УДК: 681.3

АНАЛИЗ ВЛИЯНИЯ ПРЕДВАРИТЕЛЬНОЙ ВЫБОРКИ В КЭШ-ПАМЯТЬ НА ПРОИЗВОДИТЕЛЬНОСТЬ МИКРОПРОЦЕССОРА

Б.З. Шмейлин, старший научный сотрудник

Федеральный исследовательский центр «Информатика и управление» РАН,
г. Москва, 119333 Россия
Автор для переписки, e-mail: shmeilin@mail.ru

В последние годы все большее влияние на производительность микропроцессорных систем оказывает задержка при обращении в память. Применение встроенной в кристалл кэш-памяти существенно снижает эту задержку. Одним из основных методов достижения эффективного использования кэш-памяти является предварительная выборка данных в кэш. Настоящая работа посвящена анализу влияния одного из видов предварительной выборки, а именно Марковской предварительной выборки, на производительность микропроцессорной системы в целом. Результаты анализа позволили выработать рекомендации по выбору значений параметров марковского предвыборщика для приложений с интенсивными вычислениями и приложений с интенсивным использованием данных в памяти.

Ключевые слова: микропроцессор, производительность микропроцессора, кэш-память, предварительная выборка в кэш, Марковский предвыборщик.

THE IMPACT ANALYSIS OF PREFETCH IN THE CACHE ON THE MICROPROCESSOR PERFORMANCE

B.Z. Shmeylin

Federal Research Center «Computer Science and Control» of Russian Academy of Sciences,
Moscow, 119333 Russia
Corresponding author e-mail: shmeilin@mail.ru

Memory access delay has been a major influence on microprocessor systems performance recently. On-chip cache memory application dramatically reduces this delay. Cache prefetching method is one of the basic ones to give the most effective use of memory cache. In the present study we analyze the effect of one of the types of prefetching, especially, the Markov one, on the whole microprocessor-based system performance. The results obtained make it possible to come up with recommendations for the choice of Markov prefetcher parameters setting for both computationally intensive task and data-rich applications.

Keywords: microprocessor, microprocessor performance, cache memory, prefetching in cache, the Markov prefetcher.

Введение

Рост производительности современных микропроцессорных систем (МС) ограничивается задержками, обусловленными обращениями к памяти. Усилия разработчиков МС нацеле-

ны на снижение этого времени. Решение данной проблемы достигается преимущественно повышением эффективности использования встроенной в кристалл кэш-памяти. Эффективность оценивается по относительному числу промахов в кэше (*miss ratio*), вычисляемому как отношение числа промахов к общему числу обращений процессора к памяти. При возникновении промаха, то есть при отсутствии запрашиваемых процессором данных в кэше, запрос передается в кэш второго уровня или в главную память. Это приводит к значительной потере производительности микропроцессорной системы, так как время обращения к памяти при промахе в кэше увеличивается, по крайней мере, на порядок.

Одним из главных методов снижения относительного числа промахов в кэше является предварительная выборка данных в кэш-память, реализуемая специальным устройством предвыборки – предвыборщиком. Предвыборщик на основании анализа адресов обращений к памяти процессора выбирает данные из главной памяти и размещает их в кэше до того момента, когда за ними обратится процессор. Чтобы не препятствовать работе процессора, предварительная выборка выполняется в тактах, свободных от обращений процессора. Предварительная выборка основывается на том или ином способе предсказания промаха в кэше, который может произойти в скором времени. Таким образом, предвыборщик снижает число промахов при обращении процессора в кэш.

Качество предвыборки характеризуется точностью предвыборки и степенью покрытия предвыборки. Точность предвыборки определяется отношением количества предвыбранных данных к действительно используемым из них процессором; степень покрытия – отношением количества правильно предсказанных данных к числу обращений к памяти. Указанные показатели используются для сравнения эффективности различных алгоритмов предвыборки. Повышение значения степени покрытия достигается увеличением числа правильно предсказанных промахов. Оно может быть достигнуто путем увеличения общего числа предсказаний промахов. Однако при этом не исключен рост избыточных предсказаний, то есть предсказаний адресов обращений к кэшу, не используемых процессором, а также вытеснение нужных строк кэша, так называемое «загрязнение» кэша, и снижение точности предвыборки.

