УДК 681.3

Ю.В. Ладыженский, канд. техн. наук, доц.,
Е.В. Мельник, магистрант
ГВУЗ «Донецкий национальный технический университет», Украина
alena_mln@list.ru

Программная система для анализа производительности программного обеспечения сетевых процессоров

Рассмотрена методика применения метода средних значений для анализа производительности программного обеспечения сетевых процессоров. На основе этого метода разработаны последовательная и параллельная программы для расчетов. Проведен расчет характеристик и построены зависимости для показателей эффективности программного обеспечения для шага получения пакетов для коммутаторов четвертого и седьмого уровней.

Ключевые слова: сетевой процессор, модель производительности, анализ приложений, коммутатор.

Введение

Сетевые процессоры (СП) предназначены для обеспечения производительности и гибкости при обработке потоков пакетов за счет параллельной и программируемой архитектуры. Она дает им преимущество перед процессорами общего назначения в производительности и перед аппаратными решениями в гибкости. Важно изучить особенности архитектуры СП, чтобы использовать преимущества блоков СП для достижения требуемой производительности для сетевых приложений [1].

Производительность программного обеспечения (ПО) является важным аспектом в разработке компьютерных сетей. Актуальность вопроса объясняется постоянно возрастающей сложностью и значимостью программных средств. Особое внимание производительности уделяется:

− в инженерных и научных разработках, где часто производятся сложные длительные вычисления, а процессорное время на кластерных системах дорого и ограничено;

− в web-приложениях, в которых время генерации страницы критично для пользователя и напрямую зависит от объемов серверных мощностей;

− в встраиваемых программных продуктах и т.д.

Тщательный анализ и учет производительности используемого программного продукта может существенно сократить стоимость оборудования [2].

Сетевые процессоры не являются исключением. Постоянный рост трафика в компьютерных сетях делает актуальной проблему создания высокопроизводительных специализированных сетевых процессоров и их программного обеспечения для обработки потоков данных на разных уровнях стека протоколов.

Для программного обеспечения СП критическими характеристиками являются оценка производительности и время отклика.

Анализ производительности программного обеспечения требует определения и задания параметров используемых ресурсов, характеризующих нагрузку компонентов модели программы, для расчета по ним характеристик производительности.

Общая архитектура сетевого процессора

Архитектуры СП от разных производителей значительно различаются, но они все придерживается одних и тех же базовых концепций и структур. В общем случае СП состоит из множества программируемых пакетных процессоров (ПП) с высоко параллельной архитектурой, программируемого управляющего процессора (УП), аппаратного сопроцессора (Со) или ускорителей для обычных сетевых операций, высокоскоростных интерфейсов памяти и высокоскоростных сетевых интерфейсов. На рис. 1 показана общая архитектура сетевого процессора.

Типичный СП состоит из массива программируемых пакетных процессоров (ПП) с высоко параллельной архитектурой, программируемого управляющего процессора (базового процессора), аппаратных сопроцессоров или ускорителей для общих сетевых операций, высокоскоростных интерфейсы памяти и высокоскоростных сетевых интерфейсов.

Метод оценки производительности ПО СП

Для оценки производительности программного обеспечения коммутаторов и маршру-

тизаторов, построенных на сетевых процессорах, используется модель, изображенная на рис.2.

Программные приложения на основе СП проходят три этапа конвейерной обработки: получение пакета (ЭП), обработка пакета (ЭО) и передача пакета (ЭПП). Получение и передача пакетов могут быть реализованы либо на одном процессоре обработки пакетов, либо параллельно на нескольких процессорах. Обработка пакетов может осуществляться на нескольких таких процессорах параллельно или конвейерно.

