Информатика и её применения

Том 2 Выпуск 3 Год 2008

СОДЕРЖАНИЕ

Структурная декомпозиция матричных систем
А. С. Оленин **2**

Concurrent design and verification of digital hardware
S. Baranov, S. Frenkel, V. Sinelnikov, and V. Zakharov **7**

Функция стоимости ресурсов в экономической модели управления ГРИД
Я. М. Агаларов **26**

Многолинейная система массового обслуживания с конечным накопителем, блокировкой полумарковского потока заявок и выбиванием заявок из накопителя
В. В. Чаплыгин **34**

Сервисно-ориентированный подход к разработке мультибиометрических технологий
О. С. Ушмаев **41**

Задачи представления личностных и коллективных концептов в цифровой среде
И. М. Зацман, В. В. Косарик, О. А. Курчавова **54**

К 25-летию Института проблем информатики РАН
И. А. Соколов, В. Н. Захаров **70**

Abstracts **77**
Об авторах **80**
About Authors **80**

Технический редактор *Л. Кокушкина*
Выпускающий редактор *Т. Торжкова*
Художественный редактор *М. Седакова*
Сдано в набор 02.07.08. Подписано в печать 11.09.08. Формат 60 x 84 / 8
Бумага офсетная. Печать офсетная. Усл.-печ. л. 10,0. Уч.-изд. л. 10,2. Тираж 200 экз.
Заказ №1388
Издательство «ТОРУС ПРЕСС», Москва 121614, ул. Крылатская, 29-1-43
torus@torus-press.ru; http://www.torus-press.ru
Отпечатано в ППП «Типография «Наука» с готовых диапозитивов
Москва 121099, Шубинский пер., д. 6.

ИНФОРМАТИКА И ЕЁ ПРИМЕНЕНИЯ, 2008. Т. 2. Вып. 3. С. 2–6

СТРУКТУРНАЯ ДЕКОМПОЗИЦИЯ МАТРИЧНЫХ СИСТЕМ

А. С. Оленин[1]

Аннотация: Рассматривается один из возможных подходов к распараллеливанию матричных систем путем их структурной декомпозиции на совокупность подсистем, независимых на определенном этапе вычислений. Сформулирован конструктивный алгоритм декомпозиционного метода и дана оценка его вычислительных затрат.

Ключевые слова: матричная система, ленточная матрица; полная матрица; треугольная матрица; блочная трехдиагональная матрица; декомпозиция; вектор разбиения, отрезок разбиения; факторизация; LU-разложение; распараллеливание

Развитие современных информационных технологий вряд ли возможно без разработки рациональных подходов к структурной организации вычислительных задач и алгоритмов. От этого напрямую зависит производительность и эффективность вычислительных процессов параллельных средств обработки. Если задачи на уровне алгоритмов не обладают достаточной степенью параллелизма, то не следует ожидать высокой производительности при их решении на параллельных вычислительных системах. То же самое относится и к векторной обработке.

Нередко эффективные, обоснованные теоретически по точности, устойчивости и тщательно отработанные на практике вычислительные алгоритмы [1, 2] в своей обычной формулировке могут при некоторых условиях терять свойства параллельности и становиться преимущественно последовательными. Именно в таких ситуациях важно предпринимать шаги по структурной модификации алгоритмов, с тем чтобы повысить их степень параллелизма. Иногда эта цель достигается за счет некоторого увеличения общего числа операций, но при этом наращивание параллельных свойств позволяет создать весьма значительный потенциал производительности.

Следует отметить, что проблема структурной модификации исходного алгоритма с целью придания ему более высокой степени параллелизма является весьма непростой и, по существу, равносильна созданию нового алгоритма, в котором сохранены черты старого в специфических условиях. Подходы к решению этой проблемы разнообразны [3] и сильно зависят от характера вычислительных задач. При этом по большей части осуществляется выявление и реализация параллельных свойств алгоритмов в рамках их исходной структуры. Далее будет изложен один из возможных подходов, связанный со структурной декомпозицией матричных систем, и обсуждены его особенности.

Будем рассматривать линейное уравнение с матрицей ленточного типа, где матрица имеет размер $(N \times N)$ и ширину ленты $2m+1$, где m — число диагоналей над (под) главной диагональю

$$\sum_{j=-m}^{m} a_{i,i+j} x_{i+j} = f_i\,, \quad i = 1, 2, \ldots, N\,, \tag{1}$$

$$x_{1-m} = 0\,, \quad x_{2-m} = 0\,, \ \ldots\,, \ x_0 = 0\,,$$
$$x_{N+1} = 0\,, \quad x_{N+2} = 0\,, \ \ldots\,, \ x_{N+m} = 0\,.$$

В том случае, когда значение m по величине сравнимо с N, матрица близка к полной, и для ее решения широко используются, например, методы, основанные на процедуре факторизации. Однако анализ показывает, что типичный процесс вычислений вдоль главной диагонали в подобных методах является последовательным, а параллелизм обусловлен наличием побочных диагоналей, причем параллеллизм тем выше, чем больше задействованных диагоналей, т. е. чем больше ширина ленты матрицы. Поэтому при малой ширине ленты процесс факторизации становится преимущественно последовательным и малопригодным для счета на параллельных средствах. Между тем матричные системы с малой шириной ленты часто применяются в реальных вычислениях, и их параллельная реализация весьма актуальна.

Предлагаемый подход заключается в декомпозиции (разбиении) исходной системы на совокупность подсистем, независимых друг от друга на определенном этапе вычислений. Связь этих под-

[1]Институт проблем информатики Российской академии наук, aolenin@yandex.ru

систем осуществляется путем формулирования отдельной задачи, которая позволяет выполнить расчет подсистем в параллельном режиме, а затем вычислить окончательный результат для исходной системы.

Итак, разобьем всю последовательность неизвестных на $n+1$ отрезков с помощью n векторов $X_k = \{x_i | i = i_k, i_k+1, \ldots, i_k+m-1\}, k = 1, 2, \ldots, n$, $X_0 = 0$, $X_{n+1} = 0$. Отрезок с номером k ограничен слева вектором X_{k-1}, справа — вектором X_k.

Решение системы (1) будем искать в виде

$$x_i = y_i + \sum_{k=1}^{n} \sum_{p=1}^{m} v_{ip}^{(k)} x_{i_k+p-1}\,, \quad i = 1, 2, \ldots, N\,. \quad (2)$$

Запишем это выражение в векторно-матричной форме для произвольного вектора разбиения X_t:

$$X_t = Y_t + \sum_{k=1}^{n} V_t^{(k)} X_k\,. \quad (3)$$

Здесь векторы имеют размер m, матрицы — размер $(m \times m)$:

$$X_t = \begin{pmatrix} x_{i_t} \\ x_{i_t+1} \\ \cdots \\ x_{i_t+m-1} \end{pmatrix}; \quad Y_t = \begin{pmatrix} y_{i_t} \\ y_{i_t+1} \\ \cdots \\ y_{i_t+m-1} \end{pmatrix};$$

$$V_t^{(k)} = \begin{pmatrix} v_{i_t,1}^{(k)} & v_{i_t,2}^{(k)} & \cdots & v_{i_t,m}^{(k)} \\ v_{i_t+1,1}^{(k)} & v_{i_t+1,2}^{(k)} & \cdots & v_{i_t+1,m}^{(k)} \\ \cdots & \cdots & \cdots & \cdots \\ v_{i_t+m-1,1}^{(k)} & v_{i_t+m-1,2}^{(k)} & \cdots & v_{i_t+m-1,m}^{(k)} \end{pmatrix}.$$

Для того чтобы равенство (3) тождественно удовлетворялось на векторе X_k, необходимо выполнение условий:

$$Y_k = 0\,; \quad V_t^{(k)} = \begin{cases} E\,, & t = k\,; \\ 0\,, & t \neq k\,, \end{cases} \quad (4)$$

где E — единичная матрица размера $(m \times m)$.

Подставляя выражение (2) в систему (1), получаем серию задач:

$$\sum_{j=-m}^{m} a_{i,i+j} y_{i+j} = f_i\,;$$

$$\sum_{j=-m}^{m} a_{i,i+j} v_{i+j,p}^{(k)} = 0\,, \quad p = 1, 2, \ldots, m\,, \quad (5)$$

$$i = 1, 2, \ldots, N\,,\ k = 1, 2, \ldots, n\,.$$

Далее примем во внимание, что на границах отрезка k, т. е. на векторах X_{k-1} и X_k, согласно условиям (4), имеют место граничные равенства:

$$Y_{k-1} = 0\,; \quad Y_k = 0\,;$$

$$V_{k-1}^{(k-1)} = E\,; \quad V_k^{(k-1)} = 0\,;$$

$$V_{k-1}^{(k)} = 0\,; \quad V_k^{(k)} = E\,.$$

Из этих равенств следует, что на отрезке k только две функции ($v^{(k-1)}$ и $v^{(k)}$) из систем (5) будут иметь ненулевые решения, поскольку только они имеют ненулевые значения на границах (при нулевой правой части систем). Для удобства обозначим функцию $v^{(k-1)}$ на отрезке k через $u^{(k)}$. Придем к следующей совокупности систем на отрезке k:

$$\sum_{j=-m}^{m} a_{i,i+j}^{(k)} y_{i+j}^{(k)} = f_i^{(k)}\,;\ Y_{k-1} = 0\,;\ Y_k = 0\,; \quad (6)$$

$$\sum_{j=-m}^{m} a_{i,i+j}^{(k)} u_{i+j,p}^{(k)} = 0\,; \quad U_{k-1}^{(k)} = E\,; \quad U_k^{(k)} = 0\,, \quad p = 1, 2, \ldots, m\,; \quad (7)$$

$$\sum_{j=-m}^{m} a_{i,i+j}^{(k)} v_{i+j,p}^{(k)} = 0\,; \quad V_{k-1}^{(k)} = 0\,; \quad V_k^{(k)} = E\,, \quad p = 1, 2, \ldots, m\,. \quad (8)$$

Вследствие условий на границах неизвестных значений системы (1), $u^{(1)} = 0$ на первом отрезке, $v^{(n)} = 0$ — на отрезке $n+1$.

С учетом задач (6)–(8) выражение (2) на отрезке k принимает вид

$$x_i^{(k)} = y_i^{(k)} + \sum_{p=1}^{m} u_{ip}^{(k)} x_{i_{k-1}+p-1} + \sum_{p=1}^{m} v_{ip}^{(k)} x_{i_k+p-1}\,. \quad (9)$$

Теперь запишем систему (1) на векторе X_k, учитывая, что слева от него расположен отрезок k, а справа — отрезок $k+1$:

$$A^{(k)} X^{(k)} + B_k X_k + C^{(k+1)} X^{(k+1)} = F_k\,. \quad (10)$$

Здесь верхние индексы относятся к отрезкам, а нижние — к векторам разбиения,

$$A^{(k)} = \begin{pmatrix} a_{i_k,i_k-m} & a_{i_k,i_k-m+1} & \dots & a_{i_k,i_k-1} \\ 0 & a_{i_k+1,i_k-m+1} & \dots & a_{i_k+1,i_k-1} \\ \dots & \dots & \ddots & \dots \\ 0 & 0 & \dots & a_{i_k+m-1,i_k-1} \end{pmatrix};$$

$$B_k = \begin{pmatrix} a_{i_k,i_k} & a_{i_k,i_k+1} & \dots & a_{i_k,i_k+m-1} \\ a_{i_k+1,i_k} & a_{i_k+1,i_k+1} & \dots & a_{i_k+1,i_k+m-1} \\ \dots & \dots & \ddots & \dots \\ a_{i_k+m-1,i_k} & a_{i_k+m-1,i_k+1} & \dots & a_{i_k+m-1,i_k+m-1} \end{pmatrix};$$

$$C^{(k+1)} = \begin{pmatrix} a_{i_k,i_k+m} & 0 & \dots & 0 \\ a_{i_k+1,i_k+m} & a_{i_k+1,i_k+m+1} & \dots & 0 \\ \dots & \dots & \ddots & \dots \\ a_{i_k+m-1,i_k+m} & a_{i_k+m-1,i_k+m+1} & \dots & a_{i_k+m-1,i_k+2m-1} \end{pmatrix};$$

$$X^{(k)} = \{x_i | i = i_{k-m}, \dots, i_{k-1}\};$$
$$X_k = \{x_i | i = i_k, \dots, i_{k+m-1}\};$$
$$X^{(k+1)} = \{x_i | i = i_{k+m}, \dots, i_{k+2m-1}\};$$
$$F_k = \{f_i | i = i_k, \dots, i_{k+m-1}\},$$

где $A^{(k)}$ — верхняя треугольная матрица, связанная с отрезком k; B_k — полная матрица, связанная с вектором k; $C^{(k+1)}$ — нижняя треугольная матрица, связанная с отрезком $k+1$.

Выразим векторы $X^{(k)}$, $X^{(k+1)}$ в системе (10) через векторы разбиения, используя формулу (9):

$$X^{(k)} = Y^{(k)} + U^{(k)} X_{k-1} + V^{(k)} X_k;$$
$$X^{(k+1)} = Y^{(k+1)} + U^{(k+1)} X_k + V^{(k+1)} X_{k+1}.$$

Вид введенных здесь векторов и матриц ясен из выражения (9). После подстановки этих соотношений в систему (10), получим

$$\overline{A}_k X_{k-1} + \overline{B}_k X_k + \overline{C} X_{k+1} = \overline{F}_k, \quad k = 1, 2, \dots, n, \quad (11)$$

где

$$\overline{A}_k = A^{(k)} U^{(k)};$$
$$\overline{B}_k = A^{(k)} V^{(k)} + B_k + C^{(k+1)} U^{(k+1)};$$
$$\overline{C}_k = C^{(k+1)} V^{(k+1)};$$
$$\overline{F}_k = F_k - A^{(k)} Y^{(k)} - C^{(k+1)} Y^{(k+1)}.$$

Система (11) имеет блочную трехдиагональную матрицу и является связующей для подсистем (6)–(8).

Итак, для решения задачи (1) в соответствии с изложенной схемой декомпозиции необходимо выполнить следующие действия:

1. Решить задачи (6)–(8) на отрезках разбиения.
2. Найти решение связующей задачи (11) на векторах разбиения.
3. Вычислить неизвестные значения системы (1) по формуле (9).

Далее займемся оценкой вычислительных затрат декомпозиционной схемы. Вначале вспомним, что, непосредственно применяя для решения заданной системы (1) такой широко распространенный метод, как LU-разложение, требуется выполнить порядка m^2N операций. Следует выяснить, приводит ли декомпозиционная схема к увеличению объема вычислений и если приводит, то на какую величину по сравнению с LU-разложением.

Будем полагать, что отрезки разбиения примерно равны и имеет место обычное рабочее соотношение параметров: $m \ll n \ll N$, где $n \gg 1$. Для решения линейных систем на отрезках используем алгоритм LU-разложения. Тогда для системы (6) по порядку величины имеем n задач по $m^2(N/n)$ операций каждая, т. е. всего m^2N операций. Системы (7) и (8), если в них граничные величины перенесены в правые части, дают в совокупности порядка m^3N операций. Связующая система (11) может быть решена за m^3n операций. И, наконец, формула (9) содержит примерно mN операций. Очевидно, что в приведенной оценке определяющей является величина m^3N, которая при достаточно больших значениях ширины ленты может существенно превосходить оценку m^2N, имеющую место при решении исходной системы (1) без применения декомпозиции. И здесь возникает правомерный вопрос: не является ли дополнительная вычислительная нагрузка, связанная с декомпозицией, слишком чрезмерной? Однако эту нагрузку можно значительно снизить, если обратить внимание на то, что матрицы систем (7) и (8) не меняются по индексу p, поскольку граничные величины, как

указывалось выше, перенесены в правые части. Это обстоятельство позволяет использовать специфику LU-разложения, которая состоит в том, что этап разложения на треугольные множители для каждой из систем (7) и (8) может быть выполнен однократно. А поскольку этот этап является доминирующим по затратам, то оценку числа операций для него благодаря указанному свойству можно снизить с m^3N до m^2N. При этом на этапе решения треугольных систем алгоритма LU-разложения затраты остаются на уровне m^2N операций. Добавляя сюда число операций m^2N для системы (6), получаем оценку вычислительных затрат декомпозиционного метода порядка Cm^2N операций, где C — небольшая по величине константа, точное определение которой требует более детального рассмотрения. Принципиально важно то, что оценку, совпадающую с точностью до константы (т. е. независимую от ширины ленты) с оценкой исходного алгоритма, удалось получить, используя именно специфику LU-разложения. В случае алгоритмов, в которых отсутствует подобная специфика, затраты на декомпозицию значительно возрастают. Указанное обстоятельство наглядно демонстрирует важную для приложений ситуацию, когда структурная декомпозиция задач на подзадачи требует внимательного отношения к алгоритмам реализации.

Итак, применив декомпозиционный метод, можно приблизительно в C раз увеличить объем вычислений, но одновременно получить существенный потенциал распараллеливания. Действительно, самая затратная часть метода, связанная с системами (6)–(8), распадается на $n+1$ независимых подзадач и может выполняться как в параллельном, так и в векторном режиме. Решение связующей задачи (11) сводится к векторно-матричным операциям, которые естественно распараллеливаются и векторизуются. Формула (9) вносит менее значительный вклад в общий объем вычислений и также обладает параллельными свойствами.

Отметим, что характерная степень параллелизма схемы декомпозиции определяется числом полученных независимых подзадач, т. е. параметром n. Следовательно, неравенство $n > C$ устанавливает приближенный критерий, показывающий, когда эффект по времени вычислений за счет параллелизма схемы декомпозиции может стать выше, чем потери из-за большего числа операций. И этот эффект будет усиливаться с ростом n.

Сделаем несколько замечаний о качестве вычислений по декомпозиционной схеме. Не ухудшится ли оно по сравнению с непосредственным счетом? Качество решения линейной системы зависит, как известно, от свойств ее матрицы. Если для матрицы заданной системы (1) выполнены локальные условия устойчивости (например, диагональное преобладание), то эти условия справедливы и для подматриц, которые участвуют в подзадачах (6)–(8). Поэтому качество вычислений в подзадачах остается на уровне принятого алгоритма решения. Что касается связующей системы (11), то условия ее устойчивости также являются производными от свойств исходной матрицы и не оказывают сколько-нибудь существенного самостоятельного влияния на качество вычислительного процесса. И все же, приводя теоретические соображения, следует иметь в виду, что качество вычислительных процессов, особенно в сложных задачах, наиболее надежно обеспечивается с помощью практических вычислений.

Приведем формулировку метода декомпозиции для распространенного в реальных задачах случая $m = 1$, т. е. для системы с трехдиагональной матрицей. Такая система обычно записывается следующим образом:

$$a_i x_{i-1} + b_i x_i + c_i x_{i+1} = f_i\,, \quad i = 1, 2, \ldots, N\,;$$
$$x_0 = 0\,; \quad x_{N+1} = 0\,.$$

В этом случае вид структурной декомпозиции из векторно-матричной формы переходит в скалярную (векторы разбиения становятся скалярными величинами):

$$\begin{aligned} &a_i^{(k)} y_{i-1}^{(k)} + b_i^{(k)} y_i^{(k)} + c_i^{(k)} y_{i+1}^{(k)} = f_i^{(k)}\,,\\ &y_{i_k-1}^{(k)} = 0\,, \quad y_{i_k}^{(k)} = 0\,; \end{aligned} \tag{12}$$

$$\begin{aligned} &a_i^{(k)} u_{i-1}^{(k)} + b_i^{(k)} u_i^{(k)} + c_i^{(k)} u_{i+1}^{(k)} = 0\,,\\ &u_{i_k-1}^{(k)} = 1\,, \quad u_{i_k}^{(k)} = 0\,; \end{aligned} \tag{13}$$

$$\begin{aligned} &a_i^{(k)} v_{i-1}^{(k)} + b_i^{(k)} v_i^{(k)} + c_i^{(k)} v_{i+1}^{(k)} = 0\,,\\ &v_{i_k-1}^{(k)} = 0\,, \quad v_{i_k}^{(k)} = 1\,. \end{aligned} \tag{14}$$

Связующая система приобретает вид:

$$\overline{a}_k \overline{x}_{k-1} + \overline{b}_k \overline{x}_k + \overline{c}_k \overline{x}_{k+1} = \overline{f}_k\,,\ k = 1, 2, \ldots, n\,; \tag{15}$$

$$\overline{x} = 0\,; \quad \overline{x}_{n+1} = 0\,;$$
$$\overline{x}_{k-1} = x_{i_k-1}\,; \quad \overline{x}_k = x_{i_k}\,; \quad \overline{x}_{k+1} = x_{i_k+1}\,;$$
$$\overline{a}_k = a_{i_k-1} u_{i_k-1}^{(k)}\,;$$
$$\overline{b}_k = a_{i_k-1} v_{i_k-1}^{(k)} + b_{i_k} + c_{i_k+1} u_{i_k+1}^{(k+1)}\,;$$
$$\overline{c}_k = c_{i_k+1} v_{i_k+1}^{(k+1)}\,;$$
$$\overline{f}_k = f_{i_k} - a_{i_k-1} y_{i_k-1}^{(k)} - c_{i_k+1} y_{i_k+1}^{(k+1)}\,.$$

Неизвестные значения на отрезках вычисляются по формуле:

$$x_i^{(k)} = y_i^{(k)} + u_i^{(k)} x_{i_k-1} + v_i^{(k)} x_{i_k}\,. \tag{16}$$

В схеме (12)–(16) вычислительные затраты по сравнению со случаем, когда ширина ленты велика, минимальны, и, соответственно, роль специфики применяемых алгоритмов (в рассмотренном выше смысле) также минимальна.

Отметим, что обычно алгоритмы решения таких систем имеют существенно ограниченные возможности для распараллеливания, поэтому применение декомпозиционных подходов для их распараллеливания, когда это необходимо, является особенно полезным. Например, системы с ленточными матрицами весьма часто встречаются при численном решении задач математической физики.

Возможно также приложение метода в задачах построения канонических разложений случайных функций [4] (проблема распараллеливания таких задач обсуждалась с академиком В. С. Бурцевым и к. ф-м. н. А. М. Степановым).

В заключение подчеркнем, что методы структурной декомпозиции, несмотря на разнообразие всевозможных приложений и реализаций, составляют по сути единый, весьма мощный фундаментальный инструмент распараллеливания задач и поэтому представляют значительный интерес для вычислительных аспектов информатики и информационных технологий.

Литература

1. *Марчук Г. И.* Методы вычислительной математики. — М.: Наука, 1989.
2. *Фаддеев Д. К., Фаддеева В. Н.* Вычислительные методы линейной алгебры. — СПб.: Лань, 2002.
3. *Воеводин В. В., Воеводин Вл. В.* Параллельные вычисления. — СПб.: БХВ-Петербург, 2002.
4. *Пугачев В. С., Синицын И. Н.* Теория стохастических систем. 2-е изд. — М.: Логос, 2004.

ИНФОРМАТИКА И ЕЁ ПРИМЕНЕНИЯ, 2008. Т. 2. Вып. 3. С. 7–25

CONCURRENT DESIGN AND VERIFICATION OF DIGITAL HARDWARE

S. Baranov[1], S. Frenkel[2], V. Sinelnikov[3], and V. Zakharov[4]

Abstract: The main goal of this paper is to present a new design verification methodology for complicated digital systems, designed by high-level synthesis. This methodology is based on Algorithmic State Machine (ASM) transformations (composition, minimization, extraction, etc.), special algorithms for Data Path and Control Unit (CU) design, and very fast optimizing synthesis of finite state machines (FSM) and combinational circuits with hardly any constraints on their size, that is, the number of inputs, outputs, and states. Design tools supporting this methodology allow very fast implement, check and estimate many possible design versions, to find an optimized decision of the design problem and to simplify the verification problem for digital systems. In contrast to existent semi-formal approaches to verification of industrial systems, based on combination of simulation and formal verification approaches, a formalized method based on concurrency of synthesis and verification that is providing regular efficient way to verify the system designed properties starting from its semi-formal specification up to field programmable gate array (FPGA) implementation is considered.

Keywords: digital systems design; formal verification; finite state machine

1 Introduction

Designers of complex digital systems on chips need in practical validation methods and tools to guarantee a perfect design before the process of manufacturing is started. This validation is performed mostly as a "verification," checking if a system design is correct with respect to a specification (which is understood here as an initial description of aimed design on a given representation level (e.g., FSM, register-transfer (RTL), or gate level) [1]. The goal of design verification is to ensure both the functional and timing correctness of a design implementation with respect to its specification.

How to verify the correctness of high-level synthesis becomes a key issue before mapping the synthesis results onto a silicon. In the meantime, it has been observed that verification becomes the major bottleneck, i.e., up to 70%–80% of the overall design costs are due to verification.

To provide an ideal verification, one needs a specification of the target design, giving a unique description of requirements to the target design. In very general case, this is, in fact, a necessary condition of exhaustive verification. But realization of the algorithm verification is a very difficult problem, because presently, the design activity being a hierarchical in nature, involves different professionals in the design process. For example, a system designer transforms algorithms of the target design to a behavioral RTL description, logic designer transforms the behavioral RTL description to a structural RTL description. Circuit designers generate transistor-level netlists that implement the structural RTL. Layout designers generate layouts for the transistor-level schematics.

At this stage, there are numerous verification problems. They include functional (logic) verification, timing verification, electrical verification, and physical design rule verification.

Full-automatic verification of complex processor design is a dream of all system designers. Hardware module is formally verified by stating a property on the design and then checking that the design satisfies the property. Ususally, the property expresses a condition on the hardware module that should never happen in a reachable state (or conversely, a condition that should always be true in a reachable state). Formally, the property is an *invariant* which is a boolean formula over some signals of the module. The module M satisfies the invariant I if every reachable state of the module satisfies I [2].

Unfortunately, the exponential complexity of all formal verification algorithms is well known that, it seems, excludes this dream realization, at least for large designs with very large state spaces which cannot be handled. Therefore, one of the ways to overcome the verification complexity is using of the combination of formal and

[1]Holon Institute of Technology, Holon, Israel, samary@012.net.il
[2]Institute of Informatics Problems, Moscow, Russia, slf-ipiran@mtu-net.ru
[3]Holon Institute of Technology, Holon, Israel, samary@012.net.il
[4]Institute of Informatics Problems, Moscow, Russia, VZakharov@ipiran.ru

informal verification models [3] to solve the verification problem as effectively as possible.

Speaking on informal specification, they usually take into account the checking via simulation (driven by random and handwritten inputs). However, simulation test cannot handle all possible cases and also is rather expensive and not scalable [4].

Further, from the economical point of view, it is desirable that a hardware design project should start with a modular and abstract specification. An implementation should be constructed through successive refinement of the specification while maintaining the interfaces between modules.

However, usually formal verification models are automatically compiled from RTL source code, that is, from rather late and error-prone stages of design process, when designer have to describe the logical structure of design manually.

It would be desirable to use some knowledge received during synthesis, and moreover, the verification technique should match a target system synthesis. For this goal, a designer needs tools and methodology for concurrent activity in design and verification.

The main goal of this paper is to analyze the state-of-the-art of the current design verification practice to submit, relying on this analysis, a new methodology for high-level and logic design and verification of complicated digital systems. This methodology is based on using of the ASM for specification of digital systems at the high level [5]. In fact, the ASM model description of a digital system is a flowchart with microoperations and conditions of their execution in its nodes. One of novel aspects of this approach is that the verification should match the design (synthesis) technique. In fact, we can speak about "concurrency" of the synthesis and verification activities in the framework of this approach.

This methodology is supported by software design tool Abelite which allows very fast implement, check and estimate many possible design versions, to find an optimized decision of the design problems and to simplify the verification problem for digital systems. We consider a possibility to provide semi-formal specification, harmonized with the synthesis process, providing the receiving test benches for verification by simulation, guaranteed enough coverage if all synthesis requirements are carried out. The synthesis takes into account area/delay overhead constraints.

2 Principal Notions

Let us outline some principal definitions for the hardware verification area.

2.1 Errors

As the cause of possible incorrectness of a design, there are some bugs of designers (at various stages of the design process). Let us consider, first of all, the fault model, presence of which forces us to study the verification. The causes of the faults (or bugs) can be the following:

- incorrect specifications;
- misinterpretation of specifications;
- missed cases;
- protocol nonconformance; and
- timing errors.

Note that one of a frequent cause of these errors can be some misunderstanding between designers working at various design stages.

So, we should take into account these possible errors ("bugs") in our verification studies. As it was mentioned above, we can try to understand if there are some of the bugs in the design either by simulation of the design model under some stimulus or by formal verification.

These causes can lead to various errors at the structural level of modeling, in CU and Data Path, in particular. These are, for example [6]:

- *Bus error*: a bus of one or more lines is (totally) stuck-at-0 or stuck-at-1 if all lines in the bus are stuck at logic level 0 or 1. This generalization of the standard single stuck line (SSL) model was introduced in [6] in the context of physical fault testing.
- *Module substitution error*: This refers to mistakenly replacing a module by another module with the same number of inputs and outputs. This class includes word gate substitution errors and extra/missing inversion errors.
- *Bus source error*: this error corresponds to connecting a module input to a wrong source.

More advanced fault models of VHDL[1] RTL design were considered in [7]. For example, they have considered change of the value of any constant and of any occurrence of a variable, driving it to one of the values which have an Hamming distance equal to one from the fault free configuration (e.g., given the expression $a+b+a$, then the two a may fail either together or separately). Also, the following possible errors have been considered:

- change of the value returned by a function, allowing it to return any of the faulty values with a unitary Hamming distance from the fault free value; and
- incorrect loop conditions such that the cycle is never executed.

[1] VHDL — very high-speed integrated circuits hardware description language.

2.2 Verification test plans

The verification test plan identifies the features of the design that are to be verified. Generally, features to be verified are interpreted and extracted from the design's specification or requirements document, as well as being described or enumerated by the engineer based on his knowledge of the design implementation.

2.3 Verification levels

The verification process is generally partitioned into the following design levels of granularity: block-level (sometimes referred to as unit-level verification), chip-level, and system-level. But from the modeling point of view, we have the following granularity: algoritmic level, computational model level (usually, FSM), RTL, gate level, netlist, layout, and silicon [1]. Each design level can be considered in terms of the modeling levels.

3 State-of-the-Art of Verification

Let us consider briefly all approaches to the design verification.

Figure 1 shows a general model of verification where "Design specification" and "Implementation" correspond to different stages of design process. For example, "Design specification" may correspond to an FSM while latter is an RTL one.

Design automation and verification from RTL and below has received most of the interests until the end of the 1990s, resulting in different software tools. At RTL, one or more automatic algorithms (BDD[1]-based equivalence checking, satisfiability (SAT) solving, FSM, model checking (MC), and bounded MC) are applied to: (i) prove a design equivalent to a specified logic function or to output of the circuit simulation; and (ii) prove that the circuit design fulfills a set of logical and temporal properties [1]. Despite the inherent complexity of these methods, their combination with the techniques such as symbolic simulation (semi-formal, in fact) has been reported as being essential in the verification of processor components and memory subsystems [8].

In contrast, the first formalized functional specification of a SoC[2], expressed at the so-called "transaction level," is usually written in terms of algorithms and abstract communications between large macroblocks.

Designers extensively execute test cases on simulation models written in languages such as Matlab, C, SystemC, etc. for which no formal hardware semantics and associated formal model extraction is available.

[1]BDD — binary decision diagram.

[2]SoC — system-on-chip.

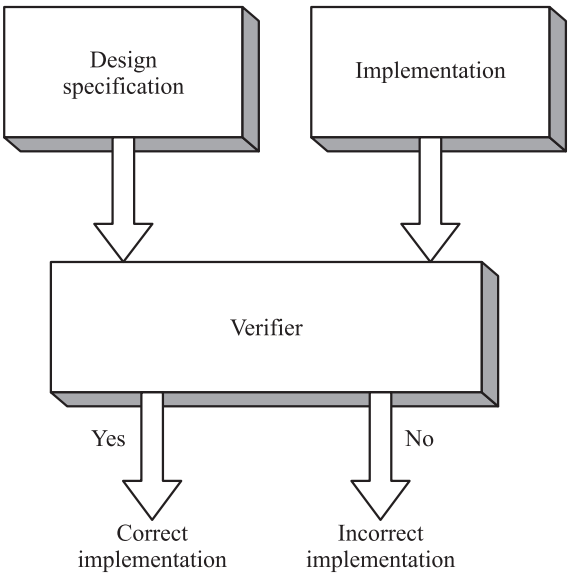

Figure 1 General verification model

In general, there are three classes of verification techniques: Simulation, Formal Verification, and Semi-formal Verification. Below we outline these techniques.

3.1 Simulation-based verification

In the modern chip design practice, design of micro-architectural specification is typically not formal and consists of textual descriptions, block diagrams, and parameter values. Hardware (in particular, RTL) description languages (HDLs) are such as Verilog or VHDL. Simulation-based design verification tries to cover design errors by detecting a circuit's faulty behavior when deterministic (functional) or pseudorandom tests (simulation vectors) are applied [3]. Simulation-based methods are readily applicable as typical design flows use an HDL, e.g., VHDL or Verilog, descriptions that can be simulated using standard logic simulation tools, or C/C++ descriptions in conjunction with a proprietary (cycle-based) simulator. Also, logic simulation is the area hardware designers are very familiar with.

Besides the immediate input vector simulation, so-called "Symbolic simulation" is used. The symbolic fault simulation is performed to avoid generation of new test vectors whenever a fault is covered by already generated test vectors. The internal BDD-based representation allows the analysis of more than one fault concurrently, thus improving the overall efficiency of the test generation procedure; in fact, implicit analysis of faults can be performed, while previous approaches required explicit analysis of each fault. More details are not reported here due to the space constraints [9, 10].

As for abstraction level, this approach deals with relation between faults at the behavioral and register-transfer or logic levels.

The simulation algorithms depend on the design error models. A set of error models that satisfy the requirements for the restricted case of gate-level logic circuits was developed in [3, 11].

The effectiveness of simulation can be increased if it is guided by coverage analysis [3]. A variety of coverage metrics have been proposed [6]: code coverage from software testing [12], FSM-based metrics [13]. The shortcoming of these metrics is that the relationship between a metric and the classes of design errors that they detect is not well understood.

The most important issues in simulation-based functional design verification are test generation, correctness checking, and functional quality measuring. Note that the effectiveness of the simulation depends on the design level. Block-level verification offers the best simulation performance (i.e., takes less time to verify) while system-level verification offers the worst performance. A drawback of block-level verification is that creating a test-bench, which is required to model the block-level environment (i.e., used to drive input stimulus into the block-level design and validate output responses), might become as complex as the actual design itself.

3.2 Formal verification

From the point of view of verification reliability, the main benefit of formal verification is that its every step is completely justified, that means that a formal proof of correctness can be easily performed. These methods rely on a language with mathematically-defined syntax and semantics. It is important that formal verification conducts exhaustive exploration of *all* possible behaviors compare to simulation, which explores *some* of possible behaviors:

- if correct, all behaviors are verified;
- if incorrect, a *counterexample* (proof) is presented.

Then:

- correctness is guaranteed mathematically, regardless the input values;
- no need to generate expected output sequences;
- can generate an error trace if a property fails: better understand, confirm by simulation;
- formal verification is useful to detect and locate errors in designs;
- consideration of *all cases is implicit* in formal verification.

That is ensures consistency with specification for *all* possible inputs (equivalent to 100 percent coverage!).

Presently, it is possible to define three principal approaches to formal verification:

(1) model checking;

(2) theorem proving; and

(3) equivalence checking.

3.3 Theorem proving

The theorem proving approach to formal verification is to describe the implementation as well as the specification in a formal logic, that is considering the specification as a *formal theory* [14].

It means that theorem proving based *verification* refers to the use of a finite set of well-founded formulas (*axioms of a formal theory*) and proof rules to prove the correctness of systems.

That is:

1. A finite set of rules of inference is given. A rule of inference allows the derivation of a new well-founded formula from a given finite set of well-formed formulas.
2. A formal proof in the theory S is a finite state of formulas $f_1, f_2, \ldots, f_i, \ldots, f_n$ such that for every i, formula f_i is either an axiom or can be derived by one of the rules of inference from the given set of formulas.

For example, let us specify by HOL (high-order logic) language [15] formally the simple circuit from Fig. 2.

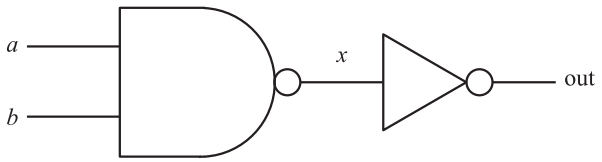

Figure 2 Possible implementation of the AND circuit in NAND-NOT elementary gate basis

We should specify the following objects:

- both NAND and NOT;
- implementation of the AND; and
- the behavior model of NAND and NOT;
- the intended behavior of the AND; and
- proof that the implementation satisfies its intended behavior.

The specification of implementation is, in fact, a definition of the AND circuit correct work. The AND gate correct work can be described by using predicate calculus (supported by HOL) in terms of correct behavior conditions of NAND and NOT:

$$\mathrm{NAND}(a,b,o) \equiv o = \neg(a\&b)\,;$$
$$\mathrm{NOT}(i,o) \equiv o = \neg i\,.$$

The AND correct implementation can be defined as:

$$\mathrm{AND\_IMP}(a,b,o) \equiv \exists x.\mathrm{NAND}(a,b,o)\&\mathrm{NOT}(x,o)$$

where $\neg$ means negation, o is the output variable, i is the formal variable of NOT input, $\mathrm{AND\_IMP}(a,b,o)$ means an implementation of desirable Boolean function by the circuit (see Fig. 2) that is the correct output of AND suppose that will be formed by correct NAND and NOT.

The desired behavior of the AND:

$$\mathrm{AND\_SPEC}(a,b,o) \equiv o = a\&b\,.$$

To define what it means for an implementation to satisfy some specification (the theorem proving), we can verify that the input/output behavior of the implementation always agrees with input/output of the specification, that is

$$\mathrm{AND\_IMP}(a,b,o) \rightarrow \mathrm{AND\_SPEC}(a,b,o)\,,$$

that is, the behavior of the implementation implies the behavior of the specification.

To prove this theorem it is enough to prove the satisfaction all of above logical conditions [14].

It is important that it ensures that parts fit together with no gaps. Many parts of a big problem can be solved automatically [9]. However, theorem prover mostly check detailed manual proof [4]. Therefore, main obstacle is that this is a time-consuming process that can be performed only by experts who are educated in logical reasoning and have considerable experience in this area.

3.4 Model checking

This is one of mostly popular formal verification methods. Model checking means to develop an abstract model of a circuit and to prove that this model satisfies the formal specification of the circuit implementation by comparing possible states of the system. If the model checker fails to prove a specified property of the model, an error is found [16–18].

In contrast to above example from the theorem proving where we describe all objects to be implemented, the MC deals mostly with *requirements* to target design, but not with the circuit structure. That is, a specification for MC verification is a collection of properties. There are many facets of hardware behavior that are of interest to the verification process including correct result generation, proper sequence of events including timing relationships between them, as well as other design properties such as arbitration "fairness." In particular, a property can be as a statement that particular pair of signals are never asserted at the same time, or it might state some complex relationship in timing of the signals.

Properties are specified in a notation called temporal logic. This allows concise specifications about temporal relationships between signals and can be automatically verified.

From the formal point of view, MC is used to verify finite state concurrent systems.

A labelled transition system is a tuple $(S, \Lambda, \rightarrow)$ where S is a set (of states), Λ is a set (of labels), and $\rightarrow S \times \Lambda \times S$ is a ternary relation (of labelled transitions) If p, $q \varepsilon S$ and $\alpha \varepsilon \Lambda$, then $(p, \alpha, q) \rightarrow$ represents the fact that there is a transition from state p to state q with label α. Labels can represent different things depending on the language of interest. Therefore, we should represent input expected, conditions that must be true to trigger transition, or actions performed during the transition.

However, there are also many reasons to use the MC for sequential hardware verification, say, for verification of their RTL implementation [16].

Before verification, it is necessary to state the properties that the design must satisfy. The specification is usually given in some logical formalism. For hardware and software systems, it is common to use *temporal logic*, usually CTL (computational tree logic) which can assert how the behavior of the system evolves over time. Let us consider briefly the CTL formulas.

There are two types of formulas in CTL: state formulas and path formulas. State formulas represent the properties of a specific state, whereas path formulas specify the properties of a specific path. From the practical point of view, the notion "path" means a path in a tree of computations (in particular, operations).

Computational tree logic introduces the following two path quantifiers:

(1) A — the universal path quantifier; and

(2) E — the existential path quantifier.

For example, let p1 and p2 denote various properties of paths. The path quantifiers are used as follows:

Ap1 — all paths beginning at some state s satisfy the property p1.

Ep2 — property p2 is satisfied for some paths beginning at some state s.

Computational tree logic introduces the following temporal operators:

X — the next time operator;

F — the eventually/future time operator; and

G — the global operator.