Следует отметить также, что повышение показателей качества предвыборки не позволяет определить, в какой степени повышается производительность МС и повышается ли она вообще при использовании предварительной выборки. Хотя степень покрытия и позволяет определить, насколько снижается относительное число промахов в кэш (*miss ratio*), но снижение *miss ratio* может не только не повысить производительность МС, а иногда даже и понизить ее.

Из вышеизложенного становится ясно, что предвыборка должна быть спроектирована так, чтобы в результате ее применения общая производительность системы повышалась, а не деградировала.

Цель настоящей работы – оценка влияния предварительной выборки данных в кэш на производительность МС в целом.

1. Предварительная выборка данных в кэш

Обычно выборка из памяти выполняется по запросу процессора. Если запрашиваемые данные отсутствуют в кэше, то запрос адресуется к внешней памяти, то есть происходит промах в кэш, что ведет к потере производительности микропроцессорной системы. Задача предвыборки устранить такие промахи. Устройство предвыборки, или предвыборщик, анализируя последовательность адресов обращения к памяти, при которых происходят промахи, заранее выбирает по этим адресам данные и передает их в кэш. Чем больше доля правильно предвыбранных данных, тем эффективней работа предвыборщика.

Существует несколько методов предварительной выборки:

- потоковая, или последовательная, предвыборка (*stream prefetching*), самая простая, когда в качестве кандидата на предвыборку выбирается следующий по адресу блок данных;
- пошаговая (*stride prefetching*), при которой в результате анализа работы приложения

обнаруживаются циклические обращения к элементам данных с определенным шагом; при пошаговой предвыборке в качестве кандидата на предвыборку выбирается блок данных, отстоящий по адресу на определенный шаг.

При работе таких предвыборщиков для исключения предполагаемого промаха заранее, до обращения в кэш процессора, предвыборщиком в кэш помещается строка, адрес которой определен по тому или иному алгоритму.

Указанные методы позволяют повысить эффективность кэша при работе со структурированными приложениями, то есть с приложениями, которые характеризуются циклическим обращением к данным.

В то же время существует большое количество неструктурированных приложений, для которых весьма затруднительно предсказать адрес предполагаемого промаха. Для неструктурированных прикладных программ наиболее перспективен метод предварительной выборки, основанный на Марковской модели промахов в кэш, при котором в качестве источника информации для предсказания следующего промаха используется поток случайных адресов промахов в кэше, аппроксимированный Марковской моделью промахов [1–3].

При разработке марковского предвыборщика предполагается, что в одной и той же среде сохраняется случайная последовательность адресов промахов [1]. То есть, если один раз обнаружено, что после промаха по адресу *A* происходит промах по адресу *B*, либо по адресу *C*, то весьма вероятно, что одна из таких последовательностей сохранится и при повторном запуске приложения. Отсюда следует, чтобы избежать промаха по адресу *B* или *C*, необходимо после промаха по адресу *A* выполнить предварительную выборку по адресам *B* и *C*. По такому принципу строится Марковский предвыборщик. Далее рассмотрим конфигурацию и работу Марковского предвыборщика.

2. Марковский предвыборщик

При возникновении промаха в кэше первого уровня запрос передается в кэш второго уровня, а при промахе в нем – в основную память. Процессор вынужден приостановить свою работу до того момента, когда запрошенные данные могут быть выбраны из кэша. Однако он может выполнять работу по предварительной выборке с помощью программы предвыборки.

Во время работы такой программы формируется таблица адресов промахов в кэше (*Miss Addresses Table* – MAT) [3], запись в которую выполняется при каждом промахе в кэше. В каждой строке MAT содержится одна из последовательностей адресов промахов. Первым элементом строки, называемым индексом, является адрес начала последовательности, затем располагаются последующие адреса последовательности в порядке убывания вероятностей переходов к этим адресам. Вероятность переходов к последующим адресам определяется числом переходов из адреса промаха, являющегося индексом строки MAT, к последующим адресам промахов.

