моноопераційные ассоциативные правила не оказывают существенного влияния на безопасность системы: *если в системе с дискреционным разграничением доступа присутствуют только моноопераційные ассоциативные правила, то подсистема безопасности такой системы эквивалентна подсистеме безопасности некоторой системы без ассоциативных правил.*

Ситуация значительно усложняется при наличии в системе мультиоперационных ассоциативных правил. Согласно дискреционной политике безопасности, основанной на матрице доступов, доступ субъекта к объекту с правом r разрешен тогда и только тогда, когда это право присутствует в соответствующей ячейке матрицы доступов M. Мультиоперационные ассоциативные правила приводят к тому, что при прямом запросе доступ может быть запрещен, но доступ становится возможным, если субъект выполнит ряд других доступов. Возникает неоднозначная, с точки зрения возможных каналов утечки прав, ситуация.

Неоднозначность может быть устранена, если для анализа утечек прав доступов использовать не матрицу доступов M, а *матрицу возможных доступов* MV. Ячейка матрицы возможных доступов $MV[s, o]$ содержит право r, если возможен доступ субъекта s к объекту o с правом r либо напрямую, либо в результате выполнения ряда операций.

Основываясь на древовидности орграфа монооперационных ассоциативных правил, можно доказать, что *для систем только с монооперационными ассоциативными правилами матрица возможных доступов совпадает с матрицей доступов.*

В общем случае построение матрицы возможных доступов осложняется тем, что в некоторых системах возникает ситуация зацикливания ассоциативных правил. Приведем пример такого зацикливания, которое может наблюдаться в операционных системах семейства Windows. Пусть заданы три права:

r_1 = «общее право записи»,

r_2 = «право записи в DACL»,

r_3 = «полный доступ».

Рассмотрим ассоциативные правила между этими правами. «Общее право записи» в результате декомпозиции включает в себя «право записи в DACL», то есть выполняется монооперационное правило $r_1 \rightarrow r_2$. Используя «право записи в DACL», пользователь может внести в DACL любое право в том числе и право на «полный доступ». В результате пользователь получает полный доступ к объекту, то есть выполняется мультиоперационное ассоциативное правило $r_2 \Longrightarrow r_3$. Из «полного доступа» в результате декомпозиции следует «общее право записи», то есть выполняется монооперационное ассоциативное правило $r_3 \rightarrow r_1$.

В дальнейшем необходимо исследовать вопрос алгоритмической разрешимости задачи построения матрицы возможных доступов для произвольной системы при известной матрице доступов и заданном наборе ассоциативных правил. Кроме того, планируется рассмотреть влияние ассоциативных правил на безопасность современных операционных систем.

ЛИТЕРАТУРА

1. Гостехкомиссия России. Руководящий документ: Средства вычислительной техники. Защита от несанкционированного доступа к информации. Показатели защищенности от несанкционированного доступа к информации. М. : ГТК 1992.

2. Гостехкомиссия России. Руководящий документ: Защита от несанкционированного доступа к информации. Термины и определения. М. : ГТК 1992.

3. *Белим С. В.* Защита в операционных системах : учебное пособие. Омск : ОмГУ, 2011. 52 с.

4. *Руссинович М., Соломон Д.* Внутреннее устройство Microsoft Windows. СПб. : Питер, 2005. 992 с.

5. Linux с улучшенной безопасностью. Руководство Пользователя. Документация к Fedora Linux. URL: https://docs.fedoraproject.org/ru-RU/Fedora/13/html/Security-Enhanced_Linux/index.html (дата обращения: 25.09.2013).

УДК 519.86

С.В. Белим, А.Н. Мироненко
Омский государственный университет им. Ф.М. Достоевского

ПРОТОКОЛ ОДНОРАЗОВЫХ ПАРОЛЕЙ НА ОСНОВЕ НЕЙРОННЫХ СЕТЕЙ

Технология одноразовых паролей находит все большее распространение в системах аутентификации пользователей и рабочих станций [1; 2]. Как правило, системы одноразовых паролей строятся исходя из криптографических протоколов с применением различных алгоритмов шифрования. В данной работе предложен принципиально новый подход с реализацией системы одноразовых паролей без явных алгоритмов шифрования с применением искусственных нейронных сетей [3] и функций хеширования.

Предлагаемая технология основывается на двух протоколах – протоколе аутентификации и протоколе инициализации нейросетей. Рассмотрим сначала протокол аутентификации.

Пусть у двух абонентов A и B есть одинаково обученные нейросети NA и NB, а также общая ключевая хеш-функция h(k,M). Ключ k является общим и держится в секрете. Нейросети одинаковы по архитектуре и обучены на одинаковом обучающем наборе. Протокол аутентификации состоит из следующих шагов:

1) A->B: запрос на аутентификацию.

2) B: вырабатывает случайное число R. Формирует на основе R входной вектор для нейросети NB и получает отклик b.

3) B->A: R.

4)A: Подает случайное число R на свою нейросеть NA и получает отклик a.

5) A->B: a.

6) B: сравнивает значения a и b. Если они равны, то аутентификация прошла успешно.

7) B->A: ATHENTICATION_TRUE.

8) A,B: дообучают свои нейросети, добавляя к обучающему набору значения h(k,R) на входном векторе R.

Для атаки на протокол злоумышленнику необходимо обучить аналогичную нейросеть зная начальное состояние и значения ключевой хеш-функции на случайных векторах, передаваемых по открытой сети.

Рассмотрим протокол начального обучения сетей:

1) A и B вырабатывают общий ключ k по какому-либо криптографическому протоколу, например, Диффи-Хеллмана.

2) A и B выполняют следующую последовательность действий:

$h_0 = h(k,k)$, $h_{i+1} = h(k,h_i)$, $i = 1,\ldots,M$.

M – открытая константа.

Обучение нейросети проводится на наборах данных (h_i,h_{i+1}), $i = 1,\ldots,M-1$.

Для определения стойкости протокола проведен анализ на случайность данных, передаваемых по открытой сети с помощью компьютерного эксперимента.

ЛИТЕРАТУРА

1. *Lamport L, Malkhi D., Zhou L.* Stoppable Paxos // Microsoft Research (28 April 2008).

2. *Lamport L.* Time, clocks, and the ordering of events in a distributed system // Communications of the ACM. 1978. V. 21 (7). P. 558–565.

3. *Вороновский Г. К., Махотило К. В., Петрашев С. Н., Сергеев С. А.* Генетические алгоритмы, искусственные нейронные сети и проблемы виртуальной реальности. М. : Основа, 1997.

УДК 519.86

А.Н. Мироненко

Омский государственный университет им. Ф.М. Достоевского

ФИЛЬТРАЦИЯ СМС-СПАМА С ПОМОЩЬЮ НЕЙРОННЫХ СЕТЕЙ И SMV

Все реже возникает проблема спама в электронной почте и все чаще возникает новая проблема – спам в смс сообщениях. Исследования в данной области практически отсутствую, поэтому можно говорить об актуальности решения задачи фильтрации смс спама.

