Кудряшова Е. С.
E. S. Kudryashova

ПРИМЕНЕНИЕ ВРЕМЕННЫХ СЕТЕЙ ПЕТРИ ДЛЯ РАЗРАБОТКИ СИСТЕМ РЕАЛЬНОГО ВРЕМЕНИ МОНИТОРИНГА СОСТОЯНИЯ ОБЪЕКТОВ

APPLICATION OF TIME PETRI NETS TO THE DEVELOPMENT OF REAL TIME SYSTEMS FOR OBJECTS' STATUS MONITORING

Кудряшова Екатерина Сергеевна – аспирант Комсомольского-на-Амуре государственного технического университета (Россия, Комсомольск-на-Амуре); 681013, г. Комсомольск-на-Амуре, пр. Ленина, д. 27; 89638209000. E-mail: ekatt@inbox.ru.
Ms. Ekaterina S. Kudryashova – PhD Candidate, Komsomolsk-on-Amur State Technical University (Russia, Komsomolsk-on-Amur); 681013, Komsomolsk-on-Amur, 27, Lenin Pr.; +7 963 8209000. E-mail: ekatt@inbox.ru

Аннотация. Работа посвящена применению временных сетей Петри для создания программного обеспечения мониторинга группы виртуальных машин с использованием QNX Neutrino RTOS. В работе рассматривается сущность временных сетей Петри, их использование для моделирования систем, успешность работы которых зависит от времени. В работе моделируется функционирование системы мониторинга в нотации временных сетей Петри, рассматривается алгоритм разработки программного обеспечения для отслеживания работы сети, с возможностью получения результатов в режиме реального времени.

Summary. This paper deals with the issue of using Time Petri Nets to develop software for monitoring virtual machines using QNX Neutrino RTOS. The paper considers the nature of time Petri nets and the issue o their application to modeling systems whose success is time-dependent. The operation of a monitoring system in the notation of Time Petri Nets is simulated, and the algorithm of developing specialized software to monitor the network with the opportunity of obtaining real time results is examined.

Ключевые слова: временные сети Петри, система мониторинга сети, моделирование, системы реального времени, временные дистрибутивные асинхронные автоматы.

Key words: Time Petri Nets, network monitoring system, simulation, real-time systems, distributed time asynchronous automata

УДК 681.3.06

Введение

Для описания и исследования поведения параллельных систем [7] в настоящее время широко используются различные математические структуры, такие как временные системы переходов [10], временные структуры событий [1], временные системы переходов с независимостью [4], временные сети Петри [2; 3; 6; 9].

На первый взгляд понятия времени и параллелизма имеют не много общего. Однако существует большое количество примеров из разных областей, иллюстрирующих эту связь. По этой причине было введено и изучено множество различных сетей Петри, зависящих от времени [12]. Для моделирования работы системы мониторинга в данной статье применяются временные сети Петри. Этот класс сетей является производным от классических сетей Петри. Преимущество моделирования в нотации временных сетей Петри заключается в наглядности разрабатываемой математической модели, простоте и однозначности интерпре-

тации. В отличие от других моделей, использующихся для моделирования систем реального времени, во временных сетях Петри моделируется не только множество различных состояний системы, но и переходы из одного состояния в другое. Также временные сети Петри позволяют однозначно определить порядок срабатывания переходов с помощью разметки.

Помимо математической модели в статье описывается программное обеспечение мониторинга группы виртуальных машин, разработанное на базе операционной системы реального времени. Известно, что в настоящее время на рынке существует множество систем мониторинга сети. Проблема заключается в том, что ни одна из них не работает в режиме реального времени, так необходимом для получения точных результатов и своевременного выявления неисправностей. В этой связи возникла потребность в написании системы мониторинга сети на базе операционной системы жесткого реального времени QNX Neutrino, сочетающей в себе все свойства, необходимые для качественного отслеживания работы сети, с возможностью получения результатов в режиме реального времени. Такой режим подразумевает отсутствие отказов системы, «зависаний», жесткий контроль сбоев в работе. Приложения, разработанные на базе операционных систем реального времени, отличаются очень высокими показателями надежности и непрерывности управления, времени реакции на аварийные ситуации. Разработанное программное обеспечение призвано снизить трудоемкость работ по мониторингу сети, сократить время выявления неисправностей, и, главное, увеличить отказоустойчивость системы мониторинга сети.

