doi:10.3772/j.issn.1002-0470.2015.06.004

基于神经网络预测模型的异构多核处理器调度①

王 磊②*** 陆 超*** 章隆兵*** 王 剑***

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

(**中国科学院大学 北京 100049)

(***龙芯中科技术有限公司 北京 100190)

摘 要 为了提高异构多核处理器的性能和资源利用率,研究了优化异构多核处理器的程序调度方法。针对异构多核处理器的特点,提出了一种基于神经网络的低开销程序性能预测的调度模型。该调度模型根据程序固有特征预测各个程序在不同处理器核上的性能,然后根据性能预测找出程序与处理器核之间的最优匹配方案进行调度。试验证明,该调度模型对于异构多核处理器的性能和能效都取得了很好的提升效果,超过了现有的轮转调度、抽样调度和性能影响评估(PIE)调度。相比于轮转调度,该调度模型在处理器性能和能效上分别取得了13.64%和10.78%的提升。

关键词 异构多核处理器,多道程序,程序固有特征,神经网络预测模型,基于神经网络的调度模型

0 引 言

异构多核处理器出于对面积、功耗和性能等方面的综合考虑,在一个片上系统集成多种不同类型的处理器核,以满足不同程序各自的硬件资源需求。异构多核处理器所面临的一个最重要的问题是如何把多道程序(multi-programmed workload)中并行运行的程序各自调度到最适当的处理器核上运行,从而提升系统资源利用率。异构多核处理器的异构性通常表现在指令集、发射宽度、缓存大小以及其它基本的微架构上。本文主要关注单指令集异构多核处理器。在单指令集异构多核处理器中,所有的处理器核共享一套指令集,程序可以不经修改地在任何核上正确执行,从而给调度带来了极大灵活性。常见的单指令集异构多核处理器产品包括NVidia公司的Kal-EI[1]以及ARM公司的big. LITTLE[2]等。

当前操作系统采用完全公平调度(completely fair schedule,CFS)策略,假设底层硬件能够提供完全相同的性能,通过轮转调度(round robin schedule)的方式使得所有程序公平共享所有处理器核[3]。轮转调度适用于同构多核处理器环境。为了解决异构多核处理器中多道程序的调度问题,Kumar等[4,5]提出了一种两阶段启发式抽样调度:抽样阶段试运行程序与处理器核所有可能的匹配方式;稳定阶段挑选抽样阶段中表现最优的匹配方式作为调度决策。抽样调度的缺陷在于确定最优调度方案前需要进行抽样,这会导致频繁的线程迁移以及大量运行时间被消耗在次优调度方案。为了避免抽样调度的开销,随后的研究利用缓存失效率等访存信息作为调度依据[6-11]。这类调度模型根据访存信息将程序分为计算密集型和访存密集型,计算密集型的程序

① 国家"核高基"科技重大专项课题(2009ZX01028-002-003,2009ZX01029-001-003,2010ZX01036-001-002,2012ZX01029-001-002-002),国家自然科学基金(61221062,61100163,61133004,61173001,61232009,61222204,61432016)和863计划(2012AA010901,2012AA011002,2012AA012202,2013AA014301)资助项目。

② 男,1986年生,博士生;研究方向:计算机体系结构,多核共享资源调度管理;联系人,E-mail:wanglei-cpu@ict.ac.cn

(收稿日期:2014-12-25)