В работе предлагается алгоритм определения принадлежности смс сообщения к спаму, по его содержанию. В алгоритме предлагается использовать нейронные сети в сочетании с SMV (методом опорных векторов) [1; 2].

Алгоритм подготовки данных для дальнейшей фильтрации [3]:

1. На основе обучающего набора сообщений формируется словарь слов (термов), в котором каждому терму соответствует два числа – частота встречаемости в спам-СМС и частота встречаемости в обычных сообщениях.

2. Формируется двумерное множество точек. Каждому терму соответствует одна точка с двумя координатами. По оси абсцисс появление слова в обычных СМС, по оси ординат – в спам-СМС.

3. Для полученного множества точек решается задача таксономии.

4. Для каждого таксона определяются координаты центра масс.

5. Таксоны упорядочиваются по величине $G = Y/X$, где X и Y – координаты центра масс таксона.

6. Вводится равномерная шкала, приписывающая каждому таксону уровень спамности в интервале от 0 до 1 (значения 0 и 1 не используются).

Алгоритм определения спамности СМС сообщения [3]:

1. Для каждого «пришедшего» СМС сообщения производится разбор его на отдельные термы.

2. Определяется принадлежность каждого терма к одному из таксонов.

3. Формируется вектор, характеризующий данное сообщение. Координатами вектора служат количества термов с данным значением спамности.

4. Полученный вектор подается на вход нейросети. В качестве весовых коэффициентов входных синапсов выбирается количество термов с данным коэффициентом спамности в частотном словаре.

5. Выходной сигнал нейросети, лежащий в интервале от 0 до 1, интерпретируется как одно из трех решений: сообщение является спамом (R1), сообщение не является спамом (R2), невозможно определить является ли сообщение спамом или нет (R3). Значения R1, R2 и R3 выбираются экспериментально.

ЛИТЕРАТУРА

1. *Хайкин С.* Нейронные сети: полный курс. 2-е изд. : пер. с англ. М. : Изд. дом «Вильяме», 2006. 1104 с.

2. *Christopher M.* Bishop // Pattern recognition and machine learning (2006).

3. *Мироненко А. Н.* Алгоритм контентной фильтрации спама на базе совмещения метода опорных векторов и нейронных сетей : автореф. дис. канд. тех. наук. СПб., 2012. С. 6–7.

УДК 004.492.3

Р.С. Прохоров
Омский государственный университет им Ф.М. Достоевского

ИДЕНТИФИКАЦИЯ ПРОЦЕССОВ В ОПЕРАЦИОННОЙ СИСТЕМЕ WINDOWS 7 ПО ИХ ПОВЕДЕНИЮ

В данной статье описывается методика анализа поведения процессов в операционной системе Windows 7. Данная методика позволяет идентифицировать процесс по его действиям в среде с достаточно высокой точностью. Анализатор способен опознавать один архетип процессов по принципу «свой - чужой».

Анализ поведения процессов, нейронный сети, анализ поведения, нейросетевые структуры.

Введение

На данный момент существует два основных метода по классификации процессов, запущенных в компьютерной системе: сигнатурный анализ и анализ поведения. Наиболее распространенным является сигнатурный анализ [1–4], когда тело исполняемого файла подвергается проверке на наличие последовательностей «опасных» команд. В последнее время увеличивается число систем и методик по бихевиористическому анализу или анализу поведения [5–7] уже действующего в системе процесса. Также существуют и иммунные системы, основанные на комбинации этих типов анализа. [8]. В данной статье описана методика классификации процессов на основе анализа поведения.

Постановка задачи

Основной задачей описываемого анализатора является распознавание процессорного поведения и присвоение рассматриваемому процессу архетипа «свой» или «чужой». В качестве входных данных используется последовательность событий, произведенных всеми процессами за некоторое время их функционирования.

Теоретический анализ

Основная идея бихевиористического анализа состоит в том, чтобы представить поведение субъекта в среде в виде последователь-

ности событий, связанных с данным субъектом: таких событий, которые были порождены им непосредственно.

Так как в компьютерной среде могут произойти только события конечного числа классов, было определено множество классов событий E для операционной системы Windows 7.

Поведение каждого конкретного процесса было представлено в виде последовательности событий $A = (a_1,a_2,...,a_n)$, где каждое a_i имеет строго один класс $e \in E$.

Методика классификации

Весь поток событий в системе сортируется по процессам, которые его породили. Таким образом, получается несколько последовательностей по числу процессов, функционирующих в системе на время регистрации потока событий. Каждая такая последовательность – поведение отдельно взятого процесса. Далее каждая такая последовательность разбивается на подпоследовательности малой длины. Установлено, что наиболее эффективной длиной такой подпоследовательности будет являться число, соизмеримое с числом классов обрабатываемых событий.

Для обработки множества подпоследовательностей используется анализатор, основанный на нескольких персептронах вида N-N-2 с сигмоидной функцией активации и способный опознавать строго один архетип процесса и выделять «свои» подпоследовательности из общего потока подпоследовательностей.

Число классов событий чрезмерно велико (157), что порождает следующую проблему: нейронные сети неспособны обработать входные векторы вида (0,0,...,K,...,0), где $K \in N$, а остальные – нули. Для предотвращения данной проблемы была построена специальная структура, которая содержит несколько простых персептронов вида N-N-2, а также модуль разбиения входа и модуль согласования выхода (арбитра). В данной статье рассмотрена группа из четырех нейронных сетей: 3-3-2, 5-5-2, 5-5-2 и 4-4-2. Обучение каждого персептрона осуществляется методом обратного распространения ошибки.

Компьютерный эксперимент и результаты

Для проведения эксперимента был сформирован общий поток событий длиной в восемь миллионов событий с помощью библиотеки Event Trace for Windows (ETW) [9] и разбит на три множества: обучающее, тестирующее и проверяющее.

Как показывают результаты исследования, данный подход крайне эффективен для «редких» процессов со специфическим поведением (максимальная эффективность – 0,99) и средне эффективен для «популярных» процессов (минимальная эффективность – 0,66) с довольно широким спектром порождаемых событий.

Заключение

В данной статье был рассмотрен подход, позволяющий осуществить идентификацию процесса по его поведению. Эффективность колеблется в районе 0,66–0,99. Так как вредоносное программное обеспечение зачастую является сугубо специфичным, то такая система должна достаточно эффективно определять архетип данных процессов. Преодолена проблема входных векторов для нейронных сетей, почти полностью заполненных нулями.