Временные сети Петри

Временная сеть Петри – это шестиэлементный кортеж

$$\mathcal{N} = (P, T, F, m^0, Eft, Lft),$$

где $P = \{p_1, \dots, p_n\}$ – конечное множество позиций; $T = \{t_1, \dots, t_n\}$ – конечное множество переходов; $F: (P \times T) \cup (T \times P) \to \mathbb{N}$ – отношение инцидентности; $m^0: P \to \mathbb{N}$ – начальная маркировка; $Eft: T \to \mathbb{N}, Lft: T \to \mathbb{N} \cup \{\infty\}$ – функции, описывающие соответственно раннее и позднее время доступности переходов, которые удовлетворяют ограничению $Eft(t) \le Lft(t)$ для каждого $t \in T$ [11].

Функция F определяет структуру ориентированного графа, множеством вершин которого является $P \cup T$, а стрелками – пары $(a, b) \in (P \times T) \cup (T \times P)$. Состояния сети Петри задаются разметками, которые получаются из начальной разметки с помощью срабатывания конечных последовательностей переходов. Переход t является доступным, если $\forall (p \in P)\ m^0(p) \ge F(p, t)$ и время, прошедшее с момента доступности перехода t, достигло значения Eft.

Раннее и позднее время доступности перехода t соответствует временному интервалу, в который переход t может сработать. Если же время, прошедшее с того момента, когда переход t стал доступен, достигло значения $Lft(t)$, то переход может сработать до тех пор, пока не будет блокирован срабатыванием другого перехода.

Математическая модель

На рис. 1 изображена временная сеть Петри, моделирующая работу системы мониторинга сети.

Представленная математическая модель сочетает в себе клиентскую и серверную части. Каждому переходу в сети назначен определенный временной интервал. При этом используются буквы латинского алфавита как обозначение определенного известного значения времени, знак бесконечности и дельта – для обозначения небольшого значения времени. Интервалы, ограниченные справа бесконечностью, означают, что данное действие не является критическим по времени, оно может происходить долго, и оно обязательно произойдет.

