 DOI: https://doi.org/10.12349/iser.v7i6.10803

Research on Optimization of Hardware Cache Mechanisms and Improvement of Access Efficiency in Computers

Jun Chen

Xinghua Ruixing Labor Service Co., Ltd., Xinghua, Jiangsu, 225700, China

Abstract

As a critical hardware component for reducing the speed gap between the CPU and memory, the cache plays a vital role in enhancing the overall access speed of computer systems. With the advancement of multi-core processors and increasingly deep memory hierarchies, traditional cache designs can no longer meet current requirements, exhibiting shortcomings in cache hit rate, consistency assurance, and energy efficiency. This paper examines three key aspects: cache mapping mechanisms, replacement algorithms, and prefetching techniques, proposing a dynamic partitioning mapping approach, an improved history-aware replacement method, and a spatiotemporal correlation prefetching strategy. Through theoretical derivations and simulation validation, these methods demonstrate effective improvements in cache hit rate and reduction of average memory access latency, thereby contributing to enhanced system performance.

Keywords

cache; mapping mechanism; replacement algorithm; prefetching technique; access efficiency

计算机硬件高速缓存机制优化与访问效率提升研究

陈珺

兴化市瑞兴劳务服务有限公司，中国 · 江苏 兴化 225700

摘　要

作为减少CPU与内存之间速度差距的重要硬件设备，缓存对整个计算机系统的访问速度起到至关重要的作用。随着多核处理器的发展以及存储层次变得越来越深，传统的缓存设计已经不能满足目前的需求，在缓存命中率、一致性保障以及能耗方面均存在不足。本文从缓存映射方式、替换算法以及预取三个方面进行探讨，给出了基于动态分区映射的方法、改进的历史感知替换法以及时空相关预取方法。通过对这些方法进行理论推导和仿真验证，发现它们都可以有效提高缓存命中率，并且降低平均访存延迟，从而为提高计算机系统性能作出贡献。

关键词

高速缓存；映射机制；替换算法；预取技术；访问效率

1 引言

在计算机系统中，处理器运算速度与主存访问延迟之间存在着巨大且日益扩大的差距。高速缓存通过利用程序访存的空间—时间局部性来将经常被使用的数据暂存在离处理器较近的高速存储器上以解决这个问题。但是现在大多数处理器都采用多核结构以及深流水线的设计，而应用程序也各种各样并且不断变化着，在这种情况下传统的缓存设计对于满足不同的需求显得捉襟见肘，单一的映射方式不能很好地权衡灵活性和查询效率，替换策略在某些情况下的命中率会急剧下降，静态预取又会造成大量的无效带宽浪费。因此对缓存映射、替换以及预取这三个方面进行系统的研究是很有必要的，也有其重要性和紧迫性。

2 缓存映射机制的结构优化

2.1 基于访问热度动态调整的组相联映射

传统的组相联映射使用固定的相联度，高频热点的数据与低频冷门的数据在同一组中为争夺一个缓存行而发生冲突。本文提出一种可变相联度的方法，在每个缓存组内增加一个访问计数器数组来记录每组中的数据被访问次数，如果某组的冲突缺失率高于一定比例，则硬件自动从附近的空闲位置借一块作为这个组的一部分，从而使得其相联度增大，而其中的热数据有更多的选择余地，而冷数据由于仍然只有一块因此保持较低的相联度节省硬件开销。这种方法需要有快速重映射表保存这些借用关系以及用有限状态机逐步改变组大小来防止由于参数变化而导致流水线中断 [1]。

【作者简介】陈珺（1988–），女，中国江苏兴化人，本科，助理工程师，从事计算机及网络（硬件）研究。

2.2 面向相邻行预载的连续块映射策略

对于具有较强空间局部性的流式访问模式，在传统映射方式下，连续地址块分布在不同的组里，每次访问都要进行一次地址译码和一次标记比较。而连续块映射将物理地址中用于组索引的一些位重新定义为一个较大的块号，这样连续的 N 个缓存行都被映射到同一个组的不同路位置上。当第一块被放入缓存后，处理器预测接下来连续地址的命中率很高，于是就把紧接着的 N-1 行也一起加载进同一个组的空闲路中。这种方式减少了对内存控制器的多次请求需求，在顺序读取的情况下可以减少一半左右的标记比较次数。但是为了避免大的块覆盖缓存，这种方法还需要配合一个访问跨度检测器，只有在检测到连续递增的访问模式时才会发生这种映射。

