doi:10.11887/j.cn.202205012 http://journal.nudt.edu.cn

利用深度学习的硬件计数器复用估计算法*

王一超,王鎏振,林新华
(上海交通大学 网络信息中心,上海 200240)

摘 要:利用深度学习方法,为硬件计数器复用(multiplexing,MPX)提供结果精度更高的估计模型。通过对MPX估计得到的结果与实际采集的真实数据进行相似性分析,证明相同程序多次运行之间得到的硬件计数值是线性相关的。采用神经网络多层感知器(multilayer perceptron,MLP)和双向门控神经网络(bidirectional gated recurrent unit, Bi-GRU)这2种深度学习模型,对MPX数据进行拟合。基于动态时间规整(dynamic time warping, DTW),提出一个全新的评估MPX数据精度的指标DTW-cost。实验结果表明,同时收集15个硬件事件数据时,MLP方法拟合得到的13个高性能计算应用平均准确率比现有使用最广的固定插值法高出10.53%,最多可提升19.8%;而在MLP表现较差的事件上,Bi-GRU方法得到的平均准确率提升了28.8%。

关键词:硬件计数器;硬件性能事件;复用技术;深度学习
中图分类号:TP391 **文献标志码**:A **文章编号**:1001-2486(2022)05-114-10

Hardware counter multiplexing estimation algorithm using deep learning

WANG Yichao, *WANG Liuzhen*, *LIN Xinhua*
(Network & Information Center, Shanghai Jiao Tong University, Shanghai 200240, China)

Abstract: A state-of-art deep learning method was proposed to achieve higher accuracy of MPX(multiplexing) estimation. By analyzing the similarity between the MPX results and the real data, it was proved that hardware counts gained by running the same program was linear correlated. By applying the MLP(multilayer perceptron) and Bi-GRU(bidirectional gated recurrent unit) model, the MPX data was fitted. Based on DTW(dynamic time warping), a new metric DTW-cost was proposed to judge the accuracy of MPX result. Experiment results show that when sampling 15 hardware events simultaneously, average result of 13 high performance computing applications gained by the MLP model has a 10.53% higher relative accuracy than the fixed interpolation method. The MLP model has a 19.8% improvement at most. On the hardware events which MLP has a relatively poor performance, the Bi-GRU model improved relative accuracy score by 28.8% on average.

Keywords: hardware counters; hardware performance events; multiplexing; deep learning

硬件计数器是一组CPU专用寄存器。受限于芯片面积,主流X86和ARM CPU至多只能提供12个硬件计数器($C \leqslant 12$)。在代码执行期间,硬件计数器会记录性能事件发生的数量,如浮点数计算指令数、缓存命中次数等。而主流X86和ARM CPU都提供了上百个硬件事件($E > 500$)。由于C远小于E,单次采集时,C无法覆盖所有的E。

为提高单次采集硬件事件的数量,许多常用的性能分析工具如PAPI[1]、HPCToolkit[2]等提供了硬件计数器复用(multiplexing, MPX)功能。MPX包括数据采集和结果估计两大步骤:首先,MPX以一个固定频率读取硬件计数器实际采集到的硬件事件数据;其次,MPX基于估计算法,填补未被实际记录的数据。这样MPX就能在单次采集中使用少量硬件计数器C采集大量硬件事件E。基于MPX采集大量的硬件事件数据,研究人员可对应用在CPU上的性能表现进行深入的量化分析[3]。因此,性能分析工具中的MPX对于性能建模研究具有重要价值。

然而,目前PAPI MPX[1]仍采用数值拟合方法补齐未实际采样部分,例如固定插值[4]、线性插值[5]及非线性插值[6]。而这些数值拟合方法的精度较低,主要有两个原因:①基于数值拟合的估计算法无法准确预测所有事件的发生情况,因为一个硬件事件在不同CPU上运行不同程序时,

* **收稿日期**:2021-12-31
基金项目:国家自然科学基金资助项目(62072300)
作者简介:王一超(1990—),男,上海人,高级实验师,硕士,E-mail:wangyichao@sjtu.edu.cn;
林新华(通信作者),男,高级工程师,博士,博士生导师,E-mail:james@sjtu.edu.cn

并不服从任何特定随机过程;②采用数值拟合方法的 MPX 可以提升部分事件的采集精度,但有些事件精度过低,甚至出现负增益,导致 MPX 结果置信度偏低。事件数量的估计准确度对 MPX 模式下取得的结果,起到至关重要的作用。根据之前工作的评估[3,6],当 MPX 估计算法的准确性下降到 90% 以下时,数据将无法被用于性能建模。

