Пінчук В. П., Подковаліхіна О. О.

ОБ'ЄКТНО-ОРІЄНТОВАНИЙ ПІДХІД ДЛЯ ПОБУДОВИ МОДЕЛЕЙ СКЛАДНИХ СИСТЕМ

Запропоновано методика побудови динамічної моделі системи з локалізованими властивостями на основі об'єктно-орієнтованого підходу. Розглянуто приклад системи автоматичного регулювання температури робочого тіла з пропорціонально-інтегрально-диференціальним законом регулювання, наведено методику побудови класів і функцій, що складають програмну реалізацію моделі, обговорюються можливості запропонованої методики.

Ключові слова: комп'ютерне моделювання систем, моделювання динаміки системи, об'єктно-орієнтований підхід, система терморегулювання.

Pinchuk V. P., Podkovalihina E. A.

OBJECT-ORIENTED APPROACH FOR MODELS DESIGNING OF THE COMPLEX SYSTEMS

A method for constructing a dynamic model of the system with localized features based on object-oriented approach was proposed. An example of a system for automatic temperature control of the working body with a proportional-integral-differential law of regulation was considered.

A technique for constructing classes and their functions that comprise the software implementation of the model was leaded, the opportunities of the proposed method were discussed.

Key words: computer simulation of the systems, modeling the dynamics of the system, object-oriented approach, the thermal control system.

НЕЙРОІНФОРМАТИКА ТА ІНТЕЛЕКТУАЛЬНІ СИСТЕМИ

НЕЙРОИНФОРМАТИКА И ИНТЕЛЛЕКТУАЛЬНЫЕ СИСТЕМЫ

NEUROINFORMATICS AND INTELLIGENT SYSTEMS

УДК 519.168:004.658

Асеев Г. Г.

Д-р техн. наук, профессор, заведующий кафедрой Харьковской государственной академии культуры

МЕТОДЫ ИНТЕЛЛЕКТУАЛЬНОГО АНАЛИЗА ДАННЫХ В ЭЛЕКТРОННЫХ ХРАНИЛИЩАХ: ГЕНЕТИЧЕСКИЕ АЛГОРИТМЫ

Представлен один из возможных методов интеллектуального анализа данных в электронных хранилищах большого объема – генетические алгоритмы и их модификация.

Ключевые слова: интеллектуальный анализ, электронное хранилище, нейронная сеть, генетические алгоритмы.

ВВЕДЕНИЕ

В настоящее время в электронных хранилищах данных (ХД) корпоративных информационных систем хранятся терабайты различной текстовой и числовой информации. Для обнаружения, извлечения и интеллектуального анализа этих данных используются методы Knowledge Discovery in Databases и Data mining [1]. В [1] были описаны некоторые рекомендации, следуя которым можно подготовить качественные данные в нужном объеме для анализа: первичные источники данных, хранение данных, подготовка исходного набора данных, предобработка и очистка исходных данных [2], трансформация, нормализация, выдвижение гипотез и построение модели Data Mining [3]. Данная работа продолжает цикл статей, посвященных методам интеллектуального анализа данных в электронных хранилищах большого объема, в частности модификации генетических алгоритмов.

ГЕНЕТИЧЕСКИЕ АЛГОРИТМЫ – МАТЕМАТИЧЕСКИЙ АППАРАТ

Такие свойства генетических алгоритмов, как адаптивность, робастность, возможность распараллеливания вычислений и отыскание глобального экстремума принятой функции приспособленности, обеспечили их эффективное использование для решения различных задач в пространствах высокой размерности в ХД. Примером подобной задачи может служить обучение нейросети, то есть подбора таких значений весов, при которых достигается минимальная ошибка.

Из биологии мы знаем, что любой организм может быть представлен своим *фенотипом*, который фактически определяет, чем является объект в реальном мире, и *генотипом*, который содержит всю информацию об объекте на уровне хромосомного набора. При этом каждый ген, то есть элемент информации генотипа, имеет свое отражение в фенотипе. Разработчик генетических алгоритмов выступает в данном случае как «создатель», который должен правильно установить законы эволюции, чтобы достичь желаемой цели как можно быстрее. Впервые эти нестандартные идеи были применены к решению оптимизационных задач в середине 70-х годов [4]. Примерно через десять лет появились первые теоретические обоснования этого подхода [5, 6]. В дальнейшем генетические алгоритмы доказали свою конкурентоспособность при решении многих *NP*-трудных задач [7] и особенно в практических приложениях, где математические модели имеют сложную структуру и применение стандартных методов типа ветвей и границ, динамического или линейного программирования крайне затруднено.

В наиболее часто встречающейся разновидности генетического алгоритма для представления генотипа объекта применяются битовые строки. При этом каждому атрибуту объекта в фенотипе соответствует один *ген* в генотипе объекта. Ген представляет собой битовую строку, чаще всего фиксированной длины, которая представляет собой значение этого признака.

Генетический алгоритм работает с представленными в конечном алфавите строками S конечной длины l, которые используются для кодировки исходного множества альтернатив W. Строки представляют собой упорядоченные наборы из l элементов: $S=(s_1, s_2, \ldots, s_l)$, каждый из которых может быть задан в своем собственном алфавите V_i, $i=\overline{1, L}$, где алфавит V_i является множеством из r_i символов: $V_i=\{v_{ij},\ j=\overline{1, r_i}\}$. Для решения конкретной задачи требуется однозначно отобразить конечное множество альтернатив W на множество строк подходящей длины (очевидно, что длина строк зависит от алфавитов, используемых для их задания).

Для работы алгоритма необходимо на множестве строк $U^m{}_{(V_1,V_2,\ldots,V_m)}$ задать неотрицательную функцию $F(S)$, определяющую показатель качества, «ценность» строки $SO\ U^m{}_{(V_1,V_2,\ldots,V_m)}$. Алгоритм производит поиск строки, для которой

$$F^*(S)=\arg \max_{S\in U^m{}_{(V_1,V_2,\ldots,V_m)}} F(S).$$

Если на множестве W задана целевая функция $f(w)$, то функцию $F(S)$ на множестве строк $U^m{}_{(V_1,V_2,\ldots,V_m)}$ можем определить следующим образом: $F(S)=f(w)$, если элемент w при отображении исходного множества W на множество строк был сопоставлен строке S.

Генетический алгоритм за один шаг производит обработку некоторой популяции строк. Популяция $G(t)$ на шаге t представляет собой конечный набор строк:

$$G(t)=\left(S_1^t, S_2^t, \ldots, S_N^t\right),\ S_k^t \in U^m{}_{(V_1,V_2,\ldots,V_m)},\quad k=\overline{1, N},$$

где N – размер популяции, причем строки в популяции могут повторяться.

Анализ работы алгоритма удобно производить, используя аппарат схем. Схемой в генетическом алгоритме называют описание некоторого подмножества строк. Схема $H=(h_1, h_2, \ldots, h_m)$ может рассматриваться как строка, алфавиты для элементов которой дополнены специальным символом «#»:

$$H \in U^m{}_{\left(V_1^H, V_2^H, \ldots, V_m^H\right)},\quad V_i^H = V_i \cup "\#".$$

Если в некоторой позиции r схемы H присутствует символ «#», то такая позиция называется свободной, а сам символ «#» интерпретируется как произвольный символ из алфавита V_r. Позиция q схемы H называется фиксированной, если в этой позиции присутствует один из символов алфавита V_q. Схема H, в которой определены фиксированные и свободные позиции, описывает подмножество $U_H \subseteq U^m{}_{(V_1,V_2,\ldots,V_m)}$, содержащее такие строки, у которых элементы, соответствующие фиксированным позициям схемы, совпадают с символами схемы, а элементы, соответствующие свободным позициям схемы, являются произвольно заданными в соответствующих алфавитах:

$$U_H=\left\{S \middle| S \in U^m{}_{(V_1,V_2,\ldots,V_m)} \wedge \left(\forall i \left(i \in I_{[1,m]} \wedge h_i \neq "\#"\right) \rightarrow (s_i = h_i)\right)\right\},$$

где $I_{[1, m]}$ – множество целых чисел отрезка [1, m].

Например, для множества строк $U^5_{(V_1,V_2,V_3,V_4,V_5)}$, где U_i= {0, 1}, $V_i=\overline{1, 5}$, схема H_1=«1###0» задает такое множество строк, у которых первым элементом является символ «1», пятым – «0», а остальные – либо «0», либо «1». Строки «10010», «11110» являются примерами строк, принадлежащих множеству U_{H_1}.

Часть популяции $G(t)=\left(S_1^t, S_2^t, \ldots, S_N^t\right)$, строки которой удовлетворяют схеме H, обозначают $G_H(t)=\left(S_1^{H,t}, S_2^{H,t}, \ldots, S_{n(H,t)}^{H,t}\right)$, где $n(H, t)$ – число строк схемы H в популяции $G(t)$, и называют подпопуляцией, соответствующей схеме H.

ПРОЦЕДУРА ОПТИМИЗАЦИИ

В общем случае процедура оптимизации на основе обычного последовательного комплекс-метода выглядит следующим образом: требуется отыскать минимум некоторой функции, как правило, многоэкстремальной:

$$E(x) \rightarrow \min_{x\in R^n},$$

достаточно общего вида, при этом, о характере этой функции не делается практически никаких априорных предположений.

Будем использовать функции приспособленности следующего вида:

$$E(x)=\sum_{cl} E(cl)=\sum_{cl}\sum_{d} w_{cl,d} = \sum_{cl}\sum_{d} sf_{cl,d} \times$$

$$\times ids_{cl,d} = \sum_{cl}\sum_{d} sf_{cl,d} \times \log\frac{N}{ds_{cl}},$$

где $E(cl)$ – функция приспособленности для хромосомы cl; $w_{cl, d}$ – нормализованные данные о хромосоме c для документа d; $sf_{cl, d}$ – частота встречаемости терма (или набора термов), представленного хромосомой cl; $ids_{cl, d}$ – инверсная частота встречаемости терма (или

набора термов), представленного хромосомой *cl*; ds_{cl} – число документов, содержащих комбинации хромосомы *cl; N* – общее число документов в ХД.

Все данные должны быть представлены в двоичном коде (1 – если терм (или набор термов) содержится в документе, 0 – в противном случае). В ХД используются хромосомы, максимальная длина которых составляет до нескольких сотен генов, причем некоторые из них могут быть пустыми. Согласно этому утверждению, число повторяющихся термов в решении может варьироваться от 2 до нескольких сотен. Поскольку пространство решений очень большое, предлагается использовать мутацию, фиксированную между 50 и 70 процентами, и в конечном итоге каждая хромосома будет подвержена мутации для новой популяции.

Работа алгоритма начинается с формирования начального комплекса

$$x_i(0)=\left(x_{i1}(0), x_{i2}(0),..., x_{ij}(0),..., x_{in}(0)\right)^T, \quad i=1, 2,..., N\geq n+1,$$

представляющего собой «облако» (популяцию) точек (векторов), достаточно произвольно расположенных в *n*-мерном пространстве факторов. Среди множества этих точек находится «наихудшая» $x_N(0)$, в которой значение функции *E* ($x_N(0)$) максимально, после чего эта точка отражается через центр тяжести всех остальных вершин-точек, формируя новый комплекс $x_N(1)$, *i* = 1, 2,…, *N*. Такое отражение вместе с растяжением и сжатием обеспечивают движение комплекса к экстремуму функции *E* (*x*), при этом, благодаря достаточно случайному распределению точек «облака», поиск имеет глобальный характер.

С формальной точки зрения, рассмотрим процесс оптимизации на *k*-й итерации поиска, когда сформирован комплекс $x_i(k)$, *i* = 1, 2,…, *N*. Среди множества точек $x_i(k)$ находится «наихудшая», такая, что

$$E\left(x_N(k)\right)=\max_i\left\{E\left(x_1(k)\right),..., E\left(x_N(k)\right)\right\},$$

после чего определяется центр тяжести «облака» без наихудшей точки:

$$x_C(k)=\frac{1}{N-1}\left(\sum_{i=1}^{N}x_i(k)-x_N(k)\right).$$

Далее $x_N(k)$ отражается через центр тяжести $x_C(k)$, формируя новую вершину комплекса $x_R(k)$, которая теоретически расположена ближе к экстремуму, чем $x_N(k)$ и $x_C(k)$, т. е.

$$E\left(x_R(k)\right)<E\left(x_C(k)\right)<E\left(x_N(k)\right).$$

Операция отражения формально имеет следующий вид:

$$x_R(k)=x_C(k)+\eta_R\left(x_C(k)-x_N(k)\right)=$$

$$=\frac{1}{N-1}x_1(k)+...+\frac{1}{N-1}x_{N-1}(k)+\frac{\eta_R}{N-1}x_1(k)+...$$

$$...+\frac{\eta_R}{N-1}x_{N-1}(k)-\eta_R x_N(k)=X(k)R,$$

где η_R – параметр шага отражения, часто полагаемый равным единице, $X(k)=\left(x_1(k), x_2(k),..., x_{N-1}(k)\right)$ – $(n\times N)$ – матрица координат вершин комплекса, $R=\left(-\eta_R, \frac{1+\eta_R}{N-1}, ..., \frac{1+\eta_R}{N-1}\right)^T$ – $(N\times 1)$ – вектор.

В случае, если отраженная вершина $x_R(k)$ окажется «наилучшей» среди всех остальных точек комплекса, т. е.:

$$E\left(x_R(k)\right)<E\left(x_C(k)\right)<E\left(x_N(k)\right), \quad i=1, 2,..., N-1,$$

производится операция растяжения комплекса в направлении от центра тяжести $x_c(\kappa)$ до $x_R(k)$ согласно выражению

$$x_E(k)=x_C(k)+\eta_E\left(x_R(k)-x_C(k)\right)=X(k)E,$$

где η_E – параметр шага растяжения, часто полагаемый равным двум:

$$E=\left(-\eta_E\eta_R, \frac{1-\eta_E(1-\eta_R)}{N-1}, ..., \frac{1-\eta_E(1-\eta_R)}{N-1}\right)^T.$$

Если же $x_R(k)$ окажется наихудшей среди всех $x_i(k)$, комплекс сжимается согласно соотношению

$$x_S(k)=x_C(k)+\eta_S\left(x_R(k)-x_C(k)\right)=X(k)S,$$

где η_S – параметр шага сжатия, обычно полагаемый равным 0,5:

$$S=\left(-\eta_S\eta_R, \frac{1-\eta_S(1-\eta_R)}{N-1}, ..., \frac{1-\eta_S(1-\eta_R)}{N-1}\right)^T.$$

При $\eta_S=1, \eta_E=2, \eta_S=0{,}5$ приходим к простым выражениям:

$$R=\left(-1, \frac{2}{N-1}, ..., \frac{2}{N-1}\right)^T, E=\left(-2, \frac{1}{N-1}, ..., \frac{1}{N-1}\right)^T,$$

$$S=\left(-0{,}5, \frac{1}{N-1}, ..., \frac{1}{N-1}\right)^T.$$

Таким образом, в процессе своего движения к экстремуму оптимизируемой функции комплекс на каждой итерации теряет одну наихудшую вершину и приобретает одну новую точку так, что на (*k* +1)-й итерации новый комплекс также имеет *N* точек-вершин.

В генетических алгоритмах в результате селекции из популяции одновременно исключаются несколько особей с наихудшими (максимальными) значениями функции приспособленности. В связи с этим представляется целесообразным ввести алгоритм комплекс-метода с отражением, растяжением и сжатием сразу нескольких вершин [8, 9].

Итак, пусть на k-й итерации процесса оптимизации имеется комплекс $x_i(k), i = 1, 2, ..., N$ с $P<N$ наихудшими вершинами $x_{H_p}(k), p = 1, 2, ..., P$. Тогда координаты центра тяжести комплекса без вершин $x_{H_p}(k)$ задаются выражением

$$x_C(k) = \frac{1}{N-p}\left(\sum_{i=1}^{N} x_i(k) - \sum_{p=1}^{P} x_{H_p}(k)\right),$$

а процедура отражения описывается системой уравнений

$$\begin{cases} x_{R_1}(k) = x_C(k) + \eta_R\left(x_C(k) - x_{H_1}(k)\right), \\ \cdot \\ \cdot \\ \cdot \\ x_{R_P}(k) = x_C(k) + \eta_R\left(x_C(k) - x_{H_P}(k)\right). \end{cases}$$

В случае, если среди отраженных вершин оказывается $Q <= P$ наилучших, комплекс растягивается в их направлении согласно уравнениям

$$\begin{cases} x_{E_1}(k) = x_C(k) + \eta_E\left(x_{R_1}(k) - x_C(k)\right), \\ \cdot \\ \cdot \\ \cdot \\ x_{E_Q}(k) = x_C(k) + \eta_R\left(x_{R_Q}(k) - x_C(k)\right). \end{cases}$$

Если, далее, среди отражаемых вершин окажется $U <= P$ наихудших, комплекс сжимается в их направлении согласно уравнениям

$$\begin{cases} x_{S_1}(k) = x_C(k) + \eta_S\left(x_{R_1}(k) - x_C(k)\right), \\ \cdot \\ \cdot \\ \cdot \\ x_{E_U}(k) = x_C(k) + \eta_S\left(x_{R_U}(k) - x_C(k)\right). \end{cases}$$

Таким образом, комплекс-метод приобретает черты генетического алгоритма, у которого в результате селекции на каждой итерации из популяции удаляется несколько наихудших особей.

Объединяя введенную модификацию комплекс-метода с холландовской генетической процедурой, приходим к алгоритму, реализующему идею искусственного отбора, состоящую в данном случае в том, что из популяции не только удаляются наихудшие особи, но и одновременно создаются их «антиподы», обладающие улучшенными свойствами.

Работа такого алгоритма образована последовательностью следующих шагов:

– создание начальной популяции, образованной $P(0)$ особями хромосомами – вершинами комплекса;

– операция кроссовера с увеличением популяции $P_{CR}(0) > P(0)$;

– операция мутации $P_M(0) > P_{CR}(0)$;

– операция инверсии $P_1(0) > P_M(0)$;

– первая селекция (определение наихудших особей) без сокращения популяции $P_{SEL1}(0) = P_I(0)$;

– операция отражения с удалением P наихудших особей $P_R(0) < P_{SEL1}(0)$;

– операция растяжения без увеличения популяции $P_E(0) = P_R(0)$;

– операция сжатия без увеличения популяции $P_I(0) = P_E(0)$;

– вторая селекция с удалением $P_W(0)$ наихудших особей $P_{SEL2}(0) = P_I(0) = P(1)$ и формирование популяции $P(1)$ для следующей итерации алгоритма.

ВЫВОДЫ

Описанный в разделе математический аппарат холландовских генетических алгоритмов имеет ряд недостатков. В частности, они характеризуются низкой скоростью сходимости, не позволяющей им отыскивать решение за приемлемое время. Также генетические алгоритмы являются чувствительными к выбору параметров алгоритма, например, размера популяции, вероятностей кроссовера и мутации и т. п.

Эти и некоторые другие особенности генетических алгоритмов послужили толчком к созданию их различных модификаций. В некоторых модификациях, например, предлагается использовать, кроме классических генетических операторов кроссовера, мутации и инверсии дополнительные операторы. Например, такие как операторы объединения (fusion) и разделения (fission). Операция объединения заключается в том, что два аллеля соединяются в один. Операция разделения предполагает замену одного аллеля другим случайным аллелем. В результате происходит разделение кластеров [10].

В основе рассматриваемого алгоритма лежит синтез обычного эволюционного генетического подхода с идеями адаптационной оптимизации и, прежде всего, последовательного комплекс-метода отыскания экстремума функций многих переменных. При этом в каждый момент времени текущая популяция отождествляется с «облаком» – комплексом точек в пространстве переменных-факторов, а кроме традиционных генетических операторов мутации, кроссовера и инверсии дополнительно вводятся операторы комплекс-поиска, такие как отражение, растяжение и сжатие. Работа предложенного алгоритма протестирована на выборке Reuters-21578 [8, 9]. Было установлено, что предложенный алгоритм работает быстрее и дает более точные результаты (в среднем 8–10 %) по сравнению со стандартными генетическими алгоритмами.

СПИСОК ЛИТЕРАТУРЫ

1. *Асеев, Г. Г.* Проблема обнаружения нового знания в хранилищах данных методами Knowledge Discovery in Databases / Г. Г. Асеев // Вестник НТУ «ХПИ». – 2006. – № 19. – С. 62–70.
2. *Асеев, Г. Г.* Методы интеллектуальной предобработки данных в электронных хранилищах / Г. Г. Асеев // Радіоелектроніка, інформатика, управління. – 2010. – № 2(23). – С. 106–111.
3. *Асеев, Г. Г.* Методы интеллектуального анализа данных в электронных хранилищах / Г. Г. Асеев // Бионика интеллекта : науч.-техн. журнал. – 2008. – № 1(70). – С. 28–33.
4. *Растригин, Л. А.* Случайный поиск – специфика, этапы истории и предрассудки / Л. А. Растригин // Вопросы кибернетики. – Вып. 33. – 1988. – С. 3–12.
5. *Holland, J. H.* Adaptation in natural and artificial systems / John H. Holland. – Ann Arbor : University of Michigan Press, 1985. – 305 p.
6. *Rechenberg, I.* Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der Biologischen Information / Rechenberg I. – Freiburg : Fromman, 1983. – P. 135–143.
7. *Goldberg, D. E.* Genetic algorithms in search, optimization, and machine learning / David E. Goldberg. – [USA] : Addison-Wesley, 1989. – 752 p.
8. *Волкова, В. В.* Возможностная фаззи-кластеризация текстовых массивов в реальном времени на основе самообучающейся нейронной сети / В. В. Волкова, Б. В. Колчигин // Факультетская научно-практическая молодежная школа-семинар студентов, аспирантов и молодых ученых «Информационные интеллектуальные системы» : тезисы докл. – Х. : ХНУРЭ, 2008. – С. 22–25.
9. *Волкова, В. В.* Комбинированное обучение самоорганизующихся карт с нечетким выводом / В.В. Волкова, Е. В. Махиборода // Факультетская научно-практическая молодежная школа-семинар студентов, аспирантов и молодых ученых «Информационные интеллектуальные системы» : тезисы докл. – Харьков : ХНУРЭ, 2008. – С. 30–33.
10. *Рассел, С.* Искусственный интеллект. Современный подход / С. Рассел, П. Норвиг. – М. : Вильямс, 2006. – 1408 с.

Стаття надійшла до редакції 21.01.2011.

Асєєв Г. Г.

МЕТОДИ ІНТЕЛЕКТУАЛЬНОГО АНАЛІЗУ ДАНИХ В ЕЛЕКТРОННИХ СХОВИЩАХ: ГЕНЕТИЧНІ АЛГОРИТМИ

Представлено один з можливих методів інтелектуального аналізу даних в електронних сховищах великого об'єму – генетичні алгоритми і їх модифікація.

Ключові слова: інтелектуальний аналіз, електронне сховище, нейронна мережа, генетичні алгоритми.

Aseyev G. G.

METHODS OF INTELLECTUAL ANALYSIS OF DATA IN ELECTRONIC DEPOSITORIES: GENETIC ALGORITHMS

One of the possible methods of data intellectual analysis in high-volume electronic depositories is presented – genetic algorithms and their modification.

Key words: intellectual analysis, electronic depository, neuron network, genetic algorithms.

УДК 004:519.2

Кротких С. С.[1], Кириченко Л. О.[2]

[1]Аспирант Харьковского национального университета радиоэлектроники
[2]Канд. техн. наук, доцент Харьковского национального университета радиоэлектроники

ИССЛЕДОВАНИЕ ВЫЗВАННЫХ ПОТЕНЦИАЛОВ В ЭЭГ ЧЕЛОВЕКА С ПОМОЩЬЮ ДИСКРЕТНОГО ВЕЙВЛЕТ-ПРЕОБРАЗОВАНИЯ

В работе с помощью дискретного вейвлет-преобразования проведен анализ изменения частотной структуры электроэнцефалограммы с вызванными потенциалами после воздействия стимула. Реализован метод определения времени реакции на стимул, основанный на оценивании вейвлет-энтропии и относительной вейвлет-энтропии сегментов энцефалограммы.

Ключевые слова: электроэнцефалограмма, вызванные потенциалы, дискретное вейвлет-преобразование, спектр вейвлет-энергии, вейвлет-энтропия, относительная вейвлет-энтропия.

ВВЕДЕНИЕ

Головной мозг человека обладает электрическим полем, которое характеризуется электрическими скачками от нескольких милиампер до нескольких сотен милиампер. Каждый нейрон мозга генерирует изменение электрического потенциала, этот процесс можно измерить. Для анализа функций головного мозга широко используется электрический энцефалограф, который позволяет измерить изменение электрического потенциала в отдельных участках головного мозга человека. Электроэнцефалограмму (ЭЭГ) можно рассматривать как фиксацию последовательности умственных задач, выполняемых мозгом субъекта. Различные умственные задачи или условия, в которых находится человек, имеют свои уникальные отражения в результирующем ЭЭГ сигнале. Таким образом, наличие тех или иных изменений в

сигнале может свидетельствовать об изменении условий или внутреннего физиологического состояния человека [1, 2].

ЭЭГ измеряет электрический потенциал с помощью нескольких электродов, расположенных в определенных точках на поверхности головы субъекта. Количество электродов, в зависимости от модели энцефалографа, может быть от 8 до 64. Если субъект жив, электрическая мозговая активность присутствует постоянно. За исключением воздействий мускульной активности и моргания глаз, сигнал распределен на частотах от 0,5 Гц до 40 Гц. Этот промежуток принято разделять на частотные диапазоны, называемые ритмами. Под понятием «ритм» на ЭЭГ подразумевается определенная полоса частот, соответствующая некоторому состоянию мозга. К ритмам ЭЭГ взрослого бодрствующего человека относятся альфа-, бета-, гамма-, дельта- , тета- ритмы (табл. 1). Присутствие каждого из диапазонов зависит от условий и состояния субъекта [2].

Одним из методов исследования активности головного мозга является использование вызванных потенциалов, связанных с событиями (ВП), которые представляют реакцию мозга на внешние события (стимулы). В англоязычной литературе распространен термин ERP (event-related potentials). В зависимости от стимула существуют зрительные, слуховые, соматосенсорные ВП, возможно получение ВП при раздражении любого периферического нерва.

Традиционная методика регистрации ВП заключается в том, что в течение определенного времени после подачи стимула производится вычисление амплитуд ЭЭГ через временные интервалы, зависящие от частоты квантования. Полученные данные суммируются, в результате чего амплитуда стабильно возникающих после стимула потенциалов неуклонно возрастает, а амплитуда ритмов спонтанной ЭЭГ в той же степени уменьшается. Для получения истинных амплитуд ВП амплитуда в каждой точке делится на число стимулов.

По скорости реакции на стимул (латентности) ВП делятся на компоненты, вызывающие быструю реакцию в стволе головного мозга (реакция возникает в течении 0–12 мс с момента стимуляции), компоненты средней задержки (12–50 мс после стимуляции) и компоненты с долгой задержкой или когнитивные компоненты ВП (реакция возникает начиная от 50 мс после стимуляции и обычно до 500 мс). ВП с долгой задержкой связаны с когнитивными процессами, такими как обращение к памяти, эмоции или выполнение задач на внимание. Изменение амплитуды и латентности определенных компонентов позволяет диагностировать различные заболевания в ранней стадии развития, например, поражения проводящих путей или нервных центров, предынсультные состояния, психические заболевания, эпилепсию и др. [3–5].

Таблица 1. Основные ритмы ЭЭГ человека

Ритмы	Частотный диапазон
Гамма (μ)	40–70 Гц
Бета (β)	14–30 Гц
Альфа (α)	8–13 Гц
Тета (θ)	4–7 Гц
Дельта (δ)	0,5–3 Гц

ПОСТАНОВКА ЗАДАЧИ

Основным инструментом анализа частотных характеристик сигнала является преобразование Фурье. Показателем меры спектральной сложности сигнала является спектральная энтропия. Вся энергия упорядоченного сигнала, такого как синусоида, сконцентрирована в соответствующих гармониках, что говорит о низкой энтропии сигнала. С другой стороны, зашумленный сигнал содержит широкий диапазон частот, следовательно, обладает высокой энтропией. Однако применение спектрального анализа к исследованию ЭЭГ весьма ограничено, поскольку участки ВП являются весьма кратковременными и сильно нестационарными. Этот недостаток может быть частично решен использованием оконного преобразования Фурье. Но при узком окне частотное разрешение будет слишком мало, а при широком – временная локализация становится неточной. Это ограничение становится критическим, когда сигнал содержит кратковременные изменения частот, такие как ВП. Преодолеть это ограничение можно, используя частотно-временное представление сигнала, как это предусматривается вейвлет-преобразованием, не требующим допущений о стационарности. Вейвлет-анализ вызванных потенциалов в ЭЭГ человека и использование вейвлет-энтропии как меры упорядоченности участков ЭЭГ-сигнала было рассмотрено в работах [5–9].

Целью представленной работы является анализ изменения частотной структуры ЭЭГ после воздействия стимула с помощью дискретного вейвлет-преобразования и определение времени реакции на внешние воздействия, основанное на применении вейвлет-энтропии.

КРАТКИЕ СВЕДЕНИЯ ИЗ ВЕЙВЛЕТ-АНАЛИЗА

Рассмотрим основные методы исследования стохастических временных рядов, основанные на применении вейвлет-анализа [10–13]. Основная идея кратномасштабного вейвлет-анализа заключается в том, что разложение ряда производится по ортогональному базису, образованному сдвигами и кратномасштабными копиями вейвлетной функции. Базисные функции $\psi(t)$ называются вейвлетами, если они удовлетворяют ряду условий, в частности, определены на пространстве комплекснозначных функций с ограниченной энергией, колеблются вокруг оси абсцисс, быстро сходятся к нулю и имеют нулевой момент первого порядка.

Произвольная функция $s(t)$ может быть разложена по вейвлет-базису $\psi_{m,k}(t) = a^{\frac{m}{2}}\psi(a^m t + k)$ с параметрами масштаба a и сдвига k:

$$s(t) = \sum_{m,k=-\infty}^{\infty} S_{m,k}\psi_{m,k}(t), \qquad (1)$$

где коэффициенты вейвлет-спектра $S_{m,k}$ определяются скалярным произведением

$$S_{m,k} = \langle s(t), \psi_{m,k}(t) \rangle = \int_{-\infty}^{\infty} s(t), \psi_{m,k}(t) dt . \qquad (2)$$

В дискретном вейвлет-преобразовании (ДВП) параметры масштаба и сдвига задаются обычно в виде степенных функций: $a = 2^{-m}$, $k = k_0 \cdot 2^{-m}$, m, $k_0 \in Z$. Дискретные вейвлеты используются, как правило, в паре со связанными с ними дискретными скейлинг-функциями. Скейлинг-функции имеют с вейвлетами общую область задания и определенное соотношение между значениями. При заданных материнском вейвлете ψ и соответствующей скейлинг-функции φ аппроксимирующие коэффициенты $\mathrm{apr}(j,k)$ и детализирующие коэффициенты $\det(j,k)$ ДВП для процесса $X(t)$ определяются следующим образом:

$$\mathrm{apr}(j,k) = \int_{-\infty}^{\infty} X(t)\varphi_{j,k}(t)dt ,$$

$$\det(j,k) = \int_{-\infty}^{\infty} X(t)\psi_{j,k}(t)dt , \qquad (3)$$

где j – параметр масштаба, k – параметр сдвига ($j, k \in Z$) и

$$\varphi_{j,k}(t) = 2^{-j/2}\varphi(2^{-j}t - k),$$

$$\psi_{j,k}(t) = 2^{-j/2}\psi(2^{-j}t - k). \qquad (4)$$

Кратномасштабный анализ, выполняемый с помощью ДВП, заключается в разбиении исследуемого временного ряда на две составляющие: аппроксимирующую и детализирующую, с последующим аналогичным дроблением аппроксимирующей до заданного уровня декомпозиции сигнала. Временной ряд $X(t)$ представляется в виде суммы аппроксимирующей компоненты $\mathrm{approx}_N(t)$ и детализирующих компонент $\mathrm{detail}_j(t)$:

$$X(t) = \mathrm{approx}_N(t) + \sum_{j=1}^{N} \mathrm{detail}_j(t) =$$

$$= \sum_{k=1}^{N_a} \mathrm{apr}(N,k)\varphi_{j,k}(t) + \sum_{j=1}^{N}\sum_{k=1}^{N_j} \det(j,k)\psi_{j,k}(t) , \qquad (5)$$

где N – выбранный максимальный уровень разложения ряда, N_j – количество детализирующих коэффициентов на уровне j, N_a – количество аппроксимирующих коэффициентов на уровне N.

Выбор типа вейвлет-функции и количества уровней разложения является важным вопросом при выполнении ДВП. Обычно вейвлет-функция подбирается в зависимости от временных и частотных характеристик каждого анализируемого сигнала. Максимальный уровень разложения зависит от того, какие частотные диапазоны необходимо исследовать.

Величина энергии на заданном уровне вейвлет-разложения j с количеством детализирующих вейвлет-коэффициентов N_j определяется как

$$E_j = \frac{1}{N_j}\sum_{k=1}^{N_j} \det{}^2(j,k) . \qquad (6)$$

Набор величин E_j для каждого уровня разложения составляет спектр вейвлет-энергии ряда. Полная вейвлет-энергия спектра представляет собой сумму энергий каждого уровня:

$$E_{\mathrm{tot}} = \sum_{j=1}^{N} E_j , \qquad (7)$$

где N – максимальный уровень разложения. Относительная вейвлет-энергия показывает распределение вейвлет-энергии по уровням разложения:

$$p_j = \frac{E_j}{E_{\mathrm{tot}}} . \qquad (8)$$

Вейвлет-энтропия WE является количественной мерой упорядоченности сигнала и определяется по формуле:

$$WE = -\sum_{j=1}^{N} p_j \ln(p_j) . \qquad (9)$$

Относительная вейвлет-энтропия RWE является мерой подобия распределений вейвлет-энергий $\{p_j\}$ и $\{g_j\}$ для двух сигналов или двух участков одного сигнала и вычисляется как

$$\mathrm{RWE}(p|q) = \sum_{j=1}^{N} p_j \ln\left(\frac{p_j}{q_j}\right) . \qquad (10)$$

Значение относительной вейвлет-энтропии строго положительно и близко к нулю, если распределения энергий двух сигналов близки. Чем больше значение RWE, тем больше расхождение между распределениями и, соответственно, между сигналами.

АНАЛИЗ СТРУКТУРЫ ВРЕМЕННЫХ РЯДОВ С ПОМОЩЬЮ ВЕЙВЛЕТ-ЭНТРОПИИ

Покажем на модельных примерах, что адаптивное оценивание вейвлет-энтропии позволяет отследить появление сегментов с различным характером поведения. Для проведения численных исследований в качестве основных

модельных временных рядов были выбраны гармонические сигналы и реализации процессов авторегрессии.

На рис. 1 вверху слева представлен модельный сигнал, до середины реализации являющийся суммой двух синусов частотой 2 и 7 Гц, а после – синусоидой 13 Гц. Справа представлены спектры вейвлет-энергии на разных временных интервалах, полученные разложением сигнала на 7 уровней. Гармоники сигналов попадают в соответствующие диапазоны частот. Значения вейвлет-энтропий для приведенных спектров равны 1.33 и 1.21. Очевидно, что сигнал с одной гармонической составляющей является более упорядоченным, чем с двумя. Для определения структурных различий между первой и второй половинами сигнала, сигнал был разбит на участки длиной 1 с. Для каждого из участков по формуле (9) была рассчитана вейвлет-энтропия. Изменение величины вейвлет-энтропии во времени показано на рис. 1 внизу слева. Очевидно, что значение вейвлет-энтропии достигает наибольших значений в местах склеивания двух сигналов разной частоты.

На рис. 2 (вверху) представлена модельная реализация процессов авторегрессии, содержащая 5000 отсчетов. Первая часть реализации соответствует авторегрессии 1-го порядка с параметром $\phi = 0{,}3$. Вторая часть является авторегрессией 2-го порядка с коэффициентами $\phi_1 = 0{,}3$ и $\phi_2 = 0{,}4$. Для определения структурных различий между первой и второй половинами временного ряда реализация была разбита на интервалы, содержащие 500 отсчетов каждый. Для каждого из участков была рассчитана относительная вейвлет-энтропия. В средней части рис. 2 показано изменение величины относительной энтропии, вычисленной согласно формуле (10), где в качестве базового спектра $\{p_j\}$ выбран спектр авторегрессии 1-го порядка. В нижней части рис. 2 представлены значения величины относительной энтропии, где в качестве базового спектра выбран спектр вейвлет-энергии авторегрессии 2-го порядка.

Несмотря на значительную близость корреляционной структуры рядов, значения относительной вейвлет-энт-

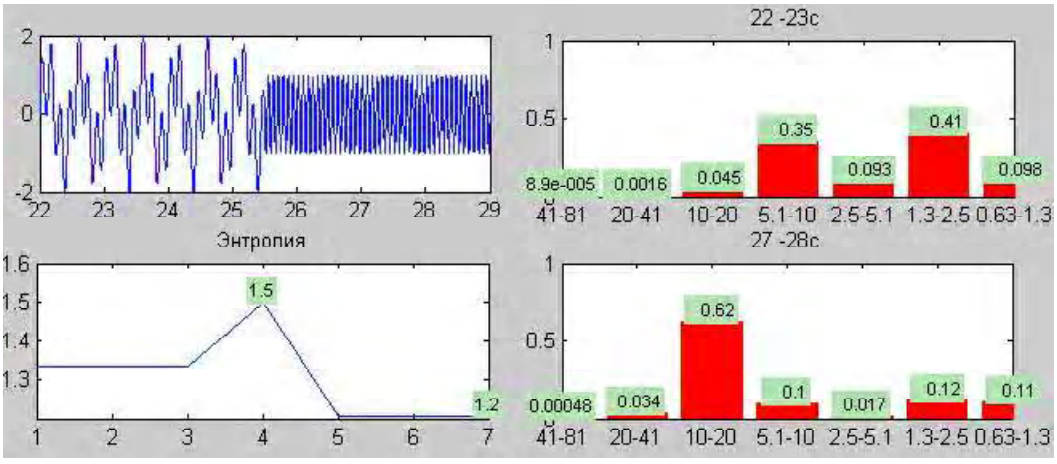


Рис. 1. Гармонический сигнал, спектр вейвлет-энергии и изменение вейвлет-энтропии

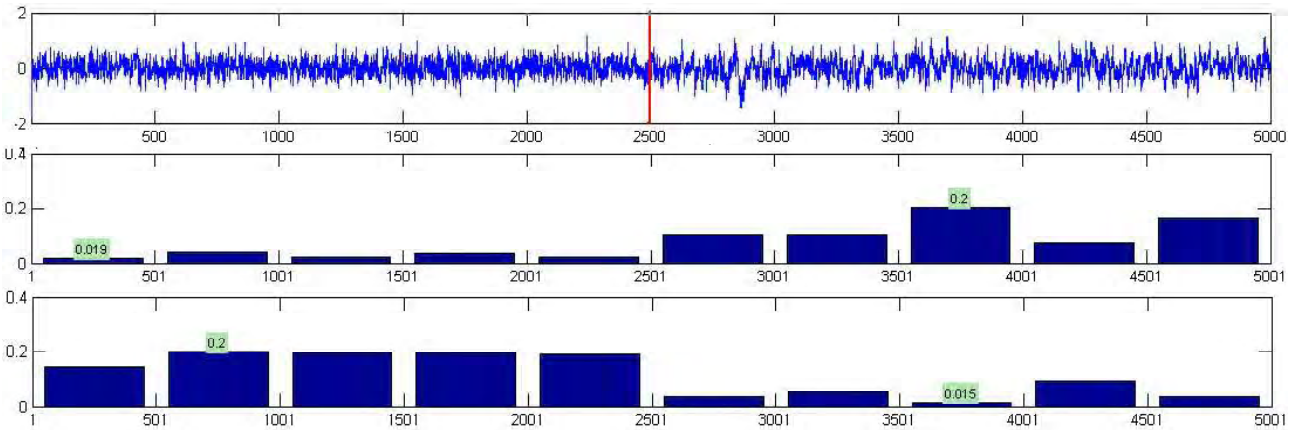


Рис. 2. Временной ряд авторегрессий 1-го и 2-го порядка (вверху); относительная вейвлет-энтропия, базовый спектр авторегрессии 1-го порядка (в середине); относительная вейвлет-энтропия, базовый спектр авторегрессии 2-го порядка (внизу)

ропии для одной половины ряда близки к нулю, а для второй на порядок больше. Это говорит о наличии частотной близости между участками сигнала и выбранным базовым вейвлет-спектром. Значения относительной вейвлет-энтропии изменяются при изменении частотной структуры временного ряда. Оба модельных примера показывают, что адаптивное оценивание вейвлет-энтропии позволяет успешно обнаруживать участки упорядоченности и изменения частотных компонент в различных сигналах.

ПРИМЕНЕНИЕ ОТНОСИТЕЛЬНОЙ ВЕЙВЛЕТ-ЭНТРОПИИ ДЛЯ АНАЛИЗА ВЫЗВАННЫХ ПОТЕНЦИАЛОВ В ЭЭГ-СИГНАЛЕ ЧЕЛОВЕКА

Исходные данные для исследований в данной работе были получены на специализированном сайте [14]. Данные содержали ЭЭГ десяти здоровых людей, записанные с помощью электроэнцефалографа системы BCI2000, который включает 64 электрода. Для каждого из испытуемых было записано по два ЭЭГ-сигнала: фоновая ЭЭГ (субъект сидит в удобном кресле с открытыми глазами в расслабленном состоянии) и ЭЭГ субъекта, выполняющего задачу на внимание. Во втором случае человек сидит в удобном кресле, напротив него расположен экран, на котором с частотой 8 секунд появляются маркеры с правой или левой стороны. При появлении маркера справа, субъект должен сжать правую кисть, слева – левую. Сигнал регистрируется с максимальной частотой 160 Гц. Длительность каждой записи ЭЭГ составляет 4 минуты.

Для исследования частотной структуры ЭЭГ было произведено многоуровневое вейвлет-разложение исходных сигналов. Количество уровней разложения было выбрано согласно основным исследуемым ритмам мозга: 40–80 Гц (гамма-ритм), 20–40 Гц (бета-ритм), 10–20 Гц (альфа-ритм), 5–10 Гц (тета-ритм), 2,5–5 Гц (дельта-ритм). Вейвлет-спектр не повторяет точно частотные ритмы человека, так как в соответствии с алгоритмом ДВП на последующих уровнях можно получить частотные диапазоны только в два раза короче, чем на предыдущем уровне. В качестве базовой вейвлет-функции был использован вейвлет семейства Добеши (Daubechie) степени 5.

На рис. 3 (вверху) показан график ЭЭГ человека, релаксирующего с закрытыми глазами. На соответствующем спектре вейвлет-энергии просматривается доминирование альфа-ритма, который активизируется при расслаблении с закрытыми глазами. В нижней части рис. 3 представлен график ЭЭГ человека, выполняющего сжимание и разжимание кисти руки. В этом случае на спектре вейвлет-энергии заметно преобладание низких частот в диапазоне 0,6–1,3 Гц.

При анализе ВП время реакции на стимул можно определять двумя способами [7]. В первом случае исследуется динамика вейвлет-энтропии WE. Время реакции t_{WE} – это время после стимула, за которое вейвлет-энтропия показывает наименьшее значение с момента подачи стимула. Данное время необходимо на перестройку частоты в период после стимула. Во втором случае анализируется динамика относительной вейвлет-энтропии, которая рассчитывается по формуле (10) с фоновым ЭЭГ-сигналом, когда субъект находится в расслабленном состоянии. Время реакции t_{RWE} – это время после стимула, за которое относительная энтропия достигает максимума. В момент t_{RWE} участок ЭЭГ с вызванными потенциалами и фоновая ЭЭГ показывают максимальную степень расхождения.

Для расчета вейвлет-спектра фонового сигнала соответствующая запись ЭЭГ была разбита на равные непересекающиеся интервалы по 500 мс, для каждого в соответствии с формулами (6)–(8) были получены и усреднены значения вейвлет-энергии. Для ЭЭГ, содержащей стимулы, были проделаны следующие операции: выбраны участки длиной 4 секунды, начиная за 2 секунды до подачи сти-

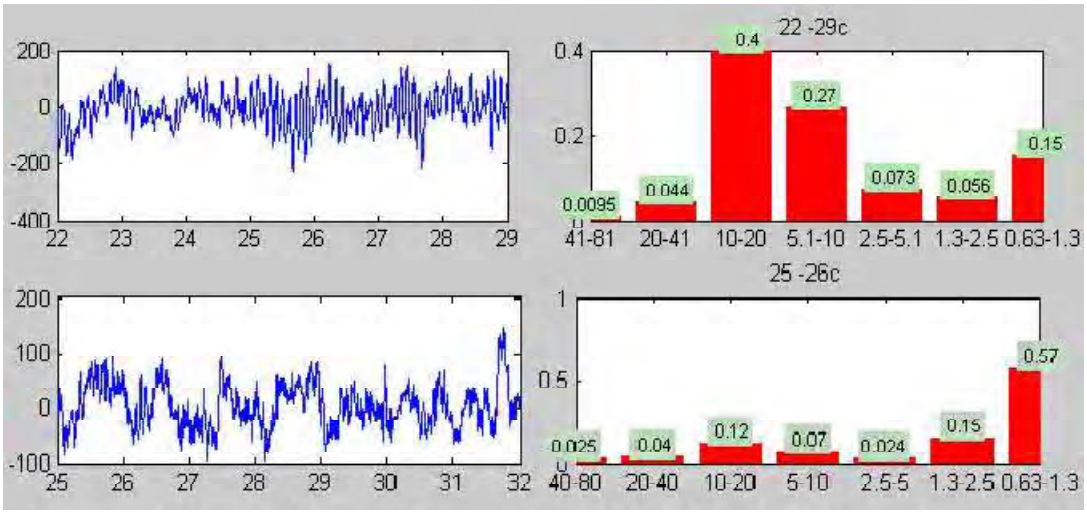


Рис. 3. ЭЭГ и спектр вейвлет-энергии: субъект в расслабленном состоянии (вверху), субъект выполняет сжимание и разжимание кисти (внизу)

мула; участки разбиты на непересекающиеся сегменты длиной 500 мс; для каждого участка получен спектр вейвлет-энергии и рассчитаны значения вейвлет-энтропии и относительной вейвлет энтропии к фоновой ЭЭГ.

Анализ распределений вейвлет-энергии (рис. 4) показывает, что после подачи стимула (вертикальная линия отмечает момент стимула) происходит смещение энергии в диапазон более низких частот: из альфа-ритма в тета-ритм. Время после стимула, в течение которого преобладает тета-ритм, варьируется от двух до четырех секунд.

Анализ распределения вейвлет-энергии в первые 500 мс после стимула показал, что распределение частот зависит от выбора электрода, с которого производится запись. Во фронтальной и центральной частях в период после стимула преобладает частота 5–10 Гц, а в затылочной – частота 10–20 Гц (рис. 5).

На рис. 6 представлены диаграммы изменения вейвлет энтропии (вверху) и относительной вейвлет-энтропии (внизу) для двух разных участков энцефаллограммы со стимулом. В результате воздействия стимула значение энтропии начинает убывать, пока не достигает минимума в момент t_{WE}. Это связано с концентрацией энергии в области определенной частоты, соответствующей данному воздействию. Относительная вейвлет-энтропия через время t_{RWE} после стимула демонстрирует всплеск, как следствие максимального расхождения между распределением энергии сигнала со стимулом и фонового сигнала. Проведенные исследования показали, что для разных случаев время реакции на внешнее воздействие варьировалось от 300 до 600 мс, причем разница между величинами t_{WE} и t_{RWE} достигала 200 мс.

