ALGORITHM FOR CONSTRUCTION OF SELF-ORTHOGONAL CODES FOR MULTITHRESHOLD DECODERS

G.V.Ovechkin, P.V.Ovechkin, N.N. Grinchenko, V.K.Stolchnev

An algorithm for self-orthogonal codes (SOC) construction is developed. These codes are applied with multithreshold decoders (MTD) and can provide good error correcting ability at high noise level. This algorithm constructs codes with lower error propagation due to optimizing weight of information and check branches. Such codes allow to perform near optimal decoding at lower signal to noise ratio than codes known earlier.

Key words: communication, error-correction coding, self-orthogonal codes, multithreshold decoding, parallel concatenation.

Oveckin Gennady Vladimirovich, doctor of technical sciences, professor, g_ovechkin@mail.ru, Russia, Ryazan, Ryazan State Radio-Engineering University,

Oveckin Pavel Vladimirovich, candidate of technical science, docent, pash_mail@mail.ru, Russia, Ryazan, Ryazan State Radio-Engineering University,

Grinchenko Natalia Nikolaevna, candidate of technical science, docent, grinchenko_nn@mail.ru, Russia, Ryazan, Ryazan State Radio-Engineering University,

Stolchnev Vyacheslav Konstantinovich, candidate of technical science, docent, vstolchnev@gmail.com, Russia, Ryazan, Ryazan State Radio-Engineering University

УДК 004.942

АВТОМАТИЧЕСКАЯ БАЛАНСИРОВКА НАГРУЗКИ В ГИБРИДНЫХ ВЫЧИСЛИТЕЛЬНЫХ СЕТЯХ

А.Н. Привалов, А.К. Клепиков

В работе описывается алгоритм работы балансировщика вычислительной нагрузки в сетях учебных заведений. Приводится описание работы программного балансировщика на основании предложенного алгоритма.

Ключевые слова: балансировка, нагрузка, облачные вычисления, сеть учебного заведения.

В настоящее время учебные заведения и предприятия могут использовать в качестве дополнительной вычислительной мощности вычислительные сервера, расположенные в облаке. Облачные сервера позволяют

расширить вычислительный потенциал учебного заведения, однако выполнение всех вычислений на облачных серверах может быть экономически не целесообразно. Для того чтобы администратор сети для каждой задачи не выбирал среду ее исполнения предлагается использовать программный монитор – распределитель (балансировщик) для балансировки вычислительной нагрузки.

С точки зрения конечного пользователя облачный сервис может выглядеть как скрытая структура, представляющая собою одну целевую структуру либо как виртуальная локальная сеть, позволяющая производить полную настройку всех компонентов. В любом выбранном случае, облачные мощности располагаются на виртуальных машинах, запущенных под управлением гипервизора, предполагается, что пользователь не имеет доступа к физической архитектуре серверов. Таким образом, условно можно полагать, что облако представляет собой одну высокопроизводительную структуру. Вычислительная мощность облачной системы потенциально выше, нежели требуется в единицу времени для любого учебного заведения.

Вычислительный потенциал облачной узла может быть ограничен естественными экономическими рамками, что позволяет говорить о стоимости вычислительных ресурсов при использовании мощности облака за единицу времени. Для распределения вычислительной нагрузки между серверами локальной сети и облаком необходимо учитывать загруженность внутренних серверов, время, отводимое на решение задачи. Для того чтобы в автоматическом режиме балансировать нагрузку потребовалось составить модель работы автоматического распределителя вычислительной нагрузки.

При составлении модели были учтены параметры, влияющие на качественные составляющие процесса обработки данных учебного заведения, это характеристики:

– c - стоимость вычисления;

– t - время, потраченное на решение вычислительной задачи.

Для указанных характеристик справедливы условия минимизации: $c \to \min, t \to \min$.

Параметры, влияющие на характеристики модели были описаны следующим образом:

– μ – загруженность локального сервера в процентах;

– ν - вычислительная сложность задачи.

Для решения поставленной задачи за основу был взят алгоритм распределения нагрузки распределённой вычислительной системы методом перебора и упорядочения её элементов по круговому циклу (round robin) [2]. Используя алгоритм без изменения, предполагалось, что обработчики задач равны по своим вычислительным мощностям, в случае с облаком имеются разно ранговые сервера: это облачный сервер с потенциально

большей вычислительной мощностью и разно ранговые сервера локальной вычислительной сети (ЛВС). В таком случае, алгоритм претерпел изменения, и при каждой новой итерации алгоритма обхода вычислительных узлов, каждый вычислительный узел получает свой коэффициент, который указывает на то, сколько времени потребуется на решения поставленной вычислительной задачи.

После перебора всех серверов ЛВС выбирается тот, который может как можно скорее обработать вычислительную задачу. При этом балансировщик после завершения опроса узловых серверов сохраняет в памяти таблицу полученных значений, и на ее основании может выбирать необходимый сервер через прямое обращение, без потребности в циклическом обходе.

