doi:10.3772/j.issn.1002-0470.2017.02.001

LFF:一种面向大数据应用的众核处理器访存公平性调度机制①

张 洋②*** 李文明* 叶笑春*** 王 达* 范东睿* 李宏亮*** 唐志敏* 孙凝晖*

(*计算机体系结构国家重点实验室(中国科学院计算技术研究所) 北京 100190)

(**中国科学院大学计算机与控制学院 北京 100049)

(***数学工程与先进计算国家重点实验室 无锡 214125)

摘 要 研究了众核处理器的访存公平性问题。针对众核处理器距离访存资源较近的处理单元拥有较大的访存带宽而造成的访存公平性问题,提出了一种面向大数据应用的众核处理器访存公平性调度机制:最少最远(LFF)优先访存。这种机制的原理如下:依据处理单元距离访存资源的距离以及处理单元访存的次数来调度访存顺序,以保证各个处理单元的公平性。首先,访问次数较少的节点被赋予更高的访存优先权。其次,在具有相同访问次数的节点中,距离更远的节点优先访存。再次,在相同距离的节点中,已被选中优先次数少的有优先级。实验评估表明,该调度机制能够有效解决众核处理器的访存公平性问题,其公平性调度效果优于 FR-FCFS,PAR-BS、ATLAS。在 1024 核情况下,系统异步率由 FR-FCFS 的15.5% 降低到1.89%。

关键词 大数据,众核处理器,公平性,调度

0 引 言

在处理大数据应用过程中,由于需要处理的数据量庞大,串行处理难以达到时间上的要求,这造成大数据应用在理论上可以串行执行,但是在实际应用中,由于其不可接受的执行时间无法串行执行的现实[1]。在目前,大数据应用多采用线程级并行(thread level parallelism,TLP)的方式进行加速[2-4]。将一个大规模的应用分解成多个小规模的子任务进行并行,最后汇总子任务的中间结果得出最终结果。线程级并行能够有效缩短大数据应用的整体执行时间,众核处理器众多的处理单元可为大数据应用的线程级并行提供良好的支持平台。线程可以分布在不同的处理单元上并行执行。为了更高效地并行,子任务一般被分解成相似的规模,以期待子任务能同步完成[5-7]。因此子任务具有相互独立、规模相当、行为类似的特点。然而,在并行过程中,由于访存的带宽不同,众核处理器存在着访存公平性问题。即距离访存控制器较近的节点访存效率较高,而距离较远的节点访存延迟较长。

随着众核处理器核数的增加,这种访存公平性问题愈加明显。这造成了即使子任务的规模相近,计算访存比和缓存命中率等程序行为相似,然而由于存在着访存公平性问题而使得子任务不能同步完成。最后一个完成的子任务决定了处理器处理整个大数据应用的最终性能,访存公平性问题关系到最慢子任务的完成时间。因此,可以说解决访存公平性问题对众核处理器大数据应用至关重要。为此,本文提出了一种根据节点位置和访存历史信息进行访存公平性调度的方法:最少最远优先(least and furthest first, LFF)访存调度。该方法根据提出访存

① 国家自然科学基金(61332009)、国家重点研发计划课题(2016YFB0200501)、国家自然科学基金创新研究群体科学基金(61521092)、北京市科委项目(Z151100003615006)资助项目。

② 男,1981 年生,博士生;研究方向:计算机体系结构;联系人,E-mail: zhangyang@ict.ac.cn

(收稿日期:2016-10-14)

请求的节点距离以及所有节点的访存历史信息来调度访存请求。首先根据节点的访存请求次数调度，次数较少的优先访存。对于访存次数相同的请求，根据节点的位置，赋予较远距离的节点更高的访存优先级，以平衡其在片上网络上的延迟。本研究对这种访存公平性调度方法进行了彻底的评估，评估结果显示该方法能有效解决众核处理器在处理大数据应用时的访存公平性问题。在 1024 核情况下，通过该方法，系统异步率可由15.5% 降低到1.89%。

