УДК 004.318

Алгоритмы сжатия данных в кэш-памяти микропроцессоров

А.С. Кожин
Московский физико-технический институт (государственный университет)
АО «МЦСТ»

Современные микропроцессоры имеют многоуровневую иерархию кэш-памяти, которая позволяет повысить пропускную способность и уменьшить среднее время доступа к данным, обладающим пространственной и временной локальностью. Среднее время доступа зависит от коэффициента попаданий в кэш-память и от времени доступа в кэш-память и оперативную память. Увеличение объема кэш-памяти позволяет повысить коэффициент попаданий, хотя может увеличить ее время доступа. В современных микропроцессорах суммарный объем кэш-памяти может достигать десятков и даже сотни мегабайт. При этом основными факторами, ограничивающими максимальный объем кэш-памяти, являются площадь кристалла и рассеиваемая мощность.

Сжатие данных может повысить эффективный объем хранимой информации. В современных вычислительных системах есть примеры использования сжатия в оперативной памяти [1], однако сжатие данных в кэш-памяти до сих пор не применяется в серийных микропроцессорах, хотя исследования на эту тему ведутся достаточно давно [2]. Основные трудности практического применения алгоритмов сжатия в кэш-памяти связаны с необходимостью изменить устоявшуюся структуру кэш-памяти, а также с достаточно большими накладными расходами и увеличением времени доступа при сложной декомпрессии.

Большинство алгоритмов реализуют сжатие отдельных блоков – кэш-строк – и размещают большее количество кэш-строк при увеличении (обычно при удвоении) числа хранимых адресных тэгов. Сложные алгоритмы обеспечивают высокую степень сжатия, до двух раз увеличивая эффективный объем хранимой в кэш-памяти информации на задачах с подходящей структурой данных. В то же время сложные алгоритмы требуют громоздких вычислений при декомпрессии и существенно увеличивают время доступа в кэш-память, поэтому могут замедлить выполнение задач, которые не относятся к категории “Cache Friendly” [3], и задач с низкой степенью сжатия рабочих данных. В качестве примеров выбраны алгоритмы FPC [4] и C-Pack [5]. В основе алгоритма FPC лежат разбиение кэш-строки на фиксированные сегменты и проверка этих сегментов по распространенным шаблонам. Более эффективный алгоритм C-Pack дополняет сжатие по статическим шаблонам динамическим справочником. В обоих алгоритмах декомпрессия выполняется последовательно, поэтому имеет достаточно большую задержку (5-8 тактов на частоте 2 ГГц и технологии 28 нм).

Более простые для реализации алгоритмы имеют меньшую степень сжатия, но при этом не увеличивают время доступа в кэш-память благодаря быстрой декомпрессии. Эти алгоритмы больше подходят для реализации в серийных микропроцессорах, так как требуют меньших ресурсов и могут обеспечить работу без ухудшения производительности на любых задачах. Алгоритм ZCA [6] выделяет специальное расширение кэш-памяти для нулевых кэш-строк, в котором хранит только их адресные тэги и состояния. Такой подход не требует никакого дополнительного времени на декомпрессию, но обеспечивает невысокую степень сжатия. Алгоритм BDI [7] разбивает кэш-строку на одинаковые сегменты и проверяет, можно ли их представить как дельты меньшего размера относительно выбранного базового сегмента. Декомпрессия выполняется параллельно во всех сегментах и добавляет всего один такт ко времени доступа в кэш-память.

В работе проведены анализ и сравнение методов сжатия данных в кэш-памяти микропроцессоров. Показано, что метод BDI имеет наибольшую эффективность для практического применения. Представлены результаты исследований с использованием прототипа микропроцессора с архитектурой Эльбрус.

Литература

1. *Abali B., Franke H., Poff D.E., Saccone R.A., Schulz C.O., Herger L.M., Smith T.B.* Memory expansion technology (MXT): software support and performance // IBM Journal of Research and Development. 2001. V. 45, N 2. P. 287-301.
2. *Sardashti S., Arelakis A., Stenström P., Wood D.A.* A primer on compression in the memory hierarchy // Synthesis Lectures on Computer Architecture. 2015. V. 10, N 5. P. 1-86.
3. *Кожин А.С., Нейман-заде М.И., Тихорский В.В.* Влияние подсистемы памяти восьмиядерного

микропроцессора «Эльбрус-8С» на его производительность // Вопросы радиоэлектроники. 2017. № 3. С. 13-21.
4. *Alameldeen A.R., Wood D.A.* Adaptive cache compression for high-performance processors // Computer Architecture, 2004. Proceedings. 31st Annual International Symposium on. IEEE, 2004. P. 212-223.
5. *Chen X., Yang L., Dick R.P., Shang L., Lekatsas H.* C-pack: A high-performance microprocessor cache compression algorithm // IEEE transactions on very large scale integration (VLSI) systems. 2010. V. 18, N 8. P. 1196-1208.
6. *Dusser J., Piquet T., Seznec A.* Zero-content augmented caches // Proceedings of the 23rd international conference on Supercomputing. ACM, 2009. P. 46-55.
7. *Pekhimenko G., Seshadri V., Mutlu O., Gibbons P.B., Kozuch M.A., Mowry T.C.* Base-delta-immediate compression: Practical data compression for on-chip caches // Proceedings of the 21st international conference on Parallel architectures and compilation techniques. ACM, 2012. P. 377-388.