Если после обхода, сервер, выполняющий задачу за приемлемое время не находится, балансировщик отдает задачу на выполнение облачному серверу.

Разработанный балансировщик относится к классу балансировщиков работающих на программном уровне, что позволяет использовать данное ПО независимо от применяемого сетевого оборудования. В данном случае балансировщик решает ряд задач:

– контроль загрузки каждого узлового вычислительного сервера ЛВС;

– организация выбора оптимального вычислительного ресурса для каждой конкретной задачи;

– распределение трафика в гибридной среде включающей в себя ЛВС и облачный узел.

Если имеется n серверов и k вычислительных задач, то схема работы алгоритма следующая: производим оценивание вычислительной сложности задач с целью деления задач на универсальные классы сложности. Данная задача выполняется для каждого вычислительного узла в ЛВС, т.к. потенциально все сервера ЛВС могут иметь разную вычислительную способность и обрабатывать различное количество операций в секунду. На выходе формируется таблица, сохраняемая в файл. Для тестирования вычисления сложности задач использовался статический анализатор кода pvs-studio, который позволяет получать время исполнения программы как на распределенных, так и на одиночных вычислительных станциях (табл. 1).

После первого обхода сети балансировщик имеет в своем распоряжении данные по всем вычислительным узлам сети.

Данные о стоимости вычислений на облаке вводятся администратором сети через интерфейс балансировщика (рис. 1).

При поступлении новой вычислительной задачи в стек задач балансировщика, программа делает круговой опрос серверов ЛВС с целью определения загрузки станций. Все отчеты также сохраняются в файл в таблич-

ном виде и имеют следующую структуру (табл. 2). Все файлы имеют формат csv, что позволяет работать с ними, используя MS Excel с целью корректировки значений в ходе работы программы.

Рис. 1. Установки стоимости облачных вычислений

Таблица 1

Вычислительная сложностью задач по классам и узлам

Uid узла	Класс сложности задачи	Время исполнения
1	1	60с
1	2	120с
1	3	180с
2	1	45с
2	2	90с

Таблица 2

Загруженность вычислительных станций

Uid узла	Загруженность узла (%)
1	76
1	5
1	15
2	24
2	65

Для того, чтобы сохранить возможность вычислительного сервера обрабатывать внутренние входящие запросы балансировщик резервирует 10% вычислительных ресурсов сервера для внутренних операций станции. Таким образом, для любого узла можно посчитать процент свободных ресурсов:

$$w_c = 100 - w_k - w_m,$$

где: w_c - количество свободных ресурсов. Учитывается только любая положительная величина;

w_k - корректировочный коэффициент задаваемый администратором сети. Обычно принимает значение от 5 до 10 процентов;

w_m - количество занятых на данный момент ресурсов сервера.

После составления таблицы загруженности серверов балансиров-

щик определяет сервер, который быстрее остальных обработает поступившую вычислительную задачу. Такая операция выполняется на основании сравнения количества операций, которое может обработать сервер и количества его свободных вычислительных ресурсов (таблица 1, таблица 2). Из n серверов для решения задачи i класса выбирается α сервер, для которого:

$$\begin{cases} w_\alpha \to \min, \\ P_{\alpha,i} \to \max. \end{cases},$$

где w_α - количество занятых ресурсов α сервера;

$P_{\alpha,i}$ - время, необходимое серверу α для решения i-ой задачи.

В случае нахождения m равнозначных серверов выбирается первый из выбранного подмножества m. После нахождения сервера вычислительная задача перенаправляется выбранному серверу α. При поступлении новой задачи происходит обход сети из локальных вычислительных серверов, после чего алгоритм повторяется до тех пор, пока все задачи не будут полностью выполнены.

Параллельно алгоритму распределения балансировщик реализовывает алгоритм сбора данных по выполненным задачам. После решения задачи i формируется набор данных, которые необходимо отправить клиенту с целью указания на завершения выполнения расчетов. В этом случае балансировщик получает пакеты данных через порт 2057. Как только на порт поступает набор данных с идентификатором клиента, данные принимаются и передаются клиенту в локальной сети учебного заведения. В качестве уникального идентификатора выступает IP адрес компьютера пользователя создавшего вычислительную задачу. Данные о поступивших, распределенных и выполняемых задачах можно контролировать через интерфейс балансировщика (рис. 2)

Номер	Инициатор	Обработчик	Статус	Поступило	Осталось времени на выполнение(с.)
1	192.168.5.73	192.168.3.15	Выполнено	17.05.2013 15:43:05	0
2	192.168.5.26	192.168.3.24	Распределено	17.05.2013 15:50:45	10
3	192.168.5.14	192.168.1.10	Принято	17.05.2013 15:50:47	-

Все процессы Показать только со статусом Все статусы

Рис. 2. Монитор контроля распределения вычислительной нагрузки

Балансировщик может работать с любым количеством вычислительных серверов в сети, если только сервер имеет статический IP адрес. Добавление вычислительных узлов в базу балансировщика осуществляется

через соответствующий интерфейс (рис. 3).