ЛИТЕРАТУРА

1. *Булахов Н. Г., Калайда В.Т.* Методы обнаружения и обезвреживания саморазмножающихся вирусов // Доклады ТУСУРа. 2008. № 2 (18). Ч. 1. июнь. С. 78–82.

2. *Christodorescu M. , Jha S., Seshia S., Song D., Bryant R.* Semantics-aware malware detection // the IEEE Symposiumon Security and Privacy, 2005.

3. M. C. a. S. Jha, "Static analysis of executables to detect malicious patterns," USENIX Security Symposium, 2003.

4. R. G. a. J. L. J. Dai, "Efficient Virus Detection Using Dynamic Instruction Sequences," Journal of Computers, 2009.

5. *Валеев С. С., Дьяконов М. Ю.* Нейросетевая система анализа аномального поведения вычислительных процессов в микроядерной операционной системе // Вестник УГАТУ. 2012. Т. 14. № 5 (40). С. 198–204.

6. David Lo, Hong Cheng,, Jiawei Han, SiauCheng Khoo, Chengnian Sun. Classification of Software Behaviors for Failure Detection: A Discriminative Pattern Mining Approach. // KDD'09, Июнь 28 – Июль 1, 2009, Париж, Франция.

7. Ahmadi M., Sami A., Rahimi H., Yadegari B. Iterative System Call Patterns Blow the Malware Cover // Security for The Next Generation, 2011.

8. *Ваганов М. Ю.* Гибридная искусственная иммунная система защиты компьютера от процессов с аномальной активностью: диссертация кандидата технических наук: 05.13.19. Санкт-Петербург, 2012. 92 с.

9. Event Tracing (Windows). URL: http://msdn.microsoft.com/ru-RU/library/windows/desktop/bb968803(v=vs.85).aspx , 18.03.2013.

УДК 004.7

И.Б. Плесовских
Омский государственный университет им. Ф.М. Достоевского

ГЕНЕТИЧЕСКИЙ АЛГОРИТМ ОПТИМИЗАЦИИ ТОПОЛОГИИ ГЛОБАЛЬНОЙ СЕТИ

Оптимизация топологии является важной частью построения вы-числительных сетей. Топология сети тесно связана с функционированием сети, ее пропускной способностью, надежностью и стоимостью. Одним из подходов к оптимизации топологии вычислительных сетей является применение генетических алгоритмов [1; 2].

Целью является разработка генетического алгоритма, результатом работы которого должна являться вычислительная сеть с оптимизированной топологией с минимальной суммарной длиной связей, обеспечивающей минимальную среднюю задержку и максимальную надежность при передаче информации от одного компьютера данной сети к другому.

Исходными данными является набор узлов – коммутаторов сети, которым присвоены декартовы координаты. Также узлам присваиваются максимальный размер трафика, который они могут генерировать в единицу времени, и максимальный размер трафика, который они могут обработать без дополнительной задержки. Коэффициент готовности коммутатора характеризует вероятность того, что данное устройство находится в работоспособном состоянии в момент получения информации от другого узла. Таким образом, задача сводится к построению графа, имеющего минимальный суммарный вес всех дуг, минимальную среднюю суммарную задержку при прохождении информации между узлами, а также максимальную общую готовность, то есть вероятность того, что информация от какого-либо узла дойдет до точки назначения без потерь. Стоимость построения сети в данном случае эквивалентна суммарной длине всех связей между узлами.

При решении поставленной задачи в качестве хромосомы используется матрица смежности вида $A = (a_{ij})$, где элемент a_{ij} равен 1, если существует связь между i-м и j-м узлами, и 0 в противном случае.

В целом, предлагаемый алгоритм представляет собой последовательность итераций. Каждая итерация, называемая поколением, состоит из нескольких шагов:

1. Выборка хромосом для скрещивания;

2. Скрещивание;

3. Мутация хромосом, получившихся в результате скрещивания;

4. Отбор наиболее приспособленных хромосом в новую популяцию.

Критерием отбора хромосом является функция приспособленности F, вычисляемая следующим образом:

$$F = k_1F_1 + k_2F_2 + k_3(\mathrm{F}_3)^{-1} + R, \qquad (1)$$

где k_1, k_2, k_3 – весовые коэффициенты, причем $k_1+k_2+k_3=1$, функция F_1 вычисляет сумму весов всех дуг графа, F_2 вычисляет среднюю суммарную задержку при прохождении информации между двумя любыми вершинами графа, F_3 вычисляет общий коэффициент готовности данной сети, R – штрафная функция, $R = 0$, если в сети нет неподключенных узлов, $R = M$, где M – некоторое достаточно большое число, в противном случае.

Для проверки работы алгоритма было проведено несколько компьютерных экспериментов с различными вариантами исходных данных. В результате было выявлено, что данный алгоритм может быть применен для построения глобальных вычислительных сетей с приемлемыми параметрами функционирования.

ЛИТЕРАТУРА

1. *Al-Bassam B., Alheraish A., Bakry S. H.* A tutorial on using genetic algorithms for the design of network topology // International Journal of Network Management. 2006. № 16(4). P. 253–262.

2. *Гужва Д. Ю.* Эволюционный синтез VPN-сетей в инфотелекоммуникационных системах // Вестник РГРТУ. 2009. № 1(27). С. 22–27.

УДК 65.012.810

С.В. Белим, Ю.С. Ракицкий
Омский государственный университет им Ф.М. Достоевского

ОБЪЕКТНО-ОРИЕНТИРОВАННАЯ МОДЕЛЬ ЗАЩИЩЕННОГО ДОКУМЕНТООБОРОТА

Моделирование документооборота является важным этапом в процессе хранения и передачи информации. Обычно моделирование строится на основе схем бумажного документооборота, традиции которого насчитывают несколько тысячелетий. Затем построенные модели переносятся на электронный документооборот, который должен обеспечивать преемственность бумажного документооборота и гарантировать непрерывность работы организации при внедрении средств автоматизации.

Первые модели документооборота были построены при внедрении средств автоматизации в правительстве США и министерстве обороны США. Модель Белла-ЛаПадулы достаточно подробно описывает состояние документов и системы документооборота в целом. Однако основной упор в данной модели сделан на разграничение доступа к уже существующим документам. Авторы модели учли как дискреционное, так и мандатное разграничение доступа. Основой модели являются свойства безопасности системы и теоремы об условиях их выполнения. Модель военных сообщений (MMS) описывает документооборот в министерстве обороны США и носит менее формальный характер. Большинство утверждений модели сформулированы не математическими отображениями и условиями, а в словесной форме, что затрудняет ее использование в автоматизированных системах документооборота.