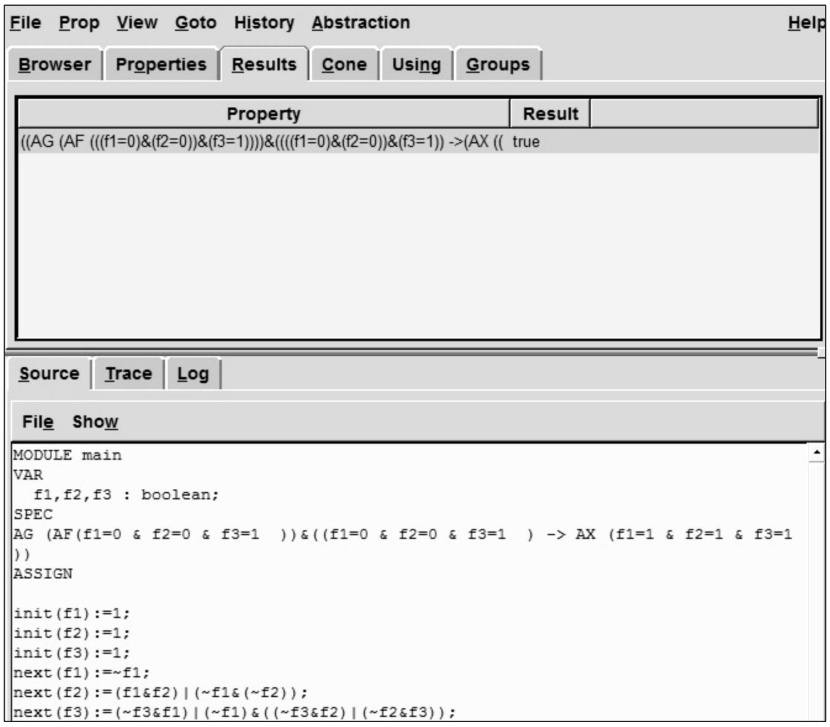

Figure 3 Three-bit counter property specification

For example, the F operator is used to express a condition that must hold true at some time in the future. The formula Fp is true at a given time if p is true at some later time. On the other hand, Gp means that p is true at all times. Usually, we read Fp as "eventually p" and Gp as "henceforth p" [17].

Using this notation, we can specify one property of a binary count Modulo 4 with the control inputs "stall" and "reset" as:

$$\mathrm{AG}[(\neg\mathrm{stall}\ \&\neg\mathrm{reset}\ \&(\mathrm{count}=C)\&(C<4)] \rightarrow \mathrm{A}x(\mathrm{count}=C+1)]$$

where "count" is the next state, C is the current one, and "$\rightarrow$" means "implies."

This means that for all times when inputs "stall" and "reset" equal to logical one, the counter transits to the next state if the current state is less the 4.

In practice, MC often is used as a Symbolic Modeling Verification (SMV), e.g., in the Cadence SMV input language, Verilog [18]. Symbolic modeling verification is quite effective in automatically verifying properties of combinational logic and interacting FSM. Sometimes, when the checking of a property fails, the tool will automatically produce a counterexample. This is a behavioral trace of the FSM that violates the specified property.

An implementation description for a circuit design at any given level serves also as a statement of the specification for a task at the next lower level. In this manner, top level specifications can be successively implemented and verified at each level, thus leading to the implementation of an overall verified system.

Symbolic modeling verification uses Ordered BDD (OBDD) algorithm to check whether the CTL specifications are met [18]. Ordered BDDs provide a canonical form for Boolean formulas that is often substantially more compact than conjunctive or disjunctive normal form. Because the symbolic representation captures some of the regularity in the state space determined by circuits and protocols, it is possible to verify systems with extremely large numbers of states (more than 10^{20}).

Figures 3 and 4 demonstrate verification a property of 3-bit counter by SMV version 2.1 tool.

We can see that even such simple design (3 possible states) requires 115 nodes of BDD to verify one property. All the more, for large designs, especially those including substantial data path components, the user must break the correctness proof down into small enough pieces for SMV to verify.

There are two mechanisms provided for this purpose: *composition* and *refinement*. In the compositional method, one verifies temporal logic properties of one part of the system and uses these properties as assumptions when verifying another part of the system. In the refinement method, one uses a high level model of the system as a specification and verifies separately that each

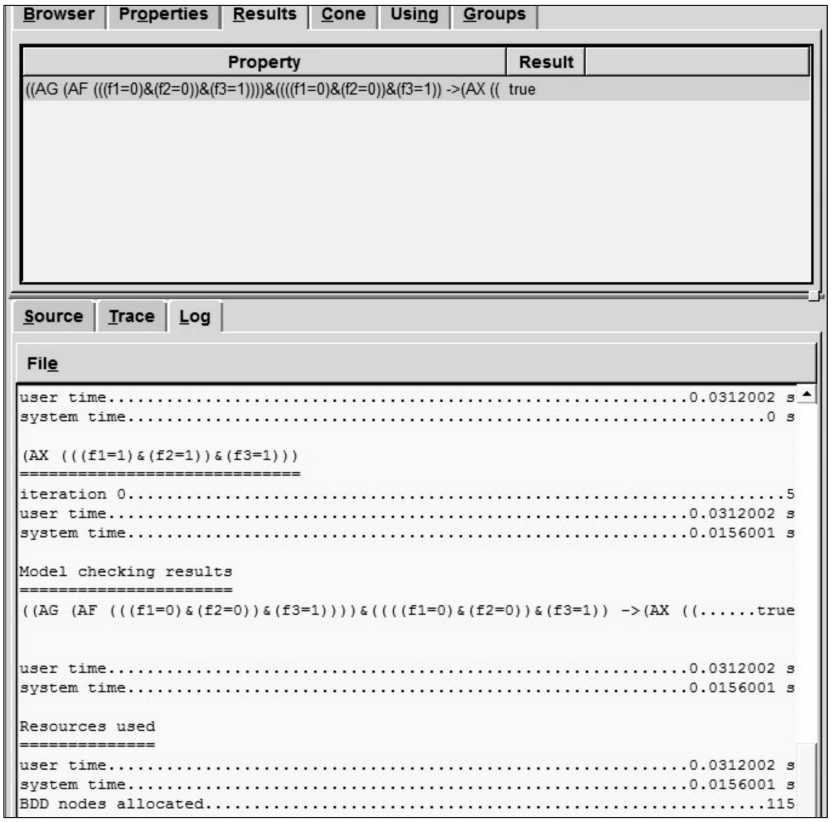

Figure 4 Result of the figure 3 property verification

system component implements its part of the high level specification [10].

In general, MC has a number of advantages over traditional other approaches to this problem that are based on simulation/testing and theorem proving mentioned above. Given sufficient resources, the procedure will always *terminate* with corresponding negative or positive answer for the verification question. Although the restriction to finite state may seem to be a major disadvantage, MC is applicable to several very important classes of systems, e.g., hardware controllers [16].

The properties descriptions are performed usually manually. Also, analysis of the verification results is the manual activity. In case of a negative result, the user is often provided with an error trace. In this case, analyzing the error trace may require a modification of the system and reapplication of the MC algorithm.

Moreover, since it is necessary to convert a design into a formalism accepted by an MC tool, in the case of owing to limitations on time and memory, the modeling of a design may require the use of abstraction to eliminate irrelevant or unimportant details that require many efforts of the designer, and they must be experienced in formal logic.

Symbolic MC is primarily useful in verifying the control parts of a circuit. It is impractical for most data paths, since it suffers from the state explosion problem where the state space of the design that must be explored grows extremely large. In addition, performing MC on design modules requires that an interface specification for the module is available so that only legal inputs are considered. However, in practice, detailed interface specifications of design modules are rarely available and so verification often requires the creation of the interface specification. This can be a complex task. Besides, the MC formal verification approach is heavily dependent on experienced users who must specify the properties of the design that are to be checked. The reliance on a design engineer, who must provide knowledge about design behavior and the design properties to be analyzed, has limited the adoption of this technology. An additional issue is that there is no metric to judge design property coverage and, thus, there is no confidence that all design properties have been verified.

3.5 Equivalence checking

Equivalence Checking (EC) approach to design verification is used to check the functional equivalence of different versions of a design at various stages of the de-

sign and enables designers to identify and correct these errors. This approach is based on some algorithms of toggle equivalence of Boolean functions [19].

One of widely used EC-based design verification tools is Encounter Conformal EC of Cadence [19, 20]. The tool includes technologies developed independently from the design flow, including production-proven HDL parsing, synthesis, mapping, optimization, and data path algorithms. Using Encounter Conformal EC ensures that the maximum number of design bugs will be caught.

Unfortunately, it covers rather later stages of design process, usually starting from RTL, comparing, for example, RTL with gate level. Thereby, it checks only targets *implementation errors*, not *design errors*. But the hard bugs are usually in both descriptions.

Note that often designers try to combine different formal verification tools and methodologies, for example, theorem proving (for combinatorial parts of a design) and trajectory evaluation (that is a modification of MC) [9].

3.6 Petri Nets based verification

Petri Net theory formal verification algorithms are used presently to detect design errors for high-level synthesis of dataflow algorithms. In [21], the Petri Net theory is adopted to verify the correctness of the synthesis result, because the Petri Net model has the nature of dataflow algorithms. In fact, it proposes three approaches to realize the Petri Net based formal verification algorithm and conclude the best one which outperforms the others in terms of processing speed and resource usage. Presently, there are several examples of using the Petri Nets in synthesis and verification of asynchronous circuit [21, 22].

3.7 Semi-formal verification

As it follows from the above, the formal verification tehniques do not address the problems of finding bugs during verification. On the other hand, the most common problem is that these tools fail when they hit a capacity limit in time, memory, or both.

However, the simulation techniques are also very time-consumting due to the large number of test vectors needed to manifest all functional issues also prohibitively large.

This problem leads to the concept of semi-formal verification [23]. The idea is to combine the strengths of simulation — namely, ease of use and the ability to handle large designs with the thoroughness of formal verification along with various combinations of formal and simulation-based approaches to verification cost reduction.

In fact, there are two possibilities to perform such combination:

(1) a "mechanical" combination of the verification techniques: part of design is verified by simulation, while another by a formal method; and

(2) by using a semi-formal specification for implementation and establishment of properties for the formal verification. For example, Property Specification Language (PSL) can be used for an alignment circuit [24]. Finally, a set of properties for the verification of this module was established and proved using two verification tools.

Property specification language is an extension of the CTL. The integrating feature of PSL tool (that is both formal and simulation-based abilities) is due to the fact that it can be used for functional specification on one hand and as input to functional verification tools on the other.

Property specification language can also be used as input to verification tools, for both verification by simulation, and as formal verification using a model checker or a theorem prover. Besides, PSL specification should be used to automatically generate checks of simulations. This can be done, for example, by directly integrating the checks in the simulation tool, by interpreting PSL properties in a testbench automation tool that drives the simulator, by generating HDL monitors that are simulated alongside the design, or by analyzing the traces produced at the end of the simulation.

Since complex microprocessor systems design verification activity deals, in general, with many optional variants (formal verification and simulation), it should be useful to have a characterization of both verification algorithms and verification process on a whole which includes the decomposition issues, dividing possible (potential) design bugs classes between formal verification and simulation, final quality analysis, etc. Obviously, we need a guide to provide this hybridization [3]. Following well showed itself conception of coverage analysis used widely in test pattern generation practice, it would be very attractively to have also similar one for the design formal verification. Some steps towards this notion development are just in progress [11, 25]. As for formal verification, the notion of coverage in functional verification is to cover the entire functionality specification required from the implementation. This notion involves two questions [26]:

(1) whether we can provide (to take into account) (explicitly or implicitly) all possible input sequence, and

(2) whether the specification contains a sufficient set of properties.

However, in fact, the coverage metrics computation has the same exponential complexity as the properties modeling, or exhaustive simulation. Thereby, it remains the mentioned above problem to find the design bugs relying on the semi-formal verification results. We can overcome these drawbacks if the verification activity would follow the target design synthesis. In particular, these circumstances allow the RTL as a higher possible level of design. Meanwhile, as it is shown by numerous results, many design bugs can be detected at the earlier design stages.

Therefore, let us consider a new approach to the verification which takes into account the structure of synthesized system at every of abstraction levels, starting from algorithmic one up to gate level.

4 Algorithmic-State-Machine-Based Synthesis and Verification

Let us consider an approach to concurrent synthesis and verification of digital systems, including complex microprocessors, based on design specification by ASM.

Algorithmic state machine is a flowchart consisting of state boxes, condition boxes, and boxes "Begin" and "End."

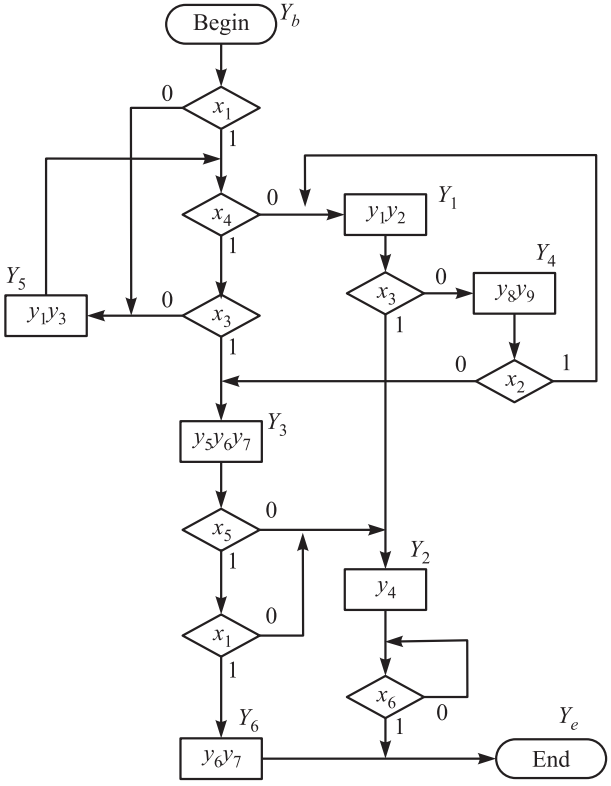

Figure 5 Example of ASM

As an example, we will use the ASM in Fig. 5. Let us look at operators following operator Y_b. The fact that operator Y_3 is implemented after Y_b when $x_1x_4x_3 = 1$, operator Y_1 is implemented after Y_b when $x_1x_4' = 1$, and operator Y_5 is implemented after Y_b when $x_1x_4x_3' = 1$ or $x_1' = 1$, can be represented as a formula

$$Y_b \to x_1x_4x_3Y_3 + x_1x_4x_3'Y_5 + x_1x_4'Y_1 + x_1'Y_5\,. \quad (1)$$

The ASM-based synthesis is based on ASM transformations (composition, minimization, extraction, etc.), special algorithms for Data Path and CU design, and very fast optimizing synthesis of FSM and combinational circuits with hardly any constraints on their size, that is, the number of inputs, outputs, and states.

From the functional point of view, each rectangle (an operator vertex) of the diagram corresponds to execution of some microoperations written inside, and from the timing viewpoint, its passing means to transit to the next clock of synchronous system designed.

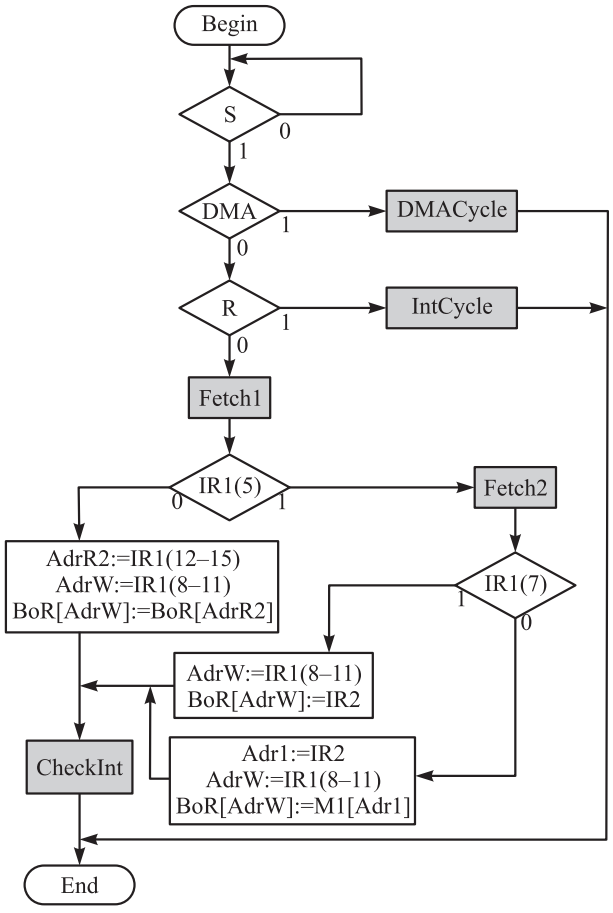

Figure 6 Algorithmic state machine with included blocks (15 vertices)

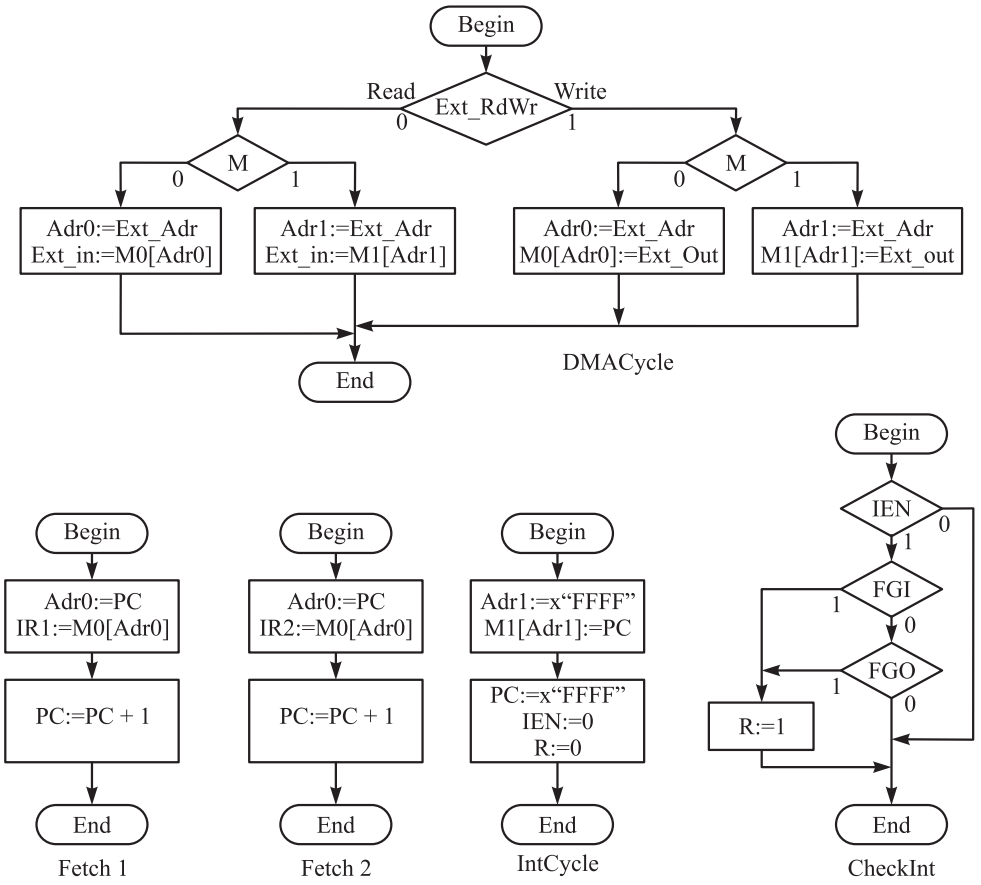

Figure 7 Included blocks

Each rhomb means corresponding conditions (conditional vertexes) — Boolean variable. That is, the ASM represent both algorithmic and time-ordering specification of target systems with desirable ordering of the microoperations. The aim of this synthesis is to get both a CU and Data Path of the target design.

High-level synthesis based on the ASM contains three stages (Figs. 6 to 8).

Stage 1 corresponds to the construction of combined functional ASM and FSM.

In particular, drawing functional ASM in ASM Creator *is presented in* box 1 in Fig. 6. At this stage, we suppose that we know the main units of Data Path, say memory, Arithmetic Logic Unit (ALU), Block of Registers (BoR), program counter, instruction register, input and output registers, etc. However, at this stage, we do not have a structure of Data Path — we do not know how these units are connected, what buses (multiplexers) should be used, which units are connected directly without buses, etc.

Let us consider principal steps of this stage.

Step 1. Functional ASMs specification (see box 1 in Fig. 8). When constructing a very complicated digital system, sometimes it is very difficult to describe its whole behavior by one ASM. In such case, it is possible to describe separate subbehaviors with ASMs $G_1 \ldots G_Q$ and then to combine them into one combined functional ASM G. From the verification point of view, the decomposition ability described above is very important feature if the system models become very large. The drawing and decomposition are provided by the software tool named *ASM Creator*. At this step, a designer draws separate ASMs for each subbehavior (for each operation or a group of operations) in our *ASM Creator*. We can use included blocks as well. It means that one operator vertex can represent a whole ASM or a subgraph of ASM. An example of ASM with four included blocks (see shaded operator vertices) is presented in Fig. 6. Blocks themselves are shown in Fig. 7.

Step 2. Combining of several Functional ASMs into one Combined Functional ASM (see box 2 *ASM combiner*, Fig. 8). At this step, separate subbehaviors presented by ASMs $G_1, \ldots, G_Q$ are combined into one combined functional ASM G. At the same time, this procedure minimizes the number of operator vertices in the combined ASM.

Step 3. Minimization of combined Functional ASM (see box 3 *ASM minimizer*, Fig. 8). This procedure

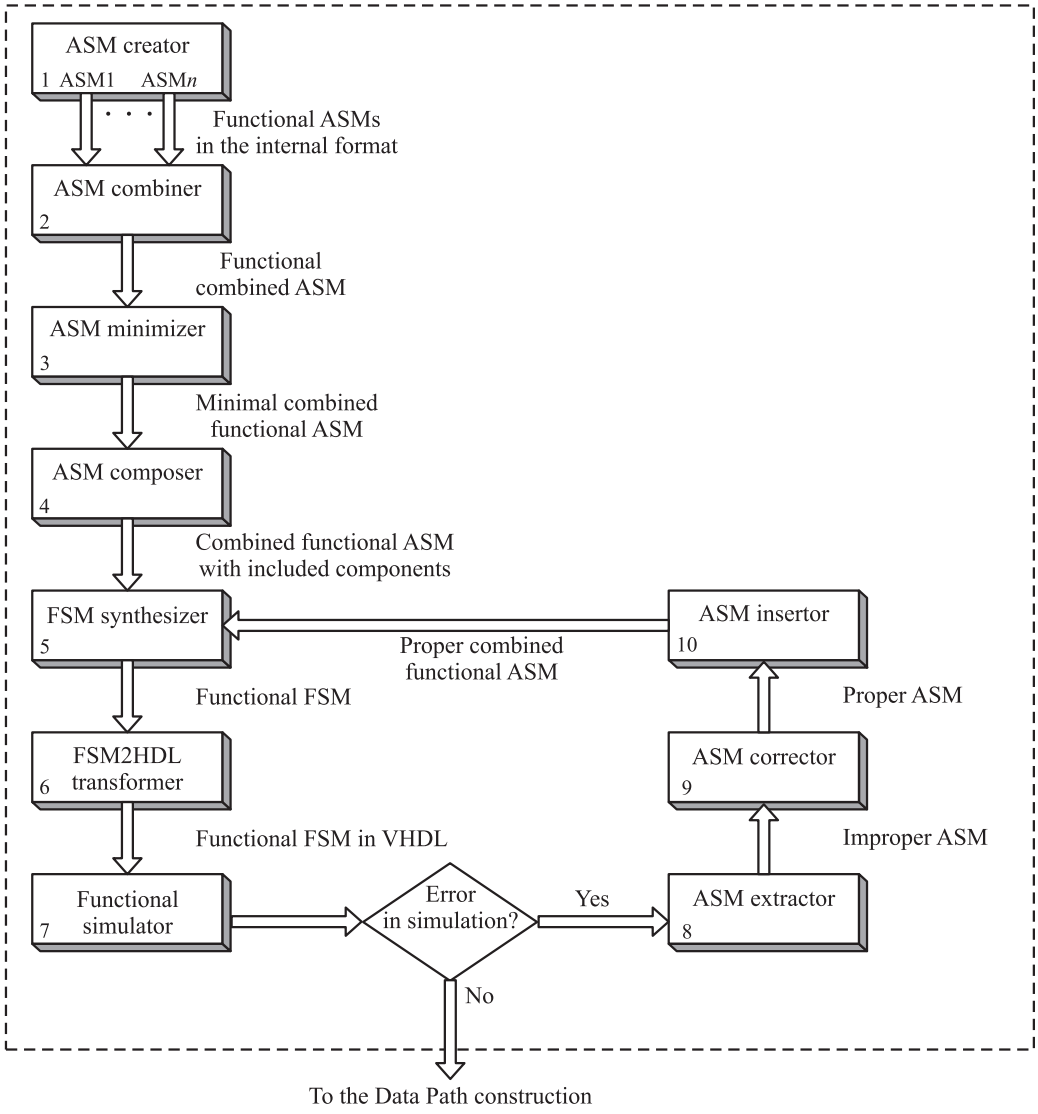

Figure 8 Construction of combined functional ASM and FSM

minimizes the number of conditional vertices in the combined functional ASM. Such minimization allows us to reduce dramatically the number of vertices in the ASM (sometimes, in two or three times) and to reduce the complexity of logic circuits at the stage of logic design.

Again, in the description at the level of a functional ASM, we do not have Data Path structure and each unit in such functional ASM is presented as a variable. For example, in microoperation BoR[AdrW] := M[Adr1] for the word of memory M with address Adr, we do not know yet, and would not like to know, how these units are connected and what signals must come from the CU to Data Path to implement this transfer. Really, a Data Path does not exist yet and our goal at the second stage (Data Path Construction, see Fig. 8) is to construct a Data Path formally using only the combined functional ASM. But we have not finished yet with the first stage.

Step 4. Synthesis of FSM from combined Functional ASM (see box 5 *FSM synthesizer*). This procedure constructs various types of minimized FSM tables for Mealy, Moore, and their combined model with or without state assignment (*log* or *one-hot*).

Step 5. Construction of VHDL (Verilog) code for the functional FSM (see box 6 *FSM2HDL tranformer* that can include both *VHDL and Verilog transformers*).

We can use Functional FSM in VHDL for a **functional simulation** (not an RTL simulation, it is possible to make it later when we construct a Data Path).

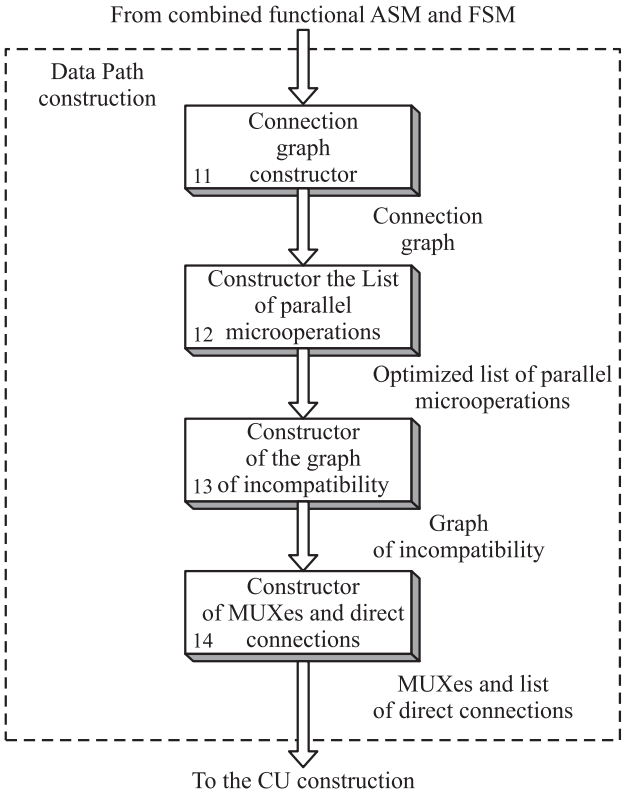

Figure 9 Data Path design

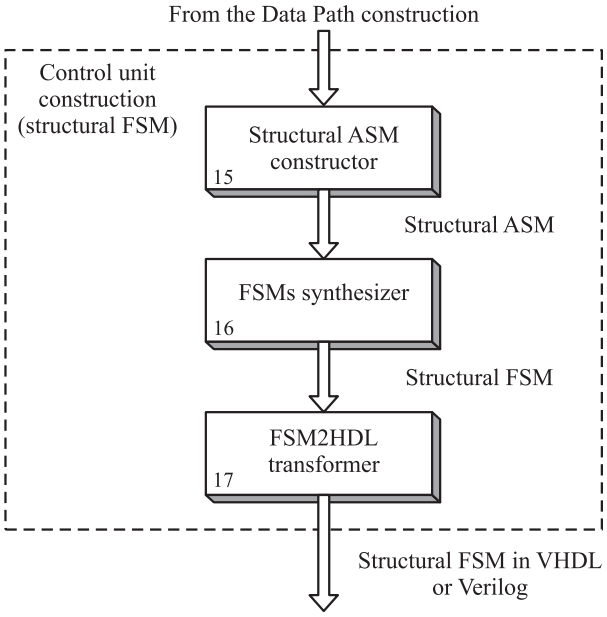

Figure 10 Control unit design

Step 6. Extraction of one or several ASMs from Combined ASM (see box 8 *ASM extractor*). If we detected error during the simulation, we can extract ASM with error (see box 8 *ASM extractor*), repair it (see box 9 *ASM corrector*), and return corrected ASM into combined Functional ASM (see box 10 *ASM inserter*).

We have a special program "Check ASM equivalence" that verifies the equivalence of two ASMs. That permits designer to check each step of ASM transformer.

Stage 2 corresponds to Data Path synthesis.

First, a Connection graph from the functional ASM design is constructed. Such graph contains a list of sources and targets for each component of an operational unit and some metrics that will be used in the optimization of the Data Path. Next, an optimized list of parallel microoperations to increase the speed of the designed system is constructed. Then, we design the Graph of incompatibility from the Connection graph and the List of parallel microoperations. On the final step of Data Path synthesis, we construct multiplexers (buses) by coloring the Graph of incompatibility and the List of direct connections from the Connection graph (Fig. 9).

Stage 3 deals with a CU of the target system (Fig. 10). It contains the foloowing three steps.

Step 1. After design of the Data Path, we can immediately transfer from the functional ASM to the structural ASM.

Step 2. Construction of VHDL (Verilog) code for the structural FSM (see box 17 *FSM2HDL transformer* that can include both *VHDL and Verilog transformers*).

That is, using the functional ASM (stage 1) and the MUXes and the List of direct connections, we immediately construct the structural ASM. This ASM describes the behavior of the CU corresponding to the Data Path. On the last step of this stage we construct the FSM and its multilevel logic circuit.

Step 3. VHDL code design.

The Data Path constructed according to our design methodology does not contain any "cloud" (irregular) circuits. It makes it possible to simplify considerably VHDL or Verilog code for the Data Path using the structure style of VHDL to combine VHDL or Verilog codes of units. Moreover, the presented formal methods of system design allow us to formalize the design of test bench for the Data Path and to reduce it considerably. VHDL code for the CU is constructed automatically. VHDL code for the top level of the system is the result of combining VHDL codes of the Data Path and the CU.

The design methodology is implemented in *Abelite* software tool [27].

4.1 About algorithmic state machine formalities

A possibility to use some ASM-based formalized verification is due to some formal rules, used for ASM flowchart construction [5, 27]. Namely, to provide this unique correspondence between the ASM flowchart and a target data path and CU, it is enough that a synthesis algorithm would obey the following rules [5]:

1. State boxes should contain only register statements, control signals in parentheses.
2. All operations within a state box should be concurrently executable in one clock cycle.
3. If the operations in two consecutive state boxes can be executed in the same clock cycle, then these two state boxes can be combined into one state box.
4. For each register-transfer statement, there must be a path between the source and destination registers.

If a transformation takes place during the transfer, then a combinational device, such as an adder or ALU, must be inserted into the path between the source and destination registers if there are several paths.

Finally, control signal inputs must be attached to each register and multiplexer so that register transfers can be precisely controlled. It is essential that as it was mentioned above, we may simulate the ASM description to check its correctness regarding to some initial description (formal or informal) of the target design, providing necessary numbers of feedbacks, including customer's checking. This activity, as it was mentioned above, deals with generation of some test benches. Let us define the possible bugs of ASM description as *Behavioral faults*.

4.2 Verification test benches preparation during synthesis

It can be shown that if a target system has been synthesized with the use of design methodology described above, all paths of its Data Path are activated by the microoperations, represented in minimized ASM (obtained at Stage 1, see Fig. 8). It means the verification of the Data Path if the synthesis leads to irredundant logical structure. This is achieved due to minimization mentioned above.

Let us describe explicitly some properties of the Abelite design methodology, affecting the verification problem, as well as some assumptions necessary to correctly map of behavioral faults to structural bugs in the synthesized structures [7].

1. The behavioral synthesis tool maps each high-level operator to a regular structure of modules (in a single *module, in particular*), and each *such structure* corresponds to a single microoperation. It means that the allocation algorithm cannot share *modules* between equal microoperations (that is provided by the synthesis technique mentioned above).
2. Taking into account the delay (in term of clock cycles) introduced in the high-level description, it is possible to include explicitly various synchronization statements in the behavioral (ASM) description. That is the description contains the ordering of microoperations, namely, each of rectangle take one clock for its execution.
3. The structural bugs listed above of a structural (RTL, in particular) implementation are considered only at the interface of each block, that is, faults internal to blocks are not considered (this assumption will be partially removed in the following).

This set of faults at the structural (Data Path, RT) level is called SF.

Let BF is a set of behavior design faults (possible incorrectness in the ASM description). Let MSF be the set of structural faults on the input and output lines of each *module of data path* [7].

Proposition 1. Each fault $\mathrm{msf} \in \mathrm{MSF}$ is equivalent to a fault belonging to BF and *vice versa*.

The msf fault modifies maps the functionality of *module* M into the *module* M'. From property 1, there is one and only one micro*operation OP* that is implemented by the *module* M. Moreover, from the definition of the behavioral fault, there is a fault $\mathrm{bf} \in \mathrm{BF}$ on the same input (or output) line affected by msf so that the faulty microoperation OP', associated with bf, has the same behavior of M'. Moreover, the faulty behavior of the FSM-based CU is equivalent to the faulty behavioral description where OP' replaces OP. The equality of the faulty descriptions implies that each fault of MSF is equivalent to a fault of BF. The opposite can be shown in a similar way.

The most important result of the proposition concerns translation of behavioral test vectors into RTL test sequences [7]. In fact, a test vector, identified for a behavioral fault bf, detects the equivalent RTL fault smf, if the test vector is applied to the sequential circuit for T clock cycles, where T is at most the sequential depth of the faulty circuit. Since this information cannot be known at the behavioral level, it can be upper-bounded by using the worst-case complexity of the algorithm and the information concerning the chosen microoperations ordering.

This proposition [7] shows that it is possible to formalize and to automate design of the test bench for Data Path using the Abelite design methodology. As it was mentioned above, the main thing in this methodology is the isomorphism of the Data Paths and the ASM paths, and the ability to minimize the number of possible ASM paths.

Table 1 Instruction set

Type	Name	Description	Code $b0$–$b4$	Format and addressing mode		
				Short $b5=0$	Long $b5=1$	
				Dir	Dir $b7=0$	Imm $b7=1$
aosh	add	Add Op1 and Op2; store result in Op1	00 001	•		
	and	Bitwise logical AND between Op1 and Op2 store result in Op1	00 010	•		
	sub	Subtract Op2 from Op1; store result in Op1	00 011	•		
	sh1	Shift Op2 one bit to the left; store result in Op1	00 100	•		
	shr	Shift Op2 one bit to the right; store in Op1	00 101	•		
	cil	Rotate Op2 to the left; store result in Op1	00 110	•		
	cir	Rotate Op2 to the right; store result in Op1	00 111	•		
load	lod	Copy Op2 into Op1	10 000	•	•	•
	str	Copy Op1 into Op2	10 001	•	•	
	inc	Increment Op1; store result in Op1	10 010	•		
	dec	Decrement Op1; store result in Op1	10 011	•		
	com	Complement Op1; store result in Op1	10 100	•		
branch	bcz	If $z=1$, load new address into PC	01 000	•		•
	bcf	If $v=1$, load new address into PC	01 001	•		•
	bcc	If $c=1$, load new address into PC	01 010	•		•
	bun	Branch unconditionally to a new address	01 011	•		•
inout	ski	Skip next instruction (if flag of input fgi $=1$, increment PC twice)	11 000			
	sko	Skip next instruction (if flag of output fgo $=1$, increment PC twice)	11 001			
	inp	Copy input register into one of the registers of BOR and reset fgi	11 010	•		
	out	Copy one of the registers from BOR into output register and reset fgo	11 011	•		
	ion	Set flag interrupt enable (ien) to 1	11 100			
	iof	Set flag interrupt enable (ien) to 0	11 101			

So, taking into account the mentioned above properties of designs synthesized by Abelite, we can produce in semiautomatic manner a set of test benches (both in VHDL and Verilog) to verify both ACM flowchart and synthesized circuit, including Data Path and CU, both functional (FSM-based) and RT levels.

The input for these test benches generation are some tables data which are collected during the synthesis. These data are the logical values which determine corresponding branches in the ASM flowchart. These data are used in corresponding section of VHDL (or Verilog) program of the target design, generated automatically during the synthesis. For example, let us consider the design of rather simple RISC processor, with instructions set from Table 1.

The test benches to verify the correspondence of the design functional and structural level can be extracted from Table 2, generated during the synthesis. This table, in fact, establishes a mapping between functional, microoperation-based model of the microprocessor (left part of the table), and structural representation of the Data Path. This is what we can use to generate any test benches both for functional and Data Path models verification.

The desirable behavior we can get from ***functional simulation***. Functional FSM simulation in VHDL for not an RTL simulation — it is possible to make such later when we construct a Data Path.

As for post-RTL levels verification (gate level, etc.), we can use presently the widely-spread tools such as Mentor Graphic, Cadence, Synopsis, using their VHDL (Verilog) models, generated as a result of described above synthesis with Abelite (Fig. 11).

In contrast to the considered semi-formal methods, the ASM-based verification methodology saves a connection between diagrammatic description (formal, in fact) and its structural representation, as it deals not only with some system's properties, but also with structural elements, mapped from ASM by corresponding algorithm synthesis. It allows to localize errors at different description levels, while recognition of design errors is rather difficult to derive from a fact of violation of some properties.

On the other hand, while traditional formal methods deal explicitly either with functional faults or timing ones, ASM-based specification represents both functional (as microoperations) and timing-ordering aspects.

