存储器行缓冲区命中预测研究

王得利 高德远 王党辉 孙华锦
(西北工业大学计算机学院 西安 710072)

摘 要 存储系统已经成为提高计算机系统性能的一个瓶颈。现利用DRAM存储器的访问特性来减少存储器访问操作的平均延迟。首先对存储器行缓冲区的控制策略进行研究,提出了读写分离式页模式预测器,并提出了双饱和计数器预测器和2级预测器等两种预测器方案;然后以SimpleScalar搭建的仿真平台对提出的预测方案进行了性能评估。结果显示,与缓冲区"关"策略相比,平均访问延迟减少了26%,IPC平均提高了4.3%;与缓冲区"开"策略相比,平均访问延迟减少了19.6%,IPC平均提高了2.5%。

关键词 存储系统,行缓冲区,页模式预测,读写分离
中图法分类号 TP302 **文献标识码** A

Research on Row Buffer Hit Prediction for Memory Access

WANG De-li GAO De-yuan WANG Dang-hui SUN Hua-jin
(College of Computer, Northwestern Polytechnical University, Xi'an 710072, China)

Abstract Memory system becomes a bottle-neck for the overall computer system performance. This paper reduced the average memory access latency using the characteristics of DRAM. Firstly, the control strategy for memory row buffer was studied. Secondly, a Read and Write Separated Predictor (RWS) for page mode prediction was proposed, a two saturated counter and a two level predictor to realize RWS were also proposed. Finally, a simulation platform based on SimpleScalar was constructed to evaluate the proposed predictors. The result shows that compared with CLOSE strategy, the average memory access latency is reduced about 26%, the average IPC speedup is about 4.3%. It is also shown that the average memory access latency is reduced about 19.6%, the average IPC speedup is about 2.5% Compared with OPEN strategy.

Keywords Memory system, Row buffer, Page mode prediction, Separated read and write

1 简介

集成电路工艺和微处理器体系结构的发展,促使微处理器的频率以每年60%的速率增长。而存储器的访问速度每年仅增长7%[1]。存储系统已经成为提高计算机系统性能的瓶颈之一。

现代计算机系统的主存通常由DRAM(Dynamic Random Access Memory)构成,其访问延迟在访存延迟中占有很大比重。而主存的访问传统上被简化成一个简单模型,即所有主存访问延迟均相同。事实上,DRAM在不同的访问条件下的访问延迟是不一样的,通过一些优化方法,可以降低主存的平均访问延迟[2]。

一个完整存储器访问操作的访问延迟包括预充电、行访问和列访问三部分。如果行缓冲区中包含有效的所需数据,则可以从行缓冲区中直接读出数据,不需要进行预充电和行访问操作,此时的访问延迟仅为列访问时间。因此常用的行缓冲区控制策略可分为两种:"开"策略和"关"策略。"开"策略在当前存储器访问操作完毕后,继续保持行缓冲区的数据有效,直到下一次访问操作开始执行,这样如果后续访问操作对同一行数据进行访问(行缓冲区中命中),则可以直接从行缓冲区中读出所需数据,从而获得最小的访问延迟。"关"策略,在存储器访问操作执行完毕后,随后进行预充电操作,从而使下一次存储器访问延迟可以减少为行访问时间加列访问时间。在行缓冲区不命中时,相对"开"策略,可以减少存储器访问延迟,但是在行缓冲区命中,访问延迟却相对"开"策略有所增加。

从上述分析中可知,通过行缓冲区可以减少存储器的平均访问延迟。然而常用的行缓冲区控制策略:"开"策略和"关"策略都有一定的局限性,"开"策略适合于行缓冲区命中率较高的场合,"关"策略在行缓冲区命中率较低时效果较好。为了能够在各种场合下均能有效减少存储器平均访问延迟,需要根据行缓冲区是否命中来动态控制行缓冲区的"开"与

到稿日期:2009-07-28 返修日期:2009-10-19 本文受国家自然科学基金项目(60573107),国家自然科学基金项目(60573143),国家863项目(2007aa010402)资助。

王得利(1981-),男,博士生,CCF会员,主要从事计算机体系结构方面的研究,E-mail:wdl900@mail.nwpu.edu.cn;高德远 男,博士生导师,主要从事计算机体系结构方面的研究;王党辉 男,博士,副教授,主要从事计算机体系结构方面的研究;孙华锦 男,博士,主要从事计算机体系结构方面的研究。

"关",即行缓冲区命中时使用"开"策略,反之,在不命中时使用"关"策略,这样既能在行缓冲区命中时利用行缓冲区来减少存储器访问延迟,也能在行缓冲区不命中时掩盖预充电时间,从而更为有效地减少存储器平均访问延迟。