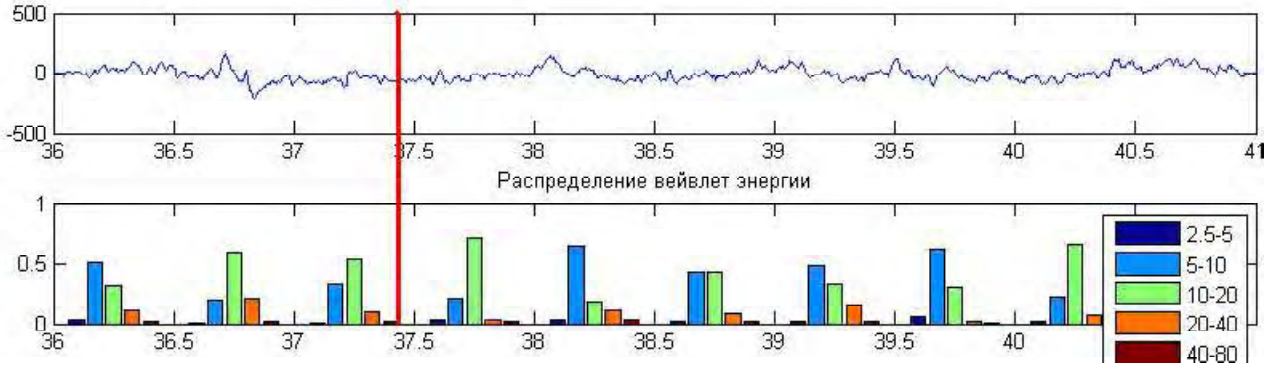


Рис. 4. Спектры вейвлет-энергии ЭЭГ-сигнала до и после стимула

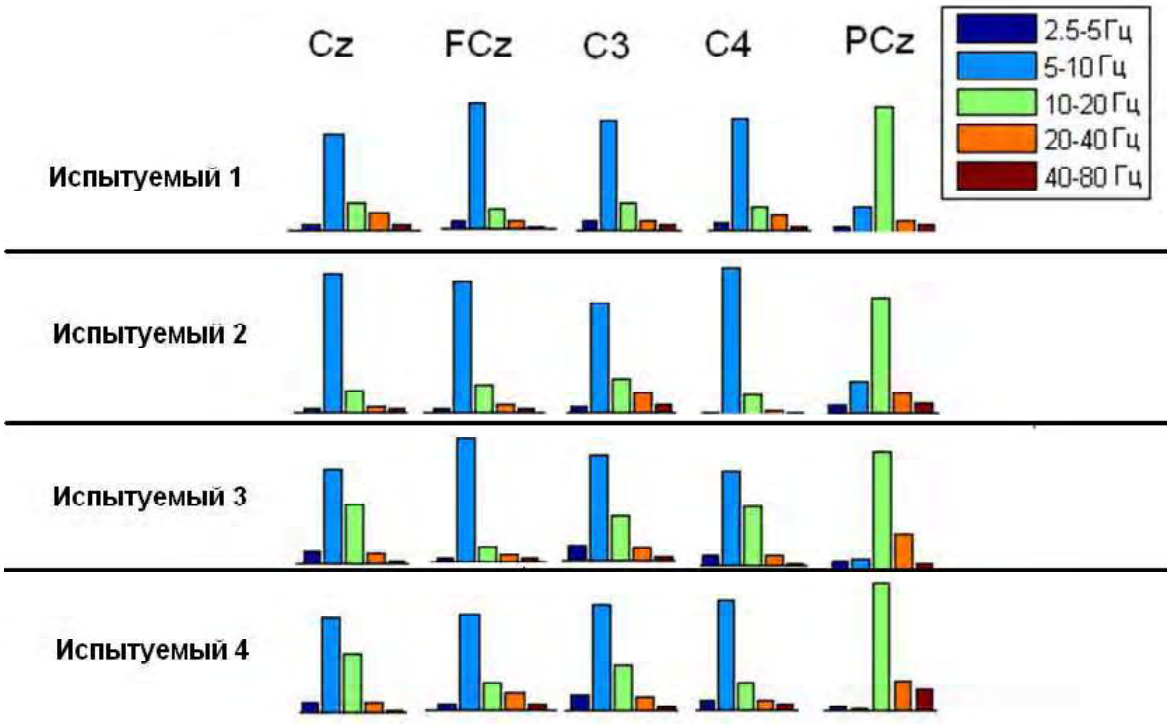
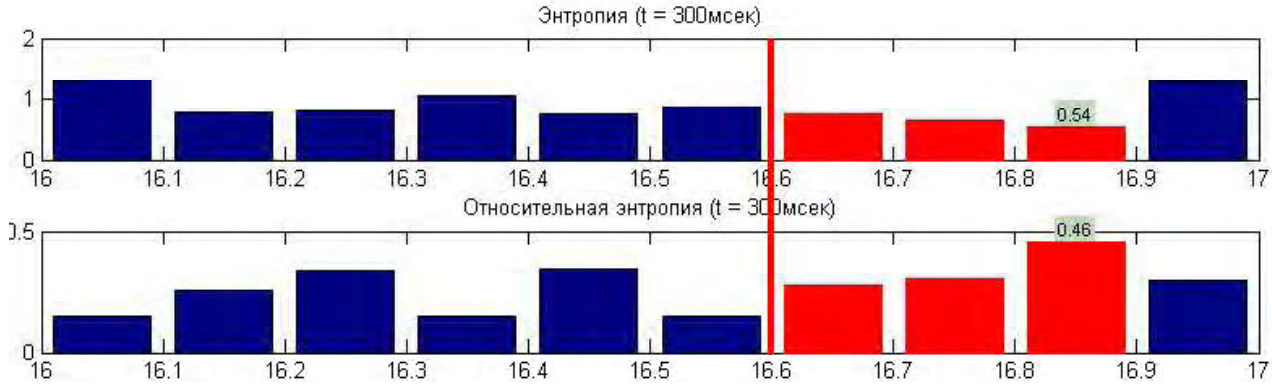
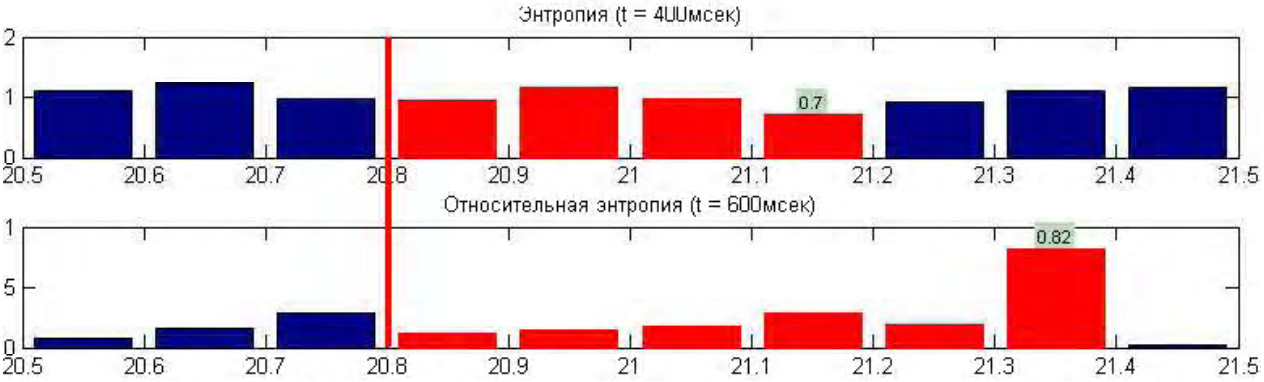


Рис. 5. Распределения вейвлет-энергии в первые 500 мс после стимула для 5 основных отведений ЭЭГ



а) Участок ЭЭГ 16–17 с



б) Участок ЭЭГ 20,5–21,5 с

Рис. 6. Изменение вейвлет-энтропии после стимула (вверху), изменение относительной энтропии после стимула (внизу); вертикальная линия отмечает моменты стимулов

ВЫВОДЫ

В данной работе продемонстрировано, что ДВП является эффективным инструментом для анализа ЭЭГ и обнаружения отличительных особенностей колебаний связанных с ВП до и после стимула. Исследования ЭЭГ показали, что в состоянии покоя с открытыми глазами в ЭЭГ человека преобладает альфа ритм, однако при подаче визуального стимула происходит кратковременная синхронизация, характеризующаяся снижением вейвлет-энтропии, затем наблюдается период длиной 300–600 мс с преобладанием тета-ритма. Адаптивное оценивание вейвлет-энтропии и относительной энтропии на основе фонового ЭЭГ-сигнала позволяет определить время реакции на внешний возбудитель. Результаты работы могут быть полезны при диагностике неврологических заболеваний, где необходима точная оценка времени реакции на стимул и характер колебании ЭЭГ после стимула.

СПИСОК ЛИТЕРАТУРЫ

1. *Рангайян, Р. М.* Анализ биомедицинских сигналов: практический подход: пер. с англ. / Р. М. Рангайян. – М. : Физматлит, 2007. – 440 с.
2. *Зенков, Л. Р.* Функциональная диагностика нервных болезней: руководство для врачей / Л. Р. Зенков, М.А. Ронкин. – М. : Медицина, 1991. – 640 с.
3. *Рутман, Э. М.* Вызванные потенциалы в психологии и психофизиологии / Э.М. Рутман. – М.: Наука, 1979. – 216 с.
4. Опыт применения вызванных потенциалов в клинической практике / под ред. В. В. Гнездицкого, А. М. Шамшиновой. – М. : АОЗТ «Антидор», 2001. – 480 с.
5. *Basar, E.* EEG Brain Dynamics. Relation between EEG and Brain Evoked Potentials / E.Basar. – Amsterdam: Elsevier, 1980. – 434 p.
6. *Basar, E.* Event-related oscillations are ‘real brain responses’/ wavelet-analysis and new strategies / E. Basar, M. Schurmann, T. Demiralp, C. Basar-Eroglu, A. Ademoglu // International Journal of Psychophysiology. –2001. – V. 39. – P. 91–127.
7. *Rosso, O. A.* Wavelet entropy: a new tool for analysis of short duration brain electrical signals / O. A. Rosso // Journal of Neuroscience Methods. – 2001. – V. 105. – P. 65–75.
8. *Yordanova, J.* Wavelet entropy analysis of event-related potentials indicates modality-independent theta dominance / J. Yordanova, V. Kolev, O. A. Rosso, M. Schьrmann, O.W. Sakowitz, M. Цzgцren, E. Basar // Journal of Neuroscience Methods. – 2002. –V. 117. – P. 99–109.
9. *Giannakakis, G. A.* Entropy Differentiations of Event Related Potentials in Dyslexia / G. A. Giannakakis, N. N. Tsiaparas, M. S. Xenikou, Ch. Papageorgiou, K. S. Nikita // proceeding 8th IEEE International Conference on BioInformatics and Bioengineering, Athens, Greece, 2008.
10. *Малла, С.* Вэйвлеты в обработке сигналов: пер. с англ. / С. Малла. – М. : Мир, – 2005. – 671 с.

11. *Смоленцев, Н. К.* Основы теории вейвлетов. Вейвлеты в MATLAB / Н. К. Смоленцев. – М. : ДМК Пресс, 2005. – 304 с.
12. *Дьяконов, В. П.* Вейвлеты. От теории к практике / В. П. Дьяконов – М. : СОЛОН-Пресс, 2002. – 440 с.
13 *Zunino, L.* Wavelet Entropy of Stochastic Processes / L. Zunino, D. G. Perez, M. Garavaglia, O. A. Rosso // Physica A. – 2007. – V. 379, N. 2. – P. 503–512.
14. PhysioNet: the research resource for complex physiologic signals [Электронный ресурс] – Режим доступа: www.physionet.org

Стаття надійшла до редакції 19.05.2011.

Кротких С. С., Кіріченко Л. О.

ДОСЛІДЖЕННЯ ВИКЛИКАНИХ ПОТЕНЦІАЛІВ У ЕЕГ ЛЮДИНИ ЗА ДОПОМОГОЮ ДИСКРЕТНОГО ВЕЙВЛЕТ-ПЕРЕТВОРЕННЯ

В роботі за допомогою дискретного вейвлет-перетворення проведено аналіз змінювання частотної структури електроенцефалограми з викликаними потенціалами після впливу стимулу. Було реалізовано метод визначення часу реакції на стимул, який заснований на оцінюванні вейвлет-ентропії та відносної вейвлет-ентропии сегментів енцефалограми.

Ключові слова: Електроенцефалограма, викликані потенціали, дискретне вейвлет-перетворення, спектр вейвлет-енергії, вейвлет-ентропія, відносна вейвлет-ентропія.

Krotkih S. S., Kirichenko L. O.

ANALYSIS OF EVENT-RELATED POTENTIALS OF EEG SIGNAL USING DISCRETE WAVELET TRANSFORM

In this work we use discrete wavelet transform for analyzes the frequency structure of EEG signal with evoked potentials after effect of stimulus. The method for determining the response time to a stimulus, based on the evaluation of the wavelet entropy and relative wavelet entropy EEG, has been implemented.

Key words: electroencephalogram, evoked potentials, discrete wavelet transform, wavelet energy spectrum, wavelet entropy, relative wavelet entropy.

УДК 004.853+004.832

Литвин В. В.

Канд. техн. наук, доцент Національного університету «Львівська політехніка»

МОДЕЛЮВАННЯ ІНТЕЛЕКТУАЛЬНИХ СИСТЕМ ПІДТРИМКИ ПРИЙНЯТТЯ РІШЕНЬ З ВИКОРИСТАННЯМ ОНТОЛОГІЧНОГО ПІДХОДУ

Досліджено функціонування інтелектуальних систем підтримки прийняття рішень, ядром баз знань яких є онтології. Розглядаються інтелектуальні агенти чотирьох типів. Для кожного типу розроблено метрику, яку використовують для визначення релевантності пропонованих системою рішень.

Ключові слова: інтелектуальна система підтримки прийняття рішень, інтелектуальний агент.

АНАЛІЗ ДОСЛІДЖЕНЬ ТА ПОСТАНОВКА ПРОБЛЕМИ В ЗАГАЛЬНОМУ ВИДІ

Технологія інтелектуальних систем підтримки прийняття рішень (ІСППР) є одним із найрозвинутіших напрямків штучного інтелекту. Дослідження у цій області полягають у розробці автоматизованих інформаційних систем, які застосовуються у тих областях діяльності людини, які вимагають логічного міркування, певної майстерності та досвіду.

Сучасний рівень розвитку ІСППР відбувається у двох напрямках розроблення інтелектуальних агентів (ІА) [1]:

– ІА, засновані на прецедентах (англійською – Case-Based Reasoning, або CBR);

– ІА планування діяльності (пошук у просторі станів).

Вибір ІА залежить від задачі. Метод виведення за прецедентами ефективний, коли основним джерелом знань про задачу є досвід, а не теорія; рішення не є унікальними для конкретної ситуації, а можуть бути використані в інших випадках; метою розв'язування задачі є отримати не гарантований вірний розв'язок, а кращий з можливих. Виведення, засноване на прецедентах, є методом побудови ІСППР, які приймають рішення щодо даної проблеми або ситуації за наслідками пошуку аналогій, що зберігаються в базі прецедентів [2]. Такий прецедент називають релевантним. З математичної точки зору серед елементів множини прецедентів $\mathrm{Pr} = \{\mathrm{Pr}_1, \mathrm{Pr}_2, ... \mathrm{Pr}_N\}$ релевантним Pr_k є прецедент для якого відстань до поточної ситуації S є найменшою, тобто

$$\mathrm{Pr}_k = \arg\min_i d(\mathrm{Pr}_i, S).$$

ІА планування діяльності повинен досягнути цільового стану. Насамперед він повинен побудувати план досягнення цього стану із всіма можливими альтернативами. Процес планування ґрунтується на декомпозиції. Задача планування ZP містить 3 складові [3]: множину

станів *St*, множину дій *A*, множину цільових станів *Goal* (станів мети); тобто

$$ZP = \langle St,\ A\ ,\ Goal \rangle\ .$$

Отже, для планування діяльності ІА повинен вміти оцінювати стани та дії.

Як бачимо для обох класів ІСППР необхідна метрика. У першому випадку для оцінювання релевантності прецедентів, у другому випадку – для оцінювання релевантності станів. Від способу визначення цієї метрики напряму залежить ефективність функціонування ІА. На наш погляд такий спосіб повинен базуватись на чіткому і аргументованому стандарті баз знань. У галузі інженерії знань таким стандартом стали онтології [4]. Тому нами запропоновано для побудови метрики використовувати онтології.

Під моделлю онтології *O* розуміють трійку вигляду:

$$O = \langle C,\ R,\ F \rangle,$$

де *C* – поняття; *R* – відношення між поняттями; *F* – інтерпретація понять та відношень (аксіоми). Аксіоми встановлюють семантичні обмеження для системи понять та відношень.

На сьогодні розрізняють три типи онтологій: предметно-орієнтовані (Domain-oriented), орієнтовані на прикладну задачу (Task-oriented) та загальні онтології (Top-level).

Мета роботи – побудувати моделі функціонування інтелектуальних систем підтримки прийняття рішень, які базуються на онтологіях в залежності від класу задач.

Постановка задачі. Побудувати метрики для оцінювання релевантності прецедентів в задачах пошуку аналогій та оцінювання релевантності станів для задач пошуку цільового стану. Ці метрики повинні базуватись на онтологіях. Апробувати отримані моделі функціонування ІСППР шляхом розроблення прикладних ІСППР.

КЛАСИФІКАЦІЯ ПРИКЛАДНИХ ЗАДАЧ З ТОЧКИ ЗОРУ ЇХ МЕТРИЗАЦІЇ

Проаналізувавши клас задач, для яких розробляють ІСППР, можна зробити висновок, що всі задачі можна поділити на два підкласи. Існує клас задач для яких суттєве значення понять (властивостей). Сюди відносяться задачі діагностики захворювань, розпізнавання образів, класифікація явищ на основі збору даних, тощо. Такі задачі назвемо ознаковими. Для іншого класу задач не є суттєвим значення понять, а скоріше їх семантика або частотність зустрічання термінів в тексті і т. д. Сюди можна віднести кластеризацію інформаційних ресурсів, класифікацію текстів згідно УДК, інтелектуальні пошукові системи, реферування та анотування текстових документів. Такий клас задач назвемо семантичними задачами. В результаті отримаємо поділ ІСППР за двома вимірами так, як це зображено на рис. 1. У кожній чверті перераховано окремі задачі, які попадають у відповідний клас.

Рис. 1. Класи задач, для розв'язування яких використовують ІСППР

Для ефективного функціонування ІА необхідно побудувати метрику, на основі якої визначати релевантність станів чи прецедентів. На наш погляд побудова такої метрики напряму залежить від класу задач: семантичні вони чи ознакові.

Отже, загалом, на нашу думку, виділяється чотири різних класи задач, які розв'язують ІСППР. Зріз за напрямами потребує дві різні функціональні моделі (пошук релевантних прецедентів та планування діяльності), зріз за типом задачі – використання двох різних метрик для розв'язування цих задач та оцінки якості отриманих розв'яз-ків. Далі розглянемо всі ці класи задач, однак, насамперед введемо поняття адаптивної онтології.

Ефективність адаптації онтології бази знань до особливостей предметної області визначають закладені в її структуру елементи та механізми її адаптації шляхом самонавчання під час експлуатації. Одним з підходів до реалізації таких механізмів є автоматичне зважування понять бази знань (БЗ) та семантичних зв'язків між ними під час самонавчання. Цю роль беруть на себе коефіцієнти важливості понять та зв'язків [5]. Коефіцієнт важливості поняття (зв'язку) – це чисельна міра, котра характеризує значущість певного поняття (зв'язку) у конкретній предметній області і динамічно змінюється за певними правилами у процесі експлуатації системи. Отже, ми розширимо поняття онтології, ввівши в її формальний опис коефіцієнти важливості понять та відношень. Тому таку онтологію ми будемо визначати як п'ятірку:

$$O = \langle C,\ R,\ F,\ W,\ L \rangle,$$

де *W* – важливість понять *C*; *L* – важливість відношень *R*.

Визначену, таким чином, онтологію будемо називати адаптивною, тобто такою, що адаптується до ПО за рахунок модифікації понять та коефіцієнтів важливості цих понять і зв'язків між ними [6]. Така онтологія однозначно представляється у вигляді зваженого концептуального графа (КГ) [7]. Тому метрику ми будемо будувати, використовуючи зважені КГ.

МЕТРИЗАЦІЯ ІНТЕЛЕКТУАЛЬНИХ АГЕНТІВ, БАЗОВАНИХ НА ПРЕЦЕДЕНТАХ

Вочевидь, що в залежності від прецеденту ваги понять різні. Тобто насправді W – вектор вимірності кількості прецедентів $W=(W_1,W_2,\ldots,W_N)$. Надалі будемо розглядати лише один прецедент, тобто нижній індекс у вазі понять опускатимемо.

Побудуємо метрику для семантичних задач, які використовують прецеденти. Нами пропонується визначати відстань між прецедентом і ситуацією, як суму відстаней між найважливішими поняттями прецедента та поточного випадку. Таких важливих понять може бути одне, два; однак якщо їх є більше-рівне трьох, то нами пропонується вибирати перші три. Ця кількість визначена на основі опитувань експертів різних ПО і вважається ними оптимальною. У такому випадку ми маємо 3 центри ваг i-го прецедента pr_i^1, pr_i^2, pr_i^3 і 3 центри ваг поточної ситуації s^1, s^2, s^3. Тоді існує 9 різних відстаней $d\left(\mathrm{pr}_i^j, s^k\right)$, j=1,2,3; k=1, 2, 3. Вибираємо 3 найменші з них та їх сумуємо. Отримана таким чином сума й буде відстанню між прецедентом та поточною ситуацією. Найважливішим є поняття, яке є центром ваг КГ. Центром ваг КГ є поняття, середня відстань від якого до всіх інших понять є найменшою. Вочевидь, що визначена таким чином відстань залежатиме від того як ми визначимо відстань між двома суміжними вершинами КГ. Для цього пропонується визначати відстані між вершинами, що з'єднані зв'язком як

$$d_{ij} = \frac{Q}{L_{ij}(W_i + W_j)}, \qquad (1)$$

де W_i та W_j – коефіцієнти важливості вершин C_i та C_j відповідно; L_{ij} – коефіцієнт важливості зв'язку між вершинами; Q – константа, яка залежить від конкретної онтології. Приймемо, що $L_{ii} = \infty$, тоді $d_{ii} = 0$.

Далі знаходимо центри ваг концептуального графа. Це перші три вершини для яких середня відстань $\bar{d}_i$ є найменшою:

$$\bar{d}_{i^*} = \min_i \bar{d}_i. \qquad (2)$$

Середня відстань $\bar{d}_i$ для вершини C_i обчислюється згідно формули:

$$\bar{d}_i = \frac{\sum_{j=1,\ j\neq 1}^{n} d_{ij}^*}{n-1}, \qquad (3)$$

де n – кількість вершин графа; d_{ij}^* – найкоротший шлях між вершинами C_i та C_j, який обчислюється за допомогою відомих алгоритмів, наприклад Форда, Дейкстри, Флойда-Уоршалла [8].

Далі згідно концептуального графа, що задає онтологію прецедента шукаємо відстань від даного прецедента до поточної ситуації. Якщо поняття поточної ситуації не входять в концептуальний граф, то онтологію даного прецедента доповнюємо онтологією всього ІА до якого входить цей прецедент. Якщо ж необхідне поняття далі не входить в онтологію ІА, то його відсутність зумовлює ріст відстані до безмежності, що означає не близькість прецедента із поточною ситуацією.

Зазначимо, що пропонована таким чином відстань задовольняє трьом аксіомам метрики [5].

Ефективність розробленої відстані покажемо на прикладі аналізу анотацій наукових статей. Розглянемо три анотації статей.

1. The work is carried out in a direction of information technologies development focused on the natural language information processing. On position of author offered model the problem of putting a language material in order with the help of the uniform standard, it is considered rather significant for the given class of technologies. It is one of the central problem on way of development of the effective technologies for language information processing. (ШІ, № 4, 2004, с. 613)

Відповідний концептуальний граф першої анотації поданий на рис. 2.

2. The article presents the basic concepts on researching and solving the task of automatic knowledge retrieval from text documents. Industrial system that solves the stated above task is described as well as one of its main application. (ШІ, №3, 2004, с. 668).

Відповідний концептуальний граф матиме вигляд, як наведено на рис. 3.

3. The paper is dedicated to the problem of automated text consistency analysis. It is proposed to implement text consistency via text logic analysis with the attraction of the knowledge of application domain, natural language and normative base. (ШІ, № 3, 2004, с. 660).

Концептуальний граф третьої анотації наведений на рис. 4.

Значення ваг понять та зв'язків взято із тестової онтології «напряму комп'ютерних наук» для прецеденту «штучний інтелект». У випадках, коли у графі немає зв'язку, вагу такого зв'язку вважали рівною 5, що є середнім значенням ваги зв'язків. Користуючись формулою (1), в якій Q приймалося рівним 100, отримаємо зважені графи, які наведено на рис. 5.

Використовуючи алгоритм Дейкстри та роблячи відповідні обчислення за формулами (2) та (3), отримаємо, що центрами ваг відповідних графів є:

$$\bar{d}^1 = \{11\} = \{\text{'technologies'}\},\ \bar{d}^2 = \{4\} = \{\text{'system'}\},$$

$$\bar{d}^3 = \{5\} = \{\text{'knowledge'}\}.$$

1-й і 3-й тексти легко зв'язуються за допомогою вершини 'natural_language' (5-а в 1-му тексті і 7-а в 3-му тексті). Тоді $d_{5,11}^1 = 0{,}42$; $d_{5,7}^3 = 0{,}23$. А відстань між

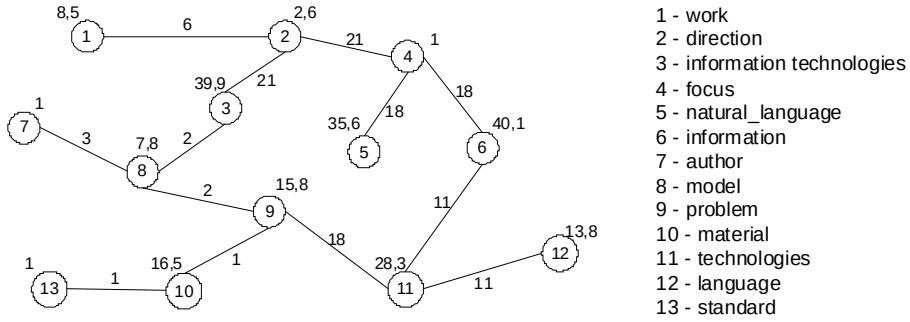


Рис. 2. Концептуальний граф 1-ї анотації

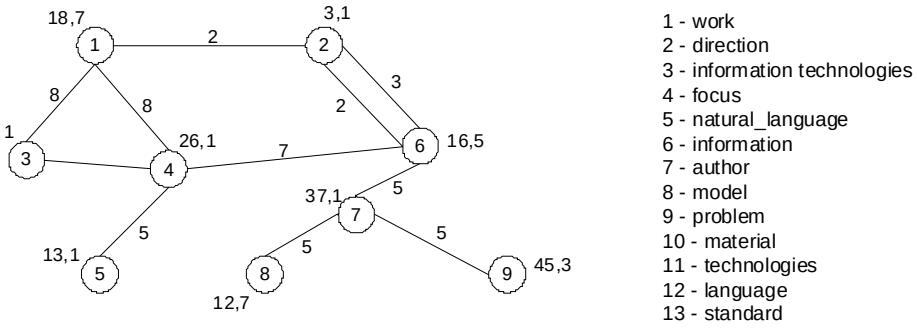


Рис. 3. Концептуальний граф 2-ї анотації

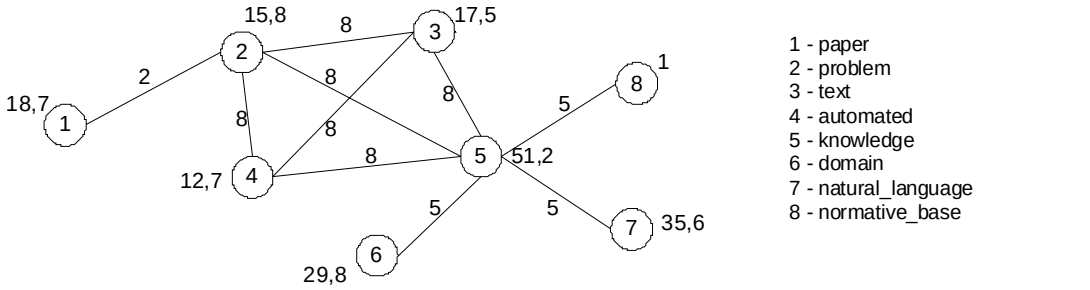


Рис. 4. Концептуальний граф 3-ї анотації

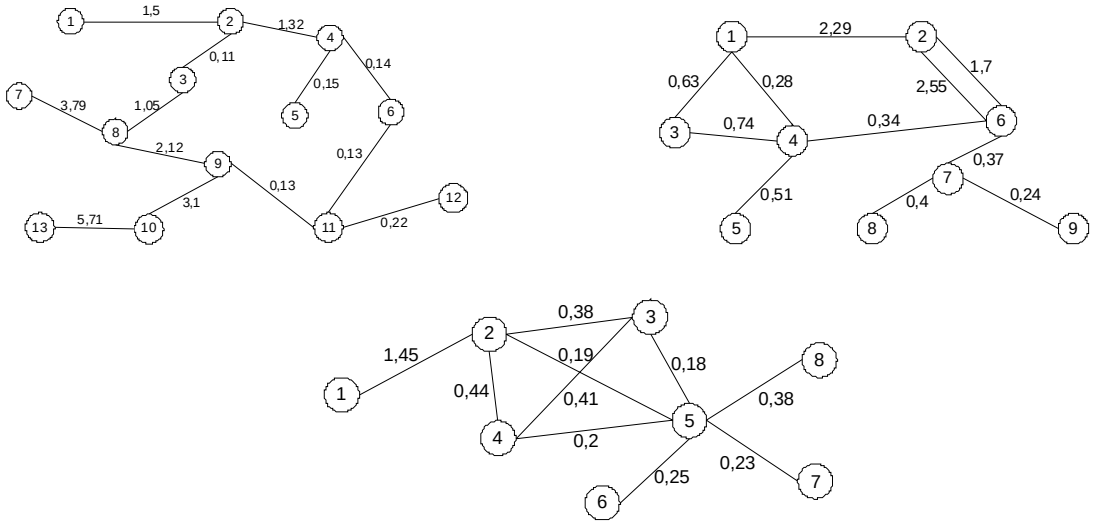


Рис. 5. Зважені графи для трьох анотацій

1-м і 3-м текстом тоді рівна 0,65.

$$\bar{d}^{13} = 0{,}42 + 0{,}23 = 0{,}65 \cdot$$

2-й і 3-й тексти легко зв'язуються за допомогою вершини 'knowledge' (7-а в 2-му тексті і 5-а в 3-му тексті). Тоді, $d^2_{4,7} = 0{,}71$, що є відстаню між 2-м та 3-м текстом, оскільки 'knowledge' – центр ваг 3-го графа. Тобто $\bar{d}^{23} = 0{,}71 \cdot$

Для обчислення відстані між 1-м і 2-м текстом скористаємося 3-м текстом, оскільки в перших двох немає спільних вершин. Тоді

$$\bar{d}^{12} = \bar{d}^{13} + \bar{d}^{23} = 0{,}65 + 0{,}71 = 1{,}36 \cdot$$

Отже, найближчими за змістом є 1-й і 3-й текст, найдальшими є 1-й і 2-й текст.

Побудуємо метрику для ознакових задач, які використовують прецеденти.

Нехай множина прецедентів $\mathrm{Pr}=\{\mathrm{Pr}_1, \mathrm{Pr}_2, \ldots, \mathrm{Pr}_N\}$ описується характеристиками (властивостями) $X=\{x_1, x_2, \ldots, x_M\}$. D_i – домен властивості x_i, w_{i_l} – коефіцієнт важливості властивості x_{i_l} прецедента Pr_i. Значення властивості x_i позначатимемо $z_i = z(x_i)$. Отже

$$\mathrm{Pr}_i \leftrightarrow X_i = \left\{x_{i_1} = z_{i_1}, x_{i_2} = z_{i_2}, \ldots, x_{i_k} = z_{i_k}\right\}, \text{ де } z_{i_j} \in D_{i_j}.$$

Позначимо I_i – множина індексів властивостей прецедента Pr_i. Тоді відстань між прецедентом Pr_i та поточною ситуацією S визначається як:

$$d_i = \sum_{i_l \in \bar{I}_i} \varphi(z_{i_l}, z^S_{i_l}), \qquad (4)$$

де z_{i_l} – значення властивості x_{i_l} прецедента Pr_i; z^S_{ij} – значення властивості x_{i_l} поточної ситуації S; $\bar{I}_i$ – множина індексів важливих властивостей прецедента Pr_i; $\bar{I}_i = \bar{I}_{i_1} \cup \bar{I}_{i_2} \cup \ldots \cup \bar{I}_{iN_i}$; N_i – кількість властивостей, необхідних розглянути щоб прийняти рішення стосовно прецедента Pr_i. Тобто

$$\bar{I}_{i1} = \left\{ i_{s1} \middle| i_{s1} = \arg\max_{i_l \in I_i} w_{i_l} \right\},$$

$$\bar{I}_{i2} = \left\{ i_{s2} \middle| i_{s2} = \arg\max_{i_l \in I_i / i_{s1}} w_{i_l} \right\},$$

$$\bar{I}_{i3} = \left\{ i_{s3} \middle| i_{s3} = \arg\max_{i_l \in I_i / i_{s1} / i_{s2}} w_{i_l} \right\}, \ldots$$

Розглянемо функцію $\varphi(\xi, \eta)$. Очевидно, що ξ – може бути діапазоном, тобто нечіткою підмножиною $\xi \subseteq D$, де D – універсальна множина; числовим значенням або нечисловим значенням. В залежності від цього $\varphi(\xi, \eta)$ визначається по своєму, а саме:

$$\varphi(\xi, \eta) = \begin{cases} 1 - \mu_\xi(\eta), \ \xi \text{ – нечітка множина}, \\ \lambda \cdot |\xi - \eta|, \ \xi, \eta \text{ – числові значення}, \\ 1 - \mu(\xi, \eta), \ \xi, \eta \text{ – нечислові значення}, \end{cases} \qquad (5)$$

де $\mu_\xi(\eta)$ – коефіцієнт впевненості того, що η належить нечіткій підмножині ξ; λ – числова величина, яка залежить від ПО, щоб добуток $\lambda \cdot |\xi - \eta| \in [0, 1]$; $\mu(\xi, \eta) \in [0, 1]$ – нечітка величина подібності значень ξ та η. Наприклад $\mu(\xi, \eta) = 1$, якщо $\xi = \eta$, $\mu(\xi, \eta) = 0{,}9$, якщо $\xi \approx \eta$, $\mu(\xi, \eta) = 0$, якщо $\xi \neq \eta$.

Розглянемо приклад функціонування ІА в області медицини, а саме захворювань опорно-рухового апарату. За основу візьмемо результати досліджень, що одержали у Львівському Національному медичному університеті ім. Данила Галицького Е. Х.Заремба й О. О. Зімба. З деякими з них можна ознайомитися в [9]. Прецедентами є можливі захворювання. Для прикладу розглянемо три хвороби: Pr={'Ревматизм', 'Артрит', 'Подагра'}.

Провівши онтологічний інжиніринг ПО, ми одержали множину властивостей, які необхідно дослідити та їх важливість в залежності від прецеденту. Наведемо деякі з них: X={*Ступінь недиференційованої дисплазії сполучної тканини*, *Концентрація ендотеліну-1 у плазмі крові*, *Температура*, *Біль у суглобі*}. Розглянемо детальніше ці ознаки. Домен першої є нечислові значення, другої і третьої – відрізки, четвертої – бінарне значення. Так $D_{С\_н\_д\_с\_т}$ = {слабке, середнє, сильне}, $D_{К\_е\_п\_к}$=[0; 3,5], $D_{\text{Температура}}$=[36,42], $D_{\text{Біль\_у\_суглобі}}$={Так(1),Немає(0)}. Значення цих ознак для ревматизму: $\text{Температура}_{\text{Ревматизм}}$ = ={(36|0), (37|0,6), (38|1), (39|1), (40|0,7)} – нечітка множина, $\text{Біль у суглобі}_{\text{Ревматизм}}$=«Так(1)».

Важливість цих властивостей (коефіцієнти W) ми одержали методами статистичного аналізу (було досліджено 110 пацієнтів). Так для захворювання на ревматизм важливими симптомами є: наявність лади (w=0,7), астенічна статура (w=0,65), тонка шкіра (w=0,45), сколіоз (w=0,35). Перший симптом являє собою бінарну величину, що приймає значення з множини {Так(1),Немає(0)}, дві наступні – нечіткі величини, що приймають значення з відрізка [0; 1] (зрозуміло, що 0 – відсутність симптому, 1 – повна впевненість у його наявності) і четвертий симптом теж нечітка змінна, тільки її носій – відрізок [0;3]. Для ревматизму приймаємо такі значення симптомів: $z_{лади}$=1, $z_{ас}$=1, $z_{тк}$=1, $z_{\text{сколіоз}} \in$[1,5; 3].

Нехай при обстеженні пацієнта Y ми одержали такі значення властивостей $y_{лади}, y_{ас}, y_{тк}, y_{\text{сколіоз}}$. Тоді відстань до прецедента 'Ревматизм' вимірюється як:

$$d_{\text{ревматизм}} = 0{,}7\cdot|1-y_{\text{лади}}|+0{,}65\cdot|1-y_{ac}|+0{,}45\cdot|1-$$
$$-y_{тк}|+0{,}35\cdot|1-\mu_{[1{,}5;3]}(y_{\text{сколіоз}})|.$$

Аналогічним чином можна виміряти відстань до інших захворювань (‘Артрит’, ‘Подагра’). Пацієнт хворий тим захворюванням, відстань до якого є найменшою.

МЕТРИЗАЦІЯ ІНТЕЛЕКТУАЛЬНИХ АГЕНТІВ ПЛАНУВАННЯ ДІЯЛЬНОСТІ

Розглянемо спочатку ознакові задачі. Нехай $v(St(i))$ – оцінка стану $St(i)$. a_{ij}^k – перехід із стану $St(i)$ в стан $St(j)$, використовуючи альтернативу α_k. Наприклад, для зняття захисного покриття з поверхні трубопроводу можна використати три альтернативи: *механічний, хімічний або термічний спосіб зняття захисного покриття*. $v(a_{ij}^k)$ – оцінка дії a_{ij}^k. Стан мети *Goal* визначається необхідністю деякій підмножині ознак X досягнути певних значень $z(x, Goal)\ \forall x \in X$.

Будь-який стан $St(i)$ задається своєю множиною ознак Y_i, які набувають значень $z(y, St(i))\ \ \forall y \in Y_i$.

Для оцінки стану $St(i)$ необхідно здійснити відображення ψ множини ознак та їх значень стану $St(i)$ в множину ознак та значень стану *Goal* за рахунок онтології, тобто

$$\psi : Y_i \xrightarrow{0} X. \qquad (6)$$

Тоді оцінка стану $v(St(i))$ обчислюється

$$v(St(i)) = d(St(i), Goal) = \sum_{x\in X} w_x \varphi(z(x, St(i)), z(x, Goal)), \qquad (7)$$

де w_x – важливість ознаки x в адаптивній онтології, функція φ така ж як у формулі (5).

Вочевидь, що чим оцінка стану менша, тим стан кращий.

У наших дослідженнях для вибору дій ІА ми спиратимемось на раціональність агента як прагнення мінімізувати витрати ресурсів для досягнення кінцевого стану. Тому вважатимемо, що кожна дія a_{ij}^k однозначно визначається витратами ресурсів g_{ij}^k (ціна переходу зі стану в стан), де $k = 1, 2, ..., n_i$. n_i – кількість альтернатив α_k для здійснення переходу a_{ij}. Кожна з альтернатив характеризується витратами ресурсів та терміном експлуатування. Інформація про альтернативи та витрати ресурсів повинна зберігатися в онтології. Інформація про значення ознак та виграш від переходу в стан (терміни експлуатування тощо) зберігається в базі даних. Вочевидь, що можуть появлятися нові альтернативи, тому інтелектуальному агенту необхідно постійно відстежувати наукові видання з метою їх пошуку та заносити в онтологію.

Оцінка дії прямопропорційна витраті ресурсів, тобто:

$$v(a_{ij}^k) = E \cdot g_{ij}^k, \qquad (8)$$

де E – скалярна величина, яка зводить вимір оцінки дії до одного виміру з оцінкою станів.

Загалом рішення стосовно вибору дії на основі альтернативи здійснюємо згідно формули:

$$0(a_{ij}^k) = \omega v(a_{ij}^k) + (1-\omega)v(St(j)), \qquad (9)$$

де $\omega \in [0, 1]$ – частка альтернативі, яку ІА віддає в процесі прийняття рішення, інша частка належить стану в який він перейде. Перехід здійснюємо у той стан для якого $0(a_{ij}^k)$ найменше.

Після оцінки дій та станів, задача вибору шляху зводиться до задачі асинхронного динамічного програмування [10]. Використовуючи методи придатні для розв’язування таких задач, знаходимо розв’язок у вигляді шляху переходу з початкового у кінцевий стан.

Для семантичних задач про стан мети *Goal* наперед щось важко сказати. Наприклад, для задачі реферування текстових документів станом мети є кінцевий реферат, однак ми лише можемо собі уявляти як він приблизно має виглядати. Оцінка стану в такій задачі співпадає з оцінкою важливості семантичної одиниці (слово, лексема, речення), залежно від задачі. Нами запропоновано для таких задач будувати метрику на основі зважування міри TF-IDF онтологією ПО [11]. Тобто

$$v(St) = (TF - IDF)\cdot W. \qquad (10)$$

Така оцінка містить істотні переваги у порівнянні з іншими оцінками, оскільки у ній одночасно враховується як частотний аналіз зустрічання термінів у тексті (TF-IDF), так і специфіка предметної області до якої належить тематика цього тексту. Новий стан для задач реферування полягає в додаванні у квазіреферат нових речень. Детальніше ця задача нами розглянута у [11].

МЕТОДИ ЗАДАННЯ ВАГ ВАЖЛИВОСТІ ПОНЯТЬ ТА ЗВ’ЯЗКІВ

Розглянемо методи задання початкових коефіцієнтів важливості понять та зв’язків та їх модифікацію в процесі функціонування інтелектуального агента для розв’язування ознакових та семантичних задач.

Методи задання початкових ваг (коефіцієнтів важливості) понять та зв’язків:

– за рахунок експертних оцінок;

– присвоєння випадковим чином;

– за рахунок аналізу (статистичного, інтелектуального) інформаційних джерел, які описують ПО в якій функціонує ІА.

Окрім того, ці ваги можуть мати обмеження на величину, наприклад їх значення знаходяться у відрізку [0, 1] (ймовірнісні методи), або без обмежень на величину (нагромаджувальні методи). Оскільки онтологія формує собою таксономію понять, то використовуючи мову

об'єктно-орієнтованого підходу, кожне поняття являє собою клас.

Для семантичних задач нами запропоновано такий метод обчислення ваг класів:

1. Повна вага W_j^i класу онтології дорівнює сумі власної ваги $W0_j^i$, ваги підкласів Ws_j^i та ваги суміжних класів Wn_j^i (класів, зв'язаних з даним класом не IS-A зв'язком):

$$W_j^i = W0_j^i + Ws_j^i + Wn_j^i, \quad (11)$$

де $Ws_j^i = \sum_k Wc_k^{i+1} \cdot L_{j,k}$ – вага k підкласів j-го класу i-го рівня, причому для кореневого класу рівень i = 0; $Wc_k^{i+1} = W0_k^{i+1} + Ws_k^{i+1}$ – вага класу C_k^{i+1}; $L_{j,k}$ – вага зв'язку між класами C_j^i та C_k^{i+1}.

Перерахунок окремих компонент повної ваги класу відображено на схемі (рис. 6).

2. У момент внесення на i+1-й рівень нового підкласу йому присвоюється власна вага $W0_j^{i+1}$, рівна половині власної ваги класу вищого i-го рівня:

$$W0_j^{i+1} = \frac{1}{2} W0_j^i. \quad (12)$$

Вага класу Wc_j^i та усіх батьківських класів аж до кореневого збільшується на величину ваги новоствореного підкласу:

$$Wc_j^m = Wc_j^m + W0_j^{i+1}, \ \forall m \le i. \quad (13)$$

3. Під час встановлення зв'язку між поняттями k_1 та k_2 між відповідними вершинами графа онтології з'являється ребро, а до ваги суміжних класів Wn_1 додається вага Wc_2 і навпаки – до Wn_2 додається вага нового суміжного до нього класу Wc_1, так що:

$$Wn_j = \sum_k Wc_k \cdot L_{j,k}. \quad (14)$$

Повторне встановлення зв'язків приводить до появи кратних ребер у графі.

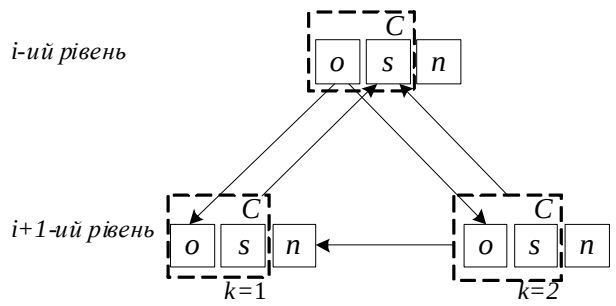


Рис. 6. Схема перерахунку окремих компонент повної ваги класу

4. Кратність ребер відображає частоту зустрічання V пари семантично пов'язаних понять $L_{i+1} = V \cdot L_i$. Кратні ребра після перерахунку не збільшують валентність вершини.

5. Вага екземпляра БЗ дорівнює повній вазі його класу.

Таким чином, визначена модель онтології БЗ дає змогу розраховувати вагові коефіцієнти своїх компонентів у процесі їх додавання, вилучення і використання під час експлуатації системи, завдяки чому реалізує механізм адаптації до заданої користувачем ПО [12].

Зауважимо, що насправді коефіцієнти важливості понять та зв'язків між ними є векторними величинами розмірності кількості прецедентів або різних тематик визначених в конкретній ПО. Так, якщо ми розглядаємо задачу класифікації текстових документів, то в межах онтології має описуватись кілька різних тем. Тому коефіцієнти важливості понять залежать від тематики. Нехай онтологія O описує m тем ПО – $Th_1, Th_2, ..., Th_m$, тоді коефіцієнти ваг понять є вектор, компонентами якого є відповідні значення в залежності від теми, тобто $W = (W_1, W_2, ..., W_m)$. Для процесу автоматизованого реферування наперед треба вибрати тему, до якої належатиме текстовий документ, що опрацьовується, щоб система використовувала правильні ваги [13].

Розглянемо визначення ваг понять для ознакових задач на основі інтелектуального аналізу даних, а саме на основі побудови дерева рішень (ДР). Як відомо ознакові задачі дозволяють для пошуку релевантних прецедентів будувати ДР [14]. Однак, ДР не є панацеєю, оскільки згадувані ознаки, що лежать на відповідній гілці, що задає прецедент, не гарантують врахування повної множини ознак, які необхідно врахувати для знаходження релевантного прецедента. Нами пропонується використовувати ДР для визначення ваг базових термінів, які задають деякий прецедент, а потім на основі онтології ПО розвинути отримані ваги на всю онтологію для відповідного прецедента. Тоді для пошуку релевантного прецедента слід використовувати значення тих n понять, які для відповідного прецедента мають найбільші ваги.

Розглянемо гілку дерева рішень. Вершини (ознаки) цієї гілки знаходяться на k рівнях. Вочевидь, що чим вищий рівень, тим значуща ознака, яка на цьому рівні знаходиться. Ця евристична думка має бути відображена в значеннях ваг цих ознак. Крім того, пропонується ці ваги нормувати, тобто щоб їх сума для кожного прецедента (гілки) була рівна 1.

Розглянемо два способи визначення ваг базових ознак, які задовольняють вищі описані два припущення.

1 спосіб. Арифметичні ваги. Визначаються як відношення різниці (k+1) рівня дерева та рівня, на якому знаходиться ознака до суми всіх рівнів гілки, тобто базуються на сумі арифметичної прогресії:

$$w_i = \frac{k+1-i}{\sum_{j=1}^{k} j} = \frac{k+1-i}{\frac{(1+k)k}{2}}.$$

2 спосіб. Геометричні ваги. Базуються на сумі геометричної прогресії:

$$w_i = \frac{2^{k-i}}{2^k - 1}.$$

Отримані на основі ДР ваги назвемо вагами базових ознак прецедента і позначимо таку множину ваг W_{B}. Тепер необхідно їх розвинути на всю онтологію ПО, використовуючи таксономію понять онтології, відношення між поняттями та їх інтерпретацію. Математично (формально) цей процес запишемо у вигляді:

$$W_{\text{B}} \xrightarrow{0} W. \qquad (15)$$

Розмноження ваг на всю онтологію залежить від визначення (аксіоматизації) класів, їх ієрархії (вертикальний зв'язок) та горизонтальних зв'язків. У роботі [15] типам зв'язків присвоєні такі коефіцієнти важливості: для зв'язку типу спеціалізації («IS-A») – $\sigma = 0{,}9$; для узагальнення («kind-of») – $\gamma = 0{,}4$; для причинного зв'язку («caused-by») – $\rho_{\text{CBY}} = 0{,}3$; для характеризуючого зв'язку («characterized-by») – $\rho_{\text{WRT}} = 0{,}2$. Саме такі значення ми використовували під час розроблення вище описаних прикладних ІСППР.

Отже, загалом процес функціонування ІСППР на основі онтологій складається із кроків, наведених на рис. 7. Для його зображення використано діаграму діяльності (UML Activity) [16].

Для розроблення ІСППР було обрано такі програмні засоби (ПЗ):

– Для побудови онтології – редактор онтологій Protégé ;

– Для запису правил бази знань – SWRL, який входить в Protégé ;

– База даних – СКБД MySQL;

– Модуль керування процесом розв'язування задачі – мова програмування PHP.

В залежності від типу задачі в базі даних зберігаються: перелік прецедентів та відповідних до них рішень, ваги важливості понять адаптивної онтології, типи відношень та їх ваги, значення ознак, історія значень ознак (для задач планування).

ВИСНОВКИ

Розроблено математичні моделі функціонування інтелектуальних систем підтримки прийняття рішень в залежності від класу задачі. Всі ці моделі використовують метрику для знаходження релевантних прецедентів або визначення релевантності станів. Для побудови таких метрик використовуються онтології. З цією метою у загальноприйнятий трьохелементний кортеж, який задає онтологію (множина понять, відношень та їх інтерпретація), авторами додано дві скалярні величини (важливість понять та відношень), які використовуються для обчислення необхідних відстаней. Розглянуто способи завдання початкових коефіцієнтів важливості понять та зв'язків, зокрема на основі інтелектуального аналізу даних.

Розглянуті чотири класи задач. Визначено перелік задач, які входять до цих класів. Наведено приклади функціонування окремих інтелектуальних систем, які базуються на розроблених моделях.

СПИСОК ЛІТЕРАТУРИ

1. *Каменнова, М. С.* Корпоративные информационные системы: технологии и решения / М.С.Каменнова // Системы Управления Базами Данных. – 1995. – № 3. – С. 88–99.

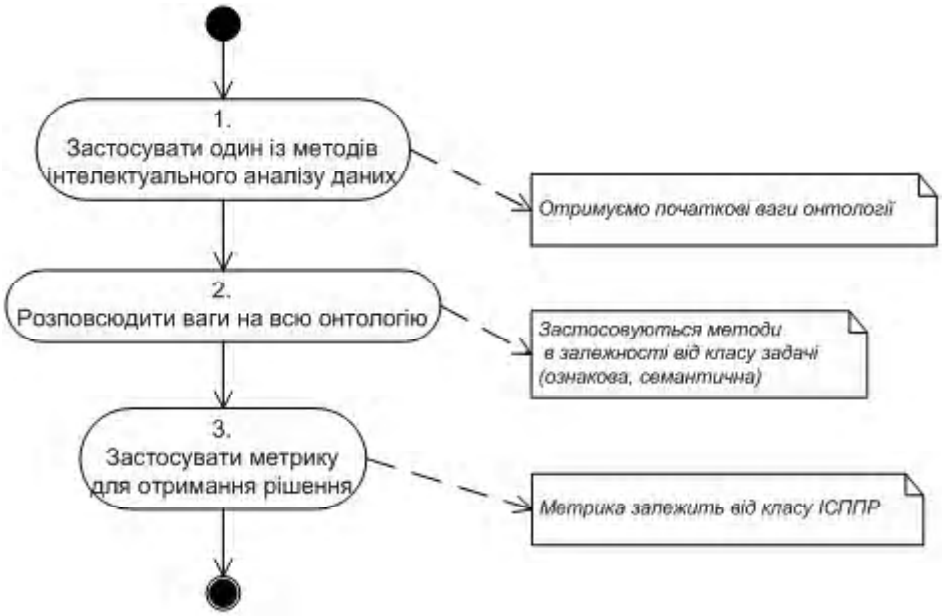


Рис. 7. Діаграма діяльності ІСППР на основі онтологій

2. *Funk, P.* Advances in Case-Based Reasoning / P. Funr, P. A. Gonzбlez-Calero // 7th European Conference, ECCBR 2004. – Madrid, Spain. – P. 375–380.
3. *Рассел, С.* Искусственный интеллект / С. Рассел, П. Норвиг. – М. ; СПб. ; К. : Вильямс, 2006. – 1408 с.
4. *Gruber, T. R.* A translation approach to portable ontologies / T.R.Gruber // Knowledge Acquisition. – 1993. – N 5 (2). – P. 199–220.
5. Інтелектуальні системи, базовані на онтологіях // Д. Г. Досин, В. В. Литвин, Ю. В. Нікольський, В. В. Пасічник. – Львів : Цивілізація, 2009. – 414 с.
6. *Литвин, В. В.* Мультиагентні системи підтримки прийняття рішень, що базуються на прецедентах та використовують адаптивні онтології / В. В. Литвин // Радіоелектроніка, інформатика, управління. – 2009 . – № 2 (21). – С. 120–126.
7. *Sowa, J.* Conceptual graphs for a database interface / J.Sowa // IBM Journal of Research and Development. – Vol. 20. – № 4. – 1976. – P. 336–357.
8. *Свами, М.* Графы, сети и алгоритмы / М. Свами, К. Тхуласираман. – М. : Наука, 1984. – 408 с.
9. *Литвин, В. В.* Проектування інтелектуальних агентів прийняття рішень в просторі ознак з використанням онтологічного підходу / В. В. Литвин, Р. Р. Даревич, Д. Г. Досин, Н. В. Шкутяк // Штучний інтелект. – 2010. – № 3. – С. 254–262.
10. *Скобелев, В. Г.* Локальные алгоритмы на графах / В. Г.Скобелев. – Донецк : ИПММ НАН Украины, 2003. – 217 с.
11. *Крайовський, В. Я.* Основні підходи до розроблення программного комплексу автоматичного реферування текстових документів / В. Я. Крайовський В. В. Литвин, Н. Б. Шаховська // Інститут проблем моделювання в енергетиці. –2009. – Випуск 51. – С. 178–186.
12. *Даревич, Р. Р.* Оцінка подібності текстових документів на основі виз-начення інформаційної ваги елементів бази знань / Р. Р. Даревич, Д. Г. Досин, В. В. Литвин, З. Т. Назарчук // Штучний інтелект. – 2006. – № 3. – С. 500–509.
13. *Литвин, В. В.* Метод автоматизованого реферування текстових документів з використанням онтологій / В. В. Литвин, В. А. Гайдін, О. Ю. Пшеничний // Складні системи і процеси. – Запоріжжя. – 2009. – № 1. – С. 81–87.
14. *Цвєтков, А. М.* Разработка алгоритмов индуктивного вывода с использованием деревьев решений / А. М. Цветков // Кибернетика и системный анализ. – № 1. – 1993. – С. 174–178.
15. *Bulskov, H.* On Querying Ontologies and Databases / H. Bulskov, R. Knappe, R. Andreasen // FQAS. – 2004. – P. 191–202.
16. *Фаулер, М.* UML в кратком изложении / М. Фаулер, К. Скотт. – М. : Мир, 1999. –340 с.