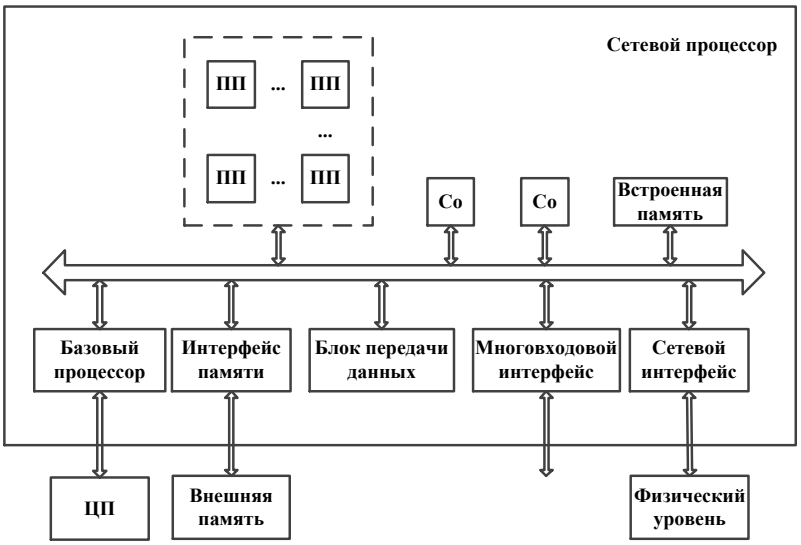

Рисунок 1 – Обобщенная архитектура сетевого процессора

В этой модели вводятся отдельные входные очереди для каждого процессора обработки пакетов (ПП), сопроцессора (Со) и модуля памяти. Рассматриваются три уровня иерархии памяти: встроенная память (ВП), статическая память (СтП) и динамическая память (ДП).

Управляющий процессор (УП) используется только для управления конфигурациями и обработки исключений. Он мало влияет на общую производительность приложения и не включен в состав модели.

Получение и передача пакетов осуществляются одним пакетным процессором.

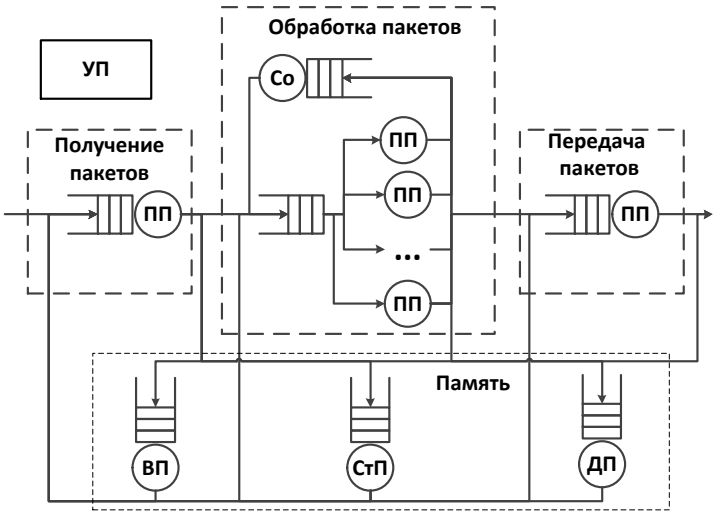

Рисунок 2 – Сетевая модель системы

Обработка пакетов осуществляется параллельно несколькими пакетными процессорами. Использование механизма многопоточности позволяет пакетным процессорам не ждать ответа от сопроцессора. Сопроцессоры моделируются так же, как и пакетные процессоры.

Производительность на уровне компонентов модели оценивается с использованием теории

замкнутых сетей очередей. Предполагается, что внутри компонента циркулирует заданное количество запросов.

Для расчета характеристик замкнутой сети массового обслуживания (СМО) применяется метод приближенного анализа средних значений (MVA)[5]. Время пребывания (R'_i), пропускная способность (X_0) и средняя длина очереди (Q_i) вычисляются по формулам:

$$R_i^{'}(n) = \begin{cases} D_i & - ресурс\ с\ задержкой, \\ D_i[1+Q_i(n-1)] & - ресурс\ с\ очередью, \end{cases} \quad (1)$$

$$X_0(n) = \frac{n}{\sum_{i=1}^{K} R_i^{'}(n)}, \quad (2)$$

$$Q_i(n) = X_0(n) * R_i^{'}(n). \quad (3)$$

Здесь i - индекс блока архитектуры СП, K - общее количество блоков, n - общее число запросов, находящихся в замкнутой сети. Затраты времени на обслуживание D_i в каждом блоке СП определяются путем анализа псевдокода алгоритма обработки пакета, реализуемого блоком.

Перед использованием алгоритма анализа средних значений необходимо найти границу пропускной способности компонента.

В соответствии с законом затрат времени на обслуживание [6], $D_i=U_i/X_0$, где U_i - загрузка блока i, X_0 - пропускная способность компонента. Так как загрузка любого ресурса не может превышать 100%, $X_0 \le 1/D_i$ для любого ресурса i. Тогда:

$$X_0 \le \frac{1}{\max_{i \in 1,K} D_i} \quad (4)$$

Максимальная пропускная способность компонента определяется блоком с наибольшими затратами времени на обслуживание. Такой блок называют узким местом в замкнутой сети.