要实现行缓冲区的动态控制策略,需要知道下一存储器访问操作能否在行缓冲区中命中。对于连续的存储器访问操作,即当前存储器访问操作还未完成,下一访问操作已经到来,此时存储器控制器已经知道下一存储器访问操作的地址,可以确认其是否在行缓冲区中命中,故可根据不同的情况,采取相应的操作。但是对于离散的存储器访问操作,也就是说当前存储器访问操作完成时,下一访问操作还未到来,此时存储器处于空闲状态,无法知道下一存储器访问操作的地址,也就无法知道下一存储器访问操作在当前行缓冲区中是否命中。为此,首先需要对行缓冲区是否命中进行预测,然后根据预测结果决定行缓冲区的"开"或"关"。相应地,行缓冲区命中预测器预测准确性的高低就决定了行缓冲区动态控制策略下存储器平均访问延迟的大小。

2 行缓冲区动态控制预测的必要性与可行性

行缓冲区的动态控制建立在高效的预测机制基础上,预测的前提条件是相邻两次存储器访问操作之间有空闲时间,即当存储器处于空闲状态时,才需要对下一次存储器访问请求进行预测,下一次即将到来的存储器访问操作也被称为可预测存储器访问操作,如果可预测存储器访问操作在行缓冲区中发生不命中,则称为可预测缓冲区不命中。通过仿真平台统计了各个仿真程序存储器访问操作总数和可预测存储器访问操作数,计算出各个仿真程序可预测存储器访问操作占所有存储器访问操作的百分比,如图 1 所示。

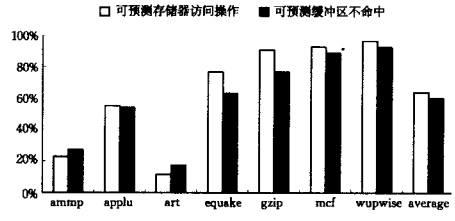

图 1 可预测存储器访问操作和可预测缓冲区不命中概率

从图 1 中可以看到,可预测存储器访问操作占所有存储器访问操作的百分比平均为 63.6%,其中最高的 wupwise 为 96.5%,最低的 art 为 11.6%(art 是 SPEC2000 中典型的存储器访问量大,对存储器带宽要求高的应用程序[3],故其可预测存储器访问操作相对较少)。图 1 中还列出了可预测行缓冲区不命中占所有行缓冲区不命中的百分比,从图中可以看出,两者相差不大,后者的均值也达到了 59.8%,这说明在存储器空闲时对存储器访问操作进行预测是有实际意义的。

有了可预测的存储器访问操作,还需要考虑影响行缓冲区动态控制策略性能的另一个因素,即可预测存储器访问操作的空闲间隔时间。这是因为由存储器访问操作预测结果引发的存储器操作需要一定的时间来完成,例如,本文的存储器模型中,进行一次预充电操作需要 3 个时钟周期,一次行访问需要 3 个时钟周期,这样如果在最恶劣的情况下,预测器预测下一次存储器访问操作与当前操作不同行,则需进行一次预充电和一次行访问操作,这需要 6 个时钟周期的时间。如果空闲间隔时间太短,不足 6 个时钟周期,就会给下一次存储器访问带来额外延迟,从而影响动态控制策略的性能。同样通过仿真平台,本文统计了可预测存储器访问操作的平均空闲间隔时间,如表 1 所列。

表 1 可预测存储器访问操作平均空闲间隔时间

SPEC 程序	ammp	applu	art	equake	gzip	mcf	wupwise
空闲间隔时间(Cycles)	113.3	11.6	6.1	21.3	337.6	6.7	59.5

从表 1 中可以看到,art 的平均空闲间隔时间最短为 6.1 个时钟周期,超过了上述所需的 6 个时钟周期,最多的 gzip 可达到几百个时钟周期,所以,可以认为可预测存储器访问操作的空闲间隔时间足够预测器完成相关的操作,不会引起额外的延迟,故在本文的存储器仿真模型中没有考虑预测器可能引起的额外延迟。

3 存储器行缓冲区命中预测器

根据行缓冲区的读、写操作行缓冲区命中率差异显著的特点,本文提出了读写分离式预测器(Read Write Separated Predictor RWS),其读、写操作分别由不同结构的预测器进行预测。

