2. *Dikarev N.I., Shabanov B.M.* Arkhitektura protsessora i ee vliyanie na proizvoditel'-nost' superEVM [Processor architecture and its influence on the performance of supercomputers], Programmnye produkty i sistemy [Software products and systems], 2007, No. 2 (78), pp. 2-5.
3. *Arvind and R.S.Nikhil.* Executing a program on the MIT tagged-token data-flow architecture, *IEEE Trans. Comput*., 1990, Vol. C-39, No. 3, pp. 300-318.
4. Manchester Dataflow Research Project. Available at: http://intranet.cs.man.ac.uk/cnc/progects/dataflow.html.
5. *Sakai S. et al*. An Architecture of a Dataflow Single-Chip Processor, *Proc. 16-th Ann. Symp. on Computer Architecture,* 1989, pp. 252-273.
6. *Papadopoulos G.V., Traub K.R.* Multithreading: A revisionist view of dataflow architectures, *Proc. 18-th Ann. Symp. on Computer Architecture,* 1991, pp. 342-351.
7. *Culler D.E. and Arvind*. Resource requirements of dataflow programs, *Proc. 15-th Ann. Symp. on Computer Architecture,* 1988, pp. 141-150.
8. The WaveScalar Architecture. Available at: http://wavescalar.sc.washington.edu.
9. TRIPS Processor Architecture. Available at: http://www.cs.utexas.edu/users/cart/trips.
10. *Hake J.F. and Homberg W*. The impact of memory organization on the performance of matrix calculations, *Parallel Computing,* 1991, Vol. 17, No. 2/3, pp. 311-327.

Статью рекомендовал к опубликованию д.т.н., профессор И.И. Левин.

Дикарев Николай Иванович – Межведомственный суперкомпьютерный центр РАН (МСЦ РАН); e-mail: nic@jscc.ru; 119991, Москва, Ленинский пр., 32а; тел.: 84959386144; зав. лабораторией; к.т.н.

Шабанов Борис Михайлович – e-mail: shabanov@jscc.ru; тел.: 84951373361; зам. директора; к.т.н.; доцент.

Шмелев Александр Сергеевич – e-mail: guest8993@rambler.ru; тел.: 84959386144; н.с.

Dikarev Nikolay Ivanovich – Joint Supercomputer Center of the Russian Academy of Sciences (JSCC RAS); e-mail: nic@jscc.ru; 32a, Leninsky Prospect, Moscow, Russia, 119991; phone: +74959386144; head of laboratory; cand. of eng. sc.

Shabanov Boris Mikhaylovich – e-mail: shabanov@jscc.ru; phone: +74951373361; deputy director; cand. of eng. sc.; associate professor.

Shmelev Aleksandr Sergeevich – e-mail: guest8993@rambler.ru; phone: +74959386144; researcher.

УДК 004.272.43

А.В. Колодзей

КОМПРОМИСС «ВРЕМЯ/ПАМЯТЬ» В РЕКОНФИГУРИРУЕМЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМАХ

В настоящее время является актуальной задача создания проблемно-ориентированных вычислительных систем (ПОВС), то есть предназначенных для решения конкретных классов задач. Одним из таких классов задач является обращение функций при помощи проведенных заранее предварительных вычислений. Решается задача достижения баланса между вычислительной производительностью и пропускной способностью системы памяти при использовании метода так называемых радужных таблиц. Даются оценки времени решения и необходимого объема памяти в зависимости от сложности задачи и конфигурации вычислительной системы. Полученные оценки позволяют выбрать оптимальное значение одного их основных параметров метода радужных цепочек – длины цепочки L в зависимости от текущей конфигурации вычислительной системы. Рост сложности решаемых задач может требовать модернизации ПОВС. Оказывается, если модернизацию планируется вести «экстенсивным» образом, когда относительный рост числа вычислительных

элементов (ВЭ) опережает относительный рост производительности отдельных ВЭ, то при росте общей интенсивности запросов к памяти, их удельная интенсивность снижается. При «интенсивном» развитии ПОВС, когда относительный рост производительности отдельных ВЭ опережает относительный рост числа ВЭ, растут и требования к удельной производительности системы памяти.