Последовательность случайных адресов промахов при работе приложения можно представить в виде ориентированного графа, пример которого представлен на рис. 1. Вершины графа представляют случайные адреса промахов, а дуги – отражают последовательность промахов. Каждая из дуг графа помечена вероятностью соответствующего перехода.

Пусть у нас получилась следующая последовательность адресов промахов:

A, B, C, D, C, F, A, C, F, F, E, A, A, B, C, D, E, A, B, C, D, E, A, B, C, D, C.

Предположим, что переход от промаха по адресу *A* к промаху по адресу *B* произошел 3 раза, к промаху по адресу *C* – 2 раза, а промаху по адресу *A* – 1 раз. Тогда вероятность перехода от промаха по адресу *A* к промаху по адресу *B*(p_{ab}) равна 0.5, к промаху по адресу *C*(p_{ac}) – 0.33 и к промаху по адресу *A*(p_{aa}) – 0.17.

Описанная выше модель промахов отражена в таблице адресов промахов MAT [3].

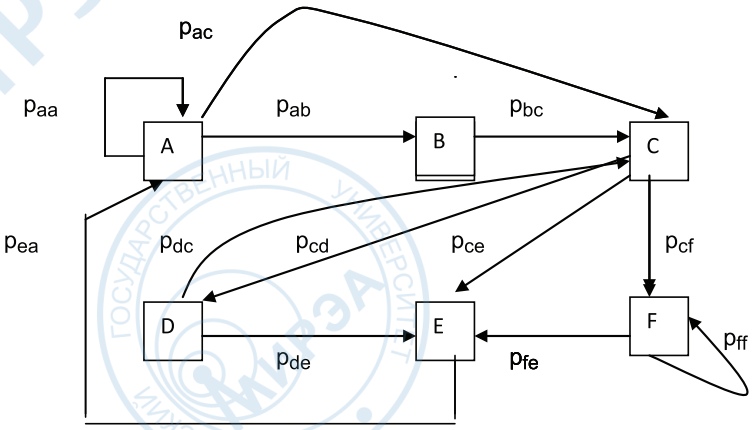

Рис. 1. Граф последовательности промахов.

В табл. 1 показан пример таблицы адресов промахов, соответствующий последовательности промахов, представленной на рис. 1.

Таблица 1. Пример таблицы адресов промахов

Адрес начала последовательности промахов	Последующие адреса промахов
A	B C A
B	C
C	D E F
D	C E
E	A
F	A F

При возникновении промаха в кэше проверяют, имеется ли среди индексов в MAT возникший адрес промаха. Если такой имеется, то диспетчер помещает в специальный буфер предвыборки (*prefetch buffer – PB*) [3] все имеющиеся в данной строке MAT адреса вместе с данными по этим адресам в качестве возможных последующих обращений процессора.

PB – это полностью ассоциативная память небольшого объема (8, 16 или 32 строки). Каждая строка содержит адреса и данные предвыборок. Замещение строк *PB* выполняется по алгоритму LRU – *least recently used* – наиболее давно используемая строка. Буфер предвыборки является по существу дополнением к кэшу. Возможна реализация Марковского предвыборщика, при которой диспетчер предвыборки размещает информацию непосредственно в кэше. Однако при этом будут вытесняться строки кэша, к которым в скором времени обратится процессор. Такое явление называется «загрязнением» кэша (см. введение). Поэтому, как правило, при реализации Марковского предвыборщика используется буфер предвыборки.

При работе приложения запрос процессора направляется одновременно в кэш и буфер предвыборки. Если в кэше или в буфере предвыборки содержится запрашиваемый адрес, то данные по нему передаются процессору. В противном случае происходит промах, и обращение передается в кэш следующего уровня или в главную память, как и при работе приложения без предвыборщика.

3. Влияние параметров Марковского предвыборщика на показатели качества предвыборки

В табл. 2 приведены данные по степени покрытия для разных приложений из набора SPEC2000 при различных размерах MAT при двух и четырех предсказаниях последующих промахов для каждого размера MAT. В табл. 3 суммированы данные по точности для разных приложений из набора SPEC2000 при различных размерах MAT и двух и четырех предсказаниях последующих промахов для каждого размера MAT [4].