Стаття надійшла до редакції 02.12.2010.

Лытвын В. В.

МОДЕЛИРОВАНИЕ ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМ ПОДДЕРЖКИ ПРИНЯТИЯ РЕШЕНИЙ С ИСПОЛЬЗОВАНИЕМ ОНТОЛОГИЧЕСКОГО ПОДХОДА

Исследовано функционирование интеллектуальных систем поддержки принятия решений, ядром баз знаний которых являются онтологии. Рассматриваются интеллектуальные агенты четырех типов. Для каждого типа разработана метрика, которая используется для определения релевантности предлагаемых системой решений.

Ключевые слова: интеллектуальная система поддержки принятия решений, интеллектуальный агент.

Lytvyn V. V.

INTELLIGENT DECISION SUPPORT SYSTEMS MODELING USING ONTOLOGICAL APPROACH

The functioning of intelligent decision support systems with knowledge base based on ontology is investigated. Four types of intelligent agents are considered. Metrics for each of them is developed and used to determine the relevance of solutions proposed by the system.

Keywords: Intelligent Decision Support System, Intelligent Agent.

УДК 004.3

Баркалов А. А.[1], Зеленева И. Я.[2], Цололо С. А.[3], Биайрак Х.[4]

[1] Д-р техн. наук, профессор Донецкого национального технического университета
[2,3] Канд. техн. наук, доцент Донецкого национального технического университета
[4] Аспирант Донецкого национального технического университета

УМЕНЬШЕНИЕ ПЛОЩАДИ МАТРИЧНОЙ СХЕМЫ УСТРОЙСТВА УПРАВЛЕНИЯ С РАЗДЕЛЕНИЕМ КОДОВ

В статье предложена модель композиционного микропрограммного устройства управления с разделением кодов, ориентированная на реализацию логической схемы устройства в базисе заказных матриц. В модели используется представление адреса вершины алгоритма управления в виде конкатенации кодов ОЛЦ и кода компоненты ОЛЦ. Этот подход позволяет уменьшить число входов и выходов схемы формирования функций возбуждения.

Ключевые слова: композиционное устройство управления, матричная схема, операторная линейная цепь (олц), разделение кодов.

ВВЕДЕНИЕ

При реализации схемы устройства управления (УУ) необходимо учитывать особенности элементного базиса и алгоритмы управления [1]. Если алгоритм управления представлен линейной граф-схемой алгоритма (ГСА), то для его реализации целесообразно использо-

вать модель композиционного микропрограммного устройства управления (КМУУ) [2]. В настоящей работе рассматривается задача реализации схемы КМУУ в базисе заказных матричных схем [3], которые широко используются при массовом производстве средств вычислительной техники и автоматики. В этом случае возникает задача уменьшения площади кристалла, занимаемого схемой КМУУ [4; 5]. Решение этой задачи позволяет уменьшить потребляемую мощность и повысить выпуск годных кристаллов. При этом рассматривается случай выполнения условий, позволяющих использовать модель КМУУ с разделением кодов [2].

Целью исследования является уменьшение аппаратурных затрат в схеме КМУУ с разделением кодов за счет использования нескольких источников классов псевдоэквивалентных ОЛЦ.

Задачей исследования является разработка метода, позволяющего уменьшить площадь, занимаемую схемой адресации микрокоманд в КМУУ с разделением кодов.

РЕАЛИЗАЦИЯ КМУУ С РАЗДЕЛЕНИЕМ КОДОВ НА ЗАКАЗНЫХ МАТРИЦАХ

Пусть ГСА Г представлена множествами вершин B и дуг E, соединяющих эти вершины. При этом $B = \{b_0, b_E\} \cup E_1 \cup E_2$, где b_0 – начальная вершина ГСА, b_E – конечная вершина ГСА, E_1 – множество операторных вершин, где $|E_1| = M$, E_2 – множество условных вершин. В вершинах $b_q \in E_1$ записаны наборы микроопераций $Y(b_q) \subseteq Y$, где $Y = \{y_1, ..., y_N\}$ – множество микроопераций. В вершинах $b_q \in E_2$ записаны элементы множества логических условий $X = \{x_1, ..., x_L\}$. Пусть ГСА является линейной, то есть включает более 75 % операторных вершин [3].

Сформируем множество ОЛЦ $C = \{\alpha_1, ..., \alpha_G\}$ ГСА Г, где каждая из ОЛЦ является последовательностью операторных вершин и каждой паре ее соседних компонент b_i, b_j соответствует дуга $\langle b_i, b_j \rangle \in E$. Каждая ОЛЦ имеет только один выход O_g и произвольное число входов I_g ($g = 1, ..., G$). Формальные определения ОЛЦ, их входов и выходов можно найти в [3]. Каждая вершина $b_q \in E_1$ соответствует микрокоманде MI_q, хранимой в управляющей памяти (УП) КМУУ по адресу $A(b_q)$. Для адресации микрокоманд достаточно

$$R = \lceil \log_2 M \rceil \quad (1)$$

бит, представленных переменными $T_r \in T$, где $|T| = R$. Пусть ОЛЦ $\alpha_g \in C$ включает F_g компонент и пусть $Q = \max(F_1, ..., F_G)$. В этом случае для кодирования компонент достаточно

$$R_C = \lceil \log_2 Q \rceil \quad (2)$$

переменных, образующих множество τ, где $|\tau| = R_C$. Для кодирования ОЛЦ $\alpha_g \in C$ достаточно

$$R_C = \lceil \log_2 G \rceil \quad (3)$$

переменных, образующих множество T, где $|T| = R_0$.

Пусть $K(\alpha_g)$, $K(b_q)$ соответственно код ОЛЦ $\alpha_g \in C$ и код компоненты некоторой ОЛЦ. Тогда адрес микрокоманды, соответствующей вершине $b_q \in E_1$, может быть представлен в виде конкатенации

$$A(b_q) = K(\alpha_g) * K(b_q). \quad (4)$$

В выражении (4) b_q является компонентой ОЛЦ $\alpha_g \in C$, а операция конкатенации обозначается знаком *. Выполним адресацию компонент так, чтобы для каждой ОЛЦ $\alpha_g \in C$ их коды возрастали в естественном порядке. При этом первая компонента любой ОЛЦ имеет код 0, вторая – 1, и так далее до Q.

В этом случае для реализации схемы УУ может быть использована модель КМУУ с разделением кодов (рис. 1).

Назовем эту модель КМУУ U_1. В КМУУ U_1 матрицы M_1 и M_2 образуют блок адресации микрокоманд (БАМ), а матрицы M_3 и M_4 – управляющую память (УП). Блок БАМ реализует системы функций возбуждения триггеров счетчика СТ и регистра RG :

$$\Phi = \Phi(X, \tau), \quad (5)$$

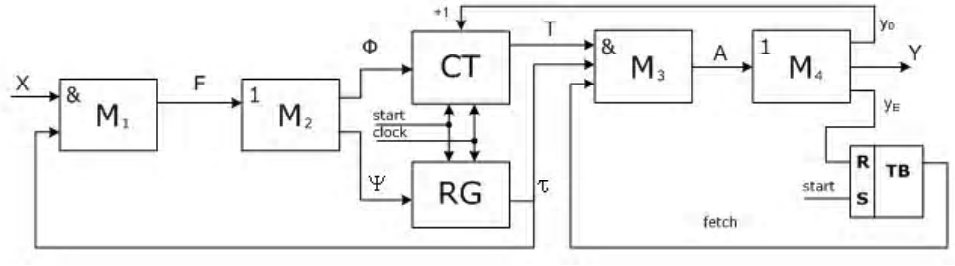


Рис. 1. Матричная реализация КМУУ с разделением кодов

$$\Psi = \Psi(X, \tau). \quad (6)$$

При этом матрица M_1 реализует систему термов $F = \{F_1, ..., F_H\}$, входящих в функции (5)–(6). Матрица M_2 реализует функции (5)–(6), как дизъюнкции термов $F_h \in F$:

$$\left.\begin{aligned} \varphi_r &= \bigvee_{h=1}^{H} C_{rh} F_h (r = \overline{1, R_0}); \\ \psi_r &= \bigvee_{h=1}^{H} C_{rh} F_h (r = \overline{1, R_C}). \end{aligned}\right\} \quad (7)$$

В функциях (7) $C_{rh} \in \{0,1\}$ и $C_{rh} = 1$, если и только если терм F_h входит в функцию φ_r или ψ_r.

Матрица M_3 реализует систему функций A, соответствующих адресам микрокоманд (4):

$$A(b_q) = \left(\bigwedge_{r=1}^{R_0} \tau_r^{l_{gr}} \right) \cdot \left(\bigwedge_{r=1}^{R_C} T_r^{l_{qr}} \right). \quad (8)$$

В формуле (8) $l_{gr} \in \{0,1\}$ – значение r-го разряда кода $K(\alpha_g)$, где $\alpha_g \in C$ и b_q входит в эту ОЛЦ; $l_{qr} \in \{0,1\}$ – значение r-го разряда кода $K(b_q)$; $\tau_r^0 = \overline{\tau_r}$, $\tau_r^1 = \tau_r (r = \overline{1, R_0})$, $\tau_r^0 = \overline{\tau_r}$, $\tau_r^1 = \tau_r (r = \overline{1, R_C})$. Отметим, что некоторые разряды этих кодов могут быть несущественными, однако этот случай в данном случае не рассматривается. Для формирования терма $A_m \in A(m = \overline{1, M})$ соответствующий терм $A(b_q)$ умножается на переменную Fetch. При Fetch=1 разрешается выборка микрокоманд из УП.

Матрица M_4 реализует систему функций

$$y_n = \bigvee_{m=1}^{M} C_{nm} A_m, \quad (9)$$

где $n \in \{0,1,..., N, E\}$, а $C_{nm} = 1$, если и только если функция y_n формируется в микрокоманде с индексом m. Переменная y_0 служит для увеличения содержимого СТ на единицу, что позволяет адресовать компонент одной ОЛЦ без использования блока БАМ. Переменная y_E формируется при достижении конечной вершины ГСА Γ и служит признаком окончания алгоритма. Если $y_E = 1$, то триггер TF обнуляется, переменная Fetch=1 и выборка микрокоманд прекращается.

Как видно из рис. 1, сигнал Start служит для установки начальных значений (Fetch=1, CT=RG=0). Сигнал Clock используется для переключения элементов памяти (СТ и RG).

Очевидно, КМУУ U_1 является автоматом Мура, состояния которого представлены функциями $\tau_r \in \tau$. Недостатком этого устройства является значительное число термов $F_h \in F$, реализуемых матрицей M_1. Для уменьшения этого параметра можно ввести преобразователь кодов (ПК) [6]. Этот подход позволяет уменьшить число термов в системе (5)–(6) до некоторой величины H_0, определяемой характеристиками эквивалентного автомата Мили. Однако ПК потребляет некоторые ресурсы кристалла. В настоящей статье предлагается подход, позволяющий гарантировано уменьшить параметр H до H_0 и уменьшить площадь, занимаемую схемой ПК.

ОСНОВНАЯ ИДЕЯ ПРЕДЛАГАЕМОГО МЕТОДА

Напомним, что ОЛЦ $\alpha_i, \alpha_j \in C$ являются псевдоэквивалентными (ПОЛЦ), если их выходы связаны с входом одной и той же вершиной ГСА Γ [2]. Пусть $\Pi_C = \{B_1, ..., B_I\}$ разбиение множества ОЛЦ C_1 на классы ПОЛЦ. При этом ОЛЦ $\alpha_g \in C_1$, если ее выход не связан с вершиной b_E. Выполним кодирование ОЛЦ $\alpha_g \in C$ так, чтобы максимально возможное число классов $B_i \in \Pi_C$ входило в один обобщенный интервал R_0-мерного булева пространства.

Теперь множество Π_C может быть представлено в виде $\Pi_C^1 \cup \Pi_C^2$. Пусть $B_i \in \Pi_C^1$, если класс $B_i \in \Pi_C$ представляется одним обобщенным интервалом R_0-мерного булева пространства. В противном случае $B_i \in \Pi_C^2$. Очевидно, $\Pi_C^1 \cap \Pi_C^2 = \varnothing$ и $\Pi_C^1 \cup \Pi_C^2 = \Pi_C$. Закодируем классы $B_i \in \Pi_C^2$ двоичными кодами $K(B_i)$ разрядности

$$R_2 = \lceil \log_2 G_2 \rceil, \quad (10)$$

где $G_2 = \left| \Pi_C^2 \right| + 1$. Используем для кодирования классов $B_i \in \Pi_C^2$ переменные $z_r \in Z$, где $|Z| = R_2$.

Исходная ГСА Γ служит для нахождения системы обобщенных формул переходов [2]. Разделим эту систему S на две подсистемы: $S = S_1 \cup S_2$. Пусть подсистема S_1 задает переходы для классов $B_i \in \Pi_C^1$, а подсистема S_2 – для классов $B_i \in \Pi_C^2$. В этом случае для реализации схемы КМУУ на заказных матрицах предлагается модель U_2 (рис. 2).

В КМУУ U_2 блок БАМ представлен матрицами M_1^1, M_1^2 и M_2. Матрица M_1^1 реализует термы $F_h \in F^1$, входящие в подсистему формул перехода S_1. Термы $F_h \in F^1$ задаются формулами:

$$F_h = \left(\bigwedge_{r=1}^{R_0} \tau_r^{l_{gr}} \right) \cdot \mathrm{X}_h \ \left(h = \overline{1, H_1} \right). \quad (11)$$

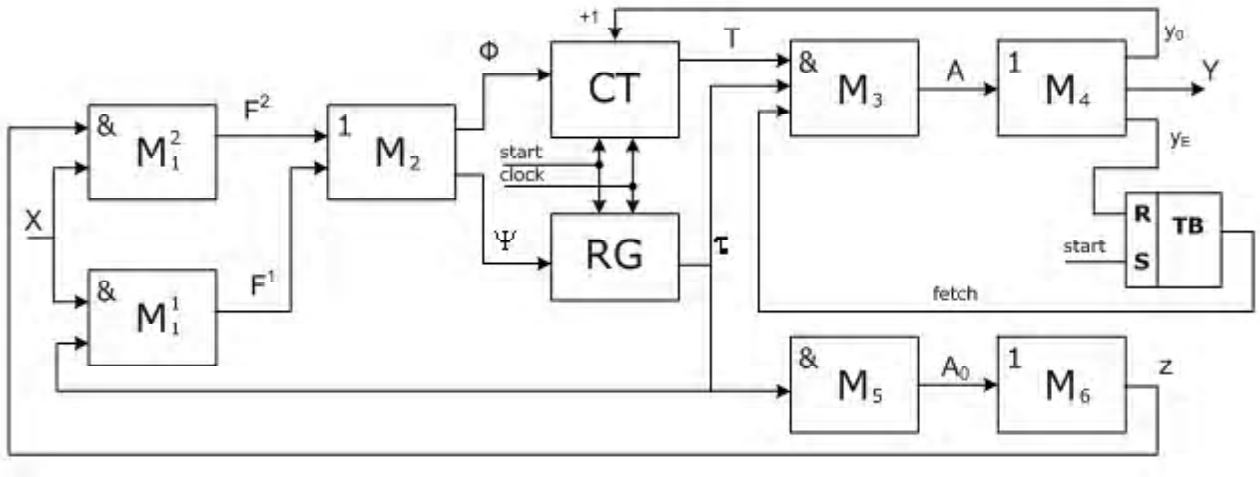


Рис. 2. Структурная схема КМУУ

В формуле (11) конъюнкция X_h соответствует части формул переходов, содержащей логические условия $x_l \in X$. Отметим, что $l_{gr} \in \{0,1,*\}$ и τ_r^*. Это связано с тем, что классы $B_i \in \Pi_C^1$ представляются обобщенными интервалами, то есть некоторые кодирующие переменные имеют неопределенные значения.

Матрица M_1^2 реализует термы $F_h \in F^2$, входящие в подсистему формул S_2. Термы $F_h \in F^2$ задаются формулами:

$$F_h = \left(\bigwedge_{r=1}^{R_2} z_r^{\,l_{ir}} \right) \cdot X_h \ \left(h = \overline{1, H_2} \right). \qquad (12)$$

В формуле (12) $l_{ir} \in \{0,1,*\}$ – значение r-го разряда кода $K(B_i)$ класса $B_i \in \Pi_C^2$. При этом неопределенное значение * может появиться, если число возможных кодов больше числа классов $B_i \in \Pi_C^1$. Очевидно, $z_r^0 = \overline{z_r}$, $z_r^1 = z_r$ и $z_r^* = 1 \ \left(r = \overline{1, R_2} \right)$. Кроме того, выполняется равенство $H_0 = H_1 + H_2$.

Матрица M_2 реализует функции (7), однако теперь они зависят от термов $F_h \in F^1 \cup F^2$. Матрицы M_3 и M_4 реализуют управляющую память; они полностью идентичны соответствующим матрицам КМУУ U_1. Это же справедливо и для триггера TF. Матрицы M_5 и M_6 реализуют преобразователь кодов ПК. Матрица M_5 реализует термы $A_i \in A_0$, входящие в систему функций

$$Z = Z(\tau). \qquad (13)$$

Термы A_i определяются выражением

$$A_i = \bigwedge_{r=1}^{R_0} \tau_r^{\,l_{ir}} \ \left(i = \overline{1, M_0} \right), \qquad (14)$$

где $l_{ir} \in \{0,1,*\}$ – значение r-го разряда кода класса $B_i \in \Pi_C^2$, $\tau_r^0 = \overline{\tau_r}$, $\tau_r^1 = \tau_r$ и $\tau_r^* = 1 \ \left(r = \overline{1, R_0} \right)$. Параметр M_0 определяют в ходе синтеза КМУУ U_2; он зависит от результат кодирования ОЛЦ $\alpha_g \in C$. Матрица M_6 формирует функции (13), как некоторые дизъюнкции от термов (14):

$$z_r = \bigvee_{i=1}^{M_0} C_{ri} A_i \ \left(r = \overline{1, R_0} \right). \qquad (15)$$

В формуле (15) булева переменная $C_{ri} = 1$, если и только если функция z_r зависит от терма A_i.

Принцип функционирования КМУУ U_2 очевиден. Разница между КМУУ U_1 и U_2 заключается в следующем:

1. Используются два источника кодов классов псевдоэквивалентных ОЛЦ. Один источник – регистр RG, второй – блок преобразователя кодов ОЛЦ в коды классов ПОЛЦ.

2. Число термов в схеме БАМ гарантировано равняется этому параметру эквивалентного автомата Мили. При этом площадь, занимаемая блоком ПК, уменьшается по сравнению с подходом, основанным на преобразовании кодов всех ОЛЦ $\alpha_g \in C_1$.

Отметим, что КМУУ с разделением кодов реализуется только при выполнении условия

$$R_0 + R_C = R. \qquad (16)$$

МЕТОД СИНТЕЗА И РАЗВИТИЕ ОСНОВНОЙ ИДЕИ

В настоящей работе предлагается метод синтеза КМУУ U_2, включающий следующие этапы:

1. Формирование множества ОЛЦ C и C_1 по исходной граф-схеме алгоритма Γ.

2. Определение параметров R, R_0 и R_C. Если условие (16) выполняется, то процесс синтеза продолжается.

3. Формирование разбиения Π_C множества ОЛЦ C_1 на классы псевдоэквивалентных ОЛЦ.

4. Кодирование ОЛЦ $\alpha_g \in C$ и их компонент.

5. Формирование содержимого управляющей памяти.

6. Формирование множеств Π_C^1 и Π_C^2. Кодирование классов $B_i \in \Pi_C^2$.

7. Формирование системы обобщенных формул переходов S и разбиение ее на подсистемы S_1 и S_2.

8. Формирование таблицы переходов для классов $B_i \in \Pi_C^1$ по системе S_1. Построение системы функций $\Phi = \Phi(\tau, X)$ и системы $\Psi = \Psi(\tau, X)$.

9. Формирование таблицы переходов для классов $B_i \in \Pi_C^2$ по системе S_2. Построение системы функций $\Phi = \Phi(Z, X)$ и системы $\Psi = \Psi(Z, X)$.

10. Построение таблицы преобразователя кодов и формирование системы функций $Z = Z(\tau)$.

11. Реализация схемы КМУУ на заказных матрицах по полученным системам функций.

Детализируем некоторые этапы синтеза, связанные с построением таблиц переходов. Пусть система S_1 включает формулу перехода:

$$B_3 \to x_1 x_2 b_3 \vee x_1 \overline{x_2} b_8 \vee \overline{x_1} b_{12}. \qquad (17)$$

Пусть $K(B_3) = 011$, $A(B_3) = 10010$, $A(B_8) = 11001$ и $A(B_{12}) = 11100$. Очевидно, $R_0 = 2$, $T = \{T_1, T_2\}$, $R_C = 3$, $\tau = \{\tau_1, \tau_2, \tau_3\}$. Таблица переходов для классов $B_i \in \Pi_C^1$ имеет следующие столбцы: B_i – класс ПОЛЦ; $K(B_i)$ – код класса $B_i \in \Pi_C^1$; $A(b_q)$ – адрес входа некоторой ОЛЦ $\alpha_g \in C$; X_h – входной сигнал, определяющий переход из входов ОЛЦ $\alpha_g \in B_i$ в вершину b_q; Ψ_h – функция возбуждения триггеров регистра RG, принимающие единичное значение на переходе $\langle B_i, b_q \rangle$; Φ_h – функция возбуждения триггеров счетчика СТ, принимающие единичное значение на переходе $\langle B_i, b_q \rangle$; $h = \overline{1, H_1}$ – номер перехода. Для формулы (17) этот фрагмент таблицы имеет следующий вид (табл. 1).

Как следует из табл. 1, $\Psi = \{D_1, D_2, D_3\}$ и $\Phi = \{D_4, D_5\}$. Из табл. 1 имеем, например, следующие функции:

$$\begin{aligned} D_2 &= \overline{\tau_1}\tau_2\tau_3 x_1 \overline{x_2} \vee \overline{\tau_1}\tau_2\tau_3 \overline{x_1}; \\ D_4 &= \overline{\tau_1}\tau_2\tau_3 x_1 x_2. \end{aligned} \qquad (18)$$

Термы системы (18) реализуются на матрице M_1^1, а сами функции – на матрице M_2.

Пусть система S_2 включает формулу перехода

$$B_5 \to x_2 b_{19} \vee \overline{x_3} b_{26}. \qquad (19)$$

Пусть $K(B_5) = 01$, $A(B_{19}) = 00101$, $A(B_{26}) = 01100$, то есть $R_2 = 2$, $Z = \{z_1, z_2\}$. Таблица переходов для системы S_2 имеет такой же вид, как и таблица переходов для системы S_1 (табл. 2).

Из табл. 2 можно, например, получить функции

$$D_2 = \overline{z_1} z_2 \overline{x_3}; \; D_5 = \overline{z_1} z_2 x_3. \qquad (20)$$

Термы системы (20) реализуются на матрице M_1^2, а сами функции – на матрице M_2.

Пусть $\Pi_C^2 = \{B_2, B_5, B_6\}$, $K(B_2) = 01$, $K(B_5) = 10$ и $K(B_6) = 11$. При этом код 00 является признаком того, что переходы происходят из классов $B_i \in \Pi_C^1$. Этому факту соответствует переменная y_{C1}; в данном случае $y_{C1} = \overline{z_1 z_2}$. Пусть $B_2 = \{\alpha_2, \alpha_3\}$, $B_5 = \{\alpha_4, \alpha_5\}$ и $B_6 = \{\alpha_6, \alpha_7\}$, а ОЛЦ имеют коды, соответствующие двоичным эквивалентам их индексов. Таблица преобразователя кодов имеет следующие столбцы: α_g, $K(\alpha_g)$, B_i, $K(B_i)$, Z_i, i. Здесь столбец Z_i содержит перемен-

Таблица 1. Фрагмент таблицы переходов для формулы (17)

B_i	$K(B_i)$	$A(b_q)$	X_h	Ψ_h	Φ_h	h
B_3	011	10010	$x_1 x_2$	D_1	D_4	1
		11001	$x_1 \overline{x_2}$	$D_1 D_2$	D_5	2
		11100	$\overline{x_1}$	$D_1 D_2 D_3$	–	3

Таблица 2. Фрагмент таблицы переходов для формулы (19)

B_i	$K(B_i)$	$A(b_q)$	X_h	Ψ_h	Φ_h	h
B_5	01	00101	x_3	D_3	D_5	1
		01199	$\overline{x_3}$	$D_2 D_3$	–	2

ные z_r, равные единице в коде $K(B_i)$. Для нашего примера таблица ПК представлена в табл. 3.

Из табл. 3 с учетом минимизации имеем: $z_1 = \tau_1$, $z_2 = \tau_2$. При этом матрица M_5 в схеме КМУУ отсутствует. Однако, это частный случай. В общем случае система термов, полученных из таблицы ПК, реализуется на матрице M_5, а функции $z_r \in Z$ – на матрице M_6.

Отметим, что термы $F_h \in F^1$ могут зависеть только от логических условий $x_l \in X^1$, где $X^1 \subset X$. Аналогично, термы $F_h \in F^2$ могут зависеть только от логических условий $x_l \in X^2$, где $X^2 \subset X$. В лучшем случае имеем равенство $X^1 \cap X^2 = \varnothing$. Это позволяет уточнить часть схемы КМУУ U_2, реализующей функции Ψ и Φ (рис. 3).

Отметим, что наличие переменной y_{C1} не вносит задержку во время такта КМУУ U_2 по сравнению с КМУУ U_1. Это связано с тем, что значение переменной y_{C1} формируется до прихода правильных значений логических условий. Итак, площади матриц M_1^1, M_1^2 и M_2 могут быть найдены следующим образом:

$$\begin{aligned} S\left(M_1^1\right) &= 2\left(L_1 + R_C + 1\right)\cdot H_1; \\ S\left(M_1^2\right) &= 2\left(L_2 + R_0\right)\cdot\left(H_2 + 1\right); \\ S\left(M_2\right) &= H_0\left(R_0 + R_C\right). \end{aligned} \qquad (21)$$

Таблица 3. Таблица ПК КМУУ U_2

α_g	$K(\alpha_g)$	B_i	$K(B_i)$	Z_i	i
α_2	010	B_2	01	z_2	1
α_3	011	B_2	01	z_2	2
α_4	100	B_5	10	z_1	3
α_5	101	B_5	10	z_1	4
α_6	110	B_6	11	$z_1 z_2$	5
α_7	111	B_6	11	$z_1 z_2$	6

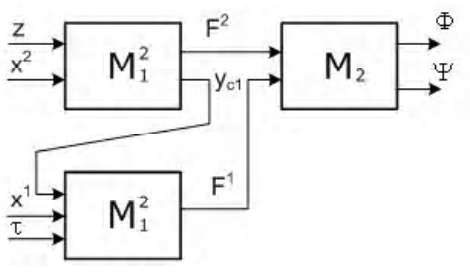


Рис. 3. Уточненная часть схемы КМУУ U_2

В системе (21) имеем $L_1 = \left|X^1\right|$, $L_2 = \left|X^2\right|$, $\left|F^1\right| = H_1$, $\left|F^2\right| = H_2$, $H_1 + H_2 = H_0$.

ЗАКЛЮЧЕНИЕ

В работе предложен метод, позволяющий уменьшить сложность матричной реализации КМУУ с разделением кодов. Этот метод основан на использовании двух источников кодов классов псевдоэквивалентных ОЛЦ. При этом матрица термов схемы адресации КМУУ разбивается на две части.

Такой подход гарантирует уменьшение числа термов в схеме адресации до величины, определяемой эквивалентным автоматом Мили. Выбор между КМУУ U_1 и U_2 может быть выполнен аналитическим путем, то есть без реализации схемы устройства. Отметим, что предложенный метод применим только для линейных граф-схем алгоритма при выполнении условия (16). Если условие (16) не выполняется, то это приводит к резкому увеличению площадей матриц M_3 и M_4 по сравнению с КМУУ с общей памятью [2].

Научная новизна статьи заключается в усовершенствовании метода синтеза КМУУ с разделением кодов. Внесенные изменения позволяют использовать несколько источников кодов псевдоэквивалентных ОЛЦ, что приводит к уменьшению аппаратурных затрат в схеме адресации.

Практическая значимость результатов работы заключается в уменьшении числа базовых элементов, которые необходимы для реализации схемы КМУУ на кристалле. Это позволяет удешевить реализацию сложных цифровых систем, в состав которых входят устройства управление, реализованный по модели КМУУ.

СПИСОК ЛИТЕРАТУРЫ

1. Baranov Samary. Logic Synthesis for Control Automata. / Samary Baranov.– Kluwer Academic Publishers, 1994. – 312 p.
2. *Barkalov, A.* Logic Synthesis for Compositional Microprogram Control Units. / A. Barkalov, L. Titarenko. – Berlin : Springer, 2008. – 273 p.
3. *Smith, Michael.* Application-Specific Integrated Circuits / Michael Smith. – Boston : Addison-Wesley, 1997. – 1040 p.
4. *Баркалов, А. А.* Синтез микропрограммных автоматов на заказных и программируемых СБИС. / А. А. Баркалов, Л. А. Титаренко. – Донецк : УНИТЕХ, 2009. – 336 с.
5. Bayrek Hathot Approach to Realization of Compositional Microprogramming Control Unit with Code Converter on Custom-Made Matrix. / A. Barkalov, I. Zelenyova, A. Miroshkin, H. Bayrek // Наукові праці Донецького національного технічного університету. Серія «Інформатика, кібернетика та обчислювальна техніка». Випуск 11(164). – Донецьк : ДонНТУ, 2010. – С. 71–74.
6. *Barkalov, A.* Basic principles of logic design. / A. Barkalov, L. Titarenko. – Zielona Gora : University of Zielona Gora Press, 2010. – 295 p.

Стаття надійшла до редакції 03.05.2011.

Баркалов О. О., Зеленьова І. Я., Цололо С. О., Біайрак Х.

ЗМЕНШЕННЯ ПЛОЩІ МАТРИЧНОЇ СХЕМИ ПРИСТРОЮ КЕРУВАННЯ З РОЗДІЛЕННЯМ КОДІВ

У статті запропонована модель композиційного мікропрограмного пристрою керування з розділенням кодів, яка орієнтована на реалізацію схеми пристрою у базисі замовних матриць. Запропонована модель використовує представлення адреси вершини алгоритму керування у вигляді конкатенації кодів ОЛЛ та коду компоненти ОЛЛ. Такий підхід зменшує число входів і виходів схеми формування функцій збудження.

Ключові слова: композиційний пристрій керування, матрична схема, операційний лінійний ланцюг (олл) , розділення кодів.

Barkalov A. A., Zelenyova I. J., Tsololo S. A., Biayarek H.

REDUCTION OF TERM MATRIX OF CONTROL UNIT WITH CODE SHARING

The structures of compositional microprogram control unit with code sharing are proposed. Structures allow reducing the complexity of the matrix realization in the device's circuit. The proposed method is based on using of node's address representation as a concatenation of OLC codes and code of OLC components. The proposed method allow reducing the complexity of the matrix realization in the device's circuit.

Key words: compositional control unit, matrix circuit, operator linear chain (olc), code shareing.

УДК 378.14:004.421

Пищухина О. А.[1], Клочок А. Ю.[2]

[1]Канд. техн. наук, доцент Национального аерокосмического университета им. М. Е. Жуковського «ХАИ»
[2]Асистент Национального аерокосмического университета им. М. Е. Жуковського «ХАИ»

ПОДХОД К ФОРМИРОВАНИЮ ОБРАТНОЙ СВЯЗИ В ИНТЕЛЛЕКТУАЛЬНЫХ ОБУЧАЮЩИХ СИСТЕМАХ В СФЕРЕ ВЫСШЕГО ТЕХНИЧЕСКОГО ОБРАЗОВАНИЯ

Предложен подход к формированию обратной связи процесса обучения с использованием интеллектуальных компьютерных систем, особенностью которого является поэтапный контроль процесса обучения, определение места возникновения ошибок, в отдельных случаях – распознавание причин их возникновения и формирование рекомендаций по их устранению.

Ключевые слова: интеллектуальные компьютерные обучающие системы, высшее техническое образование, обратная связь.

ВВЕДЕНИЕ

Сокращение часов аудиторной нагрузки и увеличение объемов самостоятельной работы студентов представляет собой одну из особенностей процесса обучения в современном вузе, ориентированном на западные стандарты образования. Основную сложность для студентов технических специальностей вызывает усвоение постоянно растущего объема информации и расширения круга задач, умение решать которые необходимо для обеспечения компетентности и конкурентоспособности выпускников технических вузов на современном рынке труда. Вместе с тем, следует учитывать ментальные отличия украинских студентов, связанные со слабой мотивацией и (или) неумением работать самостоятельно, что способствует недостаточному усвоению учебного материала, и как следствие, снижению общего образовательного уровня. Необходимо отметить также недостаточную подготовку учащихся средних школ по базовым техническим дисциплинам для эффективного обучения в высших учебных заведениях.

Эффективным путем разрешения данного противоречия, связанного с необходимостью активного обучения студентов, ограничением энергетических ресурсов преподавателя, фиксированным количеством часов практических и лабораторных занятий согласно разработанным учебным и рабочим программам вузов, является использование компьютерных средств для организации самостоятельной работы студентов, а именно – интеллектуальных обучающих систем [1, 2].

Характерной чертой обучения студентов технических специальностей является необходимость формирования у них конкретных навыков и умений, направленных на практическое решение примеров и задач, требующих конечного ответа, чаще всего в виде числа или математического выражения. Эта особенность накладывает специфические требования к структуре и наполнению интеллектуальной обучающей системы, связанные с многократным повторением и решением предложенных задач, и требует включения в ее состав следующих итерационных блоков [3, 4]: 1) информационного блока, содержащего конкретные рекомендации по способам

решения задач; 2) блока примеров, где студент частично участвует в процессе их решения; 3) блока контроля, предлагающего выбор уровня сложности тестовых примеров и задач, и реализующего оценивание студента в результате их решения; 4) блока оценивания и рекомендаций, реализующего обратную связь учебного процесса.

ПОСТАНОВКА ПРОБЛЕМЫ

В результате проведенного анализа существующих разработок компьютерных обучающих программ в сфере технического образования было выявлено, что именно формирование обратной связи представляет собой наибольшую трудность при разработке компьютерных систем обучения, т. к. в эту связь необходимо включить большое количество критериев оценивания знаний и диагностирования причин возникновения ошибки (чаще всего слабоформализуемых) [5, 6]. Вместе с тем предложенные критерии не всегда в полной мере отражают сущность и функции обратной связи, реализуемой преподавателем, а также в изученных разработках не учтены особенности и специфика обратной связи при изучении технических дисциплин (большое количество математических конструкций, сложность расчетов и т. д.). Таким образом, задача формирования обратной связи интеллектуальной обучающей системы, включающей в себя функции оценивания и рекомендации для исправления ошибок, и являющейся адекватной процессу обучения технической специальности в высшем учебном заведении представляет собой актуальную научно-практическую задачу.

РЕШЕНИЕ ПРОБЛЕМЫ

При изучении сущности обратной связи учебного процесса были выявлены ее основные составляющие, реализуемые преподавателем: функция оценивания, функция поиска места ошибки в случае неверного ответа студента и рекомендательная функция повторного изучения материала по результатам определяемого места ошибки. Две последние функции в интеллектуальной системе обучения были объединены в блок диагностирования ошибок (рис. 1).

Блок диагностирования выполняет обнаружение ошибки при решении практических задач по разветвляющемуся алгоритму: обнаружение факта ошибки при несовпадении результата решения с эталонным заданным значением, далее – обнаружение места ошибки на каком-либо этапе решения задачи, определение класса ошибки, т. е. ее принадлежность к лекционной теме или типу решаемых задач, затем – диагностирование «пробела» в знаниях и рекомендации по улучшению полученного результата, например, – изучить теоретический материал (с указанием тем или глав), повторить алгоритм решения предложенных задач, перейти к решению задач более простого уровня и т. д. (рис. 2.)

Для оценивания уровня знаний в интеллектуальной обучающей системе предлагается учитывать не только знание алгоритма решения и способность применить его к предлагаемой задаче, но также рассматривать все множество допущенных ошибок и степень их влияния на результат, с целью устранения субъективности закладываемых параметров оценки.

Множество ошибок, допускаемых студентами в процессе решения задач, предложено декомпозировать на следующие подмножества (признак классификации – уровень ошибки): комплексные или системные ошибки – подмножество $\{A\}$, грубые ошибки – подмножество $\{B\}$, исправимые ошибки – подмножество $\{C\}$. Каждое из этих подмножеств содержит определенные виды ошибок, выявленные в процессе педагогической деятельности [3], и каждому виду ошибки присвоен свой вес в зависимости от влияния этой ошибки на исход решения; значения весов сведены в матрицу P размерностью $(n\times1)$).

Например, системные ошибки, связанные с непониманием метода или неверного применения способа решения имеют наибольший вес (1–2 балла), грубые ошибки, связанные, например, с неправильной подстановкой коэффициентов имеют вес 0,5–1 балла, исправимые ошибки, такие как ошибки в расчетах или ошибки из-за невнимательности имеют минимальный вес 0,1–0,2 балла. Для того, чтобы предоставить возможность обучаемому повысить оценку и самостоятельно устранить допущенные ошибки введены параметры, отражающие способ исправления ошибки, которым также присваи-

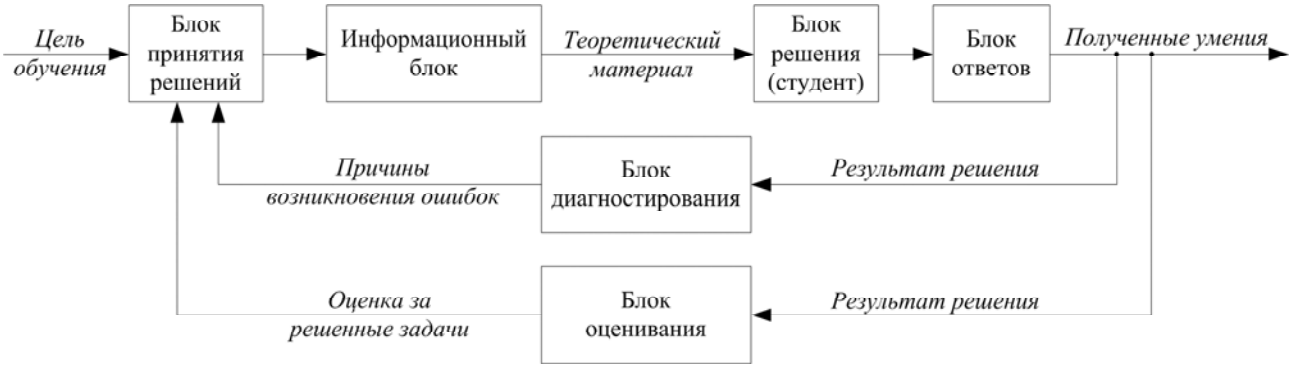


Рис. 1. Схема реализации прямых и обратных связей в интеллектуальной обучающей системе

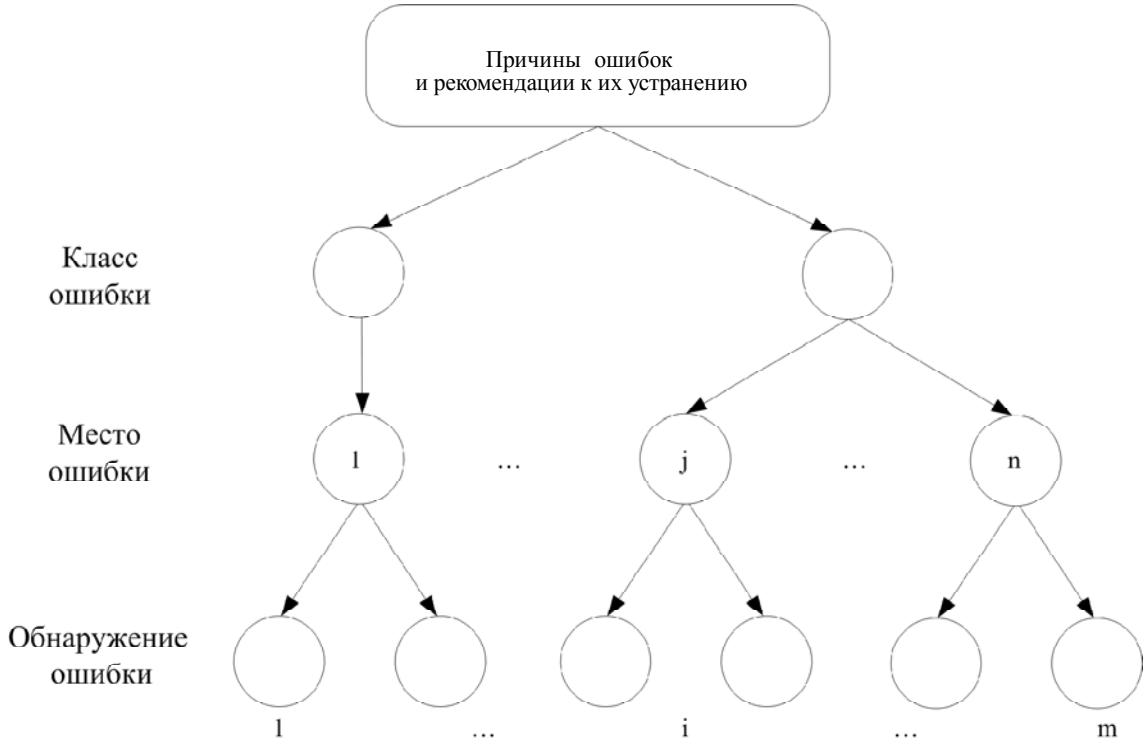


Рис. 2. Схема диагностирования возникновения ошибок

вается вес, имеющий отрицательное значение и уменьшающий баллы матрицы *P*; значение весов исправления ошибки сведено в матрицу весов исправления ошибок *Q* размерностью (*n*×*n*). Среди способов исправления выделены 3 основные группы: самостоятельное исправление (после информации о том, что она допущена), устранение ошибки после рекомендации с указанием ее места, устранение ошибки интеллектуальной системой после нескольких неудачных попыток обучаемого. Для формализации оценивания предлагается использовать критерий максимизации разности между ожидаемым результатом (высший балл по соответствующей шкале) *S* и полученными штрафными баллами за допущенные ошибки *R*, а также показатель ошибок, определяющий степень влияния допущенных ошибок на результат решения *H* (сумма элементов матрицы *H* представляет собой штрафные баллы *R*):

$$K = (S - R) \to \max\,;\; H = P \times Q\,. \qquad (1)$$

Предложенный подход к контролю и оцениванию уровня знаний, а также диагностированию и формированию множества рекомендаций при использовании компьютерных средств обучении находит применение на кафедре систем управления летательными аппаратами Национального аэрокосмического университета им. Н. Е. Жуковского «ХАИ» в рамках разработки компьютерных обучающих программ по основным читаемым курсам [7, 8, 9].

ЗАКЛЮЧЕНИЕ

Предлагаемый подход позволяет реализовать не только прямые связи процесса обучения в заданной предметной области с использованием компьютерных средств, а также его наиболее значимую часть – обратную связь, с помощью которой осуществляется поэтапный контроль процесса обучения, определяются места возникновения ошибок, в отдельных случаях распознаются причины их возникновения и формируются рекомендации по их устранению и улучшению результатов обучения.

Особенностью предложенного подхода является возможность учитывать не только количество ошибок при оценивании сформированных умений, а также степень их влияния на результат и качество помощи интеллектуальной системы, оказанной при исправлении допущенных ошибок, что более полно и точно отражает обратную связь учебного процесса, связанную с оцениванием обучающихся. Вместе с тем, реализация достаточно сложной процедуры диагностирования причин возникновения ошибок требует систематической доработки сформированных компьютерных модулей с целью повышения эффективности использования интеллектуальной обучающей системы.

СПИСОК ЛИТЕРАТУРЫ

1. Информационно-аналитические модели управления технически высшими учебными заведениями [текст] / А. Н. Гуржий, В. С. Кривцов, А. С. Кулик и др. ; Нац. аэрокосмич. ун.-т им. Н. Е. Жуковского «Харьковский авиационный институт». – Х. : ХАИ, 2004. – 386 с.

2. *Башмаков, А. И.* Разработка компьютерных учебников и обучающих систем / А. И. Башмаков, И. А. Башмаков. – М. : Информационно-издательский дом «Филинъ», 2003. – 616 с.
3. *Пищухина, О. А.* Информационная технология разработки компьютерных обучающих программ / О. А. Пищухина, Д. В. Бирюкова, О. В. Клименко // Радіоелектронні і комп'ютерні системи. – 2006. – №2 (14). – С. 57–62.
4. *Пищухина, О. А.* Интеллектуальные обучающие системы в сфере технического образования / Пищухина О. А // Інформаційні комп'ютерні технології в машинобудуванні. – 2010. – С.118.
5. *Пищухина, О. А.* Особенности разработки компьютерных обучающих программ / О. А. Пищухина, Н. В. Нечипорук, Д. В. Бирюкова, О. В. Клименко // Актуальні проблеми сучасних наук: теорія та практика. – 2006» – Том 9. – Дніпропетровськ : Наука і освіта. – 2006. – С. 84–86.
6. *Пищухина, О. А.* Формирование подхода к контролю и оценке знаний и умений при разработке компьютерных обучающих программ / О. А. Пищухина, О. В. Клименко, Д. В. Бирюкова // Системи підтримки прийняття рішень. Теорія і практика – К. : ІПММС НАНУ. – 2007. – С.134–136.
7. *Піщухіна, О. О.* Комп'ютерна програма «Навчальна програма розв'язання диференційних рівнянь операторним методом» / О. О. Піщухіна, Д. В. Бирюкова, О. В. Клименко // Свід. про реєстрацію авторського права на твір № 17725. – зареєстр. в Держ. департ. інтелектуальної власності Мін. освіти і науки України; реєстр. 28.08.2006 р.
8. *Піщухіна, О. О.* Комп'ютерна програма «Навчальна програма розв'язання диференційних рівнянь методом Ейлера» / О. О. Піщухіна, Д. В. Бирюкова, О. В. Клименко // Свід. про реєстрацію авторського права на твір № 17651. – зареєстр. в Держ. департ. інтелектуальної власності Мін. освіти і науки України; реєстр. 15.08.2006 р.
9. *Дергачев, К. Ю.* Формирование комплекса интеллектуальных обучающих программ при решении навигационных задач / К. Ю. Дергачев, О. А. Пищухина, А. Ю. Клочок // Людина і космос. – Днепропетровск, 2011. – С. 211.

Стаття надійшла до редакції 26.05.2011.

Піщухіна О. О., Клочок А. Ю.

ПІДХІД ДО ФОРМУВАННЯ ЗВОРОТНЬОГО ЗВ'ЯЗКУ В ІНТЕЛЕКТУАЛЬНИХ СИСТЕМАХ, ЩО НАВЧАЮТЬ, У СФЕРІ ВИЩОЇ ТЕХНІЧНОЇ ОСВІТИ

Запропоновано підхід до формування зворотнього зв'язку процесу навчання з використанням інтелектуальних комп'ютерних систем, особливістю якого є поетапний контроль процесу навчання, визначення місця виникнення помилок, в окремих випадках – розпізнавання причин їх виникнення і формування рекомендацій з їх усунення.

Ключові слова: інтелектуальні комп'ютерні системи, що навчають, вища технічна освіта, зворотний зв'язок.

Pishchukhina O. A., Klochok A. Yu.

APPROACH TO THE FEEDBACK FORMING IN INTELLIGENT LEARNING SYSTEM IN THE SPHERE OF HIGHER TECHNICAL EDUCATION

The approach to the feedback of learning process forming while using intelligent computer system is offered. It's feature is step-by-step control of learning process, determining of mistakes origin point, in other case – causes of its occurrence and forming recommendations of its eliminating.

Key words: intelligent computer learning system, higher technical education, feedback.

УДК 004.932.4

Степаненко О. О.[1], Піза Д. М.[2]

[1]Канд. техн. наук, доцент Запорізького національного технічного університету
[2]Д-р техн. наук, професор Запорізького національного технічного університету

ПРОГРАМНИЙ КОМПЛЕКС ДЛЯ АНАЛІЗУ Й ОБРОБКИ ЕХО-ІМПУЛЬСНИХ ЗОБРАЖЕНЬ

На основі аналізу методів обробки ехо-імпульсних зображень було виявлено недоліки існуючих підходів і запропоновано два нових методи. Розроблені методи і програмна система, що базується на них, дозволяють підвищити ефективність візуального аналізу медичних та сейсмічних ультразвукових зображень.

Ключові слова: математична модель, метод лінійного передбачення, суперпозиція імпульсного сигналу, спектр, ехо-імпульсне зображення, модуль математичних обчислень.

ВСТУП

За останні 40 років ультразвук став важливою діагностичною методикою. Його потенціал у відображенні медичної діагностики було визнано у 1930-х і 1940-х роках, коли Теодор і Фридріх Дуссіки спробували використовувати ультразвук для того, щоби діагностувати пухлини мозку. Однак, лише у 1970-х, робота цих та інших піонерів досліджень ультразвуку реально принесла свої плоди.

Разом з технологічними вдосконаленнями, ультразвук прогресував від великої громіздкої машини, яка відтво-

рювала неоптимальні зображення, до переносного, зручного для використання і складного пристрою. Така еволюція потребувала тісного поєднання фізики, фізіології, медицини, техніки і керування [1].

Однак, навіть на сучасному етапі розвитку ультразвуку зображення, які отримуються за його допомогою, завжди неідеальні і містять шуми різного виду, які вносять істотні спотворення та ускладнюють процес прийняття рішень, заснований на їх візуальному аналізі. Це пояснюється як самою природою ультразвуку, так і недосконалістю апаратури [2]. Тому актуальною є задача розробки методів та алгоритмів покращення якості ультразвукових та ехо-імпульсних зображень та їх програмна реалізація, з метою покращення їх якості та спрощення їх візуального аналізу.

ПОСТАНОВКА ЗАДАЧІ

Якщо розглядати ультразвукове зображення як набір суперпозицій невідомих імпульсних сигналів, що відповідає методам отримання цих зображень, то задача їх обробки полягає в аналізі цих суперпозицій.

