doi:10.11887/j.cn.201501001 http://journal.nudt.edu.cn

利用堆栈特征的片上末级缓存访问模式在线识别方法*

黄智濒[1,2],周 锋[1],马华东[1],祝明发[2],陶 袁[3]
(1. 北京邮电大学 计算机学院,北京 100876; 2. 北京航空航天大学 软件开发环境国家重点实验室,北京 100191;
3. 吉林师范大学 数学学院,吉林 四平 136000)

摘 要:很多优化处理器缓存利用效率的方法依赖于对访问请求序列的特征的探测或识别,例如,预取和绕开等。如何在线有效识别访问请求序列的特征依然是一个开放的问题。通过对典型访问模式的深入分析,发现其堆栈距离频度的分布展示出鲜明的特征。而模拟实验数据表明访问请求序列的特征具有一定的持续性和稳定性,具有检测和预测的可行性。因而提出了一种基于堆栈直方图峰值的在线识别访问模式的机制和方法,空间和时间开销都较小。对SPEC CPU2000/2006的15个程序的实验表明,所提方法均可正确识别测试程序的访问模式。

关键词:访问模式;识别;缓存;处理器
中图分类号:TP316 **文献标志码**:A **文章编号**:1001-2486(2015)01-001-07

An online access pattern identification method based on the stack characteristic in the on-chip last-level-cache

HUANG Zhibin[1,2], *ZHOU Feng*[1], *MA Huadong*[1], *ZHU Mingfa*[2], *TAO Yuan*[3]
(1. School of Computer Science and Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China;
2. State Key Laboratory of Software Development Environment, Beihang University, Beijing 100191, China;
3. College of Mathematics, Jilin Normal University, Siping 136000, China)

Abstract: Many methods optimizing the on-chip cache utilization are dependent on the profile or identification of the access sequence characteristics, for instance, prefetcher or bypass etc. How to identify these characteristics is still an open problem. Through a detailed theoretical analysis of typical access patterns, it is shown that the frequency of stack distance has obvious features. Furthermore, according to the results of the Simics simulation, these features present persistent and stable to a certain extent, therefore, they are feasible to be identified and predicted. An online method based on the peak value of the collected stack histogram attaching to each core is provided. In addition, the storage and time overhead is small. The experimental results based on 15 benchmarks from SPEC CPU2000/2006 show that it identifies all correctly.

Key words: access pattern; identification; cache; processor

除了传统的科学计算和在线事务处理等应用外,许多新的应用不断涌现,例如流媒体应用、虚拟化[1]等,给内存和缓存的管理提出了新的挑战。同时,随着多核处理器成为主流,在末级缓存中,多样性的负载并发混合在一起运行,使得相互之间的竞争和干扰变得更加复杂。作为事实上的工业标准的最近最少使用算法(Least Recently Used,LRU)系列替换算法,在多核末级共享缓存中表现一般,缺失量与理论优化算法之间的差距甚至达197%[2]。因此,学术界和工业界致力于优化缓存管理策略,特别是针对多核末级共享缓存。

在众多的优化缓存管理的方法中,许多的方法需要依靠差别化地处理不同的缓存请求实现缓存管理的优化。这类方法大致可分为两类,一类是通过过滤缓存请求优化缓存管理,主要是绕开或者丢弃一些重用少的缓存请求,例如传统的绕开机制及其扩展,这类方法的决策依据是由操作系统认定应用类型或者基于每一个缓存请求的地址重用历史信息,其覆盖范围和精度与其存储开销正相关。另一类方法是通过提供多种替换算法,在这几种替换算法之间切换。这类方法基于这样的一个事实,即:不同的缓存管理策略,如LRU、先进先出、随机替换或者最近最不常用算法等,针对不同的缓存访问请求序列,都有优于其他替换算法的情况[3]。这类方法考虑了缓存请求序列的多样性,依据组竞争机制或附属目录机制[3]在多个缓存替换策略之间

* **收稿日期**:2014-06-12
基金项目:中国博士后基金资助项目(2014M550662);软件开发环境国家重点实验室资助项目(SKLSDE-2014KF-04)
作者简介:黄智濒(1978—),男,湖北汉川人,博士,E-mail:huangzb@bupt.edu.cn