В настоящее время развитие систем автоматизированного документооборота приобрело значительный размах, что привело к активности в развитии соответствующих математических моделей. Так в статье [1] авторы предприняли попытку разработки классов безопасности для систем документооборота по аналогии с соответствующими классами для автоматизированных систем. Ряд работ посвящен разработке и эксплуатации конкретных систем документооборота. Так в

статье [2] авторы обсуждают систему документооборота в университете, а в работе [3] рассмотрены системы специального назначения. В ряде статей исследуются методы моделирования. Так в статье [4] предлагается когнитивный подход к моделированию, а в статье [5] функциональное моделирование. Также исследованию подвергся такой важный вопрос как техническая нагрузка на автоматизированную систему создаваемая автоматизированным документооборотом [6].

В каждой организации существуют стандартные виды документов, для которых четко задана их структура. В качестве примера можно привести приказы, распоряжения, служебные записки, листы согласования и так далее. Применим объектно-ориентированный анализ для моделирования соответствующих документов.

Отнесем все документы одного типа к некоторому классу. То есть, если задан класс документов C, то все документы имеют одинаковую жестко заданную структуру. Описание класса документов сводится к заданию шаблона, по которому будет создаваться документ, а также методов создания, согласования и работы с документом. По аналогии с объектами в языках программирования высокого уровня будем считать, что документ является объектом и содержит поля и методы их обработки. Причем поля могут быть двух видов, отражающих режим доступа к ним – открытые и закрытые. Однако существует и отличие от объектно-ориентированного программирования, состоящее в том, что режим доступа к полю может меняться в процессе существования документа. Это требование вытекает из того, что текст документа может быть открыт для редактирования в процессе его создания, и закрыт для редактирования после его подписания.

Зададим более строго структуру документа d, относящегося к классу C. Документ состоит из полей, которые мы будем обозначать, через a_i и методов m_i. Доступ к полю будем обозначать через $d.a_i$. Активизацию метода будем обозначать, через $d.m_i$. Каждое поле в свою очередь состоит из трех полей $a_i.t$, $a_i.p$ и $a_i.l$. Поле $a_i.t$ содержит некоторые записи. Поле $a_i.p$ определяет текущий режим доступа к полю a_i. Поле $a_i.l$ определяет уровень доступа к полю.

Рассмотрим стандартные поля, присущие подавляющему количеству документов. Пусть имеется документ d. Обозначим через $d.a_1$ поле, отвечающее за основное содержание документа. Текст документа будет содержаться соответственно в поле $d.a_1.t$. $d.a_2$ – автор документа, $d.a_3$ – время создания документа. Таким образом, заполнение

этих полей соответствует написанию текста, постановке подписи и даты. Кроме этого всегда необходимо поле $d.a_4$ – атрибуты документа. В качестве атрибутов могут выступать специальный бланк или записи на полях документа. Далее может следовать любое количество полей, отводимых для подписей и резолюций на документе должностных лиц.

Немаловажной составляющей защищенного документооборота является разграничение доступа к полям между пользователями системы. Обозначим множество пользователей системы, через U. Определим отображение

$$Level: U \rightarrow L,$$

где L – некоторое множество, на котором определено отношение порядка, задающее иерархию пользователей в организации. Простейшим случаем L является линейно упорядоченное множество. Однако не все организации обладают строгой линейной структурой, поэтому в качестве L может выбираться любая алгебраическая решетка. Элементы множества L также заносятся в поля $d.a_i.l$ и определяют уровень пользователей, которым разрешено заполнение данного документа.

Методы объектов определяют его жизненный цикл. У каждого объекта свой жизненный цикл, поэтому процедуры обращения с документом должны быть реализованы именно как методы, а не глобальные процессы для всей системы.

Отметим обязательные методы присущие любому документу:

1. Constructor – конструктор класса, определяющий порядок создания документа.

2. Remove – деструктор класса, определяющий порядок вывода документа из обращения и уничтожения.

3. Exec – порядок использования документа в процессе деятельности организации.

4. Arxiver – порядок архивирования документа и хранения документа в архиве.

Возможно существование и других методов, специфичных для того или иного класса документов.

Любой документ в своем жизненном цикле может находиться, как минимум, в четырех различных состояниях. Рассмотрим каждое состояние отдельно.

1. *Создание документа*. Это состояние описывается конструктором класса (Constructor). Создание документа сводится к последова-

тельному заполнению полей объекта класса. Причем должен существовать определенный порядок доступа к полям документа для заполнения. Требование последовательного доступа к заполнению полей соответствует сложившейся практике создания и последовательного согласования документа.

Определение 1. Допустимый порядок заполнения полей документа будем называть политикой заполнения документа.

Для корректного построения политики заполнения документа необходимо ввести отношение порядка на множестве полей класса. Возможны два случая – в каждый момент доступно для заполнения только одно поле, либо существуют моменты, когда для заполнения доступно сразу несколько полей. В первом случае будем говорить о *монопольном* формировании документа, во втором – о *мультипольном* формировании документа.

Утверждение 1. Для реализации монопольного формирования документа поля класса должны образовывать линейно упорядоченное множество.

Каждое поле документа состоит в свою очередь из двух подчиненных полей. Первое подчиненное поле является содержательным и включает некоторые данные. Второе подчиненное поле является флагом, свидетельствующим о завершении заполнения поля. В случае бумажного документооборота первое подполе включает, например, резолюцию ответственного лица, а второе подполе – его подпись. Для электронного документооборота первое подполе по-прежнему содержит резолюцию, а второе – электронную подпись или сертификат пользователя.

Доступность того или иного поля определяется политикой заполнения документа. В большинстве случаев политика заполнения может быть сформулирована следующим образом:

Поле документа доступно для заполнения, если заполнены все поля, над которыми доминирует данное поле.

Процесс создания документа считается завершенным, если заполнены все его поля. После создания документ переходит в одно из других состояний.

Определение 2. Формирование документа будем считать безопасным, если во все моменты времени выполняется политика заполнения документа.

Обеспечение безопасности формирования документа должно обеспечиваться корректной реализацией конструктора класса. Кроме того конструктор определяет порядок заполнения полей документа. Пусть последовательность заполнения полей документа имеет вид:

$$a_1 \rightarrow a_2 \rightarrow ... \rightarrow a_k.$$

Причем последовательность является строгой, то есть не допустим пропуск полей (монопольное заполнение полей). В дальнейшем, для удобства, будем считать, что поля пронумерованы в порядке заполнения. Кроме порядка заполнения полей важным является порядок доступа пользователей к полям документа. Пусть иерархия пользователей в организации определяется ориентированным графом G. Учтем принцип, распространенный в подавляющем количестве организаций: «подчиненный не может визировать документ позже начальника». В рамках рассматриваемой модели это приводит к следующему утверждению.

Утверждение 2. Пусть реализуется монопольный порядок доступа формирования документа. Если поле a_k доступно для заполнения пользователю u, а поле a_{k+1} доступно для заполнения пользователю u', то либо u доминирует в G над u' (u>u'), либо u и u' не сравнимы в G (u<>u').