1 相关工作

目前有大量的工作是研究访存调度的问题[8-18]。除了先来先服务（first come first serve，FCFS）之外，为提高访存性能，传统的访存调度多采用行缓冲优先（first ready first come first serve，FR-FCFS）的方式[13,14]，这种方式将行缓冲命中的访存请求优先于其他访存请求，在此基础上，最早的访存请求优先。这种方式提高了访存的性能却没有考虑访存公平性的问题。在大数据并行的应用场景下，由于各个子任务的相似性，这种调度方式和 FCFS 作用差别不大。Nesbit 等人提出网络公平性队列（network fairness queue，NFQ）算法[15]，NFQ 建立在一个 QoS 保证方法基础之上，即占总带宽 $1/N$ 的线程性能不小于其以 $1/N$ 的频率独占总带宽的性能。因此为确保性能上公平，在 NFQ 中 N 个线程平均分配总的访存带宽，则每个线程占用 $1/N$ 的总带宽。NFQ 保证了线程之间的带宽公平性，但是为监测线程带宽而付出的硬件代价较大。同时，针对带宽公平的方法，Mutlu 等人[16]提出简单的为一个线程分配的一定比例的带宽不一定能换来等比例的性能，NFQ 没有考虑行缓冲的状态和 bank 并行性，因此很难直接应用。因此 Mutlu 提出了针对时间公平的访存停顿时间（stall-time fair memory access，STFM）公平策略[16]，在 STFM 中，不公平因子 unfairness 用来表示线程中最大的存储停顿和最小存储停顿的比值，该值越大表示访存不公平越严重，反之越小表示访存越公平。当不公平因子大于某个阈值时，优先处理停顿值大的线程。否则，使用默认的行缓冲优先策略。STFM 并没有考虑并行访问 bank 的调度。并行性感知批处理（parallelism-aware batch scheduling，PAR-BS）策略[17]将访存请求按线程进行批处理以减小线程之间对某单个线程 bank 级并行访存的干扰，同时提供公平性保证和防饿死机制。PAR-BS 考虑了访存的并行性，从性能和公平性两个方面有效地对多线程访存进行了加强。但是 PAR-BS 的批处理量（marking-cap）不容易掌握，批处理量大了会造成不公平现象，处理量少了会影响行缓冲命中率进而影响性能。自适应每线程最少获得服务（adaptive per-thread least attained service，ATLAS）调度[18]通过在所有访存控制器中周期性调度最少获得访存服务的线程优先执行的方式，可以有效提高系统吞吐量。由于这种方法调度的依据是线程访存的服务次数，因此非常适用于相似性较强的大数据并行子任务的公平性调度。但是 ATLAS 需要处理多个访存控制器的协同批处理工作，随着众核核数的增加，线程数和控制器数目均随之增加，其实现访存控制器的协同批处理的开销将变得无法接受。

本文提出一种面向大数据应用的众核处理器访存公平性调度机制，该机制根据访存节点的距离以及访存历史信息来调度访存请求。在调度顺序上，首先根据节点的访存请求次数调度，次数较少的优先访存。对于访存次数相同的请求，本文根据节点的位置，赋予较远距离的节点更高的访存优先级，以平衡其在片上网络上的延迟。本文提出的方法不以线程的服务次数为依据，而以节点的访存历史为依据，大大减少了实现代价。同时，对于访存次数相同的节点，根据访存节点的距离进行调度也使得公平性变得更好。

2 线程核组

对于连接在片上网络之上的核结构，本文设计了一个线程核组（图 1），包括 8 个硬件线程（thread）交替执行以解决控制冲突并隐藏访存延迟。在提出的大数据处理器核结构中，本文在硬件线程之间共享功能部件（shared FU），地址生成单元（AGU），算数逻辑单元（ALU）和指令 cache（I $）部分，但是每

个线程分别使用各自的数据 cache（D $）部分,用来减少线程间 cache 干扰。硬件线程拥有各自独立的寄存器,硬件线程上承载要运行的大数据任务,而硬件线程的调度(dispatcher)在交替流水线中执行,保证核内线程的公平性。多个这样的线程核组结构可以同时并行,以提高处理器的吞吐总量。在模拟过程中,最高达到 1024 个线程核组一起运行,总计 8192 个线程交替执行的规模。线程核组之间由片上网络进行连接,在片上网络边缘连接二级缓存,在二级缓存(L2 $)外连接存储控制器(MC)。

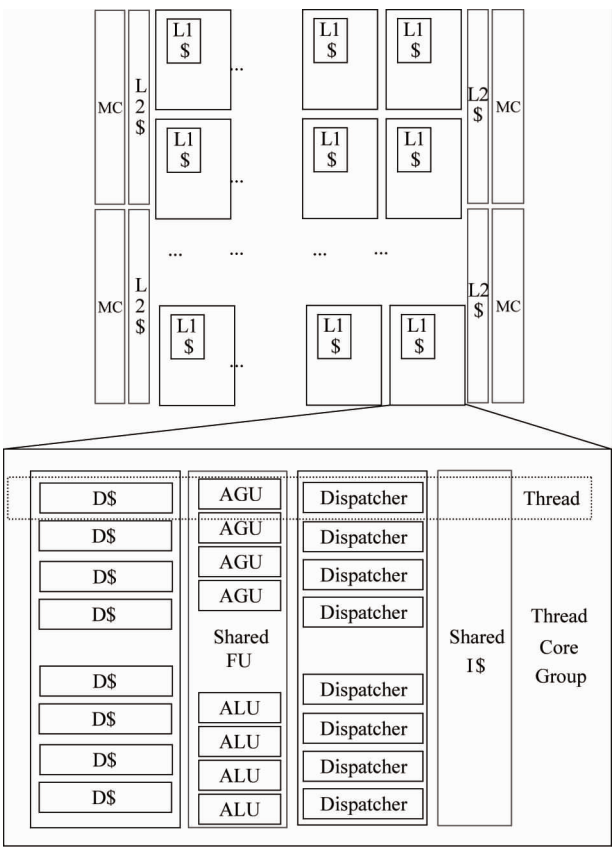

图 1　线程核组

3　最少最远优先(LFF)访存

如果没有公平性调度,大数据应用在运行一段时间后,会产生线程异步偏差,即线程运行进度不同。导致异步偏差的主要原因来自于访存距离的远近不同。距离访存控制器较近的核会用较少的时间获取需要的数据,距离较远的核抵达访存控制器后,要么前边距离较近的访存行为已经结束,要么前边距离较近的访存请求已经排在请求队列当中,并占据靠前的位置。