动态切换。在单核处理器中,这类方法表现良好,存储开销也不大。但是,在多核处理器中,这类方法遇到了很大的挑战,因为访问请求来源更加多样化,而且在末级共享缓存进行了混合,他们之间的竞争污染使组竞争机制更加复杂,精确度降低,开销呈指数级增加[4]。事实上,这类方法仅仅考虑了访问请求序列的替换结果,而没有考虑访问请求序列的特征。这也是导致存储开销急剧增加的重要原因之一。

所谓访问模式[5],就是在一段时间段内,访问请求序列中保持不变的特征。以前的研究中,针对内存访问行为或访问模式,提出对应用进行分类,例如,Chen 等[6]通过对程序访问模式的分析,总结出四种程序访存模式:标量访问、零步长访问、常步长访问、不规则访问。Qureshi 等[7]基于应用对缓存利用的特征,将应用分为低利用、饱和利用和高利用三类。Chang 等[8]进一步将高利用类的应用分成三个子类,即 Supplier,Sensitive 和 Thrashing。他们都是基于缓存访问缺失量的变化来对应用进行分类的。Jaleel 等[9]扩展了 DIP 管理策略[4],将应用分成四类,即“Cache Friendly”“Cache Fitting”“Cache Thrashing”和“Streaming”,基于多核共享末级缓存,提出了 TADIP 策略,利用组竞争机制在不同的替换算法之间切换。在 RRIP[5]机制中,Jaleel 等将访问模式分成四类,即 Recency-friendly(时间局部特征友好性访问模式)、Thrashing(颠簸访问模式)、Streaming(流访问模式)和 Mixed(混合访问模式),依据这些不同的多样化的访问模式采取不同的预测方法,优化缓存管理,但是,RRIP 依然没有解决如何进行访问模式识别的问题。章铁飞等[10]提出基于程序的访存模式优化片上缓存和主存的功耗,但是依然没有给出如何进行访问模式识别的问题的解决方法。

针对上述问题,基于 RRIP 提出的访问模式的分类方法,通过分析和大量的实验,发现:访问请求序列的特征具有一定的持续性和稳定性,具有检测和预测的可能性;而且不同的访问模式的堆栈距离频度的分布展示出鲜明的特征,如堆栈直方图曲线的峰值、拐点、形状等。因此,提出了一种访问模式的在线识别方法:峰值识别方法。该方法假设在一个短的采样周期内,访问模式都可以近似归为三种单纯的访问模式之一。这样就可以基于对这三种单纯的访问模式的显著特征的判断来识别在线的访问请求特征,特别是峰值位置和曲线形状。

1 典型访问模式

RRIP 机制中,Recency-friendly 是指时间局部特征友好性访问模式,新载入的缓存块很快会被再次访问;Thrashing 是指颠簸访问模式,新载入的缓存块在一个相对固定的间隔之后会被再次访问;Streaming 是指流访问模式,新载入的缓存块不会被再次访问;Mixed 是指混合访问模式,由三种单纯的访问模式混合而成的访问序列。以此为基础,研究如何在线识别和判断这些典型的访问模式。

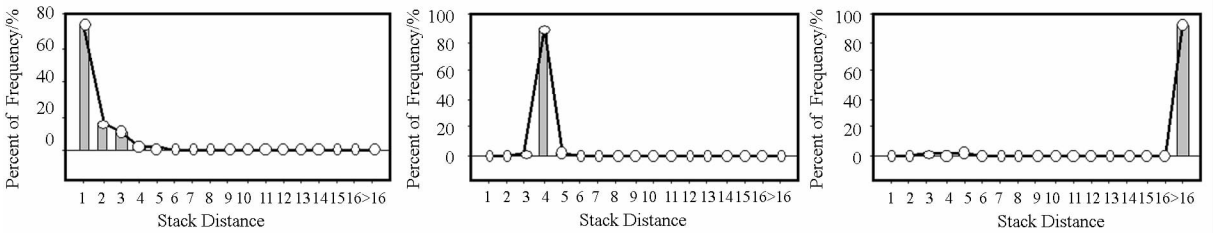
所谓时间局部特征友好性访问模式,就是最近访问过的缓存块会在最短的时间再次被访问,也称堆栈访问模式,如图 1(a)所示。对于这种访问模式而言,LRU 替换算法是非常适合的。从堆栈距离的角度看,对于任意长度的这种访问序列,堆栈距离为 $1,2,3,\cdots,k-1$ 的访问频度都是 1。假设每一个请求地址的概率都是相等的,设为 p,则长度为 k 的访问请求序列映射到同一个缓存组的概率是 k/A(A 是缓存组的关联度,且假设 $k<A$)。RD_i 代表堆栈距离等于 i 的堆栈重用频率。其中,RD_i 的最大值发生在堆栈距离等于 1 附近。RD_i 应是单调递减的,RD_i 的最大值出现在重用距离较少时。