Рассмотрим безопасность документа в процессе его формирования. Иерархия пользователей в организации определяется ориентированным графом G, при этом G соответствует решетке безопасности L. Тогда при формировании документа, заполнение полей документа d осуществляется в соответствии с некоторой подрешеткой L_1 решетки L, при этом L_1 задает иерархию $u_1<u_2<u_3<...<u_n$ (т. е. для каждого класса документа существует подрешетка, в которой элементы линейно упорядочены). Далее, пусть Level(u) - уровень пользователя в решетке безопасности, т.е. метка пользователя. Введем обозначение d.a.s - пользователь, имеющий доступ к полю a подписал документ d подписью s. Тогда возможно сформировать следующие требования к процедуре подписания документа:

1. $d.a_i.l$ = Level(u). Уровень пользователя соответствует уровню поля документа, к которому осуществляется доступ.

2. $d.a_{i-1}.s$ = true. Пользователь u_{i-1} с более низкоуровневой меткой уже подписал документ.

3. $d.a_i.s$ = false. Текущий пользователь еще не подписал документ.

Будем говорить, что пользователь u осуществляет доступ к документу d с правом p ($u \rightarrow_p d$), если p - составное право, при котором пользователь имеет доступ на запись к полю $d.a_i.s$ (т.е. право поставить подпись под документом), доступ на запись в поле $d.a_i.t$ (поле для резолюции пользователя) и доступ на чтение всех остальных полей.

Для отслеживания состояния документа в процессе его заполнения необходимо ввести метод trace. Данный метод необходим для отслеживания места нахождения документа и его передачи между пользователями. Работу метода trace для документа d можно описать следующим образом:

$d.trace(a_i)$

if i = n then message(a_1); exec(d)
else if $d.a_i.s$ = true then
if $d.a_i.t$ = NULL then i = i + 1
else message(a_1, $d.a_i.t$); remove(d)

Вызов метода trace для документа d означает ряд проверок. В первую очередь необходимо проверить, не достигнут ли конец списка пользователей, визирующих документ (i=n). Если конец списка достигнут, то автор документа получает уведомление об успешном создании документа, метод Constructor завершает свою работу и вызывается метод Exec(). Когда конец списка не достигнут, это означает, что не все подписи поставлены, метод trace проверяет наличие подписи у текущего пользователя ($d.a_i.s$ = true), если таковая присутствует, то проверяется поле, предназначенное для резолюции. Считается, что документ подписан без замечаний и рекомендаций к изменению в том случае, если поле резолюции осталось пустым ($d.a_i.t$ = NULL). В этом случае документ пересылается следующему в иерархии пользователю (i-i+1). Если же поле резолюции не пустое, то документ d считается не прошедшим согласование, автору документа посылается уведомление с содержанием резолюции message(a_1, $d.a_i.t$), после чего вызывается метод remove для уничтожения документа.

Таким образом, безопасность документа в состоянии Constructor будет зависеть от надежности прохождения маршрута согласования (документ либо пройдет маршрут до конца и его создание будет завершено, либо будет уничтожен, а автор документа получит уведомление с причиной уничтожения), а также от соблюдения политики безопасности организации при прохождении документа по маршруту (нет утечки информации, неавторизованные лица не получат доступ к

документу). Сформулируем критерий безопасности документа в состоянии Constructor.

Утверждение 3. Документ d безопасен в состоянии Constructor, если до каждой постановки подписи выполняются условия $d.a_i.l = Level(u)$, $d.a_{i-1}.s = true$, . $d.a_i.s = false$ и после каждой постановки подписи вызывается метод trace.

2. *Уничтожение документа*. Данное состояние описывается деструктором класса (Destructor). Уничтожение документа возможно только в том случае, когда на текущий момент документ находится в состоянии архивного хранения и выполнены условия завершения его хранения. Как правило, условия завершения хранения носит характер временной метки. Причем возможны как условия хранения одинаковые для всех документов одного класса, так и индивидуальные условия для отдельно взятого документа. Таким образом, при описании класса необходимо предусмотреть три дополнительных поля, создаваемых в каждом документе. Первое поле показывает дату перевода в архивное состояние. Это поле заполняется не при создании документа, а в процессе его эксплуатации. Второе поле задает стандартные сроки хранения для документов данного класса. Третье поле определяет индивидуальные сроки хранения для выделенного документа. Третье поле может быть пустым, в этом случае сроки хранения стандартные, либо заполняться при создании документа, если существует соответствующая инструкция в организации, либо заполняться при переводе документа в архивное состояние.

В данном случае мы сталкиваемся с полями, заполняемыми не при создании документа, а в процессе его функционирования. Доступ к соответствующим полям будет регламентирован ниже при описании порядка использования документа.

3. *Архивирование документа.* Данное состояние описывается методом *Arxiver*. Перемещение документа в архив инициирует пользователь, последним поставивший резолюцию. Перемещение документа на архивное хранение может быть выполнено только тогда, когда все поля документа, предназначенные для подписей и резолюций заполнены. Если данное условие выполнено, то заполняются поля, отведенные под временные метки. Также перемещение документа на архивное хранение предполагает установку на все поля закрытого режима доступа, т. е. редактировать нельзя ни одно поле документа. При истечении времени хранения документа в архиве вызывается метод Destructor.

4. *Использование документа.* Данное состояние описывается методом Exec. Переход в состояние использования предполагает, что все поля документа заполнены и документ доступен для просмотра. Документ доступен для просмотра пользователям с уровнем доступа $a_i.l$, при этом режим доступа $a_i.p$ для любого поля установлен только на просмотр, т.е. изменять содержимое документа нельзя.

ЛИТЕРАТУРА

1. *Дуйков Е. А., Сотский С. В., Щербаков А. Ю., Кирин В. И.* Формулирование требований к защищенному электронному документообороту // Вопросы защиты информации. 2008. № 4. С. 28–32.

2. *Пителинский К. В., Хачатрян А. Ю.* Интегрированная система безопасности университетского комплекса – особенности построения и эксплуатация // Вопросы защиты информации. 2010. № 3. С. 36–41.

3. *Рогозин Е. А., Лущинский В. А.* Математическая модель угроз нарушения целостности информации в автоматизированных системах специального назначения // Информация и безопасность. 2010. № 2. С. 283–284.

4. *Рытов М. Ю., Рудановский М. В.* Управление безопасностью информационных технологий на основе методов когнитивного моделирования // Информация и безопасность. 2010. № 4. С. 579–582.

5. *Волкова С. Н., Дерябин А. С.* Функциональное моделирование как инструмент исследования механизмов защиты информации // Информация и безопасность. 2010. № 2. С. 303–304.

6. *Крупенин А. В.* Анализ статистики нагрузки на информационную систему органа обработки документальной информации ОВД субъекта РФ и разработка объективной методики определения его оргштатной структуры // Вопросы защиты информации. 2009. № 1. С. 26–30.