为提升 MPX 结果精度,利用深度学习方法,基于神经网络多层感知器[7](multilayer perceptron, MLP)和双向门控神经网络[8](bidirectional gated recurrent unit, Bi-GRU)2 个模型,为 MPX 提供置信度更高的估计模型。需要说明的是,本文研究的 MPX 方法面向高性能计算中常用的性能事件,在程序单核运行阶段,对于多事件进行采样,从而得到可以用于性能建模的大量性能事件。

为了使模型提取到的特征可以覆盖高性能计算的主流应用程序,选取了在高性能计算领域主流的 Rodinia 测试集[9]。该测试集中的应用覆盖了 13 类典型的高性能计算应用,基于该测试集的数据进行训练,可以有效提升模型在高性能计算领域的泛化能力。

1　研究背景及相关工作

1.1　MPX 基本原理

MPX 采用分时复用方法,将各个硬件计时器的可用时间切分成不同时间片,然后在不同时间片上轮流记录不同硬件事件。具体包括两个步骤:

1)数据采集。MPX 以一个固定频率,读取硬件计数器实际采集到的性能事件数据,写入内存中。

2)结果估计。由于硬件计数器在多事件调度过程中,存在部分事件发生次数未被计数器记录的情况。因此,MPX 需要基于估计算法估出该时间段内事件发生的次数,补全未采集到的数据。与 MPX 相对的是使用单一寄存器记录单一事件(one counter one event, OCOE),这种采集数据的精度高,但单次采集硬件事件受限于硬件计算器数量($C \leqslant 12$)。

1.2　MPX 估计算法使用的数值拟合方法

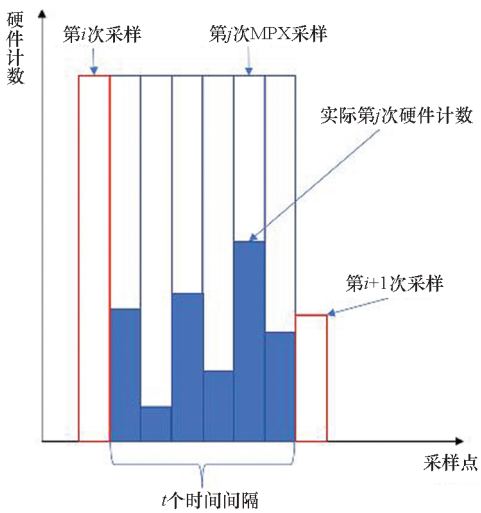
已有 MPX 估计算法都采用了数值拟合方法。根据采用的方法不同,可分为三类:

1)固定插值法。2001 年,May[4]在 PAPI[1]上开发了面向性能监控单元(performance monitor unit, PMU)事件的多路复用技术,即 PAPI 默认 MPX 实现。该实现基于固定时间片调度和固定插值的估计方法,如图 1(a)所示,本文将该方法作为基准比较对象。

2)线性插值法。2005 年,Mathur 等[5]量化了由 MPX 引起的误差,设计了 4 种插值估计算法,包含梯形面积法等,如图 1(b)所示,采用线性插值法替换固定差值法,提升了估计精度。实验数据表明[10],线性插值法会造成 42% 的估计值比实际值大 10% 以上,其中误差在 50% 的估计值最高占比达 29.3%。

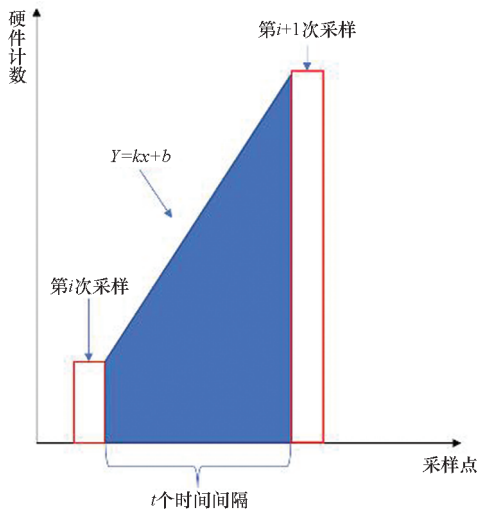
3)非线性插值法。本文作者对文献[4]的工作进行了改进[6,11],提出采用非线性插值替换线性插值的估计方法,如图 1(c)所示。