Проблемно-ориентированная вычислительная система; компромисс время-память; радужные таблицы; скорость операций ввода-вывода.

A.V. Kolodzey

TIME-MEMORY TRADE-OFF ON RECONFIGURABLE COMPUTER SYSTEMS

Now is the urgent task of creating a problem-oriented computing system (POCS) that is intended to solve specific classes of problems. One such class of problems is the reversing cryptographic hash functions by using preliminary calculations. The article solves the problem of achieving a balance between computational performance and throughput of the memory system when using the so-called rainbow tables. Provides assessment solution time and the required amount of memory depending on the complexity and configuration of the computing system. One of their main technique parameters rainbow chains is chain length L. The estimates obtained allow us to choose the optimal value of L depending on the current configuration of the computing system. Growth of complexity of the solved tasks can demand modernization of POVS. It appears if modernization it is planned to conduct in a "extensive" way when the relative growth of number of computing units (CU) advances the relative growth of productivity of separate CU, with a growth of the general intensity of inquiries to memory, their specific intensity decreases. At "intensive" development of POVS when the relative growth of productivity of separate CU advances the relative growth of number of CU, also requirements to the specific productivity of system of memory grow.

Problem-oriented computing; Time-Memory Trade-Off; Rainbow Tables; IOPS.

При создании высокопроизводительных вычислительных систем наряду с универсальными суперкомпьютерными системами, предназначенными для решения широкого класса задач, является актуальной задача создания проблемно-ориентированных вычислительных систем (ПОВС). ПОВС предназначены для решения конкретных классов задач и должны удовлетворять задаваемым требованиям времени решения задач, а также задаваемым конструктивным и технико-экономическим ограничениям, таким как максимальная потребляемая электрическая мощность, занимаемый объем, стоимость и пр. Одним из наиболее экономически эффективных способов построения высокопроизводительных проблемно-ориентированных вычислительных систем (ПОВС) в настоящее время является использование реконфигурируемых мультиконвейерных вычислителей. Общая концепция создания проблемно-ориентированных реконфигурируемых мультиконвейерных вычислительных систем представлена в [1] и целом ряде работ тех же авторов.

Будем рассматривать задачи, эффективно разбивающиеся на подзадачи, временная сложность которых может быть значительно снижена путем сохранения уже вычисленных решений подзадач. Одним из примеров классов таких задач является задача обращения функций. Пусть F – некоторое отображение конечного множества X в множество Y, и для $y_1, y_2, \ldots$ из Y требуется найти такие $x_1, x_2, \ldots$ из X, что

$$F(x_i) = y_i,\ i = 1, 2, \ldots \qquad (1)$$

Если F(x) не относится ни к какому специальному классу (линейных, монотонных и других) функций, для которых существуют эффективные методы решения уравнений (1), то, вообще говоря, остается один наиболее универсальный метод – полный перебор x из X. Такой перебор требует в наихудшем случае |X| и в среднем ½|X| вычислений функции F(x), если x равномерно распределен на множестве X. В дальнейшем под сложностью задачи будем понимать объем множества X.

Если задачу (1) нужно решать неоднократно для разных значений y, то при использовании метода полного перебора каждый раз придется выполнять одни и те же вычисления. В [2] был предложен универсальный метод, позволяющий за счет сохранения в памяти результатов предварительной работы осуществлять дальнейших поиск (оперативный этап) за меньшее время, но уже с некоторой вероятностью. Дальнейшее развитие метод компромисса время/память получил в [3], где был предложен метод так называемых радужных таблиц.

В настоящей статье рассматривается задача создания ПОВС, предназначенных для поиска решений уравнения (1) с использованием заранее рассчитанных радужных таблиц. Даются оценки, связывающие вычислительную производительность, объем и производительность системы памяти ПОВС, сложность задачи и время ее решения. Предполагается, что предварительная работа, которая требует большого объема вычислений, может быть возложена на универсальную суперкомпьютерную систему. В [5] указывается, что опубликованные ранее анализы метода радужных таблиц [3], [4], [6], [7], [8] были проведены для модели с памятью с произвольным доступом (RAM model). В данной модели предполагается, что есть центральный процессорный модуль (CPU) и модуль памяти с произвольным доступом неограниченной емкости. В [5] оцениваются среднее время поиска и вероятность успеха в модели с внешней памятью (External Memory Model). Эта модель предполагает, что есть процессор (CPU) и два уровня памяти: быстрая память ограниченной емкости и медленная память неограниченной ёмкости. В отличие от предыдущих авторов, будем рассматривать кластерную модель с распределенной памятью (Distributed Memory Model). В этой модели вычислительные узлы (ВУ) объединены при помощи некоторого интерконнекта. Каждый ВУ имеет свою оперативную память и набор относительно медленных накопителей, например на твердотельных жестких дисках (SSD).