Таблица 2. Значения степени покрытия предвыборки при различных размерах MAT [4]

Приложения	Размер MAT									
	1K		2K		4K		8K		16K	
	Число предсказаний последующих промахов									
	2	4	2	4	2	4	2	4	2	4
crafty	0.06	0.06	0.1	0.09	0.1	0.1	0.17	0.15	0.22	0.18
gcc	0.18	0.18	0.19	0.19	0.22	0.22	0.38	0.35	0.52	0.43
vpr	0.01	0.01	0.02	0.02	0.05	0.04	0.12	0.08	0.19	0.16
ammp	0.1	0.1	0.52	0.52	0.93	0.93	0.93	0.93	0.93	0.93
gzip	0.18	0.17	0.19	0.19	0.19	0.19	0.2	0.2	0.23	0.22
mcf	0.18	0.18	0.19	0.19	0.2	0.2	0.21	0.21	0.22	0.22

Таблица 3. Значения точности предвыборки при различных размерах MAT [4]

Приложения	Размер MAT									
	1K		2K		4K		8K		16K	
	Число предсказаний последующих промахов									
	2	4	2	4	2	4	2	4	2	4
crafty	0.8	0.8	0.68	0.67	0.57	0.56	0.4	0.36	0.24	0.22
gcc	0.82	0.82	0.8	0.8	0.7	0.68	0.6	0.58	0.47	0.44
vpr	0.83	0.83	0.72	0.72	0.62	0.62	0.38	0.36	0.22	0.17
ammp	0.76	0.76	0.76	0.76	0.76	0.76	0.76	0.76	0.76	0.76
gzip	0.92	0.92	0.91	0.91	0.86	0.87	0.63	0.62	0.53	0.52
mcf	0.97	0.97	0.96	0.96	0.96	0.95	0.6	0.59	0.58	0.58

В табл. 2, 3 отброшены результаты экспериментов, в которых использовали 4 и более предсказаний, так как ранее [1] установлено, что число последующих промахов редко превышает 4 промаха. В обеих таблицах приведены данные для размера буфера предвыборки в 32 строки.

На показатели качества предвыборки влияет также размер буфера предвыборки – *PB*. В работе [5] приведены данные по влиянию размера буфера *PB* на точность и степень покрытия предвыборки. Эти данные приведены в табл. 4.

По сравнению с ранее выбранным размером буфера в 32 строки при 16-ти строках точность и степень покрытия предвыборки снижается примерно на 10%.

Таблица 4. Влияние размера буфера предвыборки на показатели качества предвыборки [5]

Размер буфера предвыборки *PB* в строках	**Точность**	**Степень покрытия**
8	0.24	0.052
16	0.27	0.057
32	0.31	0.063

Из приведенных здесь данных видно, что для всех приложений значение степени покрытия повышается с увеличением размера MAT, вероятно, за счет того, что большее число строк MAT позволяет предсказать больше последующих промахов. Однако при увеличении размера MAT почти для всех приложений значение точности снижается, так как при увеличении числа предсказаний последующих промахов возрастает вероятность появления избыточных, то есть не используемых процессором, данных предвыборки.

Таким образом, нами установлено, как влияют параметры Марковского предвыборщика на показатели качества предвыборки. Однако неизвестно, как влияют эти показатели качества предвыборки на производительность МС в целом. Чтобы определить это влияние, нам необ-

ходимо знать, какие показатели быстродействия современной иерархической системы памяти оказывают наибольшее влияние на производительность МС, а затем установить зависимость этих показателей быстродействия от показателей качества предвыборки. Быстродействие памяти характеризует задержку при обращении процессора к памяти.

4. Показатели быстродействия памяти

Вопрос выбора показателей быстродействия памяти возникает по той причине, что оптимизация работы отдельного компонента иерархической памяти не обязательно приводит к повышению производительности микропроцессорной системы в целом. Вообще говоря, неизвестно, в какой мере влияет изменение того или иного параметра системы памяти на общую производительность МС.

