УДК 681.3

# МОДЕЛИРОВАНИЕ РАБОТЫ ПРИЛОЖЕНИЙ НА СЕТЕВЫХ ПРОЦЕССОРАХ

Грищенко В.И., Ладыженский Ю.В.
Донецкий Национальный Технический Университет

*Увеличение сетевого трафика и необходимость решения задач обслуживания сети со скоростью канала данных повлекло за собой создание сетевых процессоров (СП). В предложенной статье описывается комплексная методика моделирования сетевых процессоров (КМСП) приводятся результаты моделирования, а также указываются недостатки методики и пути ее развития.*

***Методика моделирования.*** В работах [1, 2] представлена комплексная методика исследования производительности сетевых процессоров. Она состоит из четырех основных этапов (см. рис. 1): выбор приложения и архитектуры, имитационное моделирование работы приложения, аналитический расчет эффективности выполнения приложения на выбранной архитектуре и визуализация результатов.

Имитационное моделирование включает расчет площади и энергопотребления кэша с использованием инструмента CACTI [3] и моделирование работы приложения в системе SimpleScalar [4]. CACTI представляет собой консольную утилиту, позволяющую на основании параметров кэша (размер, ассоциативность, технология производства и др.) определять занимаемую площадь, объем рассеиваемой энергии, скорость доступа и другие характеристики. Характеристики кэша не зависят от выполняемого приложения, поэтому могут быть рассчитаны один раз, и многократно использоваться в последствии.

Система SimpleScalar используется для детального моделирования работы RISC-процессора. Использование SimpleScalar позволяет получить характеристики выполнения приложения: интенсивность промахов кэша данных и инструкций, число операций чтения/записи и общее количество инструкций. Для анализа в SimpleScalar приложение должно быть предварительно скомпилировано с использованием специализированного кросс-компилятора, поставляемого вместе с системой.


Рис. 1 – Структура методики моделирования

Таблица 1 – Интенсивность промахов кэша данных для приложения AntNet (объем кэша — 1 КБ)

| Функция | Файл | Количество промахов | Процент от общего числа |
|---|---|---|---|
| antnet_route_packet() | antnet.h | 1273879 | 81.88 |
| antnet_init_router() | antnet.h | 107213 | 6.85 |
| fread() | fread.c | 46155 | 2.98 |
| malloc() | malloc.c | 33130 | 2.11 |
| _free_internal() | free.c | 19383 | 1.22 |
| antnet_read_packet() | antnet.h | 12756 | 0.83 |

Встроенные механизмы профилирования позволяют находить «узкие места» исследуемых приложений. В таблице 1 приведены результаты профилирования алгоритма маршрутизации AntNet, который подробно описан в работах [6, 7]. Можно видеть, что более 80% промахов кэша приходится на поиск в таблице маршрутизации и это является самым узким местом алгоритма.

***Аналитическая модель.*** Выбор структуры сетевого процессора непосредственно влияет на используемую аналитическую модель. В проведенных исследованиях использовалась кластерная структура процессора, представленная на рисунке 2. В ней СП состоит из однотипных вычислительных ядер, которые объединяются в кластеры

(группы) и внутри кластера разделяют общий интерфейс к внешней памяти. Каждое ядро имеет раздельные кэши команд и данных, и аппаратно поддерживает работу нескольких потоков. Переключение между потоками осуществляется с нулевой задержкой и происходит в те моменты, когда программа ожидает завершения команды обращения к ОЗУ.

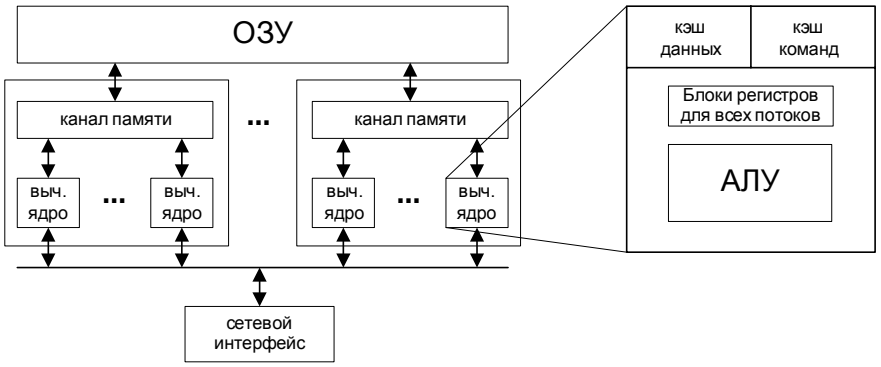


Рис. 2 – Обобщенная структура СП

При такой архитектуре вычислительная мощность всего СП определяется как сумма мощностей всех его ядер:

$$IPS = \sum_{j=1}^{m} \sum_{k=1}^{n} \rho_{p_{j,k}} \cdot f_{p_{j,k}} \qquad (1)$$

где $\rho_{p_{j,k}}$ - загруженность ядра (j, k), команд/такт, $f_{p_{j,k}}$ - тактовая частота процессора, такт/секунда, m – число кластеров в СП, n – количество процессорных ядер в каждом кластере.