Напомним основные идеи предложенного в [3] метода радужных таблиц. Выбираются параметры $L>0$ – длина цепочки, w>0 – число цепочек, набор функций редукции F_i, $i=1,\ldots,L$, отображающих множество Y в X, и множество S начальных состояний мощности w. На предварительном этапе производится вычисление концов всех цепочек

$$s_0 = s \rightarrow s_1 = F_1(F(s_1)) \rightarrow s_2 = F_2(F(s_1)) \rightarrow \ldots \rightarrow s_L = F_L(F(s_{L-1}))$$

для всех s из S. В таблицу записываются первые и последние элементы цепочек. Место l, необходимое для хранения одной цепочки, вообще говоря, зависит от сложности задачи, числа цепочек в таблице и составляет на практике обычно от 8 до 16 байт. Если множество S выбрано так, что концы всех цепочек различны, то таблица называется совершенной (perfect rainbow table).

Если требуемый x такой, что $F(x)=y$ лежит в некоторой цепочке на позиции $L–k$, то, применяя k раз отображение F и соответствующую функцию редукции, мы попадем в конец этой цепочки. Обратившись к таблице, мы найдем начало этой цепочки. Проходя $L–k$ шагов от начала этой цепочки, мы найдем этот x. Так как при поиске мы не знаем, на каком месте в цепочке лежит x, то мы, перебирая возможные значение k от 1 до L, получаем L возможных концов цепочек. Для тех из них, что найдены в таблице, идем от их начала на $L–k$ шагов. При этом возможно возникновение ложных тревог (false alarm), когда очередное вычисленное значение случайно совпадает с концом чужой цепочки.

Вероятность найти правильное решение с использованием одной совершенной радужной таблицы, как указывается в [4], есть

$$\mathbf{P} = 1 - (1 - w/(|X|)^L. \quad (2)$$

В наихудшем случае придется провести 1 + 2 + …+ L = ½L(L+1) вычислений шаговых функций $F_к(F(x))$, осуществить L обращений к таблице и провести обработку всех ложных тревог. Точную формулу для математического ожидания числа операций с учетом вероятности ложных тревог для совершенных радужных таблиц можно найти в [4].

Предположим, что ПОВС, предназначенная для оперативного этапа поиска по заранее построенным радужным таблицам, имеет следующие характеристики:

- ПОВС состоит из *k* ВУ;
- каждый ВУ имеет размер оперативной памяти *M* и n SSD-накопителей, каждый емкостью *S*;
- каждый SDD-накопитель обеспечивает *i* операций чтения с произвольным доступом в секунду (IOPS);
- каждый ВУ содержит *r* вычислительных элементов (ВЭ);
- каждый ВЭ обеспечивает вычисление *p* шаговых функций в секунду.

Для простоты будем считать, что производительность SDD-накопителей при параллельной работе внутри одного ВУ складывается и интерконнект имеет достаточную скорость. Оценки снизу для этой достаточной скорости интерконнекта также будут даны ниже.

Предположим, что время поиска одного решения задачи со сложностью $K=|X|$ не должно превышать *T* секунд при вероятности успеха не ниже *P*.

Пусть L – выбранная длина цепочки. Тогда из (2) следует, что

$$L\ ln\ (\ 1 - w/K) = \ln(1 - P)$$

и для необходимого объема радужной таблицы *V*, где $V = wl$, справедлива оценка

$$V \geq\ - ln(1 - P)\ lK/L. \qquad (3)$$

Здесь мы воспользовались приближением $ln(1 + x) = x$.