Для СП, работающего в сети, легко получить частоту поступления запросов и средний размер пакета. Однако сложно измерить средние затраты времени на обслуживание для каждого запроса и число пакетов в очереди на каждый блок. Затраты времени на обслуживание можно оценить с помощью анализа псевдокода каждого приложения. Например, ПП в сетевом процессоре IXP2400 являются RISC процессорами и большинство команд выполняются за один такт. Поэтому затраты на обслуживание для ПП можно получить на основе количества команд для обработки каждого запроса.

Средняя эффективность обращения к памяти выражается через среднее времени ожидания доступа и количество ссылок на ячейки памяти для каждого запроса.

Ниже представлены формулы для расчета затрат времени на обслуживание для ПП (5) и затрат времени на обращение к памяти (6) – (8):

$$D_{ПП} = \frac{NI}{CR}, \quad (5)$$

$$D_{СтП} = \frac{RC_1 * l_1}{CR}, \quad (6)$$

$$D_{ДП} = \frac{RC_2 * l_2}{CR}, \quad (7)$$

$$D_{ВП} = \frac{RC_3 * l_3}{CR}, \quad (8)$$

где NI – количество команд, необходимых для выполнения операции, CR – тактовая частота процессора, RC_i – количество обращений к ячейкам памяти для i-го типа памяти, l_i – задержка для i-го типа памяти: $i=1$ – встроенная память, $i=2$ – статическая память, $i=3$ – динамическая память.

Компоненты для оценки производительности на этапах приема и передачи пакетов состоят из одного ПП и трех блоков памяти: СтП, ДП и ВП. ПП содержит несколько аппаратных потоков, что не позволяет рассматривать его как блок не зависящий от нагрузки.

Потоки в ПП моделируются с использованием только блоков, не зависящих от нагрузки, в замкнутой сетевой модели (рис. 3).

Предполагается, что в этой модели в качестве количества потоков используется количество запросов.

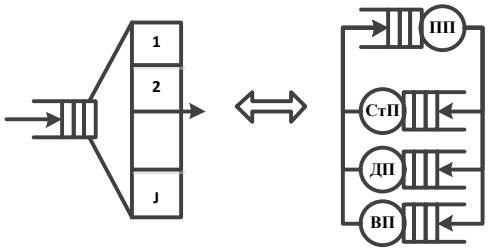

Рисунок 3 – Моделирование нескольких потоков в пакетном процессоре

На этапе обработки пакетов есть несколько пакетных процессоров, которые используются для обработки запросов из одной очереди. Очередь с m ресурсами и затратами времени на обслуживание D на каждом ресурсе может быть заменена в сети массового обслуживания двумя ресурсами. Это ресурс, независящий от нагрузки, с затратами времени на обслуживание D, и ресурс с задержкой (без очереди) – $D*(m-1)/m$.

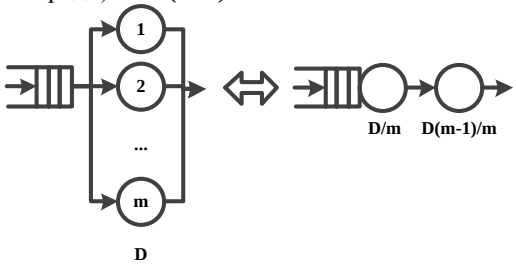

Рисунок 4 – Моделирование параллельных пакетных процессоров

Таким образом, несколько процессоров моделируются как один процессор и ресурс с задержкой (рис. 4).

Для шага обработки пакетов используется модель, изображенная на рис. 5. Компонент для этого шага состоит из одного ПП с ресурсом задержки (РЗ), трех модулей памяти: СтП, ДП и ВП.

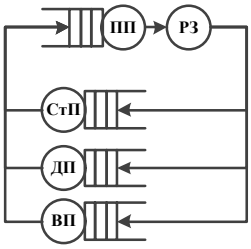

Рисунок 5 – Замкнутая модель для шага обработки пакетов

При параллельном использовании нескольких ПП на стадии обработки, они заменяются одним процессором (АПП) и ресурсом с задержкой (ЗПП). Следовательно, затраты времени на обслуживание для агрегированного процессора и блока с задержкой отличаются от аналогичных затрат времени для одного ПП:

$$D_{АПП} = \frac{IC}{CR * NME}, \quad (9)$$

$$D_{ЗПП} = \frac{IC * (NME - 1)}{CR * NME}, \quad (10)$$

где NME – количество ПП.