被调度到复杂的处理器核上运行以改善性能,访存密集型的程序被调度到结构简单的处理器核上运行以降低系统功耗。访存信息为程序在不同处理器核上的性能收益提供了粗略的估算,Vraeynest 等[12]论证了只依靠访存信息会导致次优的调度,并进一步提出了基于访存并行度和指令并行度的性能影响评估(performance impact estimation,PIE)调度,PIE 调度通过更精确地预测任意给定程序在不同类型的处理器核上的性能来做出更优的调度决策。无论是基于访存信息的调度还是 PIE 调度,都是基于程序行为信息(如缓存失效率)来预测程序在不同处理器核上的性能,而程序行为与具体的处理器微体系结构配置相关(如缓存架构),由程序自身特性和具体的执行环境共同决定。这导致在一个处理器核上采集的程序行为信息很难直接用于预测该程序在不同处理器核上的性能。用程序行为构建性能预测模型增加了模型的复杂度,同时也影响了性能预测模型的预测能力。一些研究工作发现微体系结构无关的程序特征能够准确反映程序的固有特征[13]。程序固有特征(区分于程序行为)在编译阶段确定与具体的处理器微体系结构无关,可用来有效区分软硬件因素及其交互作用对程序性能的影响,从而简化了性能预测模型的构建[14]。另外,由于程序固有特征是微体系结构无关的,使得在一个处理器核上采集到的程序固有特征可以直接用于预测该程序在另一个处理器核上的性能。本研究采用当前流行的机器学习算法,首先从大量的程序固有特征中提取出 13 种对于程序性能有着较大影响的关键特征,然后基于提取的程序固有特征,构建了神经网络性能预测模型,相比于 PIE 调度的预测模型,该模型取得了更低的平均绝对预测误差 9.68%(PIE:14.57%)。试验证明,本文提出的基于神经网络预测模型的调度模型比轮转调度、抽样调度和 PIE 调度都取得了更高的性能和能效提升。

1 异构多核调度框架

在这一节中,我们首先介绍异构多核处理器调度的整体框架及其各组成单元的作用。异构多核调度的整体框架示例如图 1 所示。一个片上系统集成了多个类型的处理器核,由多个单线程程序组成的多道程序并行运行在系统中。由于不同程序有不同的硬件资源需求,甚至同一个程序的硬件资源需求也会随着程序特征改变而改变[15]。因此,需要根据程序各自的硬件资源需求动态地将其分别调度到最适当的处理器核上运行,从而提高系统资源利用率。最直接的方法是运用抽样调度[4,5],但是抽样调度开销过大。为了避免这种开销,调度模型需要能够预测各个程序在不同处理器核上的性能,并根据预测结果做出调度决策。

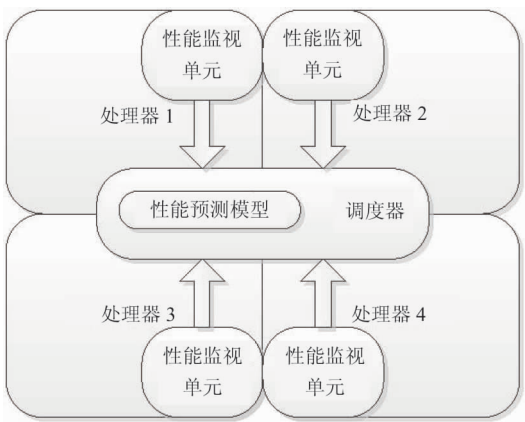

图 1 异构多核调度框架示例

进行性能预测通常需要在线获取一些程序信息作为依据。当前已经有各种软硬件方法可以帮助我们在线获取各种程序信息,例如 Linux 的性能分析工具以及 Intel/ARM 的性能监视单元(performance monitor unit,PMU)。在龙芯 3A 上也实现了类似的 PMU,并定义了包括程序行为和程序固有特征在内的 32 种可计数事件可供使用。如引言所述,本文主要基于程序固有特征进行性能预测,图 1 中各处理器核上的 PMU 用于采集程序的程序固有特征。

调度器中包含一个性能预测模型。性能预测模型根据运行在各个处理器核上的程序的固有特征分别预测其运行在其它处理器核上的性能。调度器根据预测结果做出最优的调度决策,并引发中断将调度决策通知给操作系统。操作系统根据调度决策将各个程序分别调度到目标处理器核的任务队列中,从而完成一次调度。

2　性能预测模型

在整个异构多核调度模型中,核心单元是性能预测模型,性能预测的准确度直接影响到调度决策的优劣。然而随着现代处理器复杂度的增加,根据程序信息预测程序在特定处理器上的性能越来越困难,困难主要体现在:(1)在众多的软硬件因素中,哪些因素对程序性能的影响最大;(2)这些因素以怎样的方式(线性/非线性)影响程序在特定处理器核上的性能。