Среднее число *ENF* вычислений шаговой функции за время поиска можно представить в следующем виде:

$$ENF\ = \sum_{k=1}^{L}(k + \ q_k w_k(L - k)),$$

где q_k – вероятность ложной тревоги после раскручивания цепочки на длину k; w_k – число ложных тревог. Для совершенных таблиц w_k может принимать только значения 0 и 1 и $q_k w_k = \ q_k$. Точные формулы для q_k в случае совершенных таблиц можно найти в [4]. Количество вычислений *NF* шаговой функции на этапе поиска будет случайной величиной. Для совершенной радужной таблицы

$$½L(L+1) \leq\ NF\ \leq L(L+1).$$

В дальнейшем будем считать, что $NF = P_f L^2$, где P_f – некоторая ограниченная положительная величина, зависящая от конкретной таблицы.

Так как вычисление цепочек и поиск в таблице уже вычисленных концов цепочек можно вести параллельно, то для времени поиска T будет справедливо неравенство

$$\mathrm{T} \geq \max\ \left\{\frac{P_f\ L^2}{krp}, \frac{L}{I}\right\}, \qquad (4)$$

где I – суммарная пропускная способность памяти, в которой расположена радужная таблица.

Здесь мы считаем, что суммарная производительность ВЭ определяется произведением их числа на производительность каждого из них. Для достижения этого на практике, конечно, придется приложить ряд усилий. В частности, придется учитывать, что трудоемкость обработки цепочки и вероятность ложной тревоги зависят от длины этой цепочки. Аналогично оценивается и производительность системы памяти. Если радужная таблица помещается в оперативную систему одного ВУ, то I можно полагать произведением числа запросов в секунду к опера-

тивной памяти на число ВУ. Если таблица не помещается в памяти, то опять, в зависимости от ее размера, на каждом узле может быть своя копия таблицы или таблица распределена между различными дисками различных узлов. В случае распределенной таблицы запросы на поиск в таблице должны пересылаться от ВУ к тому ВУ, где находится нужный фрагмент таблицы. Считая интерконнект достаточно быстрым по сравнению с обращениями к SSD, мы этими накладными расходами в нашей модели пренебрегаем. При расположении таблиц на SSD-накопителях считаем I = nik.

Из неравенств (3), (4) для объема таблицы *V* получаются такие оценки:

$$V \geq \max \{V_1, V_2\}, \quad (5)$$

где $V_1 = -\ln(1-P)lK\sqrt{\frac{P_f}{Tkrp}}$, $V_2 = -\frac{\ln(1-P)lK}{TI}$.

При этом интенсивность обращения к таблице будет равняться $\frac{L}{T}$ и

$$-\frac{\ln(1-P)lK}{VT} \leq \frac{L}{T} \leq \sqrt{\frac{krp}{TP_f}}. \quad (6)$$

Общий объем передаваемых данных во время счета – случайная величина, зависящая от вероятности ложных тревог. Для совершенных таблиц ее можно оценить сверху величиной 2lL Соответственно оценкой для минимально необходимой скорости интерконнекта будет величина $\frac{2lL}{T}$.

Анализ получившихся оценок позволяет обосновать требования к характеристикам ПОВС в зависимости от требуемых вероятности успеха и времени счета и сделать некоторые выводы.

ПОВС сбалансирован на данной задаче, если $V_1 = V_2$.

Величина V_1 определяется вычислительной производительностью, которая определяется произведением числа ВЭ на их скорость. Увеличение производительности при использовании одной и той же таблицы будет пропорционально приводить к уменьшению времени счета до тех пор, пока величина V_1 не станет меньше V_2.

Увеличение производительности позволяет при том же времени счета уменьшить необходимый размер таблицы. Здесь зависимость нелинейная. Увеличение производительности в 4 раза позволяет уменьшить размер таблицы в 2 раза. Но при этом возрастет в 2 раза и интенсивность запросов к таблице. Поэтому этот путь оправдан только в том случае, если таблицу в результате удается разместить в оперативной памяти одного ВУ или распределить по оперативной памяти нескольких ВУ.