С помощью формул (1) – (4) можно оценить производительность для замкнутых моделей СМО. Это позволяет моделировать каждый многопоточный процессор как однопоточный процессор с переменной скоростью облуживания (рис. 3).

Замкнутые модели используется для расчета характеристик на уровне компонентов (прием, передача и обработка пакетов). Для анализа на уровне системы модель рассматривается как разомкнутая СМО с неограниченными очередями и переменной скоростью обслуживания. В качестве скорости обслуживания используется производительность, полученная при анализе на уровне компонентов.

Для того, чтобы получить среднее количество пакетов в системе $\overline{N}$, используется формула [7]:

$$\overline{N} = p_0 \left[\sum_{k=1}^{J} \frac{k\lambda^k}{\beta(k)} + \frac{\rho\lambda^J \left[\rho + (J+1)(1-\rho)\right]}{(1-\rho)^2 \beta(J)} \right], \quad (11)$$

где

$$p_0 = \left[1 + \sum_{k=1}^{J} \frac{\lambda^k}{\beta(k)} + \frac{\lambda^J}{\beta(J)} \frac{\rho}{1-\rho} \right]^{-1}, \quad (12)$$

λ - интенсивность входного потока,

$\beta(k) = X(1) \times X(2) \times \ldots \times X(k)$, $\rho = \lambda / X(J)$.

Средняя производительность для разомкнутых моделей СМО с бесконечной очередью X равна средней частоте поступления запросов λ: $X = \lambda$. Максимальная интенсивность входного потока не превышает максимальную производительность: $\lambda < X(J)$.

Время отклика – это среднее время, которое каждый запрос (пакет) проводит внутри СП.

Время отклика определяется согласно закону Литтла:

$$R = \overline{N} / X. \quad (13)$$

Заключительным этапом оценки характеристик является получение производительности и времени отклика для всего СП:

$$X_0 = \min(X_{эп}, X_{эо}, X_{эпп}), \quad (14)$$

$$R = R_{эп} + R_{эо} + R_{эпп}. \quad (15)$$

Проектирование и реализация программной системы для расчетов

Для проведения расчетов разработана программная система на языке C# с использованием среды MS Visual Studio 2012.

Исходными данными программной системы являются:

− количество инструкций алгоритма обработки пакетов;

− количество обращений к статической памяти (СтП);

− количество обращений к динамической памяти (ДП);

− количество обращений к встроенной памяти (ВП);

− количество блоков внутри каждого компонента;

− количество пакетов (запросов) внутри компонента;

− тактовая частота пакетного процессора (ПП);

− количество потоков ПП;

− время ожидания обслуживания для каждого типа памяти.

Программа позволяет вычислить:

− время обслуживания для каждого блока;

− максимальные время обслуживания для каждого этапа;

− максимальную производительность для каждого этапа;

− среднее время пребывания в компоненте для каждого этапа;

− производительность для всей системы;

− время отклика для всей системы.

Технические характеристики сетевого процессора Intel IXP2400, используемые в расчетах, представлены в табл. 1.

Основная функциональность системы представлена на диаграмме вариантов использования (рис. 6). Приложение позволяет производить вычисления на уровне системы, а также на уровне компонентов.