所谓颠簸访问模式,就是新载入的缓存块在一个相对固定的间隔之后会被再次访问。它与颠簸现象并不是一个概念。颠簸现象是针对某一个固定的存储空间,由于访问重用距离的长度原因,而导致数据刚被移出,又出现被访问而引起移入的现象。颠簸模式则是单纯地指初次访问缓存块后再次被访问时,间隔距离固定的一种访问序列。图 1(b)表达了一个典型的间隔为 k 的颠簸访问模式。由于存在着缓存请求流到各个缓存组的映射过程,k 的大小受映射过程的影响会各有不同,这个值称为阈值,从重用距离的角度分析,对于任意的 k 值,重用距离小于 k 或者大于 k 的频度的比例为 0。T_k 指映射到同一个缓存组的访问序列的长度,如果 T_k 大于缓存组的关联度 A,则颠簸现象就会发生。只有当堆栈距离 i 等于 $T_k \bmod A$ 时,RD_i 会取最大值,而其他的堆栈距离的重用频率均为 0。

所谓流访问模式,就是新载入的缓存块不会被再次访问,如图 1(c)所示。在颠簸访问模式中,如果 k 等于无穷大,则变成了流访问模式。这种访问模式下,任何的替换算法都不会缓存命中,因为没有数据会被重用。这样堆栈距离大于 A

的重用频率达到最大值。

从上述的分析中可以看出,这三类单纯的访问模式,堆栈距离频率的峰值和堆栈距离曲线的形状有着鲜明的特征,这些特征可以被用来在线识别这些访问模式。



$(a_1,a_2,\cdots,a_{k-1},a_k,a_k,a_{k-1},\cdots,a_2,a_1)^N$　　$(a_1,a_2,a_3,\cdots,a_{k-1},a_k)^N$　　$(a_1,a_2,a_3,\cdots,a_{k-1},a_k)$

(a) 近期友好访问模式(k 为任意自然数)　(b) 颠簸访问模式(k 大于一定的阈值)(c) 流访问模式(k 为无穷大)

(a) Recent-friendly access pattern　(b) Thashing access pattern　(c) Streaming access pattern

图 1　缓存请求访问模式及其理论堆栈距离直方图

Fig. 1　Typicalcache access patterns and their theoretical stack distance histograms

2　峰值在线识别方法

2.1　访问模式在线识别的机制

众所周知,真 LRU 替换算法具有堆栈属性。允许创建堆栈距离直方图(Stack Distance Histograms,SDH),可以由两种方式获得(在系统中单独运行线程的执行过程中,或者加入一些硬件概要计数器)。Qureshi 等[7]提出了用一个低开销的电路来测量 SDH 和 UCP(Utility-Based Cache Partitioning)缓存分区机制,使用 ATD(Auxiliary Tag Directory)统计出堆栈距离直方图。本文的目的不是为了计算 SDH,而是为了在线识别访问模式,因此不需要 ATD,而只需要将少量的缓存块组成一个地址跟踪阵列(Address Trace Array,ATA),ATA 可以通过保留一个 L2 缓存的缓存组或者使用寄存器文件构成。具体的架构如图 2 所示。其中,访问模式识别器位于 L1 缓存和 L2 缓存之间,用于监控达到 L2 缓存的来自于各个处理器核的访问请求序列,每个处理器核配置一个。