上述三种方法都受限于数值拟合的局限,结果精度仍有进一步提升的空间。有不少基于硬件



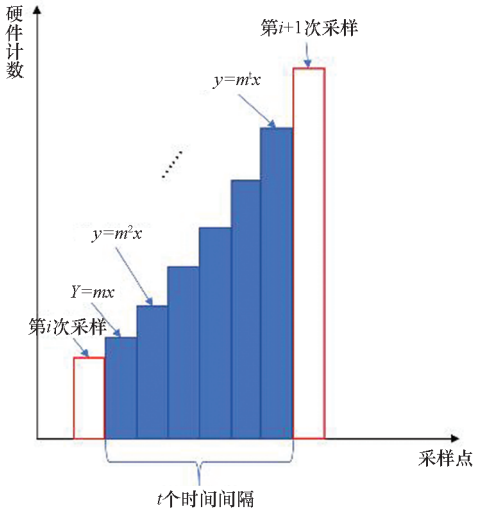
(a) 固定插值法

(a) Fixed interpolation



(b) 线性插值法

(b) Linear interpolation



(c) 非线性插值法

(c) Non-linear interpolation

图 1 现有 3 种 MPX 数值拟合方法

Fig. 1 Three MPX numerical fitting methods

事件的性能建模研究工作采用了深度学习方法作为拟合手段[12-14]。受此启发,为进一步提升 MPX 估计精度,本文采用深度学习方法进行 MPX 估计。

2 数据清洗与相关性分析

在对数据进行相关性分析之前,需要先对数据进行清洗,以修正一些明显异常的值,从而提高数据分析的准确性。数据出现明显异常的原因有两种:一种是性能分析工具如 PAPI 在采集时出现异常;另一种是由于 MPX 估计算法不能及时判断程序结束而导致的。以分支预测未命中的硬件事件 BRMIS:ALL 为例进行说明。

2.1 变量定义

首先对下文使用到的变量做如下定义:

1) $Sampled_i$ 表示 i 时刻 MPX 采集到的硬件计数;

2) $Count_i$ 表示 i 时刻估计算法给出的硬件计数;

3) $Interval$ 表示两次采样之间的时间差。

其中,$Interval$ 由 MPX 同时采样的硬件事件数量 $Event\_num$ 以及 OCOE 模式下最大支持的可编程硬件计数器数量 $Counter\_num$ 决定,即:

$$Interval = \frac{Event\_num}{Counter\_num} \tag{1}$$

PAPI 默认的 MPX 采用固定插值法,即假设同一时间片内硬件事件发生的次数保持不变,将采样间隔内某一时刻采集到的硬件计数值进行固定插值,即:

$$Counted_i = Sampled_i \times Interval \tag{2}$$

2.2 数据清洗:异常值处理策略

PAPI 在采集时出现异常值,主要有以下两个原因:

1)性能分析工具没有采集到任何硬件事件或是硬件计数出现负值。这部分异常是由于 PMU 异常溢出导致的,使用 MPX 和 OCOE 均有一定概率出现该现象。实验表明,所有未出现异常计数的采样得到的硬件计数和均小于 $0.3 \times Count_{max}$。因此针对这部分异常,本文根据采集到的硬件计数和设置了一个阈值:记硬件计数和的最大值为 $Count_{max}$,将所有硬件计数和小于 $0.2 \times Count_{max}$ 的采集舍弃,确保使用的数据都是有效采集的。其中,选择 0.2 为系数是因为采样无法保证能检测到所有情况,因此适当降低阈值避免错误地将正常采样归为异常采样。

2)采集到的硬件计数图像中存在异常值。图 2 显示的是在 Intel Xeon Gold 6248 上对 Rodinia 测试集中的 StreamCluster[9] 采集到 BRMIS:ALL 的硬件计数 - 时间步图像,异常值部分是硬件计数 - 时间步图像中出现的异常偏低点,使用蓝色矩形块框出。这一现象在文献[14]中曾提到过,可采用 K 最近邻(K-nearest neighbor, KNN)法神经网络修复。由于异常值产生的原因有很多,无法准确判断是由性能分析工具本身的不稳定还是在程序运行中出现问题而导致的。因此针对这部分异常,本文采取保留异常值的策略。

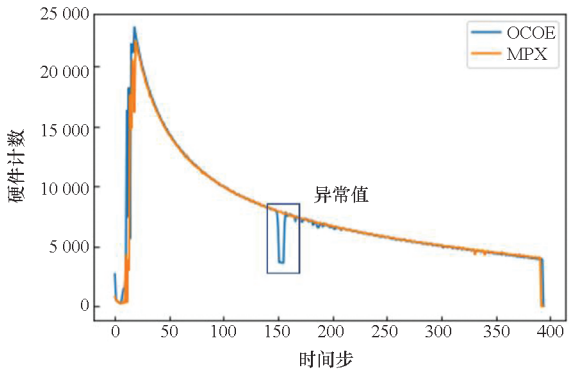


图 2 在 Intel Xeon Gold 6248 上对 StreamCluster 采集 BRMIS:ALL 产生的异常值

Fig. 2 Outlier of BRMIS:ALL event counts when sampling of StreamCluster benchmark on Intel Xeon Gold 6248