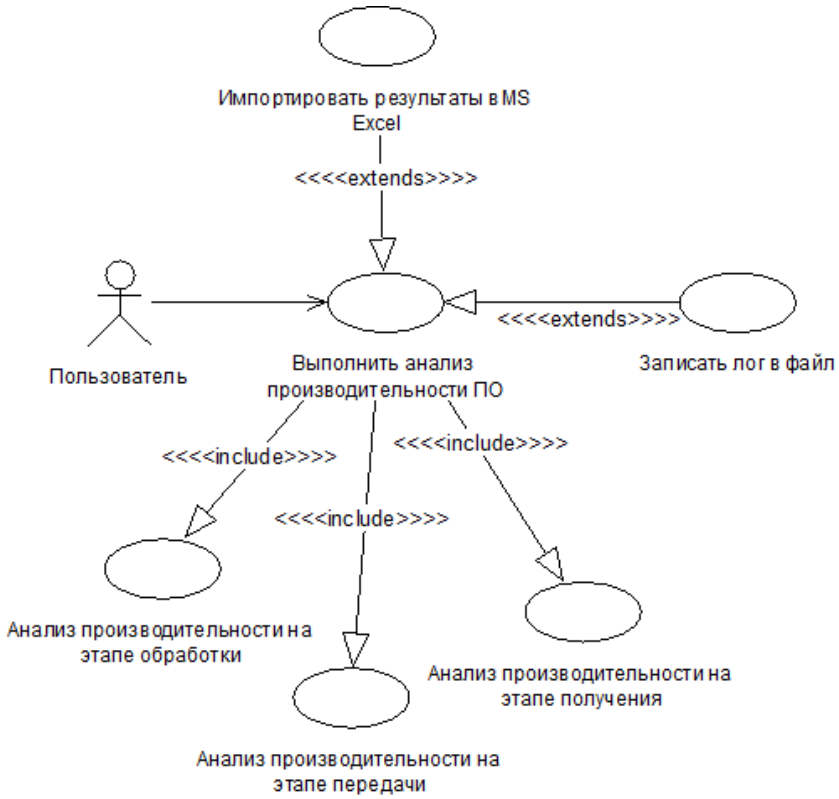

Рисунок 6 – Диаграмма вариантов использования системы анализа производительности

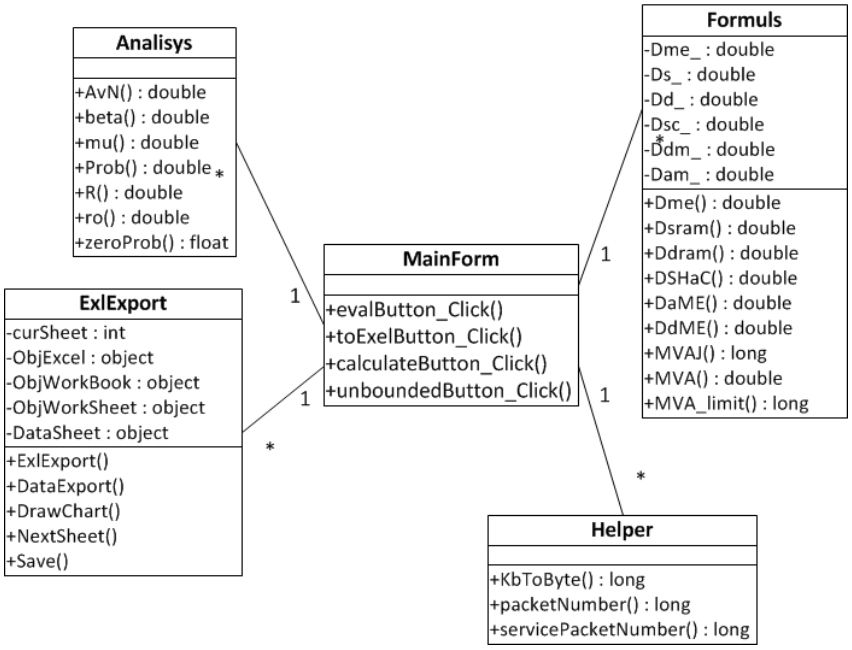

Рисунок 7 – Диаграмма классов системы анализа производительности

Таблица 1. Характеристики сетевого процессора Intel IXP2400

Тактовая частота ПП (МГц)	Потоки ПП	Время ожидания (в тактах процессора)		
		ВП	СтП	ДП
600	8	16	90	120

Таблица 2 – Описание классов системы

Класс	Назначение
MainForm	Класс главной формы. Реализует интерфейс пользователя.
Formulas	Основной класс для вычислений. Реализует алгоритм анализа средних значений. Служит для получения характеристик компонентов системы, в том числе и производительности.
Analysis	Класс для вычисления времени отклика и характеристик, связанных с ним.
Helper	Вспомогательный класс для получения исходных данных для расчетов.
ExlExport	Экспортирует полученные результаты в MS Excel в виде текста и рисует по ним графики.

Анализ производительности выполняется в 4 шага:

- анализ на этапе получения пакетов;
- анализ на этапе обработки пакетов;
- анализ на этапе передачи пакетов;
- сбор результатов предыдущих шагов и получение производительности и времени отклика всей системы.

Для программной реализации модели анализа производительности разработаны следующие классы (табл. 2).

Диаграмма классов, отражающая структуру выделенных классов, приведена на рис. 7.