行缓冲区是否命中的预测是一个二元预测,与常见的转移分支预测类似,可以利用命中历史信息来预测将来的存储器访问是否在行缓冲区命中。文献[4]所讨论的 4 位历史信息寄存器预测器(FHP)利用 4 位移位寄存器来保存命中历史信息,并根据命中历史信息进行预测。该预测器结构简单,实现容易,但是缺点也很明显,没有利用存储器行缓冲区访问的固有特性。存储器访问一般分为两种:读操作和写操作,但是两类操作的行缓冲区命中率却是不一样的,读操作的命中率远远高于写操作[5]。这是由于现代处理器的二级 Cache 对写操作一般都采用写回策略,对于采用写回策略的二级 Cache,其写操作大部分都是由于二级 Cache 不命中而引起的写回操作,写回操作地址与其之前的读操作地址仅仅是 Cache 的标志位不同,其 Cache 组地址是相同的,映射到存储器中必然会使得写回操作的体地址与前面的读操作相同,而行地址不同,这样写回操作必然会引起行缓冲区的不命中。即使使用写缓冲区来缓存写操作,让读操作优先执行,但对提高行缓冲区命中率作用也不明显[5]。图 2 统计了所选仿真程序在"开"策略下读、写操作的行缓冲区命中率。从图中可以看出,读操作的行缓冲区命中率平均达到 61.7%,而写操作仅为 0.8%,二者差异显著。

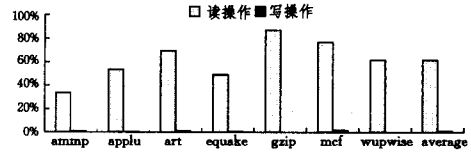

图 2 不同应用程序读、写操作行缓冲区命中率

根据行缓冲区的上述特点,本文提出读写分离式行缓冲区命中预测器,并根据读操作预测机制的不同设计两种不同结构的读写分离式预测器,分别为双饱和计数器预测器(Two Saturate Counter Predictor TSC)和 2 级预测器(Two Level Predictor TLP)。

3.1 双饱和计数器预测器

饱和计数器是最基本、最简单的二元预测器,也广泛应用于转移分支预测中,故最简单的读写分离式行缓冲区命中预

测器就是使用两个两位饱和计数器分别预测读、写操作的命中与否。考虑到读操作的行缓冲区命中率较高，写操作的命中率较低，故读操作采用普通结构饱和计数器，当饱和计数器的值大于等于预设阈值 2 时，预测行缓冲区命中。反之则预测缓冲区不命中，更新时根据行缓冲区实际命中结果更新饱和计数器的值，若缓冲区命中，则计数器加 1，否则计数器减 1。写操作饱和计数器则采用滞后型饱和计数器，其状态转换如图 3 所示。

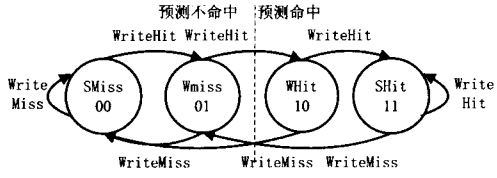

图 3　滞后型饱和计数器的状态转换图

在图 3 中，滞后型饱和计数器的阈值设为 2，在 2，3 状态做命中预测，在 0，1 状态做不命中预测，状态转换的条件为写操作在行缓冲区是否命中。在弱命中状态(2 状态)或强命中状态(3 状态)时，若写操作在行缓冲区不命中，则饱和计数器减 2，而不是减 1，也就是说会直接进入强不命中状态(2 状态)或弱不命中状态(1 状态)，而跳过了弱不命中状态(1 状态)或弱命中状态(0 状态)。其余状态的转换与正常饱和计数器一样，命中时，计数器加 1，若到了强命中状态(3 状态)，则计数器值保持不变；不命中时，计数器减 1，若到了强不命中状态(0 状态)，则计数器值保持不变。

双饱和计数器预测器的结构示意图如图 4 所示。预测时首先根据当前访问操作的不同类型选择不同的饱和计数器，若当前操作为读操作，则根据普通饱和计数器的值进行预测；反之，若是写操作，则采用滞后型饱和计数器的值进行预测。饱和计数器更新时也是根据不同访问操作更新相应的饱和计数器，读操作仅更新读饱和计数器，写操作仅更新写饱和计数器。

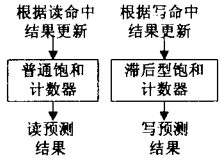

图 4　TSC 结构示意图

3.2　2 级预测器

2 级预测器的结构如图 5 所示。写操作的预测结构保持不变，仍然使用滞后型饱和计数器，读操作的预测结构借鉴转移分支 2 级预测器的思想，使用 2 级预测器，包括一个 N 位的命中历史表(Hit History Table，HHT)和一个 2^n 入口的模式历史表(Pattern History Table，PHT)。命中历史表中存储的是前 n 次存储器访问操作在行缓冲区中命中的历史状况，模式历史表中是 2^n 个 2 位饱和计数器。