2.3 数据清洗:尾部截断

如图 3 所示,对比 OCOE 和 MPX 固定插值这 2 种模式下的事件计数结果,可以看出两者结果已比较接近。然而通过放大数据图的尾部,可以

发现当 OCOE 下硬件计数已经归 0 时,MPX 下仍在基于估计算法填充缺失的数值,导致这部分计数结果与实际情况发生偏差。

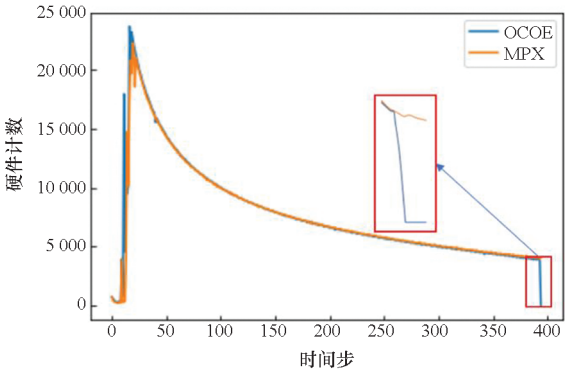


图 3　采用 OCOE 和 MPX 固定插值 2 种模式得到的 BRMIS:ALL 事件计数结果对比

Fig. 3　BRMIS:ALL count/time step curve under two modes

以上现象是由于处理器性能波动引起的,同一程序即使在相同运行参数下,在同一个处理器上每次运行的时间会有差异,这导致了在程序运行的末尾,每次运行的结果差异较大。因此,为提高 MPX 结果精度,应把程序临近结束的部分予以截断。本文选取的截断方式是舍去采样步长的末尾 2%。考虑到部分硬件事件采集到的时间片数量较少,因此在舍弃末尾 2% 的基础上进一步舍弃末尾 5 个时间步的数据。尾部截断后,重新计算得到的 BRMIS:ALL 在 OCOE 与 MPX 模式下,计数结果的曲线重合度显著提升。

2.4　数据相关性分析

对比图 2 和图 3,可以观察到以相同参数运行同一程序,不同采集方式得到的硬件计数 - 时间步图像有着较高的相似度。为验证这一发现,本文采用皮尔森相关系数[15]衡量两者之间的相似程度。皮尔森相关系数是一种用于衡量两份数据线性相关程度的指标,其范围在 -1 到 1 之间,相关系数绝对值越接近 1,相关度越强。负的相关系数则表示两者负相关。具体计算公式为:

$$r = \frac{N\sum x_i y_i - \sum x_i \sum y_i}{\sqrt{N\sum x_i^2 - \left(\sum x_i\right)^2}\sqrt{N\sum y_i^2 - \left(\sum y_i\right)^2}} \tag{3}$$

通过计算,两次采集到的 BRMIS:ALL 硬件计数之间的皮尔森相关系数为 0.93。进一步计算其他许多硬件计算器的数据,结果也类似。这证明在相同参数下,多次运行同一程序而得到的硬件计数值具有极高的线性相关性。这个发现是本文采用深度学习方法拟合 MPX 估计算法的理论依据。

与本文的这个发现不同,以往基于数值拟合的 MPX 估计算法都是基于时间局部性的假设,即认为相邻时间片内由于时间相关性,必然存在一定关联,并据此设计 MPX 估计算法。如固定插值法[4]认为相邻时间片硬件事件数量相等,线性拟合法[5]假设相邻时间片间硬件计数呈连续或离散的线性关系,而非线性拟合法[5]假设相邻时间片间硬件计数成连续或离散的指数关系。

3　深度学习模型的选取与训练

基于上述发现,考虑到硬件事件发生具有时序性的特征以及拟合的泛用性,选取深度学习中的 MLP 和 Bi-GRU 模型,以 MPX 采集到的数据为自变量,OCOE 采集到的数据为因变量,构建从 MPX 数据到 OCOE 数据的函数映射。具体而言,MPX 的输入是一个长度为 t 的序列:[*Sampled*_1, *Sampled*_2,…,*Sampled*_*t*],在经过深度学习模型进行一系列矩阵或向量相关运算后,输出一个长度同样为 t 的序列:[*Predicted*_1, *Predicted*_2,…, *Predicted*_*t*]。

3.1　MLP 模型的参数设置与模型结构

MLP[12]是经典的深度学习模型,与最小二乘法寻找拟合函数的原理类似。MLP 可以通过最小化均方误差训练出一个能将 MPX 数据映射到 OCOE 数据的模型。考虑到硬件事件时序数据没有高维特征,不需要过多的隐含层数。模型超参数选取如表 1 所示,模型结构如图 4 所示(图中圆圈表示神经元)。由于数据集较小,MLP 模型中每个隐含层的神经元都有 30% 的概率失效,添加了 *Dropout* = 0.3 避免过拟合[13]。