Table 2 Process table

Functional ASM				Structural ASM		
Microinstr		Functional microoperations		Structural microoperations	Minimized structural microoperations	
Y1	y1	AdrW:=IR1(8-11)	4	ctr_mux3 := 0	ctr_mux2[1] := 1	y1
	y2	BoR[AdrW]:=RALU	16	ctr_mux2 := 0110	ctr_mux2[2] := 1	y2
				bor_en := 1	sbor_en := 1	y3
Y2	y3	AdrR1:=IR1(8-11)	4	ralu_en := 1	ralu_en := 1	y4
	y4	AdrR2:=IR1(12-15)	4	cf_en := 1	cf_en := 1	y5
	y5	ALU1:=BoR[AdrR1]	16	zf_en := 1	zf_en := 1	y6
	y6	ALU2:=BoR[AdrR2]	16	vf_en := 1	vf_en := 1	y7
	y7	ctrALU:=IR1(0-4)	5			
	y8	RALU:=ALU	16			
	y9	cf:=c	1			
	y10	zf:=z	1			
	y11	vf:=v	1			
Y3	y4	AdrR2:=IR1(12-15)	4	ralu_en := 1	ralu_en := 1	y4
	y6	ALU2:=BoR[AdrR2]	16	cf_en := 1	cf_en := 1	y5
	y7	ctrALU:=IR1(0-4)	5	zf_en := 1	zf_en := 1	y6
	y8	RALU:=ALU	16	vf_en := 1	vf_en := 1	y7
	y9	cf:=c	1			
	y10	zf:=z	1			
	y11	vf:=v	1			
Y4	y12	Adr1:=IR2	16	ctr_mux1 := 001	ctr_mux1[2] := 1	y8
	y1	AdrW:=IR1(8-11)	4	ctr_mux3 := 0	ctr_mux2[3] := 1	y9
	y13	BoR[AdrW]:=M1[Adr1]	16	ctr_mux2 := 0001	bor_en := 1	y3
				rdwrM1 := 0		
				bor_en := 1		
Y5	y1	AdrW:=IR1(8-11)	4	ctr_mux3 := 0	ctr_mux2[0] := 1	y10
	y14	BoR[AdrW]:=IR2	16	ctr_mux2 := 1000	bor_en := 1	y3
				bor_en := 1		
Y6	y4	AdrR2:=IR1(12-15)	4	ctr_mux3 := 0	ctr_mux2[1] := 1	y1
	y1	AdrW:=IR1(8-11)	4	ctr_mux2 := 0101	ctr_mux2[3] := 1	y9
	y15	BoR[AdrW]:=BoR[AdR2]	16	bor_en := 1	bor_en := 1	y3
Y7	y12	Adr1:=IR2	16	ctr_mux1 := 001	ctr_mux1[2] := 1	y8
	y3	AdrR1:=IR1(8-11)	4	ctr_mux2 := 0000	rdwrm1 := 1	y11
	y16	M1[Adr1]:=BoR[AdrR1]	16	rdwrM1 := 1		
Y8	y3	AdrR1:=IR1(8-11)	4	ctr_mux3 := 1	ctr_mux3 := 1	y12
	y17	AdrW:=IR1(12-15)	4	ctr_mux2 := 0000	bor_en := 1	y3
	y18	BoR[AdrW]:=BoR[AdrR1]	16	bor_en := 1		
Y9	y3	AdrR1:=IR1(8-11)	4	ralu_en := 1	ralu_en := 1	y4
	y5	ALU1:=BoR[AdrR1]	16	cf_en := 1	cf_en := 1	y5
	y7	ctrALU:=IR1(0-4)	5	zf_en := 1	zf_en := 1	y6
	y8	RALU:=ALU	16	vf_en := 1	vf_en := 1	y7
	y9	cf:=c	1			
	y10	zf:=z	1			
	y11	vf:=v	1			
Y10	y4	AdrR2:=IR1(12-15)	4	ctr_mux1 := 100	ctr_mux1[0] := 1	y13
	y19	PC:=BoR[AdrR2]	16	pc_en := 1	pc_en := 1	y14
Y11	y20	PC:=IR2	16	ctr_mux1 := 001	ctr_mux1[2] := 1	y8
				pc_en := 1	pc_en := 1	y14
Y12	y21	PC:=PC+1	0	pc_count := 1	pc_count := 1	y15

(Continued)

Table 2 Process table (Continued)

Functional ASM				Structural ASM		
Microinstr		Functional microoperations		Structural microoperations	Minimized structural microoperations	
Y13	y1	AdrW:=IR1(8-11)	4	ctr_mux3 := 0	y1	y3
	y22	BoR[AdrW]:=InpR	16	ctr_mux2 := 0100		y16
	y23	FGI:=0	0	bor_en := 1		
				fgi_reset := 1		
				ctr_mux2[1] := 1		
				bor_en := 1		
				fgi_reset := 1		
Y14	y3	AdrR1:=IR1(8-11)	4	outr_en := 1	y17	
	y24	OutR:=BoR[AdrR1]	16	fgo_reset := 1	y18	
	y25	FGO:=0	0	outr_en := 1		
				fgo_reset := 1		
Y15	y26	IEN:=1	0	ien_set := 1	ien_set := 1	y19
Y16	y27	IEN:=0	0	ien_reset := 1	ien_reset := 1	y20
Y17	y28	PC:=x“FFFE”	16	ctr_mux1 := 011	ctr_mux1[1] := 1	y21
	y27	IEN:=0	0	pc_en := 1	ctr_mux1[2] := 1	y8
	y29	R:=0	0	ien_reset := 1	pc_en := 1	y14
				r_reset := 1	ien_reset := 1	y20
					r_reset := 1	y22
Y18	y30	Adr1:=x“FFFF”	16	ctr_mux1 := 101	ctr_mux1[0] := 1	y13
	y31	M1[Adr1]:=PC	16	ctr_mux2 := 1001	ctr_mux1[2] := 1	y8
				rdwrM1 := 1	ctr_mux2[0] := 1	y10
					ctr_mux2[3] := 1	y9
					rdwrm1 := 1	y11
Y19	y32	Adr0:=Ext_Adr	16	ctr_mux1 := 000	rdwrm0 := 1	y23
	y33	M0[Adr0]:=Ext_Out	16	rdwrM0 := 1		
Y20	y34	Adr1:=Ext_Adr	16	ctr_mux1 := 000	ctr_mux2[2] := 1	y2
	y35	M1[Adr1]:=Ext_Out	16	ctr_mux2 := 0010	rdwrm1 := 1	y11
				rdwrM1 := 1		
Y21	y34	Adr1:=Ext_Adr	16	ctr_mux1 := 000	ctr_mux2[3] := 1	y9
	y36	Ext_in:=M1[Adr1]	16	ctr_mux2 := 0001		
				rdwrM1 := 0		
Y22	y32	Adr0:=Ext_Adr	16	ctr_mux1 := 000	ctr_mux2[2] := 1	y2
	y37	Ext_in:=M0[Adr0]	16	ctr_mux2 := 0011	ctr_mux2[3] := 1	y9
				rdwrM0 := 0		
Y23	y38	R:=1	0	r_set := 1	r_set := 1	y24
Y24	y39	Adr0:=PC	16	ctr_mux1 := 010	ctr_mux1[1] := 1	y21
	y40	IR1:=M0[Adr0]	16	rdwrM0 := 0	ir1_en := 1	y25
				ir1_en := 1		
Y25	y39	Adr0:=PC	16	ctr_mux1 := 010	ctr_mux1[1] := 1	y21
	y41	IR2:=M0[Adr0]	16	rdwrM0 := 0	ir2_en := 1	y26
				ir2_en := 1		

As it was mentioned above, one of very important thing in this methodology is the ability to minimize the number of possible ASM paths. For example, let a microprocessor that should be synthesized contains a multiplexer presented in Fig. 12.

From this figure for MUX1, it is evident that ext_adr (in0) must be transferred only to two inputs of our units — adr0 and adr1, ir2 (in1) — to adr1 and pc, all other inputs of MUX1 must be transferred only to one of the three possible units reachable from MUX1. So, instead of checking $6 \times 3 = 18$ transfers through MUX1, we must check only 9 (the number of units written over the inputs of MUX1). In the same way, instead of checking $10 \times 3 = 30$ transfers, it is sufficient to check only 11 transfers through MUX2 during simulation of Data Path.

The test benches for CUs in case of using Abelite will be obtained automatically.

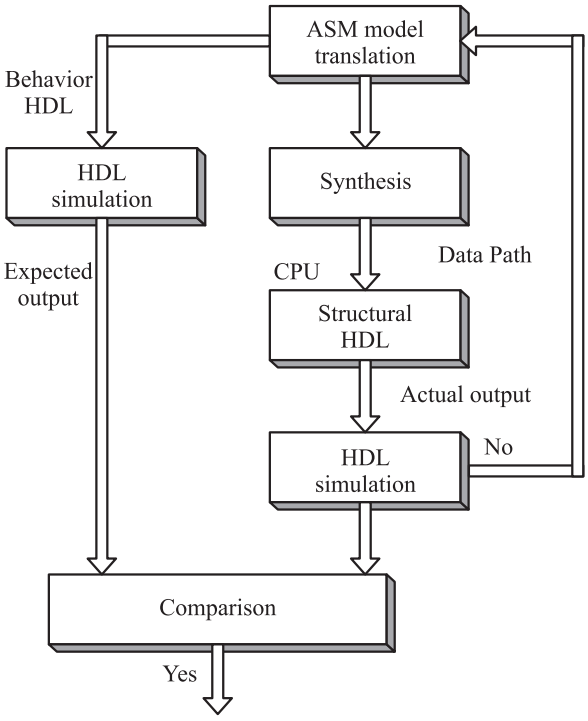

Figure 11 Algorithmic-state-machine-based verification

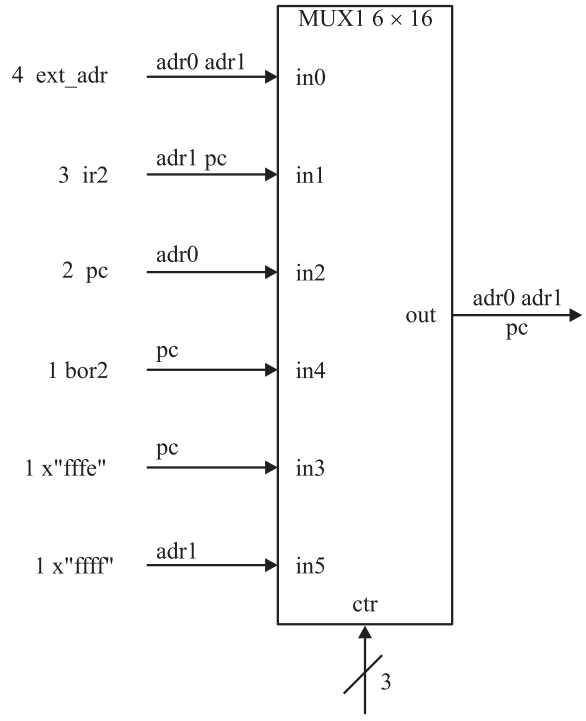

Figure 12 Multiplexer 6 × 16

Another experience of using Abelite for verification goals dealt with design and verification of a bridge for PCI to AXI protocols buses. The formal model of the bridge verification (e.g., based on a language of the temporal logic) becomes very big and time-consuming for implementation [28]. The semi-formal Abelite methodology allowed to prepare some test benches with reasonable time expenses relying on ASM-based high-level specification of the bus protocols for verification purposes.

5 Concluding Remarks

Main benefit of formal-verification approahes is that they do not require the user to create any test vectors and can significantly accelerate verification efforts for large designs.

However, formal verification is a time-consuming process that can be performed only by experts who are educated in logical reasoning and have considerable experience. Besides, present formal verification techniques allow the model levels starting from FSM and RTL levels, while the higher levels can be very error-prone ones.

Various semi-formal approaches dealing with combinations of formal methods with some fault-simulation techniques also require a lot of manual work. This activity affects considerably on the amount of verification cost of general design process [3].

The suggested approach allows to overcome these drawbacks in cases, when the design process is beginning just from an algorithmic level of a target system. It is possible also to use this approach in cases, when there is an HDL description at RTL level, and we would like to verify relatively the algorithm of the target system.

These possibilities are due to concurrency of design synthesis and verification test bench generation, which provide a congruency of models at different design levels, which are the objects of comparison for design verification. Note that in contrast to formal verification, which do not address the problems of finding bugs during verification purgatory, the Abelite-based verification allows to localize possible errors (bugs) in target designs.

References

1. *Bening L., Foster H.* Principles of verifiable RTL design. — Kluwer Academic Publishers, 2001.
2. *Henzinger T. A., Liu X., Qadeer S., Rajamani S.* Formal specification and verification of a dataflow processor array // ICCAD99: IEEE/ACM International Conference on Computer-Aided Design, 1999. P. 494–499.
3. *Frenkel S. L.* Verification model structures for digital systems design // 17th European Simulation Multiconference ESM2003 Proceedings. Trent University, Nottingham, England. 9–11 June. 2003. P. 462–467.

4. *Seger C-J. H., Jones R. B., O'Leary J. W., Melham T. F., Aagaard M., Barret C.* An industrial effective environment for formal hardware verification // IEEE Transaction on Computer-Aided Design of Integrated Circuits and Systems, 2005. Vol. 24. No. 9. P. 1381–1405.
5. *Abielmona R.* Advanced Topic Lecture #3, ASM, Design High Level Computer Systems Design, SMRLab, University of Ottawa, CEG 3151, May 21, 2003.
6. *Campenhout D., Mudge T., John P.* Hayes, evaluation of design error models for verification testing of microprocessors // IEEE 1st International Workshop on Microprocessor Test and Verification. Washington DC, October 23. 1998.
7. *Buonanno G., Ferrandi F., Ferrandi L., Fummi F., Sciuto D.* How an "evolving" fault model improves the behavioral test generation // IEEE Seventh Great lakes Simposium on VLSI. March, 1997.
8. *Borrione D., Gascard E., Helmy A., Morin-Allory K., Oddos Y., Pierre L., Schmaltz J.* Multi-paradigm formal methods in the design flow. TIMA ANNUEL report. TIMA Lab., Grenoble, France, 2006.
9. *Jones R.B., Seger C.-J. H., Aagaard M.* Combining theorem proving and trajectory evaluation in an industrial environment // 35th Design Automation Conference (DAC 98) Proceedings. ACM Press, 1998. P. 538–541.
10. *Mir A. A., Balakrishnan S., Tahar S.* Modeling and verification of embedded systems using cadence SMV // 2000 Canadian Conference on Electrical and Computer Engineering Proceedings, 2000.
11. *Chockler H., Kupferman O., Kurshan R., Vardi M.* A practical approach to coverage in model checking // 13th International Conference CAV 2001 Proceedings. Paris, France, 2001. LNCS 2102. P. 66.
12. *Vemuri R., Kalyanaraman R.* Generation of design verification tests from behavioral VHDL programs using path enumeration and constraint programming // IEEE Trans. on VLSI, 1995. P. 201–214.
13. *Ho R. C., Yang C. H., Horowitz M. A., Dill D. L.* Architecture validation for processors // International Symposium Computer Architecture Proceedings, 1995. P. 404–413.
14. *Seger C-J.* An introduction to formal hardware verification 92-13. Technical Report 92-13. Department of Computer Science, University of Bruish Columbia, June 1992.
15. Introduction to HOL: A theorem proving environment for higher-order logic / Eds. Gordon M. J. C., Melham T. F. Cambridge Univ. Press, 1993.
16. *Burch J. R., Clarke E. M., Long D. E., McMillan K. L., Dill D. L.* Symbolic model checking for sequential circuit verification // IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 1994. Vol. 13. No. 4. P. 401–424.
17. *Kern C., Greenstreet M.* Formal verification in hardware design: A survey // ACM Transactions on Design Automation of E. Systems, 1999. Vol. 4. P. 123–193.
18. *McMillan K.* Getting started with SMV. User's Manual. Cadence Berkeley Laboratories, USA, 1998.
19. *Goldberg E., Gulati K., Khatri S.* Toggle equivalence preserving (TEP) logic synthesis // IWLS-2007. San Diego, 2007. http://eigold.tripod.com/papers/iwls-2007-tep.pdf.
20. Cadence Design System Products, 2007.
21. *Cortes L. A., Eles P., Peng Z.* Modeling and formal verification of embedded systems based on Petri Net representation // J. Systems Architecture, 2003. Vol. 49. No. 12–15. P. 571–598.
22. *Semenov A., Koelmans I., Yakovlev A.* Designing an asynchronous processor using Petri Nets // IEEE Micro, 1997. Vol. 17. No. 2. P. 54–65.
23. *Bhadra J., Abadir M., Ray S., Wang Li.* A survey of hybrid techniques for functional verification // IEEE Design & Test of Computers, 2007. P. 112–122.
24. Property specification language. Reference Manual, version 1.1. ACCELERA, 2004.
25. *Hoskote Y., Kam T., Ho Pei-Hsin, Zhao Xudong.* Coverage estimation for symbolic model checking // DAC'99, 1999. P. 300.
26. *Katz D., Geist D., Grumberg O.* Have I written enough properties — a method of comparison between specification and implementation // 10th CHARME Proceedings, 1999. LNCS 1703. P. 280–297.
27. *Baranov S.* Logic and system design of digital systems. — Tallinn: TUT Press, 2008.
28. *Mokkedem A., Hosabettu R., Gopalakrishnan G.* Formalization and proof of a solution to the PCI 2.1 bus transaction ordering problem. FMCAsD, 1998.

О ВЕРИФИКАЦИИ НА ЭТАПЕ СИНТЕЗА ЦИФРОВЫХ СИСТЕМ

С. Баранов[1], С. Л. Френкель[2], В. Синельников[3], В. Н. Захаров[4]

[1]Холонский технологический институт, Израиль, samary@012.net.il
[2]Институт проблем информатики Российской академии наук, slf-ipiran@mtu-net.ru
[3]Университет Бар-Илан, Израиль, samary@012.net.il
[4]Институт проблем информатики Российской академии наук, VZakharov@ipiran.ru

Аннотация: Описана новая методология учета требований верификации проектов цифровых систем в процессе их разработки, начиная с этапа алгоритмического описания. Методология основана на использовании модели Algorithmic State Machine (ASM) и алгоритмах их композиции и минимизации.

Данный подход предназначен для синтеза в виде управляющего автомата и подсистемы обработки данных (Data Path) цифровых систем любого типа и сложности (в том числе микропроцессоров с конвейером), микропрограммных автоматов, контроллеров протоколов и т. д.). В отличие от известных полуформальных подходов к верификации проектов, основанных на комбинации моделирования и формальной верификации различных частей проекта, рассматривается формализованная процедура, позволяющая получить тестбенчи для верификации всех элементов синтезируемой схемы непосредственно из исходного алгоритмического описания, с учетом особенностей реализации структуры Data Path и управляющего автомата.

Ключевые слова: проектирование цифровых систем; формальная верификация; конечные автоматы

ИНФОРМАТИКА И ЕЁ ПРИМЕНЕНИЯ, 2008. Т. 2. Вып. 3. С. 26–33

ФУНКЦИЯ СТОИМОСТИ РЕСУРСОВ В ЭКОНОМИЧЕСКОЙ МОДЕЛИ УПРАВЛЕНИЯ ГРИД

Я. М. Агаларов[1]

Аннотация: Рассматривается задача максимизации дохода владельца ресурсов локального узла грид с экономической моделью управления, в которой выделяемые внешнему пользователю ресурсы оплачиваются, а сумма платы зависит от спроса и предложения на ресурсы. В рассматриваемой модели очередь глобальных заданий формируется только в центре планирования ресурсов грид, в котором осуществляется поиск, выбор и резервирование требуемых ресурсов, и отправка задания на ресурсы происходит одновременно с их выбором и резервированием. Предлагается функция стоимости ресурсов, использование которой позволит их владельцу осуществить эффективное разделение ресурсов локального узла грид между глобальными и локальными заданиями. Приведены результаты аналитического исследования рассматриваемой задачи и сравнительного анализа предлагаемых решений с использованием компьютерного моделирования.

Ключевые слова: грид; модель распределения ресурсов; многопроцессорные задания; владелец ресурсов; марковский процесс; стратегия

1 Введение

Одной из эффективных моделей распределения ресурсов грид считается экономическая, где цена каждого ресурса определяется потребностями в нем пользователей и его доступностью [2, 1]. Главными субъектами в этой модели являются пользователи и владельцы ресурсов. Оба субъекта имеют собственные стратегии. Пользователи ресурсов применяют стратегии решения своих прикладных задач в зависимости от требуемого времени и наличного бюджета. Владельцы ресурсов используют стратегию получения наибольшей выгоды от вложенных средств.

Задача владельца — распределение принадлежащих ему вычислительных ресурсов (процессоров, оперативной и постоянной памяти, программ и данных) между глобальными и локальными заданиями. Жесткое разделение ресурсов вычислительных комплексов (ВК) между локальными и глобальными потоками приводит к фрагментации ресурсов и их простою [3]. Лучше, если обоим потокам доступны все ресурсы (часть ресурсов), но тогда необходим механизм, который бы, сохраняя права владельцев ресурсов, регулировал интенсивность поступления глобальных и локальных заданий.

В качестве такого механизма в экономической модели управления грид предлагается конкуренция локальных и глобальных заданий на основе сравнения их локальных приоритетов, назначаемых локальной системой управления ресурсами (ЛСУР). Вычисление локального приоритета глобального задания предлагается производить (например, как в [4, 5]) функцией, зависящей от потребляемых ресурсов и платы за задание. Конкретный вид зависимости устанавливается владельцем каждого ВК. В частности, внешний поток может быть перекрыт полностью. Однако остается вопрос о виде зависимости стоимости предоставления ресурсов от типа задания.

Ниже рассматривается модель грид с неотчуждаемыми ресурсами (совместно используемыми внешними пользователями и их владельцами), в которой реализована двухуровневая организация управления ресурсами (рис. 1) [2].

Основные функции верхнего (глобального) уровня — это формирование очереди глобальных заданий и в соответствии с приоритетами и запросами пользователей осуществление поиска и выделение требуемых ресурсов. К основным функциям нижнего (локального) уровня относятся: формирование ответа на запрос верхнего уровня, мониторинг состояния ВК, взаимодействие с владельцем ресурсов и, после того как задание доставлено на ВК, осуществление контроля над его выполнением.

Глобальное задание в рассматриваемой модели отправляется на ВК только после выбора ему верхним и нижним уровнями управления требуемых вычислительных ресурсов (процессоров). Выбор ресурсов осуществляется в соответствии с требованиями задания и согласно установленному в

[1]Институт проблем информатики Российской академии наук, yagalarov@ipiran.ru

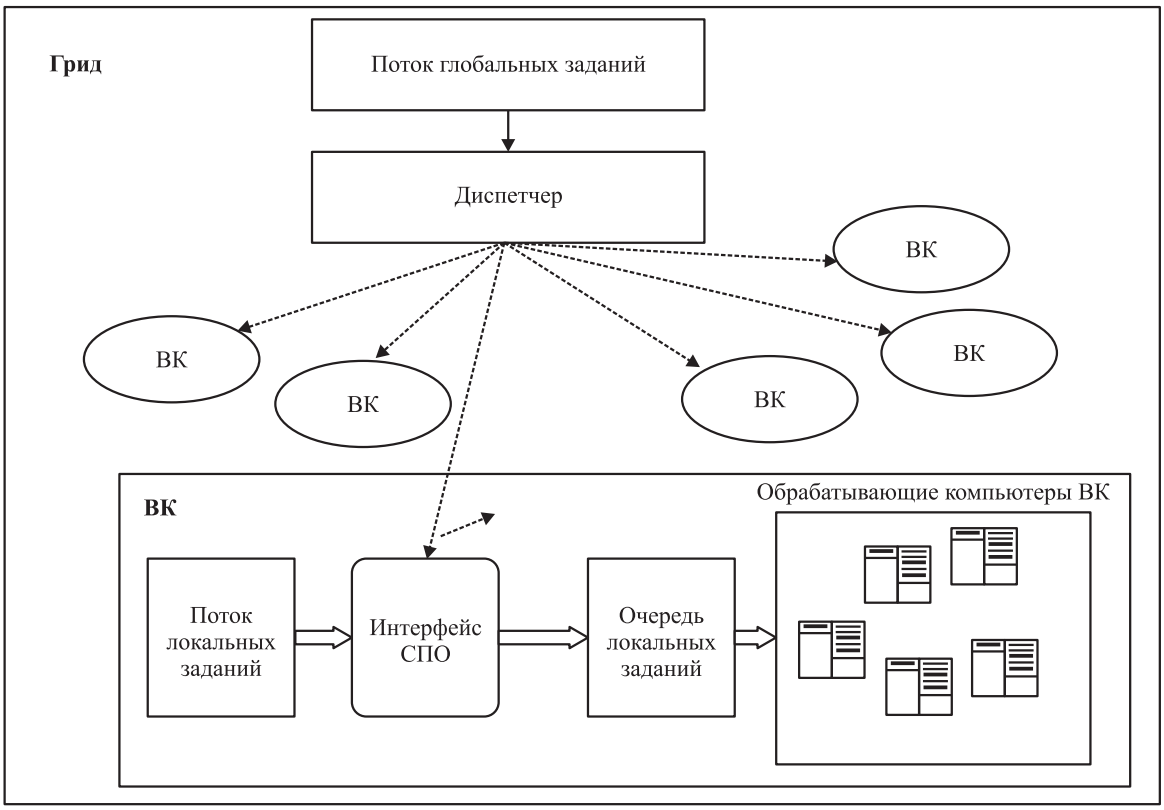

Рис. 1 Структурная схема грид

локальном узле соглашению между владельцем и пользователем (посредством ЛСУР). Глобальные задания могут быть как однопроцессорные, так и многопроцессорные, причем одновременно используемые многопроцессорными заданиями вычислительные ресурсы могут принадлежать разным ВК. Локальные задания в данной модели всегда однопроцессорные. Задания различаются по локальным приоритетам, которые учитываются при распределении ресурсов следующим образом. Доставленное на локальный уровень глобальное задание сразу же занимает все выделенные ему ресурсы ВК и после завершения выполнения освобождает все занятые ресурсы одновременно. Если в момент поступления глобального задания выделенные ему ресурсы заняты локальными заданиями, то последние освобождают эти ресурсы, переходят в состояние ожидания и продолжают выполнение после освобождения ресурсов. Приоритеты глобальных заданий по отношению друг к другу определяются величинами платы за требуемые ресурсы, определяемыми на верхнем уровне, и учитываются на стадии выделения ресурсов при формировании очереди на глобальном уровне. Локальное задание, в отличие от глобальных, при отсутствии свободного ресурса требуемого типа становится в очередь в накопителе ВК (при наличии вакантного места) и при появлении свободного ресурса требуемого типа занимает его и выполняется согласно выбранной в ВК дисциплине обслуживания. При отсутствии свободных мест в накопителе локальное задание теряется.

В рассматриваемой модели принято следующее соглашение между глобальными пользователями и владельцами ресурсов:

(1) если в момент поступления глобального задания накопитель (постоянная память) хотя бы у одного требуемого типа процессоров полностью занят (отсутствует требуемый объем свободной памяти) или нет в наличии необходимого числа свободных (или занятых локальными заданиями) процессоров требуемого типа, то оно не получает вычислительные ресурсы в данном ВК;

(2) если нет ограничений, указанных в п. 1, то глобальное задание получают процессоры данного ВК (в первую очередь, свободные) согласно выбранной владельцем ресурсов функции стоимости, если в данный момент времени плата пользователя не меньше значения функции стоимости, иначе получает отказ;

(3) если глобальное задание получают процессоры, занятые локальными заданиями, то они освобождаются, а занявшие их локальные задания занимают свободные места в соответствующем накопителе. После этого поступившая глобальная заявка занимает выделенные ей ресурсы, а выбывшее локальное задание дообслуживается первым же освободившимся процессором требуемого типа.

Предлагается функция стоимости предоставления ресурсов для глобальных и локальных заданий, увеличивающая доход владельца ВК. Данная функция определяет зависимость требуемой платы за выполнение задания от текущего состояния ресурсов. Предлагаемая функция стоимости может быть использована владельцем ресурсов ВК для выгодного разделения вычислительных ресурсов между внешними и внутренними пользователями.

2 Постановка задачи

В качестве модели ВК рассматривается система массового обслуживания (СМО) с I различными пуассоновскими потоками заявок (заданий), M различными типами приборов (ресурсов) и B_l приборами l-го типа, $l = \overline{1, M}$. Пусть потоки заявок пронумерованы числами $1, \ldots, I$, типы приборов числами $1, \ldots, M$, $I \geq M$.

Введены следующие предположения:

1. Интенсивности поступающих на СМО потоков равны $0 < \lambda_i < \infty$, $i = \overline{1, I}$. Для каждого потока с номером i задано множество типов приборов L_i, на котором он может обслуживаться. Заявка потока с номером i (i-заявка), где $i \in \overline{1, M}$ (локальные заявки), может занимать один любой свободный прибор i-го типа, т. е. $L_i = \{i\}$ для $i \in \overline{1, M}$. Заявки остальных потоков (с номерами $i \in \overline{M+1, I}$) (глобальные заявки) могут требовать для обслуживания одновременно несколько приборов, причем обязательно различного типа (т. е. множество L_i может состоять из более чем одного элемента).
2. Приборы l-го типа, $l \in \overline{1, M}$, имеют общий накопитель емкости A_l, который могут занимать только локальные l-заявки.
3. Если в момент поступления локальной i-заявки, $i \in \overline{1, M}$, в накопителе приборов i-го типа нет свободных мест, она сразу покидает узел, в противном случае при наличии свободного прибора соответствующего типа сразу поступает на обслуживание, а при отсутствии — занимает место в накопителе.
4. Если в момент поступления глобальной i-заявки, $i \in \overline{M+1, I}$, хотя бы у одного требуемого типа приборов в накопителе отсутствует свободное место или все приборы заняты глобальными заявками, она сразу покидает систему.
5. Если в момент поступления глобальной заявки нет ограничений, указанных в п. 4, то заявка допускается или не допускается на выполнение согласно выбранной стратегии распределения ресурсов, зависящей от текущего состояния системы (числа заявок каждого типа в системе).
6. Если глобальная заявка допускается в систему и все приборы некоторого требуемого типа заняты заявками, то одна из локальных заявок, занимающая требуемый прибор, освобождает его и занимает свободное место в соответствующем накопителе. Сразу после этого поступившая глобальная заявка занимает по одному свободному прибору требуемого типа, а каждая выбывшая заявка дообслуживается первым же освободившимся прибором требуемого типа.
7. Время обслуживания i-заявки (суммарное время занятия приборов i-заявкой) — экспоненциально распределенная случайная величина с параметром $0 < \mu < \infty$, независимая от других случайных событий в узле, $i = \overline{1, I}$.
8. Выполненная заявка освобождает одновременно все занятые ею приборы и покидает систему навсегда.
9. Известна плата за выполнение локальной i-заявки, $i = \overline{1, M}$.

Введем обозначения:

$d_i > 0$ — плата за выполнение i-заявки, $i = \overline{1, I}$;

k_i — число i-заявок в системе в некоторый момент времени, $i = \overline{1, I}$;

$\overline{k} = (k_1, \ldots, k_I)$ — вектор-столбец, описывающий состояние системы;

$\overline{b}_l^T = (b_{1l}, \ldots, b_{Il})$ — вектор-строка, где $b_{il} = 0$, если приборы l-го типа не требуются для выполнения i-заявки, и $b_{il} = 1$ в противном случае;

$m_l = \overline{b}_l^T \overline{k} = \sum\limits_{i=1}^{I} b_{il} k_i$ — суммарное число заявок в приборах l-го типа и в их очереди;

$N = \{\overline{k} : \ 0 \leq \overline{b}_l^T \overline{k} \leq B_l + A_l$ при $l = 1, \ldots, M$ и $0 \leq k_i \leq \min\limits_{l \in L_i}\{B_l\}$ при $i = M+1, \ldots, I\}$ — пространство состояний системы (множество всех возможных состояний системы);

$D_l = B_l - \sum\limits_{i=M+1}^{I} b_{il} k_i$ — число приборов l-го типа, не занятых глобальными заявками;

$N_i = \{\overline{k} : \overline{k} \in N,\ k_i < D_i + A_i\}$ — множество состояний, при которых в системе есть хотя бы один свободный прибор для i-заявок, $i \in (\overline{1,\ M})$;

$N_i = \{\overline{k} : \overline{k} \in N_l,\ D_l > 0$ для всех $l \in L_i\}$ — множество состояний, при которых в системе есть требуемое число приборов для i-заявок, $i \in (\overline{M+1,\ I})$;

$\overline{N}_i = \{\overline{k} : \overline{k} \in N,\ k_i = D_i + A_i\}$ — множество состояний, при которых в системе нет требуемого числа приборов для i-заявок, $i \in (\overline{1,\ M})$;

$\overline{N}_i = \{\overline{k} :\ D_l = 0$ или $\overline{k} \in \overline{N}_l$ хотя бы для одного $l \in L_i\}$ — множество состояний, при которых в системе нет требуемого числа приборов для i-заявок, $i \in (\overline{M+1,\ I})$;

$\chi(A)$ — функция-индикатор:

$$\chi(A) = \begin{cases} 1, \text{если выполняется условие } A\,, \\ 0, \text{если не выполняется условие } A\,. \end{cases} .$$

Пусть $\overline{r}(\overline{k}) = (r_1(\overline{k}), \ldots, r_I(\overline{k}))$, где каждая компонента $r(\overline{k})$ может принимать одно из значений: 0 или 1, если $\overline{k} \in N_i$, и 1, если $\overline{k} \in \overline{N}_i$. Политику распределения ресурсов ВК определим с помощью целочисленной функции $\overline{r}(\overline{k})$ следующим образом. Если в момент поступления i-заявки система находится в состоянии $\overline{k}$, то она принимается в систему в случае $r_i(\overline{k}) = 0$ и не принимается при $r_i(\overline{k}) = 1$, $i = 1, \ldots, I$. Набор $\overline{r} = \{\overline{r}(\overline{k}),\ \overline{k} \in N\}$ будем называть *стратегией распределения приборов (ресурсов)*. В дальнейшем будем рассматривать только стационарные стратегии (постоянные во времени) и под словом стратегия будем понимать стационарную стратегию. Обозначим через $g^{\overline{r}}$ среднее суммарного дохода (предполагаемой платы за выполнение заявок, включая и локальные), теряемого системой в единицу времени из-за занятости приборов или недопуска в систему заявок. Пусть R — множество всех возможных $\overline{r}$.

Ставится задача: найти стратегию $\overline{r}^* \in R$ такую, что при $\overline{r} = \overline{r}^*$ функция $g^{\overline{r}}$ достигает минимального значения, т. е.

$$\min_{\overline{r} \in R} g^{\overline{r}} = g^{\overline{r}^*} .$$

3 Решение задачи

Для решения задачи воспользуемся аппаратом теории марковских процессов принятия решений, а именно итерационным алгоритмом динамического программирования, называемым итерационным алгоритмом Ховарда [6, 7].

Отметим, что для каждой фиксированной стратегии $\overline{r} \in R$ процесс перехода рассматриваемой СМО из одного состояния в другое описывается марковским процессом [8]. Пусть $\lambda_i(\overline{k})$ — интенсивность поступающего в систему потока i-заявок в состоянии $\overline{k}$, $\mu(k_i)$ — интенсивность обслуженного потока i-заявок в состоянии $\overline{k}$ и $\overline{r}$ — фиксированная стратегия. Тогда из введенных предположений и определения стратегии $\overline{r}$ следует, что для любого $\overline{k} \in N$

$$\lambda_i(\overline{k}) = \begin{cases} 0, & \text{если } r_i(\overline{k}) = 1\,, \\ \lambda_i\,, & \text{если } r_i(\overline{k}) = 0\,, \end{cases}$$

$$\mu(k_i) = \begin{cases} \mu D_i\,, & \text{если } k_i > D_i\,,\ i \in (\overline{1,\ M})\,, \\ \mu k_i, & \text{если } k_i \le D_i \text{ и} \\ & i \in (\overline{1,\ M}) \text{ или } i \in (\overline{M+1,\ I})\,; \end{cases}$$
$$i = (\overline{1,\ I})\,.$$

Как видно, в рассматриваемом марковском процессе переход системы из одного состояния в другое может произойти только в двух случаях: при поступлении заявки или при окончании выполнения заявки. Матрица интенсивностей перехода $A^{\overline{r}} = (a^{\overline{r}}(\overline{k}, \overline{j}))_{(\overline{k},\overline{j}) \in N}$ данного марковского процесса имеет следующий вид. Интенсивность перехода из состояния $\overline{k}$ в $\overline{j}$, вызванная поступлением или окончанием выполнения i-заявки, равна:

$$a_i^{r_i}(\overline{k}, \overline{j}) = \begin{cases} \lambda_i & \text{при } \overline{k} \in N_i\,,\ \overline{j} = \overline{k} + 1_i\,, \\ & r_i(\overline{k}) = 0\,; \\ \mu(k_i) & \text{при } k_i \ge 1\,, \overline{j} = \overline{k} - 1_i\,, \\ & \overline{k} \in N\,; \\ 0 & \text{в остальных случаях} \end{cases} \qquad (1)$$

при $\overline{k} \ne \overline{j}$, где 1_i — вектор-столбец, у которого i-я компонента равна 1, а остальные равны 0, $i = (\overline{1,\ I})$, $\overline{k}, \overline{j} \in N$. Тогда по определению матрицы интенсивностей перехода $a^{\overline{r}}(\overline{k}, \overline{j}) = \sum\limits_{i=1}^{I} a_i^{r_i}(\overline{k}, \overline{j})$ при $\overline{k} \ne \overline{j}$ и $a^{\overline{r}}(\overline{k}, \overline{k}) = -\sum\limits_{\overline{j} \in N,\, \overline{j} \ne \overline{k}} a^{\overline{r}}(\overline{k}, \overline{j})$, $\overline{k}, \overline{j} \in N$.

Отметим, что рассматриваемый марковский процесс всегда имеет стационарные вероятности состояний (это следует из введенных предположений и типа рассматриваемой СМО — неприводимая марковская цепь и конечное число состояний) [8].

В дальнейшем всюду будем предполагать, что множество R включает только те стратегии, при которых рассматриваемый марковский процесс имеет только одно эргодическое множество состояний.

Определим начальную стратегию $\overline{r}^{\text{н}}$:

$$r_i^{\text{н}}(\overline{k}) = \begin{cases} 0 & \text{при } \overline{k} \in N_i\,,\ \sum\limits_{l=1}^{M} b_{il} = 1\,, \\ 1 & \text{при } \overline{k} \in \overline{N}_i \text{ или } \sum\limits_{l=1}^{M} b_{il} > 1\,, \end{cases}$$
$$i = (\overline{1,\ I})\,.$$

Заметим, что при начальной стратегии каждая допускаемая в систему заявка требует для выполнения только один прибор, а все заявки, требующие одновременно более одного прибора, не допускаются в систему.

Из (1) следует, что при $\overline{r} = \overline{r}^{\text{н}}$ интенсивности $a_i^{r_i}(\overline{k}, \overline{j})$ имеют вид:

$$a_i^{r_i}(\overline{k}, \overline{j}) = \begin{cases} \lambda_i & \text{при } \overline{k} \in N_i\,,\ \overline{j} = \overline{k} + 1_i\,, \\ & r_i(\overline{k}) = 0\,,\ 1 \le i \le M\,; \\ \mu(k_i) & \text{при } k_i \ge 1\,, \\ & \overline{j} = \overline{k} - 1_i\,,\ \overline{k} \in N\,; \\ 0 & \text{в остальных случаях}\,; \end{cases} \tag{2}$$

$$\overline{k} \ne \overline{j}\,;\ \overline{k}, \overline{j} \in N\,;\ i = (\overline{1, I})\,.$$

Обозначим через $q_{\overline{k}}^{\overline{r}}$ норму стоимостных потерь в состоянии $\overline{k}$ (теряемый в единицу времени доход, когда система находится в состоянии $\overline{k}$) при стратегии $\overline{r}$, через $g^{\overline{r}}$ — средние стационарные стоимостные потери в единицу времени. Далее для краткости вместо слов стоимостные потери будем писать просто потери. Заметим, что при $\overline{r}^{\text{н}}$ каждые i-й тип приборов и i-й поток заявок образуют СМО типа $M/M/B_i/A_i$ с ограниченной очередью. Следовательно, имеем:

$$q_{\overline{k}}^{\overline{r}} = \sum_{i:\overline{k} \in \overline{N}_i} d_i\lambda_i + \sum_{\substack{i:\overline{k} \in N_i \\ r_i(\overline{k})=1}} d_i\lambda_i\,;$$

$$q_{\overline{k}}^{\overline{r}^{\text{н}}} = \sum_{i=M+1}^{I} d_i\lambda_i + \sum_{\substack{i:k_i=D_i+A_i, \\ i \in (\overline{1, M})}} d_i\lambda_i\,; \tag{3}$$

$$g^{\overline{r}^{\text{н}}} = \sum_{i=M+1}^{I} d_i\lambda_i + \sum_{i=1}^{M} d_i\lambda_i \pi_{B_i+A_i}^{\overline{r}^{\text{н}}}(\rho_i)\,, \quad \overline{k} \in N\,,$$

где $\pi_{B_i+A_i}^{\overline{r}''}(\rho_i)$ — вторая формула Эрланга для системы с B_i приборами, A_i местами в накопителе и нагрузкой ρ_i, $i = 1, \ldots, M$.

Согласно итерационной процедуре Ховарда [6, 7] для улучшения стратегии достаточно найти решение $\overline{r}(\overline{k})$ такое, чтобы хотя бы в одном состоянии $\overline{k}$ выполнялось условие:

$$q_{\overline{k}}^{\overline{r}} + \sum_{\overline{j} \in N} a^{\overline{r}}(\overline{k}, \overline{j}) V_{\overline{j}} < g^{\overline{r}^{\text{н}}}\,, \tag{4}$$

где $g^{\overline{r}^{\text{н}}}$ и $V_{\overline{j}}$ ($\overline{j} \in N$) являются решением системы уравнений:

$$g^{\overline{r}^{\text{н}}} = q_{\overline{k}}^{\overline{r}^{\text{н}}} + \sum_{\overline{j} \in N} a^{\overline{r}^{\text{н}}}(\overline{k}, \overline{j}) V_{\overline{j}} \tag{5}$$

при $\overline{V}_{\overline{0}} = 0$, $\overline{0} = (0, \ldots, 0)$ — вектор-столбец, соответствующий нулевому состоянию системы.

Кроме того, если $\overline{r}'$ — улучшенная стратегия, то:

$$g^{\overline{r}'} - g^{\overline{r}^{\text{н}}} = \sum_{\overline{k} \in N} \pi_{\overline{k}}^{\overline{r}'} \gamma_{\overline{k}}\,,$$

где $\gamma_{\overline{k}} = q_{\overline{k}}^{\overline{r}'} + \sum\limits_{\overline{j} \in N} a^{\overline{r}'}(\overline{k}, \overline{j}) V_{\overline{j}} - q_{\overline{k}}^{\overline{r}^{\text{н}}} - \sum\limits_{\overline{j} \in N} a^{\overline{r}^{\text{н}}}(\overline{k}, \overline{j}) V_{\overline{j}}$.