Існуючі методи аналізу висувають до сигналів суперпозиції обмеження, які не можуть бути виконані в рамках задач ультразвукової діагностики. Зокрема, метод інверсної фільтрації вимагає апріорного знання форми елементарного імпульсу, а метод кепстрального аналізу припускає, що всі імпульси мають однакову форму, причому висувається обмеження на їх число [3, 4].

Реалії практичних задач (особливо в сейсмології і медичній ультразвуковій діагностиці) змушують вести аналіз сигнальних суперпозицій, що включають сотні і тисячі елементарних імпульсів. Причому через ефекти дифракції, реверберації, розсіювання і загасання форми цих імпульсів не тільки невідомі, але й різні, що не дає змоги використати жоден із відомих методів для аналізу таких суперпозицій [5].

Отже, актуальною є розробка і програмна реалізація таких методів аналізу суперпозицій сигналів, які б не висували вимог до форми і кількості сигналів у суперпозиції.

ОСОБЛИВОСТІ РЕАЛІЗАЦІЇ РОЗРОБЛЕНИХ МЕТОДІВ ТА АЛГОРИТМІВ

У результаті вивчення існуючих методів аналізу суперпозицій імпульсних сигналів і виявлення їх недоліків було створено два нових методи обробки ехо-імпульсних зображень.

Запропоновані методи обробки ультразвукових зображень, базовані на застосуванні адаптованих до даної галузі методів лінійного передбачення, які переважно використовуються в задачах обробки голосу, та сигнального підпростору для аналізу суперпозицій імпульсних сигналів.

Метод аналізу суперпозицій імпульсних сигналів, заснований на ідеї методу параметричного спектрального аналізу, дозволяє підвищити його розподільну здатність у часовій області і достовірність виявлення меж слабоконтрастних ділянок шаруватих структур без використання будь-якої апріорної інформації щодо форми елементарних імпульсів. Метод є стійким до впливу вимірювальних шумів.

Запропонований алгоритм візуалізації результатів аналізу ехо-імпульсних зображень у рамках методу параметричного спектрального аналізу відноситься до області методів комп'ютерного бачення, оскільки візуалізуються полюси, а не реальні фізичні амплітуди, які відповідають моделі лінійного передбачення Фур'є-спектральних характеристик реєстрованих часових імпульсних суперпозицій. У межах імпульсної ультразвукової діагностики основна проблема використання методів параметричного спектрального аналізу пов'язана з використанням системи ЧАРП (часового автоматичного регулювання підсилення), що призводить до підйому амплітудно-часових характеристик, оскільки яскравість ультразвукових ехо-імпульсних зображень має бути позитивною.

Отже для реалізації методів спектрального аналізу другого порядку аналізована послідовність має бути знакозмінною. Для цих цілей доцільно використовувати перетворення Гільберта кожної аналізованої сигнальної послідовності, що забезпечує результат, близький до результату чисельного диференціювання, але гарантує стійкість до вимірювальних шумів (у порівнянні з методом чисельного диференціювання).

Додаткову обробку яскравості синтезованих зображень пропонується виконувати методами еквалізації гістограм або евристичного зрізування.

Модель суперпозиції сигналів, що перекриваються, за умови, що залежність еквівалентного шумового спектрального компонента близька до білого шуму, доцільно подати у вигляді

$$\hat{S}(f_k) = \sum_{m=1}^{M=2N} p(m) S(f_{k-m}), \; k = 1,2,..., K, \qquad (1)$$

де $p(m)$ – коефіцієнти лінійного прогнозування; M – порядок фільтра лінійного прогнозування; K – число частот, на яких вимірюється спектральна характеристика $S(f)$ (воно, як правило, збігається із числом часових відліків сигнальної послідовності $s(t)$ [6].

Задача підвищення роздільної здатності візуального аналізу суперпозицій сигналів, що перекриваються, вирішується на основі використання моделі синтезованого «відфільтрованого» ехо-імпульсного зображення:

$$\hat{s}(t) = \frac{\sigma^2}{\left[1 - \sum_{i=1}^{M} p_i e^{-j2\pi i \Delta f t}\right]^2}, \qquad (2)$$

де σ^2 розглядається як дисперсія помилки лінійного прогнозування для заданого порядку моделі M; $\Delta f = f_k - f_{k-1}$ – величина, що характеризує дискретність спектральної характеристики $S(f)$, яка виникає через кінцеве значення часового інтервалу реєстрації суперпозиції $s(t)$.

Структура запропонованого алгоритму є наступною:

1. Перехід з часової області первинних вимірювань у спектральну область на основі використання алгоритму прямого перетворення Фур'є (спектральний аналіз І-го порядку).

2. Розрахунок моделі лінійного прогнозування для спектральної області. Порядок моделі М лінійного прогнозування визначає значуще число імпульсів, що відображатимуться і які будуть продетектовані на наступній стадії.

3. Зворотний перехід зі спектральної області в часову область на основі використання алгоритму нелінійного параметричного спектрального аналізу (спектральний аналіз ІІ-го порядку) на основі синтезованої моделі (2) [6].

Схема алгоритму наведена на рис. 1.

Тобто, на вході системи присутня вхідна амплітудно-імпульсна послідовність (суперпозиція зашумлених сигналів, що перекриваються), на виході – відфільтрована «синтезована» часово-залежна послідовність, що вимірюється в відношеннях амплітуд.

Оскільки основним об'єктом досліджень були ультразвукові зображення, то вихідна інформація надавалася як зображення відповідного формату, що зчитувались у комп'ютер у вигляді чисельної матриці. При цьому описаний алгоритм застосовувався для кожного стовпця аналізованого зображення, тому якщо число стовпців дорівнювало 400, то алгоритм застосовувався 400 разів з подальшою візуалізацією результатів уже як для єдиного синтезованого зображення.

Метод обробки ультразвукових зображень, заснований на методі сигнального підпростору, направлений на нейтралізацію впливу багатократних перевідбиттів, є стійким до впливу вимірювальних і структурних шумів і не вимагає апріорного знання моделі аналізованої структури.

Він базується на використанні властивостей власних векторів кореляційної матриці C, спектральної залежності $S(f)$ сигнальної суперпозиції, що аналізується, і розкладанні ортогонального простору, утвореного власними векторами кореляційної матриці, на два ортогональних підпростори: сигнальний і шумовий. З математичної точки зору розв'язання задачі ґрунтується на пошуку такої нової системи координат, у якій інформативні гармоніки спектральної залежності $S(f)$ були б ортогональними в межах заданого частотного інтервалу F (обумовленого тривалістю інтервалу вимірів і частотою дискретизації сигнальної суперпозиції).

Матрицю C можна подати у вигляді суми двох кореляційних матриць, а перехід у часову область здійснюється на основі моделі (відображаються полюси знаменника моделі (3)):

$$\hat{s}(t) = 1/\, s_i^T C_q s_i ;\;\; S_i^T = \left\lfloor e^{-j2\pi t_i f_1} \ldots e^{-j2\pi t_i f_N} \right\rfloor ;$$
$$f \in F ;\; t_i = \text{var} , \qquad (3)$$

де інтервал $\Delta t = t_i - t_{i-1}$ може бути обраний як завгодно малим [7].

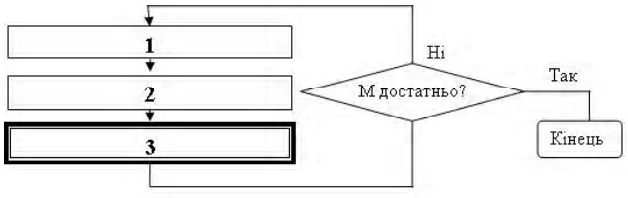


Рис. 1. Схема нового алгоритму підвищення роздільної здатності

ОСОБЛИВОСТІ РЕАЛІЗАЦІЇ ПРОГРАМНИХ КОМПОНЕНТІВ

Особливістю запропонованої системи є той факт, що методи, які вона реалізує, із мінімальними змінами можуть бути застосовані до різних практичних галузей (найбільш очевидними є галузі сейсмічної розвідки вуглеводнів і ультразвукової медичної діагностики).

Таким чином, одними з найкритичніших вимог до системи є її гнучкість і здатність масштабування. Виходячи саме з цих міркувань і була розроблена її структура.

Потреба у гнучкості системи диктувала необхідність використання для її розробки модульного підходу, причому кількість модулів не повинна була бути сталою величиною, до того ж, вони могли бути виконані з використанням різних технологій і мов програмування. Це дозволяло організувати введення до системи нових модулів (які, наприклад, могли містити модифікації методів або їх нові реалізації) без будь-яких ускладнень. Поєднання модульного підходу з об'єктно-орієнтованим підходом, окрім цього, також вирішує і проблему масштабованості системи [8].

Враховуючи те, що програмне забезпечення розроблювалось для комп'ютерів на основі операційних систем сімейства Microsoft Windows, у якості платформи розробки ядра системи було вирішено обрати платформу .NET, а в якості основної мови програмування – C#.

Значним недоліком використання платформи .NET є низька швидкодія генерованого байт-коду, тому для розробки критичних для продуктивності модулів доцільно використовувати інші мови програмування. В якості такої додаткової мови програмування використовувалась мова Fortran. Це пов'язано з її націленістю на вирішення математичних задач, а також наявністю швидких компіляторів. Для отримання відповідних модулів системи використовувалась реалізація компілятора Intel Fortran Compiler 10, як одного із найкращих компіляторів Fortran у плані швидкодії кінцевих програм і бібліотек [9].

Вищесказане висувало також вимоги до середовища, у якому велася розробка системи. Вибір середовища, адекватного вирішуваній задачі, було одним з вирішальних факторів успішної розробки. Очевидним рішенням було обрання середовища Microsoft Visual Studio 2005.

На рис. 2 наведена функціональна схема системи обробки ультразвукових зображень, розроблена з урахуванням наведених вище міркувань щодо вимог до її структури.

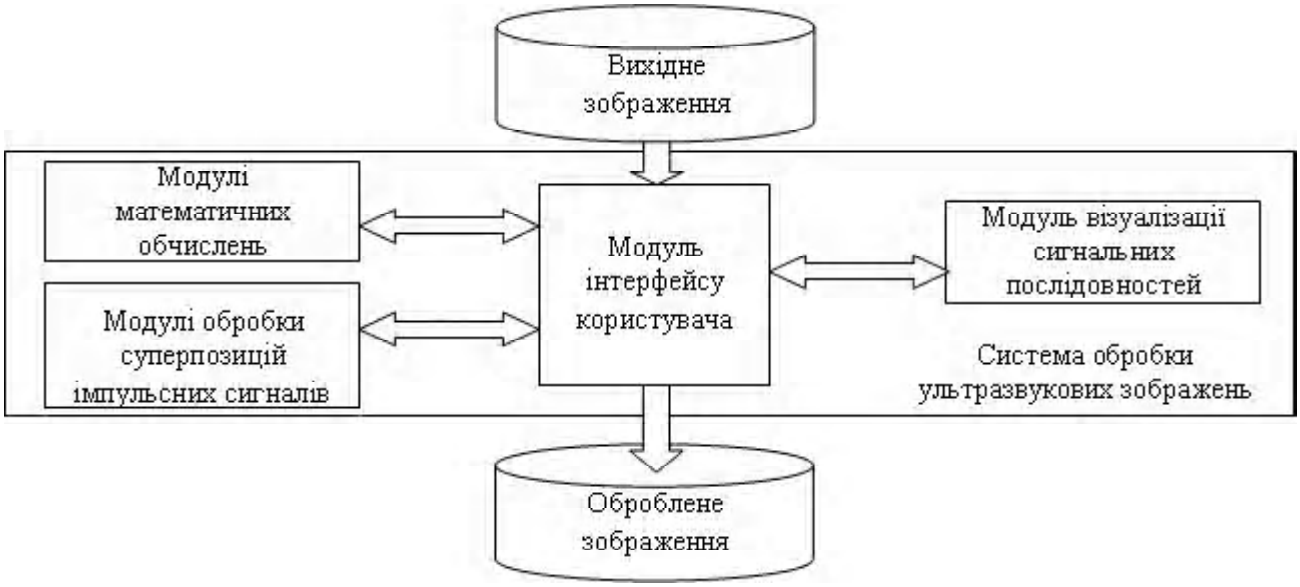


Рис. 2. Функціональна схема програми

Із рисунку видно, що система може розглядатися як сукупність чотирьох взаємопов'язаних логічних частин: модуля інтерфейсу користувача, модуля візуалізації сигнальних послідовностей, модулів математичних обчислень, модулів обробки суперпозицій імпульсних сигналів. Модуль інтерфейсу користувача є програмою, яка функціонує як сполучна ланка між іншими модулями системи. Він надає користувачеві змогу використовувати сервіси, що їх надає система, за допомогою екранних форм. При розробці модуля використовувалась мова програмування C#.

Модуль візуалізації сигнальних послідовностей являє собою елемент керування, який подає послідовність відліків дискретного сигналу у вигляді відповідного графіку сигналу або спектру. Він заснований на ієрархії класів Windows Forms і реалізований на мові програмування C#.

Модулі математичних обчислень – це поєднання розроблених за допомогою мов C# і C динамічних бібліотек, які надають можливість використання таких засобів, як комплексні числа, матриці комплексних чисел, швидке перетворення Фур'є.

Модулі обробки суперпозицій імпульсних сигналів – дві групи модулів, які реалізують адаптивний метод лінійного передбачення (бібліотека, реалізована на C#), і метод сигнального підпростору (поєднання бібліотек, реалізованих за допомогою C# і Fortran).

АЛГОРИТМ ФУНКЦІОНУВАННЯ ПРОГРАМИ

Першою дією програми після запуску є сканування директорії, в якій мають знаходитися модулі розширення системи (саме в такому вигляді було реалізовано методи обробки суперпозицій імпульсних сигналів). Знайдені коректні модулі додаються у список, який далі буде використовуватися користувачем при виборі методу обробки. Після цього користувачеві надається екранна форма, за допомогою якої буде здійснюватися взаємодія користувача з системою.

Вхідними даними для програми слугують ехоімпульсні зображення в одному із розповсюджених графічних форматів (BMP, GIF, JPEG, PNG або TIFF) і параметри обраного користувачем методу.

У загальному вигляді алгоритм функціонування програми наведено на рис. 3.

Рис. 3. Алгоритм роботи програми

Після запуску програми користувачеві надається вікно, аналогічне наведеному на рис. 4. Як видно з рисунку, головне вікно програми можна розділити на декілька областей: – *заголовок вікна* використовується стандартним чином – для переміщення вікна, його згортання, розгортання і закриття (тобто, виходу з програми); *головне меню* містить: меню роботи з файлами («File»), меню налаштування графіків («Plots»), меню допомоги («Help»); *область візуалізації зображень* містить: вхідне зображення («Input image»), графік суперпозиції сигналів стовпця вхідного зображення («Input signal»), графік амплітудного спектра суперпозиції сигналів стовпця вихідного зображення («Input spectrum»), оброблене зображення («Output image»), графік суперпозиції сигналів стовпця обробленого зображення («Output signal»), графік амплітудного спектра суперпозиції сигналів стовпця обробленого зображення («Output spectrum»), перемикач номера стовпця вхідного і обробленого зображення, який розглядається на графіках («View column»), час обробки останнього зображення («Time elapsed»).

Область інформації про вхідне зображення («Image info») містить шлях до відкритого зображення («File name») і його геометричні розміри («Image size»). Тут також знаходиться перемикач перетворення Гільберта («Apply Hilbert transform») – активувати його рекомендується при обробці ультразвукових медичних зображень; *область налаштувань методів обробки зображень («Processing options»)* містить список наявних методів («Processing») і дозволяє визначати їх параметри (кнопка «Options…»), а також – список методів обробки отриманого зображення («Postprocessing») і кнопку активації обробки («Process»); *рядок стану* містить відомості про поточний стан програми: якщо вона знаходиться у черговому стані – напис «Ready», якщо вона знаходиться у стані обробки зображення – напис «Processing», смугу поточного прогресу і кількість оброблених стовпців зображення.

ВИСНОВКИ

В роботі розглянуто проблему обробки ехо-імпульсних зображень з метою підвищення їх якості при візуалізації. Вона полягає в аналізі суперпозицій невідомих імпульсних сигналів, які складають ці зображення. Існуючі підходи до аналізу, а саме, метод інверсної фільтрації та метод кепстрального аналізу, не можуть бути використані в даній області через ряд обмежень, які вони накладають на вхідні суперпозиції. Тому було запропоновано методи, вільні від обмеження існуючих підходів, і засновані на використанні методів лінійного передбачення і сигнального підпростору у спектральній області.

За розробленими методами реалізовано програмне забезпечення для обробки ультразвукових зображень. Програму реалізовано на платформі .NET за допомогою мов C# і Fortran з використанням вільних бібліотек

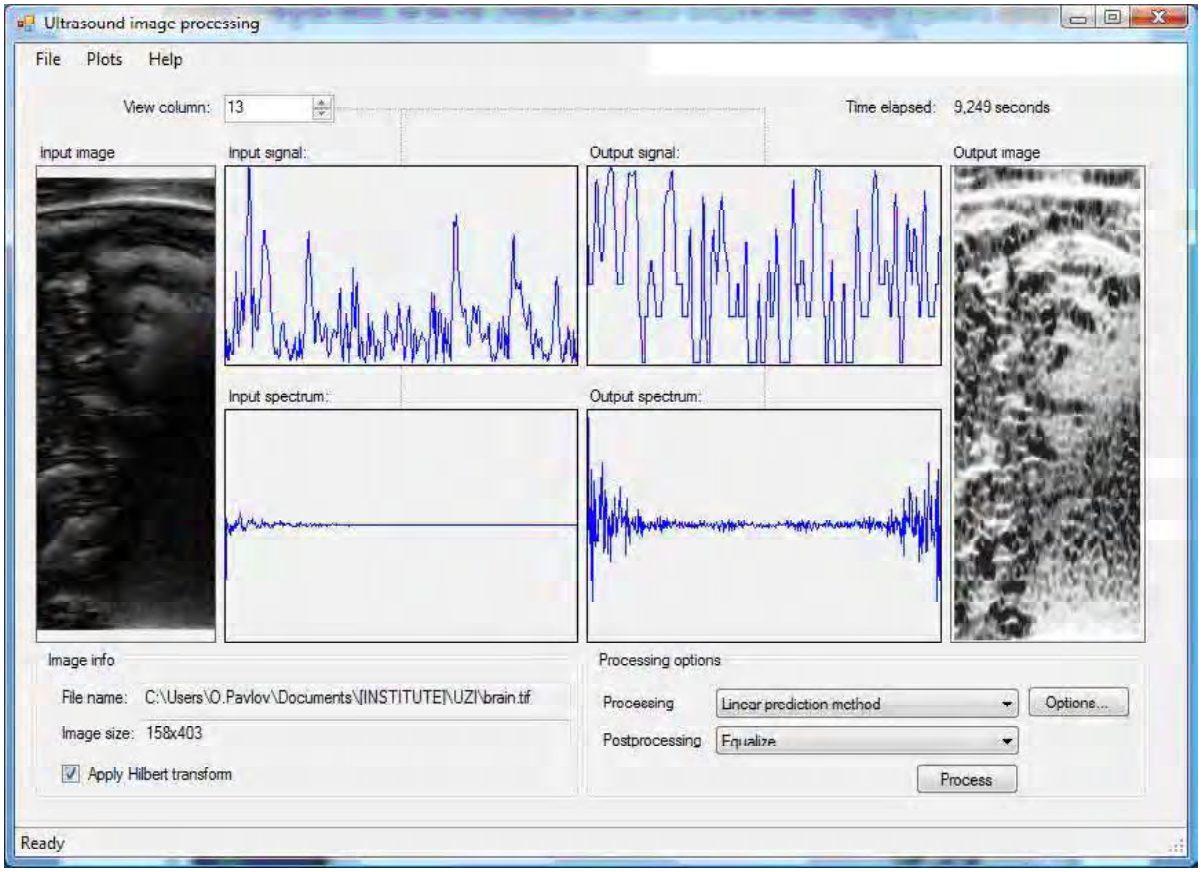


Рис. 4. Головне вікно програми

швидкого перетворення Фур'є fftw та пошуку власних значень і векторів матриць EISPACK. Програма може успішно експлуатуватися на комп'ютерах під керуванням операційних систем Microsoft Windows 2000/XP/ 2003/Vista. Мінімальними апаратними вимогами є: процесор класу Intel Pentium III тактовою частотою 500 МГц; 96 МБ оперативної пам'яті; 200 МБ вільного місця на жорсткому диску; монітор SVGA і відеоадаптер з підтримкою розподільної здатності 1024x768.

Галузь застосування програми – підвищення якості зображень, отриманих за допомогою пристроїв ультразвукових досліджень або сейсмограм.

СПИСОК ЛІТЕРАТУРИ

1. *Newman, P.* The history of ultrasound / P. Newman, G. Rozycki // Surgical Clinics of North America. – 1998. – № 2 (78). – P. 179–195.
2. *Митьков, В. В.* Практическое руководство по ультразвуковой диагностике. Общая ультразвуковая діагностика / В. В. Митьков. – М. : Видар, 2003. – 720 с.
3. *Иванов, В. К.* Теория линейных некорректных задач и ее приложения / В. К. Иванов, В. В. Васин, В. П. Танана. – М. : Наука, 1978. – 200 с.
4. *Чайделрс, Д. Д.* Кепстр и его применение при обработке данных / Д. Д. Чайделрс, Д. П. Скиннер, Р. Ч. Кемерайт // ТИИЭР. – 1977. – № 10. – С. 5–23.
5. *Степаненко, А. А.* Повышение чувствительности ультразвуковой диагностики на основе метода параметрического спектрального анализа второго порядка / А. А. Степаненко, А. М. Ахметшин // Клиническая информатика и телемедицина. – 2005. – Т.2, № 1. – С. 98–100.
6. *Степаненко, А. А.* Разложение суперпозиций неизвестных импульсных сигналов методом адаптивного спектрального анализа второго порядка / А. А. Степаненко, А. М. Ахметшин // Искусственный интеллект. – 2005. – № 3. – С. 610–618.
7. *Степаненко, А. А.* Анализ эхо-импульсных изображений слоистых структур: метод сигнального подпространства / А. А. Степаненко, А. М. Ахметшин // Радиоэлектроника, інформатика, управление. – 2004. – № 1. – С. 5–9.
8. *O'Docherty, M.* Object-Oriented Analysis and Design / M. O'Docherty. – Wiley & Sons, 2005. – 560 p.
9. *Рыжиков, Ю.* Современный Фортран / Ю. Рыжиков. – Минск : Корона Принт, 2007. – 288 с.

Стаття надійшла до редакції 14.04.2011.

Степаненко А. А., Пиза Д. М.

ПРОГРАММНЫЙ КОМПЛЕКС ДЛЯ АНАЛИЗА И ОБРАБОТКИ ЭХО-ИМПУЛЬСНЫХ ИЗОБРАЖЕНИЙ

На основе анализа методов обработки эхо-импульсных изображений были выявлены недостатки существующих подходов и предложены два новых метода. Разработанные методы и базирующаяся на них программная система позволяют повысить эффективность визуального анализа медицинских и сейсмических ультразвуковых изображений.

Ключевые слова: математическая модель, метод линейного предсказания, суперпозиция импульсного сигнала, спектр, эхо-импульсное изображение, модуль математических вычислений.

Stepanenko O. O., Piza D. M.

PROGRAM COMPLEX FOR ANALYSIS AND TREATMENT ECHO-PULSE IMAGES

On the basis of analysis of methods of processing of echo-pulse images the lacks of existent approaches were educed and two new methods are offered. Worked out methods and programmatic system which is based on them, allow to promote efficiency of visual analysis medical and seismic ultrasonic images.

Key words: mathematical model, method of linear prediction, superposition of impulsive signal, spectrum, echo-pulse images, module of mathematical calculations.

УДК 519.7:004.8

Шафроненко А. Ю.[1], Волкова В. В.[2], Бодянский Е. В.[3]

[1]Стажер-исследователь Харьковского национального университета радиоэлектроники
[2]Старший преподаватель Харьковского национального университета радиоэлектроники
[3]Д-р техн. наук, профессор Харьковского национального университета радиоэлектроники

АДАПТИВНАЯ КЛАСТЕРИЗАЦИЯ ДАННЫХ С ПРОПУЩЕННЫМИ ЗНАЧЕНИЯМИ

Предложена адаптивная нейросетевая система, позволяющая решать задачу кластеризации данных с пропущенными значениями в on-line режиме с постоянной коррекцией восстанавливаемых элементов таблицы и центроидов кластеров. Введенная нейросистема характеризуется высоким быстродействием и простотой численной реализации.

Ключевые слова: адаптивная нейросетевая система, восстановление пропусков, кластеризация.

ВВЕДЕНИЕ

Задача кластеризации данных является важным элементом общей проблемы Data Mining, а для ее решения сегодня существует множество подходов и алгоритмов: от сугубо интуитивных и эвристических до строго математических [1–9]. Вместе с тем, во многих задачах Data

Mining, включая, естественно, кластеризацию, исходные таблицы данных «объект-свойство» могут содержать пустые клетки (пропуски), информация в которых по тем или иным причинам отсутствует. Задаче восстановления таких пропущенных наблюдений уделялось достаточное внимание [6, 10–12] , при том наиболее эффективными в данной ситуации оказались подходы, основанные на математическом аппарате мягких вычислений (вычислительного интеллекта) [13, 14], и, прежде всего, искусственных нейронных сетей [12, 15–17]. Вместе с тем, известные подходы к восстановлению пропусков и традиционные алгоритмы кластеризации работоспособны лишь в случаях, когда исходная таблица задана априорно и число ее строк или столбцов не изменяется в процессе обработки. В то же время существует достаточно широкий класс задач, когда данные поступают на обработку последовательно в on-line режиме, при этом заранее неизвестно, какой из обрабатываемых векторов-образов может содержать пропуски. При этом процессы восстановления данных и их кластеризации должны протекать одновременно в реальном времени.

ПОСТАНОВКА ЗАДАЧИ

Пусть задана исходная ($N\times n$) таблица ($N\geq 1$, $n\geq 1$) «объект-свойство».

Таблица 1, содержащая информацию об N объектах, каждый из которых описывается $(1\times n)$ – вектором-строкой признаков $\underset{\rightarrow}{x}_i = (x_{i1},...,x_{ip},...,x_{ij},...,x_{in})$, при этом предполагается, что N_G строк могут иметь по одному пробелу, а $N_F = N–N_G$ заполнены полностью. В процессе обработки таблицы необходимо заполнить пропуски и сформировать m кластеров. При этом требуется, чтобы восстановленные элементы были бы в определенном смысле «наиболее правдоподобны» или «близки» к априори неизвестным закономерностям, содержащимся в таблице, которые могут неизвестным образом изменяться в процессе обработки.

АДАПТИВНЫЙ АЛГОРИТМ ЗАПОЛНЕНИЯ ПРОПУСКОВ

Для начала представим таблицу 1 в виде $(N\times n)$ – матрицы X, в которой отсутствует один элемент x_{kj} или в более общем случае отсутствуют N_G элементов. Предполагается [10], что между столбцами $\vec{x}_j = (x_{1j},...,x_{ij},...,x_{kj},...,x_{Nj})^T$ существует линейная корреляция, с учетом которой и производится восстановление пропуска с помощью уравнения линейной регрессии

$$\hat{x}_{kj} = w_{j0} + w_{j1}x_{k1} + w_{j2}x_{k2} + \\ + w_{j,j-1}x_{k,j+1} + ... + w_{jn}x_{kn} = \underset{\rightarrow}{X}_{kj} w_j , \qquad (1)$$

где $w_j = (w_{j0}, w_{j1},..., w_{jn})^T$ – $(n\times 1)$ – вектор-столбец параметров, подлежащих определению; $\underset{\rightarrow}{X}_{kj} = (1, x_{k1},..., x_{k,j-1}, x_{k,j+1},..., x_{kn})$ – $(1\times n)$ вектор-строка признаков k-того объекта без kj-того элемента и с единицей на первой позиции.

Вектор неизвестных параметров w_j может быть найден с помощью стандартного метода наименьших квадратов, для чего из матрицы X следует исключить k-ю строку, j-й столбец, добавить слева столбец, образованный единицами, и на основе полученной $((N-1)\times n)$ – матрицы X_j рассчитать оценки параметров

$$w_j = (X_j^T X_j)^+ X_j^T \vec{X}_j , \qquad (2)$$

где $\vec{X}_j = (x_{1j},..., x_{ij},..., x_{k-1,j}, x_{k+1,j},..., x_{Nj})^T$.

Если же пробелы имеются в N_G строках и в разных столбцах, из матрицы X исключаются все эти строки и на основании усеченной $(N_F \times n)$ матрицы n раз вычисляются векторы параметров w_j (2) для всех $j = 1,2,...,n$. Далее с помощью уравнения (1) заполняются все пробелы полученными оценками $\hat{x}_{kj}$.

Если же данные поступают на обработку последовательно строка за строкой, вместо оценки (2) может быть использован рекуррентный метод наименьших квадратов, который, однако, предлагает неизменность во времени всех связей, существующих в матрице данных. Экспоненциально же взвешенный рекуррентный метод наименьших квадратов может быть неустойчив при малых значениях параметра сглаживания.

В связи с этим для on-line анализа данных представляется целесообразным использование адаптивных алгоритмов обучения, имеющих как фильтрующие, так и следящие (для нестационарных ситуаций) свойства [18]. Если в процессе обработки массива данных было проанализировано и восстановлено N строк, то с приходом $(N+1)$-го наблюдения оценки w_j уточняются с помощью адаптивной рекуррентной процедуры

$$\begin{cases} w_j(N+1) = w_j(N) + r_j^{-1}(N+1)(x_{N+1,j} - \underset{\rightarrow}{X}_{N+1,j} w_j(N))\underset{\rightarrow}{X}^T_{N+1,j}, \\ r_j(N+1) = \alpha r_j(N) + \left\| \underset{\rightarrow}{X}_{N+1,j} \right\|^2, \end{cases} \qquad (3)$$

где $0 \leq \alpha \leq 1$ – параметр сглаживания, задающий компромисс между процессами фильтрации наблюдений и слежением за изменяющимися характеристиками данных.

Таблица 1.

	1	…	p	…	j	…	n
1	x_{11}	…	x_{1p}	…	x_{1j}	…	x_{1n}
…	…	…	…	…	…	…	…
i	x_{i1}	…	x_{ip}	…	x_{ij}	…	x_{in}
…	…	…	…	…	…	…	…
k	x_{k1}	…	x_{kp}	…	x_{kj}	…	x_{kn}
…	…	…	…	…	…	…	…
N	x_{N1}	…	x_{Np}	…	x_{Nj}	…	x_{Nn}
…	…	…	…	…	…	…	…

Обработку информации в режиме последовательного поступления данных удобно организовать с помощью нейросетевой системы, основными элементами которой являются *n*-параллельно работающих адаптивных линейных ассоциаторов (ALA) [19], настраиваемых с помощью алгоритма (3). При этом, если в *k*-й момент времени на вход системы поступает вектор наблюдений $\underset{\rightarrow}{x}_k = (x_{k1}, x_{k2}, ..., x_{kin})$ полный или с пропусками, на выходе появляется он же, если $\underset{\rightarrow}{x}_k$ полный, или его оценка, если в $\hat{x}_k$ содержались пробелы.

АДАПТИВНЫЙ АЛГОРИТМ КЛАСТЕРИЗАЦИИ

Для решения задачи кластеризации в on-line режиме целесообразно воспользоваться самоорганизующейся картой Т. Кохонена [20], имеющей простую архитектуру с прямой передачей информации и кроме нулевого рецепторного слоя содержащей единственный слой нейронов, чаще всего тех же адаптивних линейных ассоциаторов. Каждый нейрон связан с каждым рецептором нулевого слоя прямыми связями и со всеми остальными нейронами поперечными внутрислойными (латеральными) связями. Именно латеральные связи обеспечивают возбуждение одних нейронов и торможение других.

Благодаря такой организации сети, каждый нейрон-ALA получает всю информацию об анализируемом векторе-образе и генерирует на своем выходе соответствующий отклик, после чего между нейронами возникает конкуренция, в результате которой определяется единственный нейрон-победитель с максимальным выходным сигналом. Этот сигнал по латеральным связям обеспечивает возбуждение ближайших «соседей» победителя и подавление реакции далеко отстоящих нейронов. Таким образом в процессе конкурентного самообучения формируются группы нейронов, каждый из которых максимальным образом реагирует на образы «своих» кластеров-подобластей входного пространства сигналов.

Рассмотрим простейшую карту Кохонена, имеющую 1D топологию, *n* рецепторов и *m* (по числу возможных кластеров) нейронов, каждый из которых характеризуется собственным вектором синаптических весов $w_q^K, q = 1,2,...m$, определяющих центроид соответствующего кластера.

Каждый нейрон сети в *k*-й момент времени получает на входы *n*-мерный входной вектор

$$\tilde{x}_k = \frac{\hat{\underset{\rightarrow}{x}}_k^T}{\left\| \hat{\underset{\rightarrow}{x}}_k \right\|} \tag{4}$$

и генерирует на своем выходе сигнал

$$y_{kl} = w_l^{KT}(k-1)\tilde{x}_k, \tag{5}$$

зависящий от вектора синаптических весов $w_l^K(k-1)$, настроенных по данным, полученным до момента *k*, на определенную область (кластер) входного пространства. Близкие в смысле используемой метрики входные векторы $\tilde{x}_k$ и $\tilde{x}_i$ могут возбуждать либо один и тот же нейрон w_q^K, либо два нейрона соседа w_q^K и w_{q+1}^K или w_q^K и $w_{q-1}^k, q = 1,2,...,m.$

В основе алгоритма самоорганизации карты Кохонена лежат принципы конкурентного самообучения [19], при этом как и любая другая процедура настройки нейронной сети, работа алгоритма начинается с инициализации синаптических весов сети, которые обычно выбираются с помощью генератора случайных чисел, при этом для каждого из нейронов должно выполняться условие

$$\left\| w_l^K(0) \right\| = 1, l = 1,2,...,m. \tag{6}$$

Процедура самоорганизации реализуется в три основных этапа: конкуренции, кооперации и синаптической адаптации и начинается с анализа образа $\tilde{x}_k$, поступающего с рецепторного слоя на все нейроны сети. Для каждого из нейронов вычисляется расстояние

$$D(\tilde{x}_k, w_l^K(k-1)) = \left\| \tilde{x}_k - w_l^K(k-1) \right\|, \tag{7}$$

при этом, если входные векторы предварительно пронормированы с помощью выражения (4) так, что

$$\left\| \tilde{x}_k \right\| = 1, \tag{8}$$

а в качестве расстояния (7) используется эвклидова метрика, то мерой расстояния между $\tilde{x}_k$ и $w_l^K(k-1)$ может служить

$$D(\tilde{x}_k, w_l^K(k-1)) = 2(1 - \tilde{x}_k^T w_l^K(k-1)) =$$

$$= 2(1 - \cos(\tilde{x}_k, w_l^K(k-1))), \tag{9}$$

а мерой близости – скалярное произведение

$$-1 \le \tilde{x}_k^T w_l^K(k-1) = \cos(\tilde{x}_k w_l^K(k-1)) \le 1. \tag{10}$$

Далее определяется нейрон-победитель, «ближайший» ко входному образу $\hat{x}_k$ такой, что

$$D(\tilde{x}_k, w_l^K(k-1)) = \min_l D(\tilde{x}_k, w_l^K(k-1)), \tag{11}$$

после чего, временно опуская этап кооперации, можно подстроить синаптические веса сети с помощью модифицированного правила самообучения

$$w_q^K(k)=\begin{cases}\dfrac{w_q^K(k-1)+\eta(k)(\tilde{x}_k-w_q^K(k-1))}{\left\|w_q^K(k-1)+\eta(k)(\tilde{x}_k-w_q^K(k-1))\right\|}, \text{если } q\text{ - й нейрон победил,}\\ w_q^K(k-1) \text{ в противном случае.}\end{cases} \quad (12)$$

Несложно видеть, что процедура (12) реализует принцип «победитель получает все», при этом вектор синаптических весов $w_q^K(k-1)$ нейрона-победителя «подтягивается» ко входному вектору $\tilde{x}_k$ на расстояние, определяемое шагом настройки $\eta(k)$.

Регулирование величины шага обычно производится, исходя из эмпирических соображений, при этом весьма удобной является процедура [21]

$$\eta(k)=r^{-1}(k), r(k)=\alpha r(k-1)+\left\|\tilde{x}_k\right\|^2=\alpha r(k-1)+1. \quad (13)$$

Несложно заметить структурную схожесть алгоритмов обучения (3) и (12), (13).

Одной из особенностей карты Кохонена является наличие этапа кооперации в процессе самоорганизации, когда нейрон-победитель определяет локальную область топологического соседства, в которой возбуждается не только он сам, но и его ближайшее окружение, при этом более близкие к победителю нейроны возбуждаются сильнее чем удаленные. Эта топологическая область определяется функцией соседства φ_{ql}, зависящей от расстояния $D(w_q^K(k-1), w_l^K(k-1))$ между победителем $w_q^K(k-1)$ и любым из нейронов сети $w_l^K(k-1)$. Как правило, φ_{ql} – это ядерная функция симметричная относительно максимума в точке $D(w_q^K(k-1), w_q^K(k-1))=0$ и принимающая в ней единичное значение, монотонно убывающая с ростом расстояния и стремящаяся к нулю при $D(w_q^K(k-1), w_l^K(k-1))=2$. При этом наиболее часто в качестве функции соседства используются гауссиан, конус, функции Епанечникова, «мексиканская шляпа» и т. п.

Использование функции соседства приводит к модифицированному правилу обучения

$$\begin{cases}w_l^K(k)=\dfrac{w_l^K(k-1)+r^{-1}(k)\varphi_{ql}(\tilde{x}_k-w_l^K(k-1))}{\left\|w_l^K(k-1)+r^{-1}(k)\varphi_{ql}(\tilde{x}_k-w_l^K(k-1))\right\|}\ \forall l,\\ r(k)=\alpha r(k-1)+1,\end{cases} \quad (14)$$

реализующему принцип «победитель получает больше». При $\varphi_{ql}=\delta_{ql}=1$ при $q=l$ и 0 в остальных случаях, вновь приходим к алгоритму (12), обеспечивающему на каждом такте настройку единственного нейрона-победителя $w_q^K(k-1)$.

В принципе, для обучения самоорганизующейся карты можно вообще обойтись без нахождения победителя, а в качестве функции соседства использовать некоторую конструкцию, зависящую от выходных сигналов нейронов (5) так, что

$$\begin{cases}w_l^K(k)=\dfrac{w_l^K(k-1)+r^{-1}(k)\varphi(y_{kl})(\tilde{x}_k-w_l^K(k-1))}{\left\|w_l^K(k-1)+r^{-1}(k)\varphi(y_{kl})(\tilde{x}_k-w_l^K(k-1))\right\|}\ \forall l,\\ r(k)=\alpha r(k-1)+1.\end{cases} \quad (15)$$

Поскольку

$$y_{kl}=w_l^{KT}(k-1)\tilde{x}_k=\cos(\tilde{x}_k, w_l^K(k-1)), \quad (16)$$

то наиболее простой такой конструкцией является

$$\varphi(y_{kl})=\frac{1+y_{kl}}{2}, \quad (17)$$

удовлетворяющая всем условиям, предъявляемым к ядерным функциям соседства. Тогда окончательно алгоритм обучения карты Кохонена для нахождения центроидов *m* кластеров имеет вид

$$\begin{cases}w_l^K(k)=\dfrac{w_l^K(k-1)+r^{-1}(k)\frac{1+y_{kl}}{2}(\tilde{x}_k-w_l^K(k-1))}{\left\|w_l^K(k-1)+r^{-1}(k)\frac{1+y_{kl}}{2}(\tilde{x}_k-w_l^K(k-1))\right\|}\ \forall l,\\ r(k)=\alpha r(k-1)+1.\end{cases} \quad (18)$$

Нейросетевая архитектура, реализующая процесс кластеризации данных с пропусками, приведена на рис. 1.

Как видно, введенная нейронная сеть образована однотипными нейронами – адаптивными линейными ассоциаторами, настраиваемыми с помощью достаточно простых и очевидных алгоритмов обучения.

ЗАКЛЮЧЕНИЕ

Рассмотрена задача кластеризации данных, содержащихся в таблицах «объект-свойство» и имеющих пропуски, в режиме последовательного поступления этих данных на обработку. Предложена адаптивная нейросетевая система, позволяющая решать эту задачу в on-line режиме с постоянной коррекцией восстанавливаемых элементов таблицы и центроидов кластеров. Введенная нейросистема образована набором адаптивных линейных ассоциаторов, характеризуется высоким быстродействием и простотой численной реализации.

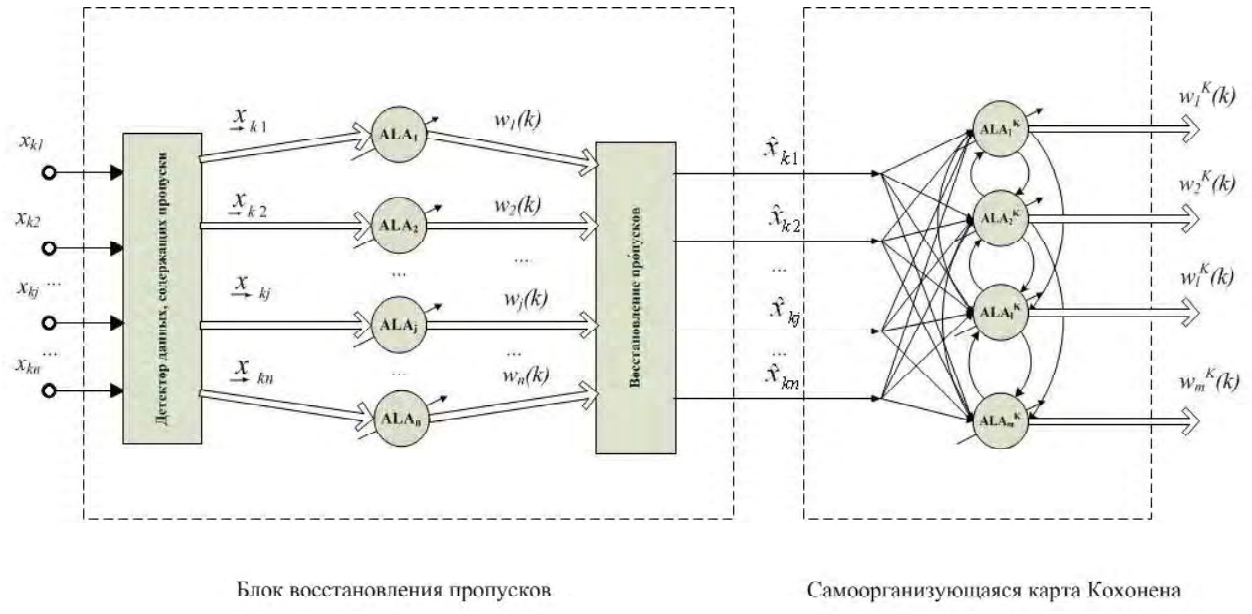


Рис. 1. Искусственная сеть для адаптивной кластеризации данных с пропущенными значениями

СПИСОК ЛИТЕРАТУРЫ

1. *Jain, A.* Data clustering: A review / A. Jain, M. Murty, P. Flynn // ACM Computing Surveys. – 1999. –№3(31). – P. 264–323.
2. *Дюк, В.* Data Mining / В. Дюк, А. Самойленко. – С. Пб. : Питер, 2001. – С. 368.
3. *Friedman, J.* The Elements of Statistical Learning. Data Mining, Inference and Prediction / J. Friedman, T. Hastie, R. Tibshirani. – Berlin : Springer, 2003. – P. 552.
4. *Xu, R.* Survey of clustering algoritms / R. Xu, D.II. Wunsch // IEEE Trans on Neural Networks. – 2005. – №3 (16). – P. 645–678.
5. *Han, J.* Data Mining: Concepts and Techniques / J. Han, M. Kamber. – Amsterdam : Morgan Kaufman Publ, 2006. – P. 754.
6. *Gan, G.* Data Clustering: Theory, Algorithms and Applications / G. Gan, Ch. Ma, J. Wu. – Philadelphia : SIAM, 2007. – P. 466.
7. *Abonyi, J.* Cluster Analysis for Data Mining and System Identification / J. Abonyi, B. Feil. – Basel : Birkhaeuser, 2007. – P. 303.
8. *Olson, D. Z.* Advanced Data Mining Techniques / D. Z. Olson, D. Dursun. – Berlin : Springer, 2008. – P. 180.
9. *Xu, R. II.* Clustering / R. Xu, D. C. Wunsch. – Hoboken, N. J. : John Wiley & Sons, Inc., 2009. – P. 341.
10. *Загоруйко, Н. Г.* Эмпирические предсказания. – Новосибирск : Наука, 1979. – С. 120.
11. *Gorban, A.* Principal Manifolds for Data Visualization and Dimension Reduction. Lecture Notes in Computational Science and Engineering, Vol. 58 / A. Gorban, B. Kegl, B. Wunsch, A. (Eds.) Zinovyev. – Berlin-Heidelberg – New York : Springer, 2007. – P. 330.
12. *Marwala, T.* Computational Intelligence for Missing Data Imputation, Estimation, and Management / Marwala T. // Knowledge Optimization Techniques. – Hershey – New York : Information Science Reference, 2009. – P. 303.
13. Maimon, O. Soft computing for Knowledge Discovery and Data Mining / O. Maimon, L. Rokach. – New York : Springer-Verlag, 2007. – P. 448.
14. *Rutkowski, L.* Computational Intelligence Methods and Techniques / L. Rutkowski. – Berlin-Heidelberg : Springer-Verlag, 2008. – P. 514.
15. *Bishop, C. M.* Neural Networks for Pattern Recognition. – Oxford : Clarendon Press, 1995. – P. 482.
16. *Gorban, A. N.* Neural network modeling of data with gaps / A. N. Gorban, A. A. Rossiev, D. C. Wunsch II //Радіоелектроніка, інформатика, управління. – 2000. – №1 (3). – С. 47–55.
17. *Tkacz, M.* Artificial neural networks in incomplete data sets processing / In: Eds. M. A. Klopotek, S. T. Wierzchon, K. Trojanowski // Intelligent Information Processing and Web Mining. – Berlin-Heidelberg : Springer – Verlag, 2005. – P. 577–583.
18. *Bodyanskiy, Ye.* An adaptive learning algorithm for a neuro-fuzzy network / Ye. Bodyanskiy, V. Kolodyazhniy, A. Stephan // Ed. B. Reusch «Computational Intelligence. Theory and Applications». – Berlin-Heidelberg – New York : Springer, 2001. – P. 68–75.
19. *Haykin, S.* Neural Networks. A Comprehensive Foundation / *Haykin, S.* – Upper Saddle River, N. J.: Prentice Hall, 1999 – P. 842.
20. *Kohonen, T.* Self-Organizing Maps / T. Kohonen. – Berlin: Springer-Verlag, 1995. – P. 362.
21. *Бодянский, Е. В.* Искусственные нейронные сети: архитектуры, обучение, применения / Е. В. Бодянский, О. Г. Руденко. – Х. : ТЕЛЕТЕХ, 2004. – С. 372.

Стаття надійшла до редакції 06.05.2011.

Шафроненко А. Ю., Волкова В. В., Бодянський Є. В.

АДАПТИВНА КЛАСТЕРИЗАЦІЯ ДАНИХ З ПРОПУЩЕНИМИ ЗНАЧЕННЯМИ.

Запропонована адаптивна нейромережева система, що дозволяє вирішувати задачу кластеризації даних з пропущеними значеннями в реальному часі з постійним коригуванням відновлюваних елементів таблиці і центроїдів кластерів. Впроваджена нейронна система характеризується високою швидкодією і простотою чисельної реалізації.

Ключові слова: адаптивна нейромережева система, відновлення пропусків, кластеризація.

Shafronenko A.Yu. Volkova V. V., Bodyanskiy Yev. V.

ADAPTIVE CLUSTERING WITH MISSING VALUES

In this paper the adaptive neural network system that solves the clustering problem of data with gaps is proposed. This system allows to process the data in the on-line mode with a constant correction of recoverable table's elements and centers of clusters. A proposed neural system has high speed and simple numerical realization.

Key words: adaptive neural network system, the restoration of gaps, clustering.

ПРОГРЕСИВНІ ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ

ПРОГРЕССИВНЫЕ ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

PROGRESSIV INFORMATICS TECHNOLOGIES

УДК 004.052

Брагина Т. И.[1], Табунщик Г. В.[2]

[1]Аспирант Запорожского национального технического университета
[2]Канд. техн. наук, доцент Запорожского национального технического университета

АНАЛИЗ ПОДХОДОВ К УПРАВЛЕНИЮ РИСКАМИ В ПРОГРАММНЫХ ПРОЕКТАХ С ИТЕРАТИВНЫМ ЖИЗНЕННЫМ ЦИКЛОМ

Выполнен обзор основных рисков программных проектов с итерационным жизненным циклом. Предложены рекомендации по управлению выявленными рисками в зависимости от выбранной модели разработки программных проектов. Выделены наиболее критичные риски и проведен их качественный анализ.

Ключевые слова: управление рисками, программные проекты, реакция на риск.

1. ВВЕДЕНИЕ

Разработка информационных систем связана с определенной совокупностью рисков – неопределенных событий или условий, наступление которых отрицательно или положительно сказывается на целях проекта. Каждый проект задумывается и разрабатывается на основании ряда гипотез, сценариев и допущений – факторов, которые для целей планирования считаются верными, реальными или определенными без привлечения доказательств. Неопределенность в рамках программного проекта (ПП) следует обязательно рассматривать в качестве потенциального источника возникновения рисков проекта, так как решение проектных задач управления в условиях неопределенности порождает риски выхода из бюджета и расписания, что ставит под угрозу достижимость целей ПП. Следовательно, процессы управления рисками являются наиболее важной компонентой процессов принятия решений и управления проектами.

2. ПОСТАНОВКА ЗАДАЧИ

Процессы управления рисками проекта включают в себя [1]:

– планирование управления рисками;
– идентификацию рисков;
– качественный анализ рисков;
– количественный анализ рисков;
– планирование реагирования на риски;
– мониторинг и управление рисками.

Эти процессы взаимодействуют как друг с другом, так и с процессами из других областей знаний. В зависимости от потребностей проекта в каждом процессе могут принимать участие один или несколько человек или групп. Каждый процесс имеет место, по крайней мере, один раз в ходе каждого проекта, а если проект разделен на фазы – то в одной или нескольких фазах проекта.

Исходными данными для выявления рисков является имеющаяся информация как об общих, так и о специфических для данного проекта рисках, связанная с бизнесом, технологиями, организациями и внешними условиями. Основным фактором, влияющим на возможные риски, является принятая в компании модель программного обеспечения (ПО), т. е. структура, определяющая последовательность выполнения и взаимосвязи процессов, действий и задач на протяжении жизненного цикла (ЖЦ). Существуют также дополнительные факторы, которым должно быть уделено внимание: имеющийся у команды опыт, применяемые внутри организации подходы к рискам, выраженные в виде правил и инструк-

ций, а также вся информация о проекте, известная на данный момент. После сбора необходимых данных проектная группа должна работать над заблаговременным выявлением рисков и создавать стратегии и планы по управлению ими, должны быть разработаны планы по решению проблем в случае их возникновения [2].