传统的解决方案是借助于研究人员丰富的领域知识,人工挑选出最能影响程序性能的因素,然后设计相应的预测模型及相关参数[9,11,12]。然而在现代处理器中影响程序性能的因素实在太多,这些因素之间又互相影响,并最终与程序性能间形成十分复杂的非线性关系,导致传统的解决方案存在工作量大,可移植性和扩展性差等缺点。

为了解决上述问题,本文选择借助当前流行的机器学习算法,从大量程序固有特征中自动挑选出关键特征并构造相应的预测模型。我们的性能预测模型通过离线训练完成,性能预测模型的离线训练包括两个阶段:关键特征提取和预测模型构建,其中关键特征提取是为了进一步构建预测模型而服务的。虽然目标不同,但这两个阶段有类似的训练流程,如图2所示。

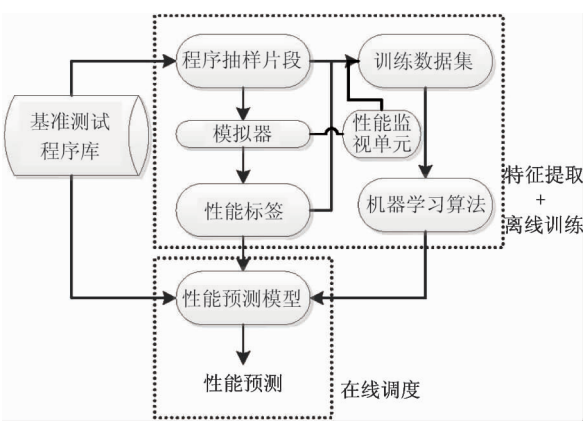

图2　预测模型离线训练流程

为了进行关键特征提取和性能预测模型的离线训练,首先需要构建一个足够有代表性的数据集来进行训练和测试。我们选择SPEC CPU2006基准测试程序构成数据集。为了捕捉更细粒度的程序特征,将单个程序切分为100~200个固定指令数的程序片段(例如,每个程序片段100万条指令),总共得到2000个程序片段。然后将这些程序片段在特定微体系结构配置的周期精确的模拟器上运行并获得相应的性能功耗标签,每个程序片段的程序固有特征向量以及相应性能功耗标签构成一个数据项。我们把全部数据项按照4:1的比例随机划分为训练样本集和测试样本集,分别用于训练预测模型和测试预测模型的准确度。

2.1　关键特征提取

在离线训练阶段,可以用各种软硬件工具获取大量的程序固有特征,它们都会对程序在处理器上的性能产生各种影响。本文借助专门用于采集微结构无关的程序特征的MICA[16]等工具抓取了35项不同的程序固有特征组成特征向量。然而,要在线获取全部的这些程序特征对于调度来说开销过大。因此,需要从大量的程序固有特征中提取对于程序性能有关键影响的特征子集作为性能预测的依据。

为了提取出对于程序性能有着关键影响的特征子集,本文选用迭代回归树模型,目前应用最广的机器学习算法之一。迭代回归树模型将多棵回归树组合起来,每棵树学习之前所有回归树预测值之和的残差(残差是所有树的预测值之和与真实值间的差值),从而最终组合成一个精度较强的学习模型。迭代回归树模型避免了单棵回归树过拟合的问题,具有良好的泛化能力。然而我们选择迭代回归树模型更重要的原因在于其能够处理高维度的数据,不用做特征选择,并且在训练完成后,能够识别哪些特征比较重要。

迭代回归树的训练阶段是个迭代过程,除第一棵树外,之后的每棵树都是在预测之前全部回归树的预测残差。对于每棵回归树,迭代回归树模型采取对特征和样本双重随机采样的策略,从而每棵回归树只跟少量的程序特征相关,但是大量回归树组合起来可以覆盖全部程序特征,因此可以不做特征选择地处理高维度数据。每棵树在采样的数据上训练完成后根据预测对全部数据集重新计算残差,即