如图 2 所示,一个 4 ×4 的 mesh 结构为例。访存控制器挂在左上角的核上,核上的数字表示核距离访存控制器的距离。在图 2 中,一个存储控制器连接在 mesh 左上的(0,0)节点上,节点上的数字代表了节点和存储控制器之间的距离。可以看到,访

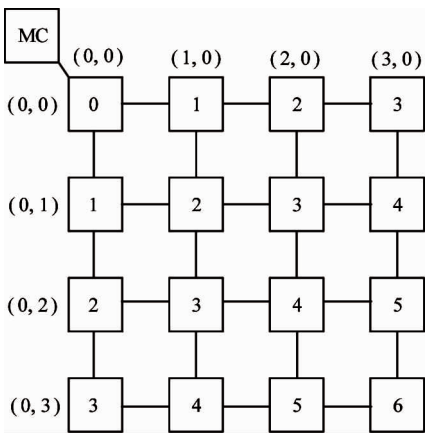

图 2　4 ×4 mesh 上的访存距离

存请求逐级汇总到(0,0)节点,在这种情况下,(0,0)节点拥有 1/2 的访存带宽,(0,1)和(1,0)节点共同拥有 1/4 的访存带宽,而到了节点(3,3)只拥有 1/64的访存带宽。由于大数据子任务行为的相似性,本文假设 16 个核在同一时刻向存储控制器发出访存请求,到达存储控制器的请求在先来先服务(FCFS)顺序下的处理顺序如图 3 所示。

图 4 所示为最少最远(LFF)优先访问队列。图 3 和图 4 中,访存队列中每个数字代表了节点距离访存控制器的距离。在 FCFS 机制下,每个时刻访存队列中最早到达队列的请求将被处理。阴影节点表示即将被服务的访存请求。假设处理访存请求的平均时间间隔为 t,各距离上核的平均访存等待时间为 $\overline{MA_0} = 0t$, $\overline{MA_1} = 1.5t$, $\overline{MA_2} = 4t$, $\overline{MA_3} = 7.5t$, $\overline{MA_4} = 11t$, $\overline{MA_5} = 13.5t$, $\overline{MA_6} = 15t$。