Найдем улучшенную стратегию. Обозначим для краткости $\pi_i = \pi_{B_i+A_i}^{\overline{r}^{\text{н}}}(\rho_i)$. Для начальной стратегии $\overline{r}^{\text{н}}$ из (5), подставив (2) и (3), получим:

$$\sum_{i=1}^{M} d_i\lambda_i\pi_i + \sum_{i=M+1}^{I} d_i\lambda_i = \sum_{i=M+1}^{I} d_i\lambda_i + \sum_{\substack{i:m_i=D_i+A_i \\ i \in (\overline{1, M})}} d_i\lambda_i + \sum_{\substack{i:m_i<D_i+A_i \\ i \in (\overline{1, M})}} \lambda_i V_{\overline{k}-1_i} + \sum_{i:k_i \ge 1} \mu(k_i) V_{\overline{k}-1_i} - \left(\sum_{\substack{i:m_i<D_i+A_i \\ i \in (\overline{1, M})}} \lambda_i + \sum_{i:k_i \ge 1} \mu(k_i) \right) V_{\overline{k}}\,. \tag{6}$$

Система уравнений (6) эквивалентна системе уравнений:

$$\sum_{i=1}^{M} d_i\lambda_i\pi_i = \sum_{\substack{i:m_i=D_i+A_i \\ i \in (\overline{1, M})}} d_i\lambda_i + \sum_{\substack{i:m_i<D_i+A_i \\ i \in (\overline{1, M})}} \lambda_i V_{\overline{k}+1_i} + \sum_{\substack{i:k_i \ge 1 \\ i \in (\overline{1, M})}} \mu(k_i) V_{\overline{k}-1_i} - \sum_{\substack{i:k_i \ge 1 \\ i \in (\overline{M+1, I})}} \mu(k_i) V_{\overline{k}-1_i} - \left(\sum_{\substack{i:m_i<D_i+A_i \\ i \in (\overline{1, M})}} \lambda_i + \sum_{\substack{i:k_i \ge 1 \\ i \in (\overline{1, M})}} \mu(k_i) + \sum_{\substack{i:k_i \ge 1 \\ i \in (\overline{M+1, I})}} \mu(k_i) \right) V_{\overline{k}}\,, \quad \overline{k} \in N\,. \tag{7}$$

Утверждение. Система уравнений (7) имеет решение вида:

$$V_{\overline{k}} = \sum_{i=1}^{M} v_{m_i}\,;$$

$$v_{m_i} = v_{m_i-1} + u_{m_i}\,;$$

$$u_{m_i} = \frac{\mu_{m_i-1}}{\lambda_i}\, u_{m_i-1} + d_i\pi_i\,;$$

$$v_{m_i} = u_{m_i} = 0 \ \text{ при } m_i = 0\,;$$

$$\mu_{m_i} = \begin{cases} \mu m_i\,, & \text{если } m_i < B_i\,, \\ \mu B_i\,, & \text{если } m_i \geq B_i\,; \end{cases} \tag{8}$$

$$\overline{k} \in N\,, \quad m_i = 1, \ldots, B_i + A_i\,, \quad i = 1, \ldots, M\,.$$

Следствие. Справедливы равенства:

$$u_{m_i} = \frac{d_i \pi^{\overline{r}^{\text{н}}}_{B_i+A_i}(\rho_i)}{\pi^{\overline{r}^{\text{н}}}_{m_i-1}(\rho_i)}\,, \tag{9}$$

где $\pi_{m_i}(\rho_i)$ — первая формула Эрланга с m_i приборами и нагрузкой ρ_i в случае $m_i < B_i$ и вторая формула Эрланга для системы с B_i приборами, $m_i - B_i$ местами в накопителе и нагрузкой ρ_i в случае $m_i \geq B_i$, $m_i = 1, \ldots, B_i + A_i$, $i = 1, \ldots, M$.

Рассмотрим левую часть неравенства (4) для некоторого фиксированного состояния $\overline{k}$. Подставив (1) и (3) в левую часть (4), получим:

$$q^{\overline{r}}_{\overline{k}} + \sum_{\overline{j} \in N} a^{\overline{r}}(\overline{k}, \overline{j}) V_{\overline{j}} = \sum_{i:\overline{k} \in \overline{N}_i} d_i \lambda_i + \sum_{\substack{i:\overline{k} \in N_i \\ r_i(\overline{k})=1}} d_i \lambda_i +$$

$$+ \sum_{\substack{i:\overline{k} \in N_i \\ r_i(\overline{k})=0}} \lambda_i V_{\overline{k}+1_i} + \sum_{i:k_i \geq 1} \mu(k_i) V_{\overline{k}-1_i} -$$

$$- \left(\sum_{\substack{i:\overline{k} \in N_i \\ r_i(\overline{k})=0}} \lambda_i + \sum_{i:k_i \geq 1} \mu(k_i) \right) V_{\overline{k}} =$$

$$= \sum_{i:\overline{k} \in \overline{N}_i} d_i \lambda_i + \sum_{\substack{i:\overline{k} \in N_i \\ r_i(\overline{k})=1}} d_i \lambda_i + \sum_{\substack{i:\overline{k} \in N_i \\ r_i(\overline{k})=0}} d_i \lambda_i - \sum_{\substack{i:\overline{k} \in N_i \\ r_i(\overline{k})=0}} d_i \lambda_i +$$

$$+ \sum_{\substack{i:\overline{k} \in N_i \\ r_i(\overline{k})=0}} \lambda_i (V_{\overline{k}+1_i} - V_{\overline{k}}) + \sum_{i:k_i \geq 1} \mu(k_i)(V_{\overline{k}-1_i} - V_{\overline{k}}) =$$

$$= \sum_{i \in (\overline{1,I})} d_i \lambda_i - \sum_{\substack{i:\overline{k} \in N_i \\ r_i(\overline{k})=0}} d_i \lambda_i + \sum_{\substack{i:\overline{k} \in N_i \\ r_i(\overline{k})=0}} \lambda_i (V_{\overline{k}+1_i} - V_{\overline{k}}) +$$

$$+ \sum_{i:k_i \geq 1} \mu(k_i)(V_{\overline{k}-1_i} - V_{\overline{k}}) =$$

$$= \sum_{i \in (\overline{1,I})} d_i \lambda_i + \sum_{\substack{i:\overline{k} \in N_i \\ r_i(\overline{k})=0}} \lambda_i (V_{\overline{k}+1_i} - V_{\overline{k}} - d_i) +$$

$$+ \sum_{i:k_i \geq 1} \mu(k_i)(V_{\overline{k}-1_i} - V_{\overline{k}})\,. \tag{10}$$

Найдем стратегию, при которой (10), т. е. левая часть (4), достигает минимума. Заметим, что в (10) все суммы, кроме второй, не зависят от стратегии. Следовательно, стратегия, на которой достигает минимума вторая сумма, т. е. выражение:

$$\sum_{\substack{i:\overline{k} \in N_i \\ r_i(\overline{k})=0}} \lambda_i (V_{\overline{k}+1_i} - V_{\overline{k}} - d_i)\,, \tag{11}$$

и будет искомой. Очевидно, что выражение (11) принимает минимальное значение при стратегии $\overline{r}$ такой, что:

$$r_i(\overline{k}) = \begin{cases} 0\,, & \text{если } V_{\overline{k}+1_i} - V_{\overline{k}} < d_i\,, \\ 1\,, & \text{если } V_{\overline{k}+1_i} - V_{\overline{k}} \geq d_i\,, \end{cases}$$

$$\overline{k} \in N_i\,, \quad i = 1, \ldots, I\,.$$

Как следует из (9), $u_{m_i} < d_i$, $m_i = 1, \ldots, B_i + A_i$, $i = 1, \ldots, M$. Следовательно, выражение (11) достигает минимума на стратегии $\overline{r}$ такой, что

$$r_i(\overline{k}) = 0\,, \quad \overline{k} \in N_i\,, \quad i = 1, \ldots, M\,,$$

$$r_i(\overline{k}) = 0\,, \quad \overline{k} \in N_i\,, \quad i = 1, \ldots, M\,,$$

$$r_i(\overline{k}) = \begin{cases} 0\,, & \text{если } d_i(\overline{k}) < d_i\,, \\ 1\,, & \text{если } d_i(\overline{k}) \geq d_i\,, \end{cases}$$

$$\overline{k} \in N_i\,, \quad i = M+1, \ldots, I\,,$$

где $d_i(\overline{k}) = \sum\limits_{l:l \in L_i} u_{m_l+1}$ — функции стоимости предоставления приборов для i-заявки.

Согласно этой стратегии:

- локальная заявка при наличии свободного места в накопителе приборов требуемого типа допускается в систему;
- глобальная заявка при наличии требуемых приборов допускается в систему, если доход за обслуживание пользователя выше значения соответствующей функции стоимости (величины $d_i(\overline{k})$), в противном случае получает отказ.

Ниже приведены результаты вычислительного эксперимента, полученные с использованием имитационной модели описанной выше системы для сравнительного анализа эффективности следующих схем доступа к ресурсам:

- NEDOP — использует функцию стоимости вида

$$d_i(\overline{k}) = \begin{cases} 0\,, & \text{если } i = 1, \ldots, M\,, \\ \infty, & \text{если } i = M+1, \ldots, I \end{cases}$$

для всех $\overline{k} \in N$ (глобальные заявки не допускаются в систему — стратегия $\overline{r}^{\text{н}}$);

- PRIOR — использует функцию стоимости вида

$$d_i(\overline{k}) = 0\,, \quad i = 1, \ldots, I, \quad \overline{k} \in N$$

(все заявки допускаются в систему);

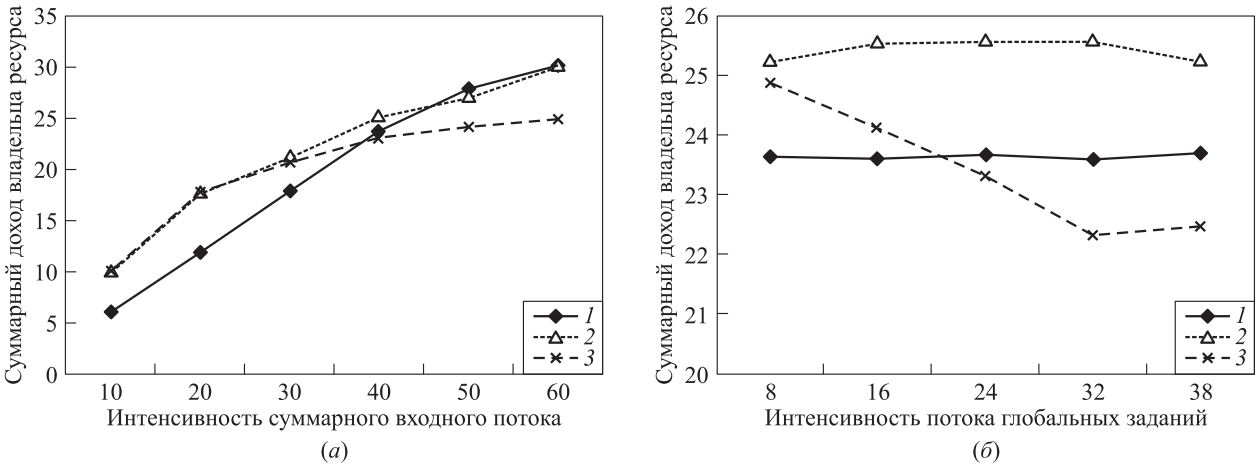

Рис. 2 Зависимость суммарного дохода владельца ресурсов от суммарной входной нагрузки (*а*) и интенсивности потока глобальных заданий (*б*): *1* — NEDOP; *2* — DINPR; *3* — PRIOR

– DINPR — использует функцию стоимости вида

$$d_i(\overline{k}) = \sum_{l:l\in L_i} u_{m_l+1}\,,$$

где u_{m_l+1}, $l = 1,\ldots,M$, вычисляются по формуле (9) (глобальные заявки допускаются в систему в зависимости от состояния системы — стратегия $\overline{r}'$).

Была рассмотрена система со следующими параметрами: $M = 3$; $I = 6$; $B_i = 10$, $A_i = 10$ при $i = 1, 2, 3$; $\mu = 1$; $d_i = 1$, $i = 1,\ldots,6$; $L_1 = 1$; $L_2 = 2$; $L_3 = 3$, $L_4 = \{1, 2\}$; $L_5 = \{2, 3\}$, $L_6 = \{1, 2, 3\}$. На рис. 2, *а* приведен график зависимости дохода владельца ВК от параметра λ — интенсивности суммарного потока заявок для указанных выше схем распределения ресурсов при $p_1 = p_2 = p_3 = 0{,}2$; $p_4 = p_5 = 0{,}1$; $p_6 = 0{,}2$, где p_i — доля потока i-го типа, $i = 1, \ldots, 6$; $\lambda = 10, 20, 30, 40, 50, 60$. На рис. 2, *б* показана зависимость дохода владельца ресурсов от величины нагрузки глобальных заданий при фиксированных значениях локальных нагрузок $\lambda_1 = \lambda_2 = \lambda_3 = 8$.

4 Заключение

Полученные результаты приводят к следующим выводам:

1. Предоставление ресурсов глобальным заданиям (схема DINPR) может существенно повысить величину дохода владельца ресурсов (согласно вычислительным экспериментам при сильно загруженных локальными заданиями ресурсах — до 10%).
2. Бесконтрольный доступ многопроцессорных заданий (схема PRIOR) может существенно снизить доход владельца ресурсов (согласно вычислительным экспериментам при сильно загруженных локальными заданиями ресурсах — до 6%).
3. Схема DINPR эффективнее схемы NEDOP (не допускающей в систему глобальные задания), если существует хотя бы один тип таких заданий, для которого плата за выполнение выше значения соответствующей функции стоимости (т. е. $d_i(\overline{k}) < d_i$) хотя бы в одном состоянии.
4. Использование адаптивной (зависящей от состояния ресурсов) функции стоимости (схема DINPR) позволяет эффективно регулировать доступ глобальных заданий к ресурсам ВК (при сильной нагрузке ресурсов локальными заданиями увеличение нагрузки глобальных заданий, требующих ресурсов данного ВК, не приводит к снижению дохода владельца ресурсов).

Литература

1. *Buyya R., Giddy J., Abramson D.* An economy grid architecture for service-oriented grid computing // 10th IEEE International Heterogeneous Computing Workshop (HCW 2001). In conjunction with IPDPS 2001. San Francisco, USA, April, 2001. http://citeseer.ist.psu.edu/buyya01economy.html.
2. *Коваленко В. Н., Коваленко Е. И., Корягин Д. А., Любимский Э. З.* Основные положения метода опережающего планирования для грид вычислительного типа // Вестник СамГу. Естественно-научная сер. «Информационно-вычислительные системы», 2006. № 4(44). С. 238–264. http://vestnik.ssu.samara.ru/est/2006web4/ivs/200642001.pdf.

3. *Snell Q., Clement M., Jackson D., Gregory Ch*. The performance impact of advance reservation meta-scheduling // Computer Science Department Brigham Young University Provo, Utah, 2000. http://supercluster.org/research/papers/ipdps2000.pdf.

4. *Коваленко В. Н., Семячкин Д. А.* Использование алгоритма Backfill в грид // Тр. Международной конференции «Распределенные вычисления и Грид-технологии в науке и образовании». Дубна, 29 июня – 2 июля 2004 г. Дубна: 11-2004-205, ОИЯИ, 2004. С. 139–144.

5. *Агаларов Я. М.* Динамическая стратегия распределения вычислительных ресурсов локального узла GRID // Системы и средства информатики. — М.: Наука, 2007. Вып. 17. С. 17–29.

6. *Ховард Р.* Динамическое программирование и марковские процессы. — М.: Советское радио, 1964. 158 с.

7. *Майн Х., Осаки С.* Марковские процессы принятия решений. — М.: Наука, 1977. 176 с.

8. *Карлин С.* Основы теории случайных процессов. — М.: Мир, 1971. 536 с.

ИНФОРМАТИКА И ЕЁ ПРИМЕНЕНИЯ, 2008. Т. 2. Вып. 3. С. 34–40

МНОГОЛИНЕЙНАЯ СИСТЕМА МАССОВОГО ОБСЛУЖИВАНИЯ С КОНЕЧНЫМ НАКОПИТЕЛЕМ, БЛОКИРОВКОЙ ПОЛУМАРКОВСКОГО ПОТОКА ЗАЯВОК И ВЫБИВАНИЕМ ЗАЯВОК ИЗ НАКОПИТЕЛЯ*

В. В. Чаплыгин[1]

Аннотация: Рассматривается многолинейная система массового обслуживания (СМО) с конечным накопителем, блокировкой полумарковского потока заявок и выбиванием заявок из накопителя первой заявкой, поступившей в систему на периоде времени, когда поток разблокирован. Периоды блокировки входящего потока и периоды, когда входящий поток разблокирован, распределены по экспоненциальному закону с разными интенсивностями. Найдены основные стационарные характеристики системы: распределение очереди, вероятность потери заявки, среднее время пребывания заявки в системе.

Ключевые слова: система массового обслуживания; полумарковский поток заявок; выбивание заявок

1 Описание системы

В последние годы построению систем распределенных вычислений как одному из перспективных направлений развития современных инфотелекоммуникационных систем уделяется большое внимание. Нередки случаи, когда доступ к ресурсам, выделенным системе распределенных вычислений, оказывается ограниченным: например, когда обслуживающие устройства, на которых функционирует система, задействованы для выполнения других задач. Организация доступа к этим ресурсам может быть самой разнообразной: приоритетный доступ, интервальный доступ, разделение ресурса между пользователями и др. В частности, настоящая работа посвящена модели с интервальным доступом к нескольким обслуживающим устройствам и является развитием работы [1], в которой рассмотрена СМО SM/PH/n/r с блокировкой полумарковского потока заявок. Методы, используемые для отыскания стационарных характеристик рассмотренной ниже системы с блокировкой полумарковского потока заявок и выбиванием заявок из накопителя, опираются на идеи, заложенные в работах [2–4] для СМО с рекуррентным и полумарковским входящими потоками и марковским обслуживанием.

Рассмотрим многолинейную СМО с накопителем конечной емкости. В системе имеется n работающих независимо друг от друга идентичных приборов, которые обслуживают поступающие на них однотипные заявки.

Каждый из приборов может находиться на одной из J, $1 \le J < \infty$, фаз обслуживания. Время обслуживания заявки на каждом приборе распределено по закону фазового типа с параметрами $\boldsymbol{h}$ и H, где $\boldsymbol{h}$ — вектор-строка размерности J, а H — квадратная матрица порядка J. Функция распределения фазового типа времени обслуживания заявки записывается в виде (ниже через $\mathbf{1}$ будем обозначать вектор-столбец из единиц, через $\mathbf{0}$ — нулевую вектор-строку, через O — нулевую матрицу, а через E — единичную матрицу, размерность и порядок которых определяются нижним индексом или из контекста):

$$H(x) = 1 - \boldsymbol{h}\, e^{Hx}\mathbf{1}\,. \qquad (1)$$

Если в систему поступает заявка, но все n приборов заняты, то эта заявка поступает в накопитель емкостью r. Будем считать, что $r \ge 2$; если $r = 0$ или 1, то часть полученных формул потребует упрощения. Если накопитель полон, то заявка, не обслуживаясь, покидает систему (теряется). Заявки из накопителя обслуживаются в порядке их поступления в систему. Обозначим: $R = n + r$.

Опишем входящий в систему поток заявок.

Рассмотрим полумарковским процесс с конечным множеством состояний $\{1, 2, \ldots, I\}$, $1 \le I < \infty$. В каждый момент изменения состояния полумарковского процесса генерируется новая заявка,

*Работа выполнена при поддержке Российского фонда фундаментальных исследований, гранты 06-07-89056 и 08-07-00152.

[1]Институт проблем информатики Российской академии наук, vchaplygin@ipiran.ru

которая готова поступить в систему на обслуживание. Вероятность того, что полумарковский процесс за время меньше x перейдет из состояния i сразу в состояние j, $i, j = \overline{1, I}$, равна $A_{ij}(x)$. Среднее время между изменениями состояний полумарковского процесса в стационарном режиме можно записать в виде

$$a = \boldsymbol{\pi}_a \int_0^\infty x \, dA(x) \, \mathbf{1} \,, \qquad (2)$$

где $\boldsymbol{\pi}_a$ — вектор-строка стационарных вероятностей вложенной цепи Маркова полумарковского процесса, $A(x)$ — матрица из элементов $A_{ij}(x)$.

Более подробное описание полумарковского потока, а также некоторые естественные дополнительные предположения относительно его параметров, которые будем считать выполненными, можно найти в работе [4].

Будем рассматривать стационарный режим функционирования полумарковского процесса генерации заявок. За «малое» время Δ с вероятностью $\alpha\Delta + o(\Delta)$ происходит блокировка потока заявок, поступающих в систему, а именно с этого момента заявки, генерируемые полумарковским процессом, в систему не попадают, а теряются. Заявки, находящиеся в системе, систему не покидают, а продолжают обслуживаться (заявки на приборах) или ожидать обслуживания (заявки в накопителе) до момента поступления первой заявки после разблокировки потока.

Если поток заблокирован, то за «малое» время Δ с вероятностью $\beta\Delta + o(\Delta)$ поток разблокируется и заявки, которые будут сгенерированы после этого момента, вновь будут поступать в систему на обслуживание. Причем первая заявка, поступившая в период времени, когда поток разблокирован, выбивает все заявки из накопителя, если таковые имеются, а сама занимает первое место в очереди. Заявки на приборах продолжают обслуживаться. Первая заявка, поступившая в период, когда входящий поток разблокирован, и заставшая полный накопитель, не выбивает заявки из накопителя и сама покидает систему, не обслужившись. Заявки, поступающие в систему до момента блокировки процесса генерации, также заявок из накопителя не выбивают, а встают в очередь, если есть свободные места.

Генерация заявок полумарковским потоком не зависит от того, заблокировано поступление заявок в систему или нет.

В настоящей работе на основе методов, подробно изложенных в работах [2, 4], получены математические соотношения для расчета стационарных характеристик системы.

2 Стационарное распределение числа заявок в системе

Воспользуемся некоторыми построениями, полученными для многолинейной СМО с конечным накопителем, полумарковским потоком заявок и марковским обслуживанием [4], а именно тем, что процесс обслуживания всеми приборами системы, обслуживание на каждом из которых распределено по закону фазового типа, может быть описан в виде марковского процесса обслуживания следующим образом.

Если в системе находится k, $0 \le k \le n + r$, заявок, то процесс обслуживания может находиться в одном из l_k, $l_k < \infty$, состояний (фаз обслуживания), причем интенсивность смены фаз марковского процесса определяется элементами матриц Λ_k, $k = \overline{0, n+r}$, если ни одна заявка не обслужилась, и элементами матриц N_k, $k = \overline{1, n+r}$, если заявка обслужилась. Предполагается, что $l_k = l$ при $k = \overline{n, n+r}$, $\Lambda_k = \Lambda$ при $k = \overline{n, n+r}$, а $N_k = N$ при $k = \overline{n+1, n+r}$. Матрицу $\Lambda + N$ будем предполагать неразложимой, а матрицу N — ненулевой.

Если в системе находится k, $k = \overline{0, n-1}$, заявок, будем предполагать, что при поступлении новой заявки в систему марковский процесс обслуживания переходит на фазу, которая определяется элементами матриц Ω_k.

Более подробное описание структуры такого марковского процесса, а также способ формирования его инфинитезимальной матрицы по заданному PH-распределению обслуживания заявки на каждом приборе можно найти в работах [2, 4].

Рассмотрим вложенную цепь Маркова, определяемую моментами смены фаз полумарковского процесса генерации заявок.

Обозначим через p_{ik}^m, $m = 0, 1$, $i = l_k(u-1) + v$, $u = \overline{1, I}$, $v = \overline{1, l_k}$, $k = \overline{0, R}$, стационарную вероятность того, что сразу после смены фаз полумарковского процесса в системе находится k заявок, фаза полумарковского процесса генерации заявок находится на фазе u и марковский процесс обслуживания находится на фазе v и если $m = 0$, то поток заявок заблокирован, а если $m = 1$, то поток заявок разблокирован. Положим $\boldsymbol{p}_k^m = (p_{1k}^m, \ldots, p_{Il_k,k}^m)$, $\boldsymbol{p}_k = (\boldsymbol{p}_k^0, \boldsymbol{p}_k^1)$, $m = 0, 1$, $k = \overline{0, R}$, $\boldsymbol{p} = (\boldsymbol{p}_0, \ldots, \boldsymbol{p}_R)$.

Для вектора $\boldsymbol{p}$ справедлива система уравнений равновесия (СУР):

$$\boldsymbol{p} = \boldsymbol{p} P \,, \qquad (3)$$

в которой матрица P переходных вероятностей вложенной цепи Маркова представима в блочном виде:

$$P = \begin{pmatrix} P_{00} & P_{01} & O & O & \dots & O & O \\ P_{10} & P_{11} & P_{12} & O & \dots & O & O \\ \vdots & \vdots & \vdots & \vdots & \dots & O & O \\ P_{R-1,0} & P_{R-1,1} & P_{R-1,2} & P_{R-1,3} & \dots & P_{R-1,R-1} & P_{R-1,R} \\ P_{R0} & P_{R1} & P_{R2} & P_{R3} & \dots & P_{R,R-1} & P_{RR} \end{pmatrix},$$

где $P_{i,j}$, $i = \overline{0,R}$, $j = \overline{0,\min\{i+1,R\}}$, — матрицы размера $l_i \times l_j$.

Чтобы получить математические соотношения для матриц P_{ij}, введем некоторые дополнительные обозначения.

Обозначим через $q_{ij}(x)$, $i,j = 0,1$, вероятность того, что через время x поступление заявок будет заблокировано, если $j = 0$, и поступление заявок будет разблокировано, если $j = 1$, при условии, что в начальный момент времени поступление заявок заблокировано, если $i = 0$, и разблокировано, если $i = 1$.

Можно показать [1], что

$$q_{00}(x) = \frac{\alpha}{\alpha+\beta} + \frac{\beta}{\alpha+\beta}\, e^{-(\alpha+\beta)x};$$

$$q_{01}(x) = \frac{\beta}{\alpha+\beta} - \frac{\beta}{\alpha+\beta}\, e^{-(\alpha+\beta)x};$$

$$q_{10}(x) = \frac{\alpha}{\alpha+\beta} - \frac{\alpha}{\alpha+\beta}\, e^{-(\alpha+\beta)x};$$

$$q_{11}(x) = \frac{\beta}{\alpha+\beta} + \frac{\alpha}{\alpha+\beta}\, e^{-(\alpha+\beta)x}.$$

Обозначим через $\tilde{q}_1(x)$ вероятность того, что за время x состояние блокировки входящего потока заявок ни разу не поменяется при условии, что в начальный момент времени поток заявок был разблокирован. Через $\tilde{q}_{11}(x)$ обозначим вероятность того, что за время x состояние блокировки поменяется хотя бы раз, но к моменту x останется разблокированным при условии, что в начальный момент времени поток заявок был разблокирован. Для функций $\tilde{q}_1(x)$ и $\tilde{q}_{11}(x)$ нетрудно получить следующие выражения:

$$\tilde{q}_1(x) = e^{-\alpha x};$$

$$\tilde{q}_{11}(x) = \frac{\beta}{\alpha+\beta} + \frac{\alpha}{\alpha+\beta} e^{-(\alpha+\beta)x} - e^{-\alpha x}.$$

Очевидно, что

$$\tilde{q}_1(x) + \tilde{q}_{11}(x) = q_{11}(x).$$

Определим следующие вспомогательные матрицы (здесь $\otimes$ — символ кронекерова произведения матриц):

$$A_k^{ij} = \int_0^\infty q_{ij}(x)\, dA(x) \otimes F_k(x),$$

$$k \geq 0,\ i,j = 0,1; \qquad (4)$$

$$\tilde{A}_k^{11} = \int_0^\infty \tilde{q}_{11}(x)\, dA(x) \otimes F_k(x),\ \ k \geq 0;$$

$$\tilde{A}_k^{1} = \int_0^\infty \tilde{q}_{1}(x)\, dA(x) \otimes F_k(x),\ \ k \geq 0;$$

$$A_{kw}^{i1} = \int_0^\infty q_{i1}(x)\, dA(x) \otimes (F_{k,w}(x)\,\Omega_w),$$

$$w = \overline{0,n},\ \ k \geq w,\ \ i = 0,1;$$

$$\tilde{A}_{kw}^{i1} = \int_0^\infty q_{i1}(x)\, dA(x) \otimes F_{k,w}(x),$$

$$w = \overline{0,n},\ \ k \geq w,\ \ i = 0,1; \qquad (5)$$

$$A_{kw}^{i0} = \int_0^\infty q_{i0}(x)\, dA(x) \otimes F_{k,w}(x),$$

$$w = \overline{0,n},\ \ k \geq w,\ \ i = 0,1, \qquad (6)$$

где матрицы $F_k(x)$, $k \geq 0$, определяются соотношениями

$$F_0(x) = e^{\Lambda x}; \qquad (7)$$

$$F_k(x) = \int_0^x F_{k-1}(y)\, N\, F_0(x-y)\, dy,\ \ k \geq 1, \qquad (8)$$

а $F_{kw}(x)$ — матрица, элементы которой представляют собой вероятности того, что за время x обслужится $(k-w)$ заявок при условии, что в начальный момент в системе было ровно k заявок с учетом соответствующей смены фаз процессом обслуживания.

Для матриц $F_{kw}(x)$ также можно выписать рекуррентные соотношения, аналогичные соотношениям (7) и (8) для матриц $F_k(x)$, которые в точности будут совпадать с формулами, полученными для матриц $F_{kw}(x)$ при исследовании системы SM/MSP/n/r в работе [4], и поэтому они здесь не приводятся. В работах [2–4] можно познакомиться

с алгоритмическими методами численного расчета матриц $F_k(x)$ и $F_{kw}(x)$ и матриц, аналогичных $A_k^{ij}(x)$ и $A_{kw}^{ij}(x)$.

Теперь матрицы P_{ij} можно представить в виде

$$P_{i,i+1} = \begin{pmatrix} O & A_{ii}^{01} \\ O & A_{ii}^{11} \end{pmatrix}, \quad i = \overline{0, n-1}\,, \tag{9}$$

$$P_{i0} = \begin{pmatrix} A_{i0}^{00} & O \\ A_{i0}^{10} & O \end{pmatrix}, \quad i = \overline{0, R}; \tag{10}$$

$$P_{ij} = \begin{pmatrix} A_{ij}^{00} & A_{i,j-1}^{01} \\ A_{ij}^{10} & A_{i,j-1}^{11} \end{pmatrix}, \quad i = \overline{1, R}, \;\; j = \overline{1, \min\{i, n-1\}}\,; \tag{11}$$

$$P_{n,n+1} = \begin{pmatrix} O & A_0^{01} \\ O & A_0^{11} \end{pmatrix}, \quad P_{i,i+1} = \begin{pmatrix} O & O \\ O & \tilde{A}_0^1 \end{pmatrix}, \quad i = \overline{n+1, R-1}\,; \tag{12}$$

$$P_{i,n} = \begin{pmatrix} A_{i-n}^{00} & A_{i,n-1}^{01} \\ A_{i-n}^{10} & A_{i,n-1}^{11} \end{pmatrix}, \quad i = \overline{n, R}; \tag{13}$$

$$P_{i,n+1} = \begin{pmatrix} A_{i-n-1}^{00} & \sum\limits_{j=0}^{i-n} A_j^{01} \\ A_{i-n-1}^{10} & A_{i-n}^{11} + \sum\limits_{j=0}^{i-n-1} \tilde{A}_j^{11} \end{pmatrix}, \quad i = \overline{n+1, R-1}\,; \tag{14}$$

$$P_{R,n+1} = \begin{pmatrix} A_{R-n-1}^{00} & \sum\limits_{j=1}^{R-n} A_j^{01} \\ A_{R-n-1}^{10} & A_{R-n}^{11} + \sum\limits_{j=1}^{R-n-1} \tilde{A}_j^{11} \end{pmatrix}; \tag{15}$$

$$P_{ij} = \begin{pmatrix} A_{i-j}^{00} & O \\ A_{i-j}^{10} & \tilde{A}_{i-j+1}^1 \end{pmatrix}, \quad i = \overline{n+2, R}, \;\; j = \overline{n+2, \min\{i, R-1\}}\,; \tag{16}$$

$$P_{RR} = \begin{pmatrix} A_0^{00} & A_0^{01} \\ A_0^{10} & \tilde{A}_1^1 + A_0^{11} \end{pmatrix}. \tag{17}$$

Можно показать, что матрица P, образованная из заданных по формулам (9)–(17) матриц $P_{i,j}$, является неразложимой и непериодической. Методы решения СУР с матрицей такого вида изложены в работах [3, 5].

Обозначим через q_{ik}^m, $m = 0, 1$, $i = l_k(u-1) + v$, $u = \overline{1, I}$, $v = \overline{1, l_k}$, $k = \overline{0, R}$, стационарную вероятность того, что непосредственно до момента смены фаз процесса генерации заявок в системе было k других заявок, полумарковский процесс генерации заявок находился на фазе u, марковский процесс обслуживания — на фазе v и если $m = 0$, то поступление заявок в систему было заблокировано, а если $m = 1$, то поступление заявок в систему было разблокировано. Положим $\boldsymbol{q}_k^m = (q_{1k}^m, \ldots, q_{Il_k,k}^m)$, $\boldsymbol{q}_k = (\boldsymbol{q}_k^0, \boldsymbol{q}_k^1)$, $m = 0, 1$, $k = \overline{0, R}$.

Для векторов $\boldsymbol{q}_k$, $k = \overline{0, R}$, можно записать следующие соотношения:

$$\boldsymbol{q}_k = \sum_{j=k}^{R} \boldsymbol{p}_j \tilde{P}_{jk}, \quad k = \overline{0, n-1}\,; \tag{18}$$

$$\boldsymbol{q}_k = \sum_{j=k}^{R} \boldsymbol{p}_j \tilde{P}_{j-k}, \quad k = \overline{n, R}\,, \tag{19}$$

где

$$\tilde{P}_{ij} = \begin{pmatrix} A_{ij}^{00} & \tilde{A}_{ij}^{01} \\ A_{ij}^{10} & \tilde{A}_{ij}^{11} \end{pmatrix}, \quad i = \overline{0, R}, \;\; j = \overline{0, \min\{i, n-1\}}\,;$$

$$\tilde{P}_k = \begin{pmatrix} A_k^{00} & A_k^{01} \\ A_k^{10} & A_k^{11} \end{pmatrix}, \quad k \geq 0\,,$$

а матрицы A_k, $\tilde{A}_{kw}^{i1}$ и A_{kw}^{i0} определяются, в свою очередь, соотношениями (4)–(6).

Отыскав векторы $\boldsymbol{q}_k$, $k = \overline{0, R}$, можно вычислить вероятность π_1 потери новой заявки, а именно вероятность того, что заявка в момент генерации не встанет на прибор и не попадет в накопитель, а сразу покинет систему. Заявка в момент поступления теряется, если входящий в систему поток заблокирован или если в накопителе отсутствуют свободные места. Поэтому

$$\pi_1 = \sum_{i=0}^{R-1} \boldsymbol{q}_k^0 \mathbf{1} + \boldsymbol{q}_R \mathbf{1}\,. \tag{20}$$

Для стационарной вероятности π_2 того, что в момент генерации заявка не попадет в накопитель, можно получить следующее выражение:

$$\pi_2 = \sum_{i=0}^{n-1} \boldsymbol{q}_k \mathbf{1} + \sum_{i=n}^{R-1} \boldsymbol{q}_k^0 \mathbf{1} + \boldsymbol{q}_R \mathbf{1}\,. \tag{21}$$

Найдем вероятность ψ_1 того, что заявка будет обслужена при условии, что она попала в систему (на приборы или в накопитель). Если заявка попадает сразу на прибор, то она не может быть потеряна и будет обслужена в любом случае. Если же заявка попадает в накопитель, то она должна успеть обслужиться до тех пор, пока в систему после следующего момента разблокировки входящего потока не придет очередная заявка.

Обозначим через $\hat{p}_{ik}^m(j)$, $i = l(u-1)+v$, $u = \overline{1, I}$, $v = \overline{1, l}$, $k = \overline{1, r}$, $j \geq 0$, стационарную вероятность того, что произвольная заявка попадет в накопитель

и после j-го момента смены фаз полумарковского процесса останется в накопителе на k-м месте, причем полумарковский процесс генерации заявок будет находиться на фазе u, марковский процесс обслуживания — на фазе v и если $m = 0$, то поступление заявок в систему было заблокировано, а если $m = 1$, то поступление заявок в систему было разблокировано. Введем вектор-строки $\hat{\boldsymbol{p}}_k(j)$, $k = \overline{1,r}$, $j \geq 0$, следующим образом:

$$\hat{\boldsymbol{p}}_k(j) = (\hat{p}^0_{1k}(j), \ldots, \hat{p}^0_{Il,k}(j), \hat{p}^1_{1k}(j), \ldots, \hat{p}^1_{Il,k}(j)) .$$

Обозначим через $\hat{\boldsymbol{p}}(j)$, $j \geq 0$, вектор-строку $\hat{\boldsymbol{p}}(j) = (\hat{\boldsymbol{p}}_1(j), \ldots, \hat{\boldsymbol{p}}_r(j))$. Заметим, что вектор $\hat{\boldsymbol{p}}(0)$ представляет собой вектор стационарных вероятностей того, что новая заявка попадет в накопитель при соответствующих значениях фаз процессов генерации и обслуживания.

Рассмотрим блочную матрицу

$$\hat{P} = \begin{pmatrix} \hat{P}_{n+1,n+1} & O & \ldots & O & O \\ \hat{P}_{n+2,n+1} & \hat{P}_{n+2,n+2} & \ddots & O & O \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ \hat{P}_{R-1,n+1} & \hat{P}_{R-1,n+2} & \ldots & \hat{P}_{R-1,R-1} & O \\ \hat{P}_{R,n+1} & \hat{P}_{R,n+2} & \ldots & \hat{P}_{R,R-1} & \hat{P}_{RR} \end{pmatrix}, \quad (22)$$

где

$$\hat{P}_{i,j} = \begin{pmatrix} A^{00}_{i-j} & O \\ A^{10}_{i-j} & \tilde{A}^{1}_{i-j} \end{pmatrix},$$
$$i = \overline{n+1, R}, \;\; j = \overline{n+1, \min\{i, R-1\}};$$
$$\hat{P}_{R,R} = \begin{pmatrix} A^{00}_0 & A^{01}_0 \\ A^{10}_0 & \tilde{A}^{1}_1 + A^{11}_0 \end{pmatrix}.$$

Пусть очередная заявка, поступив в систему, попала в накопитель. Место в накопителе, на которое попадет заявка, определяется векторами, составляющими вектор $\hat{\boldsymbol{p}}(0)$, — вероятность того, что она окажется в накопителе на k-м месте, определяется вектором $\hat{\boldsymbol{p}}_k(0)$, $k = \overline{1,r}$. Тогда вероятность того, что сразу после следующего момента смены фазы процесса генерации она останется в накопителе (не будет выбита и не попадет на прибор), равна $\hat{\boldsymbol{p}}(1) = \hat{\boldsymbol{p}}(0)\hat{P}$, причем место, которое будет занимать заявка в накопителе, фазы процессов генерации и обслуживания, определяются координатой вектора $\hat{\boldsymbol{p}}(1)$. Вероятность того, что эта заявка останется в накопителе и после j-й, $j \geq 1$, смены фаз процесса генерации, равна

$$\hat{\boldsymbol{p}}(j) = \hat{\boldsymbol{p}}(0)\hat{P}^j, \;\; j \geq 1 .$$

Заметим, что матрица $\hat{P}$ является неотрицательной и сумма элементов в каждой строке строго меньше единицы. Поэтому максимальное по модулю собственное значение этой матрицы также меньше единицы, что гарантирует существование матрицы $(E - \hat{P})^{-1}$.

Пусть заявка после j-й смены фазы процесса генерации находится на k-м месте в накопителе (фазы процессов генерации и обслуживания, а также состояние блокировки процесса генерации определяются соответствующими координатами вектора $\hat{\boldsymbol{p}}_k(j)$, $k = \overline{1,r}$, $j \geq 0$). Чтобы эта заявка попала на прибор до очередного момента смены фаз процесса генерации, за время между сменами фаз должно обслужиться не менее k заявок. Вероятность этого события равна $\hat{\boldsymbol{p}}_k(j)\boldsymbol{p}^*_k$, где

$$\boldsymbol{p}^*_k = \begin{pmatrix} \sum\limits_{i=0}^{n} (A^{00}_{n+k,i}\mathbf{1} + A^{01}_{n+k,i}\mathbf{1}) \\ \sum\limits_{i=0}^{n} (A^{10}_{n+k,i}\mathbf{1} + A^{11}_{n+k,i}\mathbf{1}) \end{pmatrix}.$$

Вектор $\boldsymbol{p}^*_k$ представляет собой вектор стационарных вероятностей того, что заявка попадет на прибор до очередного момента смены фаз процесса генерации заявок при условии, что сразу после предыдущего момента смены фаз она находилась на k-м месте в накопителе. Обозначая через $\boldsymbol{p}^*$ вектор $\boldsymbol{p}^* = (\boldsymbol{p}^*_1, \ldots, \boldsymbol{p}^*_r)$, получаем, что вероятность того, что заявка, попав в накопитель, затем попадет на прибор между j-м и $(j+1)$-м моментами смены фаз процесса генерации, равна

$$\hat{\boldsymbol{p}}(j)\boldsymbol{p}^* = \hat{\boldsymbol{p}}(0)\hat{P}^j\boldsymbol{p}^* .$$

Вероятность того, что заявка, попав в накопитель, затем (когда-нибудь) попадет на прибор и не будет выбита другой заявкой, равна

$$\sum_{j=0}^{\infty} \hat{\boldsymbol{p}}(j)\boldsymbol{p}^* = \sum_{j=0}^{\infty} \hat{\boldsymbol{p}}(0)\hat{P}^j\boldsymbol{p}^* = \hat{\boldsymbol{p}}(0)(E - \hat{P})^{-1}\boldsymbol{p}^* .$$

Теперь осталось найти векторы $\hat{\boldsymbol{p}}_k(0)$, $k = \overline{1,r}$, тем самым определив вектор $\hat{\boldsymbol{p}}(0)$. При $k = \overline{1, r-1}$ имеет место соотношение

$$\hat{\boldsymbol{p}}_k(0) = (\mathbf{0}_{Il}, \boldsymbol{p}^1_{n+k}), \; k = \overline{1, r-1}; \quad (23)$$
$$\hat{\boldsymbol{p}}_r(0) = (\mathbf{0}_{Il}, \boldsymbol{q}^1_{n+r-1}), \quad (24)$$

где векторы $\boldsymbol{p}^1_{n+k}$ являются составляющими векторов $\boldsymbol{p}_{n+k}$, которые определяются из СУР (3) для вложенной цепи Маркова, а вектор $\boldsymbol{q}^1_{n+r-1}$ — из формулы (19).