表 1　MLP 模型超参数

Tab. 1　Hyperparameters of MLP model

超参数	数值
隐含层 1 节点数	128
隐含层 2 节点数	64
输出层节点数	1
学习率	3×10^{-3}
Epoch	1 200

3.2　Bi-GRU 模型的参数设置与模型结构

Bi-GRU 是双向长短期记忆神经网络(bidirectional long-short term memory, Bi-LSTM)[8]的简化。Bi-GRU 是循环神经网络模型中的一种,

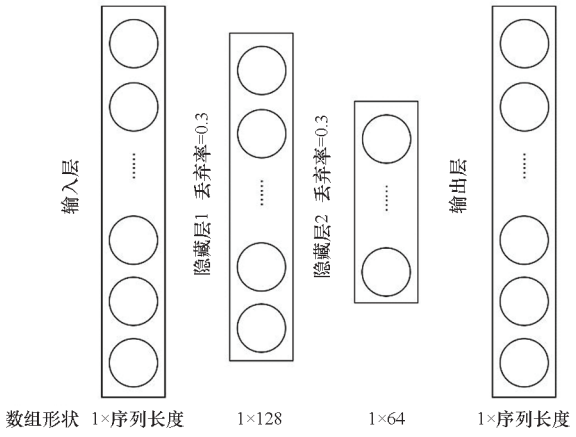


图4 MLP模型示意

Fig. 4 Illustration for MLP model

针对时序数据问题训练效果出色。其中LSTM[15]引入了遗忘门,解决了循环神经网络(recurrent neural network, RNN)在长时依赖下的梯度爆炸和梯度消失问题,而GRU[16]在保持LSTM效果的前提下对网络结构进行了精简,将遗忘门和输出门合并为更新门,有效缩短了训练神经网络所需时间。而双向意味着将时间序列按正反双向输入网络,使得输出某一时刻结果时可同时参考上下文信息。

考虑到程序运行时记录的硬件事件就是一个标准的时间序列,而且程序运行时的上下文切换也会对硬件计数造成影响,采用Bi-GRU来处理MPX估计。模型参数选取如表2所示,结构如图5所示。

表2 Bi-GRU模型参数

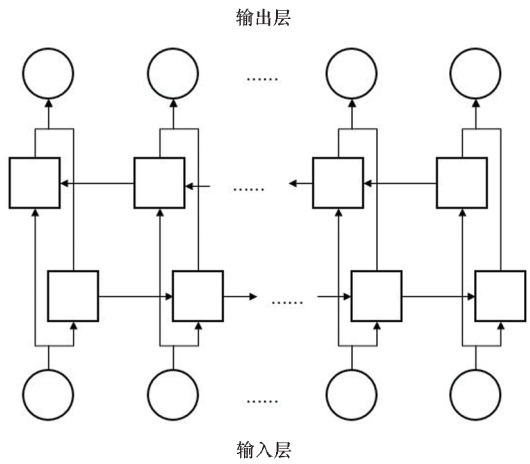
Tab. 2 Hyperparameters of Bi-GRU model

超参数	数值
GRU隐含层	64
隐含层层数	3
全连接层	1
学习率	5×10^{-3}
Epoch	600

3.3 生成训练所需的数据集

实验在Intel Xeon Gold 6248上使用基于PAPI v5.7.0开发的NeoMPX[17]进行硬件事件采样,选取Rodinia测试集[9]的13个应用,每个应用分别通过MPX和OCOE采集到15个硬件事件。

生成训练数据集需要5个步骤:①MPX运行1次应用即可采集到15个硬件事件,而OCOE需



注:正方形表示GRU单元,圆圈表示序列中的数据点。

图5 Bi-GRU模型示意

Fig. 5 Illustration for Bi-GRU

重复运行应用3次,所有应用的运行参数均按推荐值设置;②对每个应用,实验将重复运行$4x$次,其中x次以MPX方法进行采样,而另外$3x$次以OCOE方法进行采样($x>100$);③对采集到的结果均进行尾部截断(见2.3节),并通过设定计数总和阈值进行筛选,最终得到总量为100份的MPX和OCOE采样结果;④每一份采样结果包含一个时序序列,其中的采样数据量大约在200~700个硬件事件值;⑤实验将这100份结果打乱后,以7∶1∶2的比例按序号随机切分成训练集、验证集及测试集,集合两两之间没有重复元素。

3.4 模型训练步骤

使用Pytorch v1.6.0实现MLP及Bi-GRU模型,输入为70份MPX和OCOE采样结果,输出为经MPX估计后的结果。模型训练的关键步骤如下:

1)数据大小预处理。由于MPX与OCOE采样数据量庞大,通常在10^7量级,直接拟合效果不佳,因此需要对数据进行对数化处理,取对数底为10。

2)数据长度预处理。由于溢出时间的不确定性导致结束位置不同,所以数据尾部的2%会被截断;同时考虑有的硬件事件发生的时间片较短,因此在此基础上舍弃最后5个时间步的数据。

