片上多核处理器 Cache 访问均衡性研究

王子聪 陈小文 郭 阳
(国防科技大学计算机学院 长沙 410073)

摘 要 随着片上多核处理器(CMP)规模的不断扩大和处理核数的增多,系统对于片上缓存(Cache)在容量和速度方面有了更高的需求.为了能够有效利用 Cache 资源,非一致 Cache 体系结构(NUCA)被提出用于支持高容量低延迟的 Cache 组织结构.另一方面,片上网络(NoC)由于具备良好的可扩展性,在片上多核处理器的互连方式上具有显著优势.因此,基于片上网络的非一致 Cache 体系结构逐渐成为未来组织大容量 Cache 的主流系统架构.在这样的系统架构中,最后一级缓存(LLC)通常在物理上分布于每个处理节点,这些 Cache 存储体(Bank)在逻辑上共同构成一个统一的共享 Cache.当处理核发出 Cache 访问请求时,其访问时间与请求处理核节点与访问数据所在的 Bank 节点的距离有关.当距离较近时,访问时间较短;当访问距离较远的 Bank 时,访问时间较长.因此,当系统规模逐渐增大时,这种访问延迟与网络距离相关的特性会使得不同节点之间的通信距离和通信延迟的差异性逐渐增大.另外,片上网络规模的增大也会使得 Cache 访问延迟逐渐由网络延迟主导.这种延迟差异性会引起网络报文延迟不均衡问题,导致 Cache 访问延迟的非一致性进一步增大,因而出现更多的大延迟 Cache 访问并成为制约系统性能的瓶颈.因此,研究片上多核处理器的 Cache 访问均衡性对于提升网络性能和系统性能具有积极意义.该文分析了造成 Cache 访问延迟不均衡的原因,并针对延迟的两个来源:无冲突延迟和竞争延迟,分别提出了非一致存储映射和非一致链路分布的设计方法.通过非一致存储映射,我们根据 Cache 存储体在网络中的物理位置调节其相应的 Cache 块映射比例,从而均衡 Cache 请求平均访问距离;通过合理设计非一致的链路分布,我们依据各条链路上的流量负载为其分配合适的通道数量,从而缓解流量压力较大的链路上的报文竞争.全系统模拟器上的实验表明,采用面向 Cache 访问均衡性的片上多核处理器能够有效均衡 Cache 访问延迟,并减少大延迟 Cache 访问请求的数量.相比于传统的 NUCA 结构,我们的设计在最大的实验规模(64 核)下在延迟均方差、最大延迟和平均延迟上分别平均降低了 19.6%、12.8%和 6.4%,最大降低了 40.8%、29.9%和 11.9%.同时在系统性能方面,通过 PARSEC 应用程序的模拟实验表明,单位周期执行指令数(IPC)平均提升了 6.7%,最大提升了 14.0%.

关键词 片上多核处理器;非一致缓存体系结构;片上网络;均衡性;缓存访问
中图法分类号 TP393 **DOI 号** 10.11897/SP.J.1016.2019.02403

Research on Cache Access Equalization in Chip Multi-Processor

WANG Zi-Cong CHEN Xiao-Wen GUO Yang
(*College of Computer, National University of Defense Technology, Changsha* 410073)

Abstract Along with the scaling up for the size of chip multi-processor (CMP) and the increase in the number of cores, the system has a higher demand for on-chip cache in terms of capacity and speed. In order to effectively utilize cache resources, non-uniform cache architecture (NUCA) is proposed to support cache organization with high-capacity and low-latency. On the other hand, networks-on-chip (NoC) has significant advantages in terms of the interconnection of CMP due to its good scalability. Therefore, NoC-based NUCA is gradually becoming the major architecture to organize large cache. In such system architecture, last level cache (LLC) is distributed on every network node, and all the cache banks logically constitute a unified shared cache. When a core

收稿日期:2017-09-13;录用日期:2018-04-18.本课题得到国家自然科学基金(61502508,61572025)、湖南省自然科学基金(2015JJ3017)资助.
王子聪,博士研究生,主要研究方向为片上网络设计.E-mail:wangzicong@nudt.edu.cn.**陈小文**(通信作者),博士,助理研究员,主要研究方向为多核处理器设计、片上网络设计.E-mail:xwchen@nudt.edu.cn.**郭 阳**,博士,研究员,博士生导师,中国计算机学会(CCF)高级会员,主要研究领域为微处理器设计验证技术.

issues a cache access request, the access time is determined by the network distance between the core and the requested cache bank. When the cache bank is near the core, the access time is short; when accessing a cache bank with a long distance, the access time is longer. Thus, when the scale of system is gradually increased, the communication distance and latency gap between different cores is also increased due to the feature of the access latency associated with the network distance. In addition, the increase in the size of NoC will also make the cache access latency gradually dominated by the network latency. Such latency gap can cause the network latency imbalance problem, aggravate the degree of non-uniform for cache access latencies, and lead to more cache accesses with overhigh latencies which become the bottleneck of system. Hence, the research on cache access equalization in CMP has a positive meaning for the promotion of network and system performance. This paper analyzes the reasons for the cache access imbalance, and proposes design methods including non-uniform memory mapping scheme and non-uniform link distribution which aim to balance the non-contention and contention latencies respectively. In non-uniform memory mapping scheme, we adjust the proportion of cache block mapped from memory to each cache bank according to the physical location of cache bank in network, which achieves the goal of balancing the average distance of cache access. By reasonably designing non-uniform link distribution, we allocate the number of physical channels on each link according to its corresponding traffic load to alleviate the contention for those links with heavier traffic load. We perform the evaluation of our design in a full-system simulator. The experimental results show that our design can effectively balance the cache access latencies, and reduce the number of cache accesses with overhigh latencies. Compared with the traditional NUCA, our design can decrease latency standard deviation (LSD), maximum latency (ML) and average latency (AL) by 19.6%, 12.8% and 6.4% on average, and the maximum decrease can be up to 40.8%, 29.9% and 11.9%, under the largest chip scale (64 cores) in full-system experiments. At the same time, in terms of system performance, the experimental results for simulation with PARSEC benchmarks show that our design can improve instruction per cycle (IPC) by 6.7%, and the maximum improvement can be up to 14.0%.

Keywords chip multi-processor; non-uniform cache architecture; networks-on-chip; equalization; cache access

1 引 言

“存储墙”问题是制约现代计算机系统性能提升的一个重要原因,而缓存(Cache)技术是有效缓解“存储墙”问题的重要手段.随着集成电路工艺的不断提升,由大容量 Cache 催生出的非一致 Cache 体系结构(Non-Uniform Cache Architecture,NUCA)[1]逐渐成为片上多核处理器(Chip Multi-Processor,CMP)的基础设计结构.另一方面,面对日益增长的芯片规模,片上网络(Network-on-Chip,NoC)由于具备良好的可扩展性成为 NUCA 结构中组织最后一级缓存(Last Level Cache,LLC)的主要互连方式.因此,基于 NoC 的 NUCA 结构具有可扩展性好、Cache 容量大等优点,成为当前片上多核处理器的主流系统架构.

在基于 NoC 的 NUCA 结构中,LLC 通常在物理上分布于各个节点,每个节点的 Cache 存储体(Bank)在逻辑上构成一个统一的共享 Cache.在这样的 NUCA 结构中,当处理核发出 Cache 访问请求时,其访问时间与请求处理核节点与访问数据所在的 Bank 节点的距离有关.当距离较近时,访问时间较短;当访问距离较远的 Bank 时,访问时间较长.然而,随着网络规模的扩大和节点个数的增多,Bank 位置的不对等性增加,访问不同 Bank 的差异性也会随之加大.以二维网格网络为例.图 1 显示的是在的 4×4 网格网络下,各个节点上 Bank 的平均访问距离.从图 1 中可以看出,由于 Bank 所处位置

不同，中心节点的 Bank 相比于边上和角落上的节点的 Bank 在平均访问距离上更具有优势（中心节点的平均距离为 2，而边上和角落上的节点的平均距离分别为 2.5 和 3）. 随着 Bank 访问距离的不均衡性增加，位于不同节点 Bank 的 Cache 访问时间的差异性逐步增大. 这种情况会导致部分 Cache 访问请求的延迟非常大. 图 2 所示为在 8×8 网格网络下运行 PARSEC[2] 应用程序（blackscholes）收集的 Cache 访问请求报文延迟的累积分布统计数据（以 128 个周期为间隔）. 我们可以看出虽然绝大多数请求报文的延迟小于 128 个周期（约 99.16%），但剩下的仍有 0.84% 的请求报文延迟大于 128 个周期，同时最大延迟接近 3072 个周期（实测为 2889 个周期），是平均延迟（实测为 24 个周期）的 120 倍. 这些大延迟 Cache 访问请求会阻塞处理核的执行进程，成为系统瓶颈并严重影响系统整体性能. 我们将这种由于大网络规模下 NUCA 结构造成的 Cache 非一致访问恶化现象称之为 Cache 访问的均衡性问题.

