众核处理器 cache 一致性研究综述*

韩立敏，安建峰†，高德远，樊晓桠，任向隆
（西北工业大学 计算机学院，西安 710129）

摘 要： 以瓦片结构众核处理器一致性协议的设计为主线，综述了国内外近年来关于众核处理器 cache 一致性的相关研究；介绍了不同 NUCA 结构对一致性协议的影响；分析和对比了几种传统目录一致性协议的特性及其存在的问题；归纳了最新几个面向众核结构一致性协议的设计思想和特性。最后为设计具备应用程序适应性和可扩展性的 cache 一致性协议指出了几个关键的设计方向。

关键词： cache 一致性协议；众核处理器；瓦片化结构；NUCA
中图分类号： TP302 **文献标志码：** A **文章编号：** 1001-3695(2012)11-4011-06
doi:10.3969/j.issn.1001-3695.2012.11.003

Review on cache coherence for many-core CMPs

HAN Li-min, AN Jian-feng†, GAO De-yuan, FAN Xiao-ya, REN Xiang-long
(*School of Computer Science, Northwestern Polytechnical University, Xi'an* 710129, *China*)

Abstract: This paper took tiled many-core processor coherence protocol design as masterstroke, summed up many-core processor cache coherence related research work. It enumerated the influence of NUCA on cache coherence protocol schemes, explored the features and drawback in conventional directory-based coherence protocols, and exacted the characteristics of several novel cache coherence protocols oriented to many-core architecture. At last, It pointed out several design directions for a scalable and workloads adaptable coherence mechanisms adopted in many-core CMPs.

Key words: cache coherence protocol; many-core processor; tiled structure; NUCA

0 引言

技术的发展驱使单个芯片上集成的处理器核个数越来越多，各大半导体公司于 2006 年之后纷纷推出其众核处理器产品。2006 年，IBM 公司推出其 1025 核心的 Kilocore 众核处理器；Tilera 公司于 2007 年发布了其 64 核心的 TILE64，于 2009 年推出最新的 100 核心的 TILE-Gx100 众核处理器；Intel 公司于 2008 年披露了其 80 核心的 POLARIS 原型；ClearSpeed 公司在 2008 年推出其 192 核心的 CSX700 处理器。即便如此，众核处理器的很多设计问题至今依然没有得到有效解决。由于众核处理器的处理器核心数目巨大，维护处理器核心之间的数据一致性呈现出新的需求，cache 一致性协议就成为亟待解决的关键设计问题之一。以下是众核处理器系统特有的技术参数限制和负载特性：

a) 互联结构发生变革。共享总线和交叉开关这种有序连接结构不再适合大规模众核。无序的、点对点的网络连接结构因具备可扩展性而被作为未来众核处理器的片上连接结构。

b) 处理器核数目众多。基于点对点连接网络形成的瓦片结构(tiled)将是众核处理器结构的理想选择，而瓦片结构的众核处理器呈现出新的特征：核间通信延迟随着核数目增多急剧增大；维护一致性的硬件逻辑随着处理器核数目呈线性增长，一致性消息量剧增。

c) 应用的差异性。云计算和服务器应用作为众核处理器系统的主要负载，均具有异构性和不均衡性。这种特性导致单个一致性协议无法让所有程序取得高性能。

另外，在传统的目录协议中，cache 行以交叉方式分布在各个处理器节点中，这种交叉方式方便确定数据的宿主节点，但是会引入大量的 cache 一致性事务。由于在执行一致性事务之前目录协议需要获取每个存储块的共享状态，这就导致一系列的问题[1]：

a) 因目录而产生的间接访问增加了生产者和消费者之间的 cache 缺失访问延迟。很多情形下，为了得到宿主节点的信息实施的目录查询操作处于 cache 访问的关键路径上。

b) 自动地将共享只读类型的数据复制到很多个处理器核的本地 cache 中，减少了片上有效 cache 容量，导致 cache 缺失率上升。

c) 对共享数据的写操作，或者片上目录项的替换淘汰需要将所有被共享数据的副本置为无效，增加了 cache 缺失率，增加了数据缺失处理的代价，降低了协议执行效率。

根据以上分析，现存的传统多核处理器一致性协议因为性

收稿日期： 2012-05-02；**修回日期：** 2012-06-10 **基金项目：** 国家自然科学基金资助项目(61173047,61003037,60736012)；国家“863”计划资助项目(2009AA01Z110)；西北工业大学基础研究基金资助项目(JC201212)

作者简介： 韩立敏(1983-)，女，陕西扶风人，博士研究生，主要研究方向为计算机体系结构、存储系统；安建峰(1977-)，男(通信作者)，讲师，博士，主要研究方向为计算机体系结构、数字系统设计(anjf@nwpu.edu.cn)；高德远(1946-)，男，教授，博导，主要研究方向为高性能处理器体系结构、VLSI 设计及计算机网络；樊晓桠(1962-)，教授，博导，主要研究方向为高性能处理器体系结构、VLSI 设计及计算机网络；任向隆(1982-)，男，博士研究生，主要研究方向为计算机体系结构、片上网络.