Вероятность ψ_1 того, что заявка, попав в систему, будет обслужена, равна

$$\psi_1 = \frac{1}{1 - \pi_1} \left(\sum_{k=1}^{n} \boldsymbol{p}^1_k\mathbf{1} + \hat{\boldsymbol{p}}(0)(E - \hat{P})^{-1}\boldsymbol{p}^* \right). \quad (25)$$

Найдем вероятность ψ_2 того, что заявка, попав в систему, будет выбита другой заявкой. Вероятность того, что заявка, находящаяся в накопителе на k-м месте после j-й смены фаз процесса генерации с момента ее поступления в систему, будет выбита заявкой, поступившей в следующий момент смены фаз процесса генерации, равна $\hat{\boldsymbol{p}}_k(j)\bar{\boldsymbol{p}}_k$, где

$$\bar{\boldsymbol{p}}_k = \begin{pmatrix} \sum\limits_{j=0}^{k-1} A_j^{01}\mathbf{1} \\ \sum\limits_{j=0}^{k-1} \tilde{A}_j^{11}\mathbf{1} \end{pmatrix}, \quad k = \overline{1, r-1};$$

$$\bar{\boldsymbol{p}}_r = \begin{pmatrix} \sum\limits_{j=1}^{k-1} A_j^{01}\mathbf{1} \\ \sum\limits_{j=1}^{k-1} \tilde{A}_j^{11}\mathbf{1} \end{pmatrix}.$$

Обозначим через $\bar{\boldsymbol{p}}$ вектор-столбец

$$\bar{\boldsymbol{p}} = (\bar{\boldsymbol{p}}_1^T, \ldots, \bar{\boldsymbol{p}}_r^T)^T$$

и через $\bar{v}_{j+1} = \hat{\boldsymbol{p}}(j)\bar{\boldsymbol{p}}$, $j \geq 0$, вероятность того, что заявка будет выбита другой заявкой в $(j+1)$-й момент смены фаз процесса генерации. Тогда для стационарной вероятности ψ_2 того, что заявка будет выбита другой заявкой при условии, что она вообще попадет в систему, можно выписать соотношение

$$\psi_2 = \frac{1}{1-\pi_1}\sum_{j=0}^{\infty}\bar{v}_{j+1} = \frac{1}{1-\pi_1}\,\hat{\boldsymbol{p}}(0)(E-\hat{P})^{-1}\bar{\boldsymbol{p}}\,. \quad (26)$$

Нетрудно показать, что вероятности ψ_1 и ψ_2 связывает очевидное соотношение

$$\psi_1 + \psi_2 = 1\,. \quad (27)$$

Действительно, складывая соотношения (25) и (26), получаем

$$\psi_1 + \psi_2 = \frac{1}{1-\pi_1}\left(\sum_{k=1}^{n}\boldsymbol{p}_k^1\mathbf{1} + \hat{\boldsymbol{p}}(0)(E-\hat{P})^{-1}(\boldsymbol{p}^* + \bar{\boldsymbol{p}})\right). \quad (28)$$

Вспоминая выражение (22) для матрицы $\hat{P}$ и определения векторов $\boldsymbol{p}^*$ и $\bar{\boldsymbol{p}}$, а также учитывая стохастичность матрицы P переходных вероятностей вложенной цепи Маркова, можно записать следующее соотношение:

$$\boldsymbol{p}^* + \bar{\boldsymbol{p}} = (E-\hat{P})\mathbf{1}\,.$$

Подставляя эту формулу в соотношение (28) и учитывая формулы (18)–(20), нетрудно получить требуемое равенство (27).

Весьма полезной характеристикой системы является время, потраченное выбитой заявкой на ожидание обслуживания в накопителе. Для среднего времени $\bar{v}$ пребывания заявки в накопителе с момента ее поступления в систему при условии, что она будет выбита другой заявкой, справедливо соотношение

$$\bar{v} = \frac{a}{1-\pi_2}\sum_{j=1}^{\infty} j\bar{v}_j = \frac{a}{1-\pi_2}\hat{\boldsymbol{p}}(0)\sum_{j=1}^{\infty} j\hat{P}^{j-1}\bar{\boldsymbol{p}} = \frac{a}{1-\pi_2}\hat{\boldsymbol{p}}(0)(E-\hat{P})^{-2}\bar{\boldsymbol{p}}\,,$$

где среднее время a между соседними сменами фаз процесса генерации заявок определяется формулой (2), матрица $\hat{P}$ — формулой (22), вектор $\hat{\boldsymbol{p}}(0)$ — формулами (23) и (24), а вероятность π_2 — формулой (21).

Найдем среднее время w^* пребывания заявки в накопителе до момента ее поступления на прибор. Вероятность того, что заявка окажется на k-м месте в накопителе после j-го момента смены фаз процесса генерации, определяется координатами вектора $\hat{\boldsymbol{p}}_k(j)$. Тогда среднее время, проведенное заявкой в накопителе до этого момента, определяется координатами вектора $ja\hat{\boldsymbol{p}}_k(j)$.

Введем вектор-столбец

$$\boldsymbol{a} = (\boldsymbol{a}_1^T, \ldots, \boldsymbol{a}_r^T)^T\,,$$

где $\boldsymbol{a}_k$, $k = \overline{1, r}$, — вектор-столбец, координатами которого являются средние времена, проведенные до попадания на прибор заявкой, находившейся в накопителе на k-м месте в очередной момент смены фаз процесса генерации с учетом фаз процессов генерации и обслуживания, при условии, что она попадет на прибор до следующего момента смены фаз процесса генерации. Для векторов $\boldsymbol{a}_k$ справедливы соотношения

$$\boldsymbol{a}_k = \int_0^{\infty}(E - A(x)) \otimes xF_{k-1}(x)N dx\mathbf{1}, \quad k = \overline{1, r}\,,$$

где матрицы $F_k(x)$, $k \geq 0$, вычисляются с помощью формул (7) и (8).

Теперь можем записать выражение для среднего времени w^* ожидания начала обслуживания попавшей в накопитель заявки при условии, что она не будет выбита:

$$w^* = \frac{1}{1-\pi_2}\sum_{j=0}^{\infty}\hat{\boldsymbol{p}}(j)(ja\mathbf{1}+\boldsymbol{a}) =$$
$$= \frac{1}{1-\pi_2}\hat{\boldsymbol{p}}(0)(E-\hat{P})^{-1}\left(a\hat{P}(E-\hat{P})^{-1}\mathbf{1}+\boldsymbol{a}\right).$$

Поскольку время ожидания начала обслуживания заявки, попавшей сразу на приборы, равно нулю, то среднее время w ожидания начала обслуживания попавшей в систему заявки, не выбитой другими заявками, можно записать в виде

$$w = \frac{1}{1-\pi_1}\hat{\boldsymbol{p}}(0)(E-\hat{P})^{-1}\left(a\hat{P}(E-\hat{P})^{-1}\mathbf{1}+\boldsymbol{a}\right).$$

Время пребывания попавшей в систему заявки, не выбитой другими заявками, представляет собой сумму времени ожидания начала обслуживания и времени обслуживания заявки, при условии ее попадания на прибор. Поэтому среднее время v пребывания заявки в системе можно отыскать по формуле

$$v = \frac{1}{1-\pi_1}\left[b\sum_{k=1}^{n}\boldsymbol{p}_k^1\mathbf{1} + \right.$$
$$\left. + \hat{\boldsymbol{p}}(0)(E-\hat{P})^{-1}\left(a\hat{P}(E-\hat{P})^{-1}\mathbf{1}+\boldsymbol{a}+b\mathbf{1}\right)\right],$$

где $b = -\boldsymbol{h}H^{-1}\mathbf{1}$ — среднее время обслуживания заявки, а $\boldsymbol{h}$ и H — параметры распределения (1) фазового типа на отдельном приборе.

Таким образом, в настоящей работе найдены следующие показатели функционирования в стационарном режиме СМО с прерывающимся блокировкой полумарковским входящим потоком заявок, конечным накопителем, обслуживанием фазового типа на каждом приборе и выбиванием заявок из накопителя:

- распределение числа заявок в системе;
- вероятность непопадания заявки в систему;
- вероятность того, что заявка, попав в накопитель, будет обслужена;
- среднее время пребывания выбитой заявки в накопителе;
- среднее время ожидания начала обслуживания заявки, попавшей в систему и не выбитой другой заявкой;
- среднее время пребывания в системе этой заявки.

Литература

1. *Чаплыгин В. В.* Многолинейная система массового обслуживания с конечным накопителем и блокировкой полумарковского потока заявок // Информационные процессы, 2008. Т. 8. № 1. С. 1–9.
2. *Печинкин А. В., Чаплыгин В. В.* Стационарные характеристики системы массового обслуживания G/MSP/n/r // Вестник РУДН, сер. «Прикладная математика и информатика», 2003. № 1. С. 119–143.
3. *Бочаров П. П., Д'Апиче Ч., Печинкин А. В., Салерно С.* Система массового обслуживания G/MSP/1/r // Автоматика и телемеханика, 2003. № 2. С. 127–143.
4. *Печинкин А. В., Чаплыгин В. В.* Стационарные характеристики системы массового обслуживания SM/MSP/n/r // Автоматика и телемеханика, 2004. № 9. С. 85–100.
5. *Bocharov P. P., D'Apice C., Pechinkin A. V., Salerno S.* Queueing theory. — Utrecht, Boston: VSP, 2004.

ИНФОРМАТИКА И ЕЁ ПРИМЕНЕНИЯ, 2008. Т. 2. Вып. 3. С. 41–53

СЕРВИСНО-ОРИЕНТИРОВАННЫЙ ПОДХОД К РАЗРАБОТКЕ МУЛЬТИБИОМЕТРИЧЕСКИХ ТЕХНОЛОГИЙ*

О. С. Ушмаев[1]

Аннотация: В настоящее время значительное внимание уделяется технологиям мультибиометрической идентификации, т. е. идентификации человека одновременно по нескольким биометрическим признакам. В первую очередь такие технологии востребованы в перспективных системах гражданской идентификации, в частности в биометрическом паспорте. В статье предложен подход к созданию высокопроизводительных мультибиометрических технологий и систем на базе сервисно-ориентированной архитектуры. Приведены результаты разработки программного обеспечения на основе новых подходов.

Ключевые слова: биометрические технологии; мультибиометрическая идентификация; многозвенная архитектура; аппаратная независимость; сервисно-ориентированная архитектура

1 ВВЕДЕНИЕ

Биометрические технологии получили достаточное распространение в различных областях применения информационных технологий — от электронной коммерции до правоохранительной деятельности и паспортно-визового сегмента [1–7]. Суммируя накопленный опыт использования биометрических технологий, область их применения можно разделить на следующие значительные направления:

1. Коммерческие приложения:
   - разграничение физического доступа;
   - учет рабочего времени;
   - управление корпоративной сетью.
2. Доступ к секретной информации.
3. Системы идентификации личности:
   - системы гражданской идентификации (паспорт, визы, пользование общественными благами);
   - криминалистические системы.

Критерии качества биометрической системы в значительной степени зависят от назначения и условий эксплуатации. В задачах контроля доступа в помещение критичным требованием будет время создания шаблона при предъявлении биометрического измерения. Для биометрических систем гражданской идентификации (Civil ID) предъявляются максимально жесткие требования к времени сравнения, ошибкам 2-го рода и к ошибкам отказа в регистрации.

Учитывая опыт многочисленных испытаний биометрических технологий [8–14], основными критериями качества биометрических алгоритмов являются следующие статистические показатели:

- ошибка 1-го рода (False Rejection Rate, далее FRR, или False Non Match Rate);
- ошибка 2-го рода (False Acceptance Rate, далее FAR, или False Match Rate);
- среднее время сравнения 1 к 1;
- среднее время регистрации;
- доля отказов в регистрации (Fail to Enroll Rate).

Уверенный прирост производительности вычислительных средств позволяет в значительной степени пренебрегать временными характеристиками в большинстве функционирующих или перспективных приложений. Поэтому типовой оценкой качества биометрической системы является соотношение ошибок 1-го и 2-го рода. На рис. 1 приведены результаты измерения ошибок распознавания в ходе операционного тестирования различных биометрических систем [9]. На рис. 2 приведены результаты технологического тестирования современных российских биометрических технологий (взяты SDK производства 2007 г.).

*Работа поддержана грантом РФФИ (проект 07-07-00031) и Программой фундаментальных исследований ОНИТ РАН (проект 1.5).

[1]Институт проблем информатики Российской академии наук, oushmaev@ipiran.ru

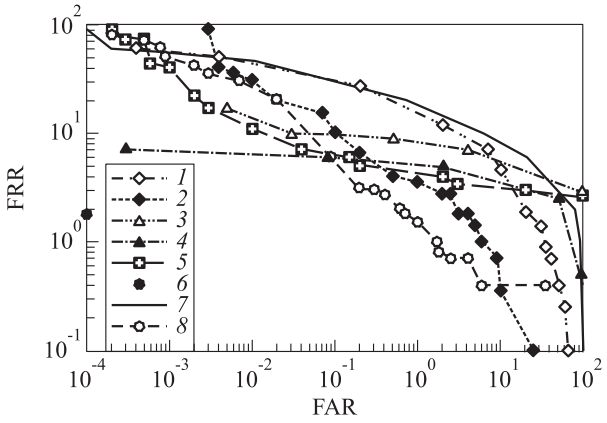

Рис. 1 Операционное тестирование биометрических систем [9], ошибки 1-го и 2-го рода: *1* и *2* — форма лица Visionics — FaceIT (*1*) и (*2*); *3* и *4* — отпечаток пальца — VeriTouch vr-3 (*1*) и(*2*); *5* — голос OTG — SecurPBX; *6* — радужная оболочка глаза — Iridian IriScan system 2200; *7* — рисунок вен — Neusciences-Biometrics; *8* — форма ладони — HandKey II

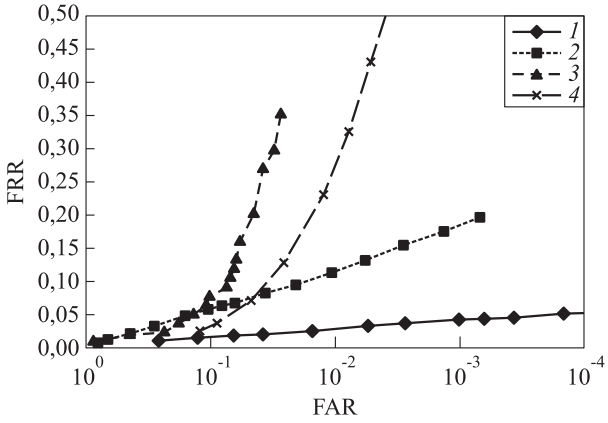

Рис. 2 Технологическое тестирование биометрических алгоритмов (2007 г.). Ошибки 1-го и 2-го рода: *1* — отпечаток пальца; *2* — изображение лица; *3* — голос; *4* — почерк

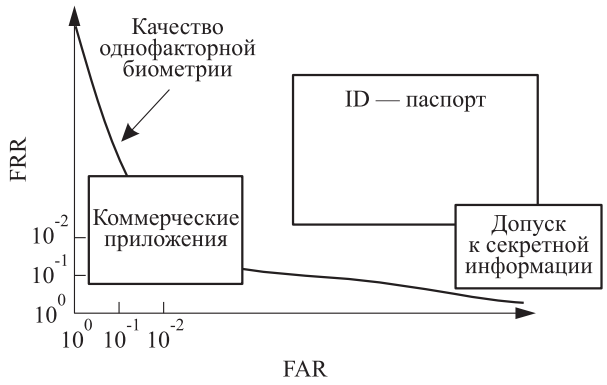

Рис. 3 Требования к ошибкам 1-го и 2-го рода в типовых приложениях

Одной биометрики достаточно для решения большей части задач типовых коммерческих приложений, таких как разграничение доступа в корпоративной сети, электронная коммерция, контроль доступа в помещение и т. д. В то же время большая часть государственных проектов предъявляет предельно жесткие требования (рис. 3), которые практически недостижимы при использовании единственной биометрики.

В такой ситуации естественным направлением развития биометрических технологий является создание мультибиометрических систем, использующих одновременно несколько биометрических измерений. На рис. 4 и 5 [4, 9, 11, 15] показан значительный прирост в качестве распознавания при одновременном использовании нескольких образцов одной биометрики или нескольких биометрических идентификаторов.

Если с алгоритмической точки зрения методы интеграции биометрических технологий частично проработаны [3–5, 15–18], то опыт разработки мультибиометрических приложений достаточно ограничен.

В ходе работ по созданию первой российской полнофункциональной системы мультибиометрической идентификации АМИС [6, 7] для использования в аналитических и криминалистических приложениях были сформированы следующие основные требования к создаваемому решению:

- возможность интеграции в рамках одной системой максимального числа методов биометрической идентификации;
- возможность одновременной комплексной идентификации по нескольким биометрическим признакам;
- возможность модернизации и замены библиотек, реализующих отдельные методы биометрической идентификации;
- гибкость в конфигурировании;
- гибкость в распределении вычислений;
- обеспечение комплексной информационной безопасности и защиты биометрической информации;
- поддержка существующих российских стандартов по биометрическим технологиям.

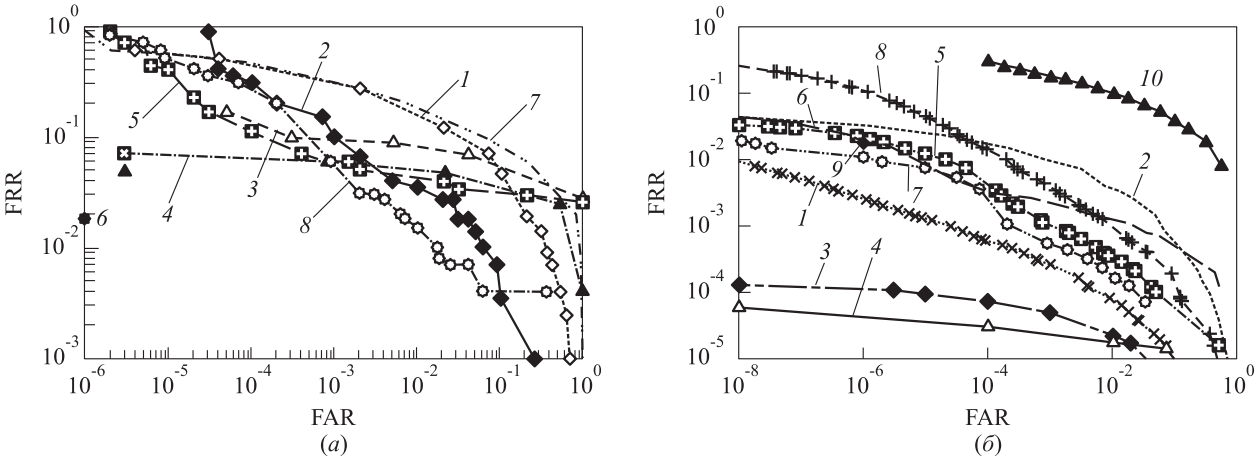

Рис. 4 Ошибки 1-го и 2-го рода различных биометрических систем по данным тестирования U.K. Biometrics Working Group [9] (*а*): *1*–*8* — см. рис. 1, — и теоретические ошибки 1-го и 2-го рода многофакторных биометрических систем [4] (*б*): *1* — форма лица + отпечаток пальца (ПП) + голос; *2* — форма лица + отпечаток пальца (ПП); *3* — форма лица + радужная оболочка глаза; *4* — отпечаток пальца (ПП) + радужная оболочка глаза; *5* — отпечаток пальца (ПП) + форма ладони; *6* — отпечаток пальца (ПП) + голос; *7* — радужная оболочка глаза + форма ладони; *8* — форма лица + форма ладони; *9* — радужная оболочка глаза; *10* — отпечаток пальца (NIST VTB)

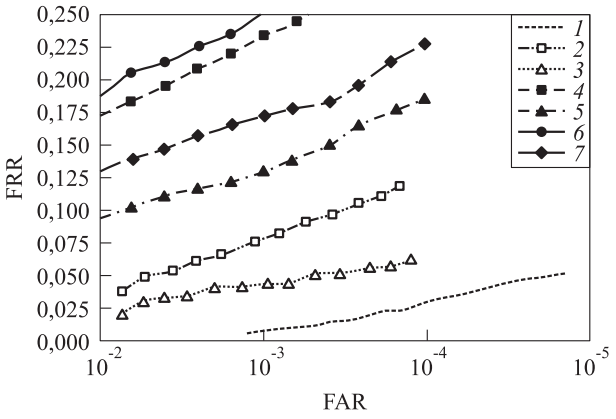

Рис. 5 Ошибки 1-го и 2-го рода распознавания по нескольким отпечаткам пальцев (NIST SD14) [15]: *1* — 4 отпечатка; *2* — 2 отпечатка (указательные); *3* — 2 отпечатка (большие); *4* — правый указательный; *5* — правый большой; *6* — левый указательный; *7* — левый большой

Далее в статье изложен подход к разработке мультибиометрических технологий, учитывающий перечисленные требования. В разделе 2 изложена методологическая основа разработки мультибиометрических технологий. Подход к созданию аппаратной архитектуры и программной инфраструктуры мультибиометрической системы, удовлетворяющей перечисленным требованиям, изложен в разделе 3. В разделе 4 изложены результаты разработки программного обеспечения, реализующего данный подход.

2 Интеграция биометрических технологий

Разработка интегрированных биометрический (многофакторных или мультибиометрических) систем имеет два существенных аспекта — технический и методологический. Технически интеграция заключается в объединении нескольких программно-аппаратных комплексов единым интерфейсом. С методологической точки зрения интеграция является достаточно сложной прикладной задачей принятия решения об идентичности наборов биометрических образцов.

Основой для разработки произвольной биометрической, в том числе и мультибиометрической, системы являются требования фактически сложившихся отраслевых стандартов. Наиболее систематизированным документом является стандарт bioAPI[1] (основа серии стандартов ГОСТ Р ИСО/МЭК 19784). Согласно стандарту bioAPI, базовыми функциями произвольной биометрической системы являются регистрация (enroll) и сравнение (match). В ходе регистрации информация, полученная при помощи биометрических сканеров, преобразуется в цифровой шаблон.

На этапе сравнения предъявляемые биометрические данные сравниваются с шаблоном регистрации. Результатом сравнения биометрических дан-

[1] BioAPI — Biometric Application Programming Interface — биометрический программный интерфейс приложений.

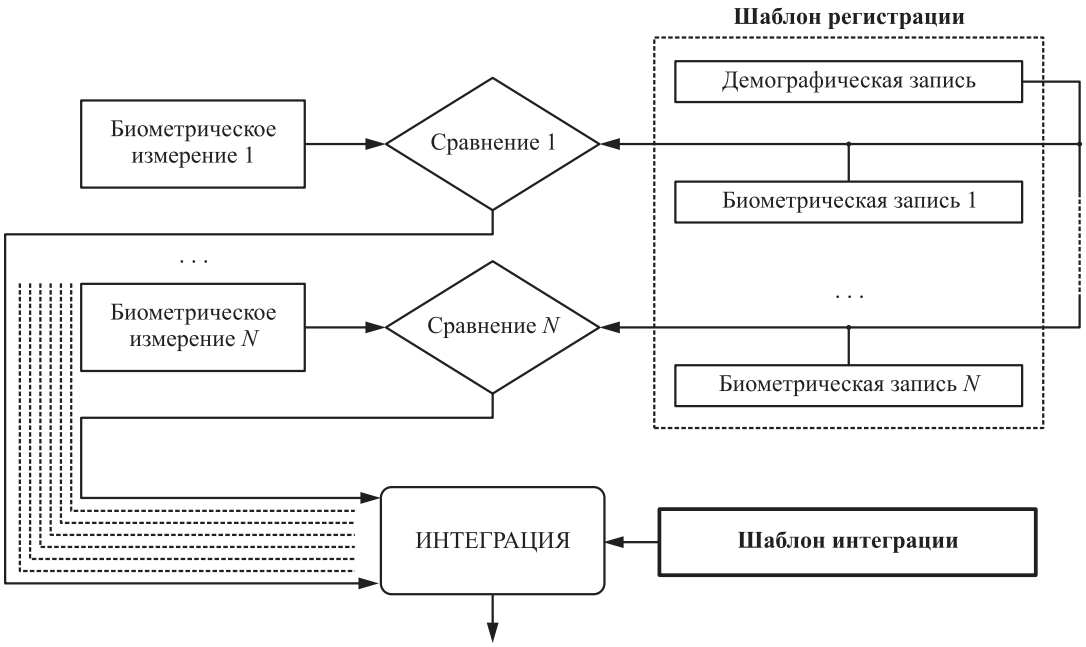

Рис. 6 Схема биометрической интеграции в паспортной задаче

ных является число, в том числе и для многофакторной биометрической системы. На рис. 6 представлена схема простейшей, с алгоритмической точки зрения, многофакторной биометрической системы — биометрический паспорт.

Отличительной упрощающей интеграцию чертой паспортной задачи (или систем гражданской идентификации в целом) является структурированный и контролируемый ввод биометрической информации. В шаблоне регистрации фиксируется стандартизированный набор биометрических измерений, например пара отпечатков пальцев и фотография лица. Соответственно в штатном режиме функционирования для верификации или идентификации личности предъявляется такой же ограниченный и полный набор данных. Такая схема оставляет минимальную свободу действий при интеграции.

Так как паспортные системы ориентированы на достижение минимальной ошибки 2-го рода, целесообразно использовать максимально возможную информацию о сравниваемых образцах, т. е. дожидаться ответа от всех используемых биометрических технологий. Данный подход позволяет максимально оптимизировать ошибки 1-го и 2-го рода за счет уменьшения производительности системы (рис. 7).

Более сложным примером многофакторной биометрической идентификации являются криминалистические системы. В данном случае нет серьезных ограничений по времени. Главным критерием являются ошибки 1-го и 2-го родов. Однако в отличие от паспортной задачи в криминалистической системе не структурирован шаблон регистрации, в системе может храниться произвольное количество биометрических образцов, принадлежащих одному субъекту. Также криминалистическая система может использовать произвольное число биометрических алгоритмов, использующих одну и ту же биометрию. В отличие от паспортной задачи, где замена алгоритмической начинки технологии максимально затруднена, криминалистические системы при разумном проектировании позволяют в штатном порядке обновлять или добавлять биометрические алгоритмы.

Соответственно, в криминалистической задаче могут быть реализованы все уровни интеграции [5–7]:

- на уровне модальностей (multimodal biometrics);
- на уровне биометрических образцов (multisample);
- на уровне алгоритмов (multialgorithm).

Интеграция на уровне образцов в данном случае предполагает систематические решения, улучшающие качество распознавания при одновременном предъявлении нескольких образцов одного биометрического измерения. На рис. 8 представлены графики ошибок распознавания в зависимости от числа предъявляемых для идентификации образцов одного отпечатка пальца.

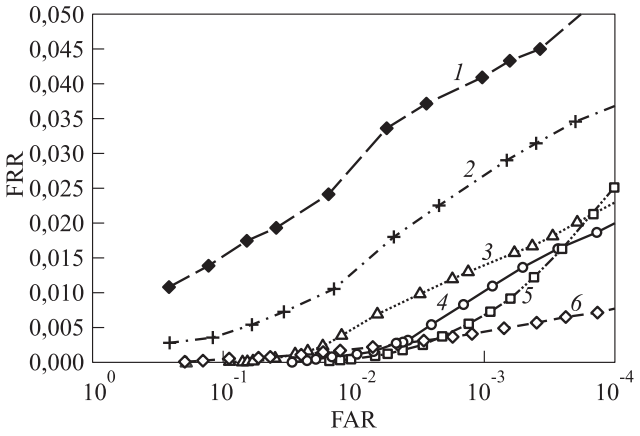

Рис. 7 Теоретические ошибки 1-го и 2-го рода многофакторных биометрических систем на основе современных алгоритмов (2007): *1* — отпечаток пальца; *2* — два образца отпечатка пальца; *3* — отпечаток пальца + голос; *4* — отпечаток пальца + почерк; *5* — лицо + голос + почерк; *6* — лицо + отпечаток пальца

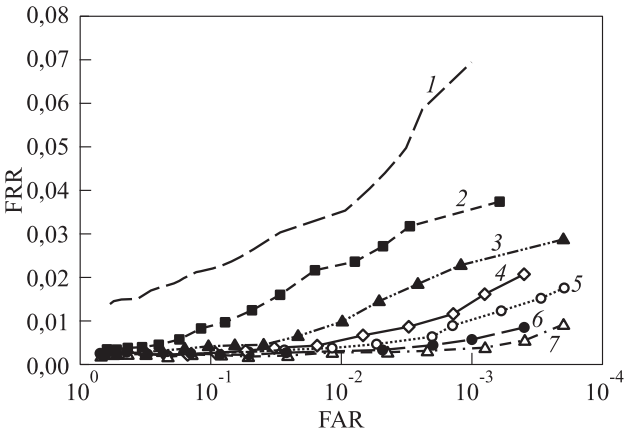

Рис. 8 Ошибки 1-го и 2-го рода распознавания по отпечаткам пальцев в зависимости от числа образцов (FVC2002 DB1): *1* — 1 образец; *2* — 2; *3* — 3; *4* — 4; *5* — 5; *6* — 6; *7* — 7 образцов

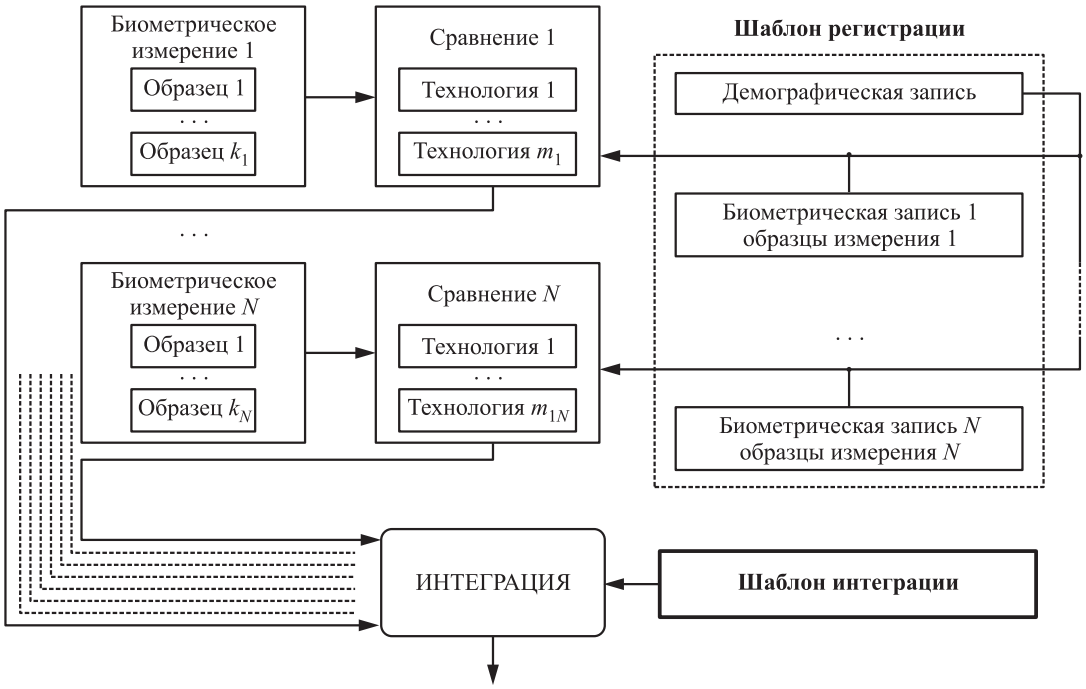

Рис. 9 Схема биометрической интеграции в общем случае

Схематично поток данных в многофакторной криминалистической системе представлен на рис. 9.

В качестве математического обеспечения блока интеграции могут использоваться произвольные алгоритмы, позволяющие достичь необходимого уровня статистических показателей качества биометрической системы. Более подробно алгоритмические проблемы биометрической интеграции изложены в [4, 15–18].

3 Программная архитектура

Принимая во внимания тенденцию к созданию крупномасштабных распределенных биометрических систем, следует изначально создавать масшта-

бируемые и распределенные архитектуры. В рамках распространенной на сегодняшний день концепции сервисо-ориентированной архитектуры [19, 20] следует, в первую очередь, разделить внутреннюю логику биометрических приложений на элементарные сервисы. Учитывая опыт разработки высокопроизводительных биометрических приложений [5], функционально можно выделить следующие сервисы:

- вычислительные сервисы, отвечающие за выполнение функций биометрических библиотек, ядро системы;
- сервисы бизнес-логики приложения;
- сервисы хранилища;
- клиентские приложения, «тонкий» клиент терминальных станций;
- вспомогательные сервисы управления, мониторинга, диагностики;
- сервисы сообщений/предоставления интерфейса, отвечающие за обмен информацией между узлами системы;
- сервисы операционной системы, распределенная среда исполнения.

На рис. 10 приведена схема взаимодействия перечисленных групп сервисов. С точки зрения аппаратных средств перечисленные сервисы могут исполняться как на одном вычислительном средстве, так и на отдельных специализированных серверах или кластерах. На рис. 11 приведен пример комплекса аппаратных средств системы гражданской идентификации.

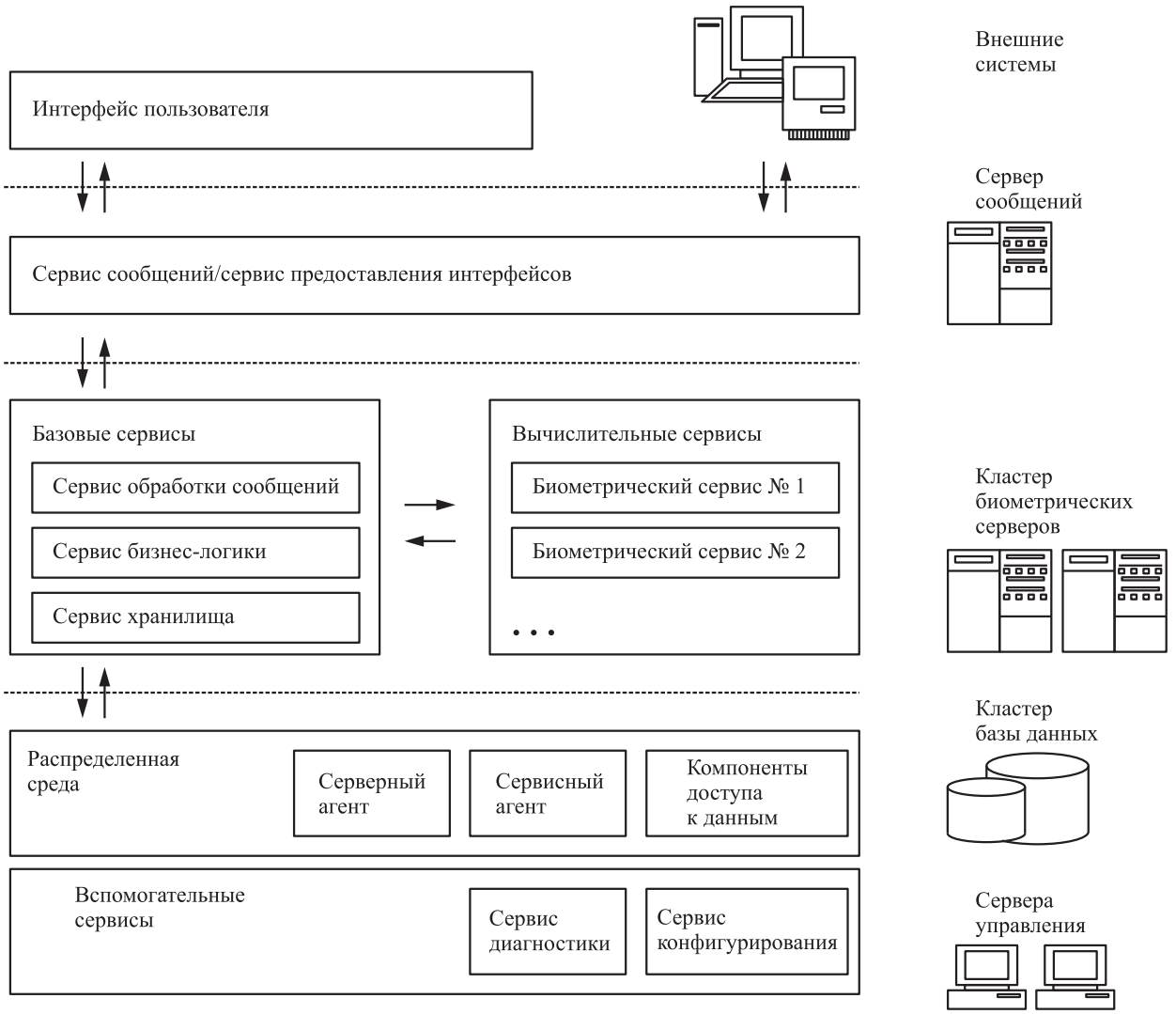

Рис. 10 Программная архитектура

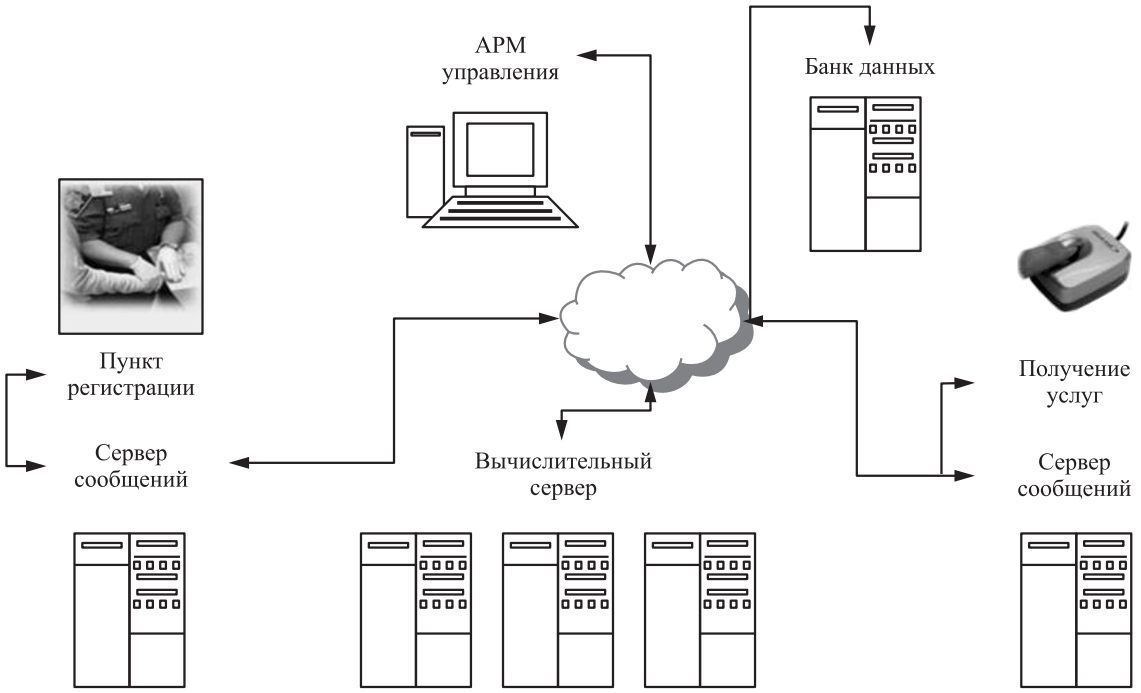

Рис. 11 Комплекс аппаратных средств системы гражданской идентификации

На рис. 12 приведен пример распределенной архитектуры системы интеллектуального видеонаблюдения [21].

По организации вычислений и логике построения информационных потоков приведенные примеры (рис. 11 и 12) отличаются только источником биометрической информации. Если следовать основным отраслевым стандартам, созданным по результатам разработки bioAPI, то биометрические приложения на уровне прикладного программного интерфейса устроены одинаково независимо от выбранного метода биометрической идентификации. Несмотря на вполне стандартизированные вызовы и протоколы обмена данными внутри биометрических приложений, работа с терминальными устройствами может требовать специальной организации вычислительного процесса. Поэтому разделение терминальных устройств и основного вычислительного узла при помощи сервера сообщений, как это приведено на рис. 11 и 12, является оправданным. Такой подход позволяет отделить работу с внешними устройствами, чей интерфейс зачастую не стандартизирован, от основного вычислительного узла, который, в свою очередь, должен быть реализован с учетом требований российских и международных стандартов.

Согласно bioAPI и российским стандартам по биометрии, реализация биометрической системы предполагает три типа данных:

(1) исходные данные, полученные с терминальных устройств;

(2) данные, обработанные функциями специализированных библиотек (фильтрация, шумоочистка и т. д.);

(3) цифровой шаблон/модель биометрического образца.

Функции биометрической системы:

– регистрация, получение шаблона на основе исходных данных;

– сравнение двух шаблонов, определение меры сходства;

– нестандартные функции дополнительной обработки биометрической информации.

При дополнительной обработке данных помимо функций возможно более сложное взаимодействие с биометрическими библиотеками, реализованное в виде контролов или ActiveX компонентов.

Соответственно функционал базовых сервисов, работающих в рамках мультибиометрического приложения, должен поддерживать перечисленные типы данных и вызовы функций. В случае биометрических приложений более сложной задачей является работа с перечисленными типами данных и их передача внутри системы.