3)模型训练。以MPX采集到的数据为自变量,OCOE采集到的数据为因变量,将它们组合成Tensor data的格式,通过Data Loader,以*batch_size* = 1的大小将数据输入2个模型中。计算完均方误差后反向传播至神经网络各神经元中。将所有训练集中数据输入进神经网络后完成一个epoch。重复上述过程完成模型训练。

4)结果估计。将测试集中的数据通过 Data Loader,以 *batch_size* = 1 的大小将数据输入训练后的模型中。在经由一系列的矩阵/向量运算后,模型将输出估计后的结果。

5)结果验证。对测试集中的 20 份数据分别依照评价标准计算模型得分,并根据这 20 份数据的平均值进行评价。

4　实验配置与评价指标

4.1　实验配置

1)实验环境。实验硬件为 Intel Xeon Gold 6248 CPU,软件是基于 PAPI v5.7.0 开发的 NeoMPX[17]。

2)测试算例。为全面考察 MPX 估计算法在高性能计算应用中的使用效果,选取 13 个来自 Rodinia 的测试程序,如表 3 所示。这些测试程序涵盖了高性能计算中 5 种不同计算模式及 9 个不同应用领域。

表 3　13 个 Rodinia 测试程序

Tab.3　Thirteen evaluated benchmarks from Rodinia

测试程序	算法类型	应用领域
Leukocyte	结构化网格	医用图像
Hotspot	结构化网格	物理模拟
SRAD	结构化网格	图像处理
Hotspot3D	结构化网格	物理模拟
Breadth-First Search(BFS)	图遍历	图算法搜索
B + Tree	图遍历	图算法搜索
Kmeans	密集线性代数	数据挖掘
KNN	密集线性代数	数据挖掘
LU-Decomposition(LUD)	密集线性代数	线性代数
Streamcluster	密集线性代数	数据挖掘
Pathfinder	动态规划	网格搜索
Needleman-Wunsch(NW)	动态规划	生物信息学
LavaMD	多体问题	分子动力学

3)硬件事件。主流 CPU 能够支持的硬件事件多达上百个,但性能检测与分析中常用的事件并不会全部用到。因此,本文选取了 15 个性能分析与建模中比较常用的硬件事件,如表 4 所示。这些事件涵盖了缓存访问、TLB 访问、内存访问以及分支指令。要采集这 15 个事件,MPX 只需运行 1 次,而 OCOE 需运行多次。

表 4　15 个用于评估的硬件事件

Tab.4　Fifteen evaluated hardware events

事件缩写	硬件事件名
BRINS:ALL	BR INST RETIRED:ALL BRANCHES
BRINS:COND	BR INST RETIRED:CONDITIONAL
BRMIS:ALL	BR MISP RETIRED:ALL BRANCHES
BRMIS:COND	BR MISP RETIRED:CONDITIONAL
DTLM:M	DTLB LOAD MISSES:MISS CAUSES A WALK
DTLM:S	DTLB LOAD MISSES:STLB HIT
L1h	MEM LOAD UOPS RETIRED:L1 HIT
L1m	MEM LOAD UOPS RETIRED:L1 MISS
L2h	MEM LOAD UOPS RETIRED:L2 HIT
L2m	MEM LOAD UOPS RETIRED:L2 MISS
Ich	ICACHE 64B:IFTAG HIT
Icm	ICAHCE 64B:IFTAG MISS
UISTALL	UOPS ISSUED:STALL CYCLES
URSTALL	UOPS RETIRED:STALL CYCLES
INST	INSTRUCTION RETIRED

4.2　评价 MPX 结果精度的指标

为准确评价 MPX 结果精度,本文提出 2 个指标。

1)相对精度(relative accuracy,RA)。该指标参考了统计学中相对误差的概念,计算公式如下:

$$RA = 1 - \frac{1}{n}\sum_{i=1}^{n}\frac{|MPX_i - OCOE_i|}{OCOE_i} \tag{4}$$

其中,MPX_i 和 $OCOE_i$ 分别代表第 i 个时间步时,MPX 的结果以及使用 OCOE 模式采集到的硬件计数。式中减数即为统计学中的相对误差。为更直观反映 MPX 估计算法的好坏,用 1 减去相对误差得到 RA 值。RA 值越高,表明由 MPX 估计算法得到的结果越接近于 OCOE 采集到的结果,即精度越高。

2)时间序列间的相似度[18](dynamic time warping,DTW)。假设 MPX 数据为 $MPX_1, MPX_2, \cdots, MPX_m$;$OCOE$ 数据为 $OCOE_1, OCOE_2, \cdots, OCOE_n$。DTW 算法首先计算所有 MPX_i 到 $OCOE_j$ 的欧氏距离,从而得到一个 $m \times n$ 的二维矩阵。将该二维矩阵视为一个网格,网格上所有的值代表经过该点的开销。最后需要求得一条通过该矩阵网格的最优路径。该路径有如下限制:

1)从(1, 1)开始,到(m, n)结束;

2)若前一个点选择了(i, j),下一个点只能在($i+1$, j),(i, $j+1$),($i+1$, $j+1$)中选择。

该路径可以通过动态规划算法得出,DTW-cost 为该路径上的开销总和,它表示 MPX 序列中元素到 OCOE 序列中最近的邻近元素累计距离之和,其值域为[0, $+\infty$),该值越小,说明两序列越相似。

DTW-cost 有效避免了欧氏距离衡量两个时间序列中,由于序列间的相位延迟、长度不匹配等

造成的评估结果与实际表现不匹配的问题。

5 实验结果

为确定测试重点，本文测试了 PAPI 默认的固定插值 MPX 在 15 个硬件事件上的效果，即硬件事件在 13 个应用上的平均 RA 分数，结果如图 6 所示。固定插值 MPX 在 8 个硬件事件上的平均 RA 分数超过了 0.85，因此结果分析重点关注余下 6 个 RA 分数低于 0.8 的硬件事件。

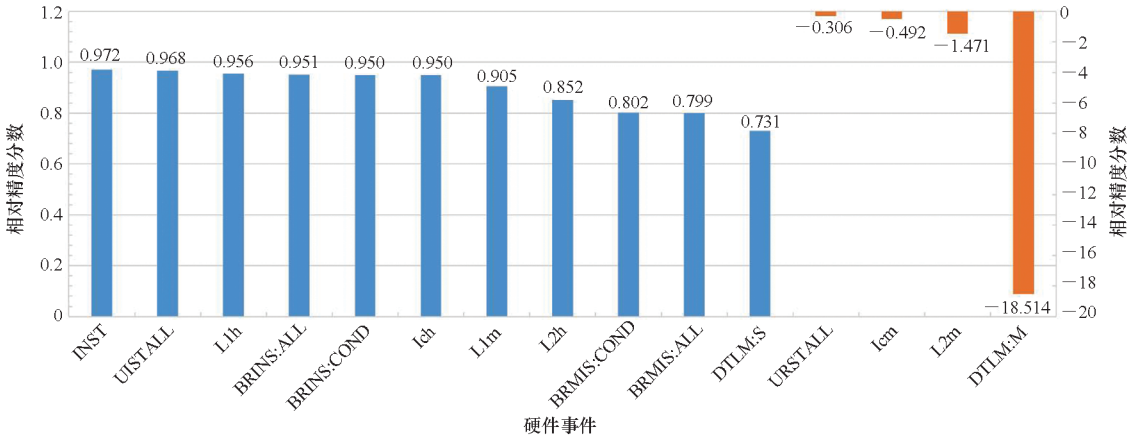


图 6 默认估计算法得到的不同硬件事件平均 RA 分数(越高越好)

Fig. 6 Average RA score on different hardware event gained by PAPI default method (higher is better)

表 5 从应用维度对比了固定插值 MPX 与基于 MLP 模型的 MPX 结果。后者在所有 Rodinia 测试集中 12 个应用的平均 RA 分数都得到了提升。以 SRAD 为例，MLP 方法相比固定插值 MPX 提升了 27.4%，从原先的 0.73 提升到 0.9 以上。

表 5 不同应用的平均 RA 分数对比(越高越好)

Tab. 5 Average RA score comparison of different applications (higher is better)

应用	固定插值	MLP	MLP 提升量
Leukocyte	0.67	0.79	0.12
Hotspot	0.66	0.78	0.12
SRAD	0.73	0.93	0.20
Hotspot3D	0.87	0.97	0.12
BFS	0.78	0.82	0.04
B + Tree	0.75	0.91	0.16
Kmeans	0.75	0.88	0.13
KNN	0.79	0.91	0.12
Streamcluster	0.78	0.72	−0.06
NW	0.81	0.88	0.07
LavaMD	0.67	0.84	0.17
LUD	0.81	0.86	0.05
均值	0.76	0.86	0.10

需要指出的是，考虑到出现负值说明预估的硬件计数要比实际硬件计数高出一倍以上，因此负值是无意义的。为不产生误解，负值将被处理为 0 后再做比较。如表 5 所示，重新计算了两种方法在各应用上的平均 RA 分数，经修正后的 MLP 得到的 RA 分数比默认方法高出 0.10，相比默认的方法提升了 13.16%。其中最大提升幅度出现在 SRAD 应用上，达到了 0.20。

表 6 从硬件事件的维度，对比了固定插值 MPX 与基于 MLP 模型的 MPX 结果。实验结果表

表 6 硬件事件平均 RA 分数对比(越高越好)

Tab. 6 Average RA score of hardware events (higher is better)