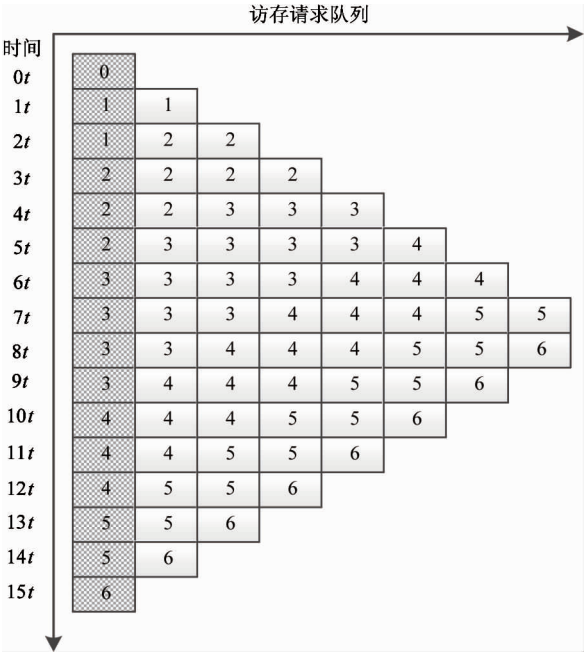

图 3　FCFS 的访存队列

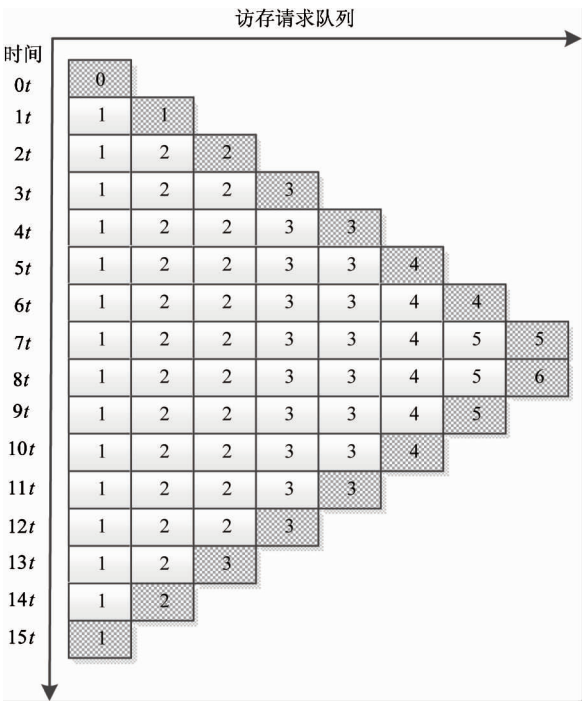

图 4　LFF 访存队列

由于距离远的核访存请求到达访存控制器时间较晚,其在队列中的位置相对靠后。这造成距离访存控制越远的核,其等待时间越长。从而影响了系统整体的同步行为。本文在访存控制器的访存队列中加入调度机制,在最少访存的一批请求中,提升距离较远的核的优先级,使得最少最远(LFF)优先访问。同时在调度历史表中记录相同距离其他核的 ID,以在后面的调度中优先考虑这些延后的核的调度。调度之后的访存顺序如图 4 所示:

在最少最远优先调度下,刚才的例子中各距离上核的平均访存等待时间为 $\overline{MA_0} = 0t$, $\overline{MA_1} = 8t$, $\overline{MA_2} = 9.7t$, $\overline{MA_3} = 7.5t$, $\overline{MA_4} = 7t$, $\overline{MA_5} = 8t$, $\overline{MA_6} = 8t$。FCFS 和 LFF 的访存等待时间对比如表 1 所示,FCFS 顺序下,访存等待时间随着核距离内存控制器的远近而呈线性关系。LFF 机制使得各个核的平均等待时间几乎接近,如果包含回程时间来看,各个核的访存时间差别更小。

表 1　FCFS 和 LFF 的等待时间

	FCFS 等待时间	LFF 等待时间
MA_1	$1.5t$	$8t$
MA_2	$4t$	$9.7t$
MA_3	$7.5t$	$7.5t$
MA_4	$11t$	$7t$
MA_5	$13.5t$	$8t$
MA_6	$15t$	$8t$

4　LFF 访存的结构

LFF 访存调度结构如图 5 所示,在访存控制器和最后一级缓存(last level cache, LLC)之前加入一个访存调度单元。包括两路并行的无序访存队列,一个用来记录各个节点访存信息的记录表,和一个有序的访存队列。在图 5 的两条无序访存队列中,每个请求上的数字代表了访存请求节点的距离以及该节点访存的次数。如 5/2 代表一个距离为 5 的节点发出的第二个访存请求。两条无序访存队列将队列头部的请求查表得知自己的访存历史信息。通过该条信息的历史信息,调度器将这两条请求分别插入到一个已经排好顺序的访存队列中。这条队列按最少最远优先请求的顺序将访存请求发送给 LLC 缓存。由于之前已经排序,在内存控制器只使用 FR-FCFS 进行访问调度。

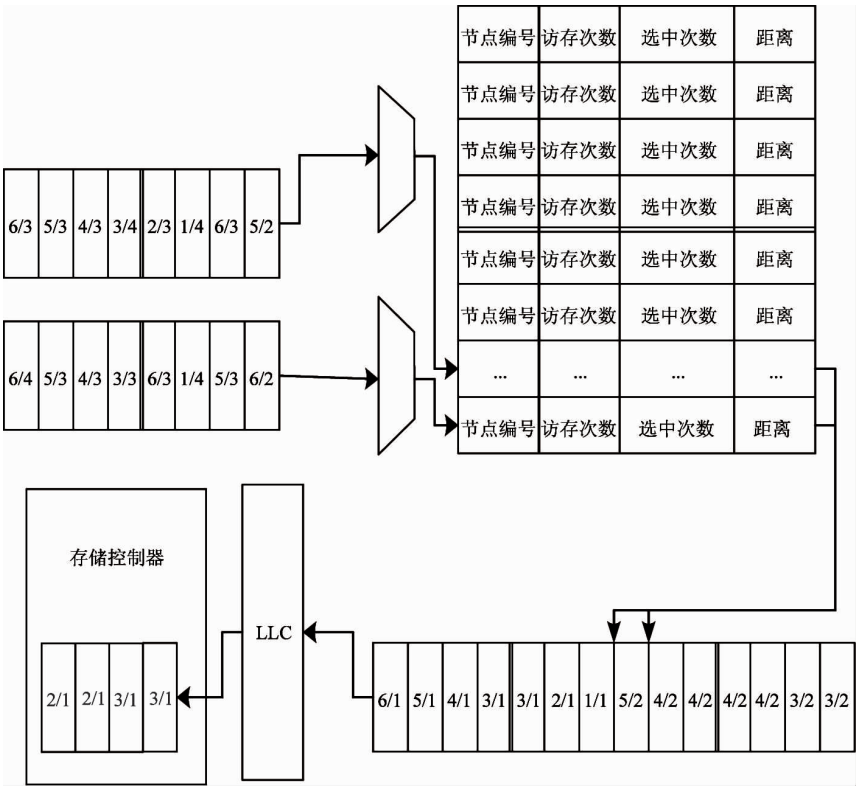

图 5　LFF 调度单元的结构

无序访存队列中在进入 LLC 缓存之前被重新排列成最少最远优先访存的有序序列。排列的依据是根据节点访存信息记录表中的记录。每个节点的访存次数会记录在节点访存信息表中,在同一次调度中,访问次数较少的节点拥有更高级的优先权。在具有相同访问次数的节点中,距离更远的具有更高的优先级。在相同距离的节点中,已被选中优先次数少的有优先级。相同选中次数的节点,随机选中其中一个。流程图如图 6 所示。