用数据集的当前目标值减去这棵树的预测值得到新的目标值。在预测阶段,待预测数据分别由每棵树各自进行预测,然后将所有树的预测值加权累加得到最终预测值。由于每棵树所用到的随机采样特征各不相同,不同回归树对于最终预测值的贡献不同。可以认为,预测能力更强的回归树所用到的程序特征对于程序性能有着更关键的影响,因而可以帮助我们提取关键程序特征。

迭代回归树模型中每棵树的深度表示树的复杂度。在本文中,我们设定每棵树的深度为 4(即每棵树随机采样 3 个程序特征),最终的迭代回归树模型由 300 棵回归树组成。我们从预测能力最强的 20 棵树所用到的程序特征(有重复)中挑选被用得最多且意义明确的 13 种程序固有特征作为关键程序特征:系统调用,主存读操作,主存写操作,分支预测指令,跳转控制指令,整型指令,浮点指令,单指令多数据流扩展指令,寄存器操作,数据重用距离,指令重用距离,指令级并行度,其它指令(所有未单独统计指令之和)。

2.2 神经网络预测模型

提取出关键程序固有特征后,需要根据这些特征进一步构建性能预测模型。实际上,2.1 节用到的迭代回归树模型本身就是一个很好的性能预测模型,但是迭代回归树模型是个组合模型,需要几百棵回归树共同预测得到最终预测值,这对于需要以较低代价快速做出决策的调度来说显得开销过大。为了构建一个更简单实用的性能预测模型,我们选择另一种被广泛使用的机器学习算法:人工神经网络。人工神经网络模型的结构示例如图 3 所示。

人工神经网络模型是一个非线性动力学习系统,能够隐式地检测程序固有特征与性能间复杂的非线性关系。借助于反向传播算法[17],人工神经网络具有自适应与自组织能力,在训练过程中通过自动调节权重进行学习。通常,人工神经网络模型拓扑结构包括输入层、隐层和输出层。各层神经元与相邻层神经元之间相互全相连,同层神经元之间无连接。为了使得预测模型的泛化能力更强,我们采用了交叉验证来构建神经网络预测模型。

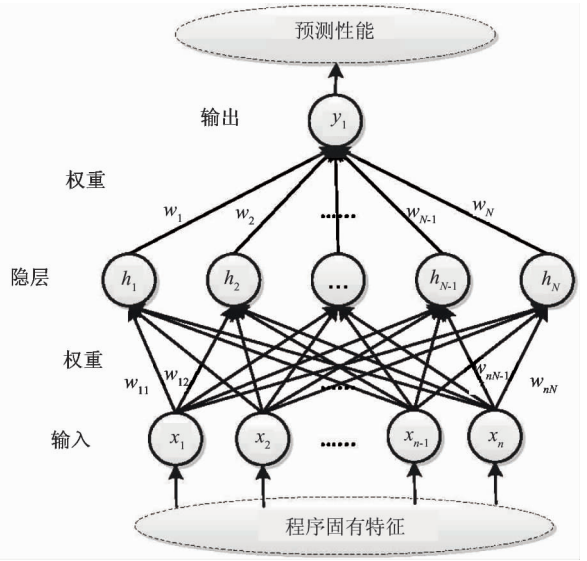

图 3 人工神经网络模型结构示例

3 试验设置

这一节介绍我们用于仿真和验证调度模型效果的试验设置。

试验中,我们考虑了 4 种异构的处理器核,表 1 中列出了主要的异构配置参数。在各个处理器核的参数配置上首先参考了常见的各类主流商用处理器[1,2];其次,为了尽量覆盖不同程序的硬件资源需求,调节了发射宽度、私有缓存大小、定浮点寄存器等对程序性能影响较大的基本配置参数,在周期精准的 gem5 模拟器[18]上进行仿真,功耗通过 McPAT 工具[19]进行评估。