能、面积、功耗等不可扩展的原因已不适用于众核处理器，如用于 Intel Xeon [2] 和 Tilera TILE64 [3] 的一致性协议就仅适用于少数几个处理器核的情形，设计人员必须使用新的办法解决 cache 一致性问题。最近 10 年来，很多人致力于可扩展的 cache 一致性协议相关研究工作。2008 年，匹兹堡大学的 Fensch 等人 [4] 从软件管理的角度研究了瓦片式多核处理器的一致性策略；2009 年，麻省理工学院的 Celio 为了研究众核 cache 一致性协议，开发了一款支持 256 核的众核仿真器 Graphite [5]，他们认为在众核中实现硬件 cache 一致性是可行的，但是为了得到最优的性能，软件开发者需要谨慎地编写代码，确保软件算法和目标硬件架构相匹配；同年，Intel 公司的 Dubey、Zhou 等人 [6,7] 认为一些面向众核处理器的并行应用算法，如 RMS，具有较少的数据共享，因此他们基于实验芯片 SCC（single-chip cloud computer）和一款 32-core 服务器初步探索了纯软件管理的一致性协议的原型系统。Kelm 等人 [8] 提出一种软件硬件协同控制的 Cohesion 一致性策略，Cohesion 结构在细粒度上实现一致性，当使用软件控制，Cohesion 无须片上目录，减少了很多一致性消息。

国内多家研究机构也在近几年做了不少相关研究。针对大规模多核处理器，为了减少基于目录的一致性协议中访问远程目录存储器的平均访问延时，清华大学信息科学与技术国家实验室的郭松柳、王海霞等人于 2009 年提出面向 CMP 结构的层次结构 cache 目录 [9]。面向片上众核处理器方面，中国科学院计算技术研究所的黄河等人 [10] 针对目录一致性协议存在难于实现、验证复杂和存储空间开销大等问题，提出一种由硬件结构支持、基于同步的高速缓存一致性协议。该方案不使用目录，而是通过布龙滤波器（Bloom-filter）表示一致性信息，在并行程序的同步点维护高速缓存一致性和解决数据冲突。中国科学院计算技术研究所的徐卫志等人 [11] 在 2010 年研究了与其 Godson-T 众核处理器一致性协议的实现密切相关的问题——同步机制，他们认为硬件支持的同步机制性能很重要，而且专用的同步机制的扩展性好；同年，基于 64 核心的 Godson-T 众核处理器；中国科学院计算技术研究所的范东睿等人为了提高对共享数据的读取效率，使用软件对共享数据实现访问控制，维护一致性 [12]。

本文主要探讨适合众核处理器的 cache 一致性协议的设计策略，介绍基于 NoC 互连的瓦片式众核处理器结构的相关特性和与一致性协议策略相关的 NUCA 结构，对比和总结各种传统的目录一致性协议，分析和对比几个用于瓦片式众核处理器的 cache 一致性协议的特性。

1 基于 NoC 的瓦片结构众核处理器

传统互连结构，如共享总线的面积会随着处理器核心数目的增加而增长过快，导致其无法用于众核处理器。受物理设计和生产工艺的限制，建议未来处理器采用点对点的互联网络连接的瓦片结构。一方面，点对点的互联网络的峰值带宽和面积开销不随着处理器核心数目的增多而增长；另一方面，瓦片式的 CMP 容易与其不同核心数目的家族产品兼容。如图 1 所示，瓦片结构 CMP 是由相同或相似的构建块（tile）的整列组成。在这样的结构中，每个构建块由一个处理器核心、一级 cache（指令 cache 和数据 cache）、末级 cache 的局部瓦片（L2 slices）以及点对点的互联网络的接口组成 [13~16]。

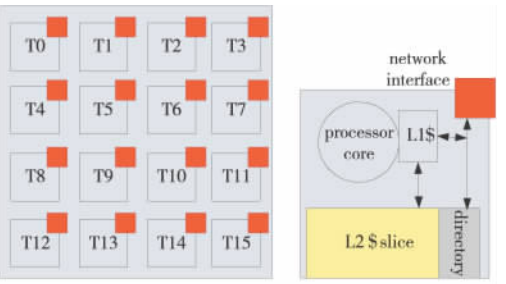

图1 瓦片化的众核处理器结构

2 NUCA 结构对一致性协议策略的影响

集中式的 cache 结构（NUA）存在两个问题：长的线延迟导致长延时的存储访问；集中式访问会造成严重的大量的存储访问冲突。针对这些问题，人们提出了 NUCA 结构，它是一种混合的私有/共享 cache 组织 [17]。另外，随着片上处理器核心的增多，片上总的存储器容量急剧增大，管理这些瓦片内的 cache 容量及其一致性呈现出新的需求。只有当 cache 的组织、目录存储器、存储器控制器和一致性引擎都是分布式的才有利于数据一致性的实现，而这样的系统才是可扩展的。基于以上两点，NUCA 结构被作为众核处理器的片上存储结构已经成为共识 [18~20]。