节点访存信息表中存储的信息如下:

节点编号	访存次数	选中次数	距离

分别存储了 10 位节点编号(NodeID)、32 位各个节点的访存次数(access times),16 位各个节点被调度器选中的次数(chosen times)以及 6 位点到存储控制器的距离(distance)。所有存储的信息将在访存调度过程中被采用作为调度流程中每个环节被判断的依据。在访存消息中,因为路由以及返回访存结果的需要,源节点 ID(NodeID)的信息被包含在访存信息内,这个信息在片上网络传播时是呈现显

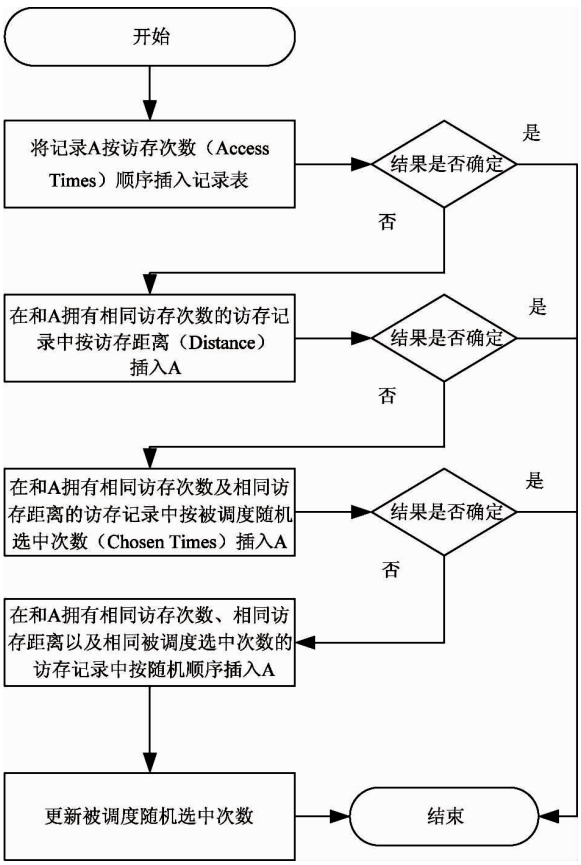

图 6　LFF 流程图

式状态的。被调度器选中的次数(chosen times)指的是节点被调度器赋予更高优先级的次数。其作用是在处理访问次数相同且距离也相同的节点访存请求时,依据它们被调度器选中的次数赋予优先级,被选中次数最少的一个请求将被选中赋予最高优先级。当一条新的信息以随机的形式插入到表中之后,在其出队列的时候,其选中次数值将根据其所在位置被增加。调度结束后,节点访存信息表被更新,以备接下来的调度使用。

由于高通量任务的并行性,在一轮访存操作中,后来到访存控制器的请求一般来自距离较远的节点,调度器中要处理的访存队列在每次选中优先级最高的节点之后,后到的请求会继续占据高优先级位置,而前边的请求会持续保持在低优先级位置,直到一轮访存请求结束。但是有时,下一轮访存请求会在前一轮处理结束之前到来,这时需要通过节点的请求次数来判断,让前一轮的近访存节点尽快得到访存权限,而结束整个一轮的访存,以防止让新一轮的远访存节点得到优先级而造成前一轮访存请求的饿死发生。

5 实验评估

本节评估最少最远(LFF)优先访存调度的作用以及整体上时间可预测调度机制的效果。

5.1 实验设置

采用模拟器 SimICT[19] 来模拟一个高通量众核模拟器的结构。SimICT 是一个模拟大规模结构的模拟框架,其使用了基于组件的设计,使得结构易于搭建以及组件易于重用。所有的体系结构部件包括处理器核,缓存,片上网络路由器,内存控制器以及本文提出的 LFF 调度器均以组件的形式实现。组件通过事先定义好的接口连接起来,通过模拟框架平台来传递消息,以实现时钟精确的模拟。

使用了 8×8、16×16 以及 32×32 三个规模的 mesh 拓扑来评估在不同规模下的最少最远访存调度的性能。Mesh 拓扑中的每个节点拥有一个私有的 I-cache 和 D-cache. 一共连接了 8 个共享的 L2-cache,分布在 mesh 的四个角上。表 2 列出了主要的体系结构模拟参数。

表 2 主要结构模拟参数

主要结构参数	
拓扑	8×8 mesh 16×16 mesh 32×32 mesh
缓存	32k L1 缓存 4M L2 缓存
访存延迟	L1 缓存: 5 周期 L2 缓存: 20 周期 Memory: 100 周期
片上网络的路由算法	维序路由