УДК 65.012.810(075.8)

С.В. Усов

Омский государственный университет им. Ф.М. Достоевского

ОБ ОТНОШЕНИИ МЕЖДУ СВОБОДНОЙ И ИЕРАРХИЧЕСКОЙ ОБЪЕКТНО-ОРИЕНТИРОВАННЫМИ МОДЕЛЯМИ БЕЗОПАСНОСТИ С ДИСКРЕЦИОННЫМ РАЗДЕЛЕНИЕМ ПРАВ ДОСТУПА

Объектно-ориентированная модель системы безопасности была введена в качестве альтернативы широко распространенному в моделировании систем безопасности субъектно-объектному [1] подходу. Дело в том, что в связи с широким распространением использования объектно-ориентированных средств при проектировании компьютерных систем традиционный субъектно-объектный подход обнажил ряд недостатков, в числе которых – невозможность адекватно описать субъектно-объектными средствами подсистему безопасности объектно-ориентированной системы.

Объектно-ориентированная дискреционная модель разделения доступа OOHRU является развитием классической модели Харрисона-Руззо-Ульмана [2], предложенной авторами в 70-х годах для анализа безопасности субъектно-объектных систем.

Наряду с дискреционными политиками безопасности часто рассматривают и мандатные политики [1], отличительной чертой которых является заданный на множестве объектов частичный порядок по уровню доступа. Кроме того, подсистемы безопасности многих популярных объектно-ориентированных компьютерных систем (СУБД Oracle 8i+[3] и др.) устроены с учетом иерархии на множестве объектов этих систем. Поэтому в дополнение к свободным OOHRU были также введены и изучены объектно-ориентированные модели с иерархией классов по наборам прав доступа.

Таким образом, возникает необходимость установить взаимосвязь между иерархической и свободной моделями OOHRU, выявить преимущества иерархических моделей, а также выделить свойства,

которые возможно сохранить при переходе от иерархических моделей к свободным.

В модели OOHRU [4] компьютерная система рассматривается в виде множества объектов *O*, представляющихся наборами открытых полей и скрытых полей, а также методов обработки полей. Каждый из объектов принадлежит какому-то классу *k* из множества всех классов системы *K*, причем объекты одного класса обладают одинаковым набором полей и методов. Для построения системы дискреционного разделения доступа каждый объект системы дополняется скрытым полем *M*, или матрицей доступа – таблицей, содержащей информацию о всех разрешенных видах доступа к полям и методам данного объекта. В частности, ячейка *o'.M*[*o*, *f*] матрицы доступа объекта *o*'содержит все права доступа, которыми обладает объект *o* на поле *f* объекта *o*'.

Определены элементарные операторы, преобразующие матрицу доступов:

1) *Create*(*o, k*) – создает объект *o* класса *k*.

2) *Destroy*(*o*) – уничтожает объект *o*.

3) *Enter*(*r, o, o'.f*) – вносит право доступа *r* в *o'.M*[*o, f*].

4) *Delete*(*r, o, o'.f*) – удаляет право r доступа из *o'.M*[*o, f*].

5) *Grant*(*o, o'.s*) – разрешает вызов объекту *o* метода *s* объекта *o'*.

6) *Deprive*(*o, o'.s*) – запрещает вызов объекту *o* метода *s* объекта *o'*.

Состояния компьютерной системы в модели HRU изменяются под воздействием запросов на модификацию матрицы доступа в виде команд следующего формата:

if <конъюнкция логических выражений вида $r \in o'.M[o, f]$ или *o'.M*[*o, s*]=1>

then **<**последовательность элементарных операторов>.

На место аргументов команды подставляются объекты либо классы, участвующие в качестве переменных в условной части либо в элементарных операторах.

HRU-модель системы безопасности называется монооперационной, если каждая команда этой модели содержит только один элементарный оператор.

HRU-модель системы безопасности называется моноусловной, если каждая команда этой модели содержит только одно условие.

HRU-модель системы безопасности объектно-ориентированной компьютерной системы называется монотонной, если команды этой модели не содержат операторов *Delete*, *Deprive* и *Destroy*.

HRU-модель системы безопасности объектно-ориентированной компьютерной системы называется однородной, если все объекты одного класса этой модели обладают одним и тем же набором прав доступа.

Было показано, что модель OOHRU допускает проверку возможности утечки права доступа в следующих случаях:

А) монооперационный

Б) монотонный моноусловный

В) однородный

Результаты для однородного случая также переносятся на более широкий класс иерархических OOHRU.

HRU-модель системы безопасности называется иерархической, если на множестве объектов O задан частичный порядок-иерархия, и в любой момент работы системы для любых двух объектов $o, o' \in O$ таких, что o' стоит по иерархии выше o, для любого поля или метода x, общего для объектов o и o', и для любого поля или метода y объекта o'' верно следующее: o обладает на поле или метод x объекта o'' меньшим по включению набором прав, чем o'; и объект o'' обладает меньшим по включению набором прав на метод x объекта o', чем на соответствующий метод объекта o. В объектно-ориентированных системах классы (и, соответственно, объекты этих классов) уже связаны частичным отношением наследственности, поэтому в данном случае иерархия строится естественным образом.

Стоит выделить два способа задания иерархии на объектах одного и того же класса. Случай, когда объекты одного класса стоят на одной ступени в иерархии, соответствует однородной иерархической модели OOHRU. В этой ситуации достаточно ограничиться рассмотрением иерархии на классах. Случай, когда объекты одного класса считаются несравнимыми, более общий. Элементарные операторы в иерархическом случае имеют более сложный вид.

По итогам сравнения свободной и иерархической моделей OOHRU были получены следующие результаты:

1. Каждая компьютерная система безопасности, представленная свободной моделью OOHRU, может быть представлена иерархической моделью OOHRU.

2. При этом свободная модель OOHRU компьютерной системы допускает проверку безопасности, если проверку безопасности допускает иерархическая модель той же системы, причем результаты проверки будут совпадать.

3. В частности, проверку на безопасность допускает однородная иерархическая модель.

4. Класс систем, представимых моделями с иерархией, оказывается более широким по сравнению с классом систем, представимых свободными моделями. То есть существуют системы безопасности, представимые иерархическими моделями, но не представимые свободными.

5. Однако для отдельных классов иерархических моделей возможно построение свободных моделей, представляющих те же системы безопасности. Для определения характера соответствия между такими моделями было введено понятие *R*-эквивалентности.

6. Для каждой монотонной иерархической модели OOHRU компьютерной системы существует *R*-эквивалентная ей монотонная расширенная свободная модель OOHRU.

7. Для каждой однородной иерархической модели OOHRU компьютерной системы существует *R*-эквивалентная ей однородная свободная модель OOHRU.