任何 cache 一致性问题都是源于 cache 的组织结构和使用方式，本章按照三种分类方法（数据在存储器中的布局策略、片上存储器的管理方式、cache 的组织方式）来分析现存的 NUCA 对一致性策略的影响。

2.1 静态和动态的 NUCA

根据数据映射和布局策略，可以将 NUCA 划分为静态 NUCA 和动态 NUCA。数据在 NUCA 中的布局决定了远程 cache 访问的频率和访问时间。静态 NUCA 采用简单的数据映射策略，这会导致物理地址连续的数据块分布在不同的存储体中，增加了远程片上存储访问的数量。处理器核数目越大，长访问延时效应越明显 [21]。当 NUCA 被静态地组织成处理器核的私有部分时，每个处理器核所连接的私有部分也可以被其他核所访问；当本地 cache 瓦片被填满，则寻找其他 core 上的 cache 瓦片填充数据。因为无法控制哪些分段被哪些处理器核所共享，这样必然导致“污染”。当处理器核数目多于 8 个时，远程访问增多，维护 cache 一致性的代价明显增大，限制了其可扩展性；动态 NUCA 则依赖于系统软件的支持实现优化的数据布局，提高了数据局部性。例如，可以通过操作系统的虚拟存储器机制，根据第一次接触原则，控制存储器到处理器核的动态的数据布局，将数据映射到线程所在处理器核的本地 cache，充分利用到数据访问的局部性，而这样的布局也有利于用一致性协议的实现。

2.2 共享模式和私有模式的 NUCA

根据片上存储的管理方式对 NUCA 分类，不同共享方式的 NUCA 结构对一致性协议会产生重大影响。从通信效率角度看，非包含式的共享 cache 会招致间接通信（即数据请求者、目录、共享者之间的通信）以及协议设计和验证的复杂性 [22,23]；

采用包含性策略的末级共享 cache 中的标记存储器可以被用来存储被共享数据的目录信息,简化了分布式目录一致性协议的实现。对于私有 NUCA,由于需要多个副本在处理器核的 L1 级 cache 中,一致性消息量大,导致一致性协议通信效率低。从实现一致性的面积开销角度看,对于瓦片结构中的末级私有 cache,实现一致性需要较大的复杂性和存储空间。例如,一个 16 核,cache 块大小是 64 Byte、每个 L2 瓦片是 1 MB 的 CMP,每个瓦片需要 288×10^3 个目录项来实现全映射的目录一致性。对于瓦片结构末级共享 cache,因为每个块在整个 L2 cache 仅仅有一个唯一的位置,所以实现一个全映射的目录每个瓦片仅需要 152 KB。

2.3　层次化结构与平坦结构的 NUCA

如果瓦片结构的众核处理器将处理器核分组打包,形成多个簇集(cluster),这些簇集共享一个中间结构的 cache,则形成一个局部共享的层次化存储结构。例如 Sun/Oracle 的 T2 系统、NVIDIA 的 Fermi GPU 和 AMD 未来的 Bulldozer 架构。在这种 cache 的组织方式中,通过调整末级 cache 在簇集中的共享度,可以实现不同于平坦结构(flat structure)的目录。在这样的系统中,处理器通过访问簇集内部的 cache,将私有 L0 cache 和共享的 L1 cache 以及分布式共享 cache 之间的距离缩小了,达到快速共享数据的目的。另外,如图 2 所示,共享度为 16 的 64 核瓦片结构的 CMP,虚线将共享末级 cache 的 4 个簇集分割开。与 Sun/Oracle 的 T2 系统中的静态的映射办法不同,这款众核处理器不是将处理器核组合成簇集,而是将 LLC 的 bank 作为邻接核的部分共享的存储空间,这样每个 core 只需访问不同组的 LLC bank [24]。

