УДК 621.3

Е.Г.Игнатенко, В.И.Бессараб, В.В.Турупалов

АЛГОРИТМ АДАПТИВНОЙ БАЛАНСИРОВКИ НАГРУЗКИ В КЛАСТЕРНЫХ СИСТЕМАХ

Вопросам управления нагрузкой в кластреных системах уделяется большое внимание в научной литературе, что подчеркивает важность и актуальность решаемой задачи. Теоретические исследования и разработка фундаментальных основ распределения нагрузки, создание математического аппарата, моделей и методов управления для распределения нагрузки в кластерных web-серверах рассматривались в работах ученых R. Mukherjee [1], G. Banga, V. Cardellini [2], E. Casalicchio [3], Xiao Qin, Hong Jiang, Yifeng Zhu, David R., Зар Ней Линг. Вопросами, связанными только с балансировкой нагрузки занимались E. Casalicchio, H.K. Lee, M. Andreolini; оптимизацией производительности – T. Schroeder [4], T. Vercauteren, X. Wang; перегрузками – A.Kamra, V. Misra [5], Черкасова Л. [6]; диспетчеризацией в географическом масштабе - V. Cardellini [7], P.Yu, Y.S. Hong [8].

Известные алгоритмы поиска решений для распределения запросов в большинстве используют приближенные или эвристический алгоритмы. Эти алгоритмы не учитывают множества параметров, таких как производительность, объем свободной оперативной памяти, скорость и стоимость вычисления, которые необходимо анализировать при реализации балансировки нагрузки (БН).

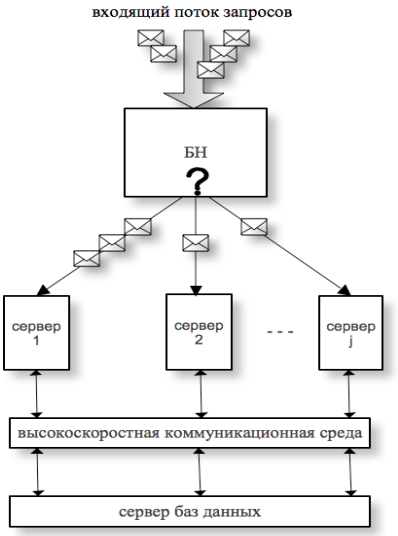

Рисунок 1 – Архитектура кластерного web сервера

Целью данной статьи является разаработка адаптивного алгоритма БН, применение которого позволит увеличить пропускную способность web кластера, уменьшить время обработки запроса и снизить затраты на вычисления.

Механизм БН играет важную роль в обеспечении пропорциональной загрузки ресурсов кластерного web сервера. В общем виде архитектуру кластерного web сервера можно представить в виде (рис.1).

Эффективность системы БН зависит от алгоритма распределения запросов. Анализ литературы [1-9,11-13] показывает, что существующие алгоритмы балансировки нагрузки можно разделить на 2 основных класса: контенто-зависимые (7-ого уровня модели OSI) и контенто-независимые (4-ого уровня модели OSI) алгоритмы. В свою очередь эти классы делятся на подклассы в зависимости от учета динамики системы.

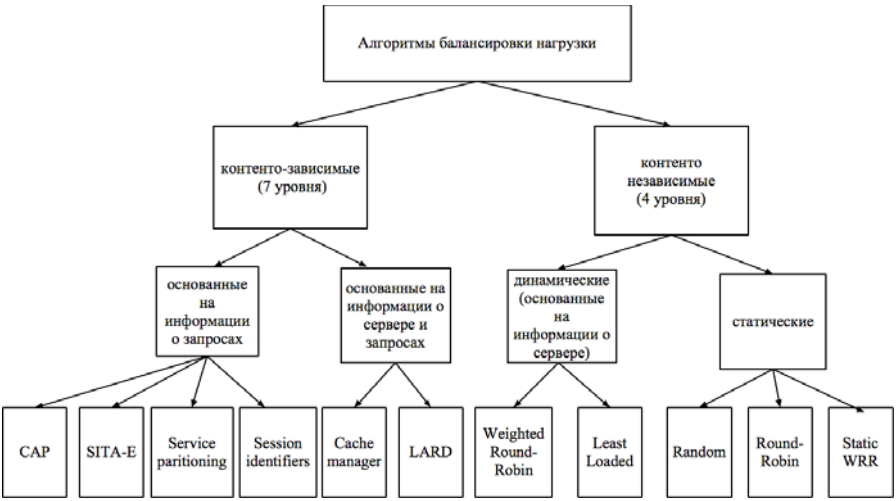

Рисунок 2 – Классификация существующих алгоритмов БН в web кластерах

С использованием разработанного авторами генератора самоподобного трафика [10], выполнен анализ известных алгоритмов балансировки, с целью выделения их достоинств и недостатков. В качестве исследуемых алгоритмов взяты четыре алгоритма каждого класса:

- RR (round robin)- статический алгоритм 4-ого уровня модели OSI;
- WRR (weighted round robin) – динамический алгоритм с обратной связью 4-ого уровня модели OSI;
- CAP (client aware policy) – контенто-зависимый алгоритм 7-го уровня модели OSI;
- LARD (locality aware policy) – контенто-зависимый алгоритм 7-го уровня модели OSI, учитывающий загрузку серверов.

Рассмотрим принципы работы каждого алгоритма:

- RR (round robin) – простейший статический алгоритм, в котором запросы поочередно направляются на сервера кластера. Порядок распределения входящих запросов с использованием алгоритма RR приведен на рисунке 3:

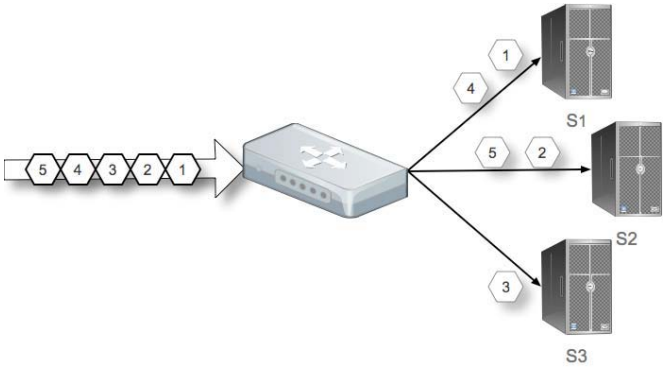

Рисунок 3 – Распределение входящих запросов с использованием алгоритма RR

- WRR (weighted round robin) – динамический алгоритм с обратной связью. Балансировщик каждый дискретный интервал контроля опрашивает сервера об их загрузке, и на основании этой информации пропорционально распределяет запросы. Порядок распределения входящих запросов с использованием алгоритма WRR приведен на рисунке 4:

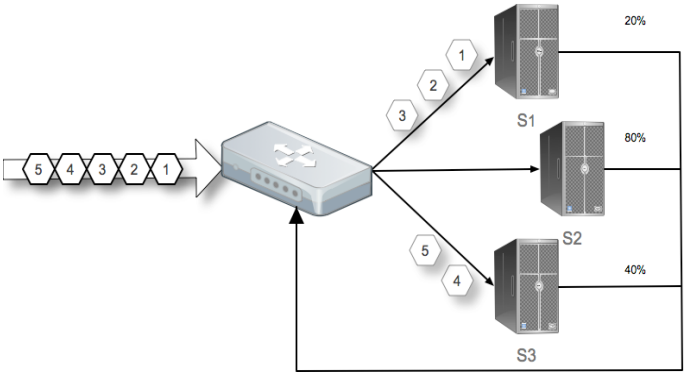

Рисунок 4 – Распределение входящих запросов с использованием алгоритма WRR

- CAP (client aware policy) – алгоритм 7-ого уровня модели OSI. Различает содержимое запроса, на основании которого и происходит балансировка [11].