表 1 处理器核的微体系结构配置

Cores	core1	core2	core3	core4
发射宽度	6	6	4	2
ROB 大小	192	128	64	32
频率	2GHz	2GHz	1GHz	1GHz
寄存器	128	96	96	64
一级缓存	64kB	32kB	32kB	16kB
	4 way	4 way	2 way	2 way

为了在多种类型的程序上充分验证调度模型的效果,我们从 SPEC CPU2006 基准测试程序集中选出 12 个具有代表性的基准程序来评估我们的方案,采用 train 输入集。这 12 个基准程序包括:perl-

bench, bzip2, h64ref, hmmer, libquantum, milc, leslie3d, namd, soplex, GemsFDTD, lbm, sphinx3。它们覆盖了整型、浮点型、计算密集型和访存密集型等各种类型的程序类型。我们把这些基准程序随机混合成不同的多道程序并行执行在异构多核处理器上进行动态调度,最终计算多组程序调度的平均优化结果。

调度过程中,需要根据调度决策将各个进程迁移到目标处理器核上继续运行。进程迁移通常会带来性能开销,主要来自于上下文切换(context switch)和私有缓存冷启动效应(cache warmup)。其中上下文切换完成处理器状态(寄存器状态等)从当前处理器核到目标处理器核的复制,通常只带来固定上百个周期的开销[6]。而私有缓存的冷启动效应是由于进程到了新的处理器核需要重新向私有缓存加载数据。假设私有缓存中的所有数据都需要从共享缓存中取回重填,对于 64KB 大小的私有缓存,会带来大概两万周期的性能开销[6]。当然实际情况通常只有部分最近需要用到的数据需要重载。多项研究[3,6,12]对进程迁移在不同调度粒度下的性能开销进行了量化分析,研究发现,通常调度粒度大于 2.5ms 时,进程迁移带来的性能开销相对于调度所能带来的性能提升可以忽略不计。基于这个结论,本文采用 10ms 的调度粒度,从而忽略进程迁移所带来的影响。

每次仿真,组成多道程序的所有程序同时进入系统并各自运行在不同处理器核上。在每个调度区间入口,调度器根据上一阶段的性能预测结果作出新的调度决策,并将各个程序调度到目标处理器核上运行,进入下一个调度间隔。每次仿真包括 20 个调度区间,即持续运行 200ms。整个调度流程如图 4 所示。

图 4　调度流程

4　试验结果

在这一节中,我们首先评估神经网络预测模型的准确度和实现开销,然后将其应用于异构多核调度。作为对比,我们也在相同试验环境下验证了轮转调度,抽样调度和性能影响评估(PIE)调度。试验表明,我们的神经网络预测模型准确度优于当前水平的 PIE 预测模型,进一步,基于该神经网络预测模型的调度模型比轮转调度、抽样调度和 PIE 调度都取得了更高的性能和能效提升。

4.1　神经网络预测模型的准确性

为了衡量性能预测模型的准确度,我们使用平均相对预测误差(average relative prediction error)这一评估指标。平均相对预测误差由

$$\frac{1}{N}\sum_{i=1}^{N}(P_i - T_i)/T_i$$

来计算,其中 P_i 表示性能预测模型给出的预测性能,T_i 表示在周期精确模拟器上运行得到的真实性能,N 表示总的测试样本数。平均相对预测误差的数值越小,则表示性能预测模型越准确。

作为对比,我们在本文的试验环境下实现了 PIE 预测模型,并分别把两个性能预测模型用于 SPEC CPU2006 基准测试程序片段测试样本集的性能预测。如图 5 所示,两个性能预测模型对不同的基准程序预测能力各有优劣,但总的来看,神经网络预测模型在更多的基准程序上表现出更好的预测精度。对整个测试样本集,PIE 预测模型的平均相对