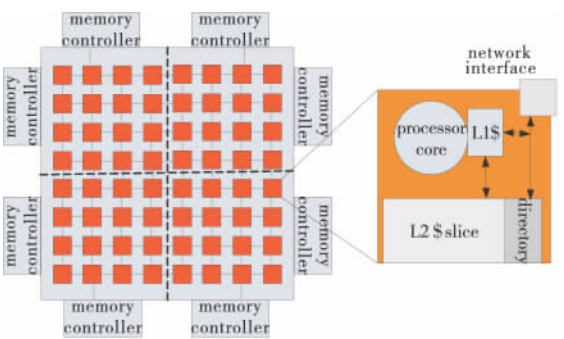

图2　共享度为16的64核瓦片结构的CMP

3　传统一致性协议的目录结构

对于共享存储器的众核 CMP 系统,基于目录的一致性协议已经是事实上的标准。目录被用来记录和跟踪共享者,一般使用向量记录和标志“共享者”。一致性协议根据这些信息仅仅给共享数据的那些节点(处理器核)发送一致性消息,这些信息起到过滤无用消息、减少片上网络通信量的作用。例如,基于 MOESI 的目录一致性协议为每个 cache 块的目录信息增加了数据“所有者”字段,这个“所有者”字段协助一致性协议检测最新数据拥有者,数据的访问请求只需要转发到“所有者”处理器核。在众核环境下,传统目录结构引入的存储开销以及高功耗随着处理器核的数目增多而呈线性或几何增长,这给众核设计实现带来阻碍,限制了一致性协议的可扩展性。基于瓦片的 CMP 结构,下文分析了各种基于目录的 cache 一致性策略的目录结构的特点和不足之处。

3.1　全映射目录

全映射目录(full-map directories)是一种直接、简单的目录实现策略,早期被使用在多机环境下。它使用一个 N 位向量保存共享者信息(N 是系统中处理器的数目),还设置一个修改位,用以指示是否某个处理器对这个数据执行了写操作。它的特点是系统需要同时保存全局存储器中每个 cache 块的共享状态[25]。在实际中,不是所有的存储器块同时出现在片上存储器中,所以全映射目录由于存储大量冗余信息而需很高的存储代价,限制了其在众核中实现的可行性。

3.2　基于复制标记字段的目录

如图 3 所示,基于复制标记字段(duplicate-tag-based directories 或 shadow-tagged directory)的目录一致性协议使用一种专用、集中式的片上存储器实现目录[26~28]。通过检查 cache 块的标记字段来确定这个块的共享状态,这种目录的相联度等于 cache 相联度与 cache 个数的乘积[29~31],这样就产生了一个大相联存储器。随着处理器核数目的增多,这种目录的相联度呈线性增长,访问相联结构的目录带来的高功耗阻止了这种目录的扩展性。

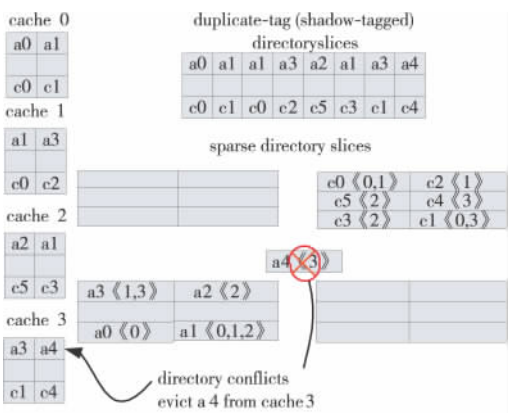

图3　基于复制标记字段的目录一致性协议

3.3　in-Cache 目录

为了节省了目录的标记存储空间,将末级共享 cache 中每个 cache 行的标记字段进行了扩展,添加共享指针和 MESI 状态则可以实现 in-Cache 目录。这是一个 16 核、L2 cache 共享的 CMP 结构,其目录和共享的 L2 cache 都是分布式的,每个目录项对应一个 L2 cache 块,具有一个共享者区域和状态区域,与全映射的目录结构一样。实际应用于 SGI Origin2000 的多处理器系统[32] 和 TILE64[33,34] 均是一个由 8×8 的瓦片矩阵组成的 64 核心的众核处理器。虽然 in-Cache 目录比全映射目录在面积方面确实进步了,即只保存片上 cache 行的共享信息,但其向量的位数等于处理器核的位数,所以它还是具有局限性。只有当 in-Cache 目录在位向量的尺寸保持在合理范围才是可行的。例如,当处理器核数目超过 128 核时,in-Cache 目录就失去了其优势,其位向量存储需求变成了主导,限制了其扩展性。另外,实现 in-Cache 目录还具有一些限制,即末级 cache 被共享,所有的处理器核的多级 cache 之间都实现包含性策略。

3.4　稀疏目录

稀疏目录(sparse 目录)也拥有与全映射目录相同的逻辑

段，也使用组相联的目录结构，每个目录项包含 cache 块的标记[4]。不同之处在于，稀疏目录减少了相联度且仅仅存储片上 cache 的共享信息，所以它克服了功耗障碍。然而，稀疏目录会遭受组间冲突，会淘汰一些原本正在使用的 cache 块的目录信息。为了减少这种布局限制带来的位置冲突效应，Sparse-full 为每个片上存储的 cache 块至少设置一个目录项[14]。如此一来，对于 N 核心的众核处理器，Sparsefull 策略的目录面积将以 $O(N^2)$ 增长，以 256 核的众核处理器为例，所有处理器核的基于向量的 L1 目录总存储容量会消耗掉 256 MB 的片上存储，超过了其 L2 cache 的总容量，所以稀疏目录对于大型的 CMP 不适用。