访问模式识别器主要由三个部件构成:1)地址跟踪阵列;2)堆栈距离直方图计数器组(Counters of Stack Distance Frequency,CSDFs),由 $A+1$ 个计数器组成,每个计数器 32 位,A 是 ATA 阵列的单元数;3)峰值识别算法。依据实验情况,ATA 只需要配置 8 个单元(缓存块)就可以正确识别访问模式。

由于受到了 L1 缓存的过滤效应的作用,访问模式识别器的带宽压力减轻,且不在访问请求的关键路径上,时间开销小。

2.2　访问模式在线识别的可行性

利用访问模式在线识别机制,设置 ATA 阵列的单元数为 16,计数器组内包含 17 个计数器,单独运行测试程序,统计到达 L2 缓存的访问请求的堆栈距离直方图,任意选择 15 个来自于 Spec CPU2000/2006 的基准测试程序。基于全系统的模拟器 Simics,在 L2 缓存中,每运行 10M 指令,采样统计堆栈距离直方图,并且连续采样 40 次,所有的堆栈直方图曲线如图 3 所示,从图 3 中,可以得出:

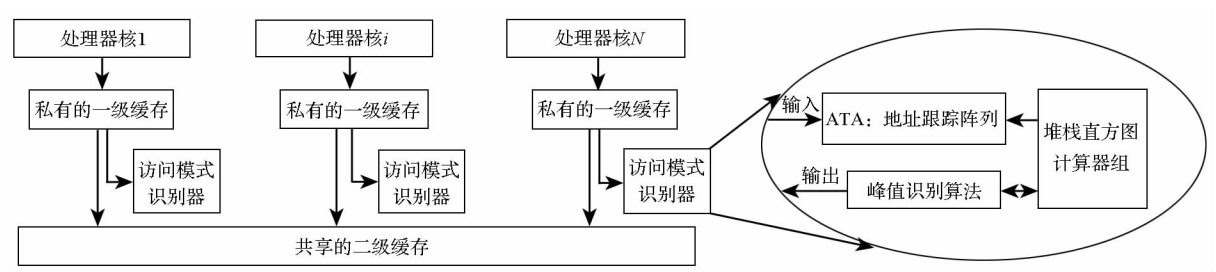


图 2　缓存请求的访问模式在线识别的架构图

Fig. 2　Online identification architecture of typical cache access patterns

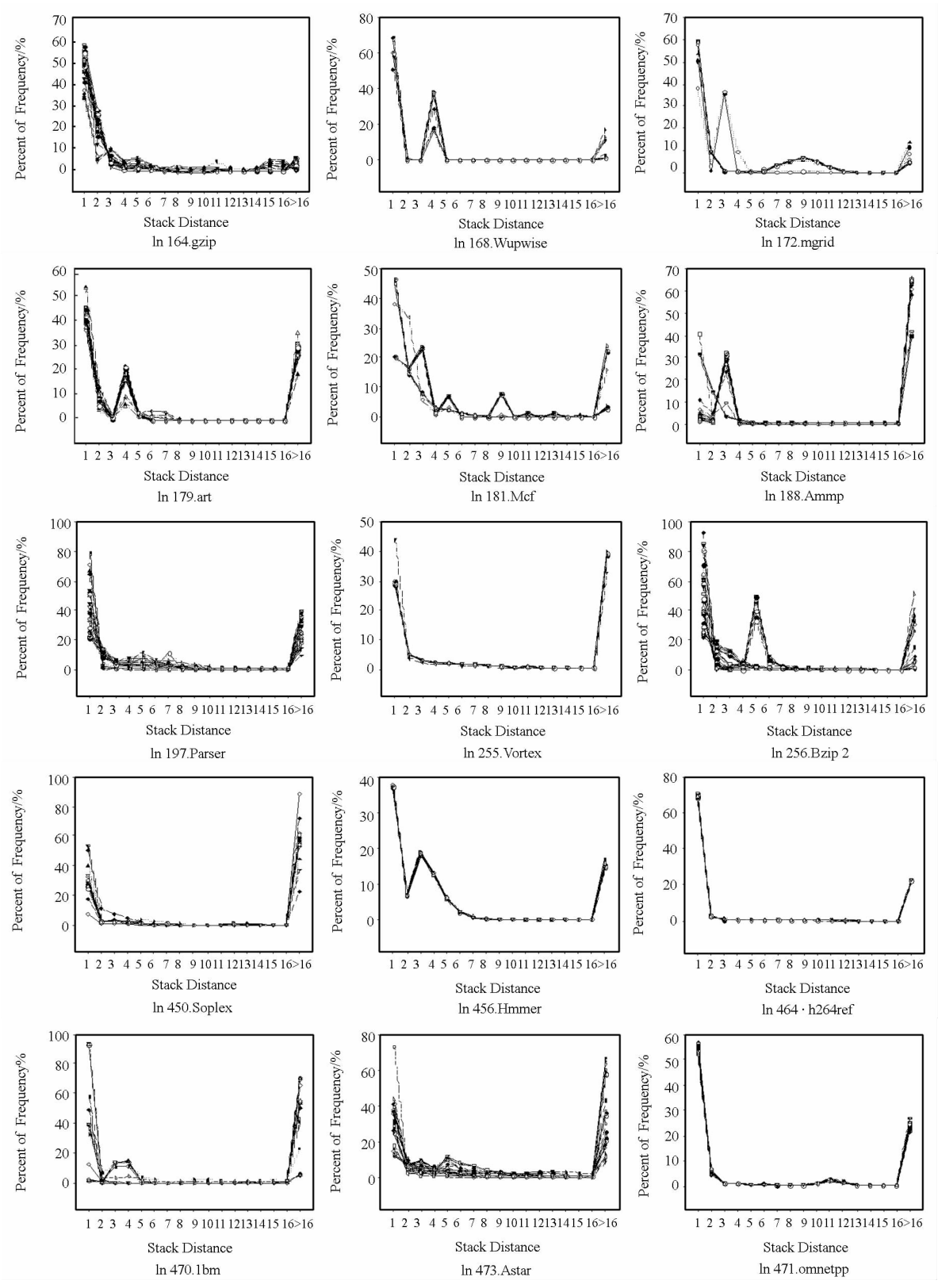


图3　基于2MB的8路共享末级缓存的15个典型程序的堆栈距离直方图(每个图包括40条采样曲线,每执行10M指令采样一次,然后统计堆栈距离为1,2,…,16以及大于16的情况)