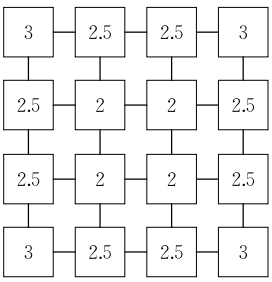


图 1　4×4 网格网络下各个节点上 Bank 的平均访问距离（单位为跳步数）

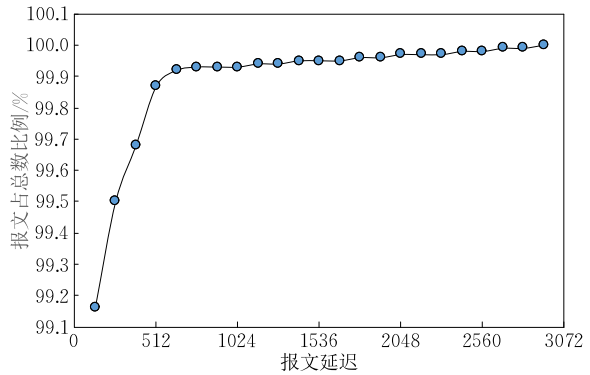


图 2　8×8 网格网络下运行 PARSEC 应用程序（blackscholes）的 Cache 访问请求报文延迟的累积分布图

为了解决 Cache 访问的均衡性问题，首先我们需要分析 Cache 访问时间的组成. 在基于 NoC 的 NUCA 结构中，Cache 访问时间由如下公式确定：

$$T_{\text{access}} = T_{\text{network}} + T_{\text{bank}} \tag{1}$$

其中，T_{network} 表示 Cache 访问时间在片上网络中的延迟，T_{bank} 表示 Cache Bank 存储体本身的访问延迟.

在采用虫孔路由的 NoC 中，网络报文延迟包括无冲突延迟和竞争延迟两部分，而无冲突延迟又包括在路由器和链路中的传输延迟以及一个微片（flit）通过链路的序列化延迟[3]，如式（2）所示.

$$T_{\text{network}} = H \times (t_r + t_l) + \left\lceil \frac{L}{b} \right\rceil + T_c \tag{2}$$

在上式中，H 表示报文从源节点到目的节点的跳步数，t_r 和 t_l 分别表示一个微片在单个路由器和单条链路的传输延迟，$\lceil L/b \rceil$ 表示序列化延迟（L 表示报文长度，b 表示链路带宽），T_c 表示报文在传输过程中总的竞争延迟.

将式（2）代入到式（1）中，可以得到 Cache 访问时间公式：

$$T_{\text{access}} = H \times (t_r + t_l) + \left\lceil \frac{L}{b} \right\rceil + T_c + T_{\text{bank}} \tag{3}$$

在确定的网络参数及报文协议和存储体实现下，t_r、t_l、L、b 和 T_{bank} 为固定常数. 因此，不同 Cache 访问请求的延迟的差异主要体现网络延迟 T_{network}，即参数 H 和 T_c 上. 其中，H 主要由请求源节点与目标存储体的位置和网络规模决定，T_c 主要由 Cache 访问的通信特性（如通信流量、报文注入率等）决定.

由上述分析可知，在基于 NoC 的 NUCA 结构中，由于网络规模增大、节点个数增多和 Cache 访问请求数量增多，H 和 T_c 的变化幅度增加，不同 Cache 访问请求的延迟差异性变大，使得共享 Cache 非一致访问的均衡性问题日益严重. 考虑到不同节点的位置是导致差异性的原因之一，本文旨在面向 Cache 访问均衡性调节不同位置的节点访问 Bank 的平均距离以及竞争延迟，通过在 NUCA 结构基础上引入非一致设计，缩短 Cache 访问延迟差异，从而提升系统性能.

NUCA 技术最初是由 Kim 等人[1] 提出的，用于应对单核处理器上 Cache 容量和延迟之间的矛盾. 随后，Beckmann 等人[4] 将 NUCA 的概念引入到多核处理器中. 根据 Cache 映射机制的不同，NUCA 结构分为两种，静态 NUCA（Static NUCA，S-NUCA）和动态 NUCA（Dynamic NUCA，D-NUCA）. 在 S-NUCA 中，物理存储空间中的一个 Cache 块会根据其物理地址被静态地映射到某个确定的 Bank 中，即对于某个给定的 Cache 块，它所在的 Bank 是

确定的[5].因此,每个 Bank 在物理存储空间中都对应着确定的映射区域.而在 D-NUCA 中,Cache 块会根据其物理地址映射到多个 Bank 中的一个,并且 Cache 块可以根据访问情况在多个 Bank 之间迁移.因此,频繁被访问的 Cache 块可以动态迁移到离请求处理核相近的 Bank 中从而降低访问延迟[6].由于 D-NUCA 在数据迁移方面具有较高的灵活性,它需要依靠较为复杂的搜索机制将 Cache 块分配在合适的位置.因此,相比于 S-NUCA,D-NUCA 的硬件开销较大.另外,D-NUCA 并不是在任何情况下都能取得比 S-NUCA 更为优异的性能[1][4].因此有时设计简单的 S-NUCA 在权衡开销与性能时更受青睐[7].考虑到 S-NUCA 结构的广泛使用,本文主要针对采用 S-NUCA 结构的片上多核处理器,并面向 Cache 访问均衡性进行优化改进.

2 面向 Cache 访问均衡性的 NUCA 结构设计

2.1 基准 S-NUCA 结构

图 3 所示的是一个典型的基于 S-NUCA 结构的片上多核处理器(以 4×4 的网格网络为例).在图 3 中,一个处理单元(Processing Element,PE)包括一个一级指令/数据 Cache,一个二级共享 Cache Bank 和一个网络接口(Network Interface,NI),每个 PE 通过 NI 连接到一个路由器上.同时,在存储器控制器的布局方面我们选择了"菱形"布局,如图 3 中的 DRAM 控制器布局所示.相比于其他的布局方式,采用"菱形"布局更能够减少报文的链路竞争以及避免热点区域的出现[8].另外,这些分布式共享的二级 Cache Bank 通过 S-NUCA 结构的方式组织起来,并且采用以 Cache 块为单位的方式进行交叉编址.同时,正因为采用了低位交叉编址方式,这样的 S-NUCA 结构在将数据从存储器映射到 LLC 时,会打破存储器中数据的局部性,将连续的数据分布到各个 Bank 中.本文主要考虑的是基于负载均衡的应用映射机制下的情况,而这种映射机制在实际中经常应用从而避免网络热点以及均衡流量分布[9-10].在这样的结构下,访问 L2 共享缓存的流量分布会趋近于 Uniform-random 流量模型[11],即网络流量会均匀地分布在各个 Bank 上,从而使得每个 Bank 被访问的概率大致相同.