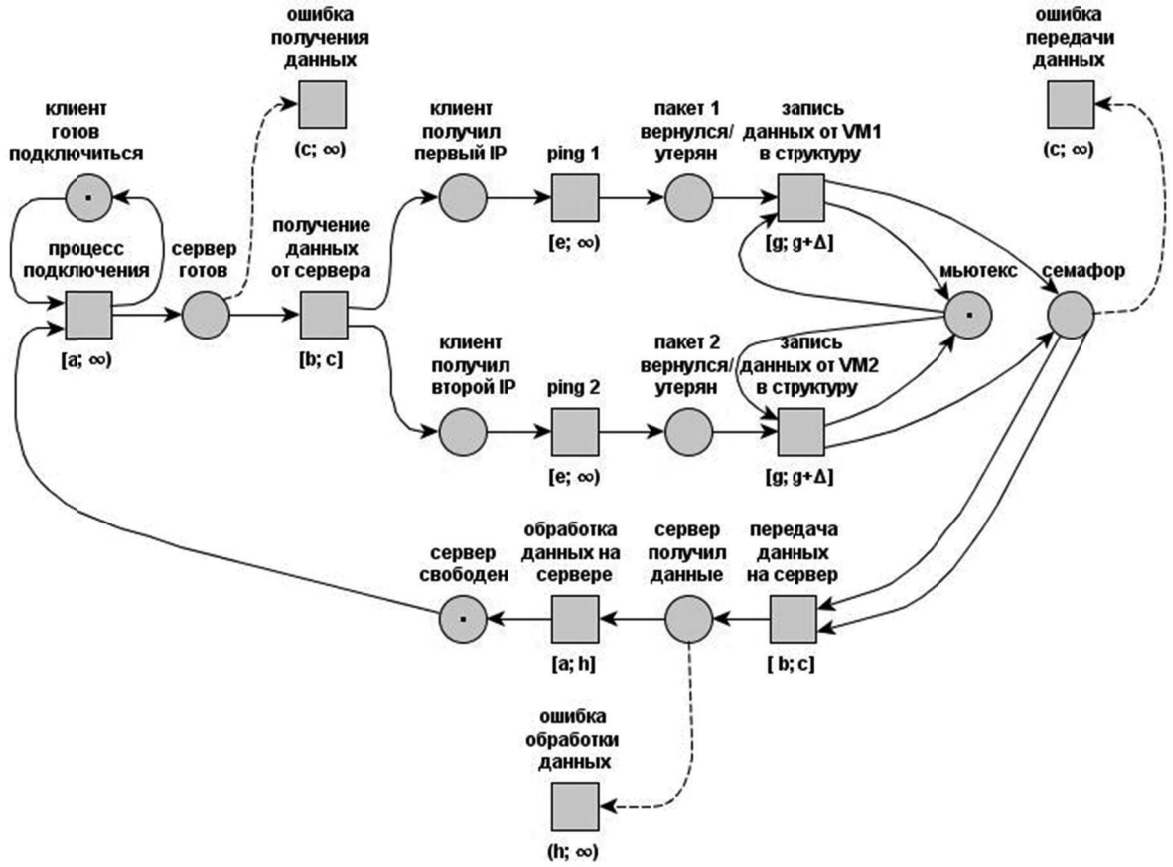

Рис. 1. Временная сеть Петри

Работа сети начинается с перехода *процесс подключения*. Он может сработать в интервале [a; ∞), при условии, что *клиент готов* и *сервер свободен*. Под этим подразумевается, что для подключения необходимо, чтобы клиент был запущен, а все операции по обработке данных на сервере к началу нового цикла мониторинга должны быть завершены.

Когда процесс подключения завершен, то есть одноименный переход сработал, сервер готов передать клиенту данные. Переход *получение данных от сервера* может сработать в интервале [b; c]. Если данные по какой-то причине не были получены, программа завершается с ошибкой (*ошибка получения данных*). Ошибки, связанные с невозможностью уложиться в заданное время, могут возникать только на стороне сервера. На стороне клиента это невозможно, так как QNX является надежной отказоустойчивой системой. После того, как клиент получил IP-адреса, он приступает к мониторингу. На рис. 1 изображена сеть Петри для мониторинга двух виртуальных машин. Мы видим два независимых потока, в каждом из которых осуществляется *ping*. Следует указать, что сеть для мониторинга большего числа виртуальных машин строится аналогично путем добавления неограниченного числа новых блоков от получения IP до записи данных в структуру. *Ping* может закончиться возвратом или потерей пакета (*пакет вернулся/утерян*). Считается, что возврат пакета и потеря пакета – это одно и то же событие, так как оба эти результата одинаково важны для дальнейшего функционирования системы. В этом случае, если пакет вернулся, то мы получили необходимую информацию о виртуальной машине, находящейся в сети, а если пакет потерян, мы можем сделать вывод, что виртуальная машина отказала. При любом исходе данной операции цель мониторинга выполняется.

После ping происходит запись данных в структуру за небольшой интервал времени (*запись данных от VM в структуру*). Переходы записи данных в структуру срабатывают по очереди, так как каждый из них может сработать, только если *мьютекс* свободен. Метка из

мьютекса попадает в переход, он срабатывает, затем метка снова возвращается в *мьютекс*. В следующий момент времени может сработать параллельно работающий поток. После срабатывания перехода *запись данных от VM в структуру* метка попадает в *семафор*. Количество меток в семафоре должно быть равно количеству виртуальных машин в сети. Количество дуг, выходящих из семафора в переход *передача данных на сервер*, также должно быть равно количеству виртуальных машин. Переход *передача данных на сервер* срабатывает только при условии, что все потоки записали свои данные в структуру, то есть если все метки попали в *семафор*. *Передача данных на сервер* также может завершиться неудачей (*ошибка передачи данных*). Если данные передались на сервер, то происходит их обработка и сервер снова свободен. Цикл повторяется.