Основой для протокола передачи исходных биометрических данных являются стандарты по обмену

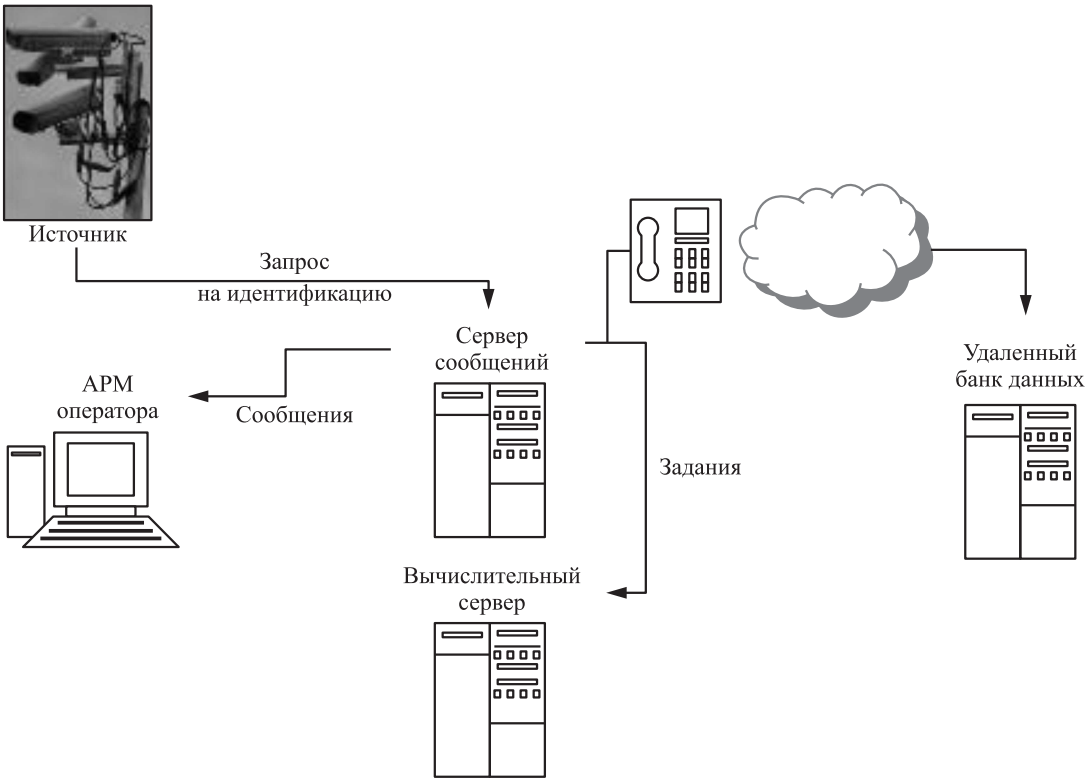

Рис. 12 Комплекс аппаратных средств системы видеонаблюдения

биометрической информации. Наиболее общими с точки зрения применимости к мультибиометрическим технологиям являются стандарты ITL-1-200x. Стандарты данной серии предполагают следующий состав необходимой информации:

- заголовок;
- сопроводительная текстовая информация;
- биометрические образцы;
- сопроводительная информация по каждому биометрическому образцу.

В настоящей редакции серия стандартов ITL 1-200x ориентирована, в первую очередь, на правоохранительную деятельность. Поэтому в данном формате данных стандартные контейнеры предназначены для хранения цифровых фотографий и дактилоскопического материала. Только в редакции 2007 г. данный стандарт пополнился контейнерами для геометрии руки и радужной оболочки глаза. Данное обстоятельство обусловлено тем, что на сегодняшний день разработаны следующие стандарты форматов обмена биометрической информацией:

- ANSI INCIST 377-2004, Information Technology — Finger Pattern Based Interchange Format;
- ANSI INCIST 378-2004, Information Technology — Finger Minutiae Format for Interchange;
- ANSI INCIST 379-2004, Information Technology — Iris Image Interchange Format;
- ANSI INCIST 381-2004, Information Technology — Finger Image-Based Data Interchange Format;
- ANSI INCIST 385-2004, Information Technology — Face Recognition Format for Data Interchange;
- ANSI INCIST 395-2005, Information Technology — Signature/Sign Data Interchange Format;
- ANSI INCIST 396-2005, Information Technology — Hand Geometry Interchange Format.

Как видно из списка, в стандартах представлены отпечатки пальцев, форма лица, геометрия руки, почерк/подпись и радужная оболочка глаза. При этом отсутствует стандарт на мультибиометрические данные. В такой ситуации целесообразным является развитие подхода ITL 1-200x. А именно: предлагается дополнить формат данных недостающими контейнерами для хранения данных, отличных от перечисленного списка стандартов. Для

Таблица 1 Список биометрических технологий АМИС

Биометрия	Основные источники	Точность идентификации
Отпечаток пальца	1. Данные обязательной дактилоскопической регистрации 2. Следы отпечатков пальцев 3. Системы контроля доступа 4. Биометрические паспорта нового поколения	Очень высокая
Форма лица	1. Фотографии; видеозаписи 2. Камеры наружного наблюдения 3. Системы контроля доступа 4. Биометрические паспорта нового поколения	Высокая
Голос	1. Телефонные линии 2. Записи разговоров	Низкая
Почерк	Произвольные рукописные образцы	Низкая

описания данных внутри контейнера можно использовать структуры CBEFF[1] и bioAPI. К настоящему времени в bioAPI зарегистрировано более 20 методов (против 5 стандартизированных) биометрической идентификации.

Для работы с обработанными данными и цифровыми биометрическими шаблонами целесообразно использовать схожую структуру CBEFF. Однако здесь следует отметить, что представленные в таком виде цифровые шаблоны слабо пригодны для обмена со сторонними автоматизированными информационными системами по причине отсутствия стандартов по форматам биометрических цифровых шаблонов для всех биометрик, кроме отпечатков пальцев.

4 Программная реализация

В данном разделе приведены результаты разработки программного комплекса, реализующего изложенную в предыдущих разделах концепцию мультибиометрической идентификации (Automatic Multibiometric Identification System, AMIS или АМИС).

В ходе работ по созданию АМИС было решено интегрировать основные систематически доступные биометрики, а именно: отпечатки пальцев, цифровое изображение лица, голос и образцы рукописного почерка. В табл. 1 представлен список биометрических технологий АМИС и потенциальные источники их получения.

Функционально АМИС предназначена для решения следующих задач:

- импорт, экспорт и хранение биометрических образцов;
- учет и обработка биометрических образцов;
- оперативная обработка запросов на биометрическую идентификацию.

С точки зрения реальных сценариев эксплуатации АМИС перечисленные задачи решаются различными группами пользователей. Учет и первичная обработка биометрической информации требует специальной квалификации и дополнительного аппаратного обеспечения, что приводит к необходимости выделения программного обеспечения отдельных автоматизированных рабочих мест (ПО АРМ) биометриста.

Подготовка запроса на биометрическую идентификацию в большинстве случаев не связана со специфической обработкой биометрической информации и может быть выполнена обычными пользователями. В частности, это позволяет использовать сервисы АМИС посредством web-интерфейса.

При разработке аппаратной архитектуры также приходилось учитывать, что биометрическая идентификация является трудоемкой процедурой для вычислительных средств общего назначения. Например, запрос на идентификацию голоса требует передачи больших объемов информации и может обрабатываться значительное время, измеряемое десятками минут или даже часами при значительном размере базы дикторов. Такая низкая производительность требует использования для выполнения вычислений максимально производительных и бесперебойно работающих ресурсов.

[1] CBEFF — Common Biometric Exchange File Format — единый формат представления биометрических данных.

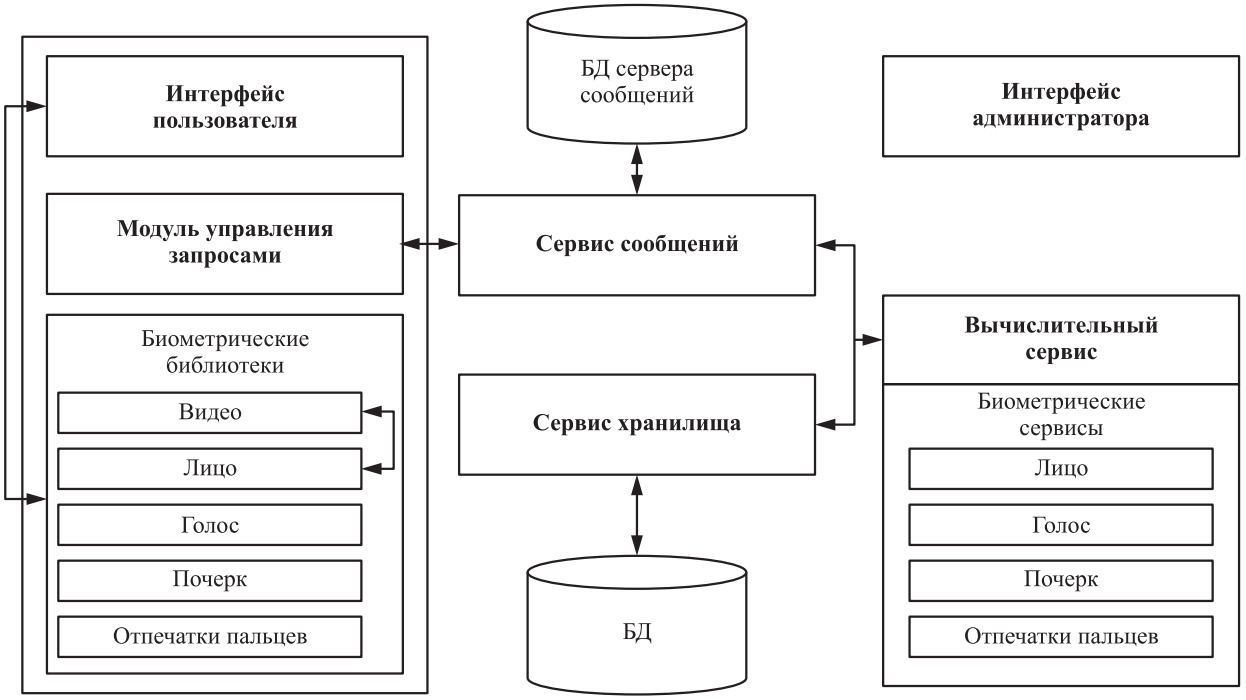

Рис. 13 Программная архитектура АМИС

Для эффективного решения поставленных задач в состав комплекса необходимо включить высокопроизводительный сервер и специально оборудованные АРМ биометриста, оснащенные специализированным аппаратным обеспечением. С учетом вышеперечисленного АМИС разрабатывалась как клиент-серверное приложение (рис. 13).

Учитывая функциональное назначение компонентов, мы предлагаем следующее базовое распределение сервисов и вычислительной нагрузки между клиентским и серверным компонентами системы.

В состав клиентского компонента АМИС включены:

- интерфейс пользователя;
- модуль управления запросами пользователя;
- биометрические сервисы;
- интерфейс администратора.

В состав серверного компонента АМИС включены:

- сервис сообщений;
- сервис хранилища;
- биометрические сервисы.

Конечному пользователю, оператору АМИС, основной функционал предоставляется посредством или Интранет-сервера АМИС (технология «тонкого клиента»), или специализированного ПО терминальных станций («толстый клиент»). Пользовательские функции АМИС можно представить в виде комбинации следующих обращений:

- Enroll, регистрация объекта учета в хранилище;
- Modify, изменение информации об объекте учета;
- Identify, биометрическая идентификация;
- Select, запрос на выборку данных из хранилища.

Перечисленные обращения схематично представлены на рис. 14.

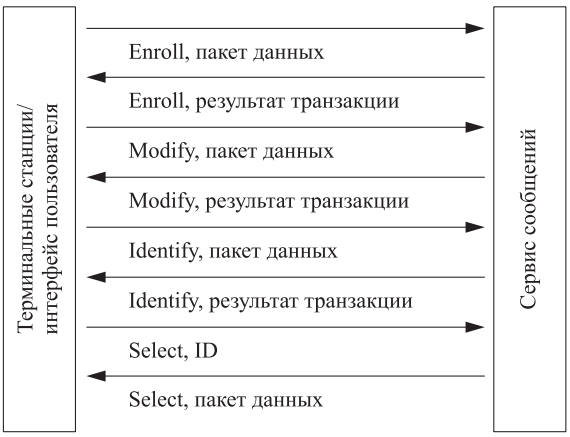

Рис. 14 Обмен данными между интерфейсом пользователя и сервисом сообщений АМИС

Таблица 2 Характеристики биометрических шаблонов

Технология	Среднее время создания шаблона, мс	Средний размер исходных данных, КБ	Средний размер шаблона, Б	Степень сжатия информации
Отпечатки пальцев. Десятипальцевая дактокарта	2349	700	26170	27
Отпечатки пальцев. След	363	35	2617	13,5
Изображение лица	595	50	2933	17
Голос	2933	2000	54370	40
Почерк	3290	780	20528	38

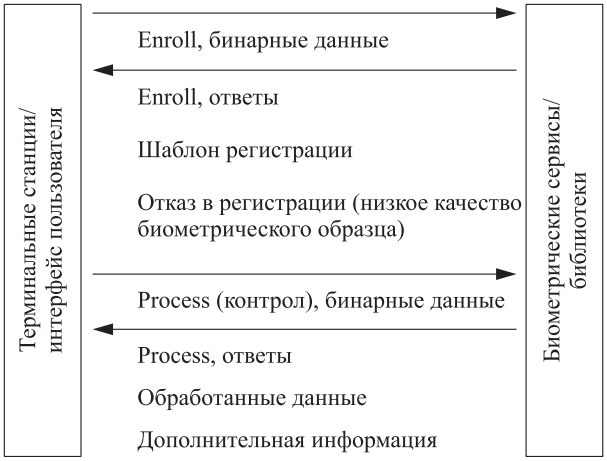

Рис. 15 Обмен данными между интерфейсом пользователя и биометрическими библиотеками клиентской части АМИС

Для большинства биометрических библиотек цифровой шаблон, полученный при вызове функции Enroll биометрической библиотеки, по размеру значительно компактней исходных необработанных данных. Сравнительный размер шаблонов и данных представлен в табл. 2. В частности, из таблицы видно, что хранение шаблонов предпочтительнее.

Для снижения нагрузки на сеть при передаче данных возможно частичное перераспределение нагрузки между клиентским АРМ и сервером. Поскольку операции, выполняемые при идентификации, требуют обращения к биометрическому хранилищу, которое является общим ресурсом для всех пользователей АМИС, целесообразным является перераспределение вычислений на клиентское вычислительное средство только на этапе подготовки данных и формирования шаблона. Схема обращений интерфейса пользователя к биометрическим библиотекам представлена на рис. 15.

Приведенная реализация клиент-серверного приложения потенциально является независимой от операционной системы. Однако, рассматривая реализацию серверного компонента, приходится учитывать специфику общесистемной части. АМИС была разработана на базе следующего общесистемного ПО производства Microsoft:

- Операционная система — MS Windows Server 2003/ XP;
- СУБД — MS SQL Server 2005;
- Транспорт — Windows Communication Foundation (WMC) из состава комплекта библиотек .NET Framework 3.0.

Данная конфигурация ПО позволяет полностью переложить трудоемкие в реализации механизмы информационной безопасности и разграничения доступа на операционную систему. А именно: встроенные механизмы СУБД и транспортной системы позволяют перейти на сквозную идентификацию посредством инструментов Windows Authentication. Соответственно, разграничение полномочий по использованию общих ресурсов осуществляется административными средствами операционной системы.

Внутри серверного компонента управление потоками данных осуществляет сервис сообщений, который обращается к общим ресурсам (хранилищу) и вычислительным библиотекам (рис. 16).

Ядром алгоритмического обеспечения АМИС являются биометрические библиотеки, входящие в состав вычислительного сервиса АМИС. В состав биометрической составляющей АМИС входят модули однофакторной биометрической идентификации, модули экспертной (первичной) обработки, модуль анализа видеоряда и модуль комплексной обработки результатов биометрической идентификации. Обращения к модулям однофакторной биометрической идентификации (рис. 17) стандартизированы ГОСТ 9784.

Модуль комплексной биометрической идентификации вычислительного сервиса АМИС, в котором производится оценка меры сходства наборов

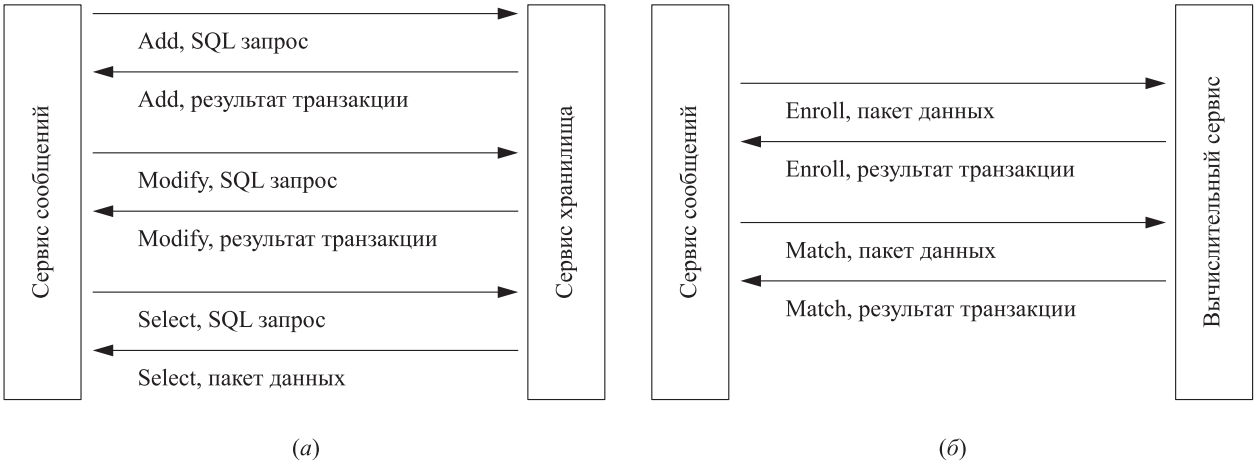

Рис. 16 Обмен данными между сервисом сообщений и (*а*) сервисом хранилища АМИС; (*б*) вычислительным сервисом АМИС

биометрических образцов по результатам идентификации по отдельным признакам, реализуется согласно общей схеме рис. 9. Внутри могут быть использованы произвольные алгоритмы, обеспечивающие требуемые ошибки 1-го и 2-го рода. В частности, в [4, 15–18] приведен широкий класс алгоритмов биометрической интеграции, предназначенные для оптимизации ошибок распознавания.

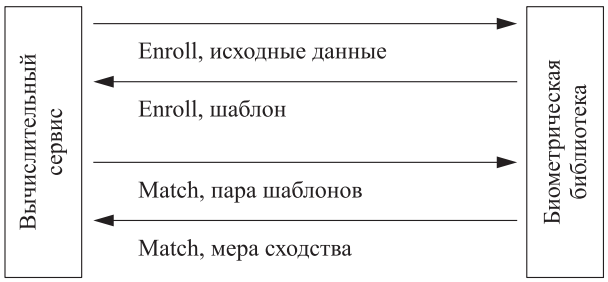

Рис. 17 Обмен данными между вычислительным сервером и биометрическими библиотеками серверной части АМИС

5 Заключение

В статье предложен новый подход к созданию мультибиометрических технологий на базе сервисно-ориентированной архитектуры. Ранее преобладали принципы разработки биометрических технологий в рамках единого приложения. В качестве основных преимуществ выработанного подхода отметим:

- программная архитектура позволяет разделить общесистемную часть и работу с конкретными биометрическими библиотеками, реализующими методы биометрической идентификации;
- реализация общесистемной части независима от состава биометрических библиотек;
- разработанная архитектура позволяет модернизировать и заменять отдельные биометрические библиотеки;
- программная архитектура обеспечивает комплексную идентификация по нескольким биометрическим признакам одновременно;
- программная архитектура обеспечивает возможность распределения основных вычислительных узлов, хранилища данных и интерфейса пользователей между различными вычислительными средствами;
- полное соответствие требованиям основных международных и российских стандартов к биометрическим системам и обмену биометрической информации;
- защита информационных ресурсов осуществляется механизмами операционной системы.

Направлениями для дальнейшего развития разработанного подхода с целью обеспечения большей универсальности, увеличения производительности и надежности являются:

- обеспечение масштабируемости;
- обеспечение отказоустойчивости;
- реализация механизмов распараллеливания вычислений на биометрических узлах.

Литература

1. *Beardsley, Ch. T.* Is your computer insecure? // IEEE Spectrum, 1972. P. 67–78.
2. *Woodward J. D., Jr.* Biometrics: Facing up to terrorism // The Biometric Consortium Conference 2002. Arlington, February, 2002.
3. *Wayman J., Jain A., Maltoni D., Maio D.* Biometric systems: Technology, design and performance evaluation. — Springer Verlag, 2004.
4. *Синицын И. Н., Новиков С. О., Ушмаев О. С.* Развитие технологий интеграции биометрической информации // Системы и средства информатики, 2004. Вып. 14. С. 5–36.
5. *Ушмаев О. С., Синицын И. Н.* Опыт проектирования многофакторных биометрических систем // Тр. VIII международной научно-технической конференции «Кибернетика и высокие технологии XXI века», 2007. Т. 1. С. 17–28.
6. *Ушмаев О. С.* Реализации концепции многофакторной биометрической идентификации в правоохранительных системах. Интерполитех-2007. http://www.dancom.ru/rus/AIA/Archive/RUVI_BioLinkSolutions_MultimodalBiometricsConcept.pdf.
7. *Ушмаев О. С., Босов А. В.* Реализация концепции многофакторной биометрической идентификации в интегрированных аналитических системах // Бизнес и безопасность в России, 2008. № 49. С. 104–105.
8. First International Competition for Fingerprint Verification Algorithms (FVC2000). http://bias.csr.unibo.it/fvc2000/.
9. *Mansfield T., Kelly G., Chandler D., Kane J.* Biometric product testing final report. U.K. Biometrics Working Group, 2001. http://www.cesg.gov.uk/site/ast/biometrics/media/BiometricTestReportpt1.pdf.
10. FVC2002, the Second International Competition for Fingerprint Verification Algorithms (FVC2002). http://bias.csr.unibo.it/fvc2002/.
11. *Mansfield A. J., Wayman J. L.* Best practices in testing and reporting performance of biometric devices. U.K. Biometrics Working Group, 2002.
12. Studies of Fingerprint Matching Using the NIST Verification Test Bed (VTB). ftp://sequoyah.nist.gov/pub/nist_internal_reports/ir_7020.pdf.
13. Face Recognition Vendor Test. http://www.frvt.org.
14. Fingerprint Vendors Technology Evaluation. http://fpvte.nist.gov.
15. *Ushmaev O. S., Novikov S. O.* Integral criteria for large-scale multiple fingerprint solutions // Biometric Technology for Human Identification / Ed. A. K. Jain, N. K. Ratha. Proceedings of SPIE. Berlingham, WA: SPIE, 2004. Vol. 5404. P. 534–543.
16. *Пугачев В. С., Синицын И. Н.* Теория стохастических систем. — М.: Логос, 2004. [Stochastic systems: Theory and applications. — World Scientific. Singapore, 2001].
17. *Ушмаев О. С., Новиков С. О., Синицын И. Н.* Стохастические проблемы интегрированной обработки информации // Проблемы и методы информатики. II Научная сессия Института проблем информатики РАН, Москва, 18–22 апреля 2005. Тезисы докладов. — М.: ИПИ РАН, 2005. С. 77–79.
18. *Ushmaev O., Novikov S.* Biometric fusion: Robust approach // MMUA06 Proceedings, 2006. Toulose, France.
19. Reference Model for Service-Oriented Architecture 1.0. http://www.oasis-open.org/committees/download.php/19679/soa-rm-cs.pdf.
20. *Cherbakov L., Galambos G., Harishankar R., Kalyana S., Rackham G.* Impact of service orientation at the business level // IBM Systems J. SOA, 2005. Vol. 44. No. 4. P. 653–668.
21. *Ушмаев О. С., Синицын И. Н.* Информационные технологии распознавания лиц в потоке // 8-я международная научно-техническая конференция «Распознавание-2008». 13–15 мая 2008 г. В печати.

ИНФОРМАТИКА И ЕЁ ПРИМЕНЕНИЯ, 2008. Т. 2. Вып. 3. С. 54–69

ЗАДАЧИ ПРЕДСТАВЛЕНИЯ ЛИЧНОСТНЫХ И КОЛЛЕКТИВНЫХ КОНЦЕПТОВ В ЦИФРОВОЙ СРЕДЕ

И. М. Зацман[1], В. В. Косарик[2], О. А. Курчавова[3]

Аннотация: В статье анализируются ключевые положения документов 7-й Рамочной программы Европейского Союза, принятой на период 2007–2013 гг., содержащие формулировки ряда новых направлений и задач, относящихся к проблематике представления знаний в информационных системах долговременного использования. Результаты анализа этих документов позволяют утверждать, что одновременно с традиционной проблемой информатики, охватывающей вопросы представления в цифровой среде уже имеющихся знаний, ориентированных на удовлетворение технологических, экономических, образовательных и других социально значимых потребностей общества, их хранения и извлечения, становится актуальной задача направляемой генерации новых целевых систем знаний в тех случаях, когда имеющиеся системы знаний не удовлетворяют этим потребностям и с этой точки зрения являются неполными. Предложен новый подход к отражению в цифровой среде личностных и коллективных концептов, а также стадий их эволюции в контексте генерации целевых систем знаний.

Ключевые слова: личностные, коллективные и конвенциональные концепты; стабильные и нестабильные концепты; представление личностных и коллективных концептов в цифровой среде

1 Введение

Согласно Ю. А. Шрейдеру, специфические проблемы информатики возникают там, где встают задачи информационного представления знаний в форме, удобной для обработки, передачи и творческого реконструирования знаний в результате усилий пользователя [1, с. 51]. При этом информатика рассматривалась им согласно парадигме Горна [2, 3] как информационно-компьютерная наука. В настоящее время наряду с традиционной для информатики проблемой представления в цифровой среде уже имеющихся знаний, их хранения и извлечения становится актуальной задача направляемой генерации новых целевых систем знаний в тех случаях, когда имеющиеся системы не удовлетворяют технологическим, экономическим, образовательным и другим социально значимым потребностям и с этой точки зрения являются неполными.

В процессе описания традиционной проблемы представления в цифровой среде уже имеющихся знаний явно или неявно присутствует положительный или отрицательный ответ на следующий основополагающий для информатики и других областей знаний вопрос: «Является ли система знаний субъекта (субъектов) и порождаемые им концепты в разные моменты времени самотождественными?» В случае отрицательного ответа формулируемая проблема является нестационарной, а в случае положительного — стационарной.

Если задавать этот вопрос в рамках других областей знаний, то, согласно М. К. Мамардашвили, в классических основаниях точных и естественных наук заложен, как правило, положительный ответ на этот вопрос. Положительный ответ, который может приниматься в явной или неявной форме, позволяет не учитывать в проблематике представления знаний точных и естественных наук эволюцию знаний субъектов в процессе их познавательной деятельности. В основаниях гуманитарных и социальных наук этот ответ часто бывает отрицательным, так как в широком спектре проблем этих областей знаний требуется учитывать эволюцию знаний субъектов [4].

В работе [5] система базовых терминов и семиотические основания информатики как информационно-компьютерной науки сформулированы таким образом, чтобы охватить оба возможных варианта ответа в предметной области представления знаний. Поэтому в процессе постановки каждой задачи представления знаний в рамках предложенного подхода к описанию семиотических оснований информатики имеется возможность явного или неявного выбора положительного или отрицательного ответа, исходя из специфики задачи.

[1]Институт проблем информатики Российской академии наук, im@a170.ipi.ac.ru
[2]Институт проблем информатики Российской академии наук, valery@a170.ipi.ac.ru
[3]Институт проблем информатики Российской академии наук, koa@a170.ipi.ac.ru

Что касается проблемы направляемой генерации новых целевых систем знаний, то в любом варианте ее описания явно или неявно будет присутствовать отрицательный ответ на сформулированный вопрос, так как само название этой проблемы говорит о том, что в ее постановке и методах решения необходимо учитывать эволюцию системы знаний субъектов и порождаемых ими концептов во времени, а также эволюцию во времени форм представления концептов в среде социальных коммуникаций и в цифровой среде[1]. Иначе говоря, проблема направляемой генерации новых целевых систем знаний является нестационарной.

В статье будут приведены два примера направляемой генерации целевых систем знаний, которые иллюстрируют необходимость учета фактора нестационарности. Первый пример — это формирование и ведение патентной классификации с учетом эволюции во времени ее семантики и смыслового содержания терминов, используемых в патентных рубриках. Второй пример — это информационный мониторинг, анализ и индикаторное оценивание научной деятельности с учетом эволюции во времени смыслового содержания индикаторов, используемых в процессе оценивания.

В настоящее время обозначилась потребность в разработке информационных систем долговременного использования, в процессе функционирования которых необходимо учитывать и фиксировать результаты разных стадий генерации и использования целевых систем знаний. Предполагается, что в информационных системах на разных стадиях их генерации могут находить свое отражение личностные и коллективные знания (личностные и коллективные концепты) создателей целевых систем знаний.

Здесь возникает естественный вопрос: чем личностные и коллективные знания и концепты отличаются от конвенциональных (общепринятых) знаний и, соответственно, от конвенциональных концептов? Как сформулировать эти отличия в такой конструктивной форме, которая позволила бы разработчикам информационных систем зафиксировать и использовать эти признаки с целью различения разных категорий концептов?

Если отвлечься от задачи формулировки в конструктивной форме признаков отличий личностных и коллективных знаний и концептов от конвенциональных и обратиться к примерам описания самого феномена личностного знания, то такое описание можно найти, например, в монографии М. Полани. В его теории неявного знания (tacit knowledge) ряд положений посвящен личностному означиванию, т. е. формированию личностных значений (концептов). В них говорится: «. . . личностное значение является самообосновывающим, если только признается его личностный характер. Оно как бы выдает «вексель» на определенные условия артикуляции[2], которые обязаны стать явными в ходе наших рефлексивных размышлений о самом этом процессе «субсидирования» доверия» [6].

Теория неявного знания сама по себе не дает разработчикам информационных систем четких признаков отличий личностных и коллективных концептов от конвенциональных, но ее можно попытаться использовать как отправную точку для формулировки этих признаков в конструктивной форме. В статье предпринята попытка разделить концепты на три категории — личностные, коллективные и конвенциональные, сформулировав их отличия на основе ряда признаков, и описать терминологически предлагаемую категоризацию концептов. Такое терминологическое описание предпринято в интересах создания тех информационных систем долговременного использования, в задачи которых входит представление концептов всех трех категорий и целевых систем знаний в цифровой среде, а также отражение их эволюции во времени.

Рассматриваются только те информационные системы долговременного использования, которые создаются в интересах обеспечения функционирования тех или иных институциональных систем. Термин *институциональная система* трактуется в соответствии с монографией Г. Б. Клейнера как совокупность взаимосвязанных институтов[3] [7, с. 7]. Одной из задач подобных информационных систем часто является представление в цифровой среде целевых систем знаний, формируемых в интересах обеспечения функционирования соответствующих институциональных систем.

Например, в интересах патентной сферы как институциональной системы была сформирована и ведется Международная патентная классифика-

[1] Цифровая среда определяется в статье как сочетание элементов цифровой вычислительной техники, средств телекоммуникации, информационно-компьютерных систем, иных цифровых средств ввода, хранения, поиска, передачи и других процессов обработки данных.

[2] Артикуляция здесь трактуется как членение системы знаний на концепты.

[3] Институты Г. Б. Клейнер определяет как относительно устойчивые по отношению к изменению поведения или интересов отдельных субъектов и их групп, а также продолжающие действовать в течение значимого периода времени формальные и неформальные нормы либо системы норм, регулирующие принятие решений, деятельность и взаимодействие социально-экономических субъектов (физических и юридических лиц, организаций) и их групп [7, с. 19].

ция (МПК) как целевая и эволюционирующая система знаний. При этом патентные электронные библиотеки и другие информационные системы, обеспечивающие представление и хранение МПК в цифровой среде, являются ключевыми компонентами патентной инфраструктуры, так как в соответствии с действующими нормативными актами любому решению о принятии или отклонении патентной заявки должен предшествовать патентный поиск по ее тематике, в процессе которого используется МПК.

Другим примером институциональной системы может служить сфера академических исследований, выполняемых государственными академиями наук в рамках единой Программы фундаментальных научных исследований в соответствии с Федеральным законом от 23 августа 1996 г. № 127-ФЗ «О науке и государственной научно-технической политике» (в редакции Федерального закона от 04.12.2006 № 202-ФЗ). Ключевым субъектом этой институциональной системы является Координационный совет, возглавляемый президентом РАН, а ключевым компонентом инфраструктуры этой сферы — Информационно-технологическая система мониторинга Программы, разрабатываемая в настоящее время специалистами РАН. Одной из задач этой информационной системы является представление экспертных знаний о новых индикаторах и процессе индикаторного оценивания результатов и эффективности выполнения Программы, в том числе представление в цифровой среде личностных и коллективных концептов в рамках целенаправленно формируемой системы экспертных знаний.

Если обратиться к зарубежному опыту, то проблема генерации целевых систем знаний, представления личностных и коллективных концептов в цифровой среде нашла свое отражение в документах 7-й Рамочной программы Европейского Союза. Эти документы содержат формулировки новых направлений и задач, относящихся к проблематике генерации, представления и эволюции систем знаний, и будут проанализированы в статье.

Задачи представления личностных и коллективных концептов рассматриваются в контексте создания информационных систем, являющихся ключевыми компонентами инфраструктуры, обеспечивающей функционирование институциональных систем. Такие информационные системы будем называть институциональными информационными системами (ИИС). Институциональные информационные системы долговременного использования имеют характерные черты, предъявляющие принципиально новые требования к их разработке.

Во-первых, период хранения документов в таких ИИС может многократно превышать цикл жизни одного поколения аппаратно-программных средств их создания и поддержки. Например, выполнение Программы фундаментальных научных исследований государственных академий наук было начато в 2008 г., и она будет регулярно обновляться не реже, чем один раз в 5 лет. За время действия этой долгосрочной программы произойдет смена нескольких поколений аппаратно-программных средств, что необходимо учитывать при проектировании Информационно-технологической системы мониторинга этой программы.

Во-вторых, во время длительного хранения документов могут существенно эволюционировать классификационные системы, тезаурусы и онтологии, используемые в процессах генерации, представления и сохранения систем знаний в ИИС. При этом в процессе их изменения иногда необходимо различать эволюцию *личностных, коллективных* и конвенциональных концептов, проводя различие между *стабильными* (*lasting concepts*) и *нестабильными концептами* (*volatile concepts*) в системах знаний ИИС.

Одна из новых задач, относящихся к проблематике представления знаний, связана именно с необходимостью категоризации концептов в ИИС долговременного использования. В данной статье задача категоризации решается на основе определения выделенных выше курсивом словосочетаний как терминов, дополняющих систему базовых терминов из работы [5]. Основная цель расширения этой системы терминов заключается в том, чтобы в явном виде описать признаки и нормативные аспекты, позволяющие разработчикам ИИС зафиксировать и использовать отличия личностных, коллективных и конвенциональных концептов.

2 Эволюционирующие системы знаний

Наглядным примером эволюционирующей системы технических знаний может служить МПК. Перечислим все ее редакции с указанием даты введения в действие и отметим момент разделения всей совокупности рубрик МПК на базовый и расширенный уровни:

- первая редакция МПК — 01.09.1968;
- вторая редакция МПК — 01.07.1974;
- третья редакция МПК — 01. 01.1980;
- четвертая редакция МПК — 01.01.1985;
- пятая редакция МПК — 01.01.1990;

– шестая редакция МПК — 01.01.1995;
– седьмая редакция МПК — 01.01.2000;
– восьмая редакция МПК — 01.01.2006 (базовый и расширенный уровни);
– девятая редакция МПК — 01.01.2009.

Приведенный перечень свидетельствует о том, что с 1968 по 2006 г. изменения вносились один раз в 5–6 лет, а разделение на базовый и расширенный уровни отсутствовало. Начиная с 2006 г. в патентной сфере вступила в силу очередная (восьмая) редакция МПК (далее по тексту — МПК-8). В процессе подготовки этой редакции МПК как основного средства классифицирования патентных документов были внесены существенные изменения и в ее структуру, и в регламент ее ведения [8].

В отличие от предыдущих 7 редакций, МПК-8 была разделена на рубрики базового и расширенного уровня. Рубрики базового уровня, выражающие стабильные концепты, могут пересматриваться не чаще, чем раз в три года, а рубрики расширенного уровня могут пересматриваться значительно чаще. Начиная с января 2007 г. вновь вводимые или изменяемые рубрики расширенного уровня подготавливаются Специальным подкомитетом Всемирной организации интеллектуальной собственности (ВОИС) по пересмотру расширенного уровня МПК с регулярностью один раз в квартал и передаются на утверждение в Международное бюро ВОИС. Оно за три месяца до ввода в действие доводит их до сведения патентных ведомств, с тем чтобы последние могли принять обеспечительные меры (перевод новых рубрик и подготовку их к публикации, ознакомление экспертов, начало их простановки на публикуемых документах и т. п.). Важно отметить, что для повышения эффективности поиска, наряду с обязательным рубрицированием изобретений в целом, МПК-8 может использоваться и для кодирования отдельных содержательных аспектов описаний изобретений [9].

Рубрики расширенного уровня на этапе рассмотрения Специальным подкомитетом ВОИС и согласования их содержания выражают личностные и/или коллективные концепты участников обсуждения. После принятия согласованного решения и его утверждения Международным бюро ВОИС, ознакомления экспертов со вновь вводимыми или измененными рубриками, начала их использования в процессе рубрицирования и кодирования документов они приобретают конвенциональный характер в пределах патентной сферы как институциональной системы и, следовательно, в рамках патентных информационных систем.

При простановке рубрик расширенного уровня на публикуемых документах одновременно указывается дата (год и месяц), когда данная рубрика была введена в действие. По мере необходимости рубрики расширенного уровня, выражающие нестабильные концепты, могут пересматриваться чаще, чем раз в три года, пока им не будет присвоен статус рубрик базового уровня.

Например, с января 2007 г. была введена рубрика расширенного уровня с кодом A62D 3/00 под названием «Способы обезвреживания или уменьшения вредности химических отравляющих веществ путем их химического изменения», а в рамках этой рубрики с января 2007 г. был определен термин *вредные химические вещества* как вещества химических отходов, которые являются опасными или токсичными для сброса их в обычные муниципальные захоронения.

Итак, начиная с 2006 г. можно наблюдать существенное ускорение темпов эволюции МПК, что выразилось в разделении всех ее рубрик на две категории в зависимости от степени стабильности. При этом почти в два раза сократился период пересмотра стабильных рубрик базового уровня (с 5–6 до 3 лет).

В завершение рассмотрения этого примера отметим, что в МПК как эволюционирующей системе знаний не отражаются личностные и коллективные концепты, которыми оперируют эксперты Специального подкомитета ВОИС на этапе рассмотрения и согласования содержания рубрик МПК. Отражаются и используются в патентных информационных системах только те конвенциональные концепты, которые сформировались как результат обсуждения личностных и коллективных концептов экспертов. Порядок и стадии рассмотрения и согласования содержания рубрик расширенного уровня определяются нормативными документами, что по определению относится к институциональному фактору эволюции МПК. Однако направленность эволюции МПК и сама потребность в существенном изменении регламента ведения МПК начиная с 2006 г. являются следствием ускорения темпов эволюции систем технических знаний.

В литературе уже можно найти примеры описания задач представления личностных и коллективных концептов в виде авторских классификационных систем и онтологий в предметной области наук о Земле [10–13]. Были рассмотрены задачи согласования нескольких авторских и экспертной онтологий, результаты решения которых предлагается использовать в процессе создания Геопространственного семантического веба (Geospatial Semantic Web — GSW). В этих задачах экспертная онтология является представлением знаний коллектива экспертов, а авторские онтологии — пред-

ставлением личностных знаний обычных пользователей, не являющихся экспертами в науках о Земле. Предложен подход, разработаны алгоритмы и программы, позволяющие фиксировать различия между авторскими и экспертной онтологиями. Эти различия систематизированы по нескольким основаниям: различия в научной парадигме, в покрытии целевой предметной области, в объемах используемых концептов (объемах значений), в дефинициях терминов и семантических отношениях между терминами. Возможности предлагаемого подхода были продемонстрированы на примере интерпретации смысла термина "region", различия в понимании которого разными пользователями были выявлены с помощью специальной анкеты [12, 13].

3 Проблематика представления знаний в 7-й Рамочной программе

Приведенные примеры иллюстрируют актуальность категоризации концептов в интересах постановки и решения проблемы направляемой генерации новых целевых систем знаний и их представления в цифровой среде. Эта проблема нашла свое отражение в документах 7-й Рамочной программы ЕС. Настоящий раздел статьи содержит краткое описание приоритетных направлений и проблемы генерации целевых систем знаний в том виде, как они нашли свое отражение в этих документах.

В области информационно-коммуникационных технологий (ИКТ) в этих документах сформулировано восемь приоритетных направлений исследований и разработок, включая в качестве отдельных направлений «Электронные библиотеки и их содержание» и «Перспективные ИКТ» [14, 15]. Именно в этих двух направлениях нашли отражение новые грани проблематики представления и сохранения знаний в цифровой среде. Однако в описаниях приоритетных направлений исследований и разработок, как правило, не определены ключевые термины.

В качестве примера рассмотрим несколько положений, взятых из формулировок приоритетных направлений в области ИКТ. Долгосрочные цели проектов по приоритетному направлению электронных библиотек в «Программе работ по ИКТ на 2007–2008 годы» сформулированы следующим образом [16, с. 36]:

«Создание новых подходов к сохранению информации и представлению знаний человека в цифровой среде на основе перспективных технологий по управлению динамически изменяемыми большими объемами данных, гарантирующих сохранение цифрового контента, выявление и **экспликацию эволюции его семантики**».