Причины, по которым изменение параметров системы памяти не может характеризовать производительность системы в целом, следующие:

1. Современные процессоры используют различные методы параллельного выполнения инструкций, чтобы совместить арифметико-логическую операцию с обращением к памяти;
2. Изменение порядка выполнения инструкций с целью совместить выполнение инструкций с обращением к памяти позволяет скрыть потери времени при промахе в кэш;
3. Многопоточное исполнение инструкций позволяет скрыть даже большие потери времени при промахе в кэш второго уровня путем перехода на другой поток;
4. Спекулятивные механизмы позволяют преодолевать зависимости по данным и тем самым не приостанавливать работу конвейера.

Кроме того, современные системы памяти используют большое число технических средств уменьшения задержек при обращении к памяти. К таким средствам относятся неблокирующий (*non-blocking*) кэш, конвейеризованный (*pipelined*) кэш, кэш с несколькими банками (*multibanked cache*) и предвыборка (*prefetching*) данных. Применение указанных средств усложняют зависимость производительности МП в целом от быстродействия памяти. Таким образом, при обращении к памяти, организованном перечисленными выше способами, выполняются одновременно десятки и сотни операций. Расчеты быстродействия памяти, учитывающие только одиночное обращение к памяти или только работу отдельного ее компонента, не соответствуют действительному параллелизму работы системы памяти.

Выделяют несколько показателей быстродействия памяти, а именно – APC (*Access Per Cycle*) – число обращений к памяти на один такт, HR (*Hit Ratio*) – относительное число попаданий в кэш, HR1K (*Hit Ratio per 1K*) – относительное число попаданий в кэш на 1К инструкций, AMP (*Average Miss Penalty*) – средняя потеря времени от промаха в кэш, AMAT (*Average Memory Access Time) –* среднее время обращения к памяти [6]. При этом для оценки влияния различных параметров иерархической памяти на общую производительность системы предлагается использовать коэффициенты корреляции между различными показателями быстродействия памяти и показателем производительности МС – числом инструкций на такт (*Instructions Per Cycle* – IPC). В результате проведенных экспериментов установлено [6], что наибольшее абсолютное значение имеют коэффициенты корреляции между APC и IPC, то есть показатель быстродействия памяти APC наиболее пригоден для оценки влияния быстродействия памяти на общую производительность МС. Естественно, значение этого коэффициента корреляции положительное, то есть при увеличении значения APC возрастает значение IPC.

Из других показателей наиболее заметна корреляция между AMAT и IPC [6], поскольку, по всей видимости, показатель AMAT отражает и относительное число промахов в кэш, и потери от таких промахов. Значение коэффициента корреляции отрицательное, то есть при увеличении значения AMAT значение IPC уменьшается.

В дальнейшем будем рассматривать только показатели APC и AMAT для оценки влияния быстродействия памяти на производительность МС.

В табл. 5 и 6 представлены результаты экспериментов, для которых каждое приложение выполнялось каждый раз с изменением одного из параметров конфигурации иерархической

системы памяти. К таким параметрам относятся размеры кэшей первого и второго уровней, их ассоциативности, различные значения задержек при обращении к главной памяти. В табл. 5 приведены результаты вычислений с низкими абсолютными значениями коэффициента корреляции между APC и IPC и между AMAT и IPC.

Таблица 5. Приложения с низкими абсолютными значениями коэффициента корреляции между APC и IPC и между AMAT и IPC

Приложения	Коэффициент корреляции	
	APC	AMAT
Games	0.8	-0,3
gromecs	0.55	-0.36
calculix	0.5	-0.21
h264ref	0.78	-0.48
Tonto	0.76	-0.7
omnetpp	0.76	-0.43
sphinx3	0.28	-0.1

В табл.6 даны результаты вычислений с высокими абсолютными значениями коэффициента корреляции между APC и IPC и между AMAT и IPC.

Таблица 6. Приложения с высокими абсолютными значениями коэффициента корреляции между APC и IPC и между AMAT и IPC

Приложения	Коэффициент корреляции	
	APC	AMAT
Bzip	0.96	-0.94
bwawes	0.92	-0.93
Mcf	0.97	-0.97
Milk	0.98	-0.94
zeusmp	0.98	-0.98
leslie3d	0.98	-0.94
Lmb	0.98	-0.98
Astar	0.98	-0.94
castusADM	0.98	-0.92
Dealll	0.76	-0.82
Tonto	0.76	-0.7

Из приведенных в табл. 5, 6 данных видно, что для различных приложений коэффициенты корреляции имеют большой разброс значений: от 0.5 до 0.98 для APC и от 0.1 до 0.98 для AMAT.