Рассмотрим некоторые переходы более подробно. На рис. 2 изображена временная сеть Петри, детализирующая переход *процесс подключения*.

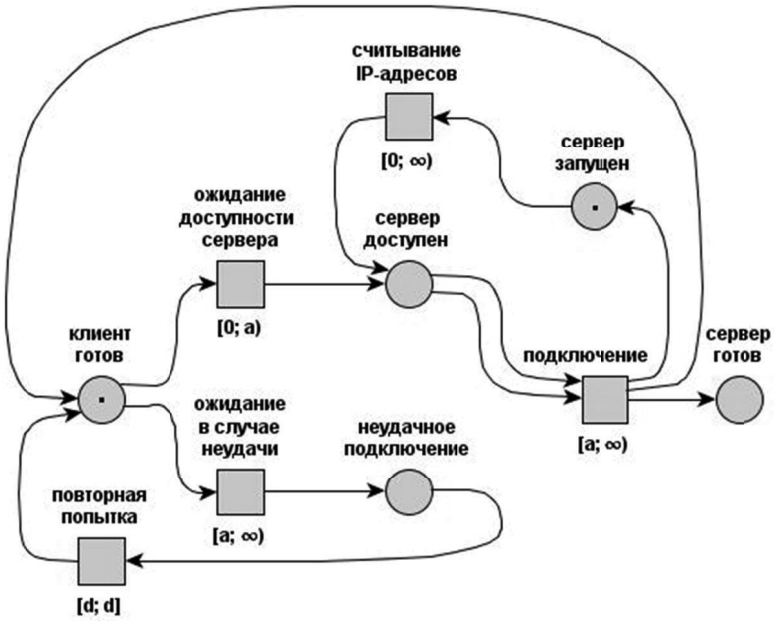

Рис. 2. Процесс подключения

Процесс подключения начинается с попытки подключения. Когда клиент готов, он пытается подключиться к серверу, проверяя тем самым доступность сервера. Некоторый интервал времени клиент ждет доступности сервера (*ожидание доступности сервера*). Если переход не успевает сработать за заданное время, это означает, что сервер не доступен в данный момент. Тогда срабатывает переход *ожидание в случае неудачи.* Затем клиент повторяет попытку подключения (*повторная попытка*). Если попытка подключения завершилась успехом и *сервер доступен*, происходит собственно *подключение*. После этого сервер готов передать данные клиенту. Доступность сервера означает то, что пользователь ввел список IP-адресов виртуальных машин (*считывание IP-адресов*). Интервал [d; d] означает, что повторная попытка происходит мгновенно.

На рис. 3 изображена временная сеть Петри, детализирующая переход *ping*.

В программе будет столько переходов *ping*, сколько виртуальных машин в сети. Каждый переход выполняется в отдельном потоке. Время работы перехода определяется интервалом [e; ∞). Это означает, что переход может срабатывать долгое время, и он обязательно сработает. Но внутри *ping* имеются жесткие временные ограничения. Именно поэтому необходимо рассмотреть его отдельно от общей сети. *Ping* может сработать после того, как клиент получил адреса виртуальных машин от сервера. Сначала происходит посылка пакета до виртуальной машины за короткий промежуток времени (*посылка пакета*). Когда пакет послан, мы ждем его возвращения. Время ожидания задано в программе. Оно является крити-

ческим. Если пакет не успел вернуться за определенное время, он считается потерянным. Возврат пакета должен произойти за время, заданное интервалом [e; f] (*возврат пакета*). Если этот переход не сработал, срабатывает переход *ожидание возврата пакета*. Независимо от того, какой из переходов сработал, мы получаем результат (*пакет вернулся/утерян*).