Для различных моделей ПО можно выделить основные, наиболее часто встречающиеся риски, характерные для каждой из них. Выделим наиболее распространенные модели, ориентированные на итеративный процесс разработки – Rational Unified Process (RUP), Microsoft Solutions Framework (MSF) и Agile (eXtreme Programming, Crystal, Feature Driven Development) [3].

Предвосхищение потенциальных проблем и заблаговременная подготовка четко составленных планов по борьбе с ними сокращает временные затраты в критических ситуациях и ограничивает негативный эффект, создаваемый рисками. Поэтому руководитель проекта, определив модель разработки ПО, которую будет использовать для проекта, должен осознавать сильные и слабые стороны данной модели, а также то, что управление рисками будет зависеть от характерных черт выбранной модели [4, 5].

Целью работы является разработка рекомендаций для управления рисками при использовании различных моделей разработки ПО. Для этого необходимо идентифицировать характерные риски разработки ПО, определить наиболее эффективные методы управления данными рисками, учитывая различия между моделями, выделить наиболее критичный риск для каждой модели.

3. ИДЕНТИФИКАЦИЯ РИСКОВ ПРОГРАММНЫХ ПРОЕКТОВ С ИТЕРАТИВНЫМ ЖИЗНЕННЫМ ЦИКЛОМ

Анализ литературы [6–10] позволяет выделить основные риски проектов по разработке ПО:

– внутренние нарушения календарного планирования;

– изменение требований;

– текучесть кадров;

– нарушение спецификаций;

– низкая производительность;

– недостаточное внимание к проекту со стороны руководства компании;

– отсутствие мотивации персонала компании.

Нарушения календарного планирования

Данный риск появляется из-за определенных нарушений (или полного несоответствия) процесса планирования бюджета времени и средств. Это можно рассматривать как ошибку собственно календарного планирования в противовес ошибкам осуществления проекта. Ошибка календарного планирования – не только реальный риск, но и самый крупный из пяти главных рисков по степени влияния на расхождение плана проекта и реального исполнения.

Ошибки календарного планирования можно рассматривать как тенденцию неверно судить о размерах продукта, который предстоит создать. В случае, когда календарное планирование строится без учета размера продукта, весьма вероятен перерасход времени на 50–80 %.

Статистика по IT-проектам показывает, что многие компании не выполняют предварительной оценки размера проекта, выбирая вместо этого календарное планирование от конца к началу или принимая желаемое за действительное. Те компании, которые прилагают серьезные усилия для определения размера продукта, могут сократить влияние ошибок календарного планирования до 15 %.

Изменение требований

Программное обеспечение, которое разрабатывается командой, всегда предназначено для того, чтобы соответствовать области деятельности клиента. Однозначно эта область не останется статичной за время создания программного обеспечения. Она будет изменяться со скоростью, диктуемой рынком и собственными темпами технологического развития. Разница между тем, что клиенты хотят в начале и в конце периода разработки возникает из-за изменений, которые произошли в данной области бизнеса за это время.

С точки зрения проекта, это изменение всегда является увеличением требований. Даже удаление того, что уже создано – своего рода увеличение, поскольку требует дополнительной работы.

Текучесть кадров

Возможность текучести кадров обычно не рассматривается в процессе оценки проекта, в то время как персонал представляет собой основной ресурс фирмы по разработке программных проектов. Текучесть кадров может привести к упущению прибыли, потере квалифицированных рабочих, снижению качества продукции и другим проблемам.

Нарушение спецификаций

Реализация риска нарушение спецификаций чаще всего является фатальной для проекта.

Нарушение спецификаций относится к краху процесса переговоров по установлению требований, которые идут в начале любого проекта. Когда существует глубокий конфликт, не позволяющий прийти к согласию, то часто результат маскируют. В итоге приходится столкнуться с отложенными проблемами, и конфликт разгорается вновь. В худшем случае это происходит на очень поздней стадии проекта, когда истрачены отведенные па проект ресурсы. Проект очень уязвим в этой стадии, и отказ любой из сторон поддерживать его может привести к быстрому прекращению разработки – процентный диапазон прекращения проектов из-за нарушения спецификаций составляет 10–15 % [6].

По отношению к данному риску может использоваться только политика ликвидации риска. Для рассмотрения проблемы неоднозначности, используемой для сокрытия разногласий, определим прекращение прений по поводу спецификаций как то, что все стороны подписа-

лись под входными и выходными граничными условиями проекта и определениями данных, вплоть до данных элементарного уровня из всех входящих и исходящих потоков данных или функций для создания данных. Хотя соглашение по потокам данных может быть только частью требуемого согласования, но это – ключевая часть. Поскольку описания данных менее склонны к неоднозначности, чем описания функций, заключаем, что отказ от претензий по входящим и исходящим потокам данных является хорошим показателем согласия. Когда такое согласие достигнуто, риск прекращения следует исключить из рассмотрения.

Низкая производительность

Различия между командами проектов в производительности в целом несколько сглажены и всегда меньше, чем индивидуальные различия. Более того, некоторые различия индивидуальной производительности возникают из-за четырех главных рисков, описанных выше. После устранения воздействия других рисков и распространения индивидуальных различий на команды можно определить, что этот фактор, как правило, сбалансирован: одинакова вероятность как позитивных, так и негативных изменений производительности. Сбалансированный риск (низкая или высокая производительность) лишь вносит шум в процесс. Он расширяет диапазон неопределенности без сдвига среднего ожидаемого показателя.

Существует множество причин (сопровождение действующих систем, повышение квалификации, участие в подготовке технико-коммерческих предложений, участие в презентациях, административная работа, отпуска, праздники, больничные), по которым участники проекта не смогут работать по проекту 100 % своего времени. При принятии риска рекомендуется планировать, что разработчики, которые назначены в проект на 100 %, будут реально работать над задачами проекта в среднем от 24 до 32 часов в неделю вместо 40 часов [11].

Недостаточное внимание к проекту со стороны руководства компании

Часть неудач по созданию ПО объясняется фрагментарностью подхода к процессу управления. Например, концентрируя свои усилия на технической стороне реализации, управляющий проектом уделяет недостаточное внимание проблемам управления человеческими ресурсами. Или, наоборот, техническая сторона контролируется не на достаточном уровне либо допускаются несоответствия в графике разработки.

Отсутствие мотивации персонала компании

Мотивация сотрудников занимает одно из центральных мест в управлении создания ПП. Руководитель организации должен сам выбирать каким образом стимулировать каждого работника для достижения целей организации. Если этот выбор сделан удачно, то руководитель получает возможность координировать усилия многих людей и сообща реализовывать потенциальные возможности группы разработчиков. В ином случае, отсутствие мотивации персонала приведет к срыву сроков, несоответствию необходимому качеству и надежности ПП, в крайнем случае прекращению разработки проекта.

4. ПЛАНИРОВАНИЕ РЕАКЦИИ НА РИСК ДЛЯ ПРОЕКТОВ С ИТЕРАЦИОННЫМ ПРОЦЕССОМ РАЗРАБОТКИ

По отношению к выявленным рискам возможны следующие действия [6]:

– ликвидация риска (например, за счет устранения причины);

– уменьшение риска (например, за счет использования дополнительных защитных средств);

– принятие риска (и выработка плана действия в соответствующих условиях);

– переадресация риска (например, путем заключения страхового соглашения).

Реакции на возникновение различных рисков и, соответственно, методы управления рисками будут варьироваться в зависимости от выбранной модели ПО.

В табл. 1 рассмотрены методы управления рисками для наиболее распространенных моделей ПО с итерационным циклом разработки – RUP, MSF и Agile [4], учитывающие особенности каждого типа проекта.

В рамках MSF управление рисками рассматривается как процесс их выявления, анализа и эффективной превентивной работы над ними. Эффективный процесс выявления и управления рисками помогает достичь разумных компромиссов между опасностями и открывающими возможностями. В случае управления проектом с большим количеством данных целесообразно использовать MSF, т.к. систематизация и структуризация информации в форме базы данных позволит упростить работу с ними. Но в то же время изменение требований потребует пересмотра структуры и формата базы данных, что повлечет большие трудовые и временные затраты.

RUP акцентирует внимание на планировании ЖЦ и отдельных итераций, управлении рисками, наблюдаемости хода проекта и метриках проекта. В RUP управление рисками означает определение и оценку рисков, принятие линии поведения, направленной на устранение, снижение вероятности риска, а также выбор действий на случай реализации риска. RUP опирается на планирование, что является сильной стороной модели, но подвергает проект повышенной угрозе возникновения ошибок календарного планирования.

Особенностью Agile является ожидание изменений, то есть в гибком процессе проектная команда не пытается зафиксировать требования в начале проекта и затем следовать жестко определенному плану. Изменения могут быть сделаны на сколь угодно позднем этапе проекта, что повышает возможность возникновения рисковых ситуаций. Для решения этих проблем в Agile чаще всего используют итеративную разработку с возможно более короткой продолжительностью итерации, опираясь на тщательный подбор разработчиков. Для работы по ме-

Таблица 1. Возможная реакция на риск в зависимости от типа программного проекта

Тип проекта / Тип риска	RUP	MSF	Agile
Нарушения календарного планирования, срыв сроков	Перерасчет сроков выполнения этапов работ	Планирование временного резерва	Выделение сверхурочных трудочасов, повышенное внимание к предварительному планированию
	Ликвидация или уменьшение данного риска возможно путем оценки длительности определенных работ или всего проекта несколькими методами: – опрос экспертов – проведение интервью соответствующих квалифицированных специалистов; – мозговой штурм – проведение общего собрания, на котором специалисты по очереди высказывают свои мнения о необходимых временных затратах на определенные типы работы. Споры и замечания не допускаются, все мнения записываются, группируются и оцениваются по типам и характеристикам работ; – сбор данных – проведение сбора данных по нескольким проектам относительно масштабов недооценки размера проекта; – имитационное моделирование – моделирование и анализ неопределенности в оценках основных показателей проекта (денежные и временные затраты).		
Изменение требований	учет увеличения трудоемкости и временных затрат в случае возможного роста требований, например, на 50 % (принятие риска)	переоценка проекта каждый раз, когда требования добавляются/изменяются (ликвидация риска)	подписание контракта с компенсацией затрат (переадресация риска заказчику)
Текучесть кадров	определение причин, по которым люди покидают компанию. Для изучения мотивов ухода большое значение имеет сбор и анализ информации о них. В первую очередь это сведения об общем числе уволившихся, долях сотрудников различных возрастных категорий, о работниках с низкой и высокой квалификацией, а также об их стаже работы и образовании	определение общих потерь времени как число месяцев, которое потребуется новому сотруднику на достижение того же уровня производительности, который был у замененного им работника	– учет в оценках трудоемкости издержек на обучение сотрудников; – уменьшение потерь от текучести кадров за счет привлечения на начальном этапе избыточного числа участников
Нарушение спецификаций	Политика ликвидации риска – подписание договора между заказчиком и компанией с описанием входных и выходных условий, оценка рисков независимыми экспертами		
Низкая производительность	улучшение организации производства и труда – повышение норм труда и расширения зон обслуживания; уменьшение числа сотрудников, не выполняющих нормы; упрощение структуры управления; повышение уровня специализации	структурные изменения – изменения удельных весов отдельных сотрудников в зависимости от трудоемкости выполняемой работы	повышение технического уровня производства – внедрение новых видов оборудования и программного обеспечения
Недостаточное внимание к проекту со стороны руководства компании	Проведение промежуточных отчетных собраний с приглашением заказчика	Промежуточная оценка прогресса разработки проекта руководством компании с составлением отчетности перед заказчиком	Составление договора между заказчиком и компанией с поэтапной сдачей проекта
Отсутствие мотивации персонала компании	Проведение тренингов, корпоративных собраний, внедрение методик team building	Премирование персонала в зависимости от успешности выполнения проектов	Распространение акций фирмы среди персонала, выделение разработчикам % от прибыли с проекта

тодологии Agile команда разработчиков должна состоять из высококвалифицированных специалистов, уход любого из которых наносит большой ущерб разработке проекта.

ВЫВОДЫ

Проведен качественный анализ рисков в зависимости от выбранной модели ПП и разработаны рекомендации по реагированию на выявленный риск.

Для RUP наиболее критичным оказался риск нарушения календарного планирования, для MSF – изменение требований, для Agile – текучесть кадров. Предложенные реакции позволят снизить негативное влияние рисков на конечные цели проектов и позволят выполнить проект в установленные сроки с заданным бюджетом.

СПИСОК ЛИТЕРАТУРЫ

1. Microsoft Solutions Framework. Дисциплина управления рисками MSF, вер. 1.1: [Электрон. ресурс]. – Режим доступа: http://www.microsoft.com/rus/msf.
2. PMBOK Руководство к Своду знаний по управлению проектами, 3-е изд. / PMBOK, Project Management Institute. – PMI, 2004. – 411 с.
3. *Груздо, И. В.* Повышение качества программного проекта за счет управления рисками / И. В. Груздо // Моделювання в економіці, організація виробництва та управління проектами. – 2009. – 1. – С. 141–146.
4. *ДеМарко, Т.* Вальсируя с медведями Управление рисками в проектах по разработке программного обеспечения / Том ДеМарко, Тимоти Листер, p.m. Office, 2005. – 190 с.
5. *Брагина, Т. И.* Сравнительный анализ итеративных моделей разработки программного обеспечения / Т. И. Брагина, Г. В. Табунщик // Радиоелектроніка, інформатика, управління. – 2010. – № 2. – С. 130–139.
6. *Bragina, T.* Comparative Analysis of Software Development Models for Electro-technical Systems / T. Bragina, G. Tabunshchik // Proc. of Int. Conf. on Modern Problem of Radio Engineering, Telecommunications and Computer Science TCSET'2010, February 19–23, 2010, Lviv–Slavsko, Ukraine. – P. 347.
7. *Брагина, Т. И.* Классификация моделей итеративной разработки программного обеспечения / Т. И. Брагина, Г. В. Табунщик // Системный анализ. Информатика. Управление : материалы Всеукраинской научно-практической конференции, САІУ–2010, Март 04–05, 2010, Запорожье, Украина. – Запорожье : КПУ, 2010. – С. 23–25.
8. *Галатенко, В. А.* Управление рисками: обзор употребительных подходов / Галатенко В.А. // Информационный бюллетень. – №11(162). – 2006. – С. 1–15
9. *Липаев, В. В.* Анализ и сокращение рисков проектов программных средств / В.В. Липаев // Jet Info. – 2005. – №1. – С. 1–36.
10. *Астахов, А.* Как управлять рисками информационной безопасности? / А. Астахов // CISA. – ноябрь 2006. – С.121 – 124.
11. *Петренко, С.* Методики и технологии управления информационными рисками / С. Петренко, С. Симонов // IT Manager. – 2003. – №3. – С. 57–61.

Стаття надійшла до редакції 09.11.2010.
Після доробки 11.03.2011.

Брагіна Т. І., Табунщик Г. В.

АНАЛІЗ ПІДХОДІВ ДО КЕРУВАННЯ РИЗИКАМИ В ПРОГРАМНИХ ПРОЕКТАХ З ІТЕРАЦІЙНИМ ЖИТТЄВИМ ЦИКЛОМ

Виконано огляд головних ризиків програмних проектів з ітераційним життєвим циклом. Запропоновані рекомендації з управління виявленими ризиками в залежності від обраної моделі розробки програмних проектів. Виділено найбільш критичні ризики та проведено їх якісний аналіз.

Ключові слова: керування ризиками, програмні проекти, реакція на ризик.

Bragina T. I., Tabunshchyk G.V.

ANALYSIS RISK MANAGEMENT APPROACHES IN SOFTWARE PROJECTS WITH AN ITERATIVE LIFECYCLE

In the article there is performed a review of the main risks in the software projects with an iterative lifecycle. The authors developed guidelines for the risks identified management, depended on the software development projects model. The most critical risks are identified and the qualitative analysis was made by the authors.

Key words: the risk management, the software projects, reaction on the risk.

УДК 004.75

Дьячук Т. С.

Асистент Запорожского национального технического университета

ОЦЕНКА ХАРАКТЕРИСТИК РАСПРЕДЕЛЕННОЙ СИСТЕМЫ

Выполнен обобщенный анализ распределенной системы с точки зрения систем массового обслуживания. Определены характеристики, которые могут быть исследованы с помощью систем массового обслуживания.

Ключевые слова: система массового обслуживания, Grid, кластер, ресурс, задача.

ВВЕДЕНИЕ

Оптимизация ресурсов является одной из важных задач в современном мире. Существует множество видов ресурсов: материальные, информационные, финансовые, интеллектуальные, трудовые, энергетические, времени и другие. Эффективное использование ресурсов

[1] в сфере высокопроизводительных вычислений не является исключением, так как вычислительная нагрузка на информационные системы постоянно растет. Получить максимальный результат при минимальных затратах стремятся как владельцы ресурсов, так и пользователи этих ресурсов. Для облегчения проектирования, контроля, планирования использования и анализа эффективности распределенной системы стремятся свести все измерители ресурсов к одному – например, денежному. При этом возникают трудности из-за отсутствия методов оценки некоторых видов ресурсов в денежном выражении. Поэтому при оценке эффективности использования ресурсов применяются несколько показателей, в совокупности отражающих уровень потребления ресурсов. В данной работе рассматривается распределенная система с помощью математического аппарата систем массового обслуживания. Данная модель является упрощенной в связи со сложностью реальных систем, но в целом позволяет получить характеристики, которые можно использовать для анализа свойств реальных распределенных систем.

ПОСТАНОВКА ЗАДАЧИ И ЦЕЛЬ РАБОТЫ

В настоящее время широко используется идея объединения ресурсов различных организаций в одну распределенную вычислительную систему в рамках Grid -проектов. В связи с чем возникает актуальная задача оптимального развития Grid, которая заключается в повышении производительности системы с одной стороны и минимизации затрат на сопровождение инфраструктуры с другой. В Grid -системе существуют два уровня планирования [2]: глобальный – распределение задач между кластерами (объединение машин в рамках одной организации) и локальный – распределение внутри кластера.

Пользовательские задачи распределяются между кластерами, учитывая информацию о задачах и текущем состоянии кластеров. Считаем, что одна задача выполняется на одном кластере, то есть не разбивается по нескольким. Подбирается кластер, с подходящими по характеристикам машинами и позволяющий решить задачу за минимальное время.

Анализируется запрос на выполнение задачи и рассылается всем кластерам, которые могут его выполнить. Каждый кластер на основании информации, содержащейся в запросе, своей текущей загрузки и уже составленного расписания возвращает время, до которого задача может быть выполнена. Принимается решение на основе этой информации и выбирается кластер, который удовлетворяет предельному времени выполнения задачи и имеет наименьшую стоимость (или удовлетворяет предельной стоимости и имеет минимальное время выполнения). Затем с выбранным кластером заключается соглашение и посылается задача на выполнение. Иерархическая система распределения задач имеет два уровня:

– распределение задач по кластерам (обслуживается брокерами);

– распределение многопроцессорных задач на кластере (обслуживается локальными планировщиками кластеров).

Миграция задач с одного кластера на другой также влияет на эффективность распределения. Задача может быть возвращена брокеру (с некоторым штрафом), если кластер получит больше выгодных задач от других брокеров или в случае сбоя. Брокер перераспределяет возвращенную задачу на другой кластер.

Выбор ресурсов и распределение вычислительной нагрузки осуществляется таким образом, чтобы удовлетворить все требования пользователей и оптимизировать общее время выполнения и стоимость используемых ресурсов.

В данной работе предложена упрощенная модель оценки характеристик кластера рабочих станций.

МОДЕЛЬ ЗАГРУЗКИ КЛАСТЕРА

Рассмотрим процесс решения задач на кластере с точки зрения систем массового обслуживания [3]. Загрузка и получение результатов на каждой рабочей станции представляют собой случайные события.

Будем считать, что вероятность получения результата на рабочей станции в малом промежутке времени Δt пропорциональна этому промежутку, то есть

$$\Delta p(t) = p(t+\Delta t) - p(t) = [1 - p(t)]\lambda\Delta t ,$$

где $p(t)$ – вероятность того, что за время t получено решение задачи; λ – коэффициент пропорциональности.

Выражение в квадратных скобках соответствует вероятности отсутствия решения в момент времени t.

При $\Delta t \to 0$ получим дифференциальное уравнение

$$dp(t) = [1 - p(t)]\lambda dt ,$$

откуда

$$\frac{dp(t)}{1-p(t)} = \lambda dt .$$

После интегрирования

$$\int_0^t \frac{dp(t)}{1-p(t)} = \int_0^t \lambda dt ,$$

получим

$$\ln[1-p(t)] = -\lambda t \;\text{ или }\; p(t) = 1 - e^{-\lambda t} .$$

Плотность распределения времени решения задачи соответственно равна

$$r(t) = \frac{dp(t)}{dt} = \lambda e^{-\lambda t} ,$$

то есть имеет экспоненциальный закон распределения.

В простейшем случае все рабочие станции имеют одинаковые характеристики, и плотности распределения

решения будут соответственно одинаковы. Величина λ отражает обобщенные характеристики рабочей станции и представляет собой интенсивность решения задач на станции.

Рассмотрим поток задач, поступающих на кластер. Будем считать вероятность поступления задачи в бесконечно малом промежутке времени Δt также пропорциональна величине этого промежутка, то есть

$$\Delta s(t) = s(t+\Delta t) - s(t) = [1 - s(t)]\mu\Delta t,$$

где $s(t)$ – вероятность того, что за время t получено решение задачи; μ – коэффициент пропорциональности.

Выражение в квадратных скобках соответствует вероятности отсутствия задач в момент времени t.

При $\Delta t \to 0$ получим дифференциальное уравнение

$$ds(t) = [1 - s(t)]\mu dt,$$

откуда

$$\frac{ds(t)}{1-s(t)} = \mu dt.$$

После интегрирования

$$\int_0^t \frac{ds(t)}{1-s(t)} = \int_0^t \mu dt,$$

получим

$$\ln[1-s(t)] = -\mu t \text{ или } s(t) = 1 - e^{-\mu t}.$$

Плотность распределения времени поступления задачи соответственно равна

$$v(t) = \frac{ds(t)}{dt} = \mu e^{-\mu t},$$

то есть также имеет экспоненциальный закон распределения.

На рис. 1 приведен граф состояний системы.

Рассмотрим состояния рабочих станций кластера:

S_0 – состояние, когда все рабочие станции загружены решением задач;

S_1 – состояние, при котором на одной из рабочих станций завершено решение;

…….

S_i – состояние, когда решение завершено на i рабочих станциях;

…….

S_n – состояние завершения решений на всех рабочих станциях.

Обозначим через $p_0(t), p_1(t), \ldots, p_i(t), \ldots, p_n(t)$ вероятности нахождения системы в соответствующих состояниях. Тогда можно записать следующие дифференциальные уравнения

$$\frac{dp_0(t)}{dt} = -\lambda_{0.1}p_0(t) + \mu_{1,0}p_1(t);$$

$$\frac{dp_1(t)}{dt} = \lambda_{0,1}p_0(t) - (\mu_{1,0} + \lambda_{1,2})p_1(t) + \mu_{2,1}p_2(t);$$

.....

$$\frac{dp_i(t)}{dt} = \lambda_{i-1,i}p_{i-1}(t) - (\mu_{i,i-1} + \lambda_{i,i+1})p_i(t) + \mu_{i+1,i}p_{i+1}(t);$$

.....

$$\frac{dp_n(t)}{dt} = \lambda_{n-1.n}p_{n-1}(t) - \mu_{n,n-1}p_n(t);$$

$$\sum_{i=0}^{n} p_i = 1.$$

В граничном стационарном режиме при $t \to \infty$ будем иметь систему алгебраических уравнений

$$0 = -\lambda_{0.1}p_0(t) + \mu_{1,0}p_1(t);$$

$$0 = \lambda_{0,1}p_0(t) - (\mu_{1,0} + \lambda_{1,2})p_1(t) + \mu_{2,1}p_2(t);$$

.....

$$0 = \lambda_{i-1,i}p_{i-1}(t) - (\mu_{i,i-1} + \lambda_{i,i+1})p_i(t) + \mu_{i+1,i}p_{i+1}(t);$$

.....

$$0 = \lambda_{n-1.n}p_{n-1}(t) - \mu_{n,n-1}p_n(t);$$

$$\sum_{i=0}^{n} p_i = 1.$$

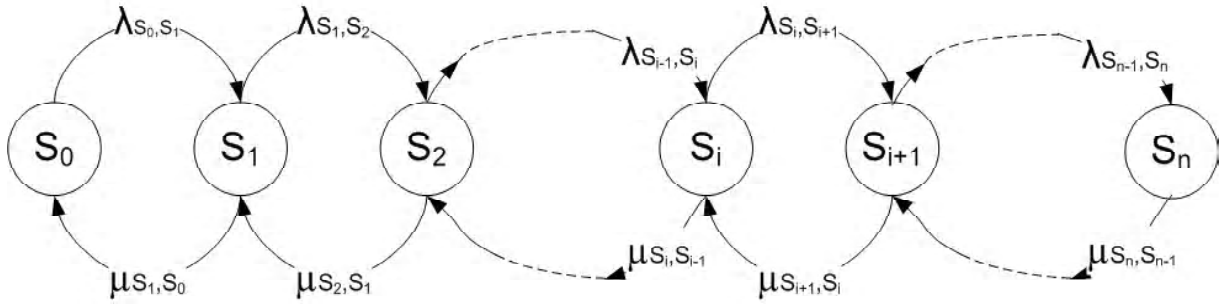


Рис. 1. Граф состояний системы

Для кластера, состоящего из n рабочих станций значения $\lambda_{i-1,i}$ и $\mu_{i,i-1}$ соответственно равны

$$\lambda_{i,i+1} = (n-i)\lambda; \quad \mu_{i+1,i} = (i+1)\mu, \quad i = 0, \ldots, n-1.$$

Ожидаемое количество свободных рабочих станций кластера $S_{\text{св}}$и средняя загрузка рабочих станций $S_{\text{заг}}$ равны соответственно

$$S_{\text{св}} = 1p_1 + 2p_2 + \ldots + np_n;$$
$$S_{\text{заг}} = np_0 + (n-1)p_1 + \ldots + p_{n-1}.$$

Справедливо также следующее утверждение $S_{\text{заг}} = n - S_{\text{св}}$, так как

$$S_{\text{св}} + S_{\text{заг}} = 1p_1 + 2p_2 + \ldots + (n-1)p_{n-1} + np_n +$$
$$+ np_0 + (n-1)p_1 + \ldots + p_{n-1} = n \cdot \sum_{i=0}^{n} p_i = n \cdot 1 = n.$$

Готовность системы к принятию задач для решения равна

$$G = p_1 + p_2 + \ldots + p_n = 1 - p_0.$$

Вероятность отказа в обслуживании:

$$P_{\text{отк}} = p_{0.}$$

Оценим значения вероятностей нахождения системы в различных состояниях. Для этого представим систему уравнений в матричном виде, убрав одно из равенств, в котором нет необходимости:

$$M \cdot P = I,$$

где

$$M = \begin{bmatrix} -n\lambda & \mu & 0 & \cdots & 0 & 0 & \cdots & 0 & 0 & 0 \\ n\lambda & -(n-1)\lambda - \mu & 2\mu & \cdots & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & (n-1)\lambda & -(n-2)\lambda - 2\mu & \cdots & 0 & 0 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & -(n-i+1)\lambda - i\mu & (i+1)\mu & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \cdots & (n-i+1)\lambda & -(n-i)\lambda - (i+1)\mu & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & (n-i)\lambda & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 2\lambda & -\lambda - (n-1)\mu & n\mu \\ 1 & 1 & 1 & 1 & 1 & 1 & \cdots & 1 & 1 & 1 \end{bmatrix}; P = \begin{bmatrix} p_0 \\ p_1 \\ p_2 \\ \vdots \\ p_{i-1} \\ p_i \\ p_{i+1} \\ \vdots \\ p_{n-1} \\ p_n \end{bmatrix}; I = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}. \quad (1)$$

Определим значение вероятности p_i с помощью определителей. Учитывая особенности матрицы, отметим следующее свойство определителей: определитель треугольной матрицы равен произведению диагональных элементов. Кроме того, покажем, что произведению диагональных элементов матрицы будет иметь определитель вида:

$$\Delta = \begin{vmatrix} a_{11} & b_{12} & 0 & \vdots & 0 & 0 & 0 & 0 & 0 & \vdots & 0 & 0 & 0 \\ 0 & a_{22} & b_{23} & \vdots & 0 & 0 & 0 & 0 & 0 & \vdots & 0 & 0 & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \vdots & \ldots & \ldots & \ldots \\ 0 & 0 & 0 & \vdots & a_{i-2,i-2} & b_{i-2,i-1} & 0 & 0 & 0 & \vdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \vdots & 0 & a_{i-1,i-1} & b_{i-1,i} & 0 & 0 & \vdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \vdots & 0 & 0 & a_{ii} & 0 & 0 & \vdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \vdots & 0 & 0 & 0 & a_{i+1,i+1} & 0 & \vdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \vdots & 0 & 0 & 0 & b_{i+2,i+1} & a_{i+2,i} & \vdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \vdots & 0 & 0 & 0 & 0 & b_{i+3,i} & \vdots & 0 & 0 & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & 0 & 0 & \vdots & 0 & 0 & 0 & 0 & 0 & \vdots & a_{n-2,n-2} & 0 & 0 \\ 0 & 0 & 0 & \vdots & 0 & 0 & 0 & 0 & 0 & \vdots & b_{n-1,n-2} & a_{n-1,n-1} & 0 \\ 0 & 0 & 0 & \vdots & 0 & 0 & 0 & 0 & 0 & \vdots & 0 & b_{n,n-1} & a_{nn} \end{vmatrix}. \quad (2)$$

Нетрудно видеть, что последовательное разложение определителей по элементам столбцов, начиная с последнего столбца до столбца $(i+1)$, которые представляют собой диагональные элементы, приведет к треугольной матрице, значение которой будет также произведение диагональных элементов. В результате получим:

$$\Delta = \prod_{i=1}^{n} a_{ii} .$$

Найдем определитель Δ матрицы M системы. Для этого преобразуем матрицу складыванием предыдущей строки с каждой последующей за исключением последней. В результате получим

$$\det \mathrm{M} = \Delta = \begin{vmatrix} -n\lambda & \mu & 0 & \cdots & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & -(n-1)\lambda & 2\mu & \cdots & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & -(n-2)\lambda & \cdots & 0 & 0 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & -(n-i+1)\lambda & (i+1)\mu & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & -(n-i)\lambda & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & -\lambda & n\mu \\ 1 & 1 & 1 & 1 & 1 & 1 & \cdots & 1 & 1 & 1 \end{vmatrix}.$$

Выполним разложение определителя по элементам нижней строки. Алгебраические дополнения первого и последнего элементов этой строки представляют собой треугольные матрицы, алгебраические дополнения остальных элементов представляют определители вида (2).

Алгебраическое дополнение элемента $(n+1)$ строки и $(n+1)$ столбца представляет собой треугольную матрицу $n \times n$, диагональ которой содержит все отрицательные элементы. Значение определителя будет иметь знак положительный, если n имеет четное значение, и отрицательный, если n – нечетное. С учетом значений элементов диагонали будем иметь $(-1)^n n!\lambda^n$.

Рассмотрим алгебраические дополнения элементов $(n+1)$ строки и i-го столбца. Знак алгебраического дополнения для элементов $(n+1)$ строки и i-го столбца будет определяться степенью $(-1)^{n+1+i}$. С другой стороны, определитель будет иметь вид матрицы (2), причем значения элементов диагонали столбцов от 1 до $(i-1)$ будет отрицательным, а значения элементов диагонали столбцов от i до n – положительным. Следовательно, знак определителя минора определится степенью $(-1)^{i-1}$. Отсюда знак алгебраического дополнения будет определяться произведением $(-1)^{n+1+i} \cdot (-1)^{i-1} = (-1)^{n+2i}$, то есть если n имеет четное значение, то положительный и отрицательный, если n – нечетное.

С учетом значений элементов диагонали будем иметь $(-1)^n \frac{n!}{i!}\lambda^{n-i} \frac{n!}{(n-i)!}\mu^i = (-1)^n n! C_n^i \lambda^{n-i}\mu^i$, где C_n^i – число сочетаний из n по i. Отсюда имеем

$$\Delta = (-1)^n n! \sum_{i=0}^{n} C_n^i \lambda^{n-i}\mu^i = (-1)^n n!(\lambda+\mu)^n,$$

где $\sum_{i=0}^{n} C_n^i \lambda^{n-i}\mu^i = (\lambda+\mu)^n$ согласно биному Ньютона.

При нахождении определителей Δ_i, у которых столбец i заменен на вектор свободных членов, который представлен 1 в $(n+1)$ строке и нулями для остальных строк, его значение и знак будет таким же, как и для алгебраического дополнения элементов $(n+1)$ строки и i столбца.

$$\Delta_i = \begin{vmatrix} -n\lambda & \mu & 0 & \cdots & 0 & 0 & \cdots & 0 & 0 & 0 \\ n\lambda & -(n-1)\lambda-\mu & 2\mu & \cdots & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & (n-1)\lambda & -(n-2)\lambda-2\mu & \cdots & 0 & 0 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & -(n-i+1)\lambda-i\mu & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \cdots & (n-i+1)\lambda & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 2\lambda & -\lambda-(n-1)\mu & n\mu \\ 1 & 1 & 1 & 1 & 1 & 1 & \cdots & 1 & 1 & 1 \end{vmatrix}.$$

Складывая предыдущую строку с каждой последующей, за исключением последней, получим

$$\Delta_i = \begin{vmatrix} -n\lambda & \mu & 0 & \cdots & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & -(n-1)\lambda & 2\mu & \cdots & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & -(n-2)\lambda & \cdots & 0 & 0 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & -(n-i+1)\lambda & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & -\lambda & n\mu \\ 1 & 1 & 1 & 1 & 1 & 1 & \cdots & 1 & 1 & 1 \end{vmatrix}.$$

После разложения по элементам столбца i получим

$$\Delta_i = (-1)^n n! C_n^i \lambda^{n-i} \mu^i .$$

Отсюда

$$p_i = \frac{\Delta_i}{\Delta} = C_n^i \frac{\lambda^{n-i}\mu^i}{(\lambda+\mu)^n} .$$

Оценим ожидаемое количество свободных станций $S_{\text{св}}$ для загрузки новых заданий:

$$S_{\text{св}} = \sum_{i=1}^{n} i C_n^i \frac{\lambda^{n-i}\mu^i}{(\lambda+\mu)^n} .$$

Так как $C_n^i = \frac{n}{i} C_{n-1}^{i-1}$, то можно записать

$$S_{\text{св}} = \sum_{i=1}^{n} i \frac{n}{i} C_{n-1}^{i-1} \frac{\lambda^{n-i}\mu^i}{(\lambda+\mu)^n} = \frac{n\mu}{(\lambda+\mu)} \sum_{i=1}^{n} C_{n-1}^{i-1} \frac{\lambda^{n-i}\mu^{i-1}}{(\lambda+\mu)^{n-1}}.$$

Выражение под знаком суммы представляет собой сумму вероятностей всех состояний системы для количества станций, равном (n–1), то есть равно единице. В результате получим

$$S_{\text{св}} = \frac{n\mu}{(\lambda+\mu)}.$$

Количество загруженных станций $S_{\text{заг}}$ равно

$$S_{\text{заг}} = n - S_{\text{св}} = n - \frac{n\mu}{\lambda+\mu} = \frac{n\lambda}{\lambda+\mu}.$$

Готовность системы для принятия новых задач определяется вероятностью

$$G = 1 - \frac{\lambda^n}{(\lambda+\mu)^n}.$$

Вероятность отказа в обслуживании:

$$P_{\text{отк}} = \frac{\lambda^n}{(\lambda+\mu)^n}.$$

Для любой системы массового обслуживания, в которой заявка может обслуживаться одним каналом, среднее число заявок, обслуживаемых в единицу времени, определяется как произведение числа занятых каналов на плотность потока обслуживания. В нашем случае, количество задач K, решаемых в единицу времени, определится произведением $S_{\text{заг}}$ на величину λ. Таким образом, получим

$$K = S_{\text{заг}}\lambda = \frac{n\lambda^2}{\lambda+\mu}.$$

Рассмотрим случай разбиения каждой задачи на подзадачи, которые могут выполняться параллельно на различных рабочих станциях. Пусть каждая задача представлена в виде m подзадач, имеющих примерно одинаковые характеристики. Среднее время решения подзадачи будет в m раз меньше и, следовательно, интенсивность ее решения $\lambda_{n/з}$ будет в m раз больше, то есть

$$\lambda_{n/з} = m\lambda .$$

Поток $\mu_{n/з}$ поступающих подзадач также увеличится в m:

$$\mu_{n/з} = m\mu .$$

Количество подзадач, решаемых в единицу времени определится выражением

$$K_{n/з} = \frac{n\lambda_{n/з}^2}{\lambda_{n/з}+\mu_{n/з}} = \frac{n(m\lambda)^2}{m\lambda+m\mu} = \frac{nm\lambda^2}{\lambda+\mu} .$$

Если в кластере одновременно решается M задач, имеющих одинаковые приоритеты, то среднее количество подзадач, относящихся к одной задаче, решаемых в

единицу времени, равно

$$k = \frac{K_{n/з}}{M} = \frac{nm\lambda^2}{(\lambda+\mu)M}.$$

Среднее время решения задачи равно

$$\bar{t} = \frac{m}{k} = \frac{m(\lambda+\mu)M}{nm\lambda^2} = \frac{(\lambda+\mu)M}{n\lambda^2}.$$

Если поступление на решение новых подзадач рассматривать как следствие получения очередного решения, то $\lambda = \mu$. Таким образом, поток запросов на кластер меняется во времени в зависимости от интенсивности решения задач. Чем быстрее решаются задачи, тем сильнее увеличивается поток задач на глобальном уровне. Если есть простаивающие станции, то поток задач увеличивается. Среднее время на решение задачи определится выражением

$$\bar{t} = \frac{2M\lambda}{n\lambda^2} = \frac{2M}{n\lambda}.$$

Отсюда вытекает очевидный факт, что ожидаемое время на решение задачи прямо пропорционально количеству задач в системе и обратно пропорционально количеству рабочих станций в кластере.

Пусть каждая задача представлена различным количеством подзадач m_i, тогда общее количество подзадач равно

$$m_\Sigma = \sum_{i=1}^{M} m_i.$$

Будем считать, что подзадачи имеют одинаковые характеристики с точки зрения времени их решения, то есть интенсивность решения одинакова:

$$\forall (i = \overline{1, M})(\frac{\lambda_i}{m_i} = \lambda_{n/з} = \text{const}),$$

где λ_i – интенсивность решения задачи i, величина обратная ожидаемому времени решения.

В этом случае среднее количество подзадач, решаемых в единицу времени для каждой задачи и определится стратегией планирования. Для случая, когда планирование осуществляется между подзадачами независимо от того, к каким задачам они относятся, среднее количество подзадач, решаемых в единицу времени, распределится пропорционально количеству подзадач в каждой задаче:

$$k_i = \frac{K_{n/з} m_i}{m_\Sigma},$$

а ожидаемое время решения задачи равно

$$\bar{t}_i = \frac{m_i}{k_i} = \frac{m_i m_\Sigma}{K_{n/з} m_i} = \frac{m_\Sigma}{K_{n/з}}.$$

Для данного случая время решения каждой задачи одинаковое.

Для кластера ожидаемое количество загруженных рабочих станций в этом случае будет иметь вид

$$S_{\text{заг}} = \frac{n\lambda_{n/з}}{\lambda_{n/з} + \mu_{n/з}},$$

а среднее количество подзадач, решаемых в единицу времени равно

$$K_{n/з} = S_{\text{заг}}\lambda_{n/з} = \frac{n\lambda_{n/з}^2}{\lambda_{n/з} + \mu_{n/з}}.$$

Если планирование осуществляется между задачами независимо от их объема (одни и те же ресурсы выделяются каждой задаче), то среднее время решения i-той задачи равно

$$\bar{t}_i = \frac{m_i}{K_{n/з}} = \frac{m_i(\lambda_{n/з} + \mu_{n/з})}{n\lambda_{n/з}^2}.$$

Из полученного выражения следует, что более короткие задачи будут решаться быстрее.

Оценим стоимость выполнения i-ой задачи

$$\text{Cost}_i = \bar{t}_i \cdot g,$$

где g – стоимость использования ресурсов кластера в единицу времени.

ВЫВОДЫ

В работе предложена упрощенная модель оценки характеристик кластера рабочих станций. Данная модель отражает основные закономерности работы многопроцессорной системы. Результаты исследований могут быть применены администраторами при настройке оборудования кластерных систем для повышения эффективности их использования. Система массового обслуживания аналогичного типа приведена в [3], только в нашем случае применена другая трактовка и расширены исследуемые характеристики системы.

Часто кластеры строятся из однородных элементов, поэтому принято допущение, что все рабочие станции кластера имеют приблизительно одинаковые характеристики. Если рассматривать случай, когда кластер строится на основе компьютеров с разными характеристиками, то количество состояний резко возрастает, и системы уравнений становятся очень громоздкими, что не позволяет получить закономерности в общем случае.

В будущей работе планируется исследовать поступление задач с разными характеристиками, а также рассмотреть возможность назначения приоритетов. Дальнейшее развитие связано также с объединением кластеров на глобальном уровне для изучения их взаимодействия.

СПИСОК ЛІТЕРАТУРИ

1. *Дьячук, Т. С.* Дослідження ефективності роботи алгоритму планування ресурсів для розподіленої системи / Т. С. Дьячук, Р. К. Кудерметов, А. М. Літава, Р. І. Сметанін // Вісник КДПУ імені Михайла Остроградського. – Кременчук : КДУ, 2010. – Вип. 1/2010 (60) частина 1. – С. 47–50.
2. Эвристики распределения задач для брокера ресурсов Grid [Электронный ресурс] / А. И. Аветисян, С. С. Гайсарян, Д. А. Грушин, Н. Н. Кузюрин, А. В. Шокуров // Труды Института системного программирования РАН – *2004*. – Режим доступа: http://citforum.univ.kiev.ua/nets/digest/grid/index.shtml – Загл. с экрана.
3. *Клейнрок, Л.* Теория массового обслуживания. Пер. с англ. / Клейнрок, Л. ; пер. И. И. Грушко; ред. В.И. Нейман. – М. : Машиностроение, 1979. – 432 с.

Стаття надійшла до редакції 17.03.2011.

Дьячук Т. С.

ОЦІНКА ХАРАКТЕРИСТИК РОЗПОДІЛЕНОЇ СИСТЕМИ

Виконано узагальнений аналіз розподіленої системи з точки зору систем масового обслуговування. Визначено характеристики, які можуть бути досліджені за допомогою систем масового обслуговування.

Ключові слова: система масового обслуговування, Grid, кластер, ресурс, завдання.

Diachuk T. S.

EVALUATION OF DISTRIBUTED SYSTEM'S CHARACTERISTICS

The distributed system in terms of queueing systems is analyzed. The characteristics that can be studied using queuing systems are presented.

Key words: queueing system, Grid, cluster, resource, task.

УДК 004.9:004.82

Гонтарь Н. А.[1], Кудерметов Р. К.[2]

[1]Ассистент Запорожского национального технического университета
[2]Канд. техн. наук, профессор Запорожского национального технического университета

РАЗРАБОТКА ОНТОЛОГИИ СИСТЕМНОГО ИНЖИНИРИНГА КОСМИЧЕСКИХ СИСТЕМ

В статье обоснована необходимость создания онтологии системного инжиниринга (СИ). Онтология СИ вместе с переходом к моделе-ориентированному СИ повысит эффективность разработки сложных систем. Предложен прототип онтологической модели, основанный на дескриптивной логике и формализованный с помощью языка OWL DL.

Ключевые слова: системный инжиниринг, онтология, MBSE, дескриптивная логика, OWL.

ВВЕДЕНИЕ

Согласно определению Международного Совета по Системной Инженерии (International Council on Systems Engineering, INCOSE) системный инжиниринг (СИ) – это междисциплинарный подход и средства для создания успешных систем [1, 2]. На протяжении всего жизненного цикла (ЖЦ) систем (концептуальное проектирование, разработка, изготовление, испытания, эксплуатация и утилизация) СИ фокусирует внимание разработчиков на глубоком анализе и отслеживании потребностей пользователей создаваемой системы и функциональных требованиях к ней. СИ интегрирует дисциплины и коллективы разработчиков в единое распределенное виртуальное пространство проекта, в котором согласованы процессы разработки и изготовления системы, взаимодействия разработчиков, применяемые ими понятия, правила, методы, инструменты и т. д.

СИ – итерационный процесс и каждое новое, принятое в ходе ЖЦ системы, решение отражается на конечном результате разработки. Поэтому важным является создание на основе современных информационных технологий благоприятной среды, отражающей текущие процессы СИ, множество моделей системы, возможные варианты решений и ошибки.

В одной из последних инициатив INCOSE – «Systems Engineering Vision 2020» декларируются принципы моделе-ориентированного подхода к СИ, который получил название MBSE (Model-based Systems Engineering) [3, 4]. MBSE – это формализованное применение моделирования для поддержки системных требований, разработки, анализа, верификации и валидации систем на фазе концептуального проектирования и далее при разработке и последующих фазах ЖЦ системы. Основными задачами MBSE являются:

– обеспечить согласованное взаимодействие между заказчиками и разработчиками систем, а также между коллективами разработчиков;

– уменьшить риски и неопределенности и контролировать их в течение всего ЖЦ системы;

– поддерживать процессы контроля границ системы, ее агрегации, декомпозиции, интеграции, сертификации и прогнозирования функциональных возможностей;

– обеспечить согласованность множества системных аспектов, видений и моделей системы;

– обеспечить обозримость и возможность отслеживания состояния разработки больших и сложных систем;

– за счет ясного, определенного представления системы в виде моделей, устранять неоднозначности при участии разработчиков с разными культурами и языками и использовании ими различных средств проектирования;

– обеспечить необходимое понимание вклада всех составных частей системы в ее миссию.

В качестве языка моделирования для СИ и MBSE перспективным считается язык SysML, построенный на базе языка UML и принятый как индустриальный стандарт консорциумом Object Management Group (OMG) в 2007 г. [5]. Язык SysML включает не только все возможности языка UML моделировать программные системы, но и дополняет его диаграммами, необходимыми для моделирования любых технических систем, в частности диаграммами требований, блоков (рис. 1) и параметрической диаграммой. Однако следует отметить, что SysML включает много избыточных и практически неиспользуемых диаграмм, а также, из-за неоднозначности некоторых конструкций и наличия в спецификации неформальных описаний, язык не всегда позволяет точно выразить семантические особенности системы.

ПОСТАНОВКА ЗАДАЧИ

В настоящее время интенсивно создаются онтологии различных предметных областей (ПрО) и исследуются возможности их использования в MBSE. В научных и промышленных сообществах системных инженеров сформировалось видение развития MBSE и интеграции его с онтологическим подходом, которое предполагает последовательное развитие онтологий от прототипов до стандартов (рис. 2) [6].

Для успешного перехода от документо-ориентированного СИ к моделе-ориентированному СИ (т. е. к MBSE) и эффективного его использования в различных ПрО необходимо создание множества моделей самого MBSE и охватываемых им процессов. В частности, для однозначного понимания командами разработчиков этапов, процессов и задач MBSE, правильной обработки его артефактов приложениями необходимо создание машинно-интерпретированных представлений всех составляющих СИ и MBSE. Решению этой задачи будет способствовать создание онтологий СИ и MBSE. В данной работе ставится цель обосновать возможность создания таких онтологий.

ОНТОЛОГИЯ СИСТЕМНОГО ИНЖИНИРИНГА

Онтология формальная спецификация разделяемой концептуальной модели. Здесь под «концептуальной» моделью подразумевается абстрактная модель, описывающая систему понятий ПрО. «Разделяемая» модель означает согласованное ее понимание определенным сообществом (группой людей). «Спецификацией» является описание системы понятий в явном виде. «Формальность» модели подразумевает машиночитаемое представление данных [7].

Создание онтологии СИ позволит получить в распоряжение системных инженеров такую модельную страту СИ, которая объединит совокупность понятий, правил, отношений, данных и информации СИ, независимых от отдельных стандартов и терминов и не скрытых в программном коде. Если в MBSE SysML обеспечивает синтаксическую совместимость моделей процессов и систем, то онтология – их семантическую интероперабельность.

Онтология СИ не будет нести в себе новых знаний, но она формализует наработанные знания и явные заключе-

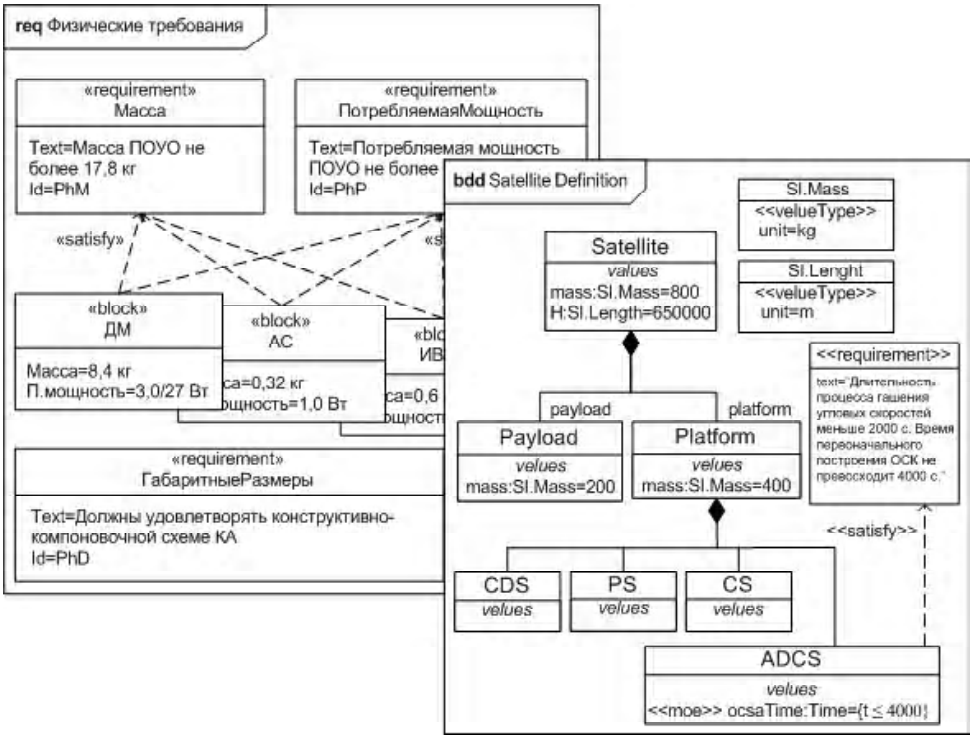


Рис. 1. Диаграммы требований и блоков для модели КС на языке SysML

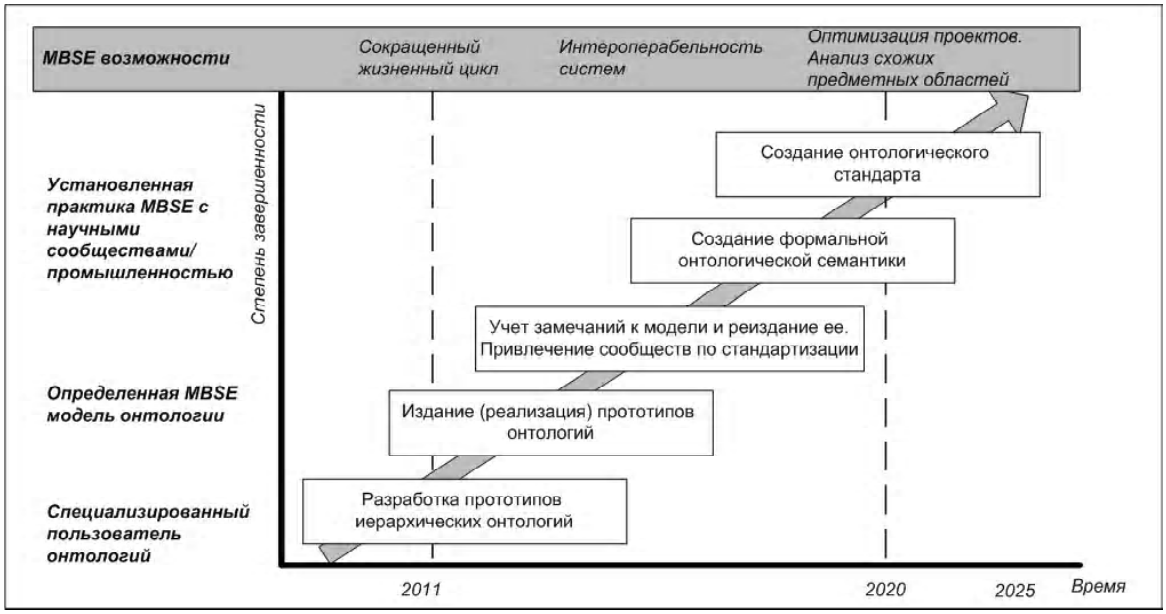


Рис. 2. Место онтологии в развитии СИ

ния и обеспечит их согласованность. Эти знания будут элементарными для восприятия системными инженерами и интероперабельными в распределенных компьютерных систем. Возможно, первые экземпляры онтологии СИ будут избыточными, громоздкими и противоречивыми в силу большого множества дисциплин самого СИ и дисциплин, которыми он управляет на протяжении ЖЦ системы.