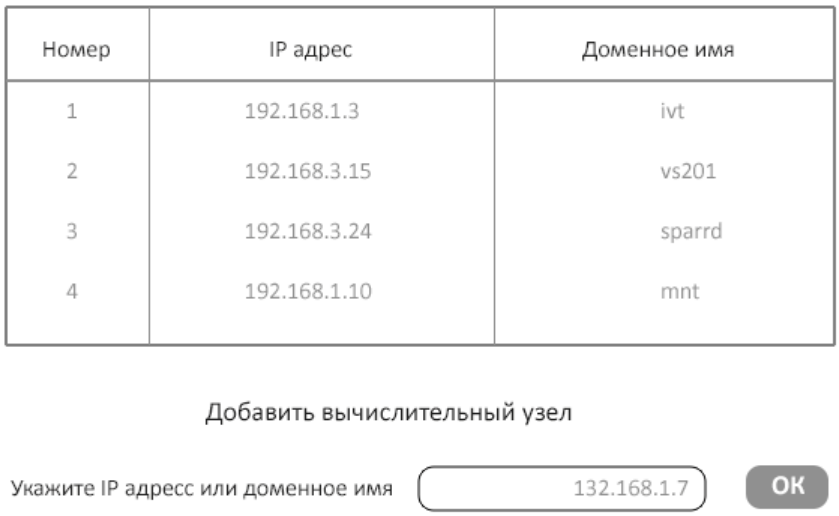

Рис. 3. Добавление вычислительных узлов в таблицу балансировщика

Балансировщик полностью реализовывает алгоритм, основы которого были описаны ранее [2], часть алгоритма была доработана и описана в данной статье. На данный момент балансирщик работает на платформе windows и позволяет взаимодействовать с сетевой структурой учебного заведения различного типа, а также с облачными провайдерами предоставляющими Paas Cloud услуги, когда потребитель получает доступ к использованию информационно-технологических платформ облачной инфраструктуры.

Список литература

1. Карпов, В. Е. Основы операционных систем [Текст] / В.Е. Карпов, К. А. Коньков. М.: Интернет-университет информационных технологий, 2005. 536 с.

2. Клепиков, А.К. Модель распределения ресурсов при "облачных вычислениях" / А.К. Клепиков, А.Н, Привалов // Известия Тульского государственного университета. Технические науки. 2012. С. 151 - 157.

3. Копысов, С. П. Динамическая балансировка нагрузки для параллельного распределённого МДО / С. П. Копысов. М.: Изд-во МГУ, 2003. С. 222. – 228.

Привалов Александр Николаевич, д-р техн. наук, проф., alexandr_prv@rambler.ru, Россия, Тула, Тульский государственный педагогический университет им. Л.Н.Толстого,

Клепиков Алексей Константинович, аспирант, Don-klepikov@yandex.ru Россия, Тула, Тульский государственный педагогический университет им. Л.Н.Толстого

AUTOMATIC LOAD BALANCING IN HYBRID COMPUTER NETWORKS

A.N. Privalov, A.K. Klepikov

The paper describes the algorithm of the computational load balancer in the networks of educational institutions. A description of the software load balancer on the basis of the proposed algorithm.

Key words: balancing, load, cloud computing, network school.

Privalov Alexander Nikolaevich, doctor of technical science, professor, alexandr_prv@rambler.ru, Russia, Tula, Tula State Lev Tolstoy Pedagogical University

Klepikov Alexey Konstantinovich, postgraduate, Don-klepikov@yandex.ru, Russia, Tula, Tula State Lev Tolstoy Pedagogical University

УДК 004.383.3

ОЦЕНКА РЕЗУЛЬТАТОВ МОДЕЛИРОВАНИЯ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ НА БАЗЕ НЕЙРОПРОЦЕССОРОВ

В.А.Романчук, В.Н.Ручкин

Рассмотрены вопросы разработки математического аппарата оценки результатов моделирования вычислительных систем на базе нейропроцессоров. Подробно описан процесс теоретико-множественного анализа вычислительной системы, классификации структур нейропроцессорных систем и аналитические выражения оценок результатов моделирования нейропроцессорных систем. Исследование выполнено при финансовой поддержке РФФИ в рамках проекта № НК-12-07-97516/13.

Ключевые слова: моделирование, нейропроцессор, система.

В настоящее время для процессоров наступил так называемый ”технологический предел”, означающий что они достигли максимального уровня повышения быстродействия. Одним из выходов из данной ситуации является новая элементная база, например использование нейрокомпьютеров. В области нейрокомпьютеров в настоящее время ведутся разработки с использованием новых технологий, перспективными можно назвать технологии создания оптических нейрокомпьютеров, нейрокомпьютеров на пластине, молекулярных и нанонейрокомпьютеров [1].

Специфика нейропроцессорных устройств заключается в том, что это устройства, обладают специальными возможностями (функции активации, взвешенного суммирования и др.), ориентированные на эмуляцию и работу с нейронными сетями. Кроме этого они высокопараллельны, что делает их особенно эффективными для специализированной обработки