Отметим, что в приведенной формулировке говорится о «семантике цифрового контента», но при этом в цитируемом документе отсутствует явное или контекстное определение смысла этого словосочетания.

Проблема представления знаний упоминается еще в одном приоритетном направлении — «Перспективные ИКТ» в теме «ИКТ долговременного применения». Ключевые положения этой темы, имеющие непосредственное отношение к проблеме представления знаний, сформулированы следующим образом [16, с. 63]:

«Разработать новые подходы к представлению и сохранению знаний, ориентированные на долговременный и безотказный доступ к ним в условиях **локальной генерации концептов**, их интеграции, а также глобального использования систем представления и сохранения знаний с учетом контекста и временной эволюции систем. При этом должна быть обеспечена **долговременная устойчивость** систем представления и сохранения знаний в условиях многообразия их использования и **эволюции семантики во времени**».

Таким образом, кроме выявления и экспликации эволюции «семантики цифрового контента» ставится задача учета «локальности генерации концептов», обеспечения «долговременной устойчивости систем представления и сохранения знаний» и «интеграции локально сгенерированных концептов», но опять нигде не определено смысловое содержание используемых словосочетаний. Отметим, что формулировки «Программы работ по ИКТ на 2007–2008 годы» весьма лаконичны, что является следствием жанра этого документа. Иногда смысл целого ряда ключевых положений трудно понять из контекста этого программного документа. Поэтому потенциальные заявители проектов нередко вынуждены обращаться в соответствующий директорат 7-й Рамочной программы ЕС с просьбой пояснить смысл положений программных документов.

Что касается других направлений исследований в рамках проблемы представления знаний, то более подробное их описание можно найти в трудах семинара "Knowledge Anywhere Anytime: "The Social Life of Knowledge", который состоялся 29–30 апреля 2004 г. в Брюсселе [17]. Материалы этого семинара в редуцированном виде использовались при формировании упомянутой «Программы работ по ИКТ».

В этих материалах отмечается, что исследование процессов генерации знаний и образования конвенциональных систем знаний, а также связанных с ними процессов является актуальной проблемой, которая остается во многом нерешенной. Одновременно фиксируется то, что содержание самой этой проблемы со временем эволюционирует. Участники семинара определили четыре актуальных направления исследований в рамках этой эволюционирующей проблемы.

Задачей первого направления является формирование научного понимания того, как знания появляются, каким образом на этот процесс и его результаты влияет совместная деятельность, как формируются конвенциональные системы знаний. Одна из задач этого направления заключается в определении и фиксировании различий в понимании идентичных информационных объектов разными участниками совместной деятельности.

Задачей второго направления является исследование многообразия форм представления одних и тех же концептов, одной и той же системы знаний. Кроме форм представления конвенциональных и стабильных концептов предметом исследования являются формы представления личностных и коллективных концептов, а также процессы возникновения и эволюции нестабильных концептов. В рамках этого направления предполагается выполнение исследований процессов образования конвенциональных концептов на основе личностных и коллективных концептов.

Задачей третьего направления является создание нового поколения интеллектуальных систем, которые должны обеспечить семантическую интероперабельность в процессе совместной работы пользователей этих систем. Если следовать описанию семиуровневой модели интероперабельности Р. Будденберга, то семантическая интероперабельность имеет отношение к следующим двум уровням этой модели: когнитивная интероперабельность (6-й уровень семиуровневой модели) и доктринальная интероперабельность (7-й уровень). В соответствии с определением Р. Будденберга реализация когнитивной интероперабельности в процессе совместной работы предполагает обеспечение согласованного понимания пользователями идентичных информационных объектов, являющихся формами представления концептов. Реализация доктринальной интероперабельности предполагает не только согласованное понимание пользователями информационных объектов, но и обеспечение принятия ими согласованных решений на основе идентичной информации. Естественно, что в интеллектуальных системах нового поколения должна обеспечиваться и технологическая интероперабельность на первых пяти уровнях семиуровневой модели, в том числе на уровнях разделяемых процедур, процессов и данных [18, 19].

В настоящее время доктринальная интероперабельность достигается в основном за счет использования организационных процедур и регламентов согласования личностных концептов в процессе принятия совместных решений.

В рамках третьего направления кроме создания методов и средств поддержки семантической интероперабельности в интеллектуальных системах планируется исследовать вопросы выявления и экспликации основных стадий эволюции систем знаний, представленных в виде классификационных систем, тезаурусов и онтологий, в процессе совместной работы пользователей. При этом не предполагается, что пользователи заранее будут владеть согласованной между ними системой терминов и единым пониманием принципов построения эволюционирующих систем знаний.

Степень новизны интеллектуальных систем представления знаний предлагается оценивать на основе их сравнения с системами «управления знаниями» (knowledge management systems), основанными на редукционистском подходе к трактовке знаний человека, в котором эволюция систем знаний во времени не учитывается, личностные, коллективные и конвенциональные концепты, стабильные и нестабильные концепты не различаются.

Задачей четвертого направления является исследование принципиальных возможностей и средств влияния на формирование необходимых целевых систем знаний в процессе совместной деятельности. При этом предполагается, что имеет место ситуация недостаточности уже имеющихся знаний. Речь идет о формировании новых систем знаний, ориентированных на удовлетворение технологических, экономических, образовательных и других социально значимых потребностей общества. В рамках этого направления предлагается исследовать, какими видами перспективных ИКТ и до какой степени можно оказывать влияние на процесс генерации целевых систем знаний, отвечающих социально значимым потребностям и необходимых для получения запланированных результатов совместной деятельности. Именно возможность оказывать влияние на процесс формирования новых целевых систем знаний является, по мнению участников семинара “Knowledge Anywhere Anytime: “The Social Life of Knowledge”, характерной чертой общества, основанного на знаниях.

Приведенный перечень из четырех направлений исследований говорит о том, что участники семинара существенно расширяют границы предметной области представления знаний, сформулированной

в рамках редукционистского подхода, за счет рассмотрения процессов целенаправленной генерации и эволюции систем знаний во времени. Проведенный в этом разделе анализ позволяет сделать вывод о том, что предлагаемое расширение предметной области происходит за счет следующих направлений исследований:

- разработка методов тестирования уже имеющихся систем знаний с точки зрения проверки их полноты, анализируемой с позиции удовлетворения тех или иных явно выраженных социально значимых потребностей общества;
- целенаправленное воздействие на процессы генерации личностных и коллективных концептов (*направляемое развитие*, в терминах Н. Н. Моисеева [20]);
- направляемая генерация новых целевых систем знаний на основе интеграции личностных и коллективных концептов;
- представление в цифровой среде личностных и коллективных концептов, целевых систем знаний и отражение стадий их эволюции во времени;
- разработка методов распространения и ориентированного использования целевых систем знаний, сгенерированных в интересах удовлетворения соответствующих социально значимых потребностей общества.

В заключение раздела отметим, что в цитируемых материалах семинара акцентируется институциональная и социальная обусловленность процессов направляемой генерации целевых систем знаний, что необходимо учитывать разработчикам в процессе создания ИИС. Это существенно усложняет создание ИИС, целью которых является воздействие на процессы генерации целевых систем знаний средствами ИКТ, так как для разработчиков ИИС институциональные и социальные факторы являются, как правило, внешними ограничениями, которые они не могут изменять [17].

Определяя ключевые направления исследований по тематике представления знаний, в том числе приводя описание проблемы направляемой генерации целевых систем знаний, участники семинара не предложили согласованную систему терминов для описания новых направлений исследований. Пока новые идеи только обсуждаются и определяются направления исследований, подобная ситуация, как правило, неизбежна, так как система терминов для их описания только начинает формироваться. Затем, по мере расширения границ предметной области представления знаний, часто возникает необходимость определять новые термины, используемые для описания новых направлений исследований, и/или уточнять содержание ранее введенных терминов.

4 Категоризация концептов

Сначала рассмотрим систему базовых терминов информатики, построенную на основе результатов работ [21–25] и описанную в [5, 26]. В рамках этой системы терминов разные категории концептов лексически не различались. Основная цель данного раздела статьи заключается в расширении системы базовых терминов с целью разделения концептов на три категории — личностные, коллективные и конвенциональные.

Основываясь на результатах работ [21–25] и понимании семиотического термина *знак* в рамках лингвоцентрического подхода [27, 28], сначала уточним дефиниции системы базовых терминов, которая была определена в работах [5, 26]. При этом будем нумеровать описываемые далее термины в соответствии с рис. 1 против часовой стрелки, выделяя их в тексте курсивом.

1. Термин *знания* будем трактовать как результаты познавательной и креативной целеустремленной деятельности человека, носителем которых может быть только человек и которые могут быть разделены на отдельные «кванты» знаний (в программных документах 7-й Рамочной программы используется словосочетание “knowledge parts”) [23, с. 33; 24, с. 185; 29, с. 814].

2. В процесах представления и сохранения знаний основное внимание будет уделяться тем «квантам» знаний, называемым *концептами*, которые являются элементарными единицами или сочетаниями элементарных единиц плана содержания[1], выражаемого в рамках некоторого естественного языка (в общем случае, в рамках той или иной знаковой системы).

   Приведенное определение термина *концепты* предполагает, что они являются результатом процесса членения знаний человека на «кванты», которые могут быть выражены в рамках некоторой знаковой системы. Процесс членения неразрывно связан с процессом выражения знаний человека в сенсорно воспринимаемой и отчужденной от человека форме, например в виде текста на естественном языке, диаграммы или в виде геоизображения на языке карты.

[1]Иначе говоря, план содержания естественного языка (знаковой системы) представляет собой совокупность тех «квантов знаний», которые могут быть выражены средствами этого языка (знаковой системы).

Рис. 1 Система базовых терминов [5]

Система знаний человека может включать один или более планов содержания, что определяется числом естественных языков или иных знаковых систем, которыми он владеет и пользуется для представления своих знаний в отчужденной форме. Согласно атомистическому подходу к описанию семантики, минимальные единицы плана содержания, имеющие значение в рамках некоторого естественного языка или иной знаковой системы, будем называть *элементарными концептами* этой знаковой системы [30].

Отметим следующую особенность описываемой системы терминов: выделение в системе знаний человека нескольких планов содержания позволяет учесть различия в членении знаний человека в разных естественных языках и других знаковых системах, что необходимо, например, в процессе проектирования многоязычных семантических словарей и тезаурусов.

Здесь также необходимо отметить разницу между элементарным концептом и значением слова или устойчивого словосочетания, когда речь идет о естественном языке. В процессах представления научно-технических знаний в отчужденной форме рассматривается только сигнификативный аспект значения. Экспрессивно-эмоциональные оценки, идиомы и коннотации не рассматриваются, так как речь идет о кодировании научно-технических знаний в цифровой среде ИИС, а не о создании художественных произведений. Следовательно, термины *концепт* и *элементарный концепт* используются для обозначения только сигнификативной составляющей значений слов и словосочетаний [31, 32].

3. Термин *информация* определим как формы эксплицитного и отчужденного от человека представления его знаний, предназначенные для передачи, непосредственного сенсорного восприятия и понимания их другими людьми. Такое определение, по сути, является выбором из некоторого множества одной дефиниции в рамках позиционирования слова *информация* как омонима, охватывающего широкий спектр феноменов, соответствующих им дефиниций и смысловых значений [33].

4. Согласно П. Ингверсену, термин *знаковая информация* определим как результат процесса представления концептов человеком в плане выражения[1] в любой знаковой форме, которая является сенсорно воспринимаемой и понимаемой другими участниками коммуникаций [34]. Отметим, что при таком определении термин *знаковая информация* имеет отношение только к знаковым формам представления концептов, а введенный ранее термин *информация* — к любым формам представления любых «квантов» знаний, включая концепты.

5. Существительные *информация* и *знаковая информация* являются неисчисляемыми, что не всегда удобно для описания процессов представления знаний в цифровой среде ИИС. Поэтому определим следующие термины с использованием исчисляемых существительных. Представление в плане выражения элементарных концептов в виде сенсорно воспринимаемых знаковых форм будем называть *элементарными информационными объектами*. Сенсорно воспринимаемые знаковые формы представления любых концептов, включая элементар-

[1] Иначе говоря, план содержания представляет собой последовательности литер слов, фраз и текста на естественном языке, рисунки и другие формы представления концептов в среде социальных коммуникаций.

ные, будем называть *информационными объектами*.

6. Термин *коды* определим как компьютерные эквиваленты двоичных цифр (или их последовательностей), которые могут представлять собой намагниченность или ее отсутствие, наличие электрического тока или его отсутствие, способность к отражению света или ее отсутствие, в цифровой среде [24, с. 86]. Отметим, что литеры «0» и «1», о которых говорится в определении термина «коды», по определению являются сущностями среды социальных коммуникаций ИИС, а их компьютерные эквиваленты — цифровой среды ИИС.

   В соответствии с работой [26] предлагается выделять среди всех возможных кодов цифровой среды ИИС три следующие категории:

   – коды, соотнесенные с концептами знаний человека-пользователя ИИС (*коды первой категории ИИС*);

   – коды, соотнесенные с эксплицитными и отчужденными от человека сенсорно воспринимаемыми знаковыми формами представления концептов в плане выражения среды социальных коммуникаций ИИС (*коды второй категории ИИС*);

   – коды, идентифицирующие программы и используемые ими информационные ресурсы, а также другие категории денотатов цифровой среды ИИС (*коды третьей категории ИИС*).

   Например, для кодирования смысла индикаторов в системе информационного мониторинга сферы науки, которая является примером ИИС, предполагается использовать коды первой категории, для кодирования названий индикаторов — коды второй категории, а для кодирования программ вычисления значений индикаторов и используемых этими программами информационных ресурсов — коды третьей категории.

7. В соответствии с работой [26] определим еще одну *нулевую категорию кодов*, которую будем называть *цифровыми данными* и к которой будем относить все коды цифровой среды ИИС, не относящиеся к трем выше определенным категориям. Таким образом, к цифровым данным будем относить, в частности, результаты любых измерений, полученных с помощью цифровых технических систем (устройств) и хранящихся в ИИС, и результаты любых вычислений, не являющихся итогом представления знаний человека-пользователя ИИС.

8. Представление цифровых данных в среде социальных коммуникаций ИИС в сенсорно воспринимаемой форме будем называть *данными*.

9. Смысл термина *ментальные образы данных* следует из его названия и рис. 1.

Разделим рассмотренные термины на три группы в зависимости от природы обозначаемых ими сущностей, т. е. среды или сферы, к которой они принадлежат (ментальная сфера, социальная или цифровая среда) [25]:

– знания, ментальные образы сенсорно воспринимаемых данных, концепты и элементарные концепты (сущности ментальной сферы пользователей ИИС);

– информация, данные, знаковая информация, информационные объекты, элементарные информационные объекты (сущности среды социальных коммуникаций пользователей ИИС);

– цифровые данные и три категории кодов (сущности цифровой среды ИИС).

В указанной системе терминов нас интересуют концепты и элементарные концепты, обозначенные на рис. 1 номером 2, которые предлагается разделить на три разные категории. Прежде чем провести категоризацию концептов, отметим, что дефиниция термина концепт существенно зависит от термина *знаковая система*, так как по определению любой концепт соотнесен с той или иной знаковой системой. Однако по определению семиотического знака и знаковой системы две стороны знака (форма — материально выраженная его составляющая — и значение — его идеальная составляющая), будучи поставлены в отношение **постоянной связи**, опосредованной сознанием, составляют **устойчивое** единство, которое посредством сенсорно воспринимаемой формы знака репрезентирует **конвенционально** приданное ему значение [35].

Следовательно, в рамках системы базовых терминов из работы [26] приведенное определение термина *концепты* по своему генезису (происхождению) предполагает, что они являются, с одной стороны, конвенциональными, с другой стороны, образуют устойчивое единство с формой их представления. Таким образом, чтобы определить личностные и коллективные концепты, а также нестабильные концепты пользователей ИИС, сначала необходимо расширить понятие знаковой системы, сняв обязательные условия конвенциональности значения и устойчивого единства концептов и форм их представления. Одним из возможных способов расширения понятия знаковой системы является введение понятий *авторского и коллективного знаков*.

Авторский знак отличается от традиционного семиотического знака тем, что две стороны авторского знака — форма и значение — могут составлять стабильное или нестабильное единство, т. е. находиться в отношении долгосрочной или временной связи, опосредованной сознанием **одного пользователя ИИС**, которая посредством сенсорно воспринимаемой формы знака этим пользователем репрезентирует **персонально** приданное ему значение в течение периода времени, указанного при регистрации авторского знака в ИИС.

Коллективный знак отличается от авторского тем, что две стороны коллективного знака могут находиться в отношении долгосрочной или временной связи, опосредованной сознанием каждого из участников совместной деятельности (членов коллектива), являющихся пользователями ИИС, и согласованной между ними, т. е. используют и **согласованно понимают** коллективные знаки как минимум **два человека**.

Таким образом, одним из ключевых признаков отличия коллективного знака от авторского является согласованность его смыслового содержания между всеми членами коллектива. Оставим пока открытым вопрос о признаках отличия коллективного знака (концепта) от конвенционального и стабильного знака (концепта) от нестабильного.

Важно отметить, что, допуская возможность временной связи формы и значения у одного пользователя или ограниченной группы пользователей, т. е. снимая обязательные условия конвенциональности значения и устойчивого единства концептов (значений знаков) и форм их представления, мы тем самым говорим об их недоступности для всех, кроме их авторов. Чтобы расширить круг пользователей авторских и коллективных знаков, предлагается регистрировать их в ИИС, фиксируя значения и формы авторских и коллективных знаков с помощью средств лингвистического обеспечения ИИС. Например, в тезаурусе ИИС можно указывать их авторов, а также период времени, в течение которого сенсорно воспринимаемая форма знака репрезентирует персонально или коллективно приданное ему значение. Если авторы проделают эту работу, то для других пользователей ИИС станут доступны их формы, описание их значений, информация об авторах, а также период времени, в течение которого установлена связь между формой и содержанием для авторских и коллективных знаков.

Авторские и коллективные знаки не возникают и тем более не функционируют раздельно. В своей совокупности с традиционными семиотическими знаками они образуют единую систему, которую будем называть *знаковой системой ИИС*. Иначе говоря, авторские и коллективные знаки регистрируются в ИИС ее пользователями и означиваются в совокупности с традиционными конвенциональными знаками, являющимися составными элементами естественного языка или иной знаковой системы[1].

Используя определения авторского и коллективного знаков, можно предложить следующий подход к категоризации концептов. Определим *элементарный личностный концепт* как значение авторского знака, а *личностный концепт* — как значение выражения на естественном языке или в рамках некоторой знаковой системы, содержащего хотя бы один авторский знак, либо новое значение выражения без авторских знаков, смысл которого определен автором в явном виде и зарегистрирован в ИИС.

Определим *элементарный коллективный концепт* как значение коллективного знака, а *коллективный концепт* — как значение выражения на естественном языке или некоторой знаковой системы, содержащего хотя бы один коллективный знак, либо новое значение выражения без коллективных знаков, смысл которого определен в явном виде, согласованно понимается как минимум двумя участниками совместной деятельности и зарегистрирован в ИИС.

Представлением личностного концепта в цифровой среде будем называть код(ы) первой категории ИИС, которые автор этого концепта выбрал и поставил в соответствие этому концепту в результате выполнения процедуры регистрации соответствующего авторского знака в ИИС.

Представлением коллективного концепта в цифровой среде будем называть код(ы) первой категории ИИС, которые участники совместной деятельности согласованно выбрали и поставили в соответствие этому концепту в результате выполнения процедуры регистрации соответствующего коллективного знака в ИИС.

Рассмотренные дефиниции трех категорий кодов, личностного и коллективного концептов могут служить основой для формирования тех цифровых сущностей, которые и являются «представителями» личностных и коллективных концептов в цифровой среде. Это дает возможность использовать предлагаемую категоризацию концептов при решении задач формирования целевых систем знаний и отражать эволюцию семантики личностных и коллективных концептов этой системы в цифровой среде, используя результаты выполнения процеду-

[1] Приведенные определения авторского и коллективного знаков основаны на результатах работы [35].

ры их регистрации в ИИС. Пример прикладного использования категоризации и регистрации личностных и коллективных концептов приводится в следующем разделе.

Теперь определим границу между *стабильными* (lasting) и *нестабильными концептами* (volatile concepts). В дефинициях авторских и коллективных знаков говорится, что их формы и значения могут составлять стабильное или нестабильное единство. При этом период времени долгосрочной или временной связи между ними указывается при регистрации в ИИС. В качестве основного признака отличия стабильных концептов от нестабильных возьмем критерий, который использовался для разделения МПК-8 на рубрики базового и расширенного уровней. Как отмечалось во втором разделе, рубрики базового уровня, выражающие стабильные концепты, могут пересматриваться не чаще, чем раз в три года, а рубрики расширенного уровня могут пересматриваться значительно чаще.

Взяв за основу этот критерий, будем говорить, что в рамках некоторой институциональной системы существует граница между стабильными и нестабильными концептами, если выполнены следующие условия:

- в этой институциональной системе имеется нормативный документ, определяющий период времени, превышение которого для зарегистрированной связи между формой и значением (концептом) знака говорит о стабильности этого знака и соответствующего ему концепта, или содержащий перечень стабильных знаков, например в виде списка рубрик базового уровня системы классификации или базовых дескрипторов тезауруса;
- средства лингвистического обеспечения ИИС содержат информацию о периоде времени зарегистрированной связи между формой и значением (концептом) знака.

Иначе говоря, должна быть определена хотя бы одна система классификации (или тезаурус), отражающие эти концепты, в которой разделены стабильные и нестабильные рубрики системы классификации (стабильные и нестабильные дескрипторы тезауруса).

Осталась неопределенной граница между коллективными и традиционными конвенциональными знаками, а также соответствующими им концептами. По своей сути эта граница является весьма размытой. Однако для разработчиков ИИС важно иметь некоторый признак, позволяющий различать коллективные и конвенциональные знаки и концепты. Поэтому при создании конкретных систем предлагается эту границу фиксировать с помощью нормативного документа.

5 Примеры описания личностных концептов

Категоризация концептов, которая предложена в предыдущем разделе, в настоящее время используется в процессе создания системы индикаторов для оценивания результатов и эффективности реализации НИОКР Программы фундаментальных научных исследований РАН на 2008–2012 гг., а также потенциала научных коллективов, выполняющих НИОКР.

Создаваемая система включает широкий спектр индикаторов, традиционно используемых в сфере науки, а также ряд новых. Пример традиционного индикатора приведен на рис. 2, на котором дано возрастное распределение сотрудников одного из научных коллективов ИПИ РАН. На этом рисунке приведены значения индикатора в виде графика процентного распределения сотрудников по 14 возрастным группам (20–24, 25–29 и далее до группы 85–89 лет). Отметим, что все графики на рис. 2 и 3 по определению являются дискретными, но для наглядности все 14 значений соединены отрезками.

Для каждой группы сотрудников вычисляется ее доля в общей численности научного коллектива. На рисунке видно, что для этого коллектива наибольшую долю имеет возрастная группа 45–49 лет (24%). Далее по убыванию следуют шесть возрастных групп: 40–44 года (20,6%), 25–29 лет (10,3%), 30–34 года, 50–54 года, 55–59 лет и 75–79 лет (по 6,9%). Затем идут пять возрастных групп с долей в 3,5%: 35–39 лет, 60–64 года, 65–69 лет, 70–74 года и 80–84 года. Представители двух оставшихся возрастных групп в этом коллективе отсутствуют: 20–24 года и 85–89 лет.

Для определения публикационной активности каждой из возрастных групп была предпринята попытка создать новый индикатор и разработать программу вычисления значений этого индикатора. С этой целью были привлечены специалисты, которые имели разные представления о создаваемом индикаторе.

Первое определение индикатора, предложенное одним из авторов этой статьи (далее по тексту — первый специалист), охватывало все публикации каждой возрастной группы. Учитывались все публикации, зарегистрированные в течение заданного периода времени в базе данных проектируемой Информационно-технологической системы мониторинга Программы фундаментальных научных исследований. Если у публикации было несколько

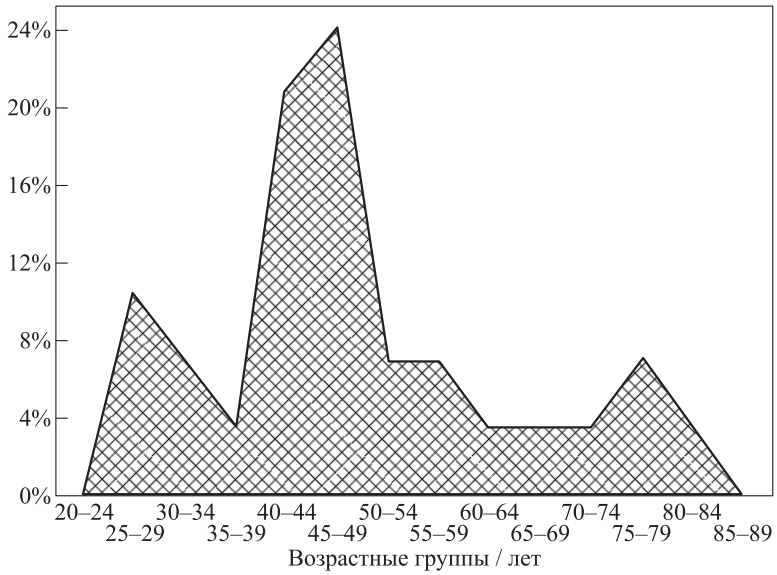

Рис. 2 График возрастного распределения сотрудников научного коллектива

авторов, то соответствующим возрастным группам добавлялось по единице. На основе личностного понимания нового индикатора первым специалистом была разработана программа вычисления его значений (см. рис. 3) для сотрудников научного коллектива, возрастное распределение которого приведено на рис. 2.

Для каждой группы сотрудников была вычислена ее доля статей в общей численности статей научного коллектива с учетом того, что в случае нескольких авторов у статьи соответствующим возрастным группам добавлялось по единице. На рисунке видно, что для этого коллектива наибольшую долю статей имеет возрастная группа 75–79 лет (39,7%). Далее по убыванию следуют семь возрастных групп: 40–44 года (24,1%), 45–49 лет (13,1%), 55–59 лет (10,1%), 25–29 лет (6,5%), 30–34 года, 35–39 лет и 50–54 года (по 1,5%). Затем идут две возрастные группы с долей в 1,0%: 70–74 года и 80–84 года. Представители двух возрастных групп в коллективе отсутствуют: 20–24 года и 85–89 лет, а две возрастные группы не имеют публикаций: 60–64 года и 65–69 лет.

На следующем этапе в процесс создания нового индикатора включился второй специалист, который предложил внести следующее изменение в определение публикационной активности возрастных групп: если в публикации указано N авторов, то соответствующим возрастным группам добавлялось не по единице, как на рис. 3, а по $1/N$. Этот график в статье не приводится. Укажем только его значения для первых пяти возрастных групп из 14: 75–79 лет (47%), 40–44 года (22,6%), 55–59 лет (9,5%), 45–49 лет (9,1%), 25–29 лет (6,3%). Из сравнения этих значений с графиком на рис. 3 следует, что из пяти перечисленных возрастных групп доля первой увеличилась, а остальных четырех групп уменьшилась. Причина в том, что в первой группе значительная часть публикаций не имеет соавторов, а в остальных четырех группах — наоборот.

Предлагаемое изменение программы вычисления значений нового индикатора мотивировалось следующим примером: если два автора, составляющих группу 50–54 года[1], в течение заданного периода времени опубликуют вдвоем в соавторстве 5 статей, а один автор в группе 55–59 лет[2] опубликует без соавторов тоже 5 статей, то в соответствии с первым вариантом программы публикационная активность первой возрастной группы будет в два раза выше, чем второй группы, хотя в обеих группах будет опубликовано одинаковое число статей — по 5.

Однако первый специалист не согласился с этим доводом и привел следующий пример: если два соавтора в течение заданного периода времени опубликуют вдвоем в соавторстве 5 статей, при этом первый соавтор относится к группе 45–49 лет, второй соавтор — к группе 50–54 года, а еще один автор в группе 55–59 лет[3] опубликует без соавторов тоже 5 статей, то в соответствии со вторым вариан-

[1] Предполагается, что в возрастной группе 50–54 года присутствуют только два человека.
[2] Предполагается, что в возрастной группе 55–59 лет присутствует только один человек.
[3] Предполагается, что в каждой из трех возрастных групп присутствует по одному человеку.

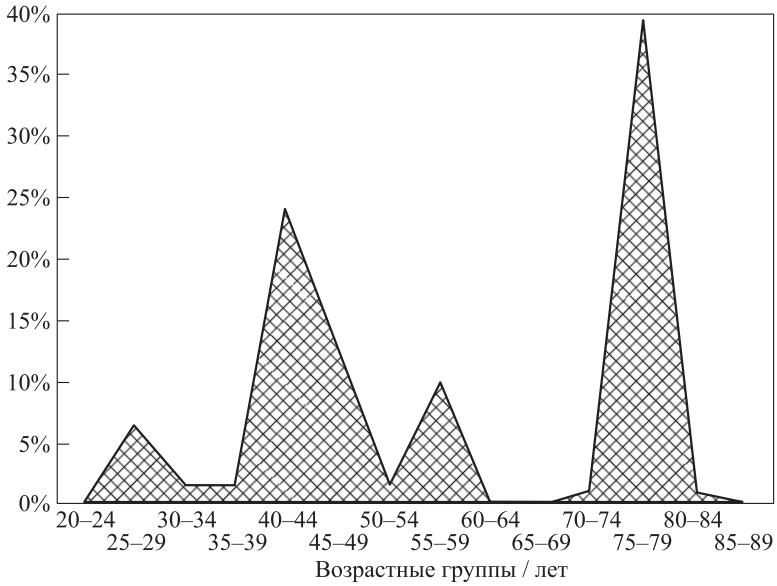

Рис. 3 График распределения публикационной активности возрастных групп

том программы публикационная активность первых двух возрастных группы будет в два раза ниже, чем третьей группы, хотя во всех трех группах будет опубликовано одинаковое число статей — по 5.

Итогом второго этапа создания нового индикатора стали два разных личностных понимания этого индикатора каждым из двух специалистов. При этом было отмечено, что предложенные личностные концепты обладают общим недостатком, так как в обоих вариантах программы их вычисления не учитывается численность возрастных групп, для которых сопоставляется публикационная активность. Например, наибольшую численность сотрудников имеет возрастная группа 45–49 лет (24%), а доля группы 75–79 лет составляет 6,9%. В соответствии с первым личностным концептом публикационная активность возрастной группы 75–79 лет равна 39,7%, а группы 45–49 лет — 13,1%. Таким образом, если не учитывать численность возрастных групп, то публикационная активность первой группы превышает активность второй группы более чем в 3 раза. Однако с учетом численности это превышение составит 10,5 раз. Еще один отмеченный недостаток заключается в том, что не учитывается рейтинг (импакт-факторы) источников публикаций.

На последующих этапах построения этого индикатора планируется разработать еще несколько вариантов программы вычисления его значений, устраняющие эти недостатки, что может привести к увеличению числа разных личностных концептов, соответствующих различным вариантам понимания нового индикатора. Важно отметить, что различным вариантам понимания могут соответствовать не только разные варианты алгоритма вычисления значений индикатора, например при учете соавторов и/или численности возрастной группы, но и разные информационные ресурсы, например списки журналов ВАК или Российского индекса научного цитирования.

В процессе совместной деятельности и расширения группы специалистов, участвующих в создании нового индикатора, могут появляться и коллективные концепты. Конвенциональный концепт появится только в тот момент, когда в рамках соответствующей институциональной системы будет принят нормативный документ, определяющий именно тот вариант определения индикатора, который и должен будет применяться для оценивания публикационной активности научных коллективов и организаций. При этом в ИИС сохранится вся история эволюции личностных, коллективных и конвенциональных концептов, имеющих отношение к созданию нового индикатора.

Ранее на примере патентной сферы уже была описана аналогичная процедура формирования конвенциональных концептов рубрик МПК. В этой процедуре документ Международного бюро ВОИС определяет дату ввода в действие новых и измененных расширенных рубрик МПК и связанных с ними терминов, а за три месяца до этой даты доводит их до сведения патентных ведомств, с тем чтобы последние могли принять обеспечительные меры. Однако в соответствующих патентных ИИС фиксируется история эволюции только конвенциональных концептов. История эволюции личностных и коллективных концептов в них не отражается.

6 Отражение эволюции концептов в ИИС

В данной статье предлагается следующий подход к отражению эволюции личностных и коллективных концептов в ИИС. С целью регистрации трех категорий концептов (личностные, коллективные и конвенциональные) все дескрипторы тезауруса ИИС предлагается разделить также на три категории. Что касается создания конвенциональных дескрипторов, то они традиционно формируются лингвистами, как правило, на основе корпуса текстов нормативных документов, толковых и других видов словарей.

Предполагается, что личностные концепты будут описывать и представлять в виде дескрипторов тезауруса их авторы в соответствии с регламентом ИИС. При этом каждый автор будет иметь возможность дополнительно отразить эволюцию личностных концептов во времени в виде серии вариантов авторских дескрипторов тезауруса ИИС. Отметим, что в примере с созданием нового индикатора эти дескрипторы должны быть связаны с вариантами программы его вычисления и используемыми этой программой информационными ресурсами [36, 37].

Чтобы отличить авторские дескрипторы от других категорий дескрипторов, предлагается помечать их как персональные, т. е. ввести для них атрибут «автор дескриптора». Кроме того, предлагается использовать атрибуты, фиксирующие моменты времени:

- включения авторских дескрипторов и их вариантов в тезаурус ИИС;
- установления их связей с другими дескрипторами, программами, информационными ресурсами и компонентами других видов обеспечения ИИС.

Это позволит, анализируя варианты и значения атрибутов, определять авторов новых индикаторов, находить варианты программ вычисления их значений и оценивать эволюцию семантики дескрипторов во времени.

Аналогично в процессе совместной деятельности предлагается описывать и помечать коллективные дескрипторы с помощью атрибутов коллективного авторства, фиксировать их варианты, моменты времени включения в тезаурус ИИС и варианты программ вычисления их значений.

С методологической точки зрения приведенное описание процедуры построения авторских и коллективных дескрипторов тезауруса ИИС и отражения эволюции концептов во многом основано на концепции противопоставления и связи двух планов, или аспектов, изучения мышления — плана образов (или знаний) и плана «процессов» (или деятельности), предложенной Г. П. Щедровицким и Н. Г. Алексеевым. Ими была сформулирована задача «операционально-деятельностного анализа понятий и знаний, позволяющего, исходя из формы какого-либо сложившегося понятия или знания, сводить его к системе операций и действий, порождающих содержание этого понятия или знания» [38, 39].

Предлагаемый в статье подход к описанию концептуализации денотатов, например программ вычисления индикаторов, с помощью авторских и коллективных дескрипторов тезауруса ИИС имеет два существенных отличия от концепции Г. П. Щедровицкого и Н. Г. Алексеева. Во-первых, в нем учитывается то, что для личностных и коллективных концептов могут отсутствать конвенциональные эксплицитные формы представления, что компенсируется их описанием в ИИС. Во-вторых, в нем учитывается то, что объемы значений личностных и коллективных концептов могут эволюционировать во времени, и предложен подход к отражению этой эволюции в эксплицитной форме с помощью средств лингвистического обеспечения ИИС.

Отметим еще один существенный аспект. Приведенные примеры использования предлагаемой системы терминов в процессе проектирования новых индикаторов иллюстрируют необходимость введения кодов третьей категории для денотатов. Как видно из примеров раздела 5, новые индикаторы на начальных стадиях их разработки еще не имеют общепринятых названий, отсутствует согласованное понимание их экспертами и другими пользователями ИИС. При этом уже разработаны варианты программ их вычисления. Поэтому в тех случаях, когда нет согласованных названий и концептов, для представления личностных и коллективных концептов в цифровой среде ИИС, а также для экспликации эволюции семантики новых индикаторов удобно использовать в качестве кодов денотатов **уникальные идентификаторы** вариантов программ для вычисления их значений и используемых этими программами других ресурсов ИИС.

Другим примером, иллюстрирующим возможности практического использования кодов денотатов в задачах формирования целевых систем знаний, является Каталог программных ошибок (The Common Weakness Enumeration — CWE), записи которого связаны между собой тезаурусными отношениями. Каждая запись этого каталога, как правило, включает дефиницию, фрагмент программного кода, который иллюстрирует смысловое содержание ошибки, родовидовые и иные тезаурусные отношения с другими описаниями ошибок. В процессе

разработки подобных каталогов идентификаторы фрагментов программного кода, иллюстрирующего дефекты программ, могут использоваться в качестве кодов денотатов. Иначе говоря, для идентификации программных ошибок на начальной стадии их описания, когда они еще не имеют общепринятых названий, удобно использовать коды денотатов в процессе описания смысла этих ошибок и выбора для них названий [40].

В заключение отметим, что новые грани проблематики представления знаний в цифровой среде, для описания которых предложены новые термины, формируются под воздействием общественно значимых потребностей в формировании целевых систем знаний в тех случаях, когда отсутствуют готовые системы знаний. Процессы направляемой генерации целевых систем знаний можно наблюдать в сфере науки, индустрии программных средств и патентной сфере как ответную реакцию на новые вызовы и общественно значимые потребности.

7 Заключение

Предложен новый подход к отражению в цифровой среде процессов генерации личностных и коллективных концептов, а также стадий их эволюции. Согласно Н. Н. Моисееву, использование цифровой среды в процессах генерации и эволюции систем знаний рождает новые системные свойства, не выводимые из свойств отдельных разумов. Их потенциальные возможности не зависят от желания и действий отдельных людей, это — результат самоорганизации и направляемого развития [20].

Вопросы самоорганизации и направляемого развития целевых систем знаний выходят за рамки настоящей статьи. В ней были рассмотрены только вопросы категоризации концептов и сформулированы признаки, позволяющие разработчикам ИИС зафиксировать и использовать отличия личностных, коллективных и конвенциональных концептов при описании процессов генерации и эволюции целевых систем знаний.

Приведенное в разделах 5 и 6 описание процедуры построения новых индикаторов и отражения эволюции соответствующих концептов в виде совокупности вариантов авторских и коллективных дескрипторов тезауруса ИИС иллюстрирует возможности практического применения предлагаемого подхода, описанного с помощью терминов, лексически фиксирующих предлагаемое разделение концептов на три категории. Это разделение концептов увеличивает число базовых терминов с 9 (см. рис. 1) до 12, так как кроме традиционного термина *концепт* появляются еще три термина — *личностный, коллективный и конвенциональный концепты*.

Предлагаемое расширение числа базовых терминов не является чисто терминологическим вопросом, так как появление новых понятий и терминов, которые они обозначают, является следствием постановки новой и актуальной проблемы генерации целевых систем знаний, а также становления новых направлений исследований и разработок, относящихся к проблематике представления знаний в цифровой среде.

Главный результат предлагаемой категоризации концептов заключается в том, что лексически акцентируются различия между тремя категориями концептов, их отличительные признаки и особенности стадий формирования, эволюции концептов и целевых систем знаний, ориентированных на удовлетворение технологических, экономических, образовательных и других социально значимых потребностей общества. Иначе говоря, наряду с традиционной проблемой извлечения уже имеющихся знаний, отвечающих социально значимым потребностям, лексически акцентируется внимание на новой проблеме формирования целевых систем знаний в тех случаях, когда отсутствуют готовые системы знаний. При этом в интересах описания данной проблемы предложено развитие ранее рассмотренной системы базовых терминов, дано описание признаков, разделяющих концепты на три категории, и процедуры отражения стадий эволюции концептов в цифровой среде ИИС.

Литература

1. *Шрейдер Ю. А.* Информация и знание // Системная концепция информационных процессов. — М.: ВНИИСИ, 1988. С. 47–52.
2. *Gorn S.* The computer and information sciences: A new basic discipline // SIAM Review, 1963. Vol. 5. No. 2. P. 150–155.
3. *Gorn S.* Informatics (computer and information science): Its ideology, methodology, and sociology // The studies of information: Interdisciplinary messages / Eds. F. Machlup, U. Mansfield. — N. Y.: Wiley, 1983. P. 121–140.
4. *Мамардашвили М. К.* Классический и неклассический идеалы рациональности. — Тбилиси: Мецниереба, 1984.
5. *Зацман И. М.* Семиотические основания и элементарные технологии информатики // Информационные технологии, 2005. № 7. С. 18–31.
6. *Полани М.* Личностное знание. — М.: Прогресс, 1985. С. 257–258.
7. *Клейнер Г. Б.* Эволюция институциональных систем. — М.: Наука, 2004.