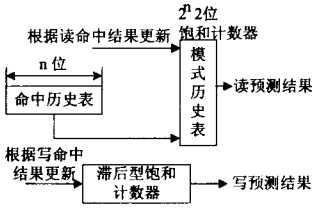

图 5　TLP 结构示意图

预测时同样要根据当前访问操作类型选择不同的预测器，若是写操作，则选择滞后型饱和计数器，若是读操作，则选择 2 级预测器，根据命中历史表中的历史状况选择模式历史表中相应的饱和计数器进行预测。更新时写操作更新滞后型饱和计数器，读操作更新 2 级预测器，包括命中历史表和模式历史表中相应的饱和计数器。

4　行缓冲区命中预测器性能分析

利用读出行缓冲区命中预测器可以实现缓冲区的动态控制，从而减少可预测存储器访问操作的平均访问延迟。为了具体分析行缓冲区命中预测器带来的性能提升，表 2 中列出了不同行缓冲区控制策略下各种情况的存储器访问延迟(表中假设预充电时间、行访问时间、列访问时间相同均为 t)。从表中可以看出，在缓冲区命中的情况下，“开”策略的存储器访问延迟最优，为 t；“关”策略的存储器访问延迟最大，为 $2t$；动态策略的存储器访问延迟在预测成功时与“开策略”一致，为 t；在预测失败时与“关”策略相同，为 $2t$。在缓冲区不命中的情况下，“关”策略的存储器访问延迟最小，为 $2t$；“开”策略的存储器访问延迟最大，为 $3t$；动态策略的存储器访问延迟在预测成功时与“关策略”一致，为 $2t$；在预测失败时与“开”策略相同，为 $3t$。

表 2　不同控制策略下存储器访问延迟表

存储器访问	“开”策略	“关”策略	动态策略	
			预测成功	预测失败
缓冲区命中	t	2t	t	2t
缓冲区不命中	3t	2t	2t	3t

根据表 2 中数据，只要预测器预测成功，动态策略的存储器访问延迟在 3 种策略中均为最小，因此，预测器预测成功率直接决定了动态策略的性能。设缓冲区命中时动态预测成功的次数为 $P1$，预测失败的次数为 $P2$，缓冲区不命中时动态预测成功的次数为 $P3$，预测失败的次数为 $P4$，则行缓冲区命中预测器预测成功率 P 可用下式计算：

$$P=\frac{P1+P3}{P1+P2+P3+P4}$$

预测器预测成功率 P 是从整体上反映预测器的性能的高低，但无法与“开”或“关”策略直接进行性能比较，为此本文定义了两个参数：“开”有效成功率 P_{open} 和“关”有效成功率 P_{close} 用于与“开”或“关”策略进行性能比较。

定义 1　“开”有效成功率 P_{open} 表征行缓冲区命中预测器相对于“开”策略的有效预测成功率，计算公式如下：

$$P_{open}=\frac{P3}{P2+P3}$$

推论 1　如果“开”有效成功率 P_{open} 大于 50%，则行缓冲区动态控制策略(使用行缓冲区命中预测器)的平均访问延迟低于“开”策略。

证明：若 $P_{open}>50\%$，则根据 P_{open} 计算公式可知 $P3>P2$；

由表 2 可知“开”策略的总访问延迟 T_{open} 为：

$T_{open}=(P1+P2)\times t+(P3+P4)\times 3t=(P1+P2+3P3+3P4)t$；

“动态”策略的总访问延迟 $T_{dynamic}$ 为：

$T_{dynamic}=P1\times t+P2\times 2t+P3\times 2t+P4\times 3t=(P1+2P2+2P3+3P4)t$；

所以 $T_{dynamic}-T_{open}=(P2-P3)t<0$；

推论成立。

定义 2　“关”有效成功率 P_{close} 表征行缓冲区命中预测器

相对于“关”策略的有效预测成功率，计算公式如下：

$$P_{close}=\frac{P1}{P1+P4}$$

推论 2 如果“关”有效成功率 P_{close} 大于 50%，则行缓冲区动态控制策略（使用行缓冲区命中预测器）的平均访问延迟低于“关”策略。

证明：若 $P_{close}>50\%$，则根据 P_{open} 计算公式可知 $P1>P4$；

由表 2 可知“关”策略的总访问延迟 $Tclose$ 为：

$T_{close}=(P1+P2)\times 2t+(P3+P4)\times 2t=(P1+P2+P3+P4)\times 2t$；

“动态”策略的总访问延迟 $T_{dynamic}$ 为：

$T_{dynamic}=P1\times t+P2\times 2t+P3\times 2t+P4\times 3t=(P1+2P2+2P3+3P4)t$；

所以 $T_{dynamic}-T_{close}=(P4-P1)t<0$；

推论成立。

5 仿真实验

5.1 仿真环境的建立

为了评估存储器行缓冲区预测器的有效性，本文选用 SimpleScalar 进行仿真评估。为了适应本文的仿真需要，在 SimpScalar 的基础上进行了扩充，增加了对存储器读出放大缓冲区动态控制策略的支持，改进了其预取机制和主存储器访问模型，使之适合本文研究的需要。扩展后的系统结构如图 6 所列。