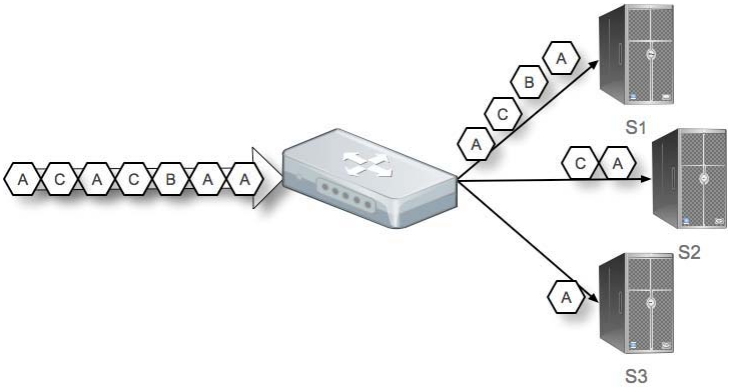

Рисунок 5 – Распределение входящих запросов с использованием алгоритма CAP

- LARD (locality aware request distribution) – алгоритм 7-ого уровня модели OSI, при балансировке учитывается тип запроса и состояние сервера. При поступлении запроса на БН определяется его тип и направляется на сервер, который обрабатывает запросы данного типа [12]. При перегрузке одного из серверов, запрос направляется на низкозагруженный сервер, если такой существует, или на наименее загруженный. Для этого опрделяется два параметра T_{low} - обозначает верхнюю границу низкой загрузки, T_{high} - нижнюю граница перегруженного состояния:

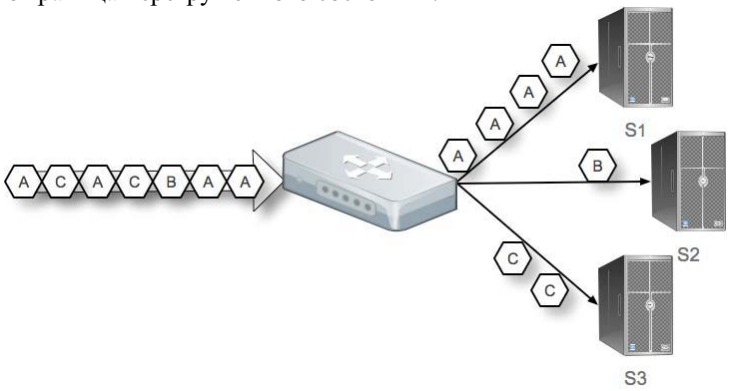

Рисунок 6 – Распределение входящих запросов с использованием алгоритма LARD

Проведем анализ работы вышеприведенных алгоритмов. Пусть кластер серверов состоит из 3 серверов различной производительности. Производительность сервера определяется нагрузкой на процессор, диск и средним временем обработки запроса. В качестве входящего потока рассмотрим поток http запросов, обладающий свойством самоподобия с показателем Херста=0,87. Поток содержит 4 типа запросов, которые на

основании рекомендаций, представленных в [12,13] и собственных исследований разделены по ожидаемому влиянию на сервер. В таблице 1 приведена классификация http запросов. Количество последовательных запросов, которые пользователь посылает на web сайт, описывается обратным распределением Гаусса, размер запрашиваемых файлов – логнормальным распределением, время обслуживания запроса на сервере - распределением Вейбула и зависит от класса запроса.

Таблица 1

Классификация http запросов по ожидаемому влиянию на сервер

Класс запроса	Пример файла	CPU требования (%)
1	Статическая информация: html	0,009-0,005
2	Динамические html страницы: php, jsp и asp	0,2-0,28
3	Информация безопасности (операция шифрования), операции поиска (требуют большое количество CPU ресурсов)	0,43-0,49
4	Мультимедиа (передач аудио и видео в реальном времени)	0,75-1,5

В качестве оценок эффективности работы алгоритмов испоьзуются:

- равномерность загруженности серверов;
- количество потерянных запросов;
- производительность серверов (пропускная способность).

Для оценки эффективности работы алгоритмов предложен интегральный критерий оптимизации системы балансировки нагрузки, основаный на загрузке ресурсов серверов обработки запросов:

$$s = \sqrt{\frac{\sum_{j=1}^{N} (\overline{U}(k) - U_j(k))^2}{j}}$$

где $U_j(k)$ - интегральный показатель загруженности *j*-ого сервера на *k*-м шаге;

$\overline{U}(k)$ - средняя загрузка серверов на *k*-м шаге;

N – количество серверов в кластере.