В настоящее время имеются огромный международный опыт и знания в области создания космических систем (КС) и, поэтому становится вопрос об эффективном хранении и обработке этих знаний. Онтологическое представление СИ КС позволит специфицировать структуру и семантику терминов двух предметных областей, выразить различные формы сложных ограничений целостности и будет поддерживать работу с распределенными ресурсами, формализованную базу единой терминологии для разработчиков, автоматическую обработку запросов, интеграцию ее или в нее других специфицированных формализованных прикладных ПрО. На рис. 3 показаны возможные области использования онтологий и в частности онтологии СИ КС .

Определение характеристик КС обычно идет от функционального описания к физическому. Физическое описание формализуется в дереве продукта, которое представляет собой структуру, исходящую из упорядоченного и исчерпывающе разбитого на составные части конечного продукта, представленного в виде последовательности уровней, на которых расположены его составляющие [8, 9]. Эти составляющие конечного продукта меняют свою сущность по мере разбиения от верхнего уровня до нижних уровней декомпозиции. Как правило, они становятся менее сложными функционально, более компактными и самодостаточными в физическом смысле, включая при этом меньшее число используемых технологий. Поэтому онтология СИ КС должна отражать не только процессы СИ, но и техническую и структурную декомпозиции КС.

Разработку принципов СИ и MBSE для КС осуществляют организации NASA, ECSS, INCOSE, OMG и др. Стандарты, спецификации и рекомендации этих организаций должны стать источником информации для формирования понятий онтологии СИ КС. В настоящее время работы по инженерии знаний и онтологии в области КС проводятся NASA в рамках проекта NExIOM (NASA Exploration Information Ontology Model) [10]. Проект NExIOM разрабатывается с целью обеспечения:

– возможности отслеживать все изменения системы и ее элементов на протяжении всего ЖЦ (трассируемость);

– возможности проводить полнофункциональный анализ системы;

– поддержки интерпретации и доступа к информации, как со стороны разработчиков, так и программных агентов.

Данный проект должен регулировать техническую, экономическую и организационную части процесса разработки систем, вплоть до стандартизации документооборота и рабочего времени инженеров. NExIOM нацелен на создание модели, которая будет содержать данные и знания о существующих стандартах хранения, организации и обработки данных, методах и средствах моделирования, опыте предыдущих успешных разрабо-

Рис. 3. Область применения онтологии СИ КС

ток систем. Предполагается, что стандарты и спецификации NExIOM будут развиваться во времени с учетом потребностей разработчиков. Одной из целей рабочей группы проекта NExIOM является поиск экономически эффективных способов обмена информацией между NASA и промышленностью и ожидается, что онтологии будут играть в этом ключевую роль.

Эффективность создания онтологий возрастает за счет того, что онтологии ПрО могут разрабатываться независимо, а затем проходить процессы согласования и адаптации между собой. Поэтому, учитывая, что проект NExIOM направлен на организацию и формализацию знаний всего процесса создания КС, целесообразно создавать онтологии отдельных дисциплин этого процесса, в частности СИ КС, как дисциплины и процесса, интегрирующего множество других дисциплин и команд разработчиков на протяжении всего ЖЦ систем.

В качестве инструмента для создания онтологии СИ КС предполагается использовать программную среду проектирования онтологий *Protégé*, которая включает редактор онтологий, позволяющий проектировать онтологию, разворачивая таксономическую структуру абстрактных или конкретных классов и слотов. Структура онтологии создается аналогично иерархической структуре каталога. На основе сформированной онтологии *Protégé* может генерировать формы получения знаний для введения экземпляров классов и подклассов. *Protégé* позволяет вводить в онтологию классы методом заполнения форм, составленных из элементов мета-описания.

Одним из важных шагов при проектировании онтологии является распределение классов в таксономию. Смысловые подклассы одного и того же класса в онтологии должны находиться на одном уровне. Подклассов должно быть не меньше двух и не больше двенадцати, иначе, по мнению разработчиков *Protégé*, можно выделить еще один подкласс.

Основные концепты (понятия) СИ КС будут представлять собой объекты онтологии, реализованные как классы с характерными атрибутами. Такой способ представления информации позволяет работать как с СИ в целом, так и с ее структурными элементами в отдельности. Мы используем способ нисходящей разработки, который начинается с определения самых общих понятий ПрО СИ КС с последующей их структуризацией. Наиболее фундаментальные понятия СИ КС будут классами, которые находятся в верхней части таксономического дерева, корнем которого является класс «*THING*», охватывающий все множество концептов онтологии.

На рис. 4 показан простейший пример, демонстрирующий возможность создания онтологии СИ КС с помощью *Protégé*. В качестве источников данных для этого примера использовались основные понятия, отношения между ними и атрибуты, изложенные в стандартах ECSS, NASA, INCOSE и публикациях, посвященных разработке КС.

Не существует единственного «правильного» способа или методологии разработки онтологий, каждый аналитик составляет план и этапы онтологического анализа, опираясь на свой опыт и существующие проекты онтологий. Тем не менее, можно выделить некоторые фундаментальные правила разработки онтологии [11]:

– необходимо рассматривать множество альтернативных способов моделирования и моделей ПрО. Лучшее решение почти всегда зависит от предполагаемого приложения и ожидаемых расширений;

– разработка онтологии – это обязательно итеративный процесс;

– понятия в онтологии должны быть близки к объектам (физическим или логическим) и отношениям в интересующей нас ПрО. Скорее всего, это существительные (объекты) или глаголы (отношения) в предложениях, которые описывают данную ПрО.

ФОРМАЛИЗАЦИЯ ОНТОЛОГИИ СИ С ПОМОЩЬЮ ДЕСКРИПТИВНОЙ ЛОГИКИ

Для выражения фундаментальных понятий при формализации данных о сложных системах и их концептуальном моделировании целесообразно использовать дескриптивные логики (ДЛ). ДЛ состоят из двух уровней: интенсионального и экстенсионального. Интенсиональный уровень определяет множество понятий и отношений, описывающих концептуальную структуру ПрО (классы, свойства, отношения, аксиомы). Экстенсиональный уровень содержит множество экземпляров элементов, описанных в интенсиональном уровне [12]. С помощью ДЛ определяются основные элементы онтологии, которые позволяют отделить знания о ПрО от оперативных знаний и опыта.

Существует несколько диалектов дескриптивной логики, которые отличаются выразительными возможностями. Для описания онтологии СИ КС выбран диалект AL, который считается базовым для ДЛ и содержит необходимые и достаточные конструкции для описания созданного безэкземплярного прототипа онтологии, в котором имеются: $\top$ – концепт «*THING*»; $\bot$ – концепт «*NOTHING*»; $\cap$ – логическая связка (конъюнкция); $\exists$ – квантор существования; $\forall$ – квантор ограничение значения; $\neg$ – логическая связка (отрицание).

Концепты онтологии определяются индуктивно следующим образом:

– все имена концептов, $\top$ и $\bot$ являются AL-концептами;

– если A является AL-концептом, то и $\neg A$ является AL-концептом;

– если $A0$ и $A1$ являются AL-концептами, а r – имя роли, то $A0 \cap A1$, $\exists r.A0$, $\exists r.A1$, $\forall r.A0$, $\forall r.A1$ являются AL-концептами.

Онтологическая модель описывается унарными и бинарными предикатами. Унарные предикаты представляют собой фундаментальные понятия и соответствуют классам, которые находятся в корне таксономического дерева. Например, для онтологии СИ КС определены унарные предикаты: *«SYSTEMS – ENGINEERING»* концепт абстрактных классов онтологии, *«REQUIREME NT –*

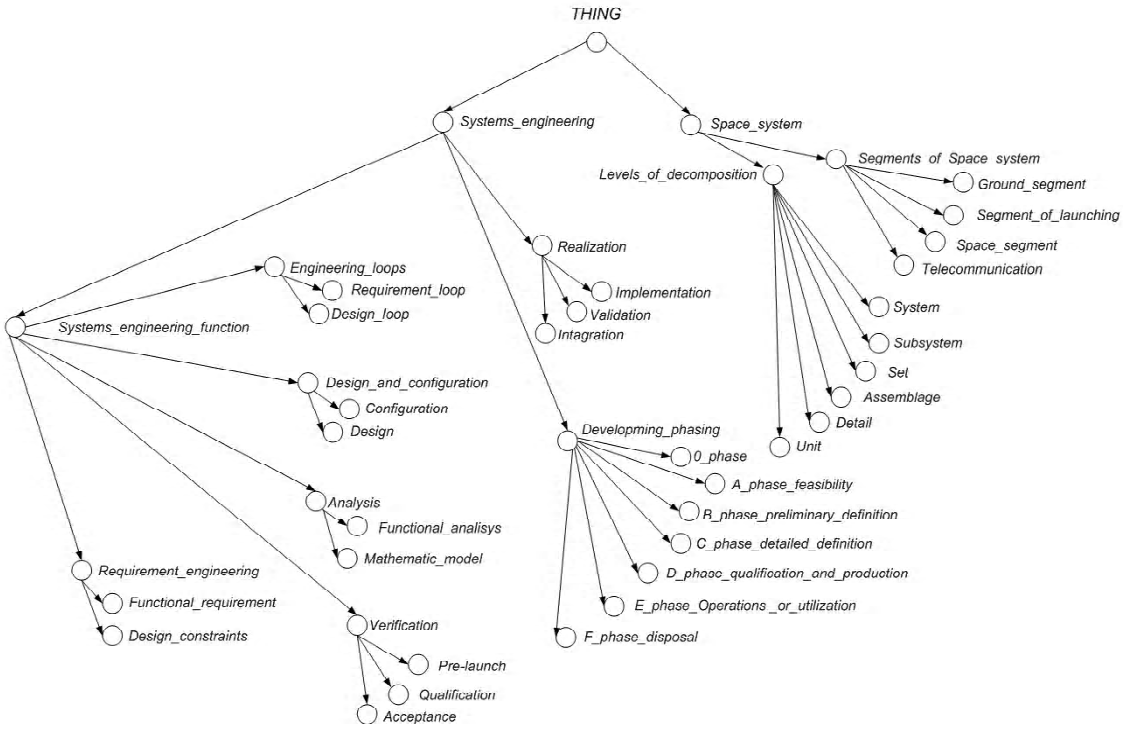


Рис. 4 . Таксономия классов онтологии СИКС

ENGINEERING» выделенное понятие ПрО СИ, определенное как класс онтологии.

Бинарные предикаты – это роли. *«hasIns tan ce»* бинарный предикат, определяющий понятие того, что с концептом, стоящим справа от роли, имеет место существование такого отношения *«hasIns tan ce.A»* .

С помощью предикатов в онтологиях строятся отношения и аксиомы. Например, следующая формула показывает отношение «*существование концепта, определенного как экземпляр класса FUNCTIONAL – REQUIREMENT*»:

$$\exists hasIns\,\tan ce.FUNCTIONAL - REQUIREMENT \quad (1)$$

В онтологиях всегда предопределены два класса «*THING*» и «*NOTHING*» . «*THING*» это множество всех концептов онтологии. «*NOTHING*» это пустое множество концептов. Таким образом, если мы введем концепт «*SYSTEMS – ENGINEERING*» , то он будет экземпляром класса :

$$SYSTEMS - ENGINEERING \exists \text{ hasInstance.T} \quad (2)$$

Для части онтологии, представляющей онтологию дисциплины «инженерия требований», СИ КС в качестве концептов или унарных предикатов и примем и , которые будут выделенными понятиями СИ и определенными как классы онтологии.

В самом общем случае, терминологические аксиомы первого рода (включений) (3) и второго рода (равенств) (4) имеют вид:

$$A0 \sqsubseteq A1,\ r0 \sqsubseteq r1; \quad (3)$$

$$A0 \equiv A1,\ r0 \equiv r1. \quad (4)$$

Для онтологии СИ КС с помощью аксиом первого рода определим таксономию концептов онтологии. Например, класс *«REQUIREMENT – ENGINEERING»* является подклассом *«SYSTEMS – ENGINEERING»*, а класс , в свою очередь, является надклассом класса :

$$«REQUIREMENT - ENGINEERING» \sqsubseteq$$
$$\sqsubseteq «SYSTEMS - ENGINEERING», \quad (5)$$

$$«FUNCTIONAL - REQUIREMENT» \sqsubseteq$$
$$\sqsubseteq «REQUIREMENT - ENGINEERING». \quad (6)$$

Таким образом, ДЛ позволяет описать модель онтологии системного инжиниринга с точки зрения математики, создать устойчивые непротиворечивые формулировки термов онтологии. ДЛ обладает мощным аппаратом формализации понятий и при дальнейшей работе с онтологией позволяет рассматривать все элементы как с точки зрения их семантики, так и со стороны логических законов.

Несмотря на существующее множество технологий и моделей хранения, интерпретации и обработки данных, онтология усиливает возможности концептуального моделирования. Работа с онтологической моделью реализует абстрактное, единое, семантическое моделирование ПрО (интенсиональный уровень), независящее от спецификации этой модели (экстенсиональный уровень) и служащее в качестве средства создания эталонного инструмента взаимодействия разработчиков между собой и между программными приложениями.

Следует учитывать, что создание онтологии требует экспертных знаний в исследуемой ПрО и значительных

затрат времени. Онтология содержит формализованные данные, но не отображает методы, функции и процессы ПрО, поэтому онтология СИ будет являться лишь универсальным источником данных о ПрО СИ.

OWL-ОНТОЛОГИЯ СИ

Для формального машинного представления онтологий Консорциум W3C предложил язык описания онтологий OWL (Ontology Web Language), который имеет различные по выразительности диалекты, спроектированные для использования отдельными сообществами разработчиков и пользователей [13].

Диалект OWL DL наиболее популярен и используется при необходимости представления знаний о ПрО в максимально выразительном формализованном виде без потери полноты вычислений (все заключения гарантировано будут вычисляемыми) и разрешимости (все вычисления гарантированно завершатся за определенное время). Формальной основой онтологий на языке OWL DL является ДЛ, т.е. логика предикатов, предназначенная для представления терминологического знания о ПрО. OWL DL включает все языковые конструкции OWL с ограничениями вроде разделения типа (класс не может быть частным свойством, а свойство не может быть индивидом или классом).

Каждый индивид в OWL представлении является членом класса `owl:Thing`. Таким образом, каждый определенный класс автоматически является подклассом `owl:Thing`. Специфичные для данной ПрО корневые классы (понятия) определяются простым объявлением именованного класса. OWL также определяет пустой класс `owl:Thing`.

Упомянутая выше программная среда проектирования онтологий *Protégé* позволяет разрабатывать OWL-онтологии, а также автоматически их генерировать из ранее созданных таксономий классов. Ниже приведены фрагменты сгенерированных в *Protégé* OWL-описаний концептов, определений классов и аксиом онтологии СИ КС. Так, введенные нами выше классы *«SYSTEMS – ENGINEERING»* и *«REQUIREMENT – ENGINEERING»*, определяются следующим образом:

```
<owl:Class rdf:ID="Systems_engineering"/>
<owl:Class rdf:ID="Requirement_engineering"/>
```

Определение общего класса онтологии *«SYSTEMS – ENGINEERING»* и принадлежность его классу «*THING*» (1) выражается так:

```
<rdfs:domain>
<owl:Class>
<owl:unionOf rdf:parseType="Collection">
<rdf:Description rdf:about=
"http://www.w3.org/2002/07/owl#Thing"/>
<owl:Class rdf:about="#Systems_engineering"/>
</owl:unionOf>
</owl:Class>
</rdfs:domain>
```

Аксиомы включения (5) и (6) онтологии СИ КС, записываются в виде:

```
<owl:Class rdf:about="#Functional_requirement">
<rdfs:subClassOf
rdf:resource="#Requirement_engineering"/>
</owl:Class>
<owl:Class rdf:about="#Requirement_engineering">
<rdfs:subClassOf
rdf:resource="#Systems_engineering"/>
</owl:Class>
```

OWL-онтологию (документ, описанный на языке OWL) СИ КС при необходимости можно преобразовать в XML-документ и использовать его для обмена в распределенной компьютерной системе поддержки СИ. С другой стороны, OWL-документ можно конвертировать в XMI-формат для обмена моделями и, затем, в модель на языке SysML (UML). Следовательно, в зависимости от целей конкретных процессов СИ и на различных этапах ЖЦ системы можно использовать языки OWL и SysML (рис. 5)

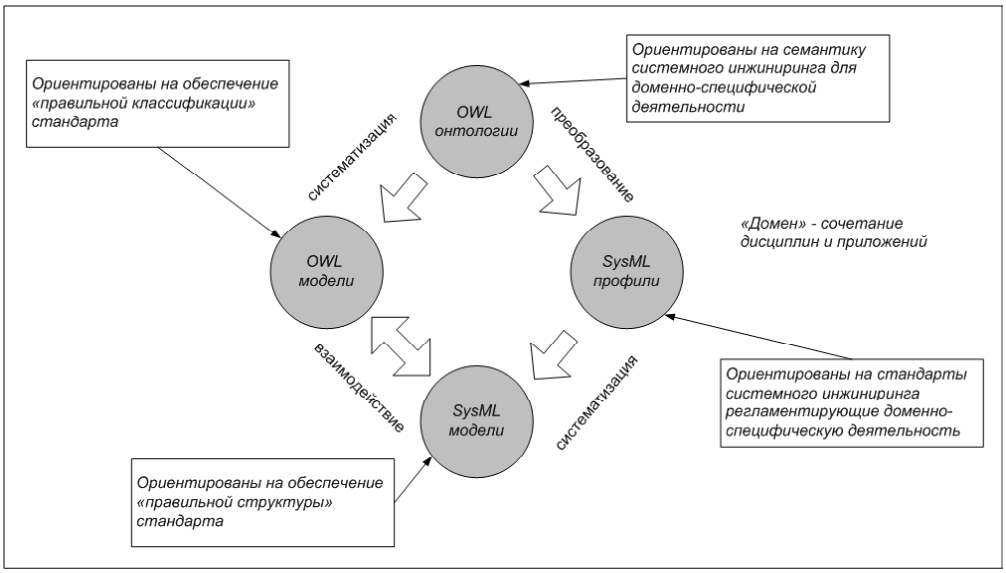


Рис. 5. Взаимодействие между OWL и SysML моделями

ВЫВОДЫ

Таким образом, в работе обоснована необходимость создания онтологии СИ КС, возможность ее формализации с помощью диалекта AL дескриптивной логики и представления ее на языке OWL DL. На примере разработанного прототипа продемонстрированы основные принципы, которые будут положены в основу дальнейшей разработки онтологии СИ КС. В качестве инструментального средства проектирования онтологии выбран пакет *Protégé* . Онтология СИ КС будет интегрирована в процесс MBSE, что позволит повысить эффективность использования MBSE для разработки сложных систем.

СПИСОК ЛИТЕРАТУРЫ

1. Systems Engineering Handbook [Text] / INCOSE. – TP-2003-016-02, Version 2a, 1 June 2004. – USA: INCOSE, 2004. – 300 p.
2. ГОСТ Р ИСО/МЭК 15288 – 2005. Системная инженерия. Процессы жизненного цикла систем [Текст]. – Введен 2005-12-29. – М. : Изд-во стандартов, 2006. – 57 с.
3. Systems Engineering Vision 2020 [Электронный ресурс] / INCOSE-TP-2004-004-02, September, 2007. – Режим доступа: http://www.incose.org/ProductsPubs/pdf/SEVision2020_20071003_v2_03.pdf
4. Kossiakoff A. Systems engineering: principles and practice [Text] / Alexander Kossiakoff et al.–2nd ed. (Wiley series in systems engineering and management; 67) – WILEY, 2011. – 559 p.
5. OMG Systems Modeling Language (OMG SysML™) [Электронный ресурс]. – Режим доступа: www.sysml.org/docs/specs/OMGSysML-v1.2-10-06-02.pdf
6. Ontology in MBSE / INCOSE // International Council on Systems Engineering IW2011 MBSE Workshop, January 29, 2011. – USA, 2011. – 11 p.
7. Gruber, Thomas R. A translation approach to portable ontology specifications [Tex] / Thomas R. Gruber // Appeared in Knowledge Acquisition. – 1993. – №5(2). – P. 199–220.
8. ECSS-E-ST-10C. Space engineering. System engineering general requirements [Text]. – 6 March 2009. – ECSS Secretariat SA-ESTEC, Requirements & Standards Division. Noordwijk, Netherlands, 2009. – 100 p.
9. Systems Engineering Handbook [Text] / Aeronautics and Space Administration, Headquarters Washington DC 20546, SP-2007-6105 Rev1. – Washington, 2007. – 360 p.
10. NASA Exploration Information Ontology Model (NExIOM) primer and vision [Text] / National Aeronautics and Space Administration, Headquarters Washington DC 20546. – Washington, 2005. – 18 p.
11. Noy N. Ontology Development 101: A Guide to Creating Your First Ontology [Электронный ресурс] / Natalya F. Noy, D. L. McGuinness // Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880, March 2001. – Режим доступа: http://protege.stanford.edu/publications/ontology_development/ontology101-noy-mcguinness.html.
12. Baader F. The Description Logic Handbook [Text] / F. Baader at al. – Cambridge University Press, 2003. – 574 p.
13. OWL Web Ontology Language Guide / W3C [Электронный ресурс]. – Режим доступа: http://www.w3.org/TR/owl-guide/

Стаття надійшла до редакції 23.05.2011.

Гонтар Н. А. , Кудерметов Р. К.

РОЗРОБКА ОНТОЛОГІЇ СИСТЕМНОГО ІНЖИНІРИНГУ КОСМІЧНИХ СИСТЕМ

У статті обгрунтована можливість створення онтологій системного інжинірингу (СІ). Онтологія СІ разом з переходом до моделє-орієнтованого СІ підвищить ефективність розробки складних систем. Запропоновано прототип онтологічної моделі, створений за допомогою дескриптивної логіки і формалізований за допомогою мови OWL DL.

Ключові слова: системний інжиніринг, онтологія, MBSE, дескриптивна логіка, OWL.

Gontar N. A., Kudermetov R. K.

WORKING OUT ONTOLOGY OF SYSTEMS ENGINEEERING OF SPACE SYSTEM

This article grounded the need to create ontology of systems engineering (SE). SE ontology with the transition to model-based approach SE will increase the efficiency of development of complex systems. The prototype of the ontology model based on description logic and formalized through language OWL DL is proposed.

Key words: systems engineering, ontology, MBSE, description logic, OWL.

УДК 004.627: 004.272.26

Скрупский С. Ю.[1], Луценко Н. В.[2], Скрупская Л. С.[3]

[1, 3]Ассистент Запорожского национального технического университета
[2] Старший преподаватель Запорожского национального технического университета

ПАРАМЕТРЫ КОМПРЕССИИ ВИДЕОИНФОРМАЦИИ В РАСПРЕДЕЛЕННЫХ СИСТЕМАХ

Проанализированы основные параметры компрессии видеоинформации в распределенных системах. Экспериментально исследовано влияние этих параметров на коэффициент сжатия, уровень искажения и время сжатия видеоинформации в распределенной системе.

Ключевые слова: видеоинформация, распределенная система, компрессия, битрейт, уровень искажения.

ВВЕДЕНИЕ

В настоящее время видеоинформацию хранят в архивах в сжатом цифровом виде, поскольку в несжатом виде она занимает значительные объемы памяти [1]. Проблема хранения видеоинформации особенно актуальна с распространением стандарта телевидения высо-

кой четкости (HDTV) [2], так как видеоинформация, удовлетворяющая такому стандарту, занимает существенно большие объемы памяти по сравнению с видеоинформацией, удовлетворяющей стандарту SDTV [3, 4]. Современные методы сжатия видеоинформации обладают высокой вычислительной сложностью [5], поэтому для достижения приемлемого для пользователя времени сжатия компрессию видеоинформации выполняют при помощи параллельных и распределенных компьютерных систем [6–8].

ПОСТАНОВКА ЗАДАЧИ

При сжатии видеоинформации в распределенной системе преследуется цель получения качественного результата с хорошим коэффициентом сжатия за относительно приемлемое для пользователя время. Достижение этой цели возможно как путем повышения производительности применяемого в распределенной системе оборудования, так и путем варьирования параметров кодера, осуществляющего компрессию видеоинформации.

Рассмотрим видеопоследовательности, разделяемые на части при помощи метода [9] и сжимаемые на узлах распределенной системы. Специфика, таких видеопоследовательностей заключается в том, что после разделения они ограничены одной сценой и их смежные кадры характеризуются значительной корреляционной зависимостью. Следовательно, наиболее трудоемкая операция оценки движения блоков в алгоритмах сжатия может быть выполнена на относительно небольшой ширине области поиска векторов движения без значительного ущерба качеству сжатия. В данной работе ставится задача экспериментально проверить эту гипотезу, а так же исследовать влияние параметров сжатия видеоинформации, разделяемой на части при помощи метода [9] и сжимаемой на узлах распределенной системы, на показатели результата компрессии.

Суть рассматриваемого метода разделения видеоинформации на части в распределенных системах заключается в вычислении коэффициентов корреляции между всеми смежными кадрами видеопоследовательности. Затем для коэффициента корреляции текущей пары кадров вычисляется значение функции чувствительности к смене сцен анализатора видеопоследовательности. Если значение функции чувствительности превышает значение модуля коэффициента корреляции, то видеопоследовательность разделяется на две части. Первая часть содержит кадры исходной видеопоследовательности до определенного с помощью функции чувствительности места слабой межкадровой корреляции, а вторая видеопоследовательность – с кадрами после места слабой межкадровой корреляции. Если же значение функции чувствительности меньше значения модуля коэффициента корреляции, тогда вычисляется значение функции чувствительности для следующей пары смежных кадров, которое сравнивается с соответствующим значением модуля коэффициента корреляции.

АНАЛИЗ ПАРАМЕТРОВ КОМПРЕССИИ ВИДЕОИНФОРМАЦИИ

Рассмотрим основные параметры компрессии видеоинформации:

– битрейт – количество бит, формируемых в единицу времени для кодирования видеопоследовательности;

– ширина области поиска векторов движения операции оценки движения (ОД) блоков кадра, которую обозначим как Ω;

– количество восстановленных кадров операции ОД блоков кадра (*ReFrames*).

Для видеопоследовательностей, удовлетворяющих стандарту HDTV, битрейт 0,5–2 Мбит/с условно считается малым, 2–6 Мбит/с – средним, а 6 Мбит/с и более на данный момент принято считать большим [10]. Чем больше битрейт закодированной видеопоследовательности, тем меньше ее искажение относительно исходной, но тем больший объем памяти она занимает.

В соответствии с операцией ОД блоков кадры разбиваются на блоки, далее в рассматриваемом кадре для каждого блока, обозначаемого C_n, выполняется поиск в области Ω восстановленного кадра наиболее «похожего» блока, обозначаемого C_k, смещенного на вектор движения [11]. Ширина области поиска влияет как на качество результата, так и на время сжатия видеопоследовательности. Типичными значениями ширины области поиска ОД блоков кадра полным перебором являются 16, 32, 64 и 128 пикселов [10].

Для более точного поиска векторов движения в современных стандартах сжатия видеоинформации применяют уточнение векторов движения до половины или даже до четверти пиксела. Для этого путем интерполяции формируют три дополнительных базовых блока, смещенных относительно найденного блока C_k на половину пиксела вправо, на половину пиксела вниз и на половину пиксела вправо-вниз. Затем из полученных четырех векторов-кандидатов выбирается лучший по какому-либо критерию вектор. Аналогично выполняется уточнение до четверти пиксела. Чем точнее поиск в операции ОД, тем ниже уровень искажения результата, но тем больше времени потребуется кодеру для сжатия видеоинформации.

Для оценки движения блоков текущего кадра могут быть использованы не один, а несколько предыдущих или последующих в порядке воспроизведения восстановленных кадров. Тогда для совместного поиска векторов движения используются эти восстановленные кадры, что позволяет улучшить качество результата сжатия, но при этом увеличивается объем вычислений в алгоритме сжатия. Типичными значениями количества восстановленных кадров операции ОД являются 1, 3, 5 и 7 кадров [10].

К основным показателям результата сжатия относятся следующие [11]:

– коэффициент сжатия;

– уровень искажения;

– время сжатия видеопоследовательности.

Коэффициент сжатия характеризует эффективность сжатия – отношение размера выходного файла, содержащего сжатую видеопоследовательность, к размеру входного файла, содержащего исходную видеопоследовательность. Чем эффективнее метод сжатия, тем меньше коэффициент сжатия.

Объективной характеристикой качества результата сжатия является уровень искажения восстановленной видеопоследовательности относительно исходной. Наиболее распространенной метрикой уровня искажения является Peak signal-to-noise ratio – PSNR, которая измеряется в дБ и вычисляется по формуле:

$$PSNR(x, y) = 10 \cdot \log_{10} \frac{255^2 \cdot W \cdot H}{\sum_{i=0}^{W} \sum_{j=0}^{H} (x_{ij} - y_{ij})^2}, \qquad (1)$$

где W и H – ширина и высота кадра видеопоследовательности, соответственно; x – восстановленная видеопоследовательность; y – исходная видеопоследовательность; i, j – индексы пиксела в кадре. Чем больше значение PSNR, тем ниже уровень искажения и качественнее сжатая видеопоследовательность.

Время сжатия видеопоследовательности – это время, затрачиваемое кодером для осуществления алгоритма сжатия видеопоследовательности.

ОРГАНИЗАЦИЯ ВЫЧИСЛИТЕЛЬНЫХ ЭКСПЕРИМЕНТОВ

В распределенную компьютерную систему компрессии видеоинформации на узел-анализатор поступает видеоинформация для сжатия. Анализатор принимает ее и выполняет разделение видеопоследовательностей на части предложенным в [9] методом.

Получаемые в результате разделения части видеопоследовательностей по мере появления становятся в очередь на сжатие вычислительными узлами. Если какой-либо вычислительный узел свободен, то он изымает видеопоследовательность из очереди и сжимает ее с заданными параметрами при помощи программы, реализующей алгоритмы стандарта MPEG-4/H264 [12]. В качестве такой программы использован видеокомпрессор FFmpeg [13]. Сжатая часть видеоинформации поступает в хранилище. Если все ресурсы заняты, то видеопоследовательность ожидает в очереди пока не освободится хотя бы один процессор. Такая организация вычислительного процесса позволяет избежать накладных расходов на синхронизацию процессоров, а так же сжимать части видеопоследовательностей по мере их поступления от узла-анализатора.

Для проведения экспериментов использован предоставленный для исследования методов сжатия видеоинформации кластер Института проблем моделирования в энергетике (ИПМЭ) им. Г. Е. Пухова г. Киев: 4 узла следующей конфигурации:

– процессор Intel Xeon 5405;

– оперативная память 4×2 ГБ DDR-2 на каждый вычислительный узел;

– коммуникационная среда InfiniBand 20Гб/с.

На узлах кластера, задействованных в экспериментах, помимо FFmpeg установлено промежуточное ПО (middleware): пакет ARC [14] с планировщиком Torque [15] на операционной системе CentOS 5.2.

В экспериментах использованы тестовые видеопоследовательности, удовлетворяющие стандарту HDTV, подробнее описанные в работе [9] и приведенные в таблицах данной статьи. Они были получены конкатенацией файлов, содержащих общепринятые для тестирования методов сжатия видеопоследовательности [16, 17] с разрешением кадра 1920×1080 пикселов. Конкатенация обеспечивала дополнительные смены сцен. После разделения тестовых видеопоследовательностей на части анализатором при помощи упомянутого выше метода, получаемые части видеопоследовательностей ограничены одной сценой, а их смежные кадры характеризуются значительной корреляционной зависимостью.

Во всех экспериментах применено дискретное косинусное преобразование (DCT) для устранения интракадровой избыточности и контекстно-адаптивное двоичное арифметическое кодирование (CABAC) для статистического кодирования без потерь [11, 18]. ОД блоков выполнялась полным перебором векторов движения с четвертьпикселной точностью. Параметры DCT и CABAC являются общепринятыми в сжатии видеоинформации, поэтому в данной работе подробно на них останавливаться не будем.

Путем варьирования параметров кодера, осуществляющего компрессию видеоинформации в распределенной системе, экспериментально было исследовано влияние каждого из проанализированных параметров компрессии на показатели результата сжатия.

РЕЗУЛЬТАТЫ ЭКСПЕРИМЕНТОВ

Далее приводятся усредненные значения результатов многократных экспериментов на четырех узлах кластера при сжатии описанных выше тестовых видеопоследовательностей. Такие видеопоследовательности однотипны, т. е. ограничены одной сценой, их смежные кадры характеризуются значительной корреляционной зависимостью, следовательно, результаты экспериментов для данной выборки видеопоследовательностей позволяют сформировать выводы в целом относительно влияния параметров сжатия видеоинформации, разделяемой на части при помощи метода [9] и сжимаемой на узлах распределенной системы, на показатели результата компрессии. В следующих таблицах величина «среднее значение PSNR» вычислена по формуле (1) и усреднена по всем кадрам видеопоследовательности.

Коэффициенты сжатия, средние значения PSNR и время сжатия, полученные в результате экспериментов при битрейте 0,5 Мбит/с и пяти восстановленных кадрах (*ReFrames=5*) для ОД блоков, приведены в табл. 1–3.

Таким образом, при битрейте 0,5 Мбит/с расширение области поиска векторов движения с 16 до 128 пикселов ведет к незначительному улучшению качества

результата (до 1,36 %) и увеличению коэффициента сжатия (до 7,14 %). Время сжатия существенно возрастает (в 9,2 раза) за счет увеличения количества сравниваемых векторов движения при расширении области поиска. При этом для описания векторов движения требуется больший объем памяти, поэтому коэффициент сжатия возрастает.

Результаты аналогичных экспериментов, но при битрейте 4 Мбит/с и пяти восстановленных кадрах для ОД блоков приведены в табл. 4–6.

Таблица 1. Коэффициенты сжатия при битрейте 0,5 Мбит/с и *ReFrames=5*

Видеопоследовательность	Ω , пикселов			
	16	32	64	128
Blue sky Pedestrian area part1	$8{,}6\cdot10^{-4}$		$8{,}7\cdot10^{-4}$	
Blue sky Pedestrian area part2	10^{-3}			
Riverbed Station part1	$2{,}4\cdot10^{-3}$		$2{,}5\cdot10^{-3}$	
Riverbed Station part2	$7{,}4\cdot10^{-4}$		$7{,}5\cdot10^{-4}$	
Rush hour Tractor part1	$8{,}1\cdot10^{-4}$		$8{,}2\cdot10^{-4}$	
Rush hour Tractor part2	$1{,}4\cdot10^{-3}$		$1{,}5\cdot10^{-3}$	

Таблица 2. Средние значения PSNR (дБ) при битрейте 0,5 Мбит/с и *ReFrames=5*

Видеопоследовательность	Ω , пикселов			
	16	32	64	128
Blue sky Pedestrian area part1	32,45	32,47	32,55	32,56
Blue sky Pedestrian area part2	32,58	32,65	32,7	32,76
Riverbed Station part1	27,98	28,16	28,28	28,36
Riverbed Station part2	34,9	34,96		34,97
Rush hour Tractor part1	35,02	35,08	35,14	
Rush hour Tractor part2	29,54	29,61	29,68	29,72

Таблица 3. Время сжатия (с) при битрейте 0,5 Мбит/с и *ReFrames=5*

Видеопоследовательность	Ω , пикселов			
	16	32	64	128
Blue sky Pedestrian area part1	18,07	28,19	57,74	147,82
Blue sky Pedestrian area part2	27,09	42,43	87,38	222,43
Riverbed Station part1	33,71	55,51	115,54	289,67
Riverbed Station part2	27,75	43,85	89,82	236
Rush hour Tractor part1	17,72	26,54	52,58	136,77
Rush hour Tractor part2	34,37	56,66	121	317,02

Таблица 4. Коэффициенты сжатия при битрейте 4 Мбит/с и *ReFrames=5*

Видеопоследовательность	Ω , пикселов
	16, 32, 64, 128
Blue sky Pedestrian area part1	$6\cdot10^{-3}$
Blue sky Pedestrian area part2	
Riverbed Station part1	$8{,}2\cdot10^{-3}$
Riverbed Station part2	$5{,}9\cdot10^{-3}$
Rush hour Tractor part1	$6{,}3\cdot10^{-3}$
Rush hour Tractor part2	$6{,}5\cdot10^{-3}$

Таким образом, при битрейте 4 Мбит/с расширение области поиска векторов движения ведет к несущественному улучшению качества результата (до 0,44 %). Качество сжатия улучшается менее заметно, чем при битрейте 0,5 Мбит/с, поскольку кодер может формировать больше бит за секунду для описания векторов движения, следовательно, расширять область поиска не обязательно. При этом коэффициент сжатия не изменяется, а время сжатия существенно увеличивается (в 11,3 раз).

Подобные эксперименты были проведены так же при битрейте 8 Мбит/с и пяти восстановленных кадрах для ОД блоков. Результаты приведены в табл. 7–9.

Таблица 5. Средние значения PSNR (дБ) при битрейте 4 Мбит/с и *ReFrames=5*

Видеопоследовательность	Ω , пикселов			
	16	32	64	128
Blue sky Pedestrian area part1	41,08	41,1	41,1	41,2
Blue sky Pedestrian area part2	40,9			40,91
Riverbed Station part1	31,55	31,6	31,65	31,69
Riverbed Station part2	41,88		41,89	
Rush hour Tractor part1	41,92	41,93		
Rush hour Tractor part2	36,09	36,11	36,14	36,16

Таблица 6. Время сжатия (с) при битрейте 4 Мбит/с и *ReFrames=5*

Видеопоследовательность	Ω , пикселов			
	16	32	64	128
Blue sky Pedestrian area part1	51,24	95,3	204,82	504
Blue sky Pedestrian area part2	70,54	129,5	283,4	728
Riverbed Station part1	70,1	129,11	287,27	746
Riverbed Station part2	61,79	112,07	247,7	698,16
Rush hour Tractor part1	57,37	104,06	230,58	588
Rush hour Tractor part2	63,53	104,51	264,1	727,1

Таблица 7. Коэффициенты сжатия при битрейте 8 Мбит/с и *ReFrames=5*

Видеопоследовательность	Ω , пикселов
	16, 32, 64, 128
Blue sky Pedestrian area part1	$12{,}2\cdot10^{-3}$
Blue sky Pedestrian area part2	$12{,}1\cdot10^{-3}$
Riverbed Station part1	$15{,}4\cdot10^{-3}$
Riverbed Station part2	$12{,}4\cdot10^{-3}$
Rush hour Tractor part1	$12{,}5\cdot10^{-3}$
Rush hour Tractor part2	$12{,}7\cdot10^{-3}$

Таблица 8. Средние значения PSNR (дБ) при битрейте 8 Мбит/с и *ReFrames=5*

Видеопоследовательность	Ω , пикселов			
	16	32	64	128
Blue sky Pedestrian area part1	42,7	42,72	42,73	42,74
Blue sky Pedestrian area part2	42,46			
Riverbed Station part1	33,97		34	
Riverbed Station part2	42,48			
Rush hour Tractor part1	42,81			
Rush hour Tractor part2	38,35	38,36	38,37	38,38

Таким образом, при битрейте 8 Мбит/с расширение области поиска векторов движения ведет к еще менее существенному улучшению качества результата (до 0,08 %). При этом коэффициент сжатия не изменяется, а время сжатия существенно увеличивается (примерно в 12,5 раз). Следовательно, расширять область поиска векторов движения при битрейте 8 Мбит/с и более не целесообразно.

Для исследования влияния количества восстановленных кадров операции ОД блоков на показатели результата сжатия выполнены аналогичные эксперименты, в которых при фиксированном битрейте 4 Мбит/с изменялось количество восстановленных кадров ОД. Результаты экспериментов приведены в табл. 10–15.

Таким образом, из таблиц 4–6 и 10–15 видно, что увеличение количества восстановленных кадров с одного до семи для ОД блоков при фиксированном битрейте 4 Мбит/с незначительно повышает качество сжатия (до 0,7 %) и не влияет на коэффициент сжатия, однако время сжатия увеличивается существенно (в 3 раза).

Таблица 9. Время сжатия (с) при битрейте 8 Мбит/с и *ReFrames=5*

Видеопоследовательность	Ω, пикселов			
	16	32	64	128
Blue sky Pedestrian area part1	81,2	155,44	324,5	778
Blue sky Pedestrian area part2	94,06	177,11	393,18	1005
Riverbed Station part1	98,24	188,12	435	1155
Riverbed Station part2	85,52	160,73	358,8	929
Rush hour Tractor part1	80,22	155,1	347	870
Rush hour Tractor part2	81,08	151,38	362,3	1017

Таблица 10. Коэффициенты сжатия при битрейте 4 Мбит/с и *ReFrames=1*

Видеопоследовательность	Ω, пикселов			
	16	32	64	128
Blue sky Pedestrian area part1	$6\cdot10^{-3}$			
Blue sky Pedestrian area part2				
Riverbed Station part1	$8,2\cdot10^{-3}$	$8,3\cdot10^{-3}$		
Riverbed Station part2	$5,9\cdot10^{-3}$			
Rush hour Tractor part1	$6,3\cdot10^{-3}$			
Rush hour Tractor part2	$6,5\cdot10^{-3}$			

Таблица 11. Средние значения PSNR (дБ) при битрейте 4 Мбит/с и *ReFrames=1*

Видеопоследовательность	Ω, пикселов			
	16	32	64	128
Blue sky Pedestrian area part1	41,07			
Blue sky Pedestrian area part2	40,81			
Riverbed Station part1	31,41	31,42	31,47	31,51
Riverbed Station part2	41,86	41,86	41,86	41,86
Rush hour Tractor part1	41,85			
Rush hour Tractor part2	36	36,02	36,04	36,06

Таблица 12. Время сжатия (с) при битрейте 4 Мбит/с и *ReFrames=1*

Видеопоследовательность	Ω, пикселов			
	16	32	64	128
Blue sky Pedestrian area part1	33,21	57,21	116,17	279,6
Blue sky Pedestrian area part2	43,12	73,35	152,43	381,6
Riverbed Station part1	36,32	60,7	129,45	328,53
Riverbed Station part2	43,18	73,35	155,48	393,75
Rush hour Tractor part1	35,66	59,28	126	315
Rush hour Tractor part2	40,57	67,91	147,42	389

Таблица 13. Коэффициенты сжатия при битрейте 4 Мбит/с и *ReFrames=7*

Видеопоследовательность	Ω, пикселов
	16, 32, 64, 128
Blue sky Pedestrian area part1	$6\cdot10^{-3}$
Blue sky Pedestrian area part2	
Riverbed Station part1	$8,2\cdot10^{-3}$
Riverbed Station part2	$5,9\cdot10^{-3}$
Rush hour Tractor part1	$6,3\cdot10^{-3}$
Rush hour Tractor part2	$6,5\cdot10^{-3}$

Таблица 14. Средние значения PSNR (дБ) при битрейте 4 Мбит/с и *ReFrames=7*

Видеопоследовательность	Ω, пикселов			
	16	32	64	128
Blue sky Pedestrian area part1	41,09		41,1	
Blue sky Pedestrian area part2	40,92			
Riverbed Station part1	31,59	31,64	31,69	31,73
Riverbed Station part2	41,88			
Rush hour Tractor part1	41,93	41,94		
Rush hour Tractor part2	36,1	36,13	36,15	36,16

Таблица 15. Время сжатия (с) при битрейте 4 Мбит/с и *ReFrames=7*

Видеопоследовательность	Ω, пикселов			
	16	32	64	128
Blue sky Pedestrian area part1	69,22	125,44	273,33	685
Blue sky Pedestrian area part2	92,81	175	388,6	1007
Riverbed Station part1	89,98	167,74	374,25	982
Riverbed Station part2	76,26	142,82	322,05	850
Rush hour Tractor part1	71,59	135,36	305,58	788
Rush hour Tractor part2	80,84	150,89	356,38	993,2

ВЫВОДЫ

Результаты экспериментов позволяют сделать выводы о том, что на видеопоследовательности, получаемые после разделения видеоинформации на части в распределенной системе предложенным в [9] методом, параметры компрессии оказывают следующее влияние:

– повышение битрейта с 0,5 Мбит/с до 8 Мбит/с ведет к существенному улучшению качества (до 31,59 %), но, в то же время, к значительному ухудшению коэффициента сжатия (в 16,75 раз) и увеличению времени сжатия (в 6,6 раз);

– увеличение количества восстановленных кадров с одного до семи для ОД блоков при фиксированном битрейте 4 Мбит/с незначительно повышает качество сжатия (до 0,7 %) и не влияет на коэффициент сжатия, при этом время сжатия увеличивается существенно (в 3 раза);

– при фиксированном битрейте расширение области поиска векторов движения для ОД блоков существенно увеличивает время сжатия (в 9,2 раз при битрейте 0,5 Мбит/с, в 11,3 раз при битрейте 4 Мбит/с и в 12,5 раз при битрейте 8 Мбит/с). При этом, чем больше битрейт, тем меньше выигрыш в качестве от расширения области поиска ОД блоков. Коэффициент сжатия при битрейте 0,5 Мбит/с увеличивается от расширения области поиска ОД блоков, при битрейте 4 Мбит/с и более – не изменяется.

В дальнейшем представляется перспективным исследовать влияние динамического изменения ширины области поиска векторов движения для каждого блока кадра видеопоследовательности на производительность компрессии видеоинформации в распределенных системах.

СПИСОК ЛИТЕРАТУРЫ

1. Lossless Video Compression for Archives: Motion JPEG2k and Other Options / Media Matters Whitepapers. – Режим доступа: \www/ URL: http://www.media-matters.net/docs/WhitePapers/WPMJ2k.pdf – 1.02.2006 г. – Загл. с экрана.
2. Parameter values for the HDTV standards for production and international programme exchange : ITU-R Recommendation BT.709, 2008. – 32 p.
3. Understanding and Using High-Deыnition Video / Adobe Systems. – Access mode: \www/ URL: http://www.adobe.com/products/premiere/pdfs/hdprimer.pdf – 1.11.2010 г. – Title from screen.
4. H.261: Video codec for audiovisual services : ITU-T Recommendation H.261, 1993. – 29 p.
5. Performance analysis and architecture evaluation of MPEG-4 video codec system : IEEE International Symposium on Circuits and Systems / H.-C. Chang, L.-G. Chen, M.-Y. Hsu, Y.-C. Chang. – Geneva, 2000. – P. 449–452.
6. Performance Evaluation of Parallel MPEG-4 Video Coding Algorithms on Clusters of Workstations : PARELEC ’04 Proceedings of the international conference on Parallel Computing in Electrical Engineering / A. Rodriguez, A. Gonzalez, M. P. Malumbres. – Washington DC, 2004. – P. 354–357.
7. Parallel Scalability of H.264 : Proceedings of the first Workshop on Programmability Issues for Multi-Core Computers / C. H. Meenderinck, A. Azevedo, M. Alvarez, B. H. Juurlink, A. Ramirez. – Goteborg, Sweden, 2008.
8. ON-DEMAND HD VIDEO USING JINI BASED GRID : ICME 2008 / S. Kent, P. Broadbent, N. Warren, S. Gulliver, 2008. – P. 1045–1048.
9. *Скрупский, С. Ю.* Повышение эффективности сжатия видеоинформации в распределенных компьютерных системах / С. Ю. Скрупский // Электронное моделирование. – 2011. – №6 (33). – С. 57–72.
10. H.264/MPEG-4 AVC – Access mode: \www/ URL: http://en.wikipedia.org/wiki/H.264/MPEG-4_AVC – 28.09.2010 г. – Title from screen.
11. *Скрупский, С. Ю.* Методы компрессии видеоинформации / С. Ю. Скрупский // Наукові праці Донецького національного технічного університету. Серія: «Обчислювальна техніка та автоматизація». – 2011, №21 (183). – С. 122–130.
12. Advanced video coding for generic audiovisual services : ITU-T Recommendation H.264 and ISO/IEC 14496-10 (AVC), 2003. – 90 p.
13. FFmpeg. – Режим доступа: \www/ URL: http://www.ffmpeg.org/– 1.10.2010 г. – Загл. с экрана.
14. *Ellert, M.* Advanced Resource Connector middleware for lightweight computational Grids / M. Ellert, M. Grønager, A. Konstantinov, B. Kónya, J. Lindemann, I. Livenson, J. L. Nielsen, M. Niinimäki, O. Smirnova, A. Wäänänen // Future Generation Computer Systems. – 2007, Vol. 23, Issue 2 – P. 219–240.
15. Система пакетной обработки заданий torque. Руководство пользователя. – М. : Т-Платформы, 2008. – 31 с.
16. YUV Video Sequences / Arizona State University. – Режим доступа: \www/ URL: http://trace.eas.asu.edu/yuv/ – 1.11.2010 г. – Загл. с экрана.
17. Xiph.org Test Media. – Режим доступа: \www/ URL: http://media.xiph.org/video/derf/ – 1.11.2010 г. – Загл. с экрана.
18. *Парфенова, А. О.* Порівняльний аналіз основних стандартів відео для передачі по 4-G мережам / А. О. Парфенова, А. Ю. Макаренко, С. Б. Могильний // Вісник Національного технічного університету України «КПІ». – 2010, №40. – С. 171–176.

Стаття надійшла до редакції 10.05.2011.

Скрупський С. Ю., Луценко Н. В., Скрупська Л. С.

ПАРАМЕТРИ КОМПРЕСІЇ ВІДЕОІНФОРМАЦІЇ В РОЗПОДІЛЕНИХ СИСТЕМАХ

Проаналізовані основні параметри компресії відеоінформації в розподілених системах. Експериментально досліджено вплив цих параметрів на коефіцієнт стискування, рівень викривлення та час стискування відеоінформації в розподіленій системі.

Ключові слова: відеоінформація, розподілена система, компресія, бітрейт, рівень викривлення.