Fig. 3　Stack distance histograms of 15 typical benchmarks in the 2MB 8-way shared last-level cache (Each chart includes 40 curves; sampling is executed in 10M instructions, then gather statistics of SDH that the stack distances are 1,2,…,16 and larger than 16)

1)访问请求序列的特征具有一定的持续性和稳定性,这为检测和预测提供了可能性。例如,对于每一个程序所有的40个采样所得出的堆栈距离直方图曲线,展现出了类似的特征,包括峰值

的位置、拐点、形状等,除了 181. Mcf, 256. Bzip2 和 470. lbm 这三个程序。但是,在这三个程序的 40 个采样的曲线中,相邻的曲线依然展示出相似的特征。因此,基于 LAST 的预测算法就可以有效地预测访问序列的特征。

2)不同的访问模式的堆栈距离频度的分布展示出鲜明的特征,例如在 164. gzip,197. Parser, 456. Hmmer,464. h264ref 和 471. omnetpp 中,重用距离为 1 的频度最大,而在 168. Wupwise,172. mgrid,179. art,181. Mcf,188. Ammp,256. Bzip2 和 470. lbm 中,重用距离为 3,4 或 5 时,频度最大。188. Ammp,255. Vortex,450. Soplex 和 473. Astar 的重用距离频度最大值发生在重用距离大于 16 时。

从上述的两个观察结果可知,在线识别访问模式是可行的。

2.3　峰值识别算法

简单的峰值识别方法假设在一个短的采样周期内,访问模式都可以近似归为三种单纯的访问模式之一。这样就可以基于对这三种单纯访问模式显著特征的判断来识别在线访问请求的特征,特别是峰值位置和曲线形状。如算法 1 所示。