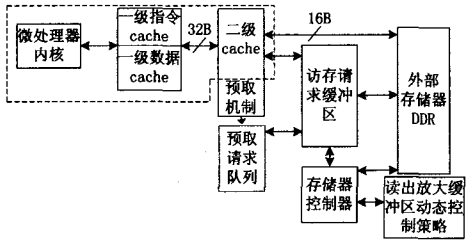

图 6 仿真系统结构示意图

图 6 中虚线框内为 SimpleScalar 仿真器原有的结构，虚线框外的部件是为了适应本文研究需要而加以扩展的，包括多种预取机制的实现、读出放大缓冲区动态控制策略的实现以及 DDR 存储器访问模型的扩展。仿真平台的存储系统包括两级 Cache 结构和 DDR 存储器，数据总线宽度为 16Bytes。仿真器的微处理内核未作改动，使用的参数如表 3 所列。本文使用 Cygnus Cygwin32 作为 SimpleSaclar 运行平台。

表 3 SimpleScalar 处理器核的基本配置参数

参数	值
前端总线/处理器时钟	800Mhz/4G
取指宽度	8
译码宽度	8
发射宽度	8，乱序发射
完成宽度	8
访存队列	32
RUU	64
功能部件	定点 ALU:4；浮点 ALU:4；定点乘法器:1；浮点乘法器:1
L1 Cache 端口数	2
转移预测错误延时	3 个时钟周期
转移预测方案	2 级预测器，2 位饱和计数器，11 位全局历史表，2048 项模式历史表；4 路 512 组 BTB；8 项的返回地址堆栈；非推测性更新
第一级数据 Cache	共 64kB，1024 组，2 路组相连，每行 32 字节，LRU，实地址索引，命中时延为 1 个时钟周期
第一级指令 Cache	共 64kB，1024 组，2 路组相连，每行 32 字节，LRU，实地址索引，命中时延为 1 个时钟周期
第二级 Cache（指令/数据混和）	共 512kB，1024 组，4 路组相连，每行 128 字节，LRU，实地址索引，命中时延为 6 个时钟周期
MSHR 数	8，每个 MSHR 可响应 4 个请求
数据 TLB	共 8kB，2 路组相连，LRU
指令 TLB	共 8kB，2 路组相连，LRU
内部总线（连接一级 Cache 和二级 Cache）宽度	32B
存储器总线（连接二级 Cache 和主存储器）宽度	16B

本文的仿真程序均从 SPEC 程序中挑选而来，其中 ammp，art，equake，gzip，mcf，wupwise 为 SimpleScalar 编译的 SPEC2000 程序，applu 是 SimpleScalar 编译的 SPEC95 程序。

SimpleScalar 的主存储器访问模型比较简单[6]，把主存储器访问延迟简化为两个值：第一次存储器访问延迟和后续访问延迟，第一次访问延迟大于后续访问延迟，类似于读出放大缓冲区命中和不命中时的访问延迟。这种方法实现简单，也近似模拟了主存储器的访问状态，但是与主存储器访问的实际情况存在较大差距，为了更为准确地模拟主存储器访问的实际情况，本文以三星 512Mb DDR400（8M × 16Bit × 4 Banks）[7] 为参照对象，建立了一个 DDR 存储器访问模型，与 SimpleScalar 类似，其仿真参数如表 4 所列。本文的存储器访问模型同样没有考虑存储器刷新操作对系统性能的影响。

表 4 主存储器模型的仿真参数表

仿真程序	输入数据	初始跳过指令数（百万）	仿真指令数（百万）
ammp	参考输入	108	100
applu	训练输入	18	100
art	参考输入	67	100
equake	参考输入	194	100
gzip	参考输入：程序	486	100
mcf	参考输入	316	100
wupwise	参考输入	584	100

为了评测各种预测器的有效性，IPC（Instruction per Cycle）是理所当然的最终评测标准，它表示的是处理器每周期执行的指令数，直接反映了系统的性能。然而 IPC 的提高除了与预测方案有关外，还与可预测存储器访问数目以及存储器访问延迟对系统性能的影响度相关。因此为了更为直接地反映不同预测器的有效性，还使用可预测存储器访问操作的平均访问延迟作为评测标准，可预测存储器访问操作的平均访问延迟指的是所有可预测存储器访问操作访问存储器所花费的平均时间，它与预测器的效率直接相关。理想情况下，所有存储器访问操作都在行缓冲区中命中，其平均访问延迟为 2 个时钟周期。最恶劣的情况每次都不命中，则平均访问延迟为 8 个时钟周期。