Увеличение размера таблицы при фиксированной производительности приводит к квадратичному уменьшению времени счета, опять же пока величина V_1 не станет меньше V_2. Однако при этом возрастает интенсивность запросов к таблице. Если критичной характеристикой является именно производительность системы памяти, то можно при увеличении размера таблицы пропорционально уменьшать вычислительную производительность. В этом случае при уменьшении времени счета будет сохраняться интенсивность запросов к памяти.

Таким образом, кроме собственно вычислительной производительности значимыми характеристиками ПОВС для решения рассмотренного класса задач являются объем и производительность системы памяти на операциях случайного чтения.

Типовые значения скорости произвольного доступа IOPS при случайном чтении 4 Кб блоков приведены в табл. 1 по материалам из [9]. Использование RAID-контроллеров позволяет увеличивать общее количество IOPS до тех пор, пока определяющими не станут ограничения самого контроллера RAID и интерфейса доступа к хранилищу данных.

Таблица 1

Значения IOPS для различных типов накопителей

	Устройство	IOPS
1	HDD 7200 RPMSATA	100
2	HDD 15KRPMSAS	180
3	SSD Intel X25 SATA II	5000–8000
4	SSD OCZ SATAIII	до 100 000
5	RAMDISK, PCIe SSD	до 25 0000

Следует отметить также, что судя по результатам тестовых программ, таких как fio [10], наибольшую производительность SSD-диски достигают с ростом параметра «глубина очереди» (Queue Depth, QD) примерно до 32–64. Queue Depth контролирует максимальное количество команд ввода-вывода, которые могут одновременно находиться в исходящих очередях портов. Слишком маленькое значение глубины очереди не позволяет раскрыть весь потенциал адаптера, слишком большая QD перегружает входящие буферы портов дискового массива.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Каляев И.А., Левин И.И., Семерников Е.А., Шмойлов В.И.* Реконфигурируемые мультиконвейерные вычислительные структуры. – Ростов-на-Дону: Изд-во ЮНЦ РАН, 2008. – 320 с.
2. *Hellman M.E.* A cryptanalytic time-memory trade-off // IEEE Transactions on Information Theory. – 1980. – № 26(4). – P. 401-406.
3. *Philippe Oechslin.* Making a Faster Cryptoanalytic Time-Memory Trade-Off. In Dan Boneh, editor, Advanced in Cryptology – CRYPTO'03, vol. 2729 of Lecture Notes in Computer Science, Santa Barbara, California, USA, August 2003. IACR, Springer-Verlag. – P. 617-630.
4. *Gildas Avoine, Pascal Junod, Philippe Oechslin:* Characterization and Improvement of Time-Memory Trade-Off Based on Perfect Tables. ACM Trans. Inf. Syst. Secur. – 2008. – No. 11 (4).
5. *Jung Woo Kim, Jin Hong, Kunsoo Park.* Analysis of the Rainbow Tradeoff Algorithm Used in Practice – JACR Cryptology ePrint Archive, 2013. URL: http://eprint.iacr.org/2013/591.pdf (дата обращения: 31.10.14).
6. *Hong J.* The cost of false alarms in Hellman and rainbow tradeoffs, Des. Codes Cryptography. – Dec. 2010. – Vol. 57. – P. 293-327.
7. *Hong J. and Moon S.* A comparison of cryptanalytic tradeoff algorithms // J. Cryptology. – 2013. – Vol. 26. – P. 559-637.
8. *Lee G.W., Hong J.* A comparison of perfect table cryptanalytic tradeoff algorithms, Cryptology ePrint Archive, Report 2012/540. URL: http://eprint.iacr.org/2012/540 (дата обращения: 31.10.14).
9. Как правильно мерять производительность диска. URL: www.habrahabr.ru/post/154235/ (дата обращения: 31.10.14).
10. Project fio. URL: http://freecode.com/projects/fio (дата обращения: 31.10.14).