针对每一个访问请求序列(来自于某一个处理器核或硬件上下文),设置一个缓冲阵列,称为请求地址跟踪阵列,使用具有堆栈属性的真 LRU 替换算法来管理,设这个阵列的大小为 T。针对这个 ATA,依据其替换堆栈,设置 $T+1$ 个计数器,设为 $CSDF_1, CSDF_2, \cdots, CSDF_T, CSDF_{T+1}$,类似于 UCP 中的堆栈距离直方图统计机制。在每一个采样周期内,针对每一个到达末级缓存的访问请求地址,在分派到末级缓存的同时,也分派到 ATA,并同时操作。在 ATA 中查找该访问请求地址是否载入过,由于 ATA 用 LRU 替换管理,依据 LRU 的堆栈属性,在 ATA 中的命中位置就是该请求地址的重用距离,这样,如果命中,就在对应的计数器 $CSDF_i$ 加 1(i 为命中时的堆栈位置)。如果没有命中,则 $CSDF_{T+1}$ 加 1。

在每一个采样周期结束时,判断 CSDF 计数器中的最大值,设为 $CSDF_m$,如果 m 等于 1,则认为在此采样周期内出现的访问请求序列是 Recency-friendly 访问模式,如果 m 等于 $T+1$,则认为是 Streaming 访问模式。如果 m 介于 1 到 T 之间,则认为是 Thrashing 访问模式。识别完成后,为了保持访问序列的历史信息,将这一个周期的访问计数器的 1/4 值作为历史数据代入到下一个采样周期。

算法 1　请求模式在线识别法之峰值识别算法

Alg. 1　Peak identification algorithm for access patterns

已知: T 为每一个 ATA 中缓存行的数量,通常可以与 LLC 缓存的关联度大小一致。R 是来自于核 i 的缓存请求。A 是请求 R 的地址。

ATA_i = 核 i 的地址跟踪阵列

$CSDF = \{CSDF_1, CSDF_2, \cdots, CSDF_T, CSDF_{T+1}\}$

```
IF 采样周期结束 THEN
BEGIN
  CSDF_m = Max{ CSDF_1, CSDF_2, ..., CSDF_T}
  IF m = 1 THEN
     判断访问模式是时间局部友好性访问模式
ELSE BEGIN
     CSDF_n = Max{ CSDF_m, CSDF_{T+1}}
     IF n < > (T+1) THEN
          判断访问模式是颠簸访问模式
     ELSE
          判断访问模式是流访问模式
  END
  重置{CSDF_i}(1 < = i < = n) 为上一个采样周期值的 1/4。
END ELSE IF 采样周期没有结束 THEN BEGIN
     p = Lookup(R, ATA_i) / * 在 ATA_i 中检索数据块 A。返回其在 LRU 堆栈中的位置。没有找到则 p = -1. */
     IF p = -1/ * 没有检索到 */ THEN BEGIN
          Select the victim line V fromATA_i
          Place the block A intoATA_i
          CSDF_{T+1} = CSDF_{T+1} + 1
     END ELSE IF p in (1,T) THEN
          CSDF_p = CSDF_p + 1
END
```

3　实验评估

基于时钟周期的全系统模拟器 Simics[11],模拟了一个 16 核的多核处理器。每一个处理器核基于 UltraSPARC IV + 的配置,具体参数见表 1。模拟器的操作系统是 Solaris10,主要的测试程序选自 SPEC CPU2000/2006。

该实验方法是在同样的缓存结构和管理机制下,依据每千条指令的缺失数(Misses per 1000 Instruction, MPKI)和缺失率曲线(Miss Rate Curve,MRC)的特征,判断该测试程序的访问请求特征,然后对基于访问模式在线识别机制得出的访问模式进行比较,看是否匹配,以验证在线访问模式识别机制的正确性。先运行 1 亿条指令消除冷缺失的影响,然后运行 10 亿条指令,统计缺失率曲线。限于篇幅,仅选择有代表性的 6 个程序

进行比较,其缺失率曲线如图4所示(横坐标/变量 C 表示缓存组数量不变时,通过缓存组的关联度的大小变化实现缓存容量的变化。纵坐标/MissRate 代表缺失率,纵坐标/MPKI 代表每千条指令的缺失量)。