5.2 行缓冲区命中预测器仿真结果

行缓冲区命中预测器通过对存储器访问在行缓冲区中命中与否进行预测来减少访存延迟，提高性能。能否实现预定目标的关键在于是否有足够高的预测成功率，因此首先统计出本节所设计的各个行缓冲区命中预测器的预测成功率 P，其中，2 级预测器模式历史表的表项数为 256。同时根据上节的分析，统计出“开”有效成功率 P_{open} 和“关”有效成功率 P_{close}，分别如图 7、图 8、图 9 所示。

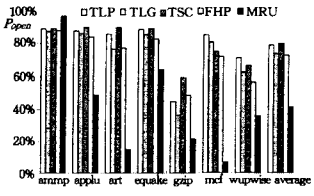

图 7　行缓冲区命中预测器预测成功率

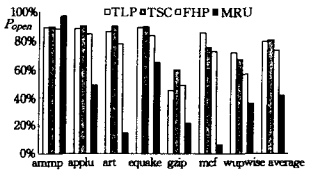

图 8　行缓冲区命中预测器“开”有效成功率

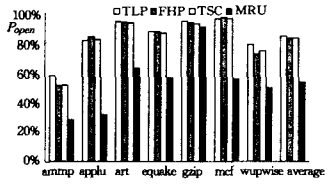

图 9　行缓冲区命中预测器“关”有效成功率

图中 TLP 代表的是图 5 所示的 2 级预测器，TSC 代表的是双饱和计数器预测器，FHP 代表的是 Alpha21174 所用的 4 位历史信息寄存器预测器，MRU 代表的是 Wong 提出的 MRU 策略。从 3 个图可以看到，除了 MRU 之外，其余预测器的预测成功率都较高，平均值达到了 80%左右，其中 2 级预测器和双饱和计数器预测器的预测成功率都达到 86%。而且其“开”有效成功率分别为 79%和 80%，“关”有效成功率为 84%和 82%，均超过了 50%，说明无论是相对于“开”策略还是“关”策略，使用行缓冲区命中预测器来动态控制缓冲区都会减少存储器平均访问延迟，从而取得性能上的提升。为了更为直观地显示行缓冲区命中预测器对存储器平均访问延迟的减少作用，统计了各个预测器下不同测试程序的可预测存储器访问操作平均访问延迟，如图 10 所示。

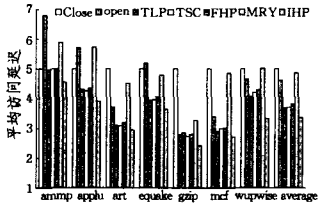

图 10　不同行缓冲区命中预测器的平均访问延迟图

从图 10 中可以看出，TLP，TLG，TSC 和 FHP 均显著减少了可预测存储器访问操作的平均访问延迟，其中 TLP 和 TSC 的效果最好，平均访问延迟被减少到 3.7/3.71 个时钟周期，相对于“关”策略的 5 个时钟周期，平均访问延迟约减少了 26%，相对于“开”策略的 4.6 个时钟周期，平均访问延迟约减少了 19.6%。图 10 中还列出理想行缓冲区预测器(IHP)的可预测访存操作平均访问延迟，约为 3.4 个时钟周期，低于所有实际行缓冲区预测器。这说明，即使对于性能最好的 TLP 和 TSC，相对于理想情况，其性能也还有一定的提升空间。

图 11 列出了不同预测器在不同测试程序中的规格化 IPC(以“关”策略为基准，下同)。从图 11 中可以看到，除了理想行缓冲区预测器(IHP)外，TLP 和 TSC 的平均规格化 IPC 最高，相对于“关”策略，IPC 平均提高了 4.3%，相对于“开”策略，IPC 平均提高了 2.5%。

从图 11 中还可以发现，MRU 策略的性能高于“关”策略，却低于“开”策略。这是因为 MRU 策略认为对同体的访问始终命中行缓冲区，而对不同体的访问始终不会命中行缓冲区，实际上存储器访问操作对行缓冲区的命中与否与是否同体没有必然的联系，再加上本文仿真条件下的行缓冲区的总体命中率是大于 50%的，因而在不同体访问操作的处理上，MRU 策略要逊色于“开”策略，而在同体访问操作的处理上，则要优于“关”策略，故造成了 MRU 策略优于“关”策略，低于“开”策略的状况。

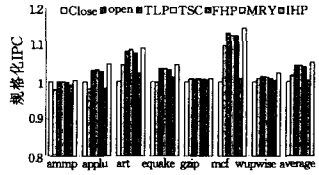

图 11　不同行缓冲区命中预测器的规格化 IPC 图

从仿真结果中可以看到，2 级预测器 TLP 和双饱和计数器 TSC 两种行缓冲区命中预测器的性能基本相当，但考虑到 TLP 需要使用模式历史表记录历史信息，而 TSC 仅需要两个饱和计数器即可完成预测，故综合实现代价和性能两个方面考虑，本文认为双饱和计数器预测器 TSC 是最优的行缓冲区命中预测器。