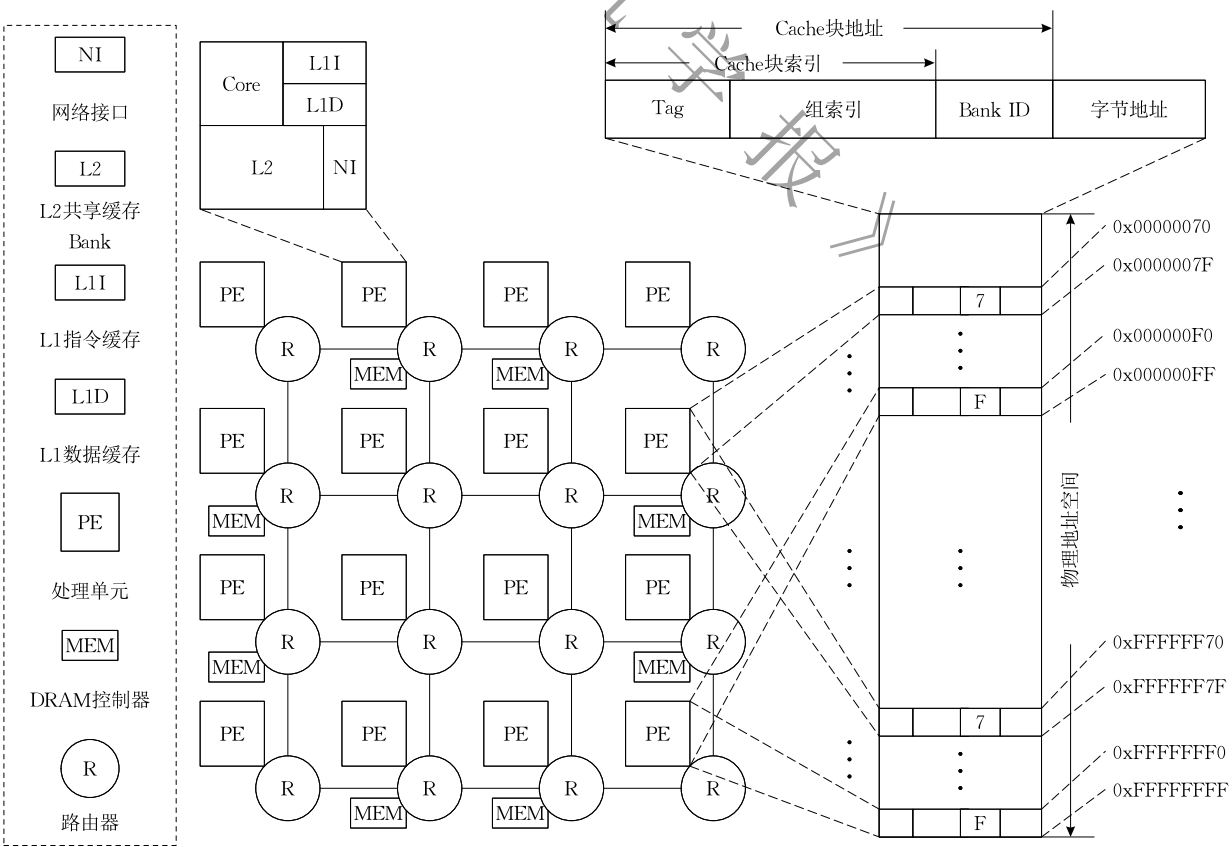


图 3 基准 S-NUCA 结构示意图(以 4×4 的网格网络为例)

2.2　面向 Cache 访问均衡性的 NUCA 结构设计方案

传统的 S-NUCA 结构为了能够适应越来越大的网络规模而实现了非一致 Cache 访问延迟，但却忽视了由此带来的对结构其他方面的影响. 因此，本文致力于在均衡负载的基础上以 Cache 访问均衡性为主要目标，从两个结构设计方面改进传统的 S-NUCA 结构. 一是针对传统 S-NUCA 结构下的一致存储映射，依据网络规模结构参数设计存储器到 LLC 的非一致存储映射；二是针对传统 S-NUCA 结构下的一致链路分布，依据流量分布特征设计链路上的物理通道数量分布，实现基于 S-NUCA 结构的非一致链路分布.

2.3　非一致存储映射

在传统的 S-NUCA 结构下，从存储器映射到每个 Bank 的 Cache 块数量相同，而从 Bank 被访问的角度来看，每个 Bank 被访问概率大致与其映射的 Cache 数量成正比，因此对于具有一致存储映射的 S-NUCA 结构，每个 Bank 被访问的概率大致相同. 然而由于各个 Bank 和处理核所处的位置不同，而不同位置又具有不同的平均访问距离，尤其是中心节点的 Bank 具有较低的平均访问距离，而角落节点的 Bank 具有较高的平均访问距离，因此可以使较多的 Cache 块映射到中心节点，同时使较少的 Cache 块映射到角落节点，从而均衡各个处理核节点的平均访问距离（即式(3)中的参数 H），以此达到均衡各个节点的 Cache 访问延迟的目的. 下面我们将以一个大小为 $M\times N$ 并采用 YX 维序路由策略（YX Dimension-Order Routing，YX-DOR）的网格网络为例，计算面向 Cache 访问均衡性的非一致存储映射分布. 假设位于节点 (m,n) 的 Bank 的被访问概率为 $p_{m,n}$，那么我们需要计算的非一致存储映射分布可以用矩阵 $\boldsymbol{P}$ 表示：

$$\boldsymbol{P}=[p_{m,n}]_{M\times N} \tag{3}$$

假设位于节点 (i,j) 的处理核需要访问位于节点 (m,n) 的 Bank，那么它们之间的访问距离可以由曼哈顿（Manhattan）距离表示：

$$d_{i,j}(m,n)=|i-m|+|j-n| \tag{4}$$

从而可以得到位于节点 (m,n) 的 Bank 的平均访问距离：

$$\begin{aligned} d_{m,n}&=\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}[d_{i,j}(m,n)\times p_{m,n}]\\ &=\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}[(|i-m|+|j-n|)\times p_{m,n}] \end{aligned} \tag{5}$$

因此，我们可以用矩阵 $\boldsymbol{D}$ 表示每个节点的 Bank 的平均访问距离：

$$\boldsymbol{D}=[d_{m,n}]_{M\times N} \tag{6}$$

然后就可以根据矩阵 $\boldsymbol{D}$ 得到所有 Bank 的平均访问距离的平均值：

$$\begin{aligned} \mu(\boldsymbol{D})&=\frac{1}{MN}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}d_{m,n}\\ &=\frac{1}{MN}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}[(|i-m|+|j-n|)\times p_{m,n}] \end{aligned} \tag{7}$$

为了能够均衡各个 Bank 的平均访问延迟，我们希望各个 Bank 的平均访问距离能够互相越接近越好，即矩阵 $\boldsymbol{D}$ 中元素集合的标准差能够越小越好，因此选取该标准差作为优化的目标函数：

$$\sigma(\boldsymbol{D})=\sqrt{\frac{1}{MN}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}(d_{m,n}-\mu(\boldsymbol{D}))^2} \tag{8}$$

另外，所有的 Bank 的访问概率之和应该等于 1，并且每个 Bank 的访问概率应当大于或等于 0，即满足如下约束公式：

$$\begin{aligned} &\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}p_{m,n}=1,\\ &p_{m,n}\geq 0(0\leq m\leq M-1,\ 0\leq n\leq N-1) \end{aligned} \tag{9}$$

根据上述总结的目标函数以及约束，面向 Cache 访问均衡性的非一致存储映射最优方案求解可以抽象为如下的非线性规划问题：

$$\begin{aligned} &\text{minimize}\ \sigma(\boldsymbol{D})\\ &\text{subject to}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}p_{m,n}=1\\ &\qquad\qquad p_{m,n}\geq 0 \end{aligned} \tag{10}$$

为了求解该非线性规划问题，我们在 MATLAB 中对此问题进行了建模，同时利用 MATLAB 提供的优化工具进行了求解. 这里我们以 8×8 网络规模为例进行相关计算，最终求解得到的 Bank 访问概率分布如矩阵 $\boldsymbol{P}$ 所示：

$$\boldsymbol{P}=\begin{bmatrix} 12 & 13 & 14 & 15 & 15 & 14 & 13 & 12\\ 13 & 15 & 16 & 17 & 17 & 16 & 15 & 13\\ 14 & 16 & 18 & 18 & 18 & 18 & 16 & 14\\ 15 & 17 & 18 & 19 & 19 & 18 & 17 & 15\\ 15 & 17 & 18 & 19 & 19 & 18 & 17 & 15\\ 14 & 16 & 18 & 18 & 18 & 18 & 16 & 14\\ 13 & 15 & 16 & 17 & 17 & 16 & 15 & 13\\ 12 & 13 & 14 & 15 & 15 & 14 & 13 & 12 \end{bmatrix}\times 10^{-3} \tag{11}$$

当得到了 Bank 访问概率分布后，我们可以通过该分布计算每个 Bank 与其访问概率一致的存储映射 Cache 块的比例. 对于大小为 8×8 网格网络构成的 S-NUCA 结构，Bank ID 字段在存储器地址中需要占据 6（即 $\log_2 64$）比特，从而能够在存储器到

LLC Bank 的映射时直接根据 Bank ID 字段得到相应 Cache 块所在的 Bank，同时这样的方式也因此保证了每个 Bank 都能映射到相同比例的 Cache 块. 然而，在我们的设计方法中，由于映射是非一致的，导致映射到各个 Bank 的 Cache 块的数量是不相同的，因此我们不能采用原先直接将 Bank ID 字段作为 Bank 地址的标识，取而代之的应是将 Bank ID 字段进行扩展（即大于 6 比特）作为 Bank 地址，并依据 Bank 访问概率对不同 Bank 地址的 Cache 块进行合理映射. Bank 地址占据的位数越多，实际映射的 Cache 块比例分布就可以越接近于理想的 Bank 访问概率分布，但另一方面也提升了映射实现的复杂度. 这里我们将 Bank ID 字段扩展为 9 比特作为 Bank 地址. 因此，下面我们需要考虑如何将这 9 比特能够表示的 Cache 块数量（即 512 个 Cache 块）按照接近于 Bank 访问概率分布的方式映射到 64 个 Bank ID 上（即 0 到 63）. 首先将矩阵 $\boldsymbol{P}$ 乘以 Cache 块总数，得到 512 个 Cache 块下各个 Bank 映射的 Cache 块数量：

$$\boldsymbol{B}=\boldsymbol{P}\times 2^{9}\approx\begin{bmatrix}6&7&7&8&8&7&7&6\\7&7&8&9&9&8&7&7\\7&8&9&9&9&9&8&7\\8&9&9&10&10&9&9&8\\8&9&9&10&10&9&9&8\\7&8&9&9&9&9&8&7\\7&7&8&9&9&8&7&7\\6&7&7&8&8&7&7&6\end{bmatrix}\tag{12}$$