本文用了 5 个大数据测试用例来评估最少最远访存调度(表 3),前 4 个来自于大数据测试用例集 BigDataBench[20],涉及搜索引擎,社交网络和电子商务等常用的大数据领域。第 5 个测试用例 LP 是运筹学中重要方法,是解决大数据相关问题的重要算法。

表 3 测试用例介绍

Benchmarks	
WordCount	一个计算文档中单词数量的并行算法
Sort	一个大规模键-值对排序并行算法
Grep	一个大规模字符串匹配算法。
Kmeans	一个经典的基于划分的聚类算法
LP	线性规划算法

5.2 评估结果

图 7 和图 8 是一个 64 核运行 100M 的 Grep 任务时的各线程时间分布。如图 7,当使用 FR-FCFS 机制时,和存储控制器比较近的核平均结束得较早,相反,和远离存储控制器的核结束较晚。这给整体调度的同步性带来了一定的损失。图 8 是利用了最少最远优先调度之后的效果,可以看到各任务之间明显在结束时间上达到了有效的平衡。本文用最慢线程用时与最快线程用时的差除以最慢线程用时来表示系统异步率。在 64 核情况下使用最少最远优先访存调度的系统异步率比单纯使用 FR-FCFS 降低了 84%,由原来2.2% 节省为0.3%。在 1024 核情

况下系统异步率由15.5%节省为1.9%。

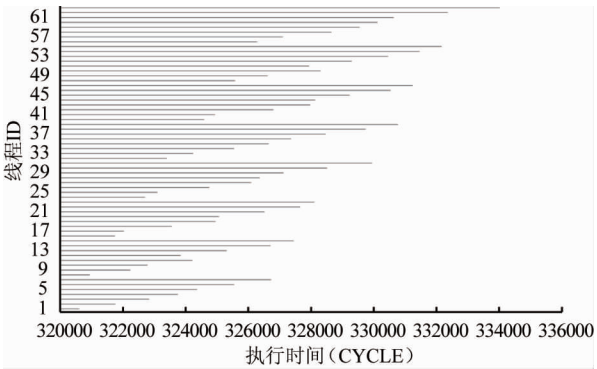

图7　FR-FCFS 调度下的 grep 执行时间

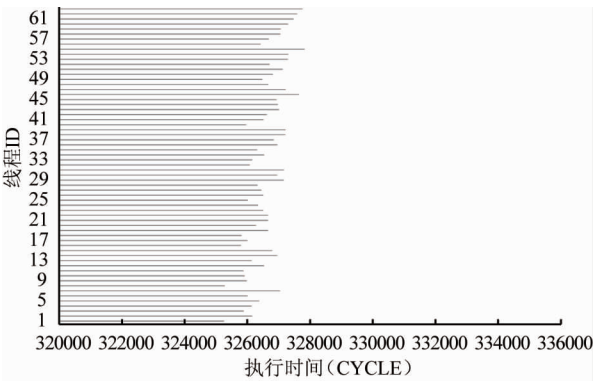

图8　LFF 调度下的 grep 执行时间

图9展示了 FR-FCFS、PAR-BS、ATLAS 以及本文提出的LFF在64核、256核以及1024核情况下的公平性调度效果图。从图中可以看到,随着众核规模的增加,各个线程的异步率开始增加。FR-FCFS 由于只着重提高访存的性能而忽视访存公平性,导致其公平性最差。在1024核情况下 FR-FCFS 系统异步率平均达到15.5%。而 PAR-BS 和 ATLAS 能较好地调度公平性,但是其部署在访存控制器上,由于 LLC 的存在,其调度范围有限,且均没有考虑众核中在片上网络上消耗的时间,因此在最终调度结果上没有最少最远优先访存效果好。在1024核的规模下,PAR-BS 的异步率为6.7%,ATLAS 为5.5%,而最少最远优先访存调度将平均异步率控制在了1.9%。

图10表现了 FR-FCFS、PAR-BS、ATLAS 和 LLF 在1024核规模下各自执行测试用例的时间对比,其结果对 FR-FCFS 进行了归一化处理。从图中可以看出,由于 LFF 更好地提供了公平性,其平均性能比 FR-FCFS 提高8.7%,比 ATLAS 提高3.2%。

LFF 的开销主要体现在第3节中提到的访存调度单元中的访存信息记录表。访存信息记录表的开销和众核的规模相关,在一个1024核的众核结构中,其访存信息记录表有1024行,每行64位。维护和查找信息记录表的主要硬件开销为3条队列,队列长度分别为256、256和512,队列中条记录的大小为38位。访存信息记录表以及维护和查找信息记录表的总开销仅相当于1个12kB 的静态存储器。