表1　实验环境的基准配置

Tab.1　Baseline configuration

处理器核	16核,每核均为1个硬件线程,且顺序执行; 时钟频率1.5GHZ; 每核配置有私有的L1数据缓存和L1指令缓存,容量均为32KB,缓存行大小为64字节,均为4路,命中延迟为3个周期,采用LRU替换算法管理
共享的L2缓存	容量2MB, 缓存行大小为64字节;8路组相联,命中延迟为11个周期,采用LRU替换算法管理,采用MESI缓存一致性协议
内存	访问延迟为200个周期
测试程序	164.gzip,168.Wupwise,172.mgrid,179.art,181.Mcf,188.Ammp,197.Parser,255.Vortex,256.Bzip2,450.Soplex,456.Hmmer, 464.h264ref,470.lbm, 473.Astar,471.omnetpp

172.mgrid 在整体运行过程中,到达L2缓存的访问请求表现出了时间局部特征友好性,缓存的关联度从4以后,其缺失率就降低到0.01以下。而通过访问模式在线识别机制统计的数据(图3)可以看出,在40个采样周期中,有36个采样周期的172.mgrid展示出时间局部特征友好性的访问模式,有4个采样周期的172.mgrid展示出颠簸访问模式。由于缺失率曲线表现的特征是整个大的采样周期结果呈现出来的特征,因此没法捕捉到颠簸访问模式。访问模式在线识别方法得出的结论不仅正确,而且粒度更细更精确。

179.art 在整体运行过程中,依据缺失率曲线可以得出,其展示出明显的颠簸现象,在缓存关联度从4到8的变化过程中,缓存缺失率基本不变。在图3中,179.art 的在线识别机制统计数据也明确地展示出其颠簸访问模式的特征。访问模式在线识别方法得出的结论正确。

181.mcf 依据缺失率曲线,在缓存关联度从2到8的变化中,缺失率变化极小(小于0.02),展示出颠簸现象。在图3中,181.mcf 在线识别的数据展示出明显的颠簸访问模式特征。访问模式在线识别方法得出的结论正确。

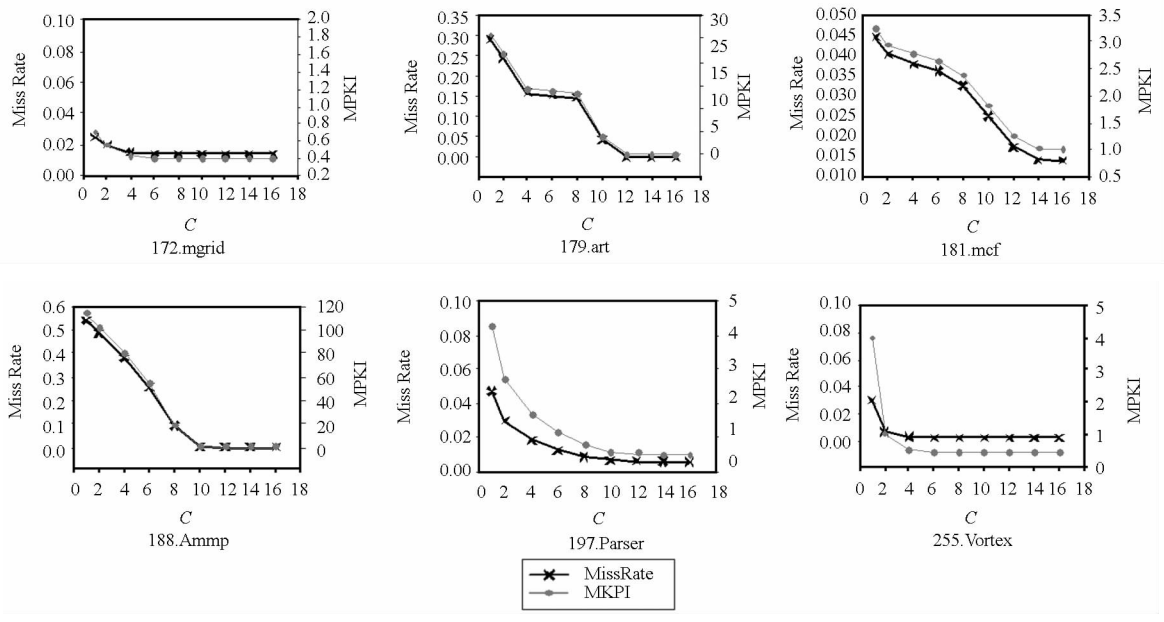


图4　测试程序的缺失率曲线

Fig.4　Miss Rate Curves(MRC)