4 面向众核的一致性协议

为了解决传统目录一致性在可扩展方面存在的种种缺憾，在过去几年里，研究人员提出各种改进的目录一致性协议和新的一致性的策略。下文分析了最新出现的几种面向众核处理器一致性协议的设计思想和特性。

4.1 消除间接访问的目录一致性协议

德国穆尔西亚大学的 Ros 等人为众核处理器设计了直接协议(direct coherence protocol)。他们根据数据块的存储访问模式，将数据分为私有数据、只读的共享数据和可读—可写的共享数据。与传统的目录协议不同，它将维护共享者信息和对访问定序的责任交付给数据的所有者节点，而不是宿主节点。对于一个 16 核、8 路的 CMP，这个协议使用一个处理器核可寻址的 8 路相连、1 KB 存储项的预测器来保存数据当前的“所有者”信息。本协议只允许可读可写的共享数据访问目录，通过重新分配维护 cache 一致性的角色来消除了大量不必要的核间通信和对目录的间接访问[35]；另外，犹他大学的 Pugsley 等人提出一种目录协议的变种，他们将数据和一致性状态信息分离存放，将数据一致性信息的所有权以及维护一致性的义务转移到访问节点，一般为写操作节点[15]。本协议仅需执行两步就可以完成置无效操作，而传统目录一致性协议需要三或四步才能完成此项操作。

4.2 目录可扩展的一致性协议

很多人通过减小目录项的尺寸来减少目录的面积开销。基于布龙滤波器(Bloom filter)，多伦多大学的 Zebchuk 等人[26]提出的无标记一致性的目录(tagless coherence directory)结构，使用一种隐蔽方式的表示共享信息的目录结构。图 4 是一个滤波器网格，每一列代表每个处理器核，每一行代表一个处理器核的所有 cache 组。对于一个 16 核的 CMP，此方法能够减少大约一半的目录存储面积。本研究的分析模型认为此策略的面积开销不随核的数目而增长，可扩展性好。但是本方法的缺点是滤波器的操作相当复杂。

另一种目录面积优化的策略是层次化的目录。在图 2 中，基于 cache 的层次化组织实现层次化一致性协议的目录。为了维护一致性，每次数据从一个共享层传递到一个低级私有层时共享级都需要记录私有副本，总体开销由需要被跟踪的 cache 容量和共享程度来决定。具有两级一致性信息并不意味着会增加目录的开销。实际上，层次化目录设计与只有一级(平坦结构，非层次级)完全私有 LLC 以及完全共享的 LLC 相比较，使用两级目录可以减少存储开销。具体地讲，对于一致性实现具有以下优势：

a) 层次化的结构仅需要少量的置无效操作数目和确认消息的数目。例如，对于一个被共享的 cache 块，末级 cache 无须为每个处理器核发布置无效消息，而是仅仅为每个 cluster 发布置无效消息，这样片间通信量大大减少，同时也避免了顺序注入成千上万的无效信息，之后又顺序处理这么多的确认信息。

b) 对于维护包含性的存储系统，一致性的存储开销大大减少。对于一个 N 核的瓦片化的 CMP，如果 N 核处理器中的 L2 是局部共享，假设其共享度是 M，那么需要 M 位向量跟踪 cluster 内部的 M 个 L1 cache，需要 N/M 位的向量跟踪这 N/M 个 L2 存储体之间的私有副本。因为私有 cache 的容量总和一般比共享末级 cache 小很多倍，所以层次一致性协议目录的面积开销会小很多。根据实验数据统计的结果，当共享度为 N 开方的 2 倍时，一致性目录的面积开销最小。

c) 采用层次化之后，访问延迟不会随着处理器核数目的增加而线性增大。

虽然具有以上优势，但这种层次化的设计也存在一些缺点，如引入额外的复杂性和额外的 cache 查询层次。

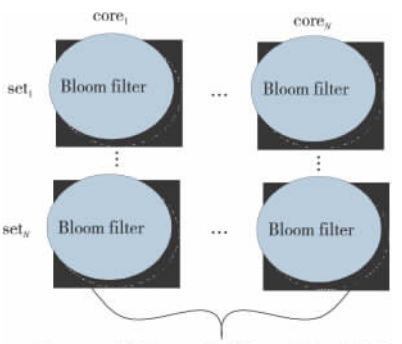

图4 无标记的一致性目录

4.3 访问时间优化的 cache 一致性

如图 5 所示，为了加速片上通信，匹兹堡大学的 Lee 等人[21]为动态共享 NUCA 的结构提出一种基于目录、有限广播的一致性协议。为了充分利用数据局部性，本文采用基于距离感知的数据布局策略将私有数据优先映射到线程所在核的 L2 cache 瓦片，如果私有数据工作集大于这个容量，则按照距离远近，将其分配在线程所在核周边其他处理器核所在的 L2 cache 瓦片中。对于共享数据，本策略启用基于目录的一致性协议；对于私有数据，因为无须维护一致性，本协议启用有限广播快速找到所需数据。这里的目录采用全分布的组织方式，目录项的数目是所有分布式 L2 cache 数据块的总和，相联度为其 L2 cache 的两倍。