Параллельная реализация программной системы

Для ускорения вычислений была разработана параллельная версия программной системы с использованием MPI.NET.

На рис. 8 изображен параллельный алгоритм оценки характеристик ПО сетевых процессоров.

Алгоритм анализа средних значений применяется параллельно для получения производительности и времени отклика для шагов, представленных на рис. 9.

Исходные данные: Di, Q

Инициализация

Вычисление производительностей для всех шагов с помощью алгоритма анализа средних значений – формулы (1) – (3).

Сбор данных о производительности на управляющем процессоре.

Получение производительности для СП по формуле (14).

Вычисление времен отклика для всех шагов по формулам (11) – (13).

Сбор данных о времени отклика на управляющем процессоре.

Получение времени отклика для СП по формуле (15).

Рисунок 8 – Параллельный алгоритм вычислений

Рисунок 9 – Потоки пакетов на пакетном процессоре

На рисунке 10 представлена схема обмена между ветвями алгоритма. Был использован коллекторный обмен (КО, All-to-one Reduction) – сбор всех данных на одном процессоре.

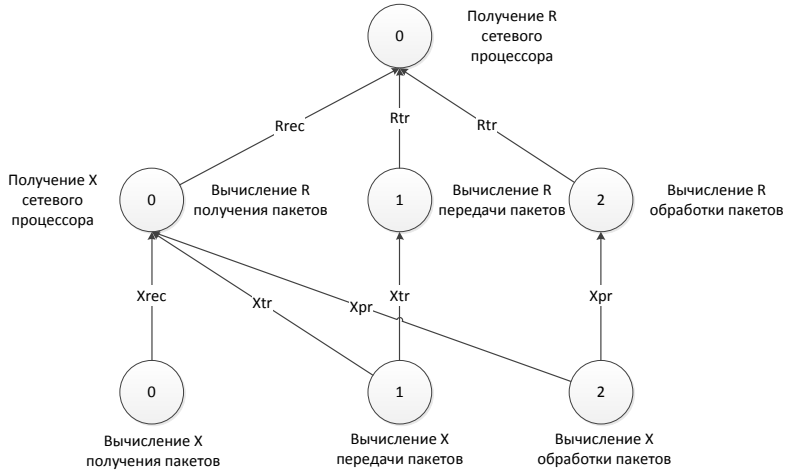

Рисунок 10 – Схема обменов между ветвями алгоритма

Пример анализа производительности коммутаторов на СП

В [3] рассматриваются коммутаторы седьмого (L7) и четвертого уровня (L4) на основе многоядерных сетевых процессоров.

На рис. 11 представлен механизм «рукопожатия. Эта операция для коммутаторов четвертого и седьмого уровня происходит по-разному.

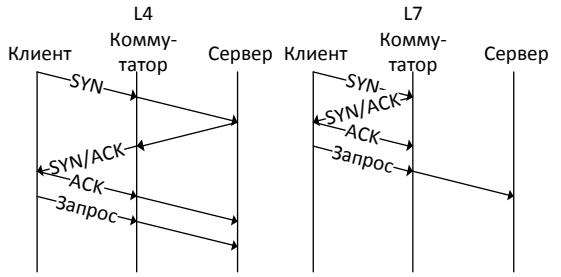

Рисунок 11 – Сравнение коммутаторов четвертого и седьмого уровней

Коммутатор четвертого уровня выполняет маршрутизацию, независимую от контента, на уровне TCP. Коммутатор определяет целевой сервер, когда клиент запрашивает TCP-соединение с помощью пакета TCP SYN. Затем коммутатор четвертого уровня ассоциирует пакеты принадлежащие этому соединению с тем же сервером. Этот механизм является эффективным для пакетов, которые не поднимаются на уровень приложений в стеке сетевых протоколов.

Коммутатор седьмого уровня (контентозависимый) работает на уровне приложений. Такой коммутатор обеспечивает управление трафиком приложений с помощью глубокого анализа пакетов.

Коммутаторы седьмого уровня выполняют стандартную процедуру «рукопожатия» с клиентами на их стороне без участия сервера. Коммутатор поддерживает постоянное соединение с каждым отдельным сервером, что позволяет снизить накладные расходы на установление соединения.

Экспериментальные расчеты проведены с использованием стандартного кода этапа получения пакетов, описанного в [4]. Предполагается, что средний размер пакетов равен 512 байтам, с помощью СП он разбивается на 8 микро пакетов (МП). Каждый МП содержит 64 байта. СП получает пакет, читая один МП за раз (вместо того, чтобы прочитать один байт) и вновь собирает пакет из МП.