Действительно, с точки зрения интенсивности использования памяти существует два типа приложений, а именно приложения с интенсивными вычислениями и приложения с интенсивным использованием данных, хранящихся в памяти. Например, в приложении *calculix* с низким значением коэффициента корреляции между APC и AMAT доля инструкций обращения к памяти составляет 26%, тогда как в приложении *castusADM* с высокими значениями коэффициентов корреляции между APC и AMAT доля инструкций обращения к памяти составляет 58%.

Анализ результатов определения коэффициентов корреляции говорит о том, что на корреляцию между показателями быстродействия памяти и производительностью МС в основном влияет характер приложения.

5. Анализ влияния показателей качества предвыборки на быстродействие памяти

В предыдущем разделе нами установлено, что производительность системы в наибольшей степени зависит от двух показателей быстродействия памяти – от количества обращений к памяти на один такт (APC) и от среднего времени обращения к памяти (AMAT). Далее предстоит выявить влияние на APC и AMAT показателей качества предвыборки: степени покрытия и точности.

Рассчитаем значения APC и AMAT для различных размеров таблицы адресов промахов – MAT. Так как нам известны значения степени покрытия и точности для каждого размера MAT (табл. 2, 3), можно определить влияние показателей качества на быстродействие памяти и косвенным путем и на производительность МС.

В расчетах воспользуемся следующими данными о времени выборки из кэшей первого и второго уровней (L1, L2) и из главной памяти, приведенными в работе [6]. Время выборки из кэш L1 (*access time* L1 – acL_1) = 2 такта; время выборки из кэш L2 (*access time* L2 – acL_2) = 12 тактов; время выборки из главной памяти (*access time* L1 – acmem) = 200 тактов. Предположим, что при отсутствии предвыборки относительное число попаданий – *hit ratio* – в кэш L1 равно 0.9, а в кэш L2 – 0.95, относительное число промахов – *miss ratio* L1 равно 0.1, *miss ratio* L2 – 0.05.

Вероятность выборки данных из кэш L1 (P_{L1}) равна относительному числу попаданий в кэш L1 (*hit ratio* L1).

Вероятность выборки данных из кэш L2 (P_{L2}) равна относительному числу промахов в кэш L1, умноженному на относительное число попаданий в кэш L2:

$$P_{L2} = miss\ ratio_{L1} \times hit\ ratio_{L2}.$$

Вероятность выборки данных из главной памяти равна относительному числу промахов в кэш L1, умноженному на относительное число промахов в кэш L2:

$$P_{mem} = miss\ ratio_{L1} \times miss\ ratio_{L2}.$$

Среднее время обращения к памяти равно времени выборки из кэш L1, умноженному на *hit ratio* L1, плюс время выборки из кэш L2, умноженное на вероятность выборки из кэш L2, плюс время выборки из главной памяти, умноженному на вероятность выборки из главной памяти:

$$AMAT = (hit\ ratio) \times ac_{L1} + P_{L2} \times ac_{L2} + P_{mem} \times ac_{mem}$$

И, наконец, примем, что время обращения к памяти на такт APC примерно равно единице, деленной на среднее время обращения к памяти – AMAT:

$$APC \approx 1/AMAT.$$

Результаты расчетов APC и AMAT приведены в табл. 7. При расчетах использовали значение степени покрытия, усредненное по всем приложениям и различному числу предсказаний. Размер буфера предвыборки *PB* равен 32 строкам.

В табл. 8 приведены результаты вычислений APC и AMAT для размера буфера предвыборки *PB*, равного 16 строкам. Для сравнения следует отметить, что при отсутствии предвыборки

AMAT ≈ 4 тактам, а APC ≈ 0.25 1/такт.

Относительное число промахов в табл. 7 и 8 уменьшалось при увеличении размера таблицы адресов промахов – MAT, так как при этом возрастает степень покрытия.