Результаты моделирования приведены на рисунках 7-10:

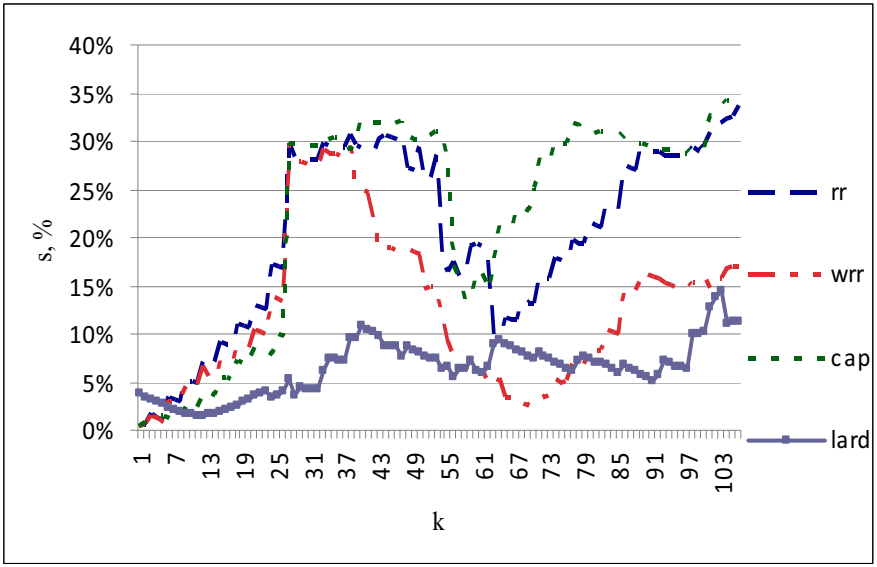

Рисунок 7 – Интегральный критерий оптимизации системы БН

Количество потерянных запросов на каждом из 3-х серверов приведено на рисунке 8:

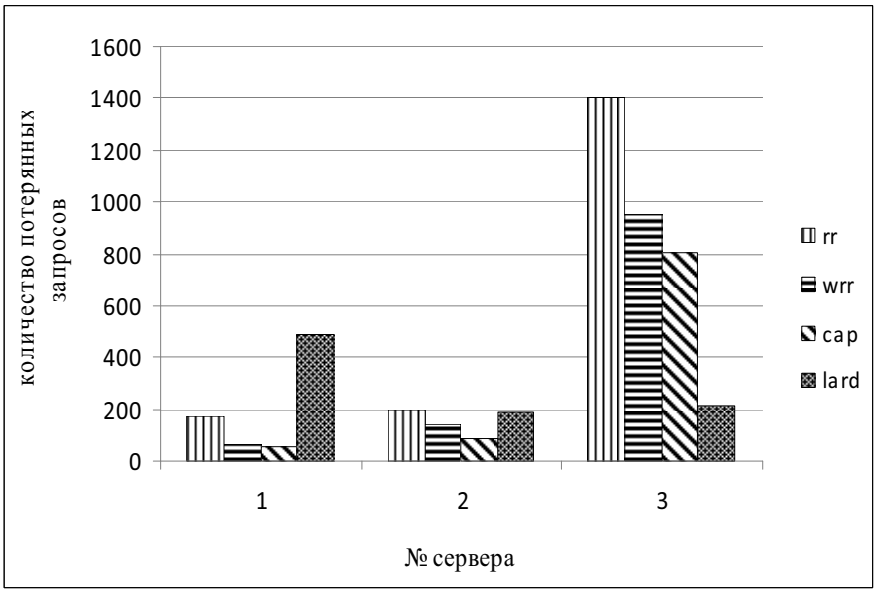

Рисунок 8 – Количество потерянных запросов

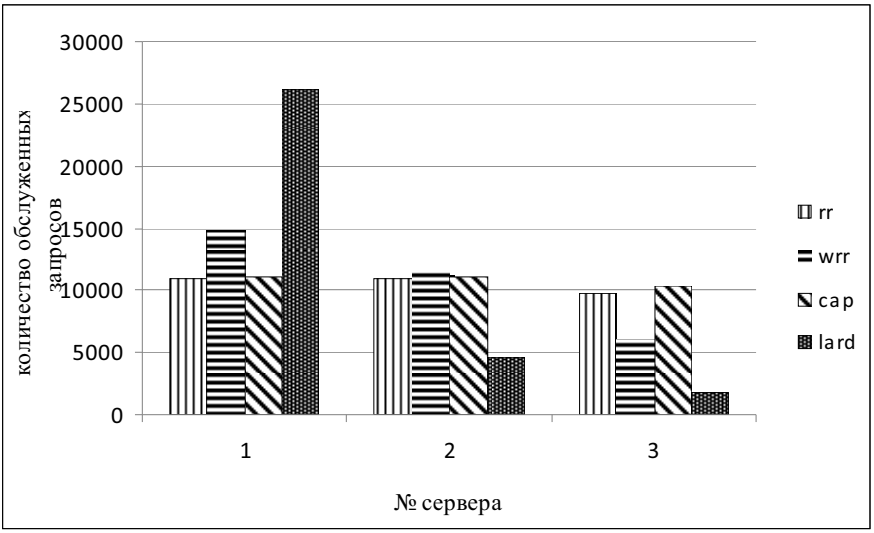

Рисунок 9 – Количество обслуженных запросов серверами

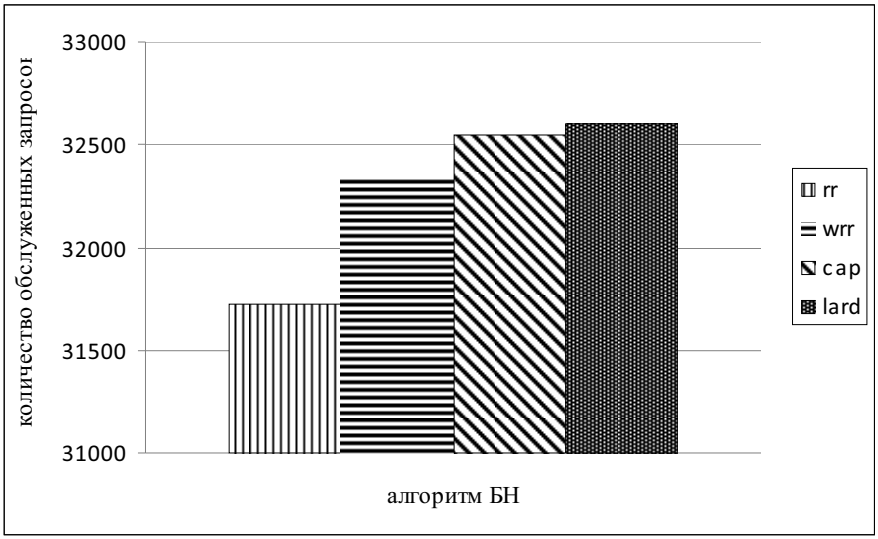

Рисунок 10 – Количество обслуженных запросов кластером

На основании проведенного моделирования можно сформулировать следующие недостатки:

1.Низкая производительность, связанная с потерями запросов.

2.Неравномерное распределение нагрузки между серверами.

3.Недостаточный учет динамики системы. Изменение параметров

входящего потока не связано с обменом информацией между БН и серверами;

4.Большие накладные расходы ресурсов, обусловленные постоянным обменом информацией (для WRR и LARD).

В связи с этим актуальной является разработка методов и алгоритмов для распределения нагрузки кластерного web-сервера. Разрабатываемый алгоритм должен быть оптимизирован для обеспечения статистически равномерного распределения нагрузки на серверах, выполняющих постоянные, относительно небольшие запросы (такой тип нагрузки типичен для веб-серверов). Алгоритм должен обеспечивать высокие показатели производительности, пропускной способности, отказоустойчивости (автоматически обнаруживая сбои узлов и перераспределяя поток данных среди оставшихся) и низкое время отклика.

Анализируя полученные результаты, можно сделать вывод о целесообразности разработки динамического контентнозависимого алгоритма с модифицированной обратной связью. Схематически концепцию предлагаемого алгоритма можно представить в виде следующей схемы:

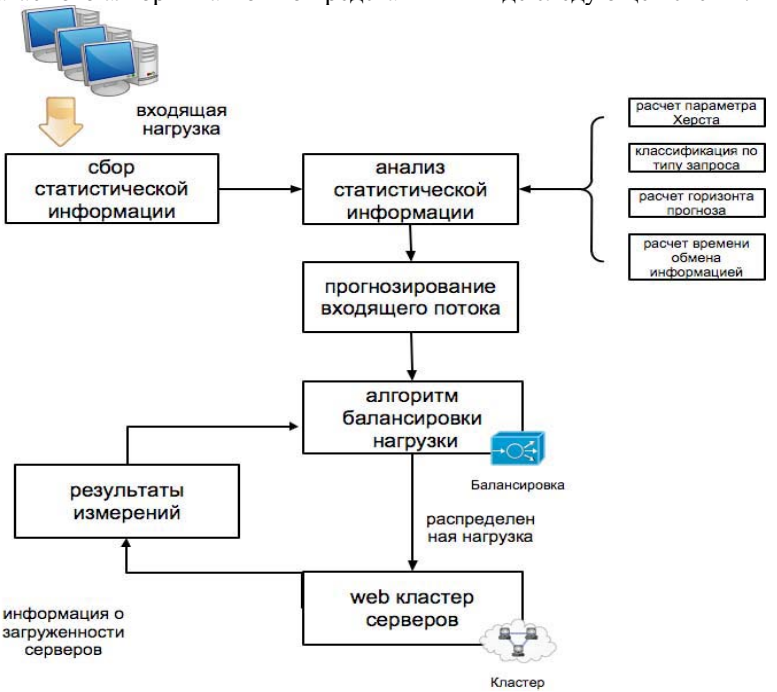

Рисунок 11 – Разрабатываемый алгоритм

Основной идеей алгоритма является распределение пользовательских запросов на основании прогноза входящего потока. При этом горизонт

прогноза на каждом шаге должен меняться в зависимости от изменений входящего потока. Выводы. В статье приведена детальная классификация существующих алгоритмов распределения нагрузки. Проанализирована их работа и оценены основные показатели работы. Исходя из анализа достоинств и недостатков существующих решений, предложен алгоритм адаптивной балансировки нагрузки. В качестве оценки эффективности работы предложен интегральный критерий оптимизации системы балансировки нагрузки, основаный на загрузке ресурсов серверов обработки запросов.

1. *R. Mukherjee,* A Scalable and Highly Available Clustered Web Server in High Performance Cluster Computing: Architectures and Systems, vol. 1, Rajkumar Buyya, Prentice Hall, 1999.
2. *V. Cardellini, E. Casalicchio, M. Colajanni*, A performance study of distributed architectures for the quality of web services, in: Proc. of the 34th Conference on System Sciences, Vol. 10, 2001.
3. *E. Casalicchio, M.Colajanni*, A client aware dispatching algorithm for web clusters providing multiple services, in: Proc. of the 10th International Conf. on WWW, 2001, pp. 535–544.
4. *T. Schroeder, S. Goddard, B. Ramamurthy*, Scalable web server clustering technologies, IEEE Network (May–June) (2000) 38–45.
5. *A. Kamra, V. Misra, E.M. Nahum*, Yaksha: a self-tuning controller for managing the performance of 3-tiered web sites, in: 12th IEEE International Workshop on Quality of Service, IWQOS 2004, 2004, pp. 47–56.
6. *L. Cherkasova, P. Phaal*, Session-based admission control: a mechanism for peak load management of commercial web sites, IEEE Transactions on Computers 51 (June (6)) (2002).
7. *V. Cardellini, E. Casalicchio, M. Colajanni, Ph.S. Yu*, The state of the art in locally distributed web-server systems, ACM Computing Surveys (CSUR) 34 (June (2)) (2002) 263–311.
8. *Y.S. Hong, J.H. No, S.Y. Kim,* DNS-based load-balancing in distributed web- server systems, in: Fourth IEEE Workshop on Software Technologies for Future Embedded and Ubiquitous Systems (WCCIA 2006), 2006, p. 4.
9. *S. Sharifian, et al.,* A predictive and probabilistic load-balancing algorithm for cluster-based web servers, Appl. Soft Comput. J. (2010), doi:10.1016/j.asoc.2010.01.017
10. *Бессараб В.И., Игнатенко Е.Г., Червинский В.В.* Генератор самоподобного трафика для моделей информационных сетей. Вісник Східноукраїнського Національного Университету ім. Володіміра Даля № 2 (144), 2010.
11. *E. Casalicchio, V. Cardinellini*. Content aware dispatching algorithms for cluster-based Web servers. Kluwer Academic Publishers. Cluster Computing 5, 2002.
12. *Ebada Sarhan, Atif Ghalwash.* Queue Weighting Load-Balancing Technique for Database Replication in Dynamic Content Web Sites. Proceedings of the 9th WSEAS International Conference on APPLIED COMPUTER SCIENCE, 2009.
13. *Zhang Lin, Li Xiao-ping.* A content based dynamic load-balancing algorithm for heterogeneous Web server cluster. ComSIS Vol.7, №1, Special Issue, 2010.

Поступила 15.09.2010р.