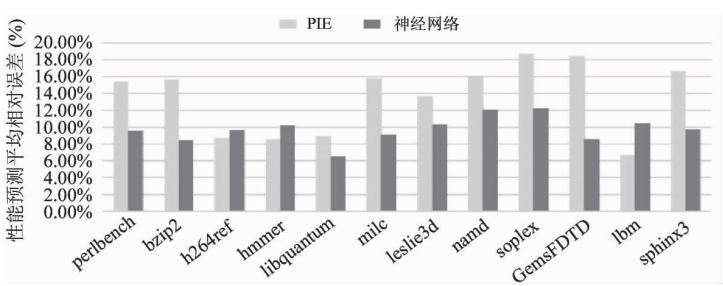

图 5　性能预测模型平均相对误差对照

误差为 13.60%,而神经网络预测模型只有 9.73% 的平均相对误差。

4.2 神经网络预测模型的实现开销

神经网络预测模型既可以在操作系统层面软件实现也可以通过独立的硬件模块实现。将神经网络预测模型以硬件的方式实现能够避免跟运行中的程序抢占处理器核,调度的时间开销可以被覆盖。本节讨论神经网络预测模型的硬件实现开销。

根据同一个程序的程序固有特征预测其在不同处理器核上的性能,需要在离线阶段分别为每个处理器核单独训练一个预测模型,各个处理器核的微体系结构配置作为隐参数被包含在模型内部的权重中。所有的预测模型具有相同的结构,所不同的只是其各自的权重。因此,在异构多核系统中只需要实现一个硬件神经网络,并分别保存每个处理器核预测模型的权重即可。神经网络的结构如 2.2 节图 3 所示,由输入层、隐层和输出层构成。其中输入层包括 13 个神经元(对应 13 个程序固有特征),一个输出神经元(对应预测性能),在模型的预测准确度和开销之间权衡后,我们把隐层包含的神经元个数设定为 10 个。因此,神经元总共需要 $(13+10+1)\times 32=768$ 位寄存器开销。另外,各层神经元与相邻层神经元之间相互全相连,每个连接对应一个权重,将权重保存在代价更低的 SRAM 中,总共需要 $(13\times 10+10)\times 32\times N$ 位,其中 N 与异构多核系统中处理器核的种类相关。除了这些存储开销外,还需要一些硬件的乘加单元用于内部运算。

相比于性能影响评估(PIE)调度,神经网络预测模型与之共有的开销是都需要性能监视单元(PMU)在线获取各种程序信息用于性能预测。不同之处在于,PIE 调度中,不同处理器核之间的程序性能预测是通过一组固定参数的线性表达式完成的,所涉及到的参数个数以及乘加运算相对于神经网络预测模型更少,因此,单个 PIE 预测模型的硬件开销低于神经网络预测模型。然而,PIE 调度需要为每两类处理器核间的性能转换分别设计两个线性表达式,也就是在一个包含 N 类处理器核的异构多核系统中,共需要用到 $A_N^2 = N\times(N-1)$ 个表达式来进行性能预测,其涉及的硬件开销随着核的种类按照 $O(N^2)$ 的复杂度增加。与之相对的,神经网络预测模型只需要对每一类处理器核保存一组权重参数,从而随着处理器核种类的增加神经网络预测模型具有更好的可扩展性。

为了进一步评估硬件神经网络模型的面积和功耗开销,本文在 Synopsys 工具上对其进行了综合,其各个部分以及总的开销如表 2 所示。可以看出,该硬件单元带来的开销很低。

表 2 硬件神经网络面积/功耗实现开销

组成	面积(mm^2)	功耗(mW)
Memory	0.098881	1.2283
Combinational	0.010896	1.6101
Clock _ network	0.008252	1.7289
Register	0.026284	3.5261
总和	0.144313	8.0934

4.3 调度结果及分析

在这一节中,本文分别以系统性能和能效为优化目标在 gem5 模拟器上进行了仿真调度。系统性能用系统吞吐量(单位时间内整个系统完成的指令数之和)来度量,而能效则用系统吞吐量和功耗的比值,即性能功耗比来度量。

作为对比,本文也分别实现了轮转调度,抽样调度以及性能影响评估(PIE)调度。PIE 调度和本文的调度模型都是根据性能预测结果选择程序与处理器核之间最优(最大化系统性能或能效)的匹配方案进行调度,调度结果的差异主要来自于预测模型的精度,错误的预测通常会带来不好的调度结果。