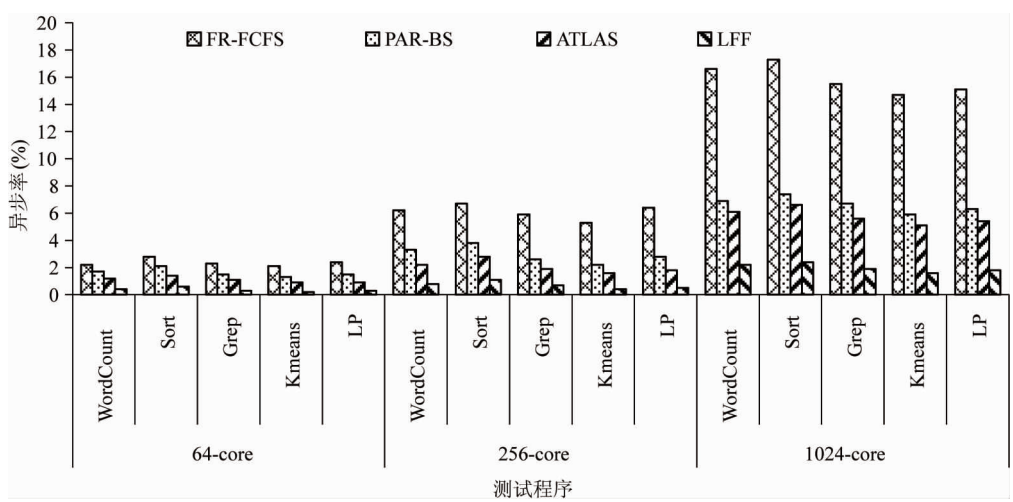

图9　线程异步率比较

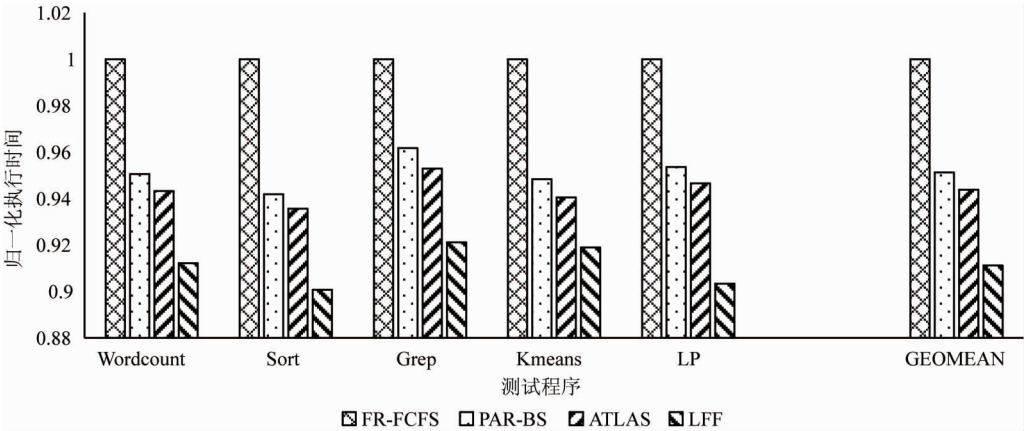

图 10 执行时间比较

6 结 论

本研究提出了一种适用于大数据众核处理器的公平性调度机制。该机制赋予最少访存且最远距离的节点以最高优先级访存。由于考虑了远节点在片上网络上消耗的时间,使得远节点更加接近近节点的访存时间,以此来控制所有节点的访存同步。

实验表明,远距离节点优先访存调度方法可以有效地减少大数据处理器众核的访存异步问题。LFF 的公平性调度效果好于 FR-FCFS、PAR-BS、ATLAS,在 64 核情况下,LFF 的平均系统异步率由 FR-FCFS 的 2.2% 降低到 0.3%。在 1024 核情况下系统平均异步率由15.5% 节省为1.89%。

LFF 公平性调度机制能够有效缓解众核处理器的访存不公平问题。访存公平性对于提高在众核处理器上各核之间访存子任务分布比较均匀的大数据应用的性能至关重要,但是对于任务分配不均匀的应用,仅仅考虑访存阶段公平性还是不够的,除了考虑访存的公平性之外任务进度的公平性也应被包含到调度当中,这将是我们在未来进一步要做的工作。

参考文献

[1] Kishor D. Big data: The new challenges in data mining. *International Journal of Innovative Research in Computer Science & Technology*,2013,1(2): 39-42

[2] Hong S, Kim H. An analytical model for a GPU architecture with memory-level and thread-level parallelism awareness. In: Proceedings of the 36th Annual International Symposium on Computer Architecture, Austin, USA,2009. 152-163

[3] Chen X, Aamodt T. A first-order fine-grained multithreaded throughput model. In: Proceedings of the 15th International Symposium on High Performance Computer Architecture, Raleigh, USA,2009,329-340

[4] Yang Y, Zhou H. CUDA-NP: Realizing nested thread-level parallelism in GPGPU applications. In: Proceedings of the 19th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming, Orlando, USA, 2014. 93-106

[5] Dai W, Ji W. A MapReduce implementation of C4.5 decision tree algorithm. *International Journal of Database Theory and Application*,2014,7(1): 49-60