188.Ammp,依据缺失率曲线,在关联度从1到10的变化中,缺失率明显降低,说明缓存访问过程中,缓存访问具有颠簸访问模式,这一点在图3的在线识别机制统计的数据中表现明显。访问模式在线识别方法得出的结论正确。

197.Parser 与 172.mgrid 类似,到达L2缓存的访问请求表现出了时间局部特征友好性,访问模式在线识别方法得出的结论正确。

255.Vortex,依据缺失率曲线,在关联度大于2之后,缺失率基本保持不变,展示出流访问模式特征,这一点在图3的在线识别机制统计的数据中表现明显。访问模式在线识别方法得出的结论

正确。

由此可见,访问模式在线识别方法可以有效地识别出到达 L2 缓存的各个处理器核的访问请求流的特征,即访问模式。

4　结论

本文通过分析和大量的实验,发现访问请求序列的特征具有一定的持续性和稳定性,具有检测和预测的可能性;而且不同访问模式的堆栈距离频度的分布展示出鲜明的特征,如堆栈直方图曲线的峰值、拐点、形状等。提出了一种访问模式的在线识别方法:峰值识别方法。通过 Simics 全系统模拟器的实验表明,峰值识别方法不仅正确识别了测试程序的访问模式,而且还检测出访问模式的波动情况,在细粒度上实现访问模式的识别。

参考文献(References)

[1] Misler M, Jerger N D E. Moths: mobile threads for on-chip networks[J]. ACM Transactions on Embedded Computing Systems ,2013:12(1s).

[2] Gallo M, Kauffmann B, Muscariello L, et al. Performance evaluation of the random replacement policy for networks of caches[J]. Performance Evaluation, 2014, 72: 16 - 36.

[3] Subramanian R, Smaragdakis Y, Loh G H. Adaptive caches: effective shaping of cache behavior to workloads [C]// Proceedings of the 39th Annual IEEE/ACM International Symposium on Microarchitecture, Orlando,2006: 385 - 396.

[4] Jaleel A, Hasenplaugh W, Qureshi M,et al. Adaptive insertion policies for managing shared caches[C]//Proceedings of the 17th International Conference on Parallel Architectures and Compilation Techniques, 2007: 208 - 219.

[5] Jaleel A, Theobald K B, Steely S C, et al. High performance cache replacement using re-reference interval prediction (RRIP)[C]//Proceedings of the 37th annual International Symposium on Computer Architecture,2010: 60 - 71.

[6] Chen T F, Baer J L. Effective hardware-based data prefetching for high-performance processors[J]. IEEE Transactions on Computers,1995,44(5):609 - 623.

[7] Qureshi M K, Patt Y N. Utility-based cache partitioning: a low-overhead, high-performance, runtime mechanism to partition shared caches[C]//Proceedings of the 39th Annual International Symposium on Microarchitecture (MICRO - 39), 2006: 423 - 432.

[8] Chang J C , Sohi G S. Cooperative cache partitioning for chip multiprocessors [C]//Proceedings of the 21th Annual International Conference on Supercomputing,2007: 242 - 252.

[9] Jaleel A, Hasenplaugh W, Qureshi M, et al. Adaptive insertion policies for managing shared caches[C]//Proceedings of the 17th International Conference on Parallel Architectures and Compilation Techniques, 2007: 208 - 219.

[10] 章铁飞, 陈天洲, 吴剑钟. 基于程序访存模式的低功耗存储技术. 软件学报. 2014,25(2):254 - 266.
ZHANG Tiefei, CHEN Tianzhou, WU Jianzhong. Exploiting memory access patterns of programs for energy-efficient memory system techniques[J]. Journal of Software, 2014,25(2): 254 - 266. (in Chinese)

[11] Magnusson P S, Christensson M, Eskilson J, et al. Simics: a full system simulation platform[J]. Computer, 2002,35(2): 50 - 58.