8. Однако, свободная модель, *R*-эквивалентная моноусловной модели с иерархией, уже не будет моноусловной в общем случае. То же верно и для свойства монооперационности. Таким образом, нельзя утверждать, что монооперационные и монотонные моноусловные иерархические модели допускают проверку на возможность утечки права доступа.

ЛИТЕРАТУРА

1. *Гайдамакин Н. А.* Разграничение доступа к информации в компьютерных системах. Екатеринбург : Изд-во Уральского университета, 2003. 328 с.

2. *Harrison M. A., Ruzzo W. L., Ulman J. D.* Protection in Operating Systems // Communications of the ACM, 1975. P. 14–25.

3. *Кайт Т.* Oracle для профессионалов: архитектура, методики программирования и особенности версий 9i, 10g и 11g. М. : Вильямс, 2011. 848 с.

4. *Усов С. В., Белим С. В., Белим С. Ю.* Дискреционное разграничение доступа для объектно-ориентированной модели компьютерной системы // Проблемы обработки и защиты информации. Книга 1. Модели политик безопасности компьютерных систем: коллективная монография / под общ. ред. С. В. Белима. Омск : Полиграфический центр КАН, 2010.

УДК 519.688

И.Б. Ларионов
Омский государственный университет им. Ф.М. Достоевского

ЗАПОЛНЕНИЕ ПРОПУСКОВ В ГРАФИЧЕСКИХ ОБЪЕКТАХ С ИСПОЛЬЗОВАНИЕМ АЛГОРИТМА ИНТЕРПОЛЯЦИИ СПЛАЙНАМИ

Надежность современных накопителей достаточно высока. Но всегда есть вероятность того, что какие-либо файлы будут испорчены. Так же современные пиринговые сети используют достаточно эффективные алгоритмы хеширования. Однако, иногда происходит изменение данных, которые не заметны для внутренних алгоритмов контроля целостности носителей или для алгоритмов хеширования. Подобные изменения иногда приводят к достаточно сильному изменению данных (на примере мультимедийных данных) в связи с тем, что современные алгоритмы сжатия с потерями неустойчивы к повреждениям.

Интерполяция бикубическими сплайнами зарекомендовала себя при заполнении получаемых пропусков при изменении размеров изображений [1], так называемый ресемплинг.

Рассмотрим изображение как набор точек, имеющих 3 компоненты: R, G и B, соответственно, для красной, зеленой и синей составляющих цвета данного пикселя. Таким образом алгоритм интерполяции бикубическими сплайнами сводится к решению следующих трех систем линейных алгебраических уравнений [1; 2]:

$$p_r(x,y) = \sum_{i=0}^{3}\sum_{j=0}^{3} a_{ij}^r x^i y^i,$$

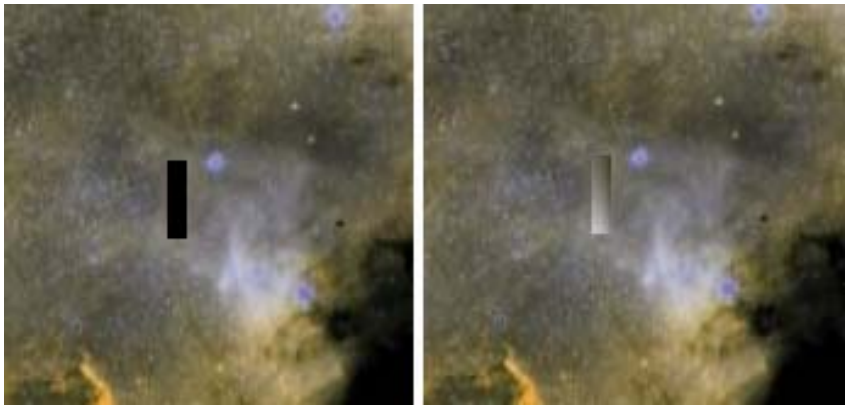
$$p_q(x,y) = \sum_{i=0}^{3}\sum_{j=0}^{3} a_{ij}^q x^i y^i,$$

$$p_b(x,y) = \sum_{i=0}^{3}\sum_{j=0}^{3} a_{ij}^b x^i y^i,$$

где p_r, p_g и p_b значения красной, зеленой и синей компоненты пикселя, а a_{ij} – искомые коэффициенты СЛАУ для соответствующих цветовых компонент.

Таким образом, при заполнении пропусков в графических объектах для решения СЛАУ следует использовать координаты «неповрежденных» пикселей и значения их цветовых компонент.

Для наглядности приведем изображение, восстановленные с помощью бикубических сплайнов (рис.) [3].



Пример восстановления

ЛИТЕРАТУРА

1. *Ларионов И. Б.* Алгоритмы предварительной обработки графических объектов со статическими пропусками в системах технического зрения : автореф.

2. *Ларионов И. Б.* Алгоритм автоматизированного восстановления поврежденных графических файлов // Вестник Омского университета. Омск : Изд-во Ом. гос. ун-та. 2011. № 2. С. 176–177.

3. *Ларионов И. Б.* Кластеризация матриц с пропусками как метод восстановления графической информации // Математические структуры и моделирования. Омск : УниПак, 2009. Вып. 20. С. 97–106.

УДК 004.053

Е.А. Тюменцев

Омский государственный университет им. Ф.М. Достоевского

БИБЛИОТЕКА HWdTech.DS С ТОЧКИ ЗРЕНИЯ МОДЕЛИ ЗРЕЛОСТИ

В последнее время наблюдается повышенный интерес к разработке параллельных программ. Это объясняется тем, что рост производительности современных процессоров обеспечивается за счет введения различных механизмов параллельного исполнения команд.

На протяжении десятилетий наиболее распространенной на практике моделью параллельных вычислений было взаимодействие на основе разделяемой памяти. В рамках данной модели выполнение параллельной программы трактуется как последовательность смен глобальных состояний программы. Смена одного состояния на следующее называется шагом. Шаг – это выполнение одного неделимого (атомарного) оператора. Распространенность такого подхода объясняется тем, что он естественным образом обобщает первые модели вычислений для одного потока команд, например, такие как, машина Тьюринга или машина Поста.

Однако, в отличие от выполнения программы в один поток, результат выполнения параллельной программы может быть неоднозначным. Неоднозначность появилась из-за того, что атомарные операторы из разных потоков могут выполняться в разной последовательности. Поэтому, основная задача, которую решает программист - это синхронизация блоков кода из разных потоков для обеспечения правильности результатов выполнения. Для этого были разработаны различные примитивы синхронизации: мьютексы, семафоры и т. д.

Взаимодействие на основе разделяемой памяти обладает двумя серьезными недостатками:

1) сложность программирования [1] и, как следствие, высокие требования к квалификации программиста.

2) требование последовательности смены состояний делает ее малопригодной для применения в распределенных системах.