2.3 非均匀多级缓存间的映射协同优化

在多级缓存系统中，一级缓存和二级缓存映射是各自独立的，存在上级缺失时，下一级缓存获取数据会造成重映射的消耗。在映射协同优化中，将所有缓存层次使用相同的索引计算函数，从而使一级缓存缺失地址在二级缓存中具有相同或者仅有一个较小偏移位置。而在实际实现中，一级缓存设置为小相联快速映射，二级缓存则具有大相联，但其索引函数是一级缓存索引函数的线性扩展。因此当发生缺失时，二级缓存查找可以利用一级缓存已经计算出索引而无需进行完全地址解析以及哈希操作。同时映射协同优化也使得上级缓存块对下一级回写能够准确找到对应位置从而降低替换选择带来的开销 [2]。

3 缓存替换算法的逻辑改进

3.1 增强型历史感知 LRU 替换策略

经典 LRU 只关注最近一次访问的时间戳，而不能分辨频繁访问与偶尔访问。改进的历史感知方法是在 LRU 的引用位链上每一块增加两个饱和计数器，来统计这一块在过去四个周期中被命中次数。在发生缺页需要淘汰某一块时，先找出那些计数值最小的候选项，然后再从这些候选项中选择一个距离 LRU 最远的那个。这样既考虑了近期又考虑了历史上的访问情况，避免了周期性冷数据对热点数据进行驱逐，特别是对于有突发访问的混合负载非常有用。

从硬件实现上看，计数器的增加与减少与缓存中一个块被访问事件相关联。当该行被命中，其计数器加一，在同组内其他行计数器基于周期性进行相应衰减操作。当饱和度达到最大值时不再递增，以防止溢出翻转影响正常工作。在不命中情况下，进入的新块初始值设置为该块的中间值，而不是零，这样可以防止新块立刻被淘汰。增加少量额外存储空间，每个块只有两个位，在当前缓存大小下是可以承受的。测试结果显示，相比于经典的 LRU 算法，可使平均缺页率下降大约 12%—18%，并且所占面积仅为整个缓存面积的一小部分，不到一半。

3.2 基于访问间隔预测的插入与提升控制

缓存替换决策不仅仅选择被淘汰项，还涉及新装入项起始放置点以及命中之后对该项优先级变化。而访问间隔预测是基于每个缓存组内所有时间戳之间的差分，构成一个队列，来估算同一地址两次访问之间的平均间隔周期数。如果间隔短，说明这个数据会很快再次被访问，那么在装入时就把它放在 LRU 链表最近使用处，并触发提升操作，使其远离被淘汰端。如果间隔长，说明这些数据不会很快被再次访问，就把它放在 LRU 链表最近使用端到最久使用端之间的某处进行降级插入。而这一过程不需要编译器帮助，自身就可以学习程序的时间局部性 [3]。

提升控制模块根据历史命中率动态改变插入点偏移量，将更多的缓存资源分配给更经常被使用的数据。即如果某组内的命中率超过一定门限，则偏移量减小，更多的行会被放置在离最近使用较近的地方；否则增加偏移量，减少低重用的数据对缓存的压力。这使得该方法能够在无需编译器反馈的情况下进行针对不同访问模式的运行时自适应。硬件开销主要是每组需要维护 4 ~ 8 个时间戳寄存器与少量的加法器、比较器等，增加约 3% 的能量消耗，在对延迟要求苛刻的嵌入式多核系统中依然能够适用。

3.3 面向多线程共享数据的感知替换策略

在多核处理器中，由于多个线程共享末级缓存而产生的数据扰动是需要解决的问题。本线程替换会移除其他线程即将使用的共享数据，造成浪费并且增加不必要的缺失以及一致性开销。感知替换在线程替换标记位的基础上增加了两个比特来表示此缓存行被多少不同的核心访问过。如果这个值为 0 或者 1，则使用之前基于历史感知的方法来进行替换。如果这个值大于等于 2，则这个缓存行为重要共享缓存行，其替换优先级大大降低。这个值是通过对总线上共享探测消息监听来进行更新的，不需要改动任何核心的 ISA.

为更好地保护共享数据，策略规定，对于共享行，其饱和计数器在命中时增加的步长大于普通行的增量，从而更不易被选作牺牲品；另外，如果系统发现某共享行在一个较长的时间段内没有被任何一个 CPU 访问，则其共享标记开始缓慢衰减，在每次经过一个固定的周期阈值后，就将其标记值减少一次，直到重新恢复正常，从而腾出空间。这种方法有效地减少了由于缓存一致性或者伪共享造成的开销。实验表明，在数据库 OLTP 以及 Web 服务等共享密集型的工作负载中，与传统的 LRU 相比，这个方法可以使得共享数据丢失率降低大约 25%，同时多核加速比提高 9%—14%。

4 预取技术的访问效能增强

4.1 基于置信度过滤的空间关联预取模型

传统的空间预取器根据一次缺页地址向前或者向后一定偏移发出预取请求，容易把很少出现的数据提前加载到缓存中，造成资源浪费。置信度过滤器为每一个预取源地址保

存一个历史命中率列表，记录该源地址发起的所有预取请求最后是否都被访问。只有置信度高于一个动态设定的门限时，才会真正地将预取请求发给内存系统处理。用两个比特位标识四种不同的置信度级别，在每次预取发生之后一段时间内如果被命中，则该源地址的置信度提升；否则降低。这样就使得预取由原来的盲目试探变成了一种有理有据的行为，在一些不太规律的应用上效果显著 [4]。