Skrupsky S. Y., Lucenko N. V., Skrupskaya L. S.

THE PARAMETERS OF VIDEO INFORMATION COMPRESSION PROCESS IN DISTRIBUTED SYSTEMS

The paper deals with the analysis of the main parameters of video information compression process in distributed systems. The influence of these parameters on compression coefficient, distortion level and compression time of video information in distributed system is experimentally investigated.

Key words: video information, distributed system, compression, bit rate, distortion level.

УДК 004.75

Сметанин Р. И.[1], Тягунова М. Ю.[2]

[1]Канд. техн. наук, доцент Запорожского национального технического университета
[2] Магистрант Запорожского национального технического университета

ПОДХОД К РАСПРЕДЕЛЕНИЮ ПАРАЛЛЕЛЬНЫХ ВЕТВЕЙ ЗАДАНИЯ В ВЫЧИСЛИТЕЛЬНОЙ СИСТЕМЕ

Рассмотрена проблема распределения заданий в вычислительной системе для случая, когда длительность выполнения параллельных ветвей задания, которые необходимо распределить, различна. Для повышения эффективности распределения разработан метод. Показано, что предложенный метод является эффективным, поскольку, используя его, время выполнения заданий заметно сокращается.

Ключевые слова: вычислительная система, grid, параллельные вычисления, ветви задания, распределение заданий, время выполнения задания, время работы системы, время ожидания.

ВВЕДЕНИЕ

Сегодня направление параллельных вычислений в GRID имеет большой интерес со стороны научных исследований, поскольку увеличение скорости вычислений является весьма актуальным для всех, чья деятельность связана с большим объемом вычислительных работ [1]. Использование параллельных вычислений позволяет решать задачи, требующие больших временных затрат, за меньшее время [2, 3]. Однако при параллельном вычислении бывают случаи, когда некоторые ветви задания вычисляются раньше и ожидают окончания работы остальных. При этом вычислительные узлы простаивают, что сказывается на общем времени выполнения заданий, а, следовательно, и на загрузке всей системы. Таким образом, сокращение времени простоя позволит увеличить скорость вычислений.

В системах, в которых приоритет заданий играет важную роль, диспетчеризация нагрузки происходит поочередно для каждого задания и большее внимание уделяется распределению ветвей параллельного задания между компьютерами. В таком случае зачастую количество ветвей любого из заданий меньше количества компьютеров.

ПОСТАНОВКА ЗАДАЧИ

Рассмотрим вычислительную систему, в которую в определенный момент времени поступают параллельные задания, т. е. состоящие из параллельно выполняемых ветвей. Для сокращения общего времени выполнения всех поступающих для решения заданий и при этом минимизации нагрузки на вычислительные ресурсы следует определить, какая ветвь выполняемого параллельного задания будет выполняться на каком компьютере. Таким образом, необходимо решить проблему эффективного распределения параллельных ветвей задания между вычислительными узлами системы.

Пусть каждый из компьютеров имеет различную производительность (у некоторых компьютеров производительность может совпадать, но не у всех), длительность выполнения каждой из ветвей одного задания также различна, в результате чего время выполнения задания в целом определяется наибольшей длительностью выполнения одной из ветвей этого задания. Также следует отметить, что количество ветвей не превышает количества компьютеров.

Пусть данная вычислительная система состоит из n компьютеров, а очередь заданий состоит из q заданий (каждое задание содержит l_i ветвей, где $i = \overline{1,q}$). Тогда математическая модель этой задачи может быть записана следующим образом.

Мощность эталонного компьютера – P^e.

Мощность k-го компьютера – P_k, где $k = \overline{1,n}$.

Тогда коэффициент мощности k-го компьютера –

$$c_k = \frac{P^e}{P_k}, \quad (1)$$

т.е. чем больше мощность k-го компьютера, тем меньше его коэффициент мощности.

Во время распределения i-го задания свободно только a_i компьютеров. Учитывая (1), соответственно, коэффициент мощности m-го компьютера –

$$c_m = \frac{P^e}{P_m}, \quad (2)$$

где $m = \overline{1,a_i}$.

Тогда время выполнения j-ой ветви i-го задания на m-м компьютере t_{ijm}:

$$t_{ijm} = t_{ij}^e \times c_m, \quad (3)$$

где $j = \overline{1,l_i}$, t_{ij}^e – время выполнения j-ой ветви i-го задания на эталонном компьютере.

Тогда время выполнения i-го задания в данной системе T_i:

$$T_i = \max\{t_{ijm}\}, \quad (4)$$

Время ожидания окончания выполнения всего задания m-м компьютером, закончившим выполнение j-ой ветви i-го задания, обозначим как Δt_{ijm}:

$$\Delta t_{ijm} = T_i - t_{ijm}. \quad (5)$$

Суммарное время ожидания окончания всех ветвей i-го задания Δt_i:

$$\Delta t_i = \sum_{j=1}^{l_i} \Delta t_{ijm}. \quad (6)$$

Таким образом, для сокращения общего времени выполнения всех поступающих для решения заданий необходимо, чтобы

$$\begin{cases} T_i \to \min \\ \Delta t_i \to \min \end{cases}. \quad (7)$$

Введем понятие времени ожидания k-м компьютером начала выполнения $(i+r)$-го задания после окончания выполнения i-го задания, где $r = \overline{1,(q-1)}$. Обозначим это время как Δt_{ki}.

Тогда время работы системы, потраченное на выполнение всех заданий, рассчитывается как максимальное время работы одного из n компьютеров этой системы и равняется сумме длительностей ветвей заданий, выполненных на конкретном k-м компьютере плюс сумма всех Δt_{ki} для этого же компьютера.

А суммарное время ожидания всей системы Δt:

$$\Delta t = \sum_{k=1}^{n}\sum_{i=1}^{q} \Delta t_{ki} + \sum_{i=1}^{q} \Delta t_i.$$

Таким образом, необходимо, чтобы, учитывая (7), время работы системы стремилось к минимуму и $\Delta t \to \min$.

СУЩЕСТВУЮЩИЕ РЕШЕНИЯ

Для сравнения рассмотрим простейший случай, когда в каждый момент времени распределяется одно задание и самая длительная ветвь задания распределяется на наиболее мощной компьютер, а далее распределение происходит в порядке возрастания длительностей ветвей заданий и убывания мощностей компьютеров.

Согласно такому подходу, порядковые номера компьютеров и ветвей совпадают, т.е. $m = j$.

Такой подход обеспечивает выполнение первого из условий (7), но не в полной мере обеспечивает второе.

Для выполнения обоих условий в данной работе предлагается метод, в котором распределяются ветви заданий таким образом, чтобы были заняты также и менее мощные компьютеры. При этом вновь рассчитанное время выполнения j-й ветви i-го задания, t_{ijm}, не превышало уже рассчитанное время выполнения i-го задания, T_i. Таким образом, получаем, что выполняются условия (5) и для распределения $(i+1)$-го задания, выполняемого одновременно с i-м на других компьютерах, остаются вычислительные узлы с большей средней мощностью. За счет этого общее время выполнения Q заданий уменьшится.

РАЗРАБОТАННЫЙ МЕТОД

Рассмотрим работу разработанного метода на примере. На рис.1 изображено задание, содержащее 6 ветвей вычислений, которое выполняется в системе, состоящей из 6 компьютеров с одинаковой мощностью, – здесь все компьютеры можно считать эталонными. Длительности каждой ветви задания соответственно равны: 100, 80, 70, 60, 30, 20 условных единиц времени.

На рис. 1 система выглядит в виде 6 строк, каждая из которых представляет собой компьютер. Первая ячейка указывает номера компьютера; вторая – коэффициент мощности компьютера (здесь он равен 1.00, т. к. любой из компьютеров можно считать эталонным); третья – непосредственно графически представленную распределенную ветвь задания. Распределенная ветвь задания представлена в виде двух частей (или одной части для самой длительной ветви): собственно время выполнения с номером ветви задания и время ожидания окончания выполнения всех ветвей задания (заштрихованная область). При этом первое число номера ветви задания, обозначает номер задания, второе – номер ветви. К примеру, номер 1.2 указывает, что это распределена вторая ветвь первого задания.

Под изображением системы на рис. 1 представлены краткие статистические данные о работе системы с распределенными заданиями: время работы T и суммарное время ожидания системы Δt.

Рис. 1. Задание, выполняемое в системе, состоящей из компьютеров одинаковой мощности

Время выполнения такого задания в данной системе: $T = \max\{100, 80, 70, 60, 30, 20\} = 100$, а суммарное время ожидания окончания всех ветвей задания:

$$\Delta t = (100-100) + (100-80) + (100-70) + \\ +(100-60) + (100-30) + (100-20) = 240.$$

Рассмотрим более сложный случай. Используем вычислительную систему из 12 компьютеров с коэффициентами мощности соответственно: 0,5; 0,75; 0,8; 1; 1,2; 1,25; 1,4; 1,5; 2; 2,1; 2,1; 2,5. Запустим для выполнения в ней два задания, таких же, как и рассмотренное выше, и проведем расчеты времени работы и суммарного времени ожидании.

Согласно рассмотренному простейшему подходу, время выполнения каждой ветви первого задания рассчитывается следующим образом:

$$t_{111} = t_{11}^e \times c_1 = 100 \times 0{,}5 = 50,$$

$$t_{122} = t_{12}^e \times c_2 = 80 \times 0{,}75 = 60,$$

$$t_{133} = t_{13}^e \times c_3 = 70 \times 0{,}8 = 56,$$

$$t_{144} = t_{14}^e \times c_4 = 60 \times 1 = 60,$$

$$t_{155} = t_{15}^e \times c_5 = 30 \times 1{,}2 = 36,$$

$$t_{166} = t_{16}^e \times c_6 = 20 \times 1{,}25 = 25.$$

Тогда время выполнения первого задания в данной системе:

$T_1 = \max\{50, 60, 56, 60, 36, 25\} = 60$, а суммарное время ожидания окончания всех ветвей первого задания:

$$\Delta t_1 = (60-50) + (60-60) + (60-56) + \\ +(60-60) + (60-36) + (60-25) = 73.$$

Для второго задания время выполнения каждой ветви имеем:

$$t_{211} = t_{21}^e \times c_7 = 100 \times 1{,}4 = 140,$$

$$t_{222} = t_{22}^e \times c_8 = 80 \times 1{,}5 = 120,$$

$$t_{233} = t_{23}^e \times c_9 = 70 \times 2 = 140,$$

$$t_{244} = t_{24}^e \times c_{10} = 60 \times 2{,}1 = 126,$$

$$t_{255} = t_{25}^e \times c_{11} = 30 \times 2{,}1 = 63,$$

$$t_{266} = t_{26}^e \times c_{12} = 20 \times 2{,}5 = 50.$$

А время выполнения второго задания в данной системе и суммарное время ожидания окончания всех ветвей второго задания соответственно:

$$T_2 = \max\{140, 120, 140, 126, 63, 50\} = 140,$$

$$\Delta t_1 = (140-140) + (140-120) + (140-140) + \\ +(140-126) + (140-63) + (140-50) = 201.$$

В таком случае время работы системы равно 140 условных единиц времени, а суммарное время ожидания системы – 274.

Согласно разработанному методу, время выполнения первого задания в данной системе рассчитывается аналогично и так же равняется $T_1 = 60$. Время выполнения каждой j-ой ветви первого задания на каждом m-м компьютере t_{1jm} представлено в табл. 1.

Из этой таблицы выбираются значения, которые максимально приближены к T_1, но не более его. Эти значения в таблице выделены серым цветом. Выбор производится по очереди для каждого столбца, начиная с первого. При этом никакие два значения не должны принадлежать одному и тому же столбцу или строке.

Тогда суммарное время ожидания окончания всех ветвей первого задания:

$$\Delta t_1 = (60-50) + (60-60) + (60-56) + \\ +(60-60) + (60-60) + (60-50) = 24.$$

Время выполнения каждой j-й ветви второго задания на каждом m-м компьютере t_{2jm} и выбранные значения представлены в табл. 2.

Таблица 1. Время выполнения ветвей первого задания

Номер ветви / Номер компьютера	1	2	3	4	5	6
1	50	40	35	30	15	10
2	75	60	52,5	45	22.5	15
3	80	64	56	48	24	16
4	100	80	70	60	30	20
5	120	96	84	72	36	24
6	125	100	87,5	75	37,5	25
7	140	112	98	84	42	28
8	150	120	105	90	45	30
9	200	160	140	120	60	40
10	210	168	147	126	63	42
11	210	168	147	126	63	42
12	250	200	175	150	75	50

Таблица 2. Время выполнения ветвей второго задания

Номер ветви / Номер компьютера	1	2	3	4	5	6
5	120	96	84	72	36	24
6	125	100	87,5	75	37,5	25
7	140	112	98	84	42	28
8	150	120	105	90	45	30
10	210	168	147	126	63	42
11	210	168	147	126	63	42

Тогда время выполнения второго задания в данной системе:

$T_2 = \max\{120, 100, 98, 90, 63, 42\} = 120$, а суммарное время ожидания окончания всех ветвей второго задания:

$$\Delta t_2 = (120-120)+(120-120)+(120-98)+$$
$$+(120-75)+(120-63)+(120-42) = 202.$$

РЕЗУЛЬТАТЫ РАБОТЫ

В результате распределения заданий в данной системе согласно разработанному методу, время работы системы равно 120 условных единиц времени, а суммарное время ожидания системы – 226.

На рис. 2, 3 очевидны преимущества предлагаемого метода:

– распределив первое задание по компьютерам системы, не смотря на то, что время выполнения одинаково, время ожидания окончания первого задания гораздо меньше, а средняя мощность компьютеров, оставшихся в распоряжении второго задания, выше;

– распределив второе задание, имеем меньшее общее время выполнения и ожидания, что и требовалось получить.

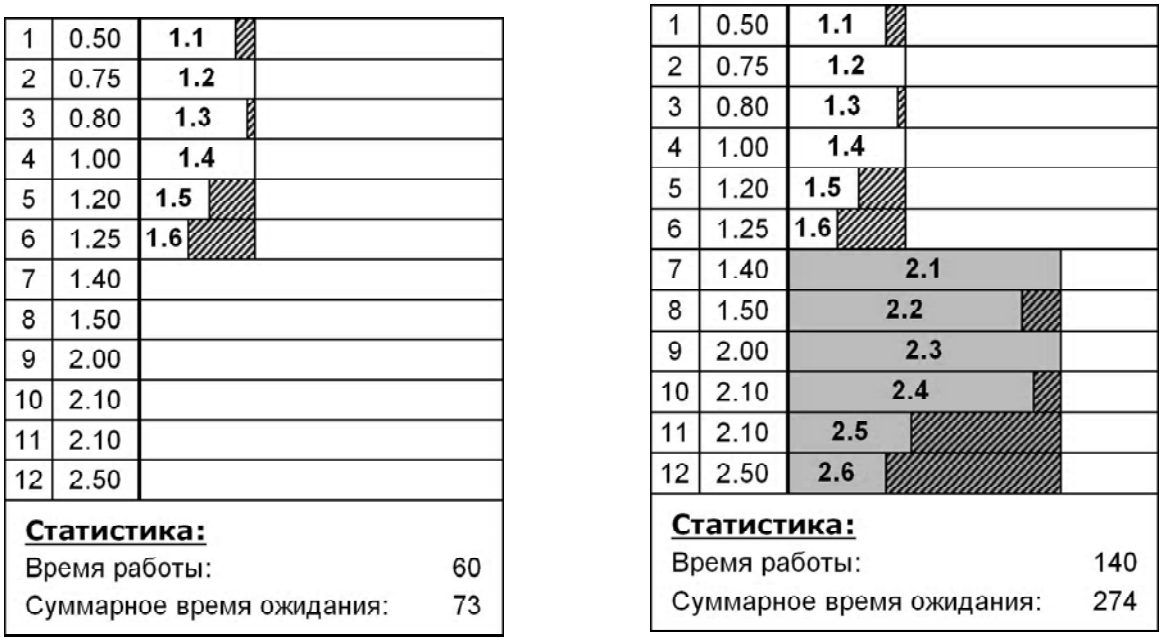


Рис. 2. Пример распределения заданий с использованием существующего метода

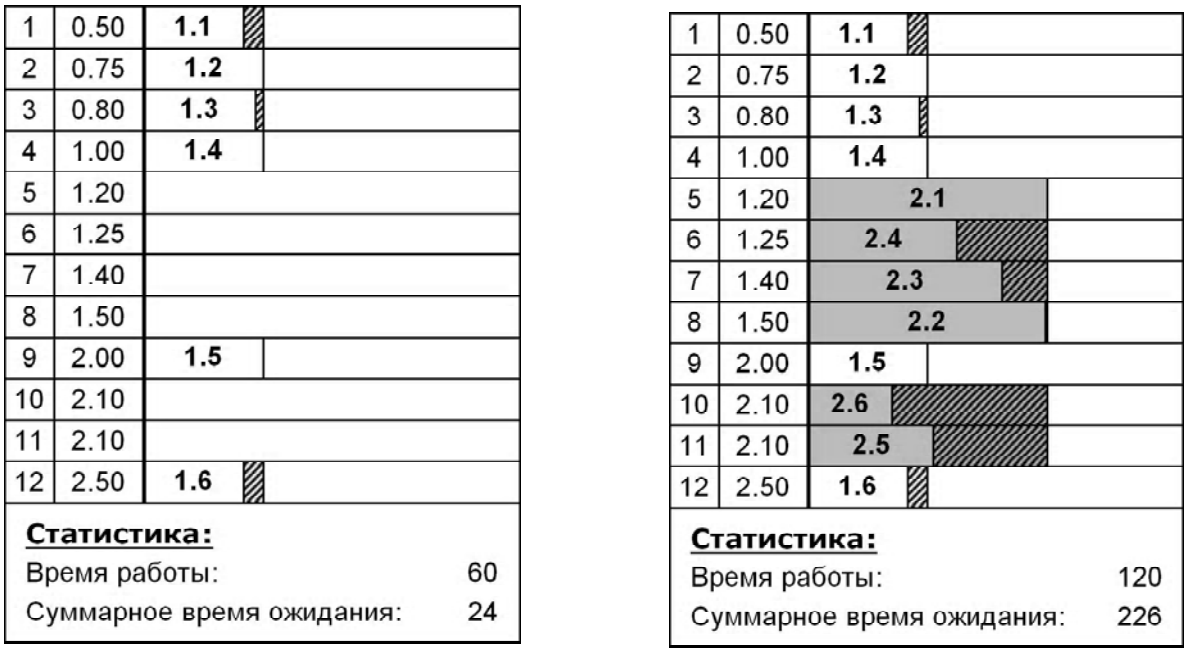


Рис. 3. Пример распределения заданий с использованием предлагаемого метода

ВЫВОДЫ

Таким образом, в данной статье был рассмотрен подход к распределению ветвей параллельного задания между компьютерами гетерогенной вычислительной среды. Был продемонстрирован пример оптимизации распределения, используя рассмотренный метод, эффективно решающий поставленную задачу.

СПИСОК ЛИТЕРАТУРЫ

1. Linux. Кластер. Практическое руководство по параллельным вычислениям [Электронный ресурс] / авт. курса Ю. Сбитнев – Электрон. текстовые дан. – [Волгоград?], [199–?] – Режим доступа: http://cluster.linux-ekb.info, свободный. – Загл. с экрана.
2. Шпаковский Г. И. Реализация параллельных вычислений: кластеры, многоядерные процессоры, грид, квантовые компьютеры / Г. И. Шпаковский. – Минск : БГУ.– 2010. – 155 с.
3. Технология запуска параллельных задач в различных распределенных средах / [В. М. Волохов, Д. А. Варламов, Н. Ф. Сурков, А. В. Пивушков, А. В. Волохов] // Распределенные вычисления и Грид-технологии в науке и образовании : труды 4-й междунар. конф. (Дубна, 28 июня – 3 июля, 2010 г.). – Дубна : ОИЯИ. – С. 329–334.

Стаття надійшла до редакції 20.05.2011.

Сметанін Р. І., Тягунова М. Ю.

ПІДХІД ДО РОЗПОДІЛУ ПАРАЛЕЛЬНИХ ГІЛОК ЗАВДАННЯ В ОБЧИСЛЮВАЛЬНІЙ СИСТЕМІ

Розглянуто проблему розподілу завдань в обчислювальній системі для випадку, коли тривалість виконання паралельних гілок завдання, які необхідно розподілити, різна. Розроблено метод для підвищення ефективності розподілу. Показано, що запропонований метод є ефективним, оскільки, використовуючи його, час виконання завдань помітно скорочується.

Ключові слова: обчислювальна система, grid, паралельні обчислення, гілки завдання, розподілення завдань, час виконання завдання, час роботи системи, час очікування.

Smetanin R. I., Tyagunova M. Yu.

APPROACH TO THE DISTRIBUTION OF PARALLEL TASK THREADS IN COMPUTING SYSTEMS

The problem of the tasks distribution in the computer system with different execution time of parallel threads, which must be distributed, is considered. Method to improve distribution efficiency is developed. Method is effective, because it much reduces execution time of task.

Keywords: computing system, grid, parallel computations, task threads, tasks distribution, task execution time, system work time, waiting time.

УПРАВЛІННЯ У ТЕХНІЧНИХ СИСТЕМАХ

УПРАВЛЕНИЕ В ТЕХНИЧЕСКИХ СИСТЕМАХ

CONTROL IN TECHNICAL SYSTEMS

УДК 621.11-32

Александрова Т. Е.[1], Кононенко В. А.[2], Лазаренко А. А.[3], Зейн Али[3]

[1]Канд. техн. наук, доцент Национального технического университета «ХПИ»
[2]Канд. техн. наук, старший научный сотрудник Национального технического университета «ХПИ»
[3] Аспирант Национального технического университета «ХПИ»

СРАВНИТЕЛЬНЫЙ АНАЛИЗ ЦИФРОВЫХ ПД-СТАБИЛИЗАТОРОВ ПОДВИЖНЫХ ОБЪЕКТОВ С НИЗКОЧАСТОТНЫМИ ФИЛЬТРАМИ БАТТЕРУОРТА И ЛАНЦОША

Рассматривается проблема построения цифровых ПД-стабилизаторов подвижных объектов с зашумленными высокочастотными помехами входными сигналами. Проведен сравнительный анализ стабилизаторов с фильтрами Баттеруорта и Ланцоша различных порядков.

Ключевые слова: цифровой ПД-стабилизатор, низкочастотный фильтр, дифференцирующий фильтр.

ПОСТАНОВКА ПРОБЛЕМЫ

В качестве чувствительных элементов систем стабилизации подвижных объектов, в частности, объектов военного назначения, обычно используется гироскопические платформы с нелинейной системой коррекции [1, 2]. В работе [3] показано, что оси таких гироплатформ совершают высокочастотные колебания, приводящие к зашумленности измеряемых параметров движения высокочастотными помехами. При разработке цифровых стабилизаторов подвижных объектов широкое распространение получили низкочастотные цифровые фильтры, подавляющие высокочастотные помехи. Среди большого разнообразия цифровых фильтров наиболее распространенными и эффективными являются низкочастотные фильтры Баттеруорта и Ланцоша [4]. В данной статье рассматривается проблема рационального использования цифровых фильтров различного порядка при построении ПД - стабилизаторов подвижных объектов.

АНАЛИЗ ПУБЛИКАЦИЙ ПО ТЕМЕ ИССЛЕДОВАНИЙ

В работах [4, 5] приведены АЧХ и ФЧХ фильтров Баттеруорта и Ланцоша различного порядка. Показано, что фильтры Баттеруорта, имея практически идеальную АЧХ, в смысле подавления высокочастотных помех, вносят значительное фазовое запаздывание при прохождении дискретного сигнала. Фильтры же Ланцоша, наоборот, вносят значительное фазовое опережение, возрастающее с повышением порядка фильтра и с улучшением его фильтрующих свойств. В работах [6, 7] получены аналитические соотношения для АЧХ и ФЧХ фильтров Баттеруорта и Ланцоша различных порядков, позволяющие произвести оценку динамических свойств цифровых стабилизаторов с данными фильтрами.

ЦЕЛЬ РАБОТЫ

Целью настоящей статьи является сравнительный анализ динамических характеристик ПД – стабилизаторов подвижных объектов при использовании цифровых фильтров различного порядка.

ОСНОВНАЯ ЧАСТЬ

Рассмотрим цифровой ПД – стабилизатор, структурная схема которого приведена на рис. 1.

Для рассмотренного стабилизатора можно записать

$$Z\{U[nT]\} = \{k_\varphi W_б(z) + k_{\dot\varphi} W_л(z)\} Z\{\varphi[nT]\}, \qquad (1)$$

где $Z\{.\}$ – символ Z-преобразования. Тогда дискретная передаточная функция ПД-стабилизатора, выполненного по параллельной схеме, равна

$$W_{ПД}(z) = \frac{Z\{U[nT]\}}{Z\{\varphi[nT]\}} = k_\varphi W_б(z) + k_\varphi W_л(z). \qquad (2)$$

Передаточные функции фильтров Баттеруорта и Ланцоша второго порядка записываются в виде [4]

$$W_б(z) = \frac{a_0 + a_1 z^{-1} + a_0 z^{-2}}{b_0 + b_1 z^{-1} + b_2 z^{-2}}; \qquad (3)$$

$$W_л(z) = c_0 + c_1 z^{-1} - c_1 z^{-3} - c_0 z^{-4}. \qquad (4)$$

Подставляя (3) и (4) в формулу (2), получим передаточную функцию ПД- стабилизатора, представленного на рис. 1.

$$W_{ПД}(z) = \frac{\begin{array}{l} k_\varphi(d_0 + d_1 z^{-1} + d_0 z^{-2}) + \\ + k_\varphi(f_0 + f_1 z^{-1} + f_2 z^{-2} + f_3 z^{-3}) - \\ - f_4 z^{-4} - f_5 z^{-5} - f_6 z^{-6} \end{array}}{1 + l_1 z^{-1} + l_1 z^{-2}}, \qquad (5)$$

где соответствующие постоянные связаны с параметрами фильтров (3) и (4) соотношениями

$$d_0 = \frac{a_0}{b_0};\ d_1 = \frac{a_1}{b_0};\ f_0 = c_0;\ f_1 = \frac{b_0 c_1 + b_1 c_0}{b_0};$$

$$f_2 = \frac{b_1 c_1 + b_2 c_0}{b_0};\ f_3 = \frac{c_1(b_2 + b_0)}{b_0};\ f_4 = \frac{b_0 c_0 + b_1 c_1}{b_0};$$

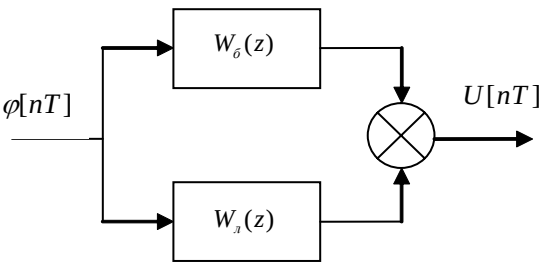


Рис. 1. ПД-стабилизатор, выполненный по параллельной схеме:

$W_б(z)$ – фильтр Баттеруорта; $W_л(z)$ – фильтр Ланцоша

$$f_5 = \frac{b_1 c_0 + b_2 c_1}{b_0};\ f_6 = \frac{b_2 c_0}{b_0};$$

$$l_1 = \frac{b_1}{b_0};\ l_2 = \frac{b_2}{b_0}.$$

К обеим частям соотношения (5) применим операцию обратного Z-преобразования, в результате чего получаем разностное уравнение рассматриваемого ПД-стабилизатора

$$\begin{array}{l} U[nT] = k_\varphi\{d_0\varphi[nT] + d_1\varphi[(n-1)T] + \\ + d_0\varphi[(n-2)T]\} + k_{\dot\varphi}\{f_0\varphi[nT] + \\ + f_1\varphi[(n-1)T] + f_2\varphi[(n-2)T] + \\ + f_3\varphi[(n-3)T] - f_4\varphi[(n-4)T] - \\ - f_5\varphi[(n-5)T] - f_6\varphi[(n-6)T]\} - \\ - l_1 U[(n-1)T] - l_2 U[(n-2)T]. \end{array} \qquad (6)$$

В качестве примера рассмотрим фильтры (3) и (4) с параметрами: a_0=0,08073; a_1=0,16147; b_0=1,48256; b_1=–1,83854; b_2=0,677890; c_0=5; c_1=2,5. Используя соотношение (6), построим АЧХ и ФЧХ рассматриваемого ПД-стабилизатора, приведенного на рис. 2, при различных значениях варьируемых параметров k_φ, $k_{\dot\varphi}$. Для этого на входе линейного цифрового стабилизатора (6) сформируем синусоидальную решетчатую функцию

$$\varphi[nT] = A\sin\omega nT. \qquad (7)$$

При этом на выходе стабилизатора , в соответствии с алгоритмом (6), имеет место синусоидальная решетчатая функция

$$U[nT] = B(\omega T)\sin[\omega(nT) + \varphi(nT)]. \qquad (8)$$

Для каждого текущего значения ωT вычисляется значение АЧХ

$$M(\omega T) = \frac{B(\omega T)}{A},$$

и измеряется фазовый сдвиг $\varphi(\omega T)$ выходного сигнала (8) относительно входного (7).

Анализ АЧХ и ФЧХ позволяет сделать вывод, что стабилизатор (6) плохо подавляет высокочастотные помехи и вносит в систему при низких частотах значительное фазовое опережение. Объясняется это тем , что в схеме, представленной на рис. 1, на фильтр Ланцоша подается зашумленный высокочастотными помехами входной сигнал, а подавление фильтром Ланцоша высоких частот не всегда эффективно.

В связи с изложенным, предлагается использование последовательно-параллельной схемы ПД-стабилизатора, представленной на рис. 3. В этой схеме на фильтр Ланцоша подается уже отфильтрованный низкочастотным фильтром Баттеруорта сигнал $\tilde\varphi(nT)]$.

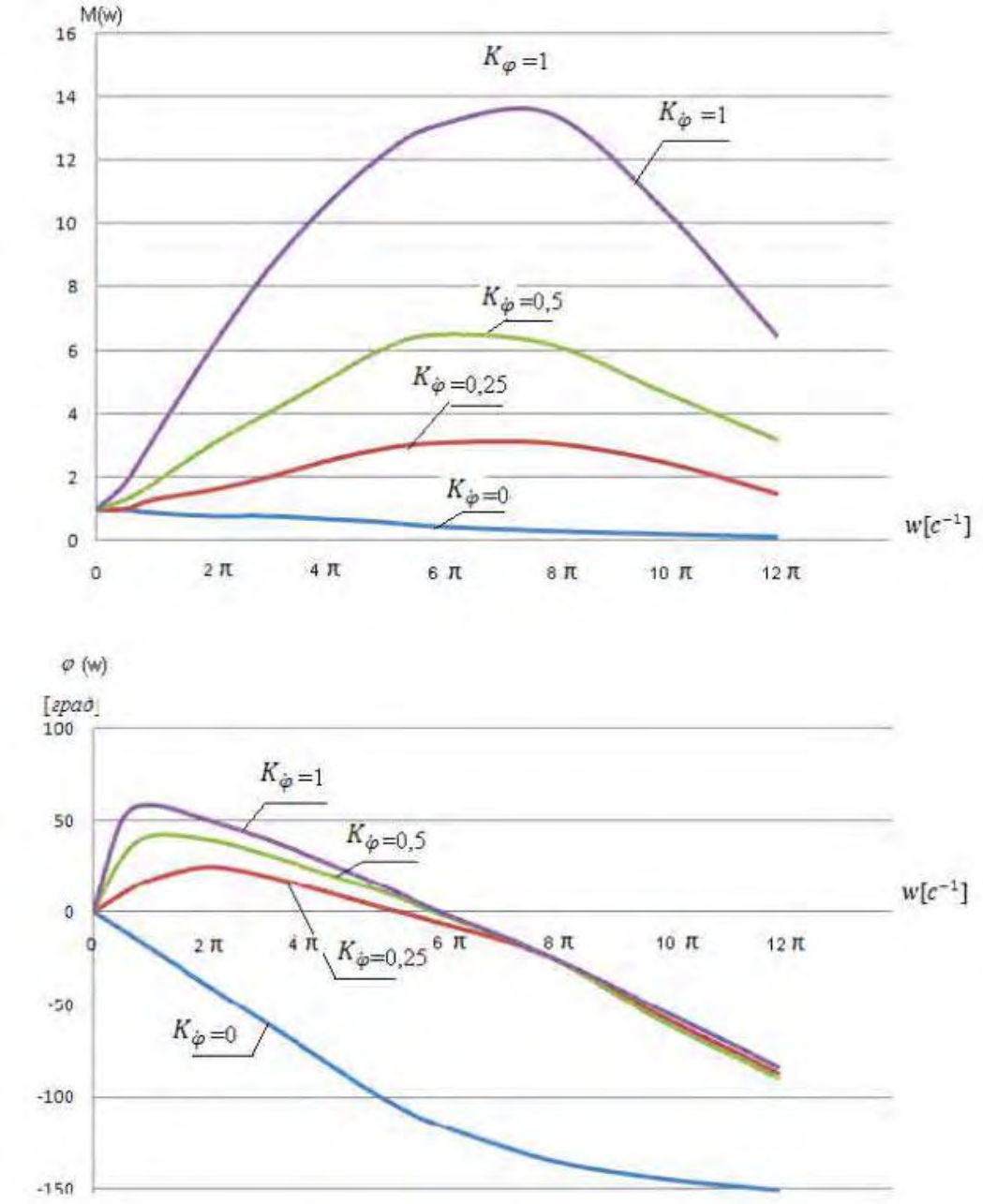


Рис. 2. АЧХ и ФЧХ ПД-стабилизатора, выполненного по параллельной схеме

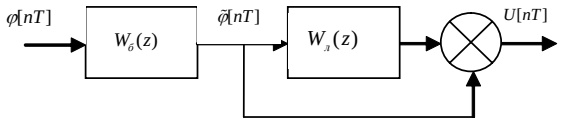


Рис. 3. ПД-стабилизатор, выполненный по последовательно-параллельной схеме

В этом случае передаточная функция стабилизатора при использовании фильтров второго порядка записывается в виде

$$W_{ПД}(z)=\frac{\begin{array}{l}k_\varphi\{d_0+d_1z^{-1}+d_0z^{-2}+\\+k_\varphi(f_0+f_1z^{-1}+f_2z^{-2}-\\-f_2z^{-4}-f_1z^{-5}-f_0z^{-6})\}\end{array}}{1+l_1z^{-1}+l_2z^{-2}}, \qquad (9)$$

где постоянные f_0, f_1 и f_2 равны:

$$f_0=\frac{a_0c_0}{b_0};\ f_1=\frac{a_0c_1+a_1c_0}{b_0};\ f_2=\frac{a_0c_0+a_1c_1}{b_0},$$

а соответствующее передаточной функции (9) разностное уравнение ПД-стабилизатора имеет вид

$$U[nT] = k_{\varphi}\{d_0\varphi[nT] + d_1\varphi[(n-1)T] + d_0\varphi[(n-2)T] + k_{\dot{\varphi}}(f_0\varphi[nT] + f_1\varphi[(n-1)T] + f_2\varphi[(n-2)T] - f_2[(n-4)T] - f_1\varphi[(n-5)T] - f_0\varphi[(n-6)T])\} - l_1U[(n-1)T] - l_2U[(n-2)T]. \quad (10)$$

На рис. 4 приведены АЧХ и ФЧХ стабилизатора (10). Подавление высоко-частотных помех стабилизатором (10) значительно эффективнее, чем стабилизатором (6), а фазовое опережение, вносимое фильтром Ланцоша, значительно ниже, чем при использовании стабилизатора (6). Кроме того, выбором значения коэффициента $k_{\dot{\varphi}}$ при определенной частоте фазовый сдвиг может быть установлен равным нулю. Обычно это целесообразно при собственной частоте колебаний подвижного объекта. Так, для объектов бронетехники собственная частота колебаний подрессоренной массы обычно составляет 1,2 – 1,5 Гц.

Анализ цифрового ПД-стабилизатора, выполненного по последовательно- параллельной схеме, показывает, что его высокие динамические свойства обусловлены тем, что на вход дифференцирующего фильтра Ланцоша подается выходной сигнал фильтра Баттеруорта,

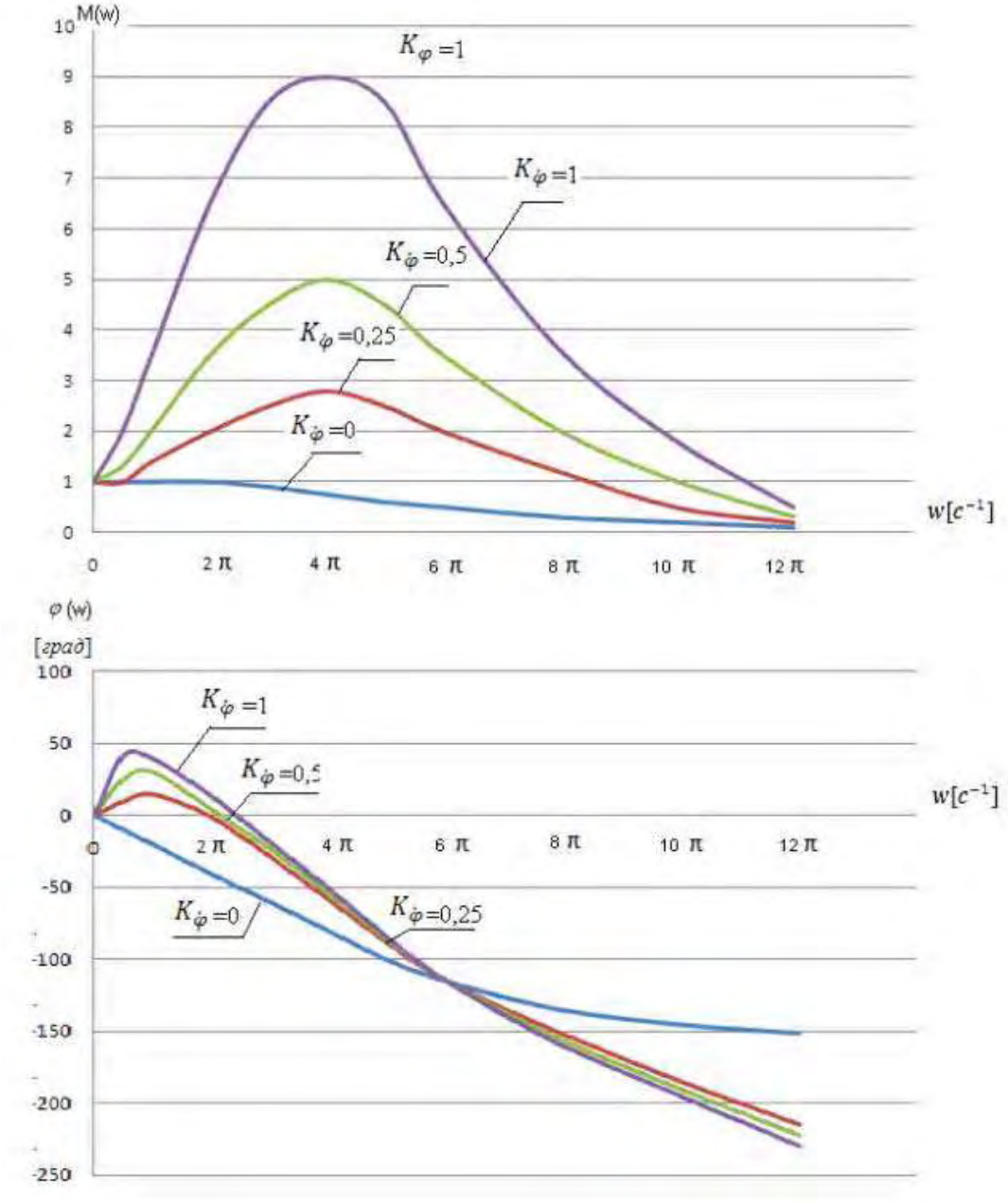


Рис. 4. АЧХ и ФЧХ ПД-стабилизатора выполненного по последовательно-параллельной схеме

который практически не содержит высокочастотных помех. Рассмотрим ПД-стабилизатор, представленный на рис. 3, с фильтром Баттеруорта третьего порядка с передаточной функцией

$$W_{\text{б}}(z)=\frac{a_0(1+3z^{-1}+3z^{-2}+z^{-3}}{1+d_{10}z^{-1}+d_{20}z^{-2}+b_{30}z^{-3}}, \qquad (11)$$

и фильтром Ланцоша первого порядка с передаточной функцией

$$W_{\text{б}}(z)=c_2(1-z^{-2}). \qquad (12)$$

В результате получаем передаточную функцию ПД-стабилизатора в виде

$$W_{\text{ПД}}(z)=\frac{\begin{array}{l}k_{\varphi}a_0\{1+3z^{-1}+3z^{-2}+z^{-3}+\\+k_{\varphi}c_2(1+3z^{-1}+2z^{-2}-2z^{-3}-\\-3z^{-4}-z^{-5})\}\end{array}}{1+d_{10}z^{-1}+d_{20}z^{-2}+d_{30}z^{-3}}, \qquad (13)$$

Соответствующий передаточной функции (13) алгоритм стабилизации записывается в виде

$$\begin{array}{c}U[nT]=k_{\varphi}a_{10}\{\varphi[nT]+3\varphi[(n-1)T]+\\+3\varphi[(n-2)T]+\varphi[(n-3)T]+\\+k_{\dot{\varphi}}c_2(\varphi[nT]+3\varphi[(n-1)T]+\\+2\varphi[(n-2)T]-2[(n-3)T]-\\-3\varphi[(n-4)T]-\varphi[(n-5)T])\}-\\-d_{10}U[(n-1)T]-d_{20}U[(n-2)T]-\\-d_{30}U[(n-2)T]. \qquad (14)\end{array}$$

Пусть параметры фильтров (11) и (12) составляют a_{10}=0,012045; d_{10}=–1,93847; d_{20}=1,37229; d_{30}=–0,33745; c_2=12,5.

Расчет АЧХ и ФЧХ ПД-стабилизатора (14) позволяет сделать вывод о том, что динамические характеристики стабилизаторов (10) и (14) практически совпадают. Некоторое ухудшение фильтрующих свойств фильтра Ланцоша первого порядка по сравнению с фильтром второго порядка компенсируется использованием фильтра Баттеруорта третьего порядка с практически идеальной АЧХ, который весьма эффективно фильтрует высокочастотные помехи.

Таким образом, цифровые ПД-стабилизаторы подвижных объектов целесообразно формировать по последовательно-параллельной схеме в соответствии с алгоритмами стабилизации (10) и (14).

СПИСОК ЛИТЕРАТУРЫ

1. *Корнеев, В. В.* Основы автоматики и танковые автоматические системы / В. В. Корнеев, Л. П. Кузьмин, К. И. Павличук, М. И. Кузнецов. – М. : Академия БТВ, 1976. – 545 с.
2. *Аблесімов, О. К.* Автоматичне керування рухомими об'єктами і технологічними процесами. т. 3. Автоматичне керування озброєнням танків / О. К. Аблесімов, Є. Є. Александров, І. Є. Александрова. – Харків : НТУ «ХПІ», 2008. – 444 с.
3. *Александров, Е. Е.* Оценка точности стабилизации поля зрения прицела танковой пушки / Е. Е. Александров, Т. Е. Александрова, А. А. Лазаренко // Артиллерийское и стрелковое вооружение. – 2011. – № 3. – с. 40–44.
4. *Хемминг, Р. В.* Цифровые фильтры / Р.В. Хемминг. – М. : Недра, 1984. – 221 с.
5. *Ивашко, А. В.* Методы и алгоритмы цифровой обработки сигналов / А. В. Ивашко. – Харьков: НТУ «ХПИ», 2005. – 240 с.
6. *Александров, Е. Е.* Синтез цифровых нерекурсивных фильтров для информационно-управляющих систем / Е. Е. Александров, Т. Е. Александрова, В. А. Кононенко / / Техническая электродинамика. Специальный выпуск. Силовая электроника и энергоэффективность. – 2011 – т. 1. – с. 163–168.
7. *Александров, Е. Е.* Сравнительный анализ цифровых дифференцирующих фильтров / Е. Е. Александров, Т. Е. Александрова, В. А. Кононенко // В кн. : Автоматика / Automatics – 2011. XVIII Міжнародна конференція з автоматичного управління. Матеріали конференції. – Львів : Видавництво Львівської політехніки, 2011. – С. 303–304.

Стаття надійшла до редакцій 25.05.2011.

Александрова Т. Є., Кононенко В. О., Лазаренко А. О., Зейн Алі

ПОРІВНЯЛЬНИЙ АНАЛІЗ ЦИФРОВИХ ПД-СТАБІЛІЗАТОРІВ РУХОМИХ ОБ'ЄКТІВ З НИЗЬКОЧАСТОТНИМИ ФІЛЬТРАМИ БАТТЕРУОРТА І ЛАНЦОША

Розглядається проблема побудови цифрових ПД-стабілізаторів рухомих об'єктів із спотвореними завадами вхідними сигналами. Проведено порівняльний аналіз стабілізаторів з фільтрами Баттеруорта і Ланцоша різних порядків.

Ключові слова: цифровий ПД-стабілізатор, низькочастотний фільтр, диференцуючий фільтр.

Alexandrova T. Ye., Kononenko V. A., Lazarenko A. A., Zein Ali

COMPARATIVE ANALYSIS OF DIGITAL PD-CTABILIZERS OF MOBILE OBJECTS WITH LOW-FREQUENCY BUTTERWORT AND LANCZOS FILTERS

The problem of constructing a digital PD-stabilizers of mobile objects with noise input signals is discussed . The comparative analysis of the stabilizers with Butterwort and Lanczos filters of the various orders is considered.

Key words: digital PD-stabilizer, low-pass filter, differentiating filter.

УДК 621.34-52

Бур'ян С. О.[1], Грищук Т. В.[2]

[1]Асистент Національного технічного університет України «Київський політехнічний інститут»
[2] Магістрант Національного технічного університет України «Київський політехнічний інститут»

ЕКСТРЕМАЛЬНА ЕЛЕКТРОМЕХАНІЧНА СИСТЕМА КЕРУВАННЯ ПАРАЛЕЛЬНО З'ЄДНАНИМИ НАСОСАМИ ВОДОПОСТАЧАННЯ

У статті йдеться про застосування екстремального енергоефективного керування для двох паралельно з'єднаних насосів. Один з насосів укомплектований частотно-керованим асинхронним електроприводом. Представлено опис алгоритму екстремального керування, його математична модель та структурна схема системи.

Ключові слова. Насос водопостачання, електромеханічна система, екстремальне керування, енергоефективність, регульований електропривод, паралельне з'єднання насосів.

ВСТУП

Системи водопостачання та насосні установки відносяться до числа найбільш енергоспоживаючих технологічних об'єктів. У зв'язку з постійним подорожчанням та вичерпуванням енергетичних та водних ресурсів питанням енергоефективного керування насосних установок почали приділяти значної уваги. Саме тому, все більш актуальним стає питання розробки та впровадження енергоощадливих та енергоефективних технологій у даній сфері. На практиці широке розповсюдження знайшли системи стабілізації напору та програмне керування насосами на основі графіків добового водоспоживання [1]. Більша енергоефективність може бути досягнута при максимально можливому зменшенні швидкості, яке допустиме для конкретного режиму водоспоживання, та забезпеченні продуктивності насоса, яка відповідає потребам споживачів. Так звані «інтерактивні» алгоритми керування забезпечують автоматичний пошук мінімально-достатньої швидкості насосу, яка задовольняє потреби споживачів [2, 3]. Розглянуті вище способи забезпечують певний рівень енергозбереження, але оскільки робоча точка не завжди опиняється на лінії максимального ККД при використанні вищерозглянутих алгоритмів керування, енергоефективність системи може знижуватися. Ця задача розв'язується за допомогою екстремальної системи керування ККД насосної установки [4]. Також використовуються двоканальні екстремальні системи автоматичного керування насосними установками [5], що дозволяють не тільки підвищити ККД насосу, але й зменшити втрати у привідному двигуні. Але у багатьох випадках системи водопостачання не обмежуються одним насосом. Застосовуються послідовне, паралельне та змішане з'єднання для реалізації певних режимів роботи. Застосування екстремального керування для послідовного з'єднання насосів, один з яких керований за швидкістю, дозволяє підвищити ККД керованого насосу [4]. Цей алгоритм може бути застосований також для паралельного з'єднання агрегатів з метою підвищення енергоефективності системи в цілому.

МЕТА РОБОТИ

Метою роботи є підвищення енергоефективності електромеханічної системи автоматичного керування паралельно з'єднаними насосами водопостачання шляхом розробки екстремального алгоритму керування ККД керованого та некерованого насосів. Працездатність системи перевіряється шляхом математичного моделювання.

ЕКСТРЕМАЛЬНИЙ АЛГОРИТМ КЕРУВАННЯ ККД КЕРОВАНОГО ЗА ШВИДКІСТЮ НАСОСА ДЛЯ ПАРАЛЕЛЬНО З'ЄДНАНИХ АГРЕГАТІВ

Розглянемо роботу екстремального алгоритму для паралельного з'єднання агрегатів, враховуючи, що один з насосів керований за швидкістю, інший – некерований. Нехай робоча точка керованого насосу A_1 знаходиться зліва від лінії максимального ККД на характеристиці 1, як показано на рис. 1. Характеристика некерованого насосу має вигляд 2, сумарна – 3. У деякий момент часу екстремальний енергоефективний контролер зменшує швидкість обертання регульованого насосу на фіксоване значення $\Delta\omega$. Його напірна характеристика позначена 5, а сумарна характеристика обох насосів – 7. Новими робочими точками відповідно стануть B, B_1 і B_2. З метою стабілізації продуктивності до значення Q_A споживачі змушені зменшити гідравлічний опір мережі до значення, при якому характеристика мережі прийме вигляд 6. Робочі точки перейдуть у положення C, C_1 та C_2 (для нерегульованого насосу H2). На наступному кроці алгоритму контролер знову зменшить оберти першого насосу на фіксоване значення $\Delta\omega_1$ (характеристика 9, сумарна – 8) і перемістить робочі точки в D, D_1 та D_2 (для насосу H2). Це викличе відповідну реакцію споживачів до стабілізації витрат води.

Такий процес відбуватиметься доти, поки робоча точка насосу H1 не виявиться праворуч кривої максимального ККД (точка G_1). Після цього контролер фіксовано збільшить оберти цього насосу, що призведе до збільшення продуктивності та необхідності споживачам прикри-

вати крани. У результаті робота першого насосу характеризуватиметься циклічною послідовністю наступних робочих точок $E_1 – F_1 – G_1 – H_1 – E_1$. Для нерегульованого насосу послідовність буде наступною $E_2 – F_2 – G_2 – F_2 – E_2$. У результаті роботи алгоритму ККД некерованого насоса буде коливатися в околі значення, що визначається продуктивністю в точці E_2, а ККД керованого коливатиметься в околі максимального значення.

Якщо споживачі не реагують на зміну швидкості (не змінюють величину гідравлічного опору), то вона буде зменшуватися доти, поки робоча точка першого насосу не виявиться нижче кривої максимального ККД. Якщо траєкторія руху робочої точки першого насосу не перетне криву максимального ККД, то швидкість зменшуватиметься до мінімально-дозволеного значення. У випадку відсутності обмеження витрати можуть знизитися до нуля.

МАТЕМАТИЧНА МОДЕЛЬ ЕКСТРЕМАЛЬНОГО КОНТРОЛЕРА

На основі вищевикладеного алгоритму керування та теорії екстремальних систем структурна схема контролера має вигляд, що показаний на рис. 2.