Альтернативой данной модели вычислений является взаимодействие на основе обмена сообщениями. Считается, что программирование на основе сообщений проще, чем программирование на основе взаимодействия через разделяемую память.

Создано несколько теорий для описания свойств систем на основе обмена сообщениями.

Мы будем использовать модель акторов [2], созданную в 1973 году. Модель акторов допускает выполнение нескольких операций в один момент времени. Поэтому она лишена отмеченных выше недостатков взаимодействия на основе разделяемой памяти. Существует довольно много промышленных реализаций модели акторов, например, Erlang, Scala.

Свойства параллельных программ на основе разделяемой памяти были хорошо изучены и обоснованы с помощью аппарата математической логики. Были предприняты попытки разработки аппарата математической логики на модель акторов. В 1988 году Хьюит и Ага опубликовали работу [3], в которой показали, что построенная логика не является дедуктивной в следующем смысле: вычислительные шаги не следуют дедуктивно из предыдущих шагов.

На практике, полученный Хьюитом и Ага результат, означает потенциальные проблемы с обнаружением ошибок и отладкой таких систем, особенно с ростом сложности программного обеспечения.

Следовательно, возникает потребность в новых подходах и инструментах к разработке, отладке и тестированию систем, основанных на обмене сообщениями.

Каждый подход или инструмент призван решать конкретный круг вопросов. Определим множество задач, которое должна решать библиотека для разработки параллельных программ. Для этого воспользуемся концепцией зрелости [4]. Под зрелостью будем понимать оценку опыта решения задач с точки зрения возможности получения воспроизводимого успешного результата.

Концепция зрелости определяет пять уровней:

1. Начальный уровень. Нет осознанного общего подхода к решению задач. Качество результата зависит от навыков конкретных исполнителей. Подходы к решению выбираются случайно. Успех в одной задаче не гарантирует результата в аналогичной другой.

2. Базовый уровень. Накопленный опыт приводит к стандартизации типовых решений. Появляется общая архитектура.

3. Уровень взаимодействия. Общая архитектура приводит к увеличению общего числа компонентов в системе. Между компонентами появляются связи и зависимости.

4. Измеримый уровень. С ростом числа связей между компонентами неизбежно встает вопрос об эффективности выбранных способов взаимодействий. Так возникает потребность в измерении полученных результатов и накоплении базы измерений.

5. Управляемый уровень. База измерений позволяет принимать решения и оценивать последствия принятых решений. Выбор становится более обоснованным.

Библиотека HWdTech.DS – библиотека для разработки параллельных приложений, основанных на обмене сообщениями, которая реализует вычисления в модели авторов. При ее разработке была поставлена задача: библиотека должна соответствовать измеримому уровню зрелости. Это значит, что HWdTech.DS должна обеспечивать:

1) общую архитектуру для реализации акторов и сообщений,

2) механизмы взаимодействия между акторами,

3) инструменты измерения для анализа эффективности работы приложения, построенного на основе данной библиотеки.

ЛИТЕРАТУРА

1. *Edward A. Lee*, The Problem With Threads. URL: http://www.eecs.berkeley.edu/Pubs/TechRpts/2006/EECS-2006-1.pdf (дата обращения: 01.09.2013).

2. *Carl Hewitt , Peter Bishop , Richard Steiger.* A universal modular ACTOR formalism for artificial intelligence. / IJCAI'73 Proceedings of the 3rd international joint conference on Artificial intelligence. P. 235–-245.

3. *Carl Hewitt, Gul Agha.* Guarded horn clause languages: are they deductive and logical? / FGCS, 1988. P. 650–-657.

4. *Керцнер Гарольд,* Стратегическое планирование для управления проектами с использованием модели зрелости : пер. с англ. М. : Компания АйТи ; ДМК Пресс, 2003. 320 с., ил.

УДК 004.75

Д.М. Бречка, В.С. Виноградов
Омский государственный университет им. Ф.М. Достоевского

КОНТЕЙНЕР СЕРВЕРНОГО JAVA-КОДА С ПОДДЕРЖКОЙ ПОСТОЯННОГО СОЕДИНЕНИЯ

Для реализации логики определенной группы клиент-серверных систем часто необходимо использовать постоянное соединение с сервером приложения. При этом использование технологии контейнера серверного кода часто является предпочтительной для разработки серверной части с точки зрения ускорения процесса разработки и достижения определенного уровня абстракции кода логики приложения. Возникает необходимость в разработке библиотеки для подобных целей, поскольку использование готовых универсальных решений не всегда является возможным.

Архитектура подобного решения должна быть максимально универсальной для разработки приложений под платформу web. Она должна предусматривать слой работы непосредственно с сокетами, которые должны быть абстрагированы от разработчика логики приложения. В архитектуре должен иметь место сам контейнер, который будет заниматься управлением объектами, содержащими серверный код.

Важными для нашего решения являются обработчики очередей и задач. Слой работы с сокетами возвращает данные, пришедшие из сети, в определенном формате. На основе этих данных формируется задача, которая должна быть обработана контейнером. Обработчик очередей помещает ее в план на обработку. Когда очередь доходит до определенной задачи, запускается обработчик задач, который производит анализ требуемых действий и выполняет запрос, а также формирует ответ, который отсылается клиентской стороне с помощью слоя сокетов.

Вся логика приложения реализуется внутри специальных объектов. Такие объекты содержат в себе особый метод, при вызове которого происходит запуск кода с логикой конкретного объекта. Аргументами этого метода являются буфер входных данных, который содержит параметры запроса, пришедшего из сети, буфер выходных

данных, представляющий из себя ответ на данный запрос, и параметр сессии соединения, который хранит определенные сеансовые параметры в течение того времени, пока соединение открыто, и позволяет получить доступ к остальным активным соединениям.

Тестирование прототипа подобной системы и сравнение ее с имеющими аналогами показало, что подобное решение является достаточно удобным для реализации определенного класса web-приложений, в котором наличие постоянного соединения является критичным. Также был выделен функционал, который мог бы быть полезен в подобной системе: система шифрования данных, работу с аннотациями и отложенную инициализацию объектов, содержащих логику, универсальный формат передачи данных.

Научное издание

МАТЕМАТИЧЕСКОЕ И КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ

Сборник материалов научной конференции

(Омск, 18 октября 2013 г.)

Издается в авторской редакции.
Макет подготовлен при участии Издательства ОмГУ

Подписано в печать 10.10.2013. Формат бумаги 60х84 1/16.
Печ. л. 6,5. Усл. печ. л. 6,3. Уч. изд. л. 6,5. Тираж 100 экз. Заказ 252.

Издательство Омского государственного университета
644077, Омск, пр. Мира, 55а
Отпечатано на полиграфической базе ОмГУ
644077, Омск, пр. Мира, 55а