Временные сети Петри являются очень удобным математическим аппаратом для моделирования подобных систем. С помощью представленной модели легко проследить работу системы мониторинга в любой момент времени, детально рассмотреть происходящие процессы. Но существуют задачи, для решения которых нужны более общие временные модели [8]. Именно поэтому в работе [5] было введено обобщение временных сетей Петри, названное временным дистрибутивным асинхронным автоматом.

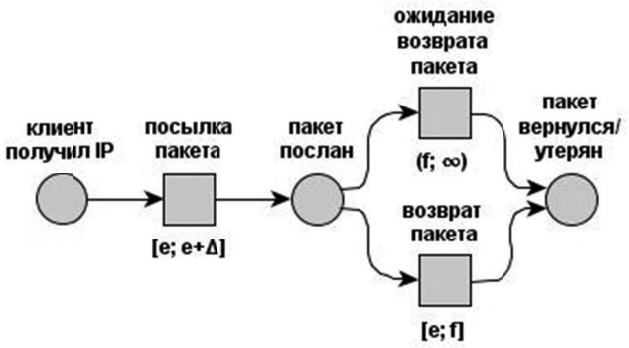

Рис. 3. Пинг

Временной дистрибутивный асинхронный автомат – это семиэлементный кортеж

$$\mathcal{A} = (S, s_0, E, I, Tran, Eft, Lft),$$

состоящий из множеств S и E, элемента $s_0 \in S$, отношения $Tran \subseteq S \times E \times S$, семейства антирефлексивных симметричных отношений $I = (I_s)_{s \in S}$, $I_s \subseteq E \times E$, пары функций $Eft: E \to \mathbb{R}_{\geq 0}$, $Lft: E \to \mathbb{R}_{\geq 0} \cup \{\infty\}$, удовлетворяющих для всех $a \in E$ неравенству $Eft(a) \leq Lft(a)$.

На рис. 4 показан временной дистрибутивный асинхронный автомат, который определяет сеть Петри «Процесс подключения», представленную на рис. 2.

Маркировкам сети Петри m_i соответствуют состояния автомата $S_i, i \in \{0,5\}$. Перечислим эти состояния: S_0 – клиент готов/ сервер запущен (начальная маркировка сети); S_1 – сервер доступен/сервер запущен; S_2 – сервер доступен (IP – адреса считаны); S_3 – клиент готов/сервер доступен; S_4 – сервер доступен/неудачное подключение; S_5 – сервер запущен/неудачное подключение. Переходам сети t_i соответствуют действия $a_i, i \in \{1,5\}$ (на рисунке изображены стрелками с временными интервалами). Заметим, что при построении временного дистрибутивного асинхронного автомата не учитывалось место *сервер готов*, так как оно играет роль исключительно для дальнейшего моделирования и в данном контексте его можно опустить.

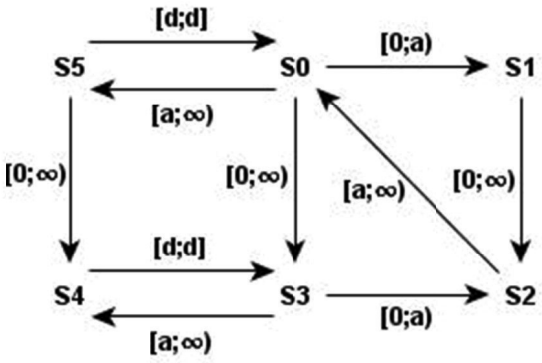

Рис. 4. Временной дистрибутивный асинхронный автомат

Построенная математическая модель позволяет легко рассчитывать время, необходимое для перехода из одного состояния в другое. Для примера рассчитаем минимальное время выполнения операций, ведущих из состояния S_1 в S_0. Минимальное время выполнения операций рассчитывается по формуле $\max(Eft(a_2), Eft(a_3))$, где $a_2 = [0; \infty), a_3 = [a; \infty)$. Соответственно минимальное время будет равно a.