Когда пакет приходит, он занимает 448 цикла инструкций, использует по одной ссылке на встроенную и статическую память, а также 16 ссылок на динамическую память. В [3] приведено разбиение кода на количество инструкций и ссылок на обращение к памяти.

Каждый http-запрос включает в себя 12 сообщений клиенту и серверу.

Результаты расчета представлены на рис. 12. Слева в окне программы задаются исходные параметры, справа показан лог выполнения программы: полученные затраты на обслуживание для каждого ресурса, максимальные затраты на обслуживание, максимальная производительность, которая достигается системой, а также производительность в зависимости от количества пакетов в системе.

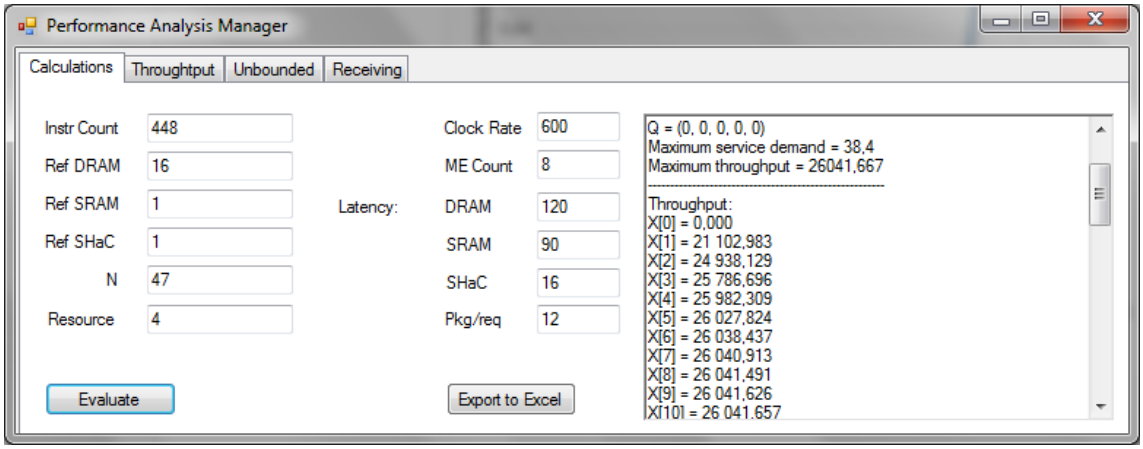

Рисунок 12 – Результаты работы программы

На рис. 13 изображены затраты времени на обслуживание для всех ресурсов, используемых на этапе получения пакетов. Как видно из графика, наибольшими являются затраты на обслуживание динамической памяти. Это можно объяснить тем, что в анализируемом алгоритме на этапе получения пакетов используется 16 ссылок на ячейки динамической памяти против 1 ссылки для других типов памяти. Максимальные затраты на обслуживание определяют предел производительности.

В результате расчетов была получена зависимость производительности от количества запросов (пакетов), находящихся в системе (рис.

14). Как видно из графика, предел производительности системы на шаге получения пакетов достигается при количестве пакетов равном 6.

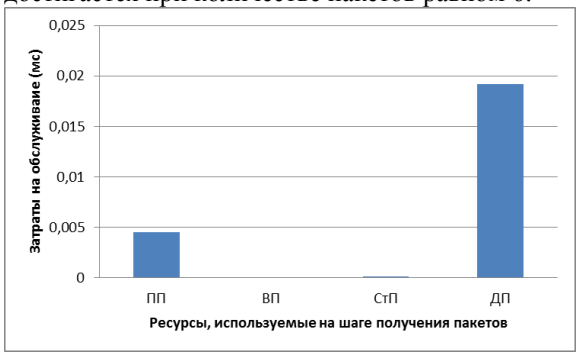

Рисунок 13 – Затраты времени на обслуживание для шага получения пакетов

Рисунок 14 – Зависимость производительности от количества пакетов в системе на шаге получения пакетов

На рис. 15 представлена зависимость времени отклика от интенсивности входного потока пакетов, полученная в результате работы программы. Как можно увидеть из графика, время отклика до того, как система достигнет предела производительности, является очень малым. Но как только система достигает своей максимальной производительности, происходит резкий рост времени отклика.