Данные со значениями APC и AMAT в табл. 7 и 8 позволяют выбрать параметры Марковского предвыборщика для приложений с различными значениями коэффициентов корреляции между APC и IPC и AMAT и IPC. Для приложений с высокими значениями коэффициентов следует использовать размеры таблицы MAT 8K или 16K строк и размер буфера предвыборки *PB* 32 строки. Для большинства приложений с низкими значениями коэффициентов целесообразно использовать размеры таблицы MAT 4K или 8K строк и размер буфера предвыборки *PB* 32 или 16 строк.

Таблица 7. Результаты расчетов APC и AMAT (размер буфера предвыборки 32 строки)

	Размер MAT в строках				
	1K	**2K**	**4K**	**8K**	**16K**
Среднее значение степени покрытия	0.091	0.2	0.28	0.32	0.38
miss ratio$_{pref}$	0.09	0.088	0.079	0.076	0.068
Вероятность выборки данных из кэш L2 (P_{L2})	0.095	0.084	0.075	0.072	0.065
Вероятность выборки данных из главной памяти (P_{mem})	0,0045	0.0044	0.0039	0.0038	0.0034
AMAT (тактов)	3.86	3.71	3.62	3.49	3.33
APC (1/такт)	0.26	0.27	0.28	0.29	0.3

Таблица 8. Результаты расчетов APC и AMAT (размер буфера предвыборки 16 строк)

	Размер MAT в строках				
	1K	**2K**	**4K**	**8K**	**16K**
Среднее значение степени покрытия	0.082	0.18	0.25	0.29	0.34
miss ratio$_{pref}$	0.098	0.097	0.087	0.084	0.075
Вероятность выборки данных из кэш L2 (P_{L2})	0.0857	0.086	0.0867	0.087	0.088
Вероятность выборки данных из главной памяти (P_{mem})	0.0049	0.0048	0.0043	0.0042	0.0037
AMAT (тактов)	3.9	3.75	3.71	3.52	3.41
APC (1/такт)	0.25	0.26	0.27	0.28	0.29

Выводы

Рассмотрено применение предвыборки данных в кэш, как эффективного средства ускорения обращения к памяти. Подробно писан предвыборщик, построенный на основе Марковской модели промахов. Он предназначен для предсказания промахов при работе приложений с неструктурированными данными, обращение к которым происходит по случайным адресам.

Прослежено влияние размеров таблицы адресов промахов и буфера предвыборки на показатели качества предвыборки. Показано, что на данном этапе невозможно оценить, как влияют эти показатели качества на производительность МС, то есть насколько возрастает производительность МС и возрастает ли она вообще при их повышении. Установлено, что наибольшее влияние на производительность МС оказывают два показателя: число обращений к памяти на один такт – APC и среднее время потерь от промаха в кэш – AMAT.

На основании вычисления влияния параметров Марковского предвыборщика и показателей качества предвыборки на APC и AMAT выработаны рекомендации по выбору значений параметров марковского предвыборщика для приложений с интенсивными вычислениями и приложений с интенсивным использованием данных в памяти.

Литература:

1. Joseph D., Grunwald D. Prefetching Using Markov Predictors // IEEE Transactions on Computers. 1999. Vol. 48. № 2. P. 121–133.

2. Попкова Е.Я., Шмейлин Б.З. Повышение производительности микропроцессорных систем путем эффективного использования кэш // Системы и средства информатики. 2006. Вып. 16. С. 87–98.

3. Pathak P., Sarwar M., Sohoni S. Markov Prediction Scheme for Cache Prefetching // Proceeding of 2nd Annual Conference on Theoretical and Applied Computer Science. November 5, 2010. Stillwater, OK: ECE Department Oklahoma State University, 2010. P. 14–19.

4. Bandyopadhyay R., Liu M, Pena N. Tradeoff between coverage of a Markov prefetcher and memory bandwidth usage // Elec525, 2005, Spring.

5. Wu J., Luo G., Fulmer M. // Trace Driven Prefetching, ECE 252, 2009.

6. Wang D, Sun X.-H. APC: A Novel Memory Metric and Measurement Methodology for Modern Memory Systems // IEEE Transactions on Computers. 2014. Vol. 63. № 7. P. 1626–1639.