以轮转调度作为基准,图 6 给出了其它三种调度模型相对于轮转调度对系统性能和能效的平均提升效果。轮转调度使得所有处理器核在程序间平等共享,但是这种调度方式忽略了程序和处理器核各自的特点和需求,无法充分发挥异构多核处理器的优势。抽样调度通过抽样找出最优的调度方案并执行,通常表现会优于轮转调度,但是在抽样阶段会带来一些性能开销,抽样调度相对于轮转调度在性能和能效上分别取得了 6.43% 和 4.35% 的提升。PIE 调度通过预测避免了抽样带来的性能开销,进一步优化了调度效果,相对于轮转调度在性能和能效上

分别取得了9.71%和7.21%的提升。本文的调度模型同样是通过性能预测来避免抽样开销,并且比PIE调度的预测准确度更高,从而做出次优调度决策的可能性更低,相对于轮转调度在性能和能效上分别取得13.64%和10.78%的提升,是所有调度模型中表现最好的。

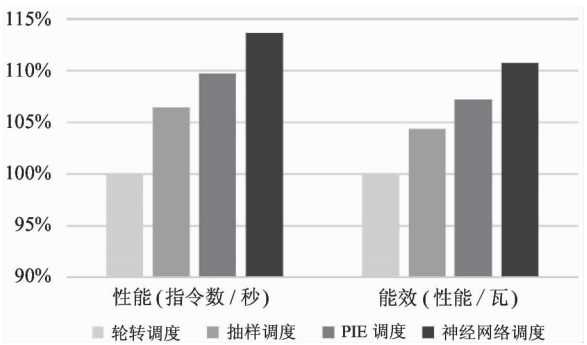

图6　调度模型的性能/能效提升

5　结　论

在异构多核处理器中,根据程序的硬件资源需求将不同程序分别调度到最适当的处理器核上运行对于整个系统的性能提升至关重要。本文提出一种基于神经网络的调度模型,该模型根据程序固有特征预测一个程序在不同处理器核运行的性能,根据预测结果以性能或能效为优化目标作出调度决策,从而提升系统资源利用率。该模型用到的神经网络算法简单,易于实现,具有较强的可扩展性和可移植性,并且表现出了良好的性能预测能力。根据预测结果,该调度模型对于系统的性能和能效取得了明显的提升效果。

参考文献

[1] NVidia. Variable SMP: a multi-core CPU architecture for low power and high performance. http://www.nvidia.com/content/PDF/tegra_white_papers/Variable-SMP-A-Multi-Core-CPU-Architecture-for-Low-Power-and-High-Performance-v1.1.pdf:NVIDIA, 2011

[2] Greenhalgh P. Big.LITTLE processing with ARM Cortex-A15 & Cortex-A7: Improving energy efficiency in high-performance mobile platforms. http://www.arm.com/files/downloads/big_LITTLE_Final_Final.pdf: ARM, 2011

[3] Rangan K, Powell M D, Wei G, et al. Achieving uniform performance and maximizing throughput in the presence of heterogeneity. In: Proceedings of the 17th International Symposium on High Performance Computer Architecture (HPCA), San Antonio, USA, 2011. 3-14

[4] Kumar R, Farkas K I, Jouppi N P, et al. Single-ISA heterogeneous multi-core architectures: The potential for processor power reduction. In: Proceedings of the 36th Annual IEEE/ACM International Symposium on Microarchitecture (Micro), San Diego, USA, 2003. 81-92

[5] Kumar R, Tullsen D M, Ranganathan P, et al. Single-ISA heterogeneous multi-core architectures for multithreaded workload performance. In: Proceedings of the 31th Annual International Symposium on Computer Architecture (ISCA), Munchen, Germany, 2004. 64-75

[6] Becchi M, Crowley P. Dynamic thread assignment on heterogeneous multiprocessor architectures. In: Proceedings of the Second Conference on Computing Frontiers (CF), Ischia, Italy, 2006. 29-40