8. Введение в МПК-8. http://www.fips.ru/ipc8/intro/mpk8.htm.
9. О новом порядке пересмотра и реализации МПК расширенного уровня. http://www.fips.ru/russite/classificators/new.htm.
10. *Зацман И. М.* Вербально-образное представление знаний в электронных библиотеках (Часть II) // Научно- техническая информация (серия 2 «Информационные процессы и системы»), 2001. № 12. С. 10–17.
11. *Зацман И. М.* Концептуальный поиск и качество информации. — М.: Наука, 2003. 271 с.
12. *Agarwal P.* Contested nature of "place": Knowledge mapping for resolving ontological distinctions between geographical concepts // Eds. M. Egenhofer, C. Freksa, M. Harvey. 3rd International Conference "GIScience 2004". LNCS 3234. — Berlin: Springer-Verlag, 2004. P. 1–21.
13. *Agarwal P., Huang Y., Dimitrova V.* Formal approach to reconciliation of individual ontologies for personalisation of geospatial semantic Web // Eds. M. Rodriguez, I. Cruz, M. Egenhofer, S. Levashkin. 1st International Conference "GeoS 2005". LNCS 3799. — Berlin: Springer-Verlag, 2005. P. 195–210.
14. Decision No. 1982/2006/EC of the European Parliament and of the Council of 18 December 2006 concerning the Seventh Framework Programme of the European Community for research, technological development and demonstration activities (2007–2013) // Official J. of the European Union L412, 30.12.2006. P. 1–41.
15. CORDIS ICT Programme Home. http://cordis.europa.eu/fp7/ict/programme/home_en.html.
16. ICT FP7 Work Programme. ftp://ftp.cordis.europa.eu/pub/fp7/ict/docs/ict-wp-2007-08_en.pdf.
17. FP7 Exploratory Workshop 4 "Knowledge Anywhere Anytime." http://cordis.europa.eu/ist/directorate_f/f_ws4.htm.
18. *Buddenberg R.* Toward an interoperability reference model. http:// web1.nps.navy.mil/~budden/lecture.notes/interop_RM.html.
19. *Buddenberg R.* FORCENet: We've been here before. http://web1.nps.navy.mil/~budden/lecture.notes/it_arch/large_info_systems.html.
20. *Моисеев Н. Н.* Универсум. Информация. Общество. — М.: Устойчивый мир, 2001.
21. *Nonaka I.* The knowledge-creating company // Harvard Business Review, 1991. Vol. 69. No. 6. P. 96–104.
22. *Nonaka I., Takeuchi H.* The knowledge-creating company. — N. Y.: Oxford University Press, 1995. (Пер.: *Нонака И., Такеучи Х.* Компания — создатель знания. — М.: ЗАО «Олимп- бизнес», 2003.)
23. *Шемакин Ю. И., Романов А. А.* Компьютерная семантика. — М.: НОЦ «Школа Китайгородской», 1995.
24. *McArthur D.* Information, its forms and functions: The elements of semiology. — Lewinton: The Edwin Mellen Press, Ltd., 1997.
25. *Колин К. К.* Становление информатики как фундаментальной науки и комплексной научной проблемы // Системы и средства информатики. Спец. вып. «Научно-методологические проблемы информатики» / Под ред. К. К. Колина. — М.: ИПИ РАН, 2006. С. 7–58.
26. *Зацман И. М.* Концептуализация данных наукометрических исследований в научных электронных библиотеках // Труды 10-й Всероссийской конференции «Электронные библиотеки: перспективные методы и технологии, электронные коллекции». — Дубна: ОИЯИ, 2008 (в печати).
27. *Eco U.* A theory of semiotics. — Bloomington: Indiana University Press, 1976.
28. *Чебанов С. В.* Новый этап становления общей семиотики: вклад техно- и биосемиотики // Вестник РФФИ, 2003. № 4(34). С. 65–71.
29. *Marwick A. D.* Knowledge management technology // IBM Systems Journal, 2001. Vol. 40. No. 4. P. 814–830.
30. *Баранов А. Н., Добровольский Д. О.* Аспекты теории фразеологии. — М.: Знак, 2008.
31. *Гак В. Г.* Лексическое значение слова // Большой энциклопедический словарь «Языкознание». — М.: Большая российская энциклопедия, 1998. С. 261–263.
32. *Баранов А. Н.* Против «разложения смысла»: узнавание в семантике идиом // Компьютерная лингвистика и интеллектуальные технологии: По материалам ежегодной Международной конференции «Диалог». Вып. 7(14). — М.: РГГУ, 2008. С. 39–44.
33. *Newman J.* Some observations on the semantics of "Information" // Information Systems Frontiers, 2001. Vol. 3. No. 2. P. 155–167.
34. *Ingwersen P.* Information and information science // Encyclopaedia of Library and Information Science. Vol. 56. Sup. 19. — N. Y.: Marcel Dekker Inc., 1992. P. 137–174.
35. *Уфимцева А. А.* Знак языковой // Большой энциклопедический словарь «Языкознание». — М.: Большая российская энциклопедия, 1998. С. 167.
36. *Зацман И. М., Кожунова О. С.* Семантический словарь системы информационного мониторинга в сфере науки: задачи и функции // Системы и средства информатики. Вып. 17. — М.: Наука, 2007. С. 124–141.
37. *Кожунова О. С., Зацман И. М.* Прагматические аспекты создания семантического словаря терминов информационного мониторинга // Труды международной конференции Диалог-2007 «Компьютерная лингвистика и интеллектуальные технологии». — М.: Издательский центр РГГУ, 2007. С. 278–285.
38. *Щедровицкий Г. П., Алексеев Н. Г.* О возможных путях исследования мышления как деятельности // Докл. АПН РСФСР, 1957. № 3; 1958. №№ 1, 4; 1959. №№ 1, 2, 4; 1960. №№ 2, 4–6; 1961. №№ 4, 5; 1962. №№ 2–6.
39. *Пископпель А. А.* К творческой биографии Г. П. Щедровицкого (1929–1994) // В кн.: Щедровицкий Г. П. Избранные труды. — М.: Шк. Культ. Полит., 1995. С. XIII–XXXVII.
40. Common Weakness Enumeration. http://cwe.mitre.org/index.html.

ИНФОРМАТИКА И ЕЁ ПРИМЕНЕНИЯ, 2008. Т. 2. Вып. 3. С. 70–76

К 25-ЛЕТИЮ ИНСТИТУТА ПРОБЛЕМ ИНФОРМАТИКИ РАН*

И. А. Соколов[1], В. Н. Захаров[2]

Аннотация: Представлена история создания ИПИ РАН, дана общая характеристика института, отражены основные этапы его развития. Рассмотрена эволюция основных направлений исследований института, охарактеризованы важнейшие полученные за 25 лет фундаментальные и прикладные результаты.

Ключевые слова: ИПИ РАН; 25 лет; история; направления исследований

1 История создания и краткая справка об институте. Основные научные направления

В середине 1983 г. руководством СССР было принято решение о расширении и усилении исследований в области информатики и вычислительной техники в рамках Академии наук СССР. С целью реализации данного решения в составе Академии наук СССР было образовано Отделение информатики, вычислительной техники и автоматизации. Главными инициаторами подготовки и принятия решений по организации нового Отделения были президент АН СССР Анатолий Петрович Александров и вице-президент АН СССР Евгений Павлович Велихов, который и возглавил новое Отделение. В состав Отделения вошли Институт прикладной математики АН СССР, Вычислительный центр АН СССР, Институт проблем передачи информации АН СССР, а также несколько вновь образованных институтов. 29 июля 1983 г. было принято постановление ЦК КПСС и Совета Министров СССР об образовании Института проблем информатики АН СССР (ИПИАН). Соответствующее Распоряжение Президиума АН СССР было датировано 2 августа 1983 г.

В начале 1980-х гг. появляются и стремительно завоевывают новые области применения массовые средства вычислительной техники — персональные компьютеры, существенное развитие получают научные основы информатики. Однако «ведомственность» организаций, занимающихся проблемами информатизации, служила тормозом для развития информатики в СССР.

Именно с целью преодоления «ведомственных» барьеров для развития методов и средств информатизации и было образовано новое Отделение АН СССР, а член-корреспондент АН СССР Б. Н. Наумов, возглавлявший Институт электронных управляющих машин Минприбора СССР (ИНЭУМ), был назначен директором-организатором одного из новых институтов — ИПИАН. Основная задача ИПИАН была определена как «проведение фундаментальных и прикладных исследований в области технических и программных средств массовой вычислительной техники и систем на их основе», а интеллектуальной базой первых научных подразделений ИПИАН стали коллективы ряда научных отделов ИНЭУМ, переведенные в ИПИАН в начале 1984 г.

В 1984–85 гг. под руководством директора ИПИАН Б. Н. Наумова (который в 1984 г. был избран действительным членом АН СССР) и в значительной степени благодаря его инициативе и настойчивости коллектив специалистов, представлявших институты Академии наук СССР, Академий наук союзных республик СССР, Академий наук стран Восточной Европы, разработал Концепцию новых поколений вычислительных систем. Разработка Концепции послужила реакцией на стратегическую программу создания ЭВМ пятого поколения, объявленную в начале 1980-х гг. в Японии и расценивавшуюся в мире как «Японский вызов».

В Концепции были представлены главные направления исследований и разработок с целью получения принципиально нового качества информационных и вычислительных систем. Концепция предполагала, что повышение качества систем может быть достигнуто на основе сочетания новых методов представления и обработки информации (данных и знаний) и возможностей технических и программных средств вычислительной техники и передачи данных. Следует особо отметить, что Концепция была направлена прежде всего на создание систем новых поколений, а не на разработ-

*Сокращенный вариант статьи, опубликованной в журнале «История науки и техники», 2008, № 7.

[1]Институт проблем информатики Российской академии наук, isokolov@ipiran.ru

[2]Институт проблем информатики Российской академии наук, vzakharov@ipiran.ru

ку ЭВМ пятого поколения. Сейчас, спустя более 20 лет после разработки Концепции, можно сказать, что именно системный подход, положенный в ее основу, был полностью подтвержден мировой практикой. Концепция создания новых поколений вычислительных систем сопровождалась детальной проработкой необходимой тематики исследований и разработок.

Концепция новых поколений вычислительных систем была реализована в программе научно-технического сотрудничества стран — членов СЭВ. В состав программы, разработанной под руководством Б. Н. Наумова, были включены фундаментальные и прикладные исследования, осуществлявшиеся в рамках следующих комплексных научных проектов:

- системы обработки знаний;
- системы обработки изображений и машинной графики;
- системы автоматизации проектирования вычислительных систем и сверхбольших интегральных схем (СБИС);
- сети ЭВМ;
- системы персональных компьютеров;
- отказоустойчивые вычислительные системы;
- новые принципы хранения информации (новые внешние запоминающие устройства);
- технологии программирования;
- новые алгоритмы и архитектуры обработки информации;
- учебная информатика.

Фундаментальные исследования в начальный период деятельности ИПИАН велись в основном в рамках работ по этим направлениям.

Институт в первые годы своего существования быстро развивался — в его состав вошли филиалы в Бердянске, Казани, Орле. В 1990 г. был образован совместный отдел с Радиотехническим институтом в Таганроге. Общая численность сотрудников института в 1988–1989 гг. превышала 1000 чел.

Начиная с 1992 г. институт называется Институтом проблем информатики Российской академии наук (ИПИ РАН). Институт входит в состав Отделения нанотехнологий и информационных технологий РАН (до 2002 г. — Отделения информатики, вычислительной техники и автоматизации, в 2002–2007 гг. — Отделения информационных технологий и вычислительных систем).

В настоящее время решением Президиума РАН утверждены следующие основные научные направления работ института:

- интегрированные информационно-телекоммуникационные системы и сети, информатизация общества;
- теоретические основы информатики и информационных технологий, включая математические модели и методы, стохастические технологии и системы;
- информационные технологии накопления, хранения, поиска, обработки, преобразования, отображения, защиты и передачи информации, когнитивные технологии;
- архитектура, системные решения и программное обеспечение (ПО) вычислительных комплексов и сетей новых поколений.

В состав института входят филиалы в Орле и Калининграде.

В институте с 1985 г. работает диссертационный совет, которому дано право проведения защит диссертаций на соискание ученых степеней доктора и кандидата наук по четырем специальностям: 05.13.11, 05.13.13, 05.13.15 и 05.13.17. С 1984 г. институт имеет базовую кафедру в Московском институте радиотехники, электроники и автоматизации (техническом университете) — МИРЭА, на которой ведут преподавание сотрудники института; многие выпускники кафедры работают в ИПИ РАН.

Институтом в течение многих лет осуществляется выпуск ежегодных сборников трудов «Системы и средства информатики»; последний по времени выпуск ежегодника за 2007 г. имеет № 17 («номерные» сборники выходят с 1989 г.; до этого было выпущено еще 3 сборника под другими названиями). Начиная с 2001 г. практически ежегодно дополнительно к основному сборнику издаются его специальные тематические выпуски. С 2007 г. по поручению Отделения нанотехнологий и информационных технологий РАН институт осуществляет издание ежеквартального научного журнала Отделения «Информатика и её применения». Сотрудниками института издано значительное число монографий, в том числе и в зарубежных издательствах.

2 Важнейшие практические результаты работ института

С первых дней существования института большинство научных исследований было связано с решением важнейших практических проблем, имеющих стратегический характер. В 1980-х гг. такой задачей являлась организация работ по созданию и использованию нового класса вычислительной техники — персональных ЭВМ. В 1984 г. (буквально в

первые месяцы существования института) по заданию Президиума АН СССР, Госкомитета по науке и технике и Госплана СССР были определены основные направления разработки и создания ПЭВМ в стране. В 1986–87 гг. в рамках сотрудничества Академий наук социалистических стран и Межправительственной комиссии по вычислительной технике была разработана Концепция и программа развития ПЭВМ в СССР и странах СЭВ на период до 1995 г. В ней были определены основные стратегические цели и выбраны технические пути их достижения. Выполнение значительной части работ в рамках этой программы было возложено на ИПИАН.

Во исполнение принятых решений были осуществлены работы по созданию отечественной 32-разрядной ПЭВМ (был выполнен логический проект микропроцессора, изготовлен, отлажен и испытан опытный образец машины). Комплект производственной документации был передан для внедрения на ряд заводов. К сожалению, большинство этих заводов находилось на территории Украины, и в результате произошедшего впоследствии распада СССР выпуск этих машин так и не начался.

В рамках программы работ по ПЭВМ, утвержденной Постановлением ЦК КПСС и Совета Министров СССР от 23 января 1986 г., институт выполнил значительную часть работ по созданию средств базового ПО. В частности, была разработана операционная система МИКРОС (функциональный аналог CP/M), поставлявшаяся в составе первых отечественных ПЭВМ ЕС-1840, НЕЙРОН И9.66, ИСКРА-1030, Электроника-85, а также микроЭВМ СМ 1800. Для тех же типов отечественных ПЭВМ была разработана мобильная операционная система ИНМОС (русскоязычный «клон» ОС UNIX), за создание и внедрение которой 5 сотрудников института были в 1988 г. удостоены премии Совета Министров СССР.

Разработанные в институте компиляторы с языков Бейсик, Паскаль, СИ, ФОРТРАН прошли испытания и были переданы в тиражирующие организации. Кроме того, были созданы базовые функциональные пакеты для основных операционных систем отечественных ПЭВМ: текстовый редактор ТЕКСТ-ОС, реляционная база данных РБД ОС, пакеты обработки текстовой информации, графической информации, табличной информации, интегрированный пакет прикладных программ анализа и обработки данных ИМАД, пакет статистической обработки.

Широкое практическое применение получили работы института по средствам цифровой обработки изображений и созданию аппаратно-программных персональных видеосистем «Микровидео», а также системы дактилоскопической идентификации личности, использовавшейся в органах внутренних дел.

Приход в институт в качестве директора в 1989 г. члена-корреспондента АН СССР Игоря Александровича Мизина, много лет до этого проработавшего в оборонной промышленности, совпал по времени с началом коренных изменений в жизни страны и Академии наук. По инициативе Игоря Александровича в институте стали активно развиваться работы в области построения интегрированных информационно-телекоммуникационных сетей и систем. Признанием научных достижений школы Мизина стало его избрание в 1997 г. действительным членом Российской академии наук. Важнейшим практическим результатом работ института в начале 1990-х гг. стала разработка Концепции создания и развития телекоммуникационных систем, содержащей обоснование и выбор международных стандартов, методов и средств, на которые предложено ориентировать развитие инфраструктуры телекоммуникаций в регионах России. Был разработан функционально полный комплекс аппаратно-программных средств (на микропроцессорной базе) построения сетей передачи данных с коммутацией пакетов и с интегральной коммутацией, включающий высокопроизводительный центр коммутации пакетов, пакетные адаптеры данных, средства управления и абонентского сопряжения с сетью передачи данных, средства электронной почты X.400. Разработанный комплекс средств нашел применение при создании информационно-телекоммуникационной системы в Псковской области, ставшей прообразом ряда проектов по созданию подобных региональных систем.

С середины 1990-х гг. институт активно участвует в работах по созданию систем информационного обеспечения процессов управления для органов государственной власти РФ. Разработаны типовые средства и комплексы информационной поддержки процессов принятия решений руководством страны в нормальных условиях и в случае чрезвычайных ситуаций. Институтом выполнены крупные системные разработки для Министерства по чрезвычайным ситуациям РФ — по развитию автоматизированной информационной управляющей системы единой государственной системы предупреждения и ликвидации чрезвычайных ситуаций в части обработки данных и обеспечения информационно-телекоммуникационного обмена единой диспетчерской системы города.

С 1999 г. директором института является Игорь Анатольевич Соколов, работавший до этого в институте заведующим отделом и заместителем директора (в 2003 г. избран членом-корреспондентом

РАН, в 2008 г. — академиком). Он продолжил разработку проектов, начатых при И. А. Мизине. Начиная с этого времени институт стал более активно участвовать в работах по информатизации Президиума РАН и его учреждений (институт был назначен головным исполнителем по целевой программе). Вышли на новый уровень работы института в интересах Банка России (участие в разработке и создании информационно-телекоммуникационной системы обеспечения электронных расчетов, создание почтовых служб ряда территориальных управлений и др.).

В настоящее время институтом ведутся крупные проекты по созданию информационно-телекоммуникационных систем с ГУСП, Федеральной службой безопасности и Министерством внутренних дел РФ.

3 Основные фундаментальные результаты прошлых лет

В первые годы существования института выделялись три основных направления проводимых исследований:

(1) разработка архитектурных решений и технических средств ЭВМ массового применения;

(2) разработка ПО ЭВМ массового применения;

(3) исследование и разработка вопросов системного применения ЭВМ.

В области **разработки технических средств** были выполнены работы по разработке архитектуры семейства универсальных 32-разрядных ЭВМ, которые завершились созданием прототипа, переданного в производство. Совместно со специалистами из Чехословакии были разработаны спецпроцессоры с RISC[1]-архитектурой, ориентированные на решение задач искусственного интеллекта. Полученные результаты по моделированию цифровых устройств, системам тестирования и оценкам характеристик средств вычислительной техники (диагностический комбайн, программные модели 386 и 387 процессоров, наборы функциональных тестов) в настоящее время используются и развиваются в работах института.

В 1980-е гг. институт проводил исследования в области **архитектурных решений и управляющих программных средств** для построения распределенных информационно-вычислительных систем на базе ЭВМ общего назначения и персональных ЭВМ. Результатом работ стало создание (совместно с НИИ ЭВМ г. Минска) одной из последних машин класса мэйнфрейм в СССР — ЕС-1130. За работы по созданию управляющих вычислительных комплексов (совместно с заводом вычислительных и управляющих машин (ВУМ) в Киеве) коллектив разработчиков, в состав которого входили Б. Н. Наумов и другие сотрудники ИПИАН, был удостоен Государственной премии Украинской ССР. А работа, результатом которой стало производство таких комплексов в Воронеже, была удостоена Государственной премии СССР (в составе коллектива лауреатов был академик Б. Н. Наумов).

Исследования и разработки в области **программного обеспечения** традиционно представляют одно из основных направлений деятельности института. Коллектив ученых ИПИАН разрабатывал базовые программные средства ПЭВМ. К достижениям того времени следует отнести создание унифицированной операционной системы УОС для микропроцессора К1810ВМ86, обладавшей уникальными для того времени возможностями — обеспечивался мультипрограммный режим, поддержка работы в локальных сетях, многооконный интерфейс, динамическая смена кодовых таблиц и совместимость с наиболее распространенными операционными системами для данного класса машин.

В институте велись работы по **созданию научно обоснованной технологии программирования**, охватывающей все этапы жизненного цикла программных средств для вычислительных систем новых поколений и создающей основу промышленного производства программ и программной документации.

Были предложены новые подходы к автоматизации разработки прикладных программ с заданными характеристиками, разработан метод порождения целевых программных систем, определены состав и структура инструментально-технологических средств поддержки новой технологии (система ГЕНПАК), проведены исследования и практические разработки по обеспечению мобильности ПО, разработана и успешно применена методика переноса программных средств.

Были проведены исследования по моделированию технологических этапов проектирования программных средств с использованием сетей Петри и созданию инструментальных средств объектно-ориентированного программирования. Значительная часть работ по практической реализации технологии программирования, созданию базовых пакетов прикладного ПО и полиграфическому сопровождению работ института велась в Бердянском филиале ИПИАН, численность сотрудников которого доходила до 200 чел. К сожалению, после распада СССР Бердянский филиал прекратил свое существование.

[1] RISC — Reduced Instruction Set Computer — компьютер с уменьшенным набором команд.

В 1980-е – начале 1990-х гг. в институте активно проводились разработки систем автоматизированного проектирования (САПР) машиностроения и микроэлектроники. Наиболее значительным достижением в этой области было создание системы проектирования печатных плат МАГИСТР-2, в которой были реализованы принципиально новые, превосходящие тогдашние зарубежные аналоги, алгоритмы трассировки двусторонних и многослойных печатных плат. В рамках международного проекта КНП-3 в институте были разработаны принципы организации САПР СБИС методом кремниевого компилирования и элементы ПО такого компилятора. Работы в области САПР машиностроения активно велись в тесной связи с промышленными предприятиями в Казанском филиале института (в частности, была создана интерактивная графическая система для конструирования и оформления чертежей — САПР МАШ). В 1997 г. Казанский филиал ИПИ РАН был преобразован в Институт проблем информатики Академии наук Татарстана.

С 1999 по 2005 гг. в институте работала группа ученых, которой руководил академик Всеволод Сергеевич Бурцев. Проводились исследования в области нетрадиционных архитектур и системного ПО высокопроизводительных информационно-вычислительных комплексов, основывающихся на ассоциативной памяти. Были разработаны программные модели новой архитектуры и подготовлена реализация ее элементов «в железе».

В институте в течение ряда лет велись работы в области геоинформационных технологий. Была разработана система сбора, структуризации, семантического моделирования и кодирования, хранения, анализа, поиска, обработки, отображения и передачи многоаспектной и разной по форме пространственной информации о Земле, обеспечивающая синтез электронного образа Земли как многомерного представления планеты. Разработана технология ввода, обработки и выдачи пространственных данных в виде электронных карт и текстовой информации, а также формирование и ведение базы метаданных электронных карт. С активнейшим участием сотрудников ИПИ РАН впервые в мировой практике разработаны Государственные (национальные) стандарты ГОСТ Р 52055-2003 «Геоинформационное картографирование. Пространственные модели местности. Общие требования» и ГОСТ Р 52293-2004 «Геоинформационное картографирование. Система электронных карт. Карты электронные топографические. Общие требования». Разработана методология построения объектно-ориентированных геоинформационных систем (ГИС). Созданы методы хранения и поиска информации в текстово-графической базе данных, которые легли в основу комплекса программ ГИС Objectland, принятой в качестве базовой для ведения автоматизированного земельного кадастра в Российской Федерации и используемой во многих областях страны.

4 Фундаментальные исследования, проводящиеся в настоящее время

В данном разделе статьи кратко представлены некоторые направления современных исследований в ИПИ РАН.

В конце 1980-х гг. в институте начались работы в области самосинхронной схемотехники, инициированные контактами с исследовательской группой из Ленинградского электротехнического института. Институт стал ведущим в стране центром исследований в этом направлении схемотехники. Были проведены исследования по теоретическому обоснованию методов строгой самосинхронизации (ССС), разработаны инструментальные средства и системы для проектирования ССС-схем, созданы библиотеки базовых элементов. Совместно с Технологическим центром МИЭТ удалось впервые в отечественной и зарубежной практике изготовить на базе полузаказных БИС тестовые кристаллы, реализующие вычислительное ядро 8-разрядного микроконтроллера. Проведенные испытания подтвердили теоретические выводы о значительном расширении диапазона работоспособности ССС-схем в области температур и напряжений по сравнению с синхронной реализацией, а также о возможности ускорения работы этих устройств. Работы в этой области продолжаются и представляются весьма перспективными; уже получен ряд патентов на изобретения.

В институте достаточно давно проводятся работы в области **периферийных устройств ПЭВМ**. Этим в основном занимается Орловский филиал института, в котором были разработаны мини-принтеры и принтеры-плоттеры, превосходившие на момент создания все имевшиеся отечественные изделия. Это направление включает в себя и теоретические исследования в области цветообразования и стереоскопического воспроизведения изображений; работы в данном направлении активно развиваются.

В институте в течение многих лет ведутся работы в области разработки методов интеграции неоднородных информационных ресурсов. Были разработаны теоретические основы, методология и архитектура интероперабельных сред неоднородных информационных ресурсов, методы и средства

спецификации и повторного совместного использования ресурсов при проектировании информационных систем — язык СИНТЕЗ, метод композиционного проектирования информационных систем, основанный на композиционном исчислении спецификаций. Эти фундаментальные результаты являются базисом проводимых работ по созданию слоя предметных посредников в электронных библиотеках, позволяющего унифицировать доступ к разнородным электронным коллекциям. После прекращения в 1991 г. сотрудничества в области интеграции неоднородных информационных ресурсов в рамках комиссии по вычислительной технике соцстран усилиями сотрудников института была организована московская секция ACM SIGMOD, ведущая активную работу и в настоящее время.

С первых дней существования института большое внимание уделялось исследованиям, связанным с **системными применениями ЭВМ** в области создания методов, алгоритмов и программ для анализа процессов и для обработки информации в стохастических системах. Эти исследования выполнялись под руководством выдающегося отечественного ученого академика Владимира Семеновича Пугачева, пришедшего в ИПИАН с группой своих учеников и сотрудников вскоре после образования института. Научным коллективом были разработаны методы, алгоритмы и программы статистического анализа и условно-оптимального оценивания случайных процессов и последовательностей, алгоритмы автоматического составления и решения уравнений с помощью ЭВМ для вероятностных характеристик процессов в нелинейных стохастических системах по исходным уравнениям модели. Академик Пугачев своей важнейшей задачей считал доведение теоретических разработок до практики их использования инженерами и исследователями. Коллективом, которым он руководил, были разработаны основы новой информационной технологии построения математических моделей сложных систем и процессов по экспериментальным данным при наличии неопределенностей, включающей теорию быстрого условно-оптимального оценивания и идентификации, управления и планирования экспериментов, проверки достоверности, принятия решений. Полученные результаты позволили существенно расширить круг специалистов, активно использующих возможности современных средств вычислительной техники и методов математического моделирования, за счет устранения необходимости заниматься собственно программированием. Были разработаны принципы создания интеллектуализированных пакетов прикладных программ для исследования стохастических моделей, реализованные программными средствами. За работы в области стохастических систем в 1990 г. академик В. С. Пугачев был удостоен Ленинской премии, причем премию он получил индивидуально и был последним, кто получил такую премию.

В институте проводятся работы в области **методов представления и обработки знаний**. Были разработаны пакеты прикладных программ, обеспечивающие морфологический анализ русскоязычных текстов, в частности пакет РАМОС, включающий словарь русского языка Ожегова. Развитием работ, связанных с обработкой текстов, стали системы анализа и рубрицирования текстов, системы построения тезаурусов. Разработана когнитивно-лингвистическая модель и система структурных правил для алгоритмов машинного перевода и обработки знаний, извлекаемых из текстов на ряде европейских языков. Построен и реализован в виде действующих систем аппарат функционально-семантической англо-русской грамматики. Разработаны ассоциативные модели представления знаний на базе семантических сетей и фреймовых моделей и создан язык представления знаний ДЕКЛ. Разработаны методы логического вывода, основанные на представлении ассоциативной модели и позволяющие извлекать семантическую информацию из текстов русского языка с автоматическим формированием структур баз знаний (БЗ) и их использованием в задачах фактографического поиска и экспертных решений. На основе этих методов создана логико-аналитическая система АНАЛИТИК, настроенная на задачи логико-аналитической обработки сводок криминальной милиции и обеспечивающая автоматическое извлечение значимой информации из текстовых сообщений и ее использование для нечеткого поиска.

В 1989 г. под руководством И. А. Мизина в институте активизировались работы в области развития **сетевых информационных технологий и технологии построения крупномасштабных информационно-коммуникационных систем**. Основным фундаментальным результатом этих исследований стала разработка взаимоувязанного комплекса математических моделей, методов и программных средств оценки общесистемных характеристик телекоммуникационных систем, использующих современные технологии передачи данных. Впоследствии под руководством И. А. Соколова была решена актуальная задача описания класса крупномасштабных телекоммуникационных систем двойного применения как полносвязных, территориально-структурированных, мультисетевых систем общенационального масштаба, совмещающих функции специальных систем и систем общего пользования.

Начиная с 1992 г. в институте большое внимание уделяется развитию **математических методов иссле-**

дования сложных информационных и телекоммуникационных систем (ИТС). Осуществлены фундаментальные теоретические исследования, разработка моделей, математических и программных средств для анализа и оптимизации характеристик реальных ИТС (процессов доставки информации с разными дисциплинами обслуживания и входными потоками, процессов управления и комплексной обработки информации). Новизна методов и результатов, полученных в этой области в ИПИ РАН, заключается в:

- разработке методов расчета сетей с учетом реальных технологий передачи, определяемых действующими международными стандартами и рекомендациями, и реальных вероятностных характеристик информационных потоков;
- комбинировании методов аналитического и имитационного моделирования, что позволяет корректировать естественные погрешности аналитических методов и учитывать специфику реальных систем;
- доведении работы до программной реализации, позволяющей проводить расчеты для реальных сетей большой размерности.

Важным этапом работ ИПИ РАН в этой области стало выполнение в 1999–2002 гг. проекта «Разработка модели коммутируемой телефонной сети ОАО «Ростелеком». Основной целью работы была разработка методов и средств моделирования коммутируемой сети ОАО «Ростелеком», предназначенных для обеспечения информационной поддержки пользователей АСУ цифровой сети в процессе принятия ими решений по оперативному управлению вторичной телефонной сетью ОАО «Ростелеком» при возникновении на ней нештатных ситуаций (в режиме, приближенном к реальному времени). В ходе работы выяснилось, что специфика данной сети не позволяла напрямую использовать ни одну из существующих в научной литературе моделей телефонных сетей. В связи с этим пришлось создавать сложную многоуровневую модель, включающую имитационную модель собственно телефонной сети с учетом всей специфики ее деятельности и алгоритмы динамического управления сетью, использующие как имитационную, так и упрощенную аналитическую модель сети.

С 2000 г. в ИПИ РАН активно развивается направление работ, связанное с исследованием систем и сетей массового обслуживания (СМО и СеМО), моделирующих современные и перспективные информационно-технические средства (ИТС) и их узлы. В рамках этого направления получены существенные теоретические результаты: разработаны новые математические методы, позволяющие вычислять на основе аналитических соотношений главные показатели функционирования СМО с различными дисциплинами обслуживания и сложными входящими потоками, получены новые результаты в части исследования СеМО, в том числе СеМО с отрицательными заявками (G-сети) и зависимыми длинами заявок на разных этапах обслуживания; разработана методика расчетов СМО и СеМО большой размерности. Результаты исследований были применены в ряде НИР и ОКР, связанных с разработкой средств моделирования реальных ИТС.

Наряду с работами по созданию математических и программных средств «непосредственного» моделирования ИТС, в ИПИ РАН осуществляются исследования по разработке теоретических вероятностно-статистических методов, ориентированных на перспективное применение в области моделирования ИТС. В частности, начиная с 1994 г. ведутся исследования в области разработки методов анализа случайных процессов сложной структуры. Основным направлением работ в данной области стало в последние годы моделирование информационных рисков и разработка методов вычисления надежностных характеристик ИТС и их элементов в интересах проектирования и прогнозирования функционирования систем в существенно стохастической среде, обеспечения их катастрофоустойчивости и информационной безопасности.

С самого начала существования института большое внимание уделялось работам в области учебной информатики. Институт принимал активное участие в разработке аппаратно-программных комплексов и пакетов прикладных программ для учебной информатики. На него была возложена головная роль в выработке требований, отборе и обеспечении поставки в образовательные учреждения первых учебных вычислительных комплексов, а также в подготовке экспериментальных учебных пособий по курсу школьной информатики. В институте были разработаны пакеты прикладных программ для обучения машинописи, основам информатики и вычислительной техники, для создания прикладных программных средств обучения. Фундаментальные результаты в области дистанционного образования на практике были опробованы в ряде международных проектов. Институт тесно сотрудничает с организациями ЮНЕСКО.

Институт проблем информатики РАН встречает свое 25-летие как признанный центр исследований и разработок в области информатики, сочетающий фундаментальные научные исследования с практикой построения важнейших крупномасштабных информационно-телекоммуникационных систем.

ABSTRACTS

STRUCTURAL MATRIX SYSTEMS DECOMPOSITION

A. Olenin

IPI RAN, aolenin@yandex.ru

One of possible approaches to matrix systems parallelizing by their structural decomposion on the set of subsystems independent at the certain stage of calculations is considered. The constructive algorithm of the decomposition method is formulated and the estimation of its computing expenses is given.

Keywords: matrix system; band matrix; full matrix; triangular matrix; block tridiagonal matrix; decomposition; partition vector; partition interval; factorization; LU-algorithm; parallelizing

CONCURRENT DESIGN AND VERIFICATION OF DIGITAL HARDWARE

S. Baranov[1], S. Frenkel[2], V. Sinelnikov[3], and V. Zakharov[4]

[1]Holon Institute of Technology, Holon, Israel, samary@012.net.il
[2]IPI RAN
[3]Holon Institute of Technology, Holon, Israel
[4]IPI RAN, VZakharov@ipiran.ru

The main goal of this paper is to present a new design verification methodology for complicated digital systems, designed by high-level synthesis. This methodology is based on Algorithmic State Machine transformations (composition, minimization, extraction, etc.), special algorithms for Data Path and Control Unit design, and very fast optimizing synthesis of finite state machines and combinational circuits with hardly any constraints on their size, that is, the number of inputs, outputs, and states. Design tools supporting this methodology allow very fast implement, check and estimate many possible design versions, to find an optimized decision of the design problem and to simplify the verification problem for digital systems. In contrast to existent semi-formal approaches to verification of industrial systems, based on combination of simulation and formal verification approaches, a formalized method based on concurrency of synthesis and verification that is providing regular efficient way to verify the system designed properties starting from its semi-formal specification up to field programmable gate array implementation is considered.

Keywords: digital systems design; formal verification; finite state machine

COST FUNCTION OF RESOURCES IN ECONOMICAL MODEL OF GRID CONTROL

Y. M. Agalarov

IPI RAN, YAgalarov@ipiran.ru

A problem of profit maximization for owner of GRID local node resources with economical model of control, in which the assigned resources for external user are paid and payment amount is dependent on supply and demand of resources is considered. In the model, the queue of global tasks is formed only in GRID resources programming center that performs search and selection of resources, and task sending to resources is performed simultaneously with it reservation and selection. A cost function of resources that allows the owner to efficiently allocate the resources of GRID local node between global and local tasks is proposed. The results of analytical investigation of the considered problem and computer-modeling-based comparative analysis of the proposed approach are presented.

Keywords: GRID; model resourse allocation; multiprocessor tasks; owner of resources; Markov process; strategy

MULTICHANNEL QUEUEING SYSTEM WITH A FINITE BUFFER, A LOCKOUT OF AN INPUT FLOW, AND A KNOCKOUT OF CUSTOMERS FROM THE BUFFER

V. Chaplygin

IPI RAN, vchaplygin@ipiran.ru

A multichannel queueing system with a finite buffer, a lockout of a semi-Markovian input flow, and a knockout of customers from the buffer by a first customer arrived into the system over the period when the input flow is unlocked is considered. The periods of lockout and the periods when the input flow is unlocked have an exponential distribution with different intensities. The main stationary characteristics such as a queue distribution, a loss probability, and a mean sojourn time are found.

Keywords: queueing system; semi-Markovian input flow; knockout of customers

SERVICE-ORIENTED APPROACH TO MULTIMODAL BIOMETRICS DESIGNING

O. S. Ushmaev

IPI RAN, oushmaev@ipiran.ru

Novadays, multimodal biometrics is rapidly replacing tedious procedures of identification. Particularly operating and perspective civil ID systems use multimodal approach. The formal method for designing high-speed multibiometric technologies and systems is suggested. The effectiveness of the approach is shown by an example of developed experimental software with service-oriented architecture.

Keywords: biometric identification; multimodal biometrics; platform independent; service-oriented architecture

PERSONAL AND COLLECTIVE CONCEPTS REPRESENTATION IN THE DIGITAL SPHERE

I. M. Zatsman[1], V. V. Kosarik[2], and O. A. Kurchavova[3]

[1]IPI RAN, im@a170.ipi.ac.ru
[2]IPI RAN, valery@a170.ipi.ac.ru
[3]IPI RAN, koa@a170.ipi.ac.ru

Key issues of the 7th Framework program documents of the European Union accepted for the period 2007–2013 are analyzed. This program contains formulations of some new directions referring to the knowledge representation problem in information systems of long-term use. The results of the analysis allow us to assert that simultaneously with a traditional informatics problem covering representation, storage, and extraction in the digital sphere of already available knowledge and meeting technological economic, educational, and other socially significant challenges, there is a problem of generation of new goal-oriented knowledge systems when available knowledge systems do not satisfy these challenges and from this point of view are incomplete. The new way to personal and collective concepts respresentation in the digital sphere and also stages of concepts evolution in the process of generation of new goal-oriented knowledge systems is offered.

Keywords: personal, collective and conventional concepts; lasting and volatile concepts; personal and collective concepts representation in the digital sphere

TOWARDS THE 25TH ANNIVERSARY OF IPI RAN

I. A. Sokolov[1] and V. N. Zakharov[2]

[1]IPI RAN, isokolov@ipiran.ru
[2]IPI RAN, vzakharov@ipiran.ru

The history of IPI RAN creation, its general characteristics, main stages of development are presented. The evolution of the main directions of scientific researches is considered. Main fundamental and applied results obtained during 25 years are described.

Keywords: Institute of Informatics Problems RAS; 25 years; history; directions of scientific researches

Об авторах

Агаларов Явер Мирзабекович (р. 1952) — кандидат технических наук, доцент, ведущий научный сотрудник ИПИ РАН

Баранов Самарий Иосифович (р. 1938) — доктор технических наук, профессор Холонского технологического института, Израиль

Захаров Виктор Николаевич (р. 1948) — кандидат технических наук, доцент, ученый секретарь ИПИ РАН

Зацман Игорь Моисеевич (р. 1952) — кандидат технических наук, заведующий отделом ИПИ РАН

Косарик Валерий Валентинович (р. 1970) — научный сотрудник ИПИ РАН

Курчавова Ольга Анатольевна (р. 1961) — старший научный сотрудник ИПИ РАН

Оленин Анатолий Степанович (р. 1946) — доктор физико-математических наук, ведущий научный сотрудник Института проблем проектирования в микроэлектронике РАН

Синельников Владимир Ефимович (р. 1945) — кандидат технических наук, преподаватель университета Бар-Илан, Израиль

Соколов Игорь Анатольевич (р. 1954) — академик (действительный член) Российской академии наук, доктор технических наук, директор ИПИ РАН

Ушмаев Олег Станиславович (р. 1981) — кандидат технических наук, старший научный сотрудник ИПИ РАН

Френкель Сергей Лазаревич (р. 1951) — кандидат технических наук, доцент, старший научный сотрудник ИПИ РАН, доцент факультета Вычислительные машины и системы Московского института радиотехники, электроники и автоматики (МИРЭА)

Чаплыгин Василий Васильевич (р. 1978) — кандидат физико-математических наук, старший научный сотрудник ИПИ РАН

About Authors

Agalarov Yaver M. (b. 1952) — Candidate of Technical Sciences (PhD); head scientist, Institute of Informatics Problems, Russian Academy of Sciences

Baranov Samary I. (b. 1938) — Doctor of Technical Sciences; professor, Department of Computer Sciences of Holon Institute of Technology

Chaplygin Vasily V. (b. 1978) — Candidate of Phisical and Mathematical Sciences (PhD); senior scientist, Institute of Informatics Problems, Russian Academy of Sciences

Frenkel Sergey L. (b. 1951) — Candidate of Technical Sciences (PhD); senior scientist, Institute of Informatics Problems, Russian Academy of Sciences; associate professor, Moscow Institute of Radio, Electronics, and Automation, Faculty of Computer Systems

Kosarik Valerii V. (b. 1970) — scientist, Institute of Informatics Problems, Russian Academy of Sciences

Kurchavova Olga A. (b. 1961) — senior scientist, Institute of Informatics Problems, Russian Academy of Sciences

Olenin Anatoly S. (b. 1946) — Doctor of Physical and Mathematical Sciences; professor in mathematics; leading scientist, Institute of Design Problems in Microelectronics, Russian Academy of Sciences

Sinelnikov Vladimir E. (b. 1945) — Candidate of Technical Sciences (PhD); lector, Bar-Ilan University, Israel

Sokolov Igor A. (b. 1954) — Academician of the Russian Academy of Sciences; Doctor of Technical Sciences; director, Institute of Informatics Problems, Russian Academy of Sciences

Ushmaev Oleg S. (b. 1981) — Candidate of Technical Sciences (PhD) senior scientist, Institute of Informatics Problems, Russian Academy of Sciences

Zakharov Victor N. (b. 1948) — Candidate of Technical Sciences (PhD); associate professor, Scientific Secretary, Institute of Informatics Problems, Russian Academy of Sciences

Zatsman Igor M. (b. 1952) — Candidate of Technical Sciences (PhD); department head, Institute of Informatics Problems, Russian Academy of Sciences