Рассматривать ядра с различной производительностью имеет смысл только в тех случаях, если они выполняют различные приложения, но для данной структуры все ядра будут параллельно обрабатывать поток данных, и, как следствие, будут испытывать одинаковую нагрузку.

Загруженность отдельного многопоточного процессорного ядра с кэшами данных и команд определяется по формуле [9]:

$$\rho_p(u) = 1 - \frac{1}{\sum_{i=0}^{u} \left( \frac{1}{p_{miss} \tau_{mem}} \right)^i \frac{u!}{(u-i)!}} \qquad (2)$$

где $\rho_p$ - загруженность вычислительного ядра, u - число аппаратно поддерживаемых потоков, $p_{miss}$ - совокупная интенсивность промахов кэша, $\tau_{mem}$ - время доступа к памяти.

Время доступа к памяти учитывает время ожидания запроса в очереди ($\tau_Q$), время физического отклика памяти ($\tau_{DRAM}$) и время передачи данных через кэш ($\tau_{transmit}$):

$$\tau_{mem} = \tau_Q + \tau_{DRAM} + \tau_{transmit} \quad (3)$$

Интенсивность промахов кэшей зависит от приложения, выполняемого на СП:

$$p_{miss} = mi_c + (f_{load} + f_{store}) \cdot md_c \quad (4)$$

где $mi_c$ и $md_c$ - интенсивности промахов кэшей команд и данных соответственно, $f_{load}$ и $f_{store}$ - частота появления в коде программы команд чтения и записи в память.

***Результаты моделирования.*** С помощью представленной методики были исследованы тестовый набор сетевых приложений CommBench [5] и алгоритм маршрутизации AntNet [6, 7]. Набор CommBench охватывает практически все задачи, которые выполняет современный маршрутизатор: фрагментация, маршрутизация, планирование очередей, обработка медиа-потока, шифрование данный и др. Результаты моделирования набора CommBench показаны в таблице 2, детальное же их рассмотрение можно найти в работе [8]. Представленные данные соответствуют однопроцессорной системе, многоядерные процессоры будут рассматриваться в дальнейших исследованиях. В качестве критерия оптимизации выбрано отношение вычислительной мощности к площади процессора (IPS/A).

Алгоритм AntNet использует принципы поведения колонии муравьев при нахождении кратчайших путей к источнику пищи. Каждый муравей оставляет за собой след из активных веществ — феромонов. И когда следующий муравей окажется рядом, то он с большей вероятностью выберет то направление, по которому до него прошел другой муравей, тем самым он увеличивает концентрацию феромонов на этом маршруте. Чем больше муравьев изберет маршрут, тем больше на нем будет феромона и тем больше вероятность того, что остальные последует за ними. Таким образом, через некоторое время вся колония будет двигаться по одному пути.

В алгоритме AntNet в качестве муравьев используются активные агенты — специализированные пакеты, несущие информацию о состоянии сети на своем пути. Когда такой муравей двигается к адресу назначения — он собирает статистику о задержках на каждом маршрутизаторе, а по возвращении он обновляет таблицы маршрутизации в соответствии с собранной статистикой.

Таблица 2 — Результаты исследования пакета CommBench

| Приложение | Описание | Оптимальный кэш данных, КБ | Оптимальный кэш команд, КБ |
|---|---|---|---|
| FRAG | Фрагментация | 2 | 2 |
| DRR | Планирование очередей | 2 | 2 |
| RTR | Маршрутизация | 4 | 32 |
| JPEG (сжатие) | Упаковка изображения в формат JPEG | 2 | 1 |
| JPEG (распаковка) | Восстановление изображения из формата JPEG | 16 | 32 |
| REED (кодирование) | Избыточное кодирование | 4 | 1 |
| REED (декодирование) | | 4 | 1 |
| CAST (кодирование) | Шифрование | 16 | 32 |
| CAST (декодирование) | Дешифрование | 16 | 64 |
| ZIP (сжатие) | | 4 | 2 |
| ZIP (распаковка) | | 4 | 2 |

Для алгоритма AntNet оптимальным оказывается процессор с кэшем данных объемом 2 КБ и кэше команд объемом 4 КБ. Однако показано, что при увеличении объема кэша данных до 128 КБ производительность возрастает вдвое, а двукратный прирост производительности может иметь существенное значение.