Программное обеспечение

Система мониторинга состояния объектов представляет собой клиент-серверное приложение с веб-интерфейсом. Назначение системы мониторинга – следить за корректным функционированием виртуальных машин в пределах одной сети. Серверная часть разрабо-

тана на базе операционной системы семейства Windows. Она представляет собой программный модуль, взаимодействующий с базой данных, а также клиентом, разработанным на базе операционной системы QNX. Клиент производит собственно мониторинг группы виртуальных машин и работает в режиме реального времени, то есть в таком режиме работы автоматизированной системы обработки информации и управления, при котором учитываются ограничения на временные характеристики функционирования.

Рассмотрим подробнее модули, входящие в состав программного обеспечения.

Клиентское приложение написано в операционной системе QNX Neutrino 6.2 на языке C. Назначение клиентского приложения – получать список IP-адресов от сервера, осуществлять непосредственно контроль сетевой активности группы виртуальных машин (производить *ping*) и возвращать результаты на сервер. Данный модуль представляет собой консольное приложение, состоящее из нескольких процедур. Перечислим главные из них: процедура установки соединения с сервером, процедура конвертации IP-адреса из байтового представления в строку, процедура запуска ping, процедура прерывания ping через определенное время, процедура конвертации данных ICMP пакета в строку, процедура анализа информации и вывода на печать, процедура вычисления времени работы ping и другие. Серверное приложение разработано в операционной системе Windows XP на языке C# с использованием средств Visual Studio 2008. Назначение серверного приложения – получать IP-адреса из базы данных, отправлять их клиенту, получать от клиента результаты, обрабатывать их и записывать в базу данных. При первом запуске системы серверное приложение также используется для ввода IP-адресов и запуска мониторинга. В серверном приложении имеется интерфейс взаимодействия с пользователем, поэтому оно может частично заменить веб-интерфейс в случае невозможности использования последнего. К основным процедурам серверного приложения можно отнести процедуры работы с базой данных (выборка и запись данных), обработчики пунктов меню (задание IP-адресов, вызов справки), процедуры инициализации форм и диалоговых окон и другие. Веб-интерфейс написан в операционной системе Windows XP на языке PHP, JavaScript и HTML с использованием Wampserver 2.1. Благодаря существованию веб-интерфейса, программное обеспечение имеет ряд дополнительных возможностей, а именно: авторизация пользователя, добавление нового пользователя, ввод и удаление IP-адресов, отображение узлов сети, построение диаграмм и сбор статистики, создание отчетов. Все результаты работы системы мониторинга сохраняются в базе данных под управлением MySQL. В базе данных хранится информация об имеющихся в сети узлах, их последний статус (неизвестно, доступен, недоступен), время ping для каждого узла в определенную сессию, список пользователей с их логином и паролем. Также в базе сохраняется дата и время каждой сессии для вывода статистики за тот период, который задаст пользователь. С помощью SQL-запросов из базы данных всегда можно получить необходимые отчеты по работе системы, а также отображать на экране состояние виртуальных машин в текущий момент времени. В процессе функционирования программного обеспечения данные, введенные пользователем, а также полученные при мониторинге, передаются от одной подсистемы к другой, обеспечивая тем самым надежную работу всего программного обеспечения в целом.

Опишем принцип работы системы мониторинга сети. Работа системы начинается с запуска серверного приложения. После запуска пользователю необходимо ввести список IP-адресов, мониторинг которых будет осуществляться. Допускается ввод как одиночных адресов, так и диапазонов. После ввода IP-адресов на экране появляется карта сети. По ней легко проследить, какие виртуальные машины в данный момент активны, на какой машине произошел сбой. Далее к серверу на Windows подключается клиент на QNX. Клиент получает от сервера список IP-адресов, анализирует их и приступает к мониторингу. Мониторинг осуществляется путем посылки ICMP-пакета к каждой виртуальной машине в сети. Данные ICMP-пакета от каждой виртуальной машины сохраняются в специальной структуре. Чтобы избежать подмены одних данных другими, на этапе записи в структуру используется мьютекс. Посылка пакета к каждой виртуальной машине происходит в отдельном потоке, и