4.2 时间关联与空间关联融合的双向预取引擎

一部分访问模式既有时间上的重复又有空间上的连续，而传统的预取只考虑一个方面。双向预取有一个单独的空间单元用来监控连续的一段地址中缺少哪些步长，还有一个单独的时间单元来保存重复出现的相同的地址以及它们被访问的次数，在空间单元检测到一个固定的步长（比如加一或者加 16 字节）时，双向预取会对当前正被访问的位置之后几个连续的块以及之前几个反向块进行预取，以达到双向的空间预取效果，这对遍历数组然后再从后向前遍历这类程序有很大帮助，可以同时覆盖这两个方向；而在时间单元检测到某个地址会周期性地出现时，则会在该地址即将被第二次访问的时候进行预取从而克服空间预取对于重复访问无能为力的缺点。

两种预取流由仲裁器进行合并，以防止对同一个位置发送多个相同请求。仲裁器更倾向于选择时间单元提出的预取建议，因为在大多数情况下，它的命中率要高于空间推测的命中率。而合并方法对于那些有着大量循环读取操作的数组计算类型负载来说，可以提供超过 60% 的缺失掩盖率。硬件成本主要是两个模式探测表，每个表有 8 到 16 个条目，每个条目包含步长、置信度以及上一次访问地址。从能量消耗的角度来看，双引擎相比于只使用空间或者时间预取的成本增加约 7%，但是它带来约 18% 的性能增益，在嵌入式多核处理器上具有良好的性价比 [5]。

4.3 感知核心负载的动态预取深度调节

预取深度是指在一次预取被触发之后紧接着发出预取的块数。在预取深度太小的情况下，不能跨越访存间歇期；在预取深度太大时，会占用过多核心请求带宽。动态深度调节根据处理器执行流水线中未命中队列以及内存控制器中的读写事务队列来进行监控。如果核心缺失队列未满且内存控制器未忙，则调节器逐渐增大预取最大深度，尽可能多地获取额外的数据块；反之，如果缺失队列已近饱和或内存控制器被卡住，则调节器立即将预取深度调到最低，以保证核心产生必要请求优先得到满足。这相当于把预取看作一种可以降低的服务质量选择，而不是必须执行的操作。

这种机制还使用一个衰减值，以减少由于突然的变化导致的深度值急剧波动。该衰减值的作用是，当负载恢复正常后，使深度值缓慢上升，从而不会因为负载短暂的变化而频繁地改变预取深度。实验结果表明，动态改变预取深度，使得由于预取而带来的额外带宽开销减少大约 35%，并且对于高性能的应用程序来说，仍然能够获得较高的有效预取命中率。从硬件角度考虑，深度调节只需要一些加法器以及比较器就可以完成，而且它对于现有的预取器几乎没有影响，因此很容易被集成。在流水线读取以及随机读取混合的工作负载下，这种方法可以降低大约 9% 的平均访存延迟，而且其增加的能量开销也仅为 1.5% 左右。

5 结语

在映射机制方面，可变组相联映射可以及时根据数据热度进行调整，连续块映射可以更好地发挥空间局部性优势，多层次缓存联合映射可以减少地址变换带来的开销；在替换算法方面，主要通过历史感知 LRU 改进来提高其命中率、基于访问间隔预测的插入和提升操作、考虑多线程共享数据感知等措施，使缓存分配更加合理；在预取方面，使用置信度过滤来避免无效预取污染，时空结合双向预取能够处理更多类型的访问行为，而动态改变预取深度则是为了应对不同工作负载。以上各种手段从各个方面协作，降低平均访存延迟并提高整个系统的吞吐量。未来工作可以研究如何利用机器学习来进行自动化的缓存管理，也可以针对新的非易失性存储设备设计一种混合缓存结构。

参考文献

[1] 徐金波,董德尊,李宝峰,张伟,邢建英,张鹏.面向集合通信硬件卸载的维序触发机制和数据缓存方法[J].国防科技大学学报,2025,47(06):13-23.

[2] 张轶,张禹苓,杨学聪.面向硬件缓存结构的数据布局优化研究综述[J].集成电路与嵌入式系统,2025,25(12):40-51.

[3] 祝金良.基于深度消隐问题的三值光学计算机并行深度缓存算法研究[D].华东交通大学,2023.

[4] 程可.计算机二级与三级缓存对存储速率的影响[J].科技资讯,2019,17(11):21-22.

[5] 刘宇胜.面向高性能计算机的分布式文件系统客户端缓存优化技术研究[D].国防科技大学,2018.