4.4 减少通信量的一致性协议

犹他大学的 Pugsley 等人[36]提出 SWEL，这是一个无目录的一致性协议。SWEL 不使用目录，而使用一种小的簿记结构跟踪私有、共享只读、共享可读—可写 cache 行的状态。允许只读数据在 L1 cache 中被复制，可读可写数据只允许出现在 L2 cache 中。因为可以消除大量的一致性操作，当应用程序的

可读可写数据较多时,SWEL 比基于目录的协议性能高出很多。

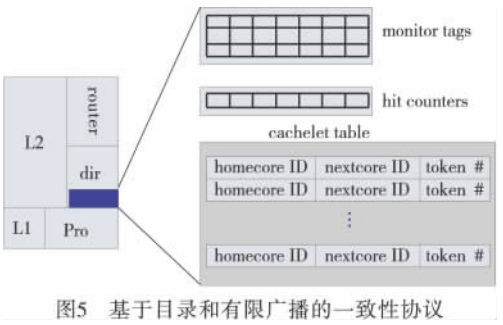

图5　基于目录和有限广播的一致性协议

4.5　负载自适应的一致性协议

利用应用程序中私有数据集和共享数据集差异性较大这个现实,麻省理工学院的 Khan 等人提出了体系结构级冗余的一致性协议 ARCc,能达到对各种应用程序具备适应性的目的。目录协议适用于私有数据比较多的一类应用。为了利用应用程序的局部性,允许数据在各个私有 cache 中被缓存的众核处理器中,用基于目录的协议来维护一致性; 共享-NUCA 结构则无须一致性协议,它适合共享数据多的应用,通过不允许可读可写数据在私有 cache 中复制维护数据的一致性,提高了片上存储的利用率,减少片外访问存储器。ARCc 一致性协议对于共享数据多的应用,启用无目录一致性协议; 对于私有数据多的应用,启用传统的目录一致性协议,实现了对具有不均衡需求应用的适应性[1]。

5　结束语

基于目录的一致性协议已经成为众核处理器事实上的标准,但是随着处理器核数目的增大存在一系列问题。应用于多核的传统一致性协议在众核处理器中不再适用,新的需求和限制因素要求人们设计具备通信局部性、减少一致性事务、减少存储代价的 cache 一致性协议。硬件 cache 一致性协议因为简化了编写并行程序的负担,与原有多处理环境兼容,依然占据很重要位置。分析和总结近十年的研究,未来适合众核处理器的一致性协议可以按照以下几点来设计:

a) 根据数据的存储访问模式设计一致性协议。利用私有数据、只读数据无须访问目录这一事实,将其与共享可读可写数据分离,不为其分配目录项,节省了面积开销,而且减少了冗余的一致性消息。数据访问模式的识别和分类机制将是研究的重点。为了对数据实现分类,可以利用 OS 来提取页表信息,根据数据所在页的属性得到数据的共享模式[14]; 也可以在运行时通过搜集统计信息,使用硬件预测机制预测共享模式[36,37]; 还可以利用编译器实现这个目的[38]。

b) 采用面积优化的数据结构存储目录信息。例如,为了减少目录的面积开销,采用层次化目录以及显式的替换操作通知等设计思想[24]。这样片上通信量、访问延迟、目录面积等参数都不会随着处理器核数目的增加而增加,使得硬件一致性协议在众核处理器中可扩展。

c) 启用优化的数据布局策略。根据各种数据的共享特性,制定对应的数据布局策略。例如,使私有数据尽量靠近线程宿主核,减少远程 cache 的访问和数据搬移操作,利用硬件性能计数器收集数据访问信息作为 OS 的宿主优化策略的输入,为后续程序提供优化的 L2 数据布局,让共享数据尽量均匀分布在各个处理器的 L2 瓦片中[37]。

d) 利用应用负载的异构性和不均衡性,设计具有适应性的一致性协议。例如,由于服务器负载没有实现一致性的需求,应当避免设置复杂的 L2 硬件一致性机制; 私有数据应当被放在请求核的本地 L2 cache,指令类型的数据可以被复制多份; 共享数据应当按照地址交错实现均匀布局,以求公平访问。

参考文献:

[1] KHAN O, HOFFMANN H, LIS M, *et al*. ARCc: a case for an architecturally redundant cache-coherence architecture for large multicores [C]//Proc of the 29th IEEE International Conference on Computer Design. Washington DC: IEEE Computer Society, 2011: 411-418.

[2] CHAIKEN D, FIELDS C, KURIHARA K, *et al*. Directory-based cache coherence in large-scale multiprocessors [J]. Computer, 1990, 23(6): 49-58.