硬件事件	固定插值	MLP	MLP 提升量
BRINS:ALL	0.95	0.96	0.01
BRINS:COND	0.95	0.96	0.01
BRMIS:ALL	0.80	0.79	−0.01
BRMIS:COND	0.80	0.81	0.01
DTLM:M	0.24	0.41	0.16
DTLM:S	0.73	0.72	−0.01
L1h	0.95	0.97	0.01
L1m	0.90	0.95	0.04
L2h	0.85	0.91	0.06
L2m	0.75	0.75	0.00
Ich	0.95	0.95	0.00
Icm	0.16	0.82	0.66
UISTALL	0.97	0.95	−0.02
URSTALL	0.34	0.95	0.61
INST	0.97	0.97	0.00
均值	0.76	0.86	0.10

明,在固定插值 MPX 达到较高 RA 的硬件事件上,MLP 能取得相近表现,至多相差 0.01。而 MLP 相比固定插值的最大提升是在 Icm、URSTALL 这两个硬件事件上,分别提升了 0.66 和 0.61,将原本无法用于实际场景的硬件事件的 RA 分数提升到 0.82 甚至 0.95 的可信度水平。

如图 7 所示,本文使用 DTW 指标评判固定插值 MPX 与基于 MLP 模型的 MPX 结果。参考表 6 与图 7 结果,从总体上来说,RA 与 DTW 两种评价指标基本一致,即 RA 越高,DTW-cost 越小,MPX 结果精度越高。

对于采用固定插值 MPX 就能达到 RA 0.9 以上的硬件事件,MLP 能够达到相近精度,但已没有进一步的提升空间。因此,对 Bi-GRU 效果的评估只考虑原本效果较差的那些硬件事件。RA 分数对比结果如图 8 所示。

针对选取的 7 个硬件事件,Bi-GRU 取得了最好的效果,其次是 MLP,两者皆优于固定插值 MPX。对于这 7 个硬件事件的平均 RA 分数,Bi-GRU 相比 MLP 提升了 0.09;相比固定插值 MPX 提升了 0.29,其中最大提升 0.75,最小提升 0.05。这表明 Bi-GRU 对于具有时间序列特征的硬件事件具有较好的效果。

如图 9 所示,本文使用 DTW-cost 指标对 Bi-GRU 算法进行评估。除 DTLM:S 的 DTW-cost 与 RA 表现相反以外,其余硬件事件均符合 DTW-cost 越低,RA 越高的规律。其中,Bi-GRU 模型取得了最低的 DTW-cost,7 个硬件事件的 DTW-cost 均值为 20.86,相比 MLP 模型的 24.42 降低了 14.58%。而 MLP 和 Bi-GRU 这 2 种基于深度学习方法的 DTW-cost 相比于数值拟合方法(50.60)分别降低了 51.74% 和 58.77%。

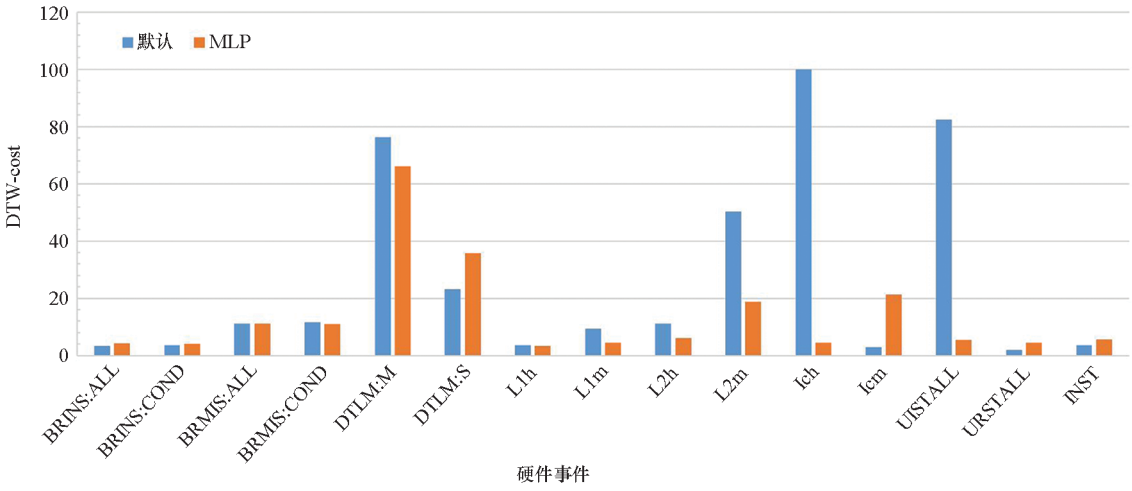


图 7　默认方法与 MLP 方法的 DTW 开销对比(越低越好)

Fig. 7　DTW cost comparison between default method and MLP method (lower is better)