综合以上仿真结果和分析，本文认为使用合适的行缓冲区命中预测器动态控制存储器行缓冲区，可以减少存储器平均访问延迟，从而提高系统性能。但是从仿真结果上可以看到，即使是理想的行缓冲区命中预测器，其平均访问延迟也有 3.36 个时钟周期，与最理想情况(存储器访问操作每次都命中)下的 2 个时钟周期还有一定的距离，且其规格化的 IPC 相对于“关”策略仅提高了 5.2%，相对于“开”策略仅提高了 3.5%。性能提升不是很明显，相当一部分原因在于其预测到行缓冲区不命中时，仅仅保守地进行了预充电，而没有更进一步地打开可能的行，以便进一步减少平均访问延迟。为此，为了进一步提高存储器行缓冲区动态控制策略的效能，需要对存储器访问行地址进行预测。

结束语　本文在分析 DRAM 存储器访问操作特性的基础上，对存储器行缓冲区的动态控制策略进行了深入研究，提出了多种存储器行缓冲区命中预测器方案，包括双饱和计数器预测器和 2 级预测器。然后以 SimpleScalar 为基础，对其主存储器访问模型及相关部件进行了扩展，搭建了一个仿真平台，并利用该平台对所提出的预测器方案进行了性能仿真评估。仿真结果显示，综合实现代价和性能两方面的考虑，本文结合存储器访问特性提出的双饱和计数器方案最优。与缓冲区“关”策略相比，平均访问延迟减少了 26%，IPC 平均提高了 4.3%；与缓冲区“开”策略相比，平均访问延迟减少了 19.6%，IPC 平均提高了 2.5%。

存储器命中预测器仿真结果还显示以存储器命中预测器为基础的存储器行缓冲区动态控制策略的性能提升不是很明显，与理想情况相比还有较大的提升空间。这其中很大一部分原因在于其预测到行缓冲区不命中时，仅仅保守地进行了预充电操作，而没有进一步预测相应的行地址，打开相应的行，以便进一步减少访问操作的平均访问延迟。为此，可以对存储器访问行地址的预测进行深入研究，以便更加有效地发挥存储器行缓冲区动态控制策略的作用，进一步减少存储器访问操作的平均访问延迟。

参 考 文 献

[1] Hennessy J L, Patterson D A. Computer Architecture: A Quantitative Approach(3nd Edition)[M]. San Mateo: Morgan Kaufmann Publishers, 2002

[2] Alexander T, Kedem G. Distributed Prefetch-buffer/Cache De-

sign for High Performance Memory Systems[A]// IEEE Proceedings of the Second Internation Symposium on High-Performance Computer Architecture[C]. 1996:254-263
[3] Lai A, Fide C, Falsafi B. Dead-Block Prediction and Dead-Block Correlating Prefetchers[A]// Proceedings of the 28th International Symposium on Computer Architecture[C]. 2001:144-154
[4] Schumann R C. Design of the 2 1 1 7 4 Memory Controller for Digital Personal Workstations[J]. Digital Technical Journal, 1997, 9(2):57-69
[5] Zhao Z, Zhichun Z, Zhang X. A permutation-Based Page Interleaving Scheme to Reduce Row-Buffer Conflicts and Exploit Data Locality[A]// Proceedings of the 33rd Annual International Symposium on Microarchitecture[C]. 2000:32-41
[6] Burger D C, Austin T M. The Simplescalar Tool Set [R]. CS-TR-97-1342. 1997
[7] Samsung Semiconductor. 512 Mb B-die DDR400 SDRAM Specification[R]. Revision 1. 0. 2003

(上接第 285 页)

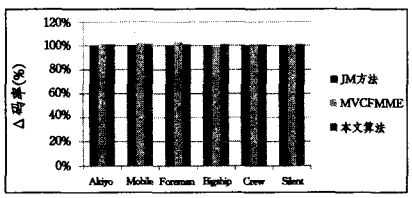

图 7 码率变化比较

速度的对比通过表 1 给出。表 1 中列出了在 16 和 28 两种量化参数下的实验结果，分别对应高码率和低码率两种情况。无论在低码率还是高码率，本文的算法对于全参考帧搜索方法(即 JM 中使用的算法)有至少一倍的提高，而对于快速算法 MVCMME，也有平均 20%的减少。在 Akiyo, Silent 和 Crew 这些运动不甚剧烈的序列中，本文算法能提供更快的速度。其原因主要是较静止的序列内周期运动和分数运动出现得较少，本文的算法可以跳过绝大多数对冗余参考帧的搜索。