[3] Tilera Corporation. TILE64 processor product brief [R/OL]. (2008-2009). http://www.tilera.com/sites/default/files/productbriefs/PB010_TILE64_Processor_A_v4.pdf.

[4] FENSCH C, CINTRA M. An OS-based alternative to full hardware coherence on tiled CMPs [C]//Proc of the 14th International Symposium on High Performance Computer Architecture. 2008: 355-366.

[5] CELIO C P. Cache coherence strategies in a many-core processor [D]. Cambridge: Massachusetts Institute of Technology, 2009.

[6] DUBEY P. Recognition, mining and synthesis moves computers to the era of tera [R]. [S.l.]: Intel Technology@ Corporation, 2005.

[7] ZHOU Xiao-cheng, CHEN Hu, LUO Sai, *et al*. A case for software managed coherence in many-core processors [C]//Proc of the 2nd USENIX Workshop on Hot Topics in Parallelism. 2010.

[8] KELM J H, JOHNSON D R, TUOHY W, *et al*. Cohesion: a hybrid memory model for accelerators [C]//Proc of the 37th International Symposium on Computer Architecture. New York: ACM, 2010: 429-440.

[9] GUO S, WANG H X, XUE Y B, *et al*. Hierarchical cache directory for CMP [J]. Journal of Computer Science and Technology, 2010, 25(2): 246-256.

[10] 黄河,刘磊,宋风龙,等. 硬件结构支持的基于同步的高速缓存一致性协议[J]. 计算机学报,2009,32(8): 1618-1630.

[11] 徐卫志,宋风龙,刘志勇,等. 众核处理器片上同步机制和评估方法研究[J]. 计算机学报,2010,33(10): 1777-1787.

[12] 包尔固德,李伟生,范东睿,等. Godson-T 众核体系结构上的 Broadcast 性能优化[J]. 计算机研究与发展,2010,47(3): 524-531.

[13] ROS A, ACACIO M E, GARCÍ J M: DiCo-CMP: efficient cache coherency in tiled CMP architectures [C]//Proc of IEEE International Symposium on Parallel and Distributed Processing. 2008: 1-11.

[14] HARDAVELLAS N, FERDMAN M, FALSAFI B, *et al*. Reactive NUCA: near-optimal block placement and replication in distributed caches [C]//Proc of the 36th Annual International Symposium on Computer Architecture. New York: ACM, 2009: 184-195.

[15] HOSSAIN H, DWARKADAS S, HUANG M C. DDCache: decoupled and delegable cache data and metadata [C]//Proc of the 18th International Symposium on Parallel Architectures and Compilation

Techniques. Washington DC: IEEE Computer Society, 2009: 227-236.

[16] CHO S, JIN Lei. Managing distributed, shared L2 caches through OS-level page allocation [C]//Proc of the 39th Annual IEEE/ACM International Symposium on Microarchitecture. Washington DC: IEEE Computer Society, 2006: 455-468.

[17] CHANG Ji-chuan, SOHI G S. Cooperative caching for chip multiprocessors [C]//Proc of the 33rd Annual International Symposium on Computer Architecture. Washington DC: IEEE Computer Society, 2006: 264-276.

[18] KIM C, BURGER D, KECKLER S W. An adaptative, non-uniform cache structure for wire-delay dominated on-chip caches [C]//Proc of the 10th International Conference on Architectural Support for Programming Languages and Operating Systems. New York: ACM, 2002: 211-222.

[19] LIRA J, MOLINA C, GONZALEZ A. Analysis of non-uniform cache architecture policies for chip-multiprocessors using the parsec benchmark suite [C]//Proc of the 2nd Workshop on Managed Multi-Core Systems. 2009: 1-8.

[20] HERRERO E, GONZÁLEZ J, CANA R. Elastic cooperative caching: An autonomous dynamically adaptive memory hierarchy for chip multiprocessors [C]//Proc of the 37th Annual International Symposium on Computer Architecture. New York: ACM, 2010: 419-428.

[21] LEE H, CHO S, CHILDERS B R. CloudCache: expanding and shrinking private caches [C]//Proc of the 17th International Symposium on High Performance Computer Architecture. Washington DC: IEEE Computer Society, 2011: 219-230.

[22] MENG Jia-yuan, SKADRON K. Avoiding cache thrashing due to private data placement in last-level cache for manycore scaling [C]//Proc of the 27th International Conference on Computer Design. Washington DC: IEEE Computer Society, 2009: 282-288.

[23] LU Qing-da, ALIAS C, BONDHUGULA U, *et al*. Data layout transformation for enhancing data locality on NUCA chip multiprocessors [C]//Proc of the 18th International Conference on Parallel Architectures and Compilation Techniques. Washington DC: IEEE Computer Society, 2009: 348-357.