REFERENCES

1. *Kalyaev I.A., Levin I.I., Semernikov E.A., Shmoylov V.I.* Rekonfiguriruemye mul'tikonveyernye vychislitel'nye struktury [Reconfigurable multiconference computational patterns]. Rostov-on-Don: Izd-vo YuNTs RAN, 2008, 320 p.
2. *Hellman M.E.* A cryptanalytic time-memory trade-off, *IEEE Transactions on Information Theory*, 1980, No. 26(4), pp. 401-406.
3. *Philippe Oechslin.* Making a Faster Cryptoanalytic Time-Memory Trade-Off. In Dan Boneh, editor, Advanced in Cryptology – CRYPTO'03, vol. 2729 of Lecture Notes in Computer Science, Santa Barbara, California, USA, August 2003. IACR, Springer-Verlag, pp. 617-630.
4. *Gildas Avoine, Pascal Junod, Philippe Oechslin:* Characterization and Improvement of Time-Memory Trade-Off Based on Perfect Tables. ACM Trans. Inf. Syst. Secur., 2008, No. 11 (4).

5. *Jung Woo Kim, Jin Hong, Kunsoo Park.* Analysis of the Rainbow Tradeoff Algorithm Used in Practice – JACR Cryptology ePrint Archive, 2013. Available at: http://eprint.iacr.org/2013/591.pdf (Accessed: 31 October 2014).
6. *Hong J.* The cost of false alarms in Hellman and rainbow tradeoffs, Des. Codes Cryptography, Dec. 2010, Vol. 57, pp. 293-327.
7. *Hong J. and Moon S.* A comparison of cryptanalytic tradeoff algorithms, *J. Cryptology*, 2013, Vol. 26, pp. 559-637.
8. *Lee G.W., Hong J.* A comparison of perfect table cryptanalytic tradeoff algorithms, Cryptology ePrint Archive, Report 2012/540. Available at: http://eprint.iacr.org/2012/540 (Accessed: 31 October 2014).
9. Как правильно мерять производительность диска. Available at: www.habrahabr.ru/post/154235/ (Accessed: 31 October 2014).
10. Project fio. Available at: http://freecode.com/projects/fio (Accessed: 31 October 2014).

Статью рекомендовал к опубликовнию д.т.н., профессор В.Ф. Гузик.

Колодзей Александр Владимирович – Научно-технический центр закрытого акционерного общества «ИнформИнвестГрупп»; e-mail: ak@iigroup.ru; 117587, г. Москва, Варшавское шоссе, 125, стр. 17; тел.: +74952870035; зам. директора по научной работе.

Kolodzey Alexander Vladimirovich – InformInvest Group, CJSC, Scientific and Technical Center; e-mail: ak@iigroup.ru; 125, Varshavskoe shosse build. 17, Moscow, 117587, Russia; phone: +74952870035; deputy director on scientific work.

УДК 004.2; 004.38; 621.38

Н.А. Лукин

ФУНКЦИОНАЛЬНО-ОРИЕНТИРОВАННЫЕ ПРОЦЕССОРЫ – КЛЮЧЕВЫЕ КОМПОНЕНТЫ ВСТРОЕННЫХ СУПЕРКОМПЬЮТЕРОВ ДЛЯ СИСТЕМ РЕАЛЬНОГО ВРЕМЕНИ*

Среди возможных разновидностей суперкомпьютеров можно выделить такие, которые встроены в состав систем реального времени. Оптимизация архитектур таких суперкомпьютеров по производительности и аппаратным затратам весьма важна с точки зрения повышения эффективности их применения. В работе рассматривается архитектурное направление обеспечения высокой производительности суперкомпьютеров, которое реализуется с помощью введения в их состав функционально-ориентированных процессоров (ФОП). Описываются результаты исследований и разработок конкретных типов таких процессоров, которые могут быть взяты за основу при создании СБИС ФОП. В частности, приведены технические характеристики следующих видов ФОП: таблично-алгоритмических процессоры для быстрого вычисления математических функций с гарантированной точностью; параллельные ФОП с архитектурой SIMD для реализации алгоритмов корреляционно-экстремальной навигации на борту летательного аппарата; двумерные SIMD- и MIMD-массивы, состоящие из битовых процессорных элементов для обработки изображений в реальном времени.

Функционально-ориентированные процессоры; специализированный параллелизм; СБИС; сложность вычислений; бортовые системы управления; однородные вычислительные системы.

* Работа выполнена частично за счет программы Президиума РАН № 18 "Алгоритмы и математическое обеспечение для вычислительных систем сверхвысокой производительности" при поддержке УрО РАН (проект 12-П-1-1028).