表 1 本文算法与 H. 264 JM 快速运动估计算法的比较结果

视频序列	量化值	格式	Δ复杂度(%)与 JM 方法	Δ复杂度(%)与 MVCMME
Akiyo	16	CIF	−56.31	−27.14
Mobile	16	4CIF	−55.27	−24.72
Foreman	16	4CIF	−45.54	−25.6
Bigship	16	720p	−53.44	−22.48
Crew	16	720p	−58.18	−21.98
Silent	16	CIF	−57.89	−28.15
Akiyo	28	CIF	−51.11	−26.3
Mobile	28	4CIF	−49.2	−25.02
Foreman	28	4CIF	−55.41	−25.03
Bigship	28	720p	−49.89	−23.57
Crew	28	720p	−50.82	−28.8
Silent	28	CIF	−52.9	−27.9

结束语 本文提出了一种联合分数运动估计的多参考帧快速选择算法。首先，通过对多参考帧选择技术和分数运动估计技术的分析发掘了两者之间的相关性；然后，利用合成的运动向量和分数残差曲面模型，确定进行分数运动估计的参考帧。该方法仅在时间相邻、运动相邻和分数预测残差最小的参考帧上进行分数运动估计，在大幅地加快多参考帧运动估计速度的同时，保持了视频编码的率失真性能。

参 考 文 献

[1] Shen Liquan, Liu Zhi, Zhang Zhaoyang, et al. An Adaptive and Fast H. 264 Multi-Frame Selection Algorithm Based on Information from Previous Searches [C]// 2007 IEEE International Conference on Multimedia and Expo. 2007
[2] Sun Qichao, Chen Xin-hao, Wu Xiaoyang, et al. A Content-adaptive Fast Multiple Reference Frames Motion Estimation in H. 264 [C]// 2007 IEEE International Symposium on Circuits and Systems (ISCAS). 2007
[3] Liu Zhenyu, Li Lingfeng, Song Yang, et al. Motion Feature and Hadamard Coefficient-Based Fast Multiple Reference Frame Motion Estimation for H. 264 [J]. IEEE Transactions on Circuits and Systems for Video Technology, 2008, 18(5):620-632
[4] Chen Mei-juan, Li Gwo-long, Chiang Yi-yen, et al. Fast multi-frame motion estimation algorithms by motion vector composition for the MPEG-4/AVC/H. 264 standard [J]. IEEE Transactions on Multimedia, 2006, 8(3):478-487
[5] Chen Zhibo, Xu Jianfeng, He Yun, et al. Fast integer-pel and fractional-pel motion estimation for H. 264/AVC [J]. Journal of Visual Communication and Image Representation, 2006, 17(2): 264-290
[6] Yi Xiaoquan, Zhang Jun, Ling Nam, et al. Improved and simplified fast motion estimation for JM, JVT-P021. doc [C]// Joint Video Team (JVT) of ISO/IEC MPEG & ITU-T VCEG(ISO/IEC JTC1/SC29/WG11 and ITU-T SG16 Q. 6)16th Meeting. Poznan, Poland, 2005
[7] Alexis T, Pankaj T. Fast Subpixel Motion Estimation Support for the Enhanced Predictive Zonal Search Scheme, JVT-Q079 [C]// Joint Video Team (JVT) of ISO/IEC MPEG & ITU-T VCEG (ISO/IEC JTC1/SC29/WG11 and ITU-T SG16 Q. 6) 17th Meeting. Nice, France, 2005
[8] 陆寄远，张培钊，朝红阳. 一种面向 H. 264 KTA 的快速分数运动估计方法 [J]. 中国图象图形学报, 2010, 15(3):367-371

(上接第 288 页)

[4] Chen L F, Liao H Y, Ko M T, et al. A new LDA-based face recognition system which can solve the small sample size problem [J]. Pattern Recognition, 2000, 33(10):1713-1726
[5] Zhuang X S, Ddai D Q. Improved discriminant analysis for high dimension data and its application to face recognition [J]. Pattern Recognition, 2007, 40(5):1570-1578
[6] 宋枫溪，程科，杨静宇. 最大散度差和大间距线性投影与支持向量[J]. 自动化学报, 2004, 30(6):890-896
[7] Yang M H. Kernel Eigenfaces vs. Kernel Fisherfaces face recognition using kernel methods [A]// Proceedings of Fifth IEEE International conference on Automatic Face and Gesture Recognition[C]. Washington DC, USA, 2002:215-220
[8] Hu H F. Orthogonal neighborhood preserving discriminant analysis for face recognition [J]. Pattern Recognition, 2008, 41(9): 2045-2054
[9] Georghiades A S, Belhumeur P, Kriegman D, et al. From Few to Many: Illumination Cone Models for Face Recognition Under Variable Lighting and Pose [J]. IEEE Trans. Pattern Anal. Mach. Intelligence, 2001, 23(6):643-660