[24] GARCÍA-GUIRADO A, FERNÁNDEZ-PASCUAL R, ROS A, *et al*. DAPSCO: distance-aware partially shared cache organization [J]. ACM Trans on Architecture and Code Optimization, 2011, 8(4): 25(1-19).

[25] CHAIKEN D, FIELDS C, KURIHARA K, *et al*. Directory-based cache coherence in large-scale multiprocessors [J]. Computer, 1990, 23(6): 49-58.

[26] ZEBCHUK J, SRINIVASAN V, QURESHI M K, *et al*. A tagless coherence directory [C]//Proc of the 42nd International Symposium on Microarchitecture. New York: ACM, 2009: 423-434.

[27] BARROSO L A, GHARACHORLOO K, MCNAMARA R, *et al*. Piranha: a scalable architecture based on single-chip multiprocessing [C]//Proc of the 27th Annual International Symposium on Computer Architecture. New York: ACM, 2000: 282-293.

[28] GOLLA R. Niagara2: a highly threaded server-on-a-chip [C]//Proc of the 18th IEEE HotChips Symposium on High-Performance Chips. 2006.

[29] MARTU M R, HILL M D. Virtual hierarchies to support server consolidation [C]//Proc of the 35th Annual International Symposium on Computer Architecture. New York: ACM, 2007: 46-56.

[30] FERDMAN M, LOTFI-KAMRAN P, BALET K, *et al*. Cuckoo directory: a scalable directory for many-core systems [C]//Proc of the 17th International Symposium on High Performance Computer Architecture. New York: ACM, 2011: 169-180.

[31] BAER J, WANG W. On the inclusion properties for multi-level cache hierarchies [C]//Proc of the 15th Annual International Symposium on Computer Architecture. New York: ACM, 1988: 345-352.

[32] LEE H, CHO S, CHILDERS B R. Perfectory: a fault-tolerant directory memory architecture [J]. IEEE Trans on Computers, 2010, 59(5): 638-650.

[33] WENTZLAFF D, GRIFFIN P, HOFFMANN H, *et al*. On-chip interconnection architecture of the tile processor [J]. IEEE Micro, 2007, 27(5): 15-31.

[34] BROWN E. 64-way chip gains Linux IDE, dev cards, design wins [EB/OL]. (2008-04-28). http://www.linuxfordevices.com/c/a/News/64way-chip-gains-Linux-IDE-dev-cards-design-wins/.

[35] ROS A, ACACIO M E, GARCIA J M. A direct coherence protocol for many-core chip multiprocessors [J]. IEEE Trans on Parallel and Distributied Systems, 2010, 21(12): 1779-1792.

[36] PUGSLEY S H, SPJUT J B, NELLANS D W, *et al*. SWEL: hardware cache coherence protocols to map shared data onto shared caches [C]//Proc of the 19th International Conference on Parallel Architectures and Compilation Techniques. New York: ACM, 2010: 465-476.

[37] CHOI I, ZHAO M-S, YANG X, *et al*. Experience with improving distributed shared cache performance on tilera's tile processor [J]. IEEE Computer Architecture Letters, 2011, 10(2): 45-48.

[38] LI Yong, A BOUSAMRA A, MELHEM R, *et al*. Compiler-assisted data distribution for chip multiprocessors [C]//Proc of the 19th International Conference on Parallel Architectures and Compilation Techniques. New York: ACM, 2010: 501-512.

(上接第4010页)

[48] MA'AYAN A. Insights into the organization of biochemical regulatory networks using graph theory analyses [J]. Journal of Biological Chemistry, 2009, 284(9): 5451-5455.

[49] MASON O, VERWOERD M. Graph theory and networks in biology [J]. IET Systems Biology, 2007, 1(2): 89-119.

[50] LUSCOMBE N M, BABU M M, YU H Y, *et al*. Genomic analysis of regulatory network dynamics reveals large topological changes [J]. Nature, 2004, 431(7006): 308-312.

[51] ALEXANDER R P, KIM P M, EMONET T, *et al*. Understanding modularity in molecular networks requires dynamics [J]. Science Signaling, 2009, 81(2): 44.

[52] IRONS D J, MONK N A M. Identifying dynamical modules from genetic regulatory systems: applications to the segment polarity network [J]. BMC Bioinformatics, 2007, 8: 413.

[53] SORANZO N, RAMEZANI F, IACONO G, *et al*. Decompositions of large-scale biological systems based on dynamical properties [J]. Bioinformatics, 2012, 28(1): 76-83.

[54] GRAMMATICOS B, CARSTEA A S, RAMANI A. On the dynamics of a gene regulatory network [J]. Journal of Physics A: Mathematical and General, 2006, 39(12): 2965-2971.

[55] BABU M M, TEICHMANN S A, ARAVIND L. Evolutionary dynamics of prokaryotic transcriptional regulatory networks [J]. Journal of Molecular Biology, 2006, 358(2): 614-633.