Рисунок 15 – Зависимость времени отклика от интенсивности входного потока на шаге получения пакета

Заключение

Рассмотрена методика применения метода средних значений для анализа производительности программного обеспечения сетевых процессоров, учитывая особенности архитектуры и организации вычислений СП.

На основе этого метода разработаны последовательная (на C#), выполняющая анализ эффективности ПО сетевых процессоров, а также экспорт полученных данных в пакет MS Excel, и параллельная (с MPI.NET) программы для расчетов.

Проведен расчет характеристик и построены зависимости для показателей эффективности программного обеспечения.

Список литературы

1. J. Lu. Analytical Performance Analysis of Network-Processor-Based Application Designs / J. Lu, J. Wang // Computer Communications and Networks: Proceedings 15th International Conference, (October 2006), 2006.
2. Анализ производительности программного обеспечения при помощи математического планирования эксперимента [Электронный ресурс]. – Режим доступа: http://habrahabr.ru/post/129346/. – Заглавие с экрана.
3. J. Lu. Performance Modeling and Analysis of Web Switch / J. Lu, J. Wang // Proceedings International CMG Conference, 2005. – PP. 665-672.
4. E. Johnson. IXP2400/2800 Programming: The Complete Microengine Coding Guide / E. Johnson, A. Kunze. – Intel Press, 2003.
5. M. Reiser. Mean-Value Analysis of Closed Multichain Queuing Networks / M. Reiser, S. Lavenburg // Journal of the Association for Computing Machinery. – 1980. – № 2.
6. P. Denning. The Operational Analysis of Queueing Network Models / P. Denning, J. Buzen // Computing Survey. – 1978. – № 3.
7. D. A. Menasce. Capacity Planning for Web Swervices: metrics, models, and methods. / D. A. Menasce, V. A. F. Almeida. – Prentice Hall, 2002.

Надійшла до редакції 10.07.2013

Ю.В. ЛАДИЖЕНСЬКИЙ, О.В. МЕЛЬНИК
ДВНЗ «Донецький національний технічний університет»

ПРОГРАМНА СИСТЕМА ДЛЯ АНАЛІЗУ ПРОДУКТИВНОСТІ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ КОМУТАТОРІВ НА МЕРЕЖНИХ ПРОЦЕСОРАХ

Розглянуто методику застосування методу середніх значень для аналізу продуктивності програмного забезпечення мережних процесорів. На основі цього методу розроблені послідовна і паралельна програми для розрахунків. Проведено розрахунок характеристик і побудовані залежності для показників ефективності програмного забезпечення для кроку отримання пакетів для комутаторів четвертого і сьомого рівнів.

Ключові слова: мережний процесор, модель продуктивності, аналіз програмного забезпечення, комутатор.

Yu.V. LADYZHENSKIY, Ye.V. MELNIK
Donetsk National Technical University

THE SOFTWARE SYSTEM FOR APPLICATION PERFORMANCE ANALYSIS OF NP-BASED WEB SWITCHES

This paper reviews an analytical performance model for analyzing the performance of application designs of NP-based switches and routers. The queuing network approach is applied to model NP resources and application work flows. The Mean Value Analysis algorithm is used to obtain performance metrics at the component level and the operational analysis is used to obtain response time metrics at the system level. At first service rate, throughput, and response time should be measured at the component level using the Mean Value Analysis (MVA) method. The next step is to measure throughput and response time at the system level by combining measuring results at the component level. Finally, we can measure the throughput and the response time of the whole network processor. This method considers the specificities of modeled network processor architectures and a computational process. The software system based on the theoretical performance analysis method is designed and developed. It consists of two applications for evaluations: C#-based sequential program and parallel program on MPI.NET. Software system allows evaluating the performance of NP-based application designs and exporting results to MS Excel. NP-based Web switch applications typically go through three major pipeline stages: packet receiving, packet processing, and packet transmitting. The MPI.NET program allows evaluating performance of these stages in parallel. As an example, the performance model was applied to the standard code for packet receiving stage of NP-based layer-4 (L4) switch and layer-7 (L7) switch. The throughput and response time of the packet receiving stage are obtained. The relationships between the critical performance metrics are plotted. There are dependencies between the arrival rate and the response time, between the arrival rate and the throughput. The bottleneck of the NP application is on DRAM access. The results can be useful for designers to troubleshoot performance bottlenecks and define the capacity of the switch.

Keywords: network processor, performance model, software analysis, web switch.