[7] Chen J, John L K. Efficient program scheduling for heterogeneous multi-core processors. In: Proceedings of the 46th Design Automation Conference (DAC), San Francisco, USA, 2009. 927-930

[8] Ghiasi S, Keller T, Rawson F. Scheduling for heterogeneous processors in server systems. In: Proceedings of the Second Conference on Computing Frontiers (CF), Ischia, Italy, 2005. 199-210

[9] Koufaty D, Reddy D, Hahn S. Bias scheduling in heterogeneous multi-core architectures. In: Proceedings of the European Conference on Computer Systems (EuroSys), Paris, France, 2010. 125-138

[10] Li T, Brett P, Knauerhase R, et al. Operating system support for overlapping-ISA heterogeneous multi-core architectures. In: Proceedings of the 16th International Symposium on High Performance Computer Architecture (HPCA), Bangalore, India, 2010. 1-12

[11] Shelepov D, Alcaide J C S, Jeffery S, et al. HASS: A scheduler for heterogeneous multicore systems. *Operating-Systems Review*, 2009, 43(2):66-75

[12] Vraeynest K V, Jaleel A, Eeckhout L, et al. Scheduling heterogeneous multi-cores through performance impact estimation (PIE). In: Proceedingsof the 39th Annual Inter-

national Symposium on Computer Architecture (ISCA), Portland, USA, 2012. 213-224

[13] Chen J, John L K, Kaseridis D. Modeling program resource demand using inherent program characteristics. *Sigmetrics Performance Evaluation Review*, 2011, 39: 1-12

[14] Weidan W, Benjamin C L. Inferred models for dynamic and sparse hardware-software spaces. In: Proceedingsof the 45th Annual IEEE/ACM International Symposium on Microarchitecture (Micro), Vancouver, Canada, 2012. 413-424

[15] Sherwood T, Sair S, Calder B. Phase tracking and prediction. In: Proceedings of the 30th Annual International Symposium on Computer Architecture(ISCA), San Diego, USA, 2003. 336-347

[16] Kenneth H, Lieven E. Microarchitecture-independent workload characterization. *IEEE Micro Hot Tutirials*, 2007, 27(3):63-72

[17] Horikawa, Furuhashi, Uchikawa. On fuzzy modeling using fuzzy neural networks with the back-propagation algorithm. *IEEE Transactions on Neural Networks*, 1992, 3(5):801-806

[18] Binkert N, Beckmann B, Black G, et al. The gem5 simulator. *ACM SIGARCH Computer Architecture News*, 2011, 39(2):1-7

[19] Li S, Ahn J H, Strong R D, et al. McPAT: an integrated power, area, and timing modeling framework for multicore and many-core architectures. In: Proceedings of the 42th Annual IEEE/ACM International Symposium on Microarchitecture (Micro), New York, USA, 2009. 469-480

Scheduling for heterogeneous multi-core processors based on the prediction model using neural networks

Wang Lei[*,**], Lu Chao[*,**], Zhang Longbing[*,***], Wang Jian[*,***]

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

([**] University of Chinese Academy of Sciences, Beijing 100049)

([***] Loongson Technology Corporation Limited, Beijing 100190)

Abstract

The optimization of the program scheduling for heterogeneous multi-core processors was studied to improve the processors' performance and resource usage, and a new scheduling model based on neural networks' low cost prediction of program performance was proposed in view of the characteristics of heterogeneous multi-core processors. The scheduling model predicts the performance of each program on different cores according to the inherent program characteristics, and then makes the best program-core matching scheme based on the program predictions for program scheduling. The experimental results demonstrate that the proposed scheduling model outperforms the existing models of the round robin scheduling, the sampling-based scheduling and the performance impact estimation (PIE) scheduling in both performance and energy efficiency. For example, compared with the round robin scheduling, the performance and the energy efficiency of the proposed model increased by 13.64% and 10.78%, respectively.

Key words: heterogeneous multicore processors, multi-programmed workloads, inherent program characteristics, neural networks prediction model, the NN-based scheduling model