Материал поступил 05.10.2012

для записи данных от конкретной машины в структуру потоку необходимо захватить мьютекс. Таким образом, в определенный момент времени может происходить запись только одного пакета данных. Чтобы отследить, были ли получены данные от всех машин в сети, в программе используется семафор. Счетчик семафора равен количеству машин в сети. В момент записи своих данных в структуру, каждый поток захватывает семафор. Таким образом, когда семафор будет полностью захвачен, мы получим данные от каждой виртуальной машины. Когда мониторинг завершен, происходит отправка данных структуры на сервер. На сервере данные анализируются, записываются в базу данных. При необходимости получения отчета соответствующие данные будут извлечены из базы. На экране отображается текущее состояние каждой виртуальной машины. Работа системы повторяется.

Заключение

Таким образом, в данной работе моделируется функционирование системы мониторинга сети в нотации временных сетей Петри, а также приводится алгоритм работы программного обеспечения.

Работа выполнена в рамках программы стратегического развития государственных образовательных учреждений высшего профессионального образования, грант № 2011-ПР-54.

ЛИТЕРАТУРА

1. Вирбицкайте, И. Б. Семантические области временных структур событий / И. Б. Вирбицкайте, Р. С. Дубцов // Программирование. – 2008. – № 3. – С. 3-20.
2. Вирбицкайте, И. Б. Использование техники частичных порядков для верификации временных сетей Петри / И. Б. Вирбицкайте, Е. А. Покозий // Программирование. – 1999. – № 1. – С. 28-41.
3. Вирбицкайте, И. Б. Метод параметрической верификации поведения временных сетей Петри / И. Б. Вирбицкайте, Е. А. Покозий // Программирование. – 1999. – № 4. – С. 16-29.
4. Дубцов, Р. С. Теоретико-категорные исследования временных систем переходов с независимостью / Р. С. Дубцов // IX Всероссийская конференция молодых ученых по математическому моделированию и информационным технологиям. – Кемерово, 2008. – 24 с.
http://www.ict.nsc.ru/ws/YM2008/14295/dubtsov.pdf
5. Кудряшова, Е. С. Обобщенные асинхронные системы / Е. С. Кудряшова, А. А. Хусаинов // Моделирование и анализ информационных систем. – 2012. – Т. 19. – № 4. – С. 78-86.
6. Покозий, Е. А. Метод верификации свойств параллелизма временных сетей Петри / Е. А. Покозий // Препринт, Институт Систем Информатики СО РАН. – Новосибирск. – 1999. – № 61. – 28 с.
7. Трещев, И. А. Методы построения систем автоматизированного распараллеливания приложений для архитектур с симметрично адресуемой памятью / И. А. Трещев // Ученые записки Комсомольского-на-Амуре государственного технического университета. Науки о природе и технике. – 2011. – № I-1(5). – С. 29-32.
8. Хусаинов, А. А. Математическая модель задачи о читателях и писателях / А. А. Хусаинов // Информационные технологии и высокопроизводительные вычисления: материалы Международной науч.-практ. конф. – Хабаровск, 2011. – С. 327-332.
9. Gonzalez del Foyo. Using Time Petri Nets for modeling and verification of timed constrained workflow systems / Pedro M. Gonzalez del Foyo, J. R. Silva // ABCM Symposium Series in Mechatronics. – vol. 3. – 2008. – P. 471-478.
10. Henzinger, T. A. Timed transition systems / T. A. Henzinger, Z. Manna, A. Pnueli. In G. Goos, J. Hartmanis, editor // Real-Time: Theory in Practice, Lecture Notes in Computer Science 600. – Springer-Verlag. – 1991. – P. 226-251.
11. Penczek, W. Advances in Verification of Time Petri Nets and Timed Automata / W. Penczek, A. Półrola // Poland: Springer. – vol. 20. – 2006. – P. 7-12.
12. Wegener. Petri Nets with Time Windows: a comparison to Classical Petri Nets / Jan-Thierry Wegener, Louchka Popova-Zeugmann // Fundamenta Informaticae. – 2009. – № 93. – P. 337-352.