从 Cache 块数量的分布可以看出，其分布趋势与我们预期相同，即更多的 Cache 块映射到了中心节点，而外围节点映射的 Cache 块数量较少. 而对于传统的 S-NUCA 结构，这 512 个 Cache 块应当均匀地分布在各个 Bank 上，即每个 Bank 映射 8 个 Cache 块，所以需要将一部分原本属于外围节点的 Cache 块映射到中心节点上. 我们将原本 6 比特的 Bank ID 字段扩展为 9 比特作为 Bank 地址，并将高 3 位作为标志位（Bank tag），而将低 6 位作为索引位（Bank index）. 根据标志位的不同，我们可以把 512 个 Cache 块分为 8 组，每组包含 64 个 Cache 块，而索引位则表示在原先的 S-NUCA 结构下该 Cache 块应当映射到的 Bank 地址.

为了能够将外围节点的 Cache 块映射到中心节点，我们将某些组下的 Cache 块的映射进行了重新调整，图 4 所示的是 8×8 网格网络下的非一致存储映射设计示意图. 注意到 8×8 网格网络下节点可以分为 4 个区域，每个区域都遵循类似的映射方式，即将靠近角落处的节点的部分 Cache 块映射到靠近中心处的节点. 以左上角区域为例，靠近中心的节点 27 需要映射 10 个 Cache 块，而位于角落的节点 0 需要映射 6 个 Cache 块，与 S-NUCA 结构下的映射相比，节点 27 多了 2 个 Cache 块而节点 0 少了 2 个 Cache 块. 因此我们将第 7、8 组（即 Bank tag 等于 6 或 7）下原本属于节点 0 的 Cache 块映射到节点 27. 同理，对于节点 1/2/8/9/16 将第 8 组（即 Bank tag 等于 7）下的 Cache 块分别映射到节点 19/11/26/18/25. 而对于位于各个区域对角线上的节点则保持其 Cache 块映射与原先相同. 通过这样的设计，一方面既保证了 Cache 块分布与 Bank 概率访问分布的最大一致，另一方面也尽可能地简化映射设计，从而可以在实际的硬件实现中通过采用线性表的设计方式直接得到 Bank ID.

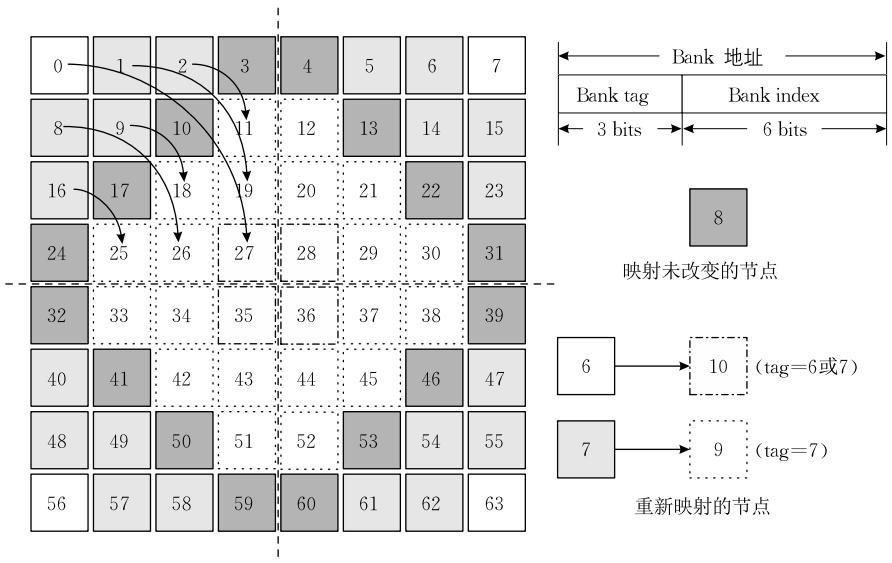


图 4 8×8 网格网络下的非一致存储映射设计示意图

图 5 显示的是 8×8 网格网络下的非一致存储映射设计中每组 Cache 块的映射结果(主要显示左上角区域部分),可以看出对于前 6 组(即 Bank tag 等于 0 至 5)映射方式与 S-NUCA 结构中保持一致;对于后 2 组(即 Bank tag 等于 6 或 7)将部分 Cache 块映射到靠近中心节点处.

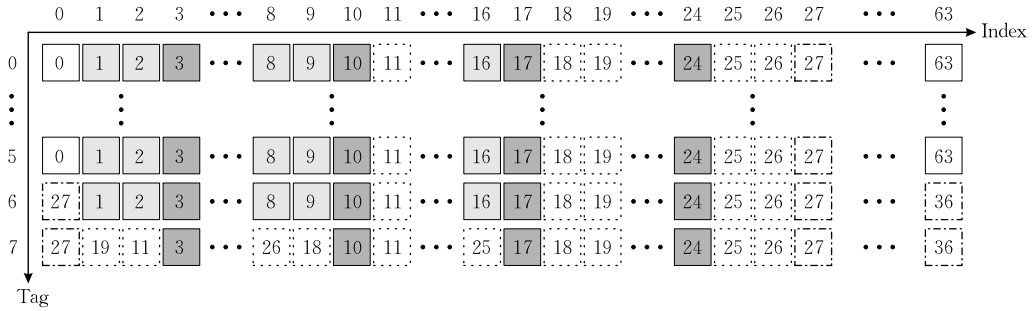


图 5　8×8 网格网络下的非一致存储映射设计中每组 Cache 块的映射结果示意图

尽管上述分析和计算是针对 8×8 网格网络下的 NUCA 结构,但对于其他规模的网格网络可以得到类似的结论,即分布的总体趋势是将外围节点的部分 Cache 块重新映射到中心节点,因此 Bank 访问概率分布(矩阵 $\boldsymbol{P}$)和 Cache 块数量分布(矩阵 $\boldsymbol{B}$)应具有类似的结构,从而能够遵循类似的映射方案,设计对存储映射进行合理优化.

2.4　非一致链路分布

非一致存储映射旨在均衡位于不同位置节点 Bank 的平均访问距离.类似地,连接处理核之间的链路同样分布在网格网络的不同位置,而由于网格网络的不对等性,这些处在不同位置的链路在流量负载上具有一定的差异性.随着处理核的增多以及网络规模的不断增大,网络中的链路数量也不断增长,并且在流量负载方面的差异性逐渐增大.这种差异性会使得某些链路具有非常高的流量负载,并增加了该条链路上报文的竞争延迟(即式(3)中的参数 T_c).另外,非一致的存储映射也增加了中心节点处链路的流量压力.因此,面向 Cache 访问均衡性的优化的第二个方面主要依据各个链路的流量负载,通过合理设计非一致的链路分布,增加流量压力较大的链路的通道数量实现各条链路上报文之间较为均衡的竞争延迟.在文献[11]中,Zhang 等人分析了在 Uniform 流量模型下网格网络中不同位置的链路的流量负载分布.然而对于非一致的存储映射下的 NUCA 结构,其流量模型不再趋近于 Uniform,因此我们需要重新计算基于非一致存储映射下各个链路的流量负载分布.

假定网络规模为 $M\times N$,那么该网络下的链路可以根据方向分为两组,即水平方向上的链路和垂直上的链路,其规模分别为 $M\times(N-1)$ 以及 $(M-1)\times N$.为了衡量基于非一致存储映射下各个链路上的流量负载情况,我们假定每个处理核按照 Cache 块数量分布(即矩阵 $\boldsymbol{B}$)向其他节点所在的 Bank 发出 Cache 访问请求,即每个处理核发出 512 个报文.我们将某条链路上报文的通过次数定义为该条链路的繁忙因子,通过计算出各条链路的繁忙因子,可以知道其流量负载相对大小,从而可以按比例对每条链路分配物理通道.

假设发出请求的处理核的节点位置为 (x,y),接收请求的 Bank 的节点位置为 (m,n),那么根据源节点和目的节点在网络中相对位置的不同,可以将其分为 4 类,如图 6 所示.同时,我们采用 YX 维序路由策略,即在选择路由路径时水平方向上的链路的优先级大于垂直方向上的链路(假定水平方向为 Y 轴,垂直方向为 X 轴).