***Совершенствование модели.*** Кластерная структура сетевого процессора позволяет исследовать многоядерные устройства с параллельной обработкой данных, однако существуют коммерческие процессоры, организованные в виде конвейеров (например, CISCO PFX [10] или семейство сетевых процессоров от Intel [11] с настраиваемой системой внутренних связей). В конвейерных структурах актуальным становится вопрос передачи информации между вычислительными ядрами. В настоящий момент распространены три подхода: передача данных через внешнюю память (рис. 3.а), использование специализированной внутренней памяти, интегрированной в кристалл процессора (рис. 3.б) и

использование разделяемых регистров (рис. 3.в). Второй и третий способ похожи между собой, их отличие заключается только в максимальных объемах передаваемой информации и скорости доступа к данными. В случае использования внешнего ОЗУ объемы памяти обмена многократно возрастают, но также увеличивается и время отклика системы.

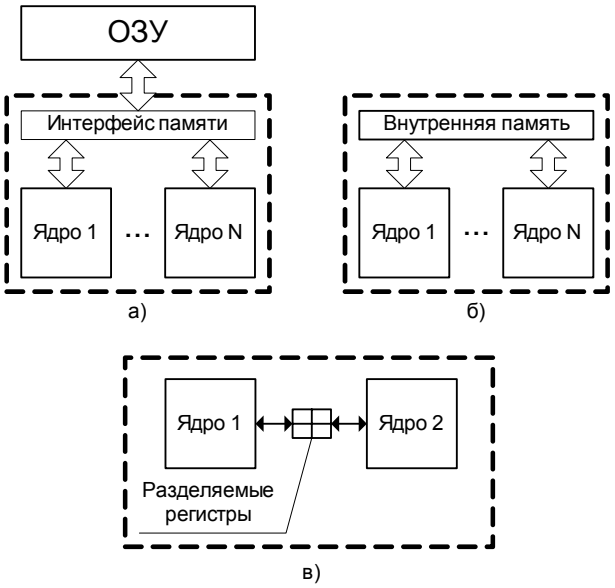


Рис. 3 – Способы организации взаимодействия между процессорными ядрами

В существующих СП часто наиболее трудоемкие функции реализуется аппаратно в виде сопроцессоров. Они позволяют решать специфические задачи (криптография, сбор статистики) практически на частоте канала данных. Учет работы сопроцессоров позволит повысить точность моделирования и получать более достоверные результаты.

***Выводы.*** В статье представлена методика моделирования сетевых процессоров, показаны ее структура и назначение отдельных этапов моделирования. Продемонстрированы результаты применения методики для исследования тестового набора приложений. Предложены пути совершенствования модели для исследования перспективных архитектур сетевых процессоров.

***Литература***

1. Tilman Wolf, Mark A. Franklin, "Performance Models for Network Processor Design," IEEE Transactions on Parallel and Distributed Systems, vol. 17, no. 6, pp. 548-561, Jun., 2006.

2. Ладыженский Ю.В., Грищенко В.И. Моделирование сетевых процессоров пакетной обработки данных. ИНТЕРНЕТ-ОБРАЗОВАНИЕ-НАУКА, пятая международная конференция ИОН-2006. 10-14 октября, 2006. Сборник материалов конференции. Том 2. – Винница: УНИВЕРСУМ-Винница, 2006, стр. 417-422.

3. Premkishore Shivakumar and Norman P. Jouppi, “CACTI 3.0: An integrated cache timing, power and area model,” Tech. Rep. WRL Research Report 2001/2, Western Research Laboratory, Palo Alto, CA, Aug. 2001.

4. Doug Burger and Todd M. Austin, “The SimpleScalar tool set, version 2.0,” Tech. Rep. 1342, Department of Computer Science, University of Wisconsin in Madison, June 1997.

5. Tilman Wolf and Mark A. Franklin, “CommBench - a telecommunications benchmark for network processors,” in Proc. of IEEEInternational Symposium on Performance Analysis of Systems and Software (ISPASS), Austin, TX, Apr. 2000, pp. 154–162.

6. Di Caro G. Dorigo. M. AntNet: Distributed Stigmergetic Control for Communications Networks: Journal of Artifical Intelligence Research. – 1998. - №9 – pp. 317-365.

7. Ладыженский Ю.В., Мирецкая В.А. Исследование муравьиных алгоритмов маршрутизации в компьютерных сетях. ИНТЕРНЕТ-ОБРАЗОВАНИЕ-НАУКА, пятая международная конференция ИОН-2006. 10-14 октября, 2006. Сборник материалов конференции. Том 2. – Винница: УНИВЕРСУМ-Винница, 2006, стр. 375-377.

8. Грищенко В.И., Ладыженский Ю.В. Исследование архитектуры кэша сетевых процессоров // Радіоелектронні і комп’ютерні системи. – 2007. – №6 (25). С.186-191.

9. Anant Agarwal, “Performance tradeoffs in multithreaded processors,” IEEE Transactions on Parallel and Distributed Systems, vol. 3, no. 5, pp. 525–539, Sept. 1992.

10. N. Shah, "Understanding network processors," Tech. Rep. Version 1.0, EECS, University of California, Berkeley, September 2001.

11. Intel IXP1200 Network Processor Family. Microcode Programmer’s Reference Manual. Intel Corporation. March 2002.

Работа поступила в редакцию 13.06.2007.