[6] Dhillon S, Kaur K. Comparative study of classification algorithms for web usage mining. *International Journal of Advanced Research in Computer Science and Software Engineering*,2014,4(7): 137-140

[7] He Q, Zhuang F, Li J, et al. Parallel implementation of classification algorithms based on MapReduce. In: Proceedings of the 5th International Conference on Rough Set and Knowledge Technology, Beijing, China, 2010. 655-662

[8] Chou Y, Fahs B, Abraham S. Microarchitecture optimizations for exploiting memory-level parallelism. In: Proceedings of the 31st Annual International Symposium on Computer Architecture, Munich, Germany,2004. 76-87

[9] Gabor R, Weiss S, Mendelson A. Fairness and throughput in switch on event multithreading. In: Proceedings of the 39th Annual IEEE/ACM International Symposium on Microarchitecture, Orlando, USA,2006. 149-160

[10] Hur I, Lin C. Adaptive history-based memory schedulers. In: Proceedings of the 37th Annual IEEE/ACM International Symposium on Microarchitecture, Portland, USA,2004. 22-29

[11] Iyer R, Zhao L, Guo F, et al. QoS policies and architecture for cache/memory in CMP platforms. In: Proceedings of the 2007 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems

(SIGMETRICS), San Diego, USA, 2007. 25-36
[12] Luo K, Gummaraju J, Franklin M. Balancing throughput and fairness in SMT processors. In: Proceedings of the 2001 International Symposium on Performance Analysis of Systems and Software, Tucson, USA, 2001. 164-172
[13] Rixner S, Dally W, Kapasi U, et al. Memory access scheduling. In: Proceedings of the 27th International Symposium on Computer Architecture, Vancouver, Canada, 2000. 128-138
[14] Zuravleff W, Robinson T. Controller for a Synchronous Dram that Maximizes Throughput by Allowing Memory Requests and Commands to Be Issued out of Order. US Patent Number 5,630,096, May 1997
[15] Nesbit K, Aggarwal N, Laudon J, et al. Fair queuing memory systems. In: Proceedings of the 39th Annual IEEE/ACM International Symposium on Microarchitecture, Orlando, USA, 2006. 208-222
[16] Mutlu O, Moscibroda T. Stall-time fair memory access scheduling for chip multiprocessors. In: Proceedings of the 40th Annual IEEE/ACM International Symposium on Microarchitecture, Chicago, USA, 2007. 146-160
[17] Mutlu O, Moscibroda T. Parallelism-aware batch scheduling: Enhancing both performance and fairness of shared DRAM systems. In: Proceedings of the 35th International Symposium on Computer Architecture, Beijing, China, 2008. 63-74
[18] Kim Y, Han D, Mutlu O, et al. ATLAS: A scalable and high-performance scheduling algorithm for multiple memory controllers. In: Proceedings of the 16th International Symposium on High-Performance Computer Architecture, Bangalore, India, 2010. 1-12
[19] Ye X, Fan D, Sun N, et al. SimICT: A fast and flexible framework for performance and power evaluation of large-scale architecture. In: Proceedings of the International Symposium on Low Power Electronics and Design, Beijing, China, 2013. 273-278
[20] Wang L, Zhan J, Luo C, et al. BigDataBench: A big data benchmark suite from internet services. In: Proceedings of the 20th IEEE International Symposium on High Performance Computer Architecture, Orlando, USA, 2014. 488-499

LFF: A many-core processor's access fairness scheduling scheme for big data applications

Zhang Yang[*,**], Li Wenming[*], Ye Xiaochun[*,***], Wang Da[*], Fan Dongrui[*], Li Hongliang[***], Tang Zhimin[*], Sun Ninghui[*]

([*] State Key Laboratory of Computer Architecture (Institute of Computing Technology, Chinese Academy of Sciences), Beijing 100190)

([**] School of Computer and Control Engineering, University of Chinese Academy of Sciences, Beijing 100049)

([***] State Key Laboratory of Mathematical Engineering and Advanced Computing, Wuxi 214125)

Abstract

The memory access fairness problem of many-core processors was studied. Aiming at the memory access fairness problem that many-core processor's process units closer to the memory controller have higher memory access bandwidth, this study proposed a many-core processor's memory access fairness scheduling scheme for big data applications, called the least and furthest first (LFF) access scheme. This scheme schedules the order of memory access requests according to the distance from a processing unit to its access resource and a processing unit's access history. Firstly, the highest memory access priority is assigned to the nodes with least access request times. Secondly, for the nodes with same access request times, the furthest nodes access memory preferentially. Thirdly, among the nodes with the same distance, the nodes with less priority-assigned history access memory firstly. Our evaluation shows that the proposed scheme can efficiently solve the memory access fairness problem in many-core processors, and its effectiveness in fairness scheduling outperforms the schemes of FR-FCFS, PAR-BS and ATLAS. In the case of 1024 cores, the execution asynchronous rate was reduced to 1.89% from 15.5% compared with the FR-FCFS.

Key words: big data, many-core processor, fairness, scheduling