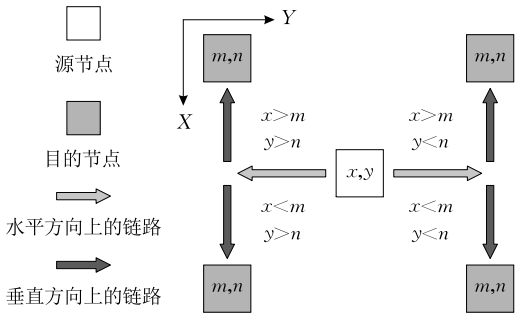


图 6　源节点 (x,y) 和目的节点 (m,n) 之间的相对位置关系示意图

基于上述假设,我们就可以按照每个处理核向各个 Bank 发送指定的报文数量,并结合处理核节点与 Bank 节点的位置关系计算各条链路的繁忙因子,如算法 1 所示.

算法 1. 计算各条链路的繁忙因子.

输入：网络规模，$M\times N$；

Cache 块数量分布，$\boldsymbol{B}=[b_{i,j}]_{M\times N}$

输出：水平方向上链路的繁忙因子分布，

$H_{busy}=[h_{i,j}]_{M\times(N-1)}$；

垂直方向上链路的繁忙因子分布，

$V_{busy}=[v_{i,j}]_{(M-1)\times N}$

FOR $(i,j)\leq(0,0)\rightarrow(M-1,N-2)$ DO

  $h_{i,j}\leq 0$

END FOR

FOR $(i,j)\leq(0,0)\rightarrow(M-2,N-1)$ DO

  $v_{i,j}\leq 0$

END FOR

FOR $(x,y)\leq(0,0)\rightarrow(M-1,N-1)$ DO

  FOR $(m,n)\leq(0,0)\rightarrow(M-1,N-1)$ DO

    IF $x>m$ THEN

      FOR $i\leq m\rightarrow x-1$ DO

        $v_{i,n}\leq v_{i,n}+b_{m,n}$

      END FOR

    ELSE IF $x<m$ THEN

      FOR $i\leq x\rightarrow m-1$ DO

        $v_{i,n}\leq v_{i,n}+b_{m,n}$

      END FOR

    END IF

    IF $y>n$ THEN

      FOR $j\leq n\rightarrow y-1$ DO

        $h_{x,j}\leq h_{x,j}+b_{m,n}$

      END FOR

    ELSE IF $y<n$ THEN

      FOR $j\leq y\rightarrow n-1$ DO

        $h_{x,j}\leq h_{x,j}+b_{m,n}$

      END FOR

    END IF

  END FOR

END FOR

RETURN H_{busy}, V_{busy}

在 8×8 的网络规模下，可以计算得到各条链路的繁忙因子，如图 7 所示. 为了能够根据各条链路的流量负载按比例增加链路带宽，缓解流量负载较大的链路（主要集中在中心节点附近）的竞争延迟，我们以网络中繁忙因子最小的链路为基准，通过每条链路的繁忙因子与最小繁忙因子的比值作为该条链路上的物理通道的数量. 假设最小繁忙因子为 $busy_{\min}$：

$$busy_{\min}=\min\{\boldsymbol{H}_{busy},\boldsymbol{V}_{busy}\} \tag{13}$$

将各条链路的繁忙因子除以 $busy_{\min}$ 并进行取整运算，即得到各条链路的物理通道数量：

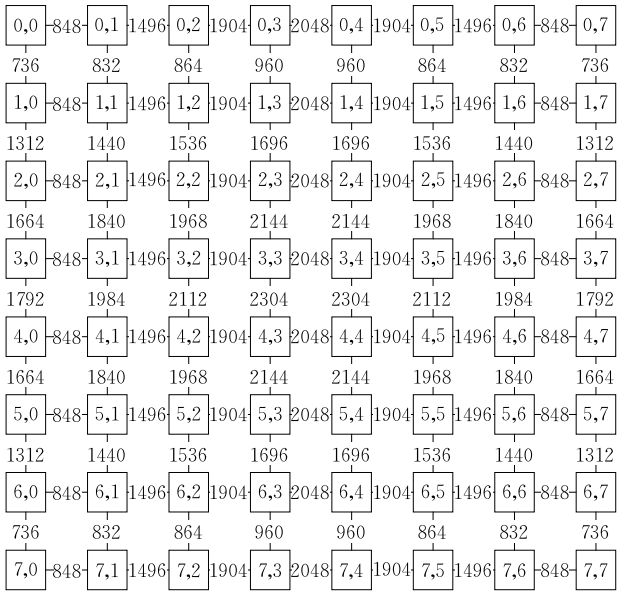


图 7 8×8 网络规模下链路的繁忙因子分布

$$H_{\text{link}}=\frac{H_{busy}}{busy_{\min}}=\left[round\left(\frac{h_{i,j}}{busy_{\min}}\right)\right]_{M\times(N-1)}$$
$$V_{\text{link}}=\frac{V_{busy}}{busy_{\min}}=\left[round\left(\frac{v_{i,j}}{busy_{\min}}\right)\right]_{(M-1)\times N} \tag{14}$$

在 8×8 的网络规模下，根据图 7 所示的链路繁忙因子分布，即可得到链路的物理通道数量分布，如图 8 所示.

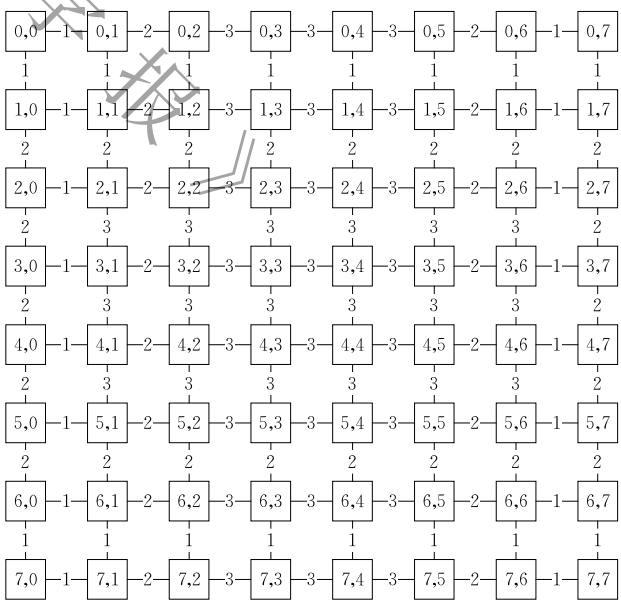


图 8 8×8 网络规模下链路的物理通道数量分布

类似于非一致存储映射优化，针对不同规模的网络，我们都可以采取上述设计方法计算链路的物理通道数量分布，从而增加中心节点的通信带宽，进一步均衡 Cache 访问.

3　实验评估

3.1　实验配置

为了评估面向 Cache 访问均衡性设计方案的有效性，在实验模拟平台的选择上，我们选择在全系统体系结构模拟器 GEM5 上开展相关实验. GEM5 模拟器是由学术界和工业界共同设计开发的一款体系结构模拟器，支持从系统结构级到微体系结构级的模拟[12]. GEM5 能够支持多种 Cache 一致性协议和主流指令集，同时包含了周期精确的互连网络模型 GARNET[13] 和精确的存储子系统模型 Ruby. 近年来出现的应用级并行模拟器 Graphite[14]、Sniper[15] 和 ZSim[16]，相比于传统的 GEM5 模拟器，虽然能够利用多进程实现数十倍的模拟运行速度[17]，但在模拟精度和准确度上有所欠缺，尤其是 GEM5 的互连网络模型 GARNET 和存储子系统模型 Ruby 能够很好地满足本文的实验需求，因此我们选择了 GEM5 作为模拟实验平台.

与计算 8×8 规模下的非一致存储映射和非一致链路分布过程类似，我们基于三种片上规模(4×4/4×8/8×8)的 S-NUCA 结构，修改并分别实现了三种规模下的面向 Cache 访问均衡性的 NUCA 结构. 实验方案分为两个部分. 首先，我们将 GEM5 模拟器切换至网络模拟模式(即采用 GEM5 中的 Garnet_standalone Cache 一致性协议)，即网络中的每个节点的处理核只作为网络流量的注入点，而不模拟运行实际的 CPU 指令，各个节点按照指定的合成流量模型向网络注入报文. 在此模式下，我们将 Bank 访问概率分布对应的流量模型作为对非一致存储映射的模拟，并结合非一致链路分布设计与 S-NUCA 结构进行实验对比. 第二，我们将 GEM5 模拟器切换至全系统模式(即在 GEM5 的 Full System 模式下采用 MESI_Two_Level Cache 一致性协议)，在每个处理核上运行真实的 CPU 指令. 在此模式下，我们按照计算得到的 Cache 块数量分布情况修改了存储器到 LLC 的映射，并在此基础上结合非一致链路分布设计作为实验对比方案. 全系统实验配置参数如表 1 所示.