На структурній схемі введені наступні позначення: Q_1 – фактична виміряна продуктивність на виході регульованого насосу; ω_1^* – бажана швидкість обертання двигуна, при якій забезпечується максимальний ККД при бажаному значенні продуктивності; ω_1 – фактична виміряна швидкість двигуна; T_0 – період квантування екстремального контролера; Δf^* – необхідна величина зменшення завдання частоти; K – параметр, що задає амплітуду зміни завдання частоти.

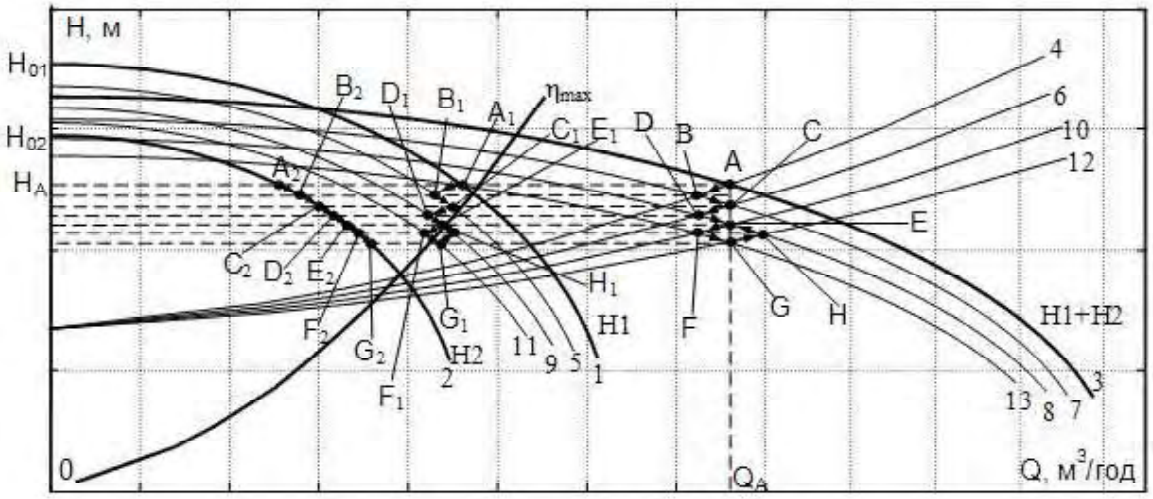


Рис. 1. Робота екстремального алгоритму при паралельному з'єднання насосних агрегатів

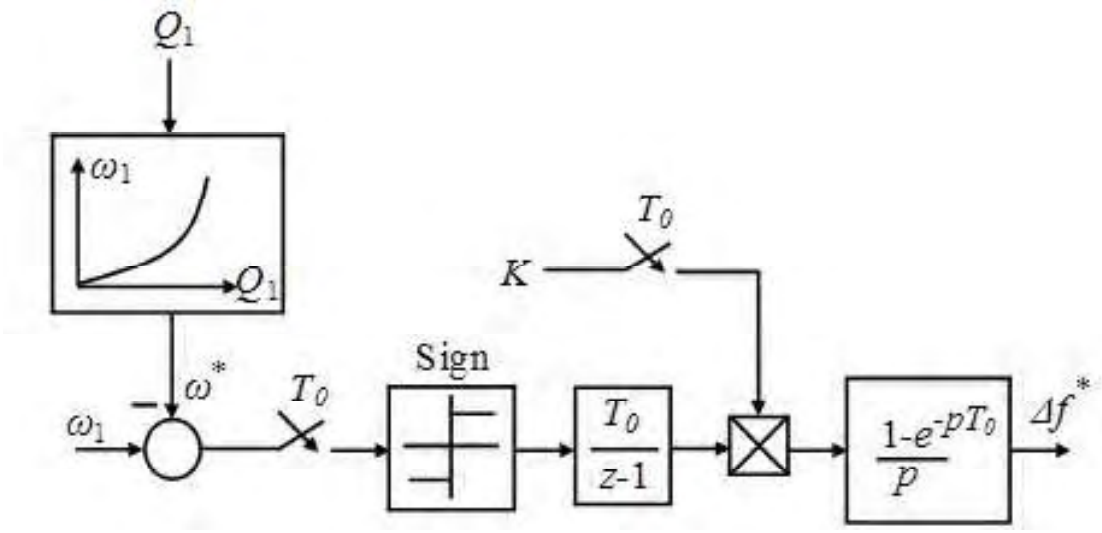


Рис. 2. Структурна схема екстремального алгоритму при паралельному з'єднання насосних агрегатів

Рівняння для вихідної заданої частоти на основі структурної схеми матиме вигляд

$$\Delta f = \frac{KT_0}{z-1}\text{sign}(\omega_1 - \omega_1^*). \quad (1)$$

Перетворимо рівняння (1) для запису у різницях

$$\Delta f((n+1)T_0) - \Delta f(nT_0) = KT_0\text{sign}(\omega_1(nT_0) - \omega_1^*(nT_0)). \quad (2)$$

Враховуючи, що бажана швидкість ω_1^* визначається на основі величини бажаної продуктивності при максимальному ККД, рівняння (2) переписується у вигляді

$$\Delta f((n+1)T_0) = \Delta f(nT_0) + KT_0\text{sign}(\omega_1(nT_0) - f(Q_1(nT_0))). \quad (3)$$

ОСОБЛИВОСТІ ЗАСТОСУВАННЯ ЕКСТРЕМАЛЬНОГО АЛГОРИТМУ КЕРУВАННЯ ККД КЕРОВАНОГО ЗА ШВИДКІСТЮ НАСОСА ДЛЯ ПАРАЛЕЛЬНО З'ЄДНАНИХ АГРЕГАТІВ З ЖОРСТКИМИ НАПІРНИМИ ХАРАКТЕРИСТИКАМИ

У випадку, якщо напірні характеристики насосів мають велику жорсткість, траєкторія руху робочої точки може не перетнути лінію максимального ККД насосу при застосуванні екстремального алгоритму через достатньо велику зміну величини продуктивності навіть при невеликій зміні частоти обертання, як показано на рис. 3.

Як видно з рисунку, за рахунок майже лінійної ділянки сумарних напірних характеристик, робоча точка регульованого насосу буде рухатися по траєкторії $A_1 - B_1 - C_1 - D_1 - E_1$, у той час як робоча точка на виході системи буде рухатися по траєкторії A –B* –B –C* –C – D* – D – E*– E. Точками на траєкторії регульованого насосу, яким відповідають точки B*, C*, D* та E* можна знехтувати, враховуючи велику жорсткість напірних характеристик. Траєкторія руху робочої точки нерегульованого насосу не зміниться від зображеної на рис. 1. Подальший такий рух без обмеження за швидкістю може призвести до того, що робоча точка регульованого насосу перейде на вісь H, тобто його продуктивність стане нульовою. Вихідна продуктивність буде дорівнювати продуктивності нерегульованого насосу.

Якщо робоча точка буде знаходитися справа від лінії максимального ККД, вона також не опиниться на ній через роботу алгоритму на збільшення швидкості у цій зоні.

У такому випадку екстремальний алгоритм необхідно скоректувати. Розташування робочої точки зліва від лінії максимального ККД не призведе до підвищення ефективності, тому розглянемо випадок, коли остання розташована справа. Якщо змінити напрям роботи алгоритму у цій зоні, то при зменшенні швидкості робоча точка може опинитися на лінії максимального ККД. Рівняння регулятора з урахуванням зміни напрямку роботи алгоритму матиме вигляд

$$\Delta f((n+1)T_0) = \Delta f(nT_0) + KT_0\text{sign}(f(Q_1(nT_0)) - \omega_1(nT_0)). \quad (4)$$

Принцип роботи регулятора зображений на рис. 4. Як видно з рис. 4, траєкторія робочих точок $A_1 - B_1 - C_1 - D_1$ регульованого насосу перетинає лінію максимального ККД за умови, що рух починається зліва і споживачі реагують на зміну продуктивності. Після перетину лінії максимального ККД ця траєкторія буде визначатися послідовними переходами з точки D_1 в C_1 і навпаки, які забезпечують коливання в зоні максимального ККД регульованого насосу. Незначними переходами, що відбуваються при стабілізації споживачами продуктивності нехтуємо, при умові, що напірні характеристики мають досить високу жорсткість.

Таке розташування вихідної робочої точки найчастіше застосовується у системах, що містять паралельне з'єднання насосів, для збільшення вихідної продуктивності. Тому екстремальний алгоритм у такій формі найкраще підходить для паралельного з'єднання насосів з жорсткими напірними характеристиками.

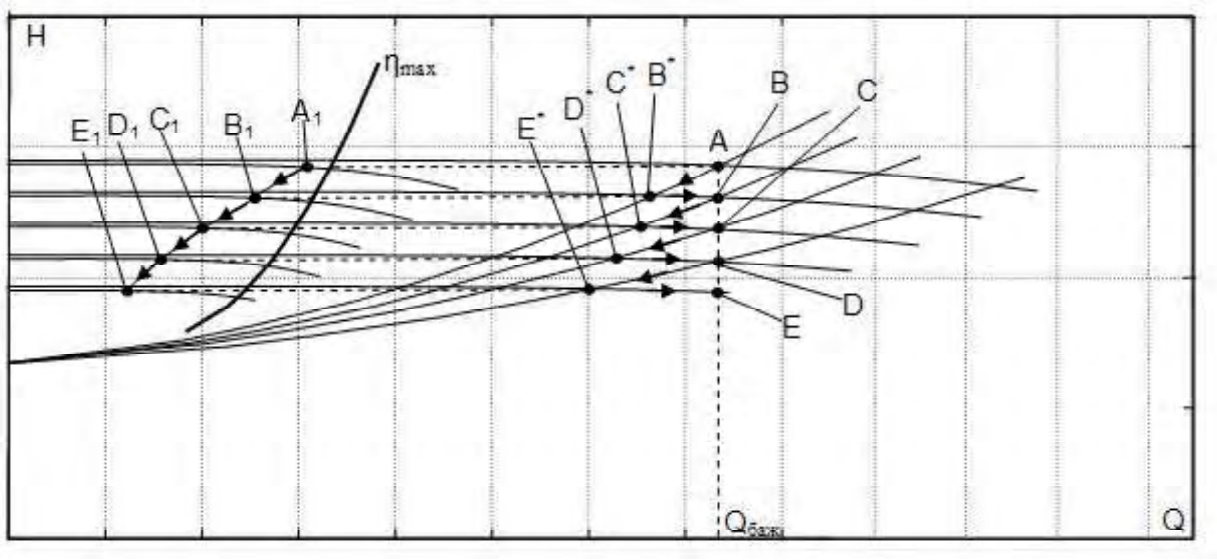


Рис. 3. Траєкторія руху робочої точки регульованого насосу з жорсткими напірними характеристиками

ЕКСТРЕМАЛЬНИЙ АЛГОРИТМ КЕРУВАННЯ ККД НЕКЕРОВАНОГО ЗА ШВИДКІСТЮ НАСОСА ДЛЯ ПАРАЛЕЛЬНО З'ЄДНАНИХ АГРЕГАТІВ

При паралельному з'єднанні насосних агрегатів можливий варіант реалізації екстремального керування ККД для некерованого за швидкістю насосом за рахунок зміни швидкості керованого.

Структурна схема екстремального регулятора представлена на рис. 5.

На структурній схемі введені наступні позначення: Q_2 – фактична виміряна продуктивність нерегульованого насосу; Q_2^* – продуктивність нерегульованого насоса при якій забезпечується максимальний ККД (для окремого насоса – постійна величина, та як його швидкість незмінна, а на кожній напірній характеристиці існує тільки одне значення продуктивності, при якій забезпечується максимальний ККД); T_0 – період квантування екстремального контролера; Δf^* – необхідна величина зменшення завдання частоти регульованого насосу; K – параметр, що задає амплітуду зміни завдання частоти.

Рівняння для вихідної заданої частоти на основі структурної схеми матиме вигляд

$$\Delta f = \frac{KT_0}{z-1}\text{sign}(Q_2 - Q_2^*). \quad (5)$$

Перетворимо рівняння (5) для запису у різницях

$$\Delta f((n+1)T_0) = \Delta f(nT_0) + KT_0\text{sign}(Q_2(nT_0) - Q_2^*(nT_0)). \quad (6)$$

Алгоритм для такого варіанту ілюструється на рис. 6.

Нехай робоча точка A_2 знаходиться зліва від лінії максимального ККД нерегульованого насосу Н2. У момент часу T_0 екстремальний енергоефективний контролер зменшує швидкість обертання регульованого насосу на фіксоване значення $\Delta\omega$. Його напірна характеристика позначена 5, а сумарна характеристика обох насосів – 7. Робоча точка нерегульованого насосу переходить у B_2.

З метою стабілізації продуктивності до значення Q_A споживачі змушені зменшити гідравлічний опір мережі до значення, при якому характеристика мережі прийме вигляд 6. Робочі точки перейдуть у положення C, C_1 та C_2 (для нерегульованого насосу Н2). На наступному кроці

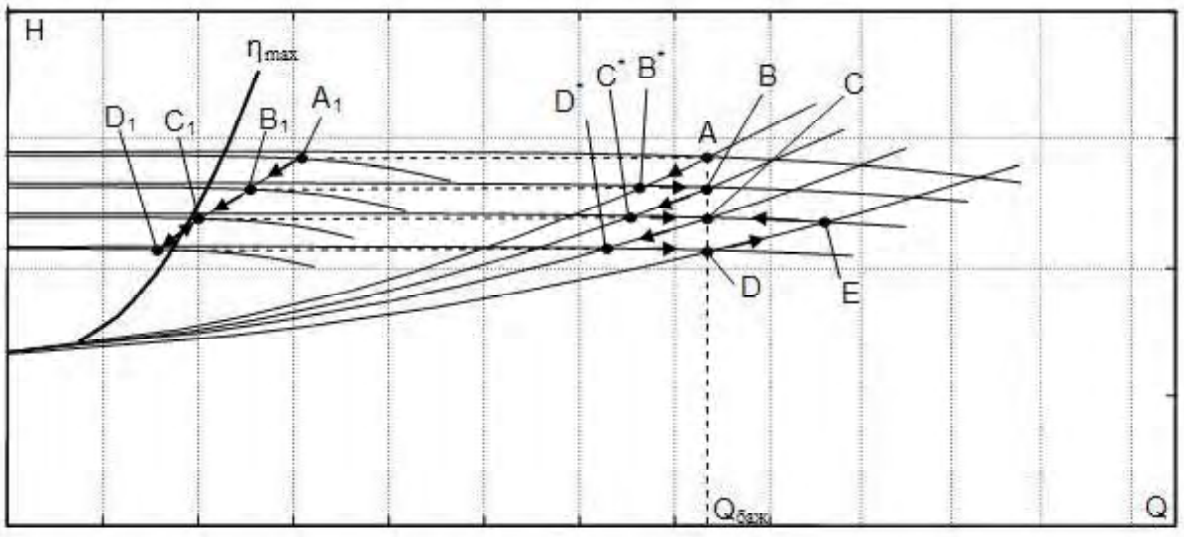


Рис. 4. Траєкторія руху робочої точки регульваного насосу з жорсткими напірними характеристиками для другої зони

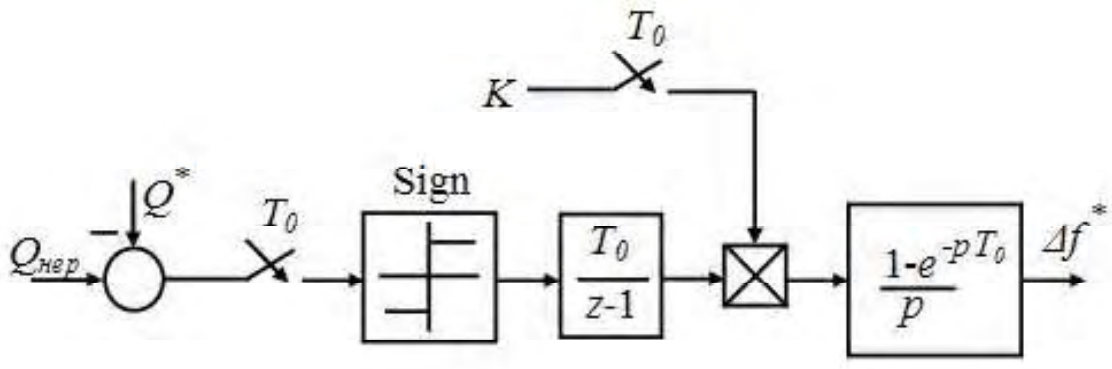


Рис. 5. Структурна схема екстремального енергоефективного регулятора ККД для нерегульованого за швидкістю насосу

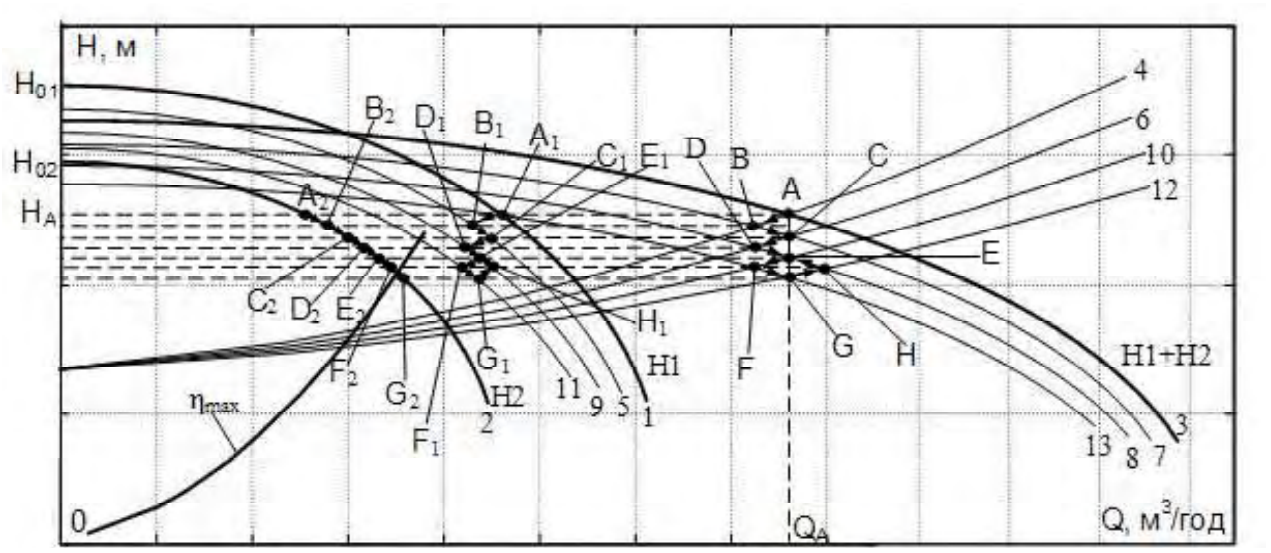


Рис. 6. Робота екстремального алгоритму при паралельному з'єднанні насосних агрегатів для нерегульованого за швидкістю насосу

алгоритму $2T_0$ контролер знову зменшить оберти першого насосу на фіксоване значення $\Delta\omega_1$ (характеристика 9, сумарна – 8) і перемістить робочі точки в D, D_1 та D_2 (для насосу Н2). Це викличе відповідну реакцію споживачів до стабілізації витрат води.

Такий процес відбуватиметься доти, поки робоча точка насосу Н2 не виявиться праворуч кривої максимального ККД (точка G_2), тобто Q_2 стане більше Q_2^*. Після цього контролер фіксовано збільшить оберти цього насосу, що призведе до збільшення продуктивності та необхідності споживачам прикривати крани. У результаті робота першого насосу характеризуватиметься циклічною послідовністю наступних робочих точок $E_1 - F_1 - G_1 - H_1 - E_1$. Для нерегульованого насосу послідовність буде наступною $E_2 - F_2 - G_2 - F_2 - E_2$. У результаті роботи алгоритму ККД некерованого насоса буде коливатися в околі значення, що визначається продуктивністю в точці G.

Якщо споживачі не реагуватимуть на зміну продуктивності, то за рахунок руху по напірній характеристиці вправо, робоча точка також опиниться за лінією максимального ККД і далі буде рухатися в її околі за рахунок зміни швидкості обертання керованого насосу.

Якщо вихідна робоча точка опиниться у другій зоні, то алгоритм буде працювати аналогічно на збільшення швидкості керованого насосу.

МАТЕМАТИЧНА МОДЕЛЬ ЕКСТРЕМАЛЬНОЇ СИСТЕМИ АВТОМАТИЧНОГО КЕРУВАННЯ ПАРАЛЕЛЬНО З'ЄДНАНИМИ НАСОСАМИ

Математична модель двох паралельно з'єднаних насосів описується наступними рівняннями [6]:

$$H = H_{01н}\omega_1^2 / \omega_{н1}^2 - a_{н1}Q_1^2 - \chi_1 dQ_1 / dt, \qquad (7)$$

$$H = H_{02н}\omega_2^2 / \omega_{н2}^2 - a_{н2}Q_2^2 - \chi_2 dQ_2 / dt, \qquad (8)$$

$$H = H_{cm} + aQ^2 - \chi dQ / dt, \qquad (9)$$

$$Q = Q_1 + Q_2, \qquad (10)$$

$$M_{C1} = \rho g Q_1 H / \eta_1 \omega_1, \qquad (11)$$

$$M_{C2} = \rho g Q_2 H / \eta_2 \omega_2, \qquad (12)$$

де Q_1, Q_2 – продуктивності першого та другого насосів відповідно; Q – сумарна продуктивність першого та другого насосу; $H_{01н}$ та $H_{02н}$ – номінальні напори при нульових подачах першого та другого насосів при номінальних швидкостях відповідно; ω_1, ω_2 – швидкості обертання першого та другого насосів відповідно; $\omega_{н1}$, $\omega_{н2}$ – номінальні швидкості обертання першого та другого насосів відповідно; χ_1, χ_2 – сталі часу інтегрування першого та другого насосів відповідно; χ – загальна стала часу інтегрування; H_{cm} – геодезична висота підйому води; $a_{н1}$, $a_{н2}$ – номінальні гідравлічні опори першого та другого насосів відповідно; a – гідравлічний опір мережі; M_{C1}, M_{C2} – моменти навантаження на валах двигунів першого та другого насосів відповідно; ρ – густина води; g – прискорення вільного падіння; η_1, η_2 – ККД першого та другого насосів відповідно; H – напір першого та другого насосу; t – час.

Наявність зворотних клапанів на виході кожного з насосів враховується наступним чином: якщо продуктивність насоса стає від'ємною, то вона приймається рівною нулю.

ККД насосів є нелінійними функціями, які залежать від положень робочих точок на напірних характеристиках $\eta_i = f(Q_i, H_i)$. В роботі ці залежності апроксимуються двошаровими нейронними мережами типу «feed-forward backpropagation» з 10, 1 нейронами та функціями активації tansig, purelin у відповідних шарах. Для навчання нейронних мереж використана база даних 4000 робочих точок з каталогу фірми виробника [7].

Крива максимального ККД керованого за обертами насоса також апроксимована аналогічною нейронною мережею, навченою на основі даних 250 робочих точок.

Привідні асинхронні двигуни насосів описуються нелінійними двофазними моделями в нерухомій системі координат статора a–b [8]. Перетворювач частоти реалізує квадратичний закон керування u/f^2=const [4].

Структурна схема екстремальної системи керування показана на рис. 7.

ДОСЛІДЖЕННЯ ЕКСТРЕМАЛЬНОЇ СИСТЕМИ КЕРУВАННЯ

Для дослідження вибрано два однакові насоси фірми Vogel Pumpen CNX-100-65-400 [7] з наступними максимальними даними: потужність 156,1 кВт, напір 240 м, продуктивність 164,9 м^3/год, ККД 62,9 %, частота обертання 2900 об/хв. Номінальні дані привідних асинхронних двигунів наступні: потужність 160 кВт, синхронна частота обертання 3000 об/хв, ККД 96 %, лінійна напруга 380 В, коефіцієнт потужності 0,9, ковзання 0,019, активний опір статора 0,0117 Ом, індуктивність статора 0,0126 Гн, активний опір ротора 0,0094 Ом, індуктивність ротора 0,0127 Гн, взаємна індуктивність статора та ротора 0,0124 Гн. При моделюванні прийнято $H_{cm} = 150$ м, $\chi = 0{,}5$ с/(м^2/год), $\chi_1 = \chi_2 = 0{,}05$ с/(м^2/год), приведений момент інерції на валах двигунів J=1,4 кг м^2. Розрахункові параметри насосів на основі напірних характеристик є наступними: $H_{01} = H_{02} = 235$ м, $a_{н1} = a_{н2} = 9018$ м/(м^3/с)^2. Напірні характеристики насосів досить жорсткі, тому під час моделювання використані алгоритми (4) та (6).

Результати досліджень екстремальної системи керування ККД керованого насосу представлені на рис. 8, а ККД некерованого насосу – на рис. 9.

При дослідженні екстремальної системи керування ККД керованого насосу вихідна робоча точка приймалась такою, щоб робоча точка керованого насосу знаходилася праворуч від кривої максимального ККД. Поведінка споживачів моделювалася як система стабілізації продуктивності з ПІ-регулятором. Результати досліджень показали, що швидкість керованого насосу ω_1 зменшується, доки робоча точка не опиниться зліва від лінії максимального ККД, при цьому задана швидкість ω_1^* починає зменшуватися а швидкість двигуна ω_1 збільшуватись. У результаті ККД керованого насосу η_1 буде коливатися навколо максимального значення, а ККД некерованого насосу η_2 буде коливатися навколо деякого значення, яке залежить від початкового розташування робочої точки. Траєкторія руху робочої точки відповідає алгоритму (4).

Підключення регулятора ККД некерованого насосу показано пунктирними лініями на рис. 7. При дослідженні

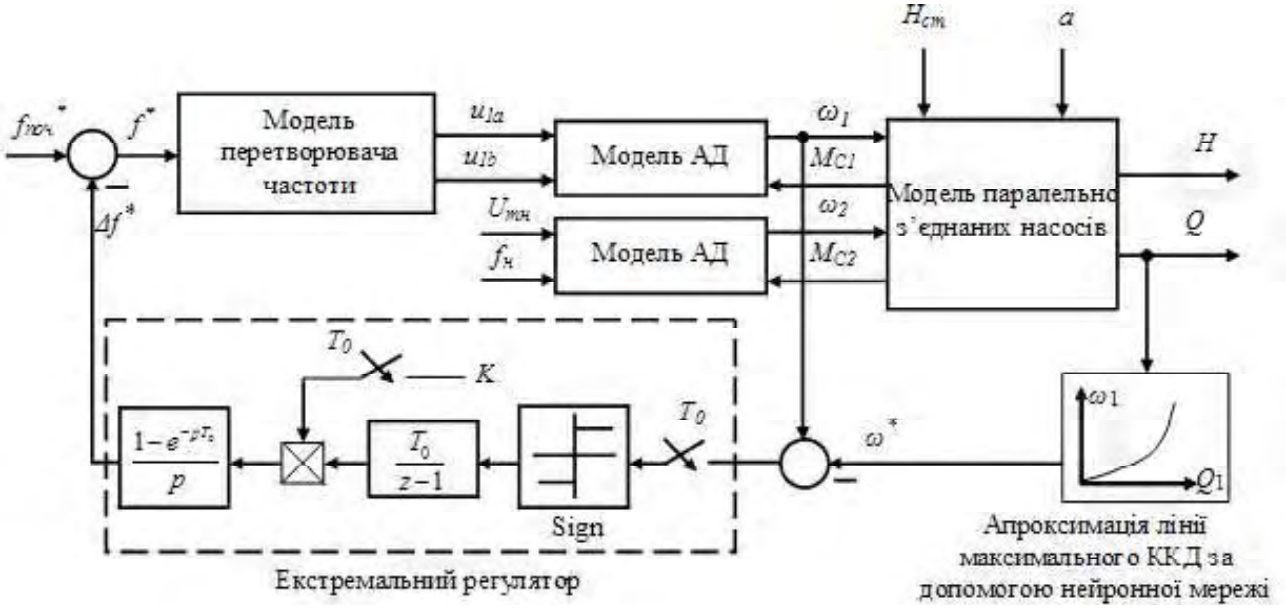


Рис. 7. Структурна схема екстремальної системи керуваненя

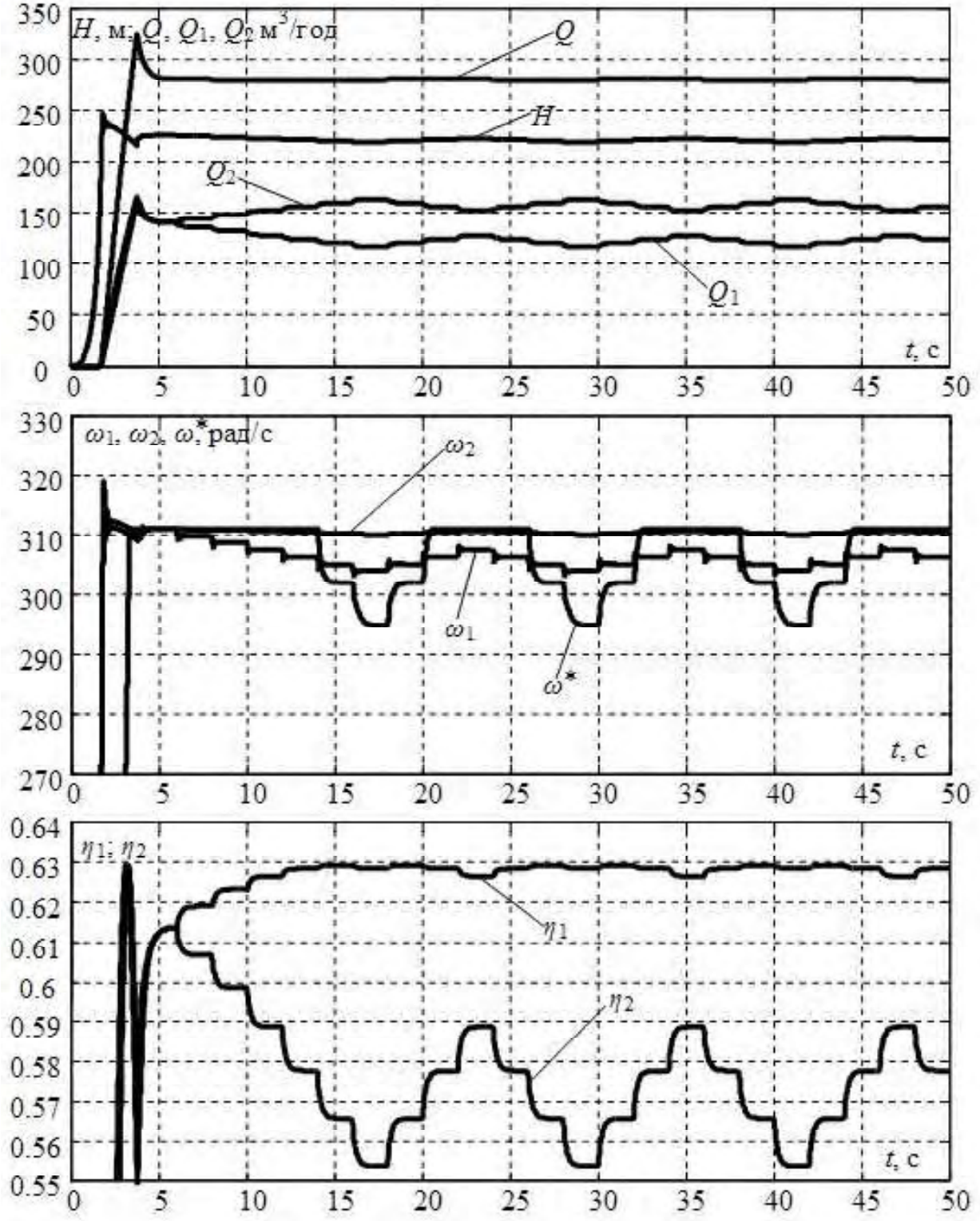


Рис. 8. Дослідження екстремальної системи керування ККД керованого насосу

системи керування вихідна робоча точка насосу знаходилася ліворуч від кривої максимального ККД. Результати досліджень показали, що швидкість керованого насосу ω_1 зменшується, доки продуктивність некерованого насосу Q_2 не стане більше за задану продуктивність Q_2^*. У результаті ККД некерованого насосу η_2 буде коливатися навколо максимального значення, а ККД керованого насосу η_1 буде коливатися навколо деякого значення, яке залежить від початкового розташування робочої точки.

При застосуванні пропонованого алгоритму у гідравлічних системах, що наповнюють великі резервуари, ви-

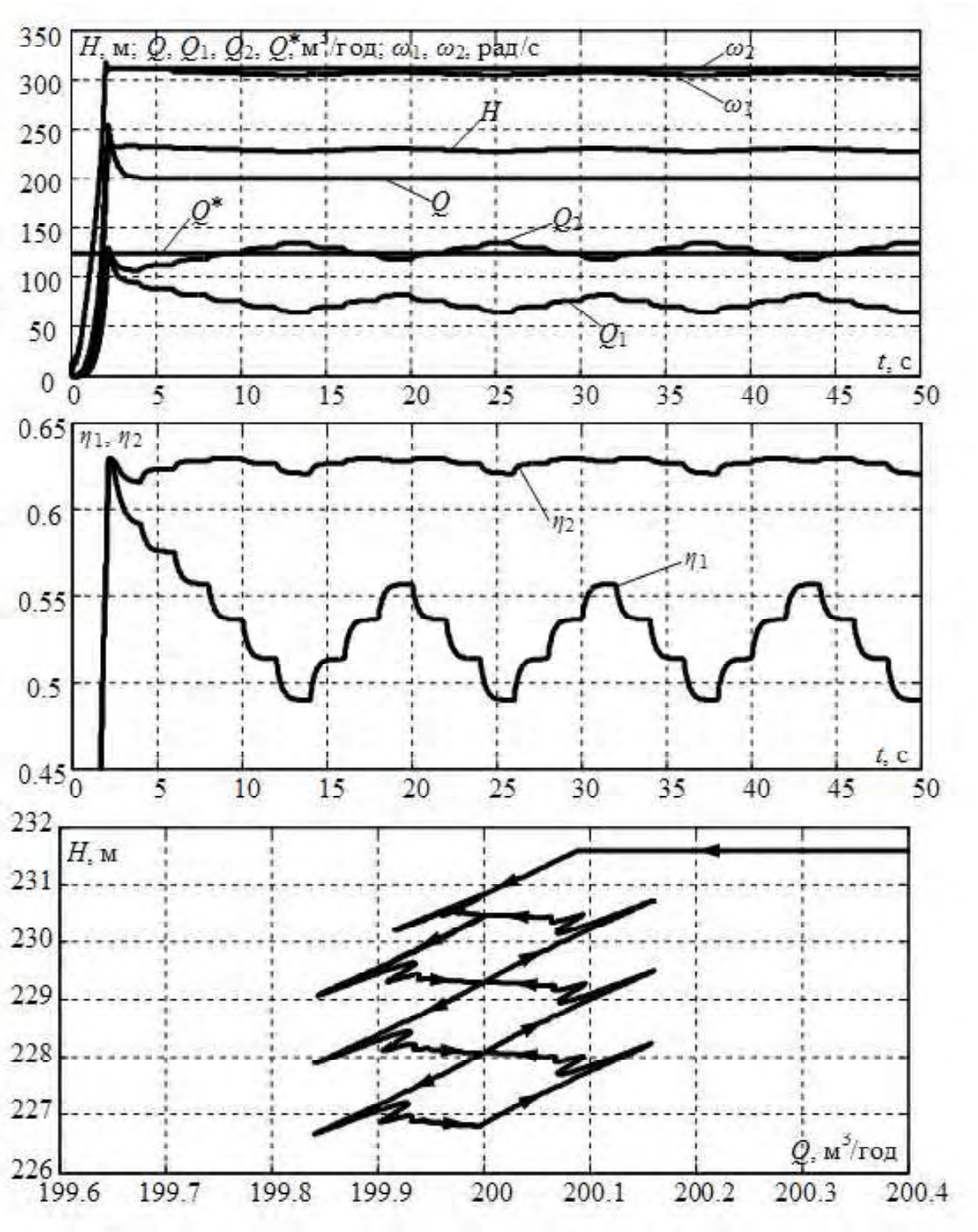


Рис. 9. Дослідження екстремальної системи керування ККД некерованого насосу

моги, щоб споживачі стабілізували продуктивність, є необов'язковими. Дослідження такої системи показані на рис. 10.

Як видно з рис. 10, швидкість регульованого насосу ω_1 поступово зменшується доки робоча точка нерегульованого насосу не опиниться справа від лінії максимального ККД. На 20-й секунді відбувається плавне зменшення гідравлічного опору мережі на 10 % від поточного значення, при цьому алгоритм не втрачає свої властивості. ККД некерованого насосу η_2 залишається коливатися в околі максимального значення, а ККД керованого η_1 збільшується. На 50-й секунді відбувається збільшення гідравлічного опору мережі на 20 %. Анало-

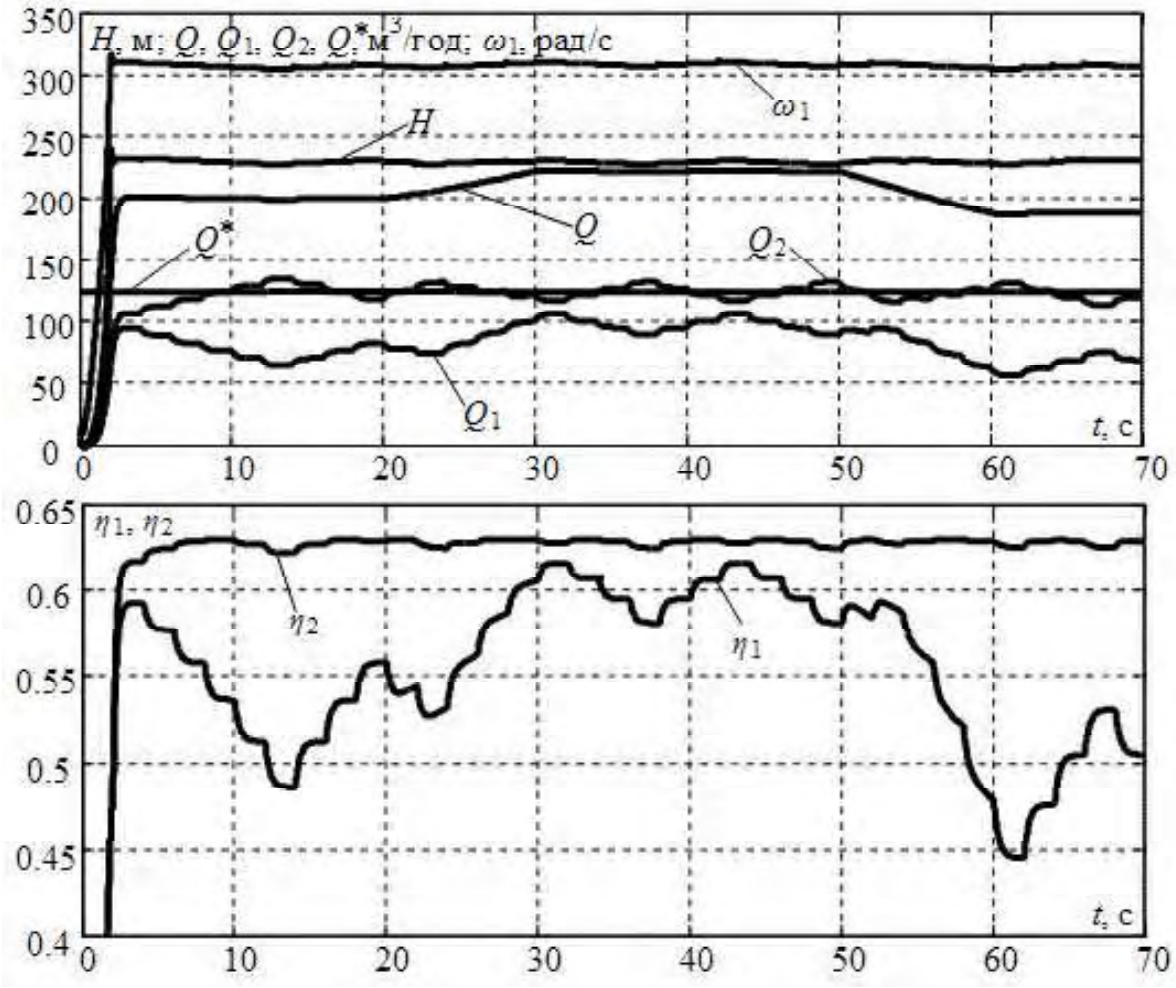


Рис. 10. Дослідження екстремальної системи керування ККД некерованого насосу за відсутності реакції споживачів

гічно попередньому випадку алгоритм залишається працездатним.

ВИСНОВКИ

При паралельному з'єднанні двох насосів, один з яких оснащений керованим асинхронним електроприводом, можливе застосування екстремального енергоефективного керування. При цьому можливий варіант керування ККД керованого насосу та ККД некерованого за рахунок зміни швидкості керованого насосу. Розташування вихідних робочих точок обмежується величиною жорсткості напірних характеристик насосів. Застосування другого варіанту керування може не передбачати реакцію споживачів, а сам алгоритм у цьому випадку нечутливий до зміни гідравлічного опору мережі під час роботи насосів. Діапазон допустимих продуктивностей установки залежить від геодезичної висоти підйому води та мінімально можливого гідравлічного опору мережі.

СПИСОК ЛІТЕРАТУРИ

1. *Kiselychnyk, O.* Overview of energy efficient control solutions for water supply systems / O. Kiselychnyk, M. Bodson, H. Werner // Transactions of Kremenchuk State Polytechnic Univ. Kremenchuk: KSPU. – N3/2009 (56), part 1. – 2009. – P. 40–45.
2. *Попович, М. Г.* Експериментальні дослідження роботи інтерактивного енергозберігаючого контролера на гібридній моделі насосної установки / М. Г. Попович, О. І. Кіселичник, С. О. Бур'ян, О. Ф. Соколовський // Вісник Кременчуцького державного політехнічного університету імені Михайла Остроградського. – Кременчук : КДПУ. – Випуск 3/2007 (44), Частина 1. – С. 72–75.
3. *Pechenik, M.* Experimental research of interactive energy saving controller of water supply pump based on flow rate measurement / M. Pechenik, O. Kiselychnyk, S. Buryan // Вісник Національного технічного університету «Харківський політехнічний інститут». Проблеми автоматизованого електроприводу. Теорія і практика. – Харків : НТУ «ХПІ». – 2010. – № 28. – С. 272–274.
4. *Popovich, M.* Extremal electromechanical control system of water supply pumps connected in series / Popovich M., Kiselychnyk O., Buryan S. // Transactions of Kremenchuk State Polytechnic Univ. Kremenchuk : KSPU. – N 3/2010 (62), part 2. – 2010. – P. 37–41.
5. *Бур'ян, С. О.* Двоканальна екстремальна електромеханічна система автоматичного керування насосною установкою / С. О. Бур'ян, Т. В. Грищук // Вісник Національного технічного університету «Харківський політехнічний інститут».

Проблеми автоматизованого електроприводу. Теорія і практика. – Х. : НТУ «ХПІ». – 2010. – № 28. – С. 176–179.

6. *Попович, М. Г.* Проблеми теорії автоматизації багатоагрегатних насосних установок на основі принципу пасивності / М. Г. Попович, О. І. Кіселичник // Технічна електродинаміка. Проблеми сучасної електротехніки. – Частина 5. – Київ. – 2006. – С. 54–59.
7. *ITT Industries*. Pump Selection Program / *ITT Industries* // Vogel Select CD. – Jan. – 2009.
8. *Marino, R.* Exponentially convergent rotor resistance estimation for induction motors / R. Marino, S. Peresada, P. Tomei // IEEE Trans.on Industrial Electronics. – 1995. – Vol. 42, No. 5. – P. 508–515.

Стаття надійшла до редакції 15.02.2011.

Бурьян С. А., Грищук Т. В.

ЭКСТРЕМАЛЬНАЯ ЭЛЕКТРОМЕХАНИЧЕСКАЯ СИСТЕМА УПРАВЛЕНИЯ ПАРАЛЛЕЛЬНО СОЕДИНЕННЫМИ НАСОСАМИ ВОДОСНАБЖЕНИЯ

В статье рассматривается применение экстремального энергоэффективного управления для двух параллельно соединенных насосов. Один из насосов комплектуется частотно-управляемым асинхронным электроприводом. Представлены описание алгоритма экстремального управления, его математическая модель и структурная схема системы.

Ключевые слова. Насос водоснабжения, электромеханическая система, экстремальное управление, энергоэффективность, регулируемый электропривод, параллельное соединение насосов.

Buryan S., Gryshuk T.

EXTREMAL ELECTROMECHANICAL CONTROL SYSTEM OF WATER SUPPLY PUMPS CONNECTED IN PARALLEL

The paper concerns the application of the extremal energy efficient control for two pumps connected in parallel. One of the pumps is with the controlled induction motor drive. The extremal control algorithm description, its mathematical model and the system block diagram are presented.

Key words: Water supply pump, electromechanical system, extremal control, efficiency, controlled drive, parallel connection of pumps.

ДО ВІДОМА АВТОРІВ

Журнал «Радіоелектроніка, інформатика, управління» (РІУ) призначений для публікації найбільш значимих наукових і практичних результатів досліджень учених вищих навчальних закладів і наукових організацій.

Журнал включений у перелік наукових видань України, у яких можуть публікуватися результати дисертаційних робіт на здобуття вчених ступенів доктора і кандидата технічних наук і фізико-математичних наук (радіофізика).

Статті, що опубліковано в журналі, реферуються в реферативних журналах і базах даних ВІНІТІ (Росія) і «Джерело» (Україна). Журнал міститься у міжнародній базі наукових видань Index Copernicus (http://journals.indexcopernicus.com/index.php). Інтернет-сторінка журналу:
http://journal.zntu.edu.ua/ric/index.php?page=index.

Журнал видається два рази в рік і розповсюджується за Каталогом періодичних видань України (передплатний індекс – 22914).

Для розгляду питання про публікацію статті в редакцію журналу необхідно вислати поштою або представити особисто наступне:

1) рукопис (роздруківку) статті, підписаний на останній сторінці всіма авторами, в двох екземплярах;

2) відомості про авторів;

3) оригінал експертного висновку про можливість відкритого опублікування статті;

4) супровідний лист-клопотання з організації, де була виконана робота (або лист автора);

5) рецензію від фахівця в даній області з вченим ступенем доктора наук. Підпис рецензента обов'язково мусить бути завірений.

6) диск з наступними файлами:

– електронна версія статті, повністю ідентична роздруківці;
– відомості про авторів;
– рисунки у графічному форматі .tif.

Файли з матеріалами статті можна вислати електронною поштою або передати особисто на оптичному диску або USB-накопичувачі.

Вимоги до оформлення статті. Приймаються статті, набрані в редакторі Microsoft Word.

Параметри сторінки:

– розмір паперу – А4 (210х297);
– орієнтація – книжкова;
– шрифт – Times New Roman, розмір – 12 pt;
– міжрядковий інтервал – полуторний;
– верхнє поле – 20 мм, нижнє – 20 мм, ліве – 25 мм, праве – 15 мм.

Сторінки рукопису повинні бути пронумеровані. Не допускаються розбіжності рукопису з електронною версією статті. Текст рукопису не повинен мати рукописних виправлень та позначок.

Послідовність розміщення матеріалу статті:

1) індекс УДК;

2) прізвища й ініціали авторів, назва статті, анотація й ключові слова українською мовою (для громадян України);

3) прізвища й ініціали авторів, назва статті, анотація й ключові слова російською мовою;

4) прізвища й ініціали авторів, назва статті, анотація й ключові слова англійською мовою;

5) текст статті;

6) список літератури.

Текст статті. Приймаються статті російською, українською та англійською мовами. Розмір статті до 0,5 авторського аркуша. У статті слід уникати зайвої деталізації, проміжних формул і висновків; не слід наводити відомі факти, повторювати зміст таблиць і ілюстрацій у тексті. Стаття не повинна мати граматичних або інших помилок, а також повинна відповідати тематиці журналу й вимогам ВАК щодо фахових видань. **Структура** тексту статті мусить містити такі необхідні елементи: постановка проблеми в загальному виді і її зв'язок з важливими науковими або практичними завданнями; аналіз останніх досліджень і публікацій, у яких розпочато розв'язання даної проблеми, і на які опирається автор; виділення нерозв'язаних раніше частин загальної проблеми, яким присвячується стаття; формулювання цілей статті (постановка завдання); виклад основного матеріалу дослідження з повним обґрунтуванням отриманих наукових результатів, висновки по даному дослідженню й перспективи подальших досліджень у даному напрямку. Матеріал публікації мусить бути розбитий на підрозділи не більше двох рівнів.

Рисунки розміщуються в тексті й додатково додаються в окремих файлах (формат .tif з роздільною здатністю 150–300 dpi, чорно-білі або у градаціях сірого). Розмір рисунків не повинен перевищувати ширини сторінки (17 см) або ширини колонки (8 см). Написи на рисунках бажано виконувати шрифтом Times New Roman, розмір 10. Рисунки нумерують і підписують унизу.

Формули виконуються за допомогою вбудованого в Word редактора Microsoft Equation. Формули нумерують у круглих дужках праворуч. Формули великого розміру записуються в кілька рядків.

Нумерація рисунків, формул і таблиць наскрізна одноріvнева.

Обсяг **анотації** не повинен перевищувати 40 слів.

Ключові слова наводяться в називному відмінку у кількості до десяти слів.

Список літератури наприкінці статті подається мовою оригіналу і складається в порядку згадування посилань у тексті й відповідно до діючого стандарту на бібліографічний опис. Посилання на літературу в тексті нумеруються послідовно й позначаються цифрою у квадратних дужках.

У **відомостях про авторів** необхідно навести:

1) прізвище, ім'я, по батькові (повністю);
2) учений ступінь;
3) посаду;
4) місце роботи;
5) електронну адресу;
6) робочий, домашній, мобільний телефони.

Статті, які не відповідають зазначеним вимогам, не приймаються до розгляду.

Всі статті проходять закрите рецензування і в разі потреби можуть бути повернуті автору на доробку. Редакція залишає за собою право на літературну редакцію тексту статті без повідомлення автору.

Рукописи й диски не вертаються, коректура та відбитки статей авторам не надсилаються.

Адреса редакції: 69063, м. Запоріжжя, вул. Жуковського, 64, ЗНТУ, редакція журнала «РІУ»
Тел.: (061) 7-698-2-96 – редакційно-видавничий відділ
(061) 7-644-6-62 – головний редактор
Факс: (061) 7-642-1-41
E-mail: rvv@zntu.edu.ua

Наукове видання

Радіоелектроніка,

інформатика,

управління

№ 2/2011

Науковий журнал

Головний редактор – д-р техн. наук Піза Д. М.

Заст. головного редактора – канд. техн. наук Дубровін В. І.

Комп'ютерний моделювання та верстання Зуб С. В.

Оригінал-макет підготовлено у редакційно-видавничому відділі ЗНТУ

Підписано до друку 06.12.2011. Формат 60×84/8.
Папір офс.Різогр. друк Ум. друк. арк. 19.
Тираж 300 прим. Зам. № 1789.

69063, м. Запоріжжя, ЗНТУ, друкарня, вул. Жуковського, 64