表 1　实验配置参数

配置参数	参数值
指令集	ALPHA
Cache 一致性协议	MESI
处理核数	16/32/64
网络拓扑结构	二维 4×4/4×8/8×8 网格网络
路由算法	YX-DOR
微片大小	64 bits
Cache 块大小	64 bytes
一级指令/数据 Cache 容量	32 kB
二级 Cache Bank 容量	256 kB
存储器控制器数量	8/8/16

3.2　合成流量模拟实验

尽管在网络模拟模式下，GEM5 模拟器并不包含存储系统的模拟，但我们可以通过设计 Bank 访问概率分布对应的合成流量模型作为对非一致存储映射的模拟，例如在 8×8 的规模下，我们将每个节点注入报文的目的节点的比例按照式(11)所示的 Bank 访问概率分布设置. 同时，结合非一致链路分布我们将该结构与 Uniform 流量模型下的结构进行了实验对比，实验结果如图 9 所示. 在实验中，我们主

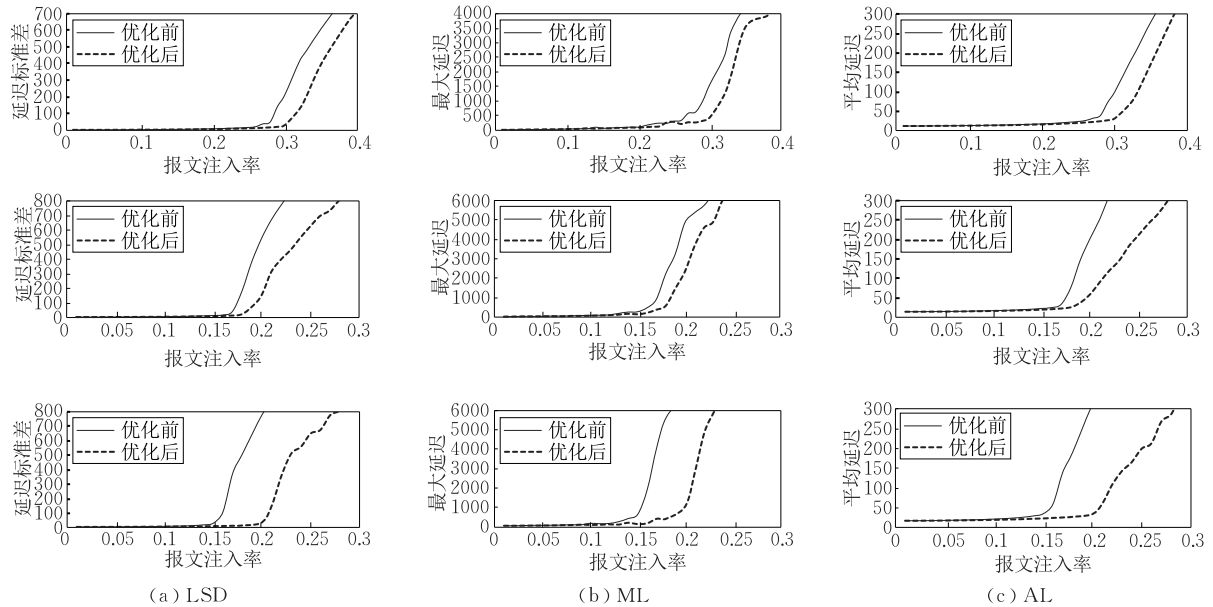


图 9　不同规模下合成流量模拟实验优化前后结果对比(从上至下依次为 16/32/64 核)

要关注三个指标:延迟标准差(Latency Standard Deviation,LSD),最大延迟(Maximum Latency,ML)和平均延迟(Average Latency,AL).其中,LSD 反映了报文延迟的均衡程度,LSD 越小表明报文延迟之间的差异性越小,且均衡性越好.ML 反映了所有报文延迟中的最大值,ML 越小表明访问延迟对系统产生的瓶颈作用越小.AL 反映了报文延迟的总体情况.从图 9 中可以看出,在三种片上规模下(4×4/4×8/8×8),相比于优化前的一致性结构,优化后的非一致性结构在三个指标上都有明显提升,并且提高了系统的饱和注入率.同时我们可以观察到随着片上规模的不断增长,系统在 LSD、ML 和 AL 上的优化效果逐步凸显,这是因为在更大的片上规模下报文延迟之间的差异性更加明显,从而使得面向均衡性的设计能够发挥更大的作用.

为了进一步验证优化结构的延迟均衡效果,我们选取 64 核 8×8 网络规模下,针对优化前的饱和注入率(即 0.15)收集了优化前后 10 000 个周期内的报文延迟的数据.图 10 显示的是报文延迟的累积分布函数图.从图 10 中可以观察到,如果我们将延迟大于 50 个周期的报文定义为高延迟访问报文,那么优化后的结构具有更少的高延迟访问报文,延迟小于 50 个周期的访问报文由优化前的 79%提升到优化后的 99%.

图 11 显示的是报文延迟的概率密度分布函数图.

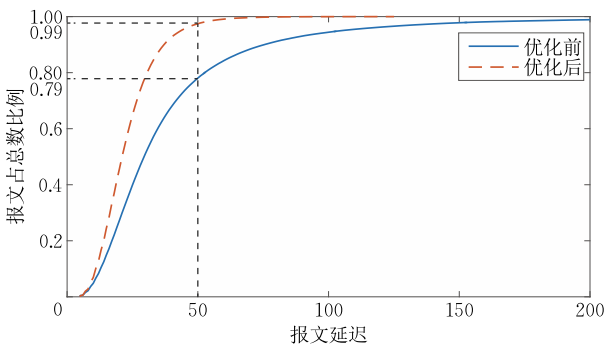


图 10 报文延迟累积分布函数图

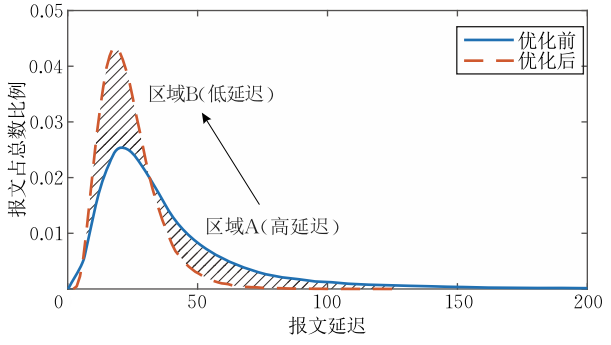


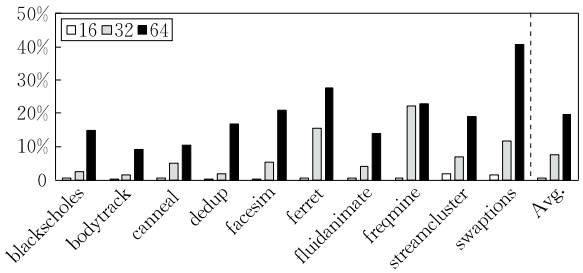
图 11 报文延迟概率密度分布图

从图 11 中可以观察到,优化后结构的报文延迟分布更加集中,并且促使一部分高延迟报文分布(区域 A)转变为低延迟报文(区域 B).因此,优化后的结构能够使得访问报文延迟分布更加集中,进而实现均衡报文延迟的目标.

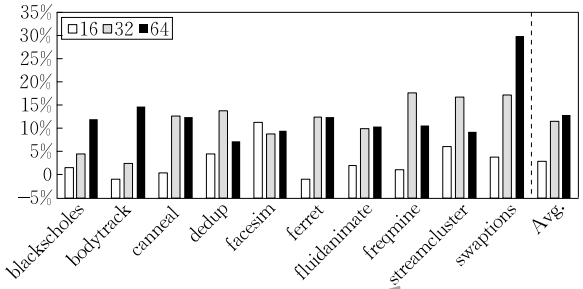
3.3 全系统模拟实验

在全系统模拟实验中,我们按照设计的映射方案修改了存储器到 LLC 的映射,并按照链路的物理通道数量分布设计了相应的网络互连结构.我们从 PARSEC 2.1 基准测试程序集中选取了 10 个典型的应用程序进行全系统模拟实验.在 GEM5 的全系统模式下,模拟器会首先启动运行一个完整的 Linux 操作系统,然后在进入系统后加载应用程序的测试脚本.每一个 PARSEC 应用程序的运行分为三个阶段:初始化阶段,并行运行阶段和结束阶段.其中初始化和结束阶段是串行运行,而并行运行阶段在 PARSEC 中被定义为兴趣区域(Region-of-Interest,ROI).相比于运行整个应用程序,运行 ROI 区域更能够真实客观地检测多线程应用程序的实际性能,因此我们的模拟实验均基于每个应用程序的 ROI 区域.

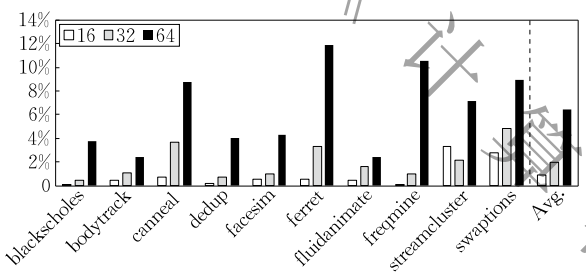
为了验证优化后结构在 Cache 访问均衡性方面的表现,我们收集了不同规模下(16/32/64 核)程序运行过程中的 Cache 访问报文延迟,并分别统计了报文延迟的 LSD、ML 和 AL 相比于优化前降低的比例,如图 12 所示.其中,各个规模下 LSD、ML 与 AL 的最大/平均性能提升如表 2 所示.相比于优化前,优化后的结构在三项指标上在不同规模下都得到了一定程度的降低.优化后的结构在 16/32/64 核规模下在 LSD 上分别平均降低了 0.7%/7.7%/19.6%,在 ML 上分别平均降低了 2.9%/11.6%/12.8%,在 AL 上分别平均降低了 0.9%/2.0%/6.4%.其中 LSD 的降低表明 Cache 访问报文延迟之间的差异性缩小,而 ML 的降低表明制约系统性能的瓶颈延迟变小,进而能够促进系统性能得到提升.但在小规模(16 核)结构下,系统在这三项指标上的改善并不明显,甚至在 ML 指标上对于某些应用程序出现了小幅度上升(bodytrack 和 ferret).这是因为在规模较小的时候 Cache 访问报文延迟之间的差异性较小,因此 Cache 访问均衡性问题并不如规模大时明显,从而限制了优化效果.但随着规模的不断增大,优化的效果随之增强,这得益于在规模增大时 Cache 访问均衡性的问题逐渐凸显.因此可以预见当规模继续增长时,本文设计在均衡性上的优化效果将更加明显.



(a) LSD



(b) ML



(c) AL

图 12　全系统模拟实验下网络性能优化结果

表 2　各个规模下 LSD、ML 与 AL 最大/平均降低百分比统计

规模	指标					
	LSD/%		ML/%		AL/%	
	最大值	平均值	最大值	平均值	最大值	平均值
16	2.0	0.7	11.2	2.9	3.3	0.9
32	22.2	7.7	17.7	11.6	4.8	2.0
64	40.8	19.6	29.9	12.8	11.9	6.4

为了衡量不同规模下 Cache 访问均衡程度，我们选择 Cache 访问延迟的变异系数(Coefficient of Variation，CV)对其进行考察. CV 是 Cache 访问延迟的离散程度的归一化量度，在本文中定义为 LSD 与 AL 之比. CV 越大，表明 Cache 访问延迟的相对离散程度越大. 图 13 所示的是不同规模下运行各个 PARSEC 应用程序统计得到的 Cache 访问延迟的变异系数变化图. 从图 13 可以看出，规模的增大会使得 CV 不断变大，即 Cache 访问延迟分布变得更加离散，导致 Cache 访问均衡性问题逐渐凸显，而这也与我们的分析相吻合.

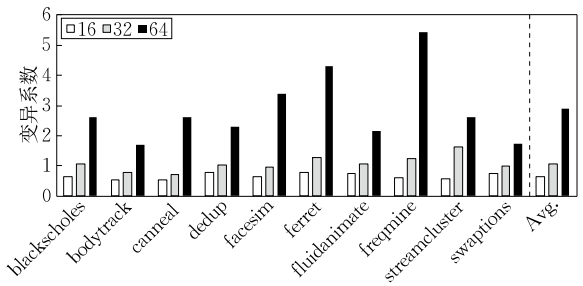


图 13　不同规模下 PARSEC 应用程序的变异系数变化图

图 14 显示的是各个 PARSEC 应用程序的模拟实验中在不同规模下优化后结构的归一化单位周期执行指令数(Instruction Per Cycle，IPC). 从图 14 中可以看出，优化后的结构在不同规模下 IPC 都得到了提升. 在不同规模下(16/32/64 核)，运行各个应用程序最终在 IPC 上分别最大提升了 2.1%/3.9%/14.0%，平均提升了 1.1%/2.1%/6.7%. 可以看出随着规模的扩大，优化后的设计在 Cache 访问均衡性上的影响逐渐显现，对系统性能的提升效果也愈加凸显.

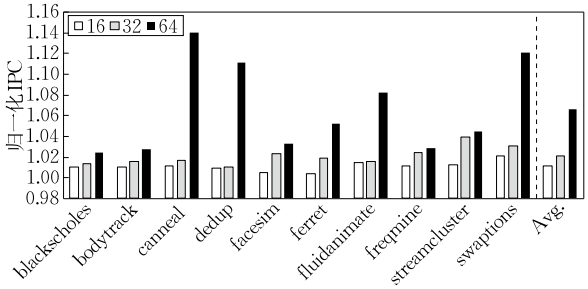


图 14　全系统模拟实验下优化后的归一化 IPC 统计图

4　相关工作

本文基于 S-NUCA 结构提出了面向 Cache 访问均衡性的非一致设计方案. 因此，以下我们将从面向均衡性的共享资源优化技术以及非一致 Cache 体系结构两个方面比较与本文相关的工作.

4.1　面向均衡性的共享资源优化技术

对于共享资源(交叉开关、链路、存储控制器等)的竞争会导致请求延迟的差异性增大[18-21]，而这种差异性会引起部分请求的延迟过大，从而成为系统性能的瓶颈. 因此，从降低差异性的角度出发，许多研究人员在多个方面探索了如何均衡请求延迟并提升公平性. Das 等人[18]通过定义网络报文的 slack 作为衡量一个报文紧急程度的指标，并据此提出了一种新的路由优先策略来提升系统整体性能及公平性. Sharifi 等人[21]指出存储访问延迟的不均衡性会降低系统性能，因此需要均衡存储访问的延迟并减少大延迟存储访问. Sharifi 等人提出了两种机制用于均衡存储访问延迟，机制一通过提升响应报文的

优先级弥补那些具有较高请求报文延迟的存储访问;机制二通过优先响应请求空闲 Bank 的访问请求报文来均衡 Bank 的利用率.另外一些研究人员发现不均衡的缓冲使用策略同样会降低系统性能[22-23].Gorgues 等人[22]提出了一种新的流控机制和路由算法来均衡缓冲器的使用.Li 等人[23]设计了一种新的交叉开关分配策略来缓解缓冲器的不均衡使用.与上述研究的角度类似,本文主要研究片上多核处理器的 Cache 访问均衡性,并依据 Cache 访问延迟均衡程度来提升 Cache 访问之间的公平性,从而提升系统总体性能.

4.2 非一致 Cache 体系结构技术

Arora 等人[6]结合 LLC 与 NoC 两方面的共同设计提出了 FP-NUCA.FP-NUCA 旨在通过针对 LLC 相关报文在路由器中设计特殊的无缓冲快速通路来优先传输,同时设计了相应的 Bank 预测器来减少报文数量.然而,FP-NUCA 的性能提升主要依赖于增设了路由器中的快速通路引擎(Fast path engine)以及 Bank 预测器,这些都显著增加了系统的硬件开销.Amor 等人[24]基于传统 NUCA 结构,通过将 NoC 逻辑上划分为多个层,每一层分配给一级 Cache,并在此基础上设计了网络多播机制以及路由算法,以此降低了 Cache 失效率和网络延迟.虽然将 NoC 划分为多个虚拟层分配给各级 Cache 的设计能够很好地支持级内流量(intra level traffics)传输以降低网络延迟,但却忽视了各个 Bank 节点在网络拓扑中的位置特点,从而可能导致 Cache 访问延迟均衡性的恶化.Lira 等人[25]基于 D-NUCA 提出了 HK-NUCA,该结构实现了一种高效以及更为低开销的 Cache 块搜索机制,能够有效降低失效延迟和网络竞争.Vanapalli 等人[26]在 HK-NUCA 的基础上进一步提出了 HKState-NUCA,该结构下设计的搜索机制能够降低搜索报文数量,并提供更快的 Cache 访问速度和更低的失效率.Beckmann 等人[27]提出了 CDCS,该方法能够在 NUCA 结构下共同调度线程和数据,是一种计算-数据的联合调度机制.HK-NUCA 与 HKState-NUCA 的主要关注点仍是基于 D-NUCA 设计更有效的搜索 Cache 块搜索机制以降低访问延迟,而 CDCS 则致力于通过共同调度线程与 Cache 数据块实现线程与数据块的最优化分配.事实上,以往大多数基于 D-NUCA 的技术仍需依靠复杂的 Cache 块搜索和迁移机制,通过利用 NUCA 结构在 Cache 访问灵活性方面的优点来提升系统性能,却很少关注到 NUCA 结构的缺点.而本文的工作侧重于考虑 NUCA 结构的缺陷,即一致性存储映射和链路分布在非一致 Cache 访问结构下的不足.另外,这些 D-NUCA 技术虽然在 Cache 访问效率上优于 S-NUCA,但硬件开销方面却远不如 S-NUCA 相关技术.

5 结束语

随着基于 NUCA 结构的片上多核处理器规模的逐渐增大,片上网络报文延迟之间的差异性变大,导致 Cache 访问延迟的非一致性更加严重.为了约束 Cache 访问延迟的非一致性,本文在 S-NUCA 结构的基础上,通过理论计算推导出面向 Cache 访问均衡性的非一致的存储映射以及链路分布,缩小 Cache 访问报文延迟之间的差异性,进而改善 Cache 访问延迟的非一致性.本文在 GEM5 体系结构模拟器上,分别在网络模拟和全系统模拟模式下,针对优化前后的结构进行了实验性能比较.在全系统模拟模式下,通过 PARSEC 应用程序的测试表明,使用面向 Cache 均衡性的优化结构能够有效提升系统性能,使得系统 IPC 在 16/32/64 核规模下分别平均提升了 1.1%/2.1%/6.7%.在未来工作中,一方面我们将把面向 Cache 访问均衡性技术从网格网络推广到更多的网络拓扑下的结构,从而使得面向 Cache 访问均衡性技术能够更好地适应不同的网络结构;另一方面我们将研究基于实时流量特征的资源动态分布机制,从而能够在更大范围内适应片上应用特征进一步提升优化效果.

参 考 文 献

[1] Kim C, Burger D, Keckler S W. An adaptive, non-uniform cache structure for wire-delay dominated on-chip caches. ACM SIGPLAN Notices, 2002, 37(10): 211-222

[2] Bienia C. Benchmarking Modern Multiprocessors [Ph. D. dissertation]. Princeton University, Princeton, USA, 2011

[3] Dally W, Towles B. Principles and Practices of Interconnection Networks. San Francisco, USA: Morgan Kaufmann Publishers Inc., 2003

[4] Beckmann B M, Wood D A. Managing wire delay in large chip-multiprocessor caches//Proceedings of the International Symposium on Microarchitecture. Portland, USA, 2004: 319-330

[5] Wang Y, Zhang L, Han Y, et al. Address remapping for static NUCA in NoC-based degradable chip-multiprocessors// Proceedings of the Pacific Rim International Symposium on Dependable Computing. Tokyo, Japan, 2011: 70-76

[6] Arora A, Harne M, Sultan H, et al. FP-NUCA: A fast NoC layer for implementing large NUCA caches. IEEE Transactions on Parallel & Distributed Systems, 2015, 26(9): 2465-2478

[7] Huh J, Kim C, Shafi H, et al. A NUCA substrate for flexible CMP cache sharing. IEEE Transactions on Parallel & Distributed Systems, 2007, 18(8): 1028-1040

[8] Abts D, Jerger N D E, Kim J, et al. Achieving predictable performance through better memory controller placement in many-core CMPs//Proceedings of the International Symposium on Computer Architecture. Austin, USA, 2010: 451-461

[9] Chou C L, Marculescu R. Contention-aware application mapping for network-on-chip communication architectures//Proceedings of the IEEE International Conference on Computer Design. Lake Tahoe, USA, 2012: 164-169

[10] Sahu P K, Chattopadhyay S. A survey on application mapping strategies for network-on-chip design. Journal of Systems Architecture, 2013, 59(1): 60-76

[11] Zhang Y, Jones A K. Non-uniform fat-meshes for chip multiprocessors//Proceedings of the IEEE International Symposium on Parallel & Distributed Processing. Rome, Italy, 2009: 1-8

[12] Binkert N, Beckmann B, Black G, et al. The GEM5 simulator. ACM Sigarch Computer Architecture News, 2011, 39(2): 1-7

[13] Agarwal N, Krishna T, Peh L S, et al. GARNET: A detailed on-chip network model inside a full-system simulator//Proceedings of the IEEE International Symposium on Performance Analysis of Systems and Software. Boston, USA, 2009: 33-42

[14] Miller J E, Kasture H, Kurian G, et al. Graphite: A distributed parallel simulator for multicores//Proceedings of the IEEE International Symposium on High Performance Computer Architecture. Bangalore, India, 2009: 1-12

[15] Carlson T E. Sniper: Exploring the level of abstraction for scalable and accurate parallel multi-core simulation//Proceedings of the 2011 International Conference for High Performance Computing, Networking, Storage and Analysis. Seattle, USA, 2011: 1-12

[16] Sanchez D, Kozyrakis C. ZSim: Fast and accurate microarchitectural simulation of thousand-core systems. ACM Sigarch Computer Architecture News, 2013, 41(3): 475-486

[17] Al-Manasia M, Chaczko Z. An overview of chip multiprocessors simulators technology//Proceedings of the Progress in Systems Engineering. Cham, Switzerland, 2015: 877-884

[18] Das R, Mutlu O, Moscibroda T, et al. Aérgia: Exploiting packet latency slack in on-chip networks//Proceedings of the International Symposium on Computer Architecture. Saint-Malo, France, 2010: 106-116

[19] Kim Y, Papamichael M, Mutlu O, et al. Thread cluster memory scheduling: Exploiting differences in memory access behavior. 2010, 31(1): 65-76

[20] Muralimanohar N, Balasubramonian R. Interconnect design considerations for large NUCA caches. ACM Sigarch Computer Architecture News, 2007, 35(2): 369-380

[21] Sharifi A, Kultursay E, Kandemir M, et al. Addressing end-to-end memory access latency in NoC-based multicores//Proceedings of the International Symposium on Microarchitecture. Vancouver, Canada, 2012: 294-304

[22] Gorgues M, Xiang D, Flich J, et al. Achieving balanced buffer utilization with a proper co-design of flow control and routing algorithm//Proceedings of the International Symposium on Networks-On-Chip. Ferrara, Italy, 2015: 25-32

[23] Li C, Dong D, Liao X, et al. CCAS: Contention and congestion aware switch allocation for network-on-chips//Proceedings of the International Conference on Computer Design. Scottsdale, USA, 2016: 41

[24] Amor H B, Sheibanyrad A, Petrot F. A distributed NUCA architecture using an efficient NoC multicasting support//Proceedings of the Euromicro Conference on Digital System Design. Vienna, Austria, 2017: 184-191

[25] Lira J, Molina C, Gonzâlez A. HK-NUCA: Boosting data searches in dynamic non-uniform cache architectures for chip multiprocessors. Biocatalysis & Biotransformation, 2011, 23(5): 419-430

[26] Vanapalli K, Kapoor H K, Das S. An efficient searching mechanism for dynamic NUCA in chip multiprocessors//Proceedings of the International Symposium on VLSI Design and Test. Ahmedabad, India, 2015: 1-5

[27] Beckmann N, Tsai P A, Sanchez D. Scaling distributed cache hierarchies through computation and data co-scheduling//Proceedings of the International Symposium on High Performance Computer Architecture, Burlingame, USA, 2015: 538-550

WANG Zi-Cong, Ph.D. candidate. His main research interest is networks-on-chip design.

CHEN Xiao-Wen, Ph. D., assistant professor. His main research interests include chip multi-processor design and networks-on-chip design.

GUO Yang, Ph. D., professor, Ph. D. supervisor. His main research interests include microprocessor design and verification.

Background

Non-uniform cache architecture (NUCA) has proposed to support chip multi-processor (CMP) with a large cache capacity. However, along with the scaling up of network size, the degree of non-uniform for cache access latencies will be aggravated, because the distance and latency gap between different cores are increasing. The consequence of more unbalanced cache access latencies is that the latency gap between cache accesses becomes larger, and there will be more cache accesses with overhigh latencies which become the bottleneck of the system.

Many previous research projects aim to improve performance by exploiting the advantage of NUCA, and rarely notice the disadvantage of NUCA. In contrast, our approach pays attention to the inherent problem of NUCA, and strive to over more uniform cache access latencies. In this paper, we study the cache access equalization, and analyze the sources of cache access imbalance. Finally, we propose the design method for NUCA-based CMP with considering the cache access equalization, and perform the evaluation of our design in a full-system simulator.

The work is supported by the National Natural Science Foundation of China under Grant Nos. 61502508 and 61572025, and the Natural Science Foundation of Hunan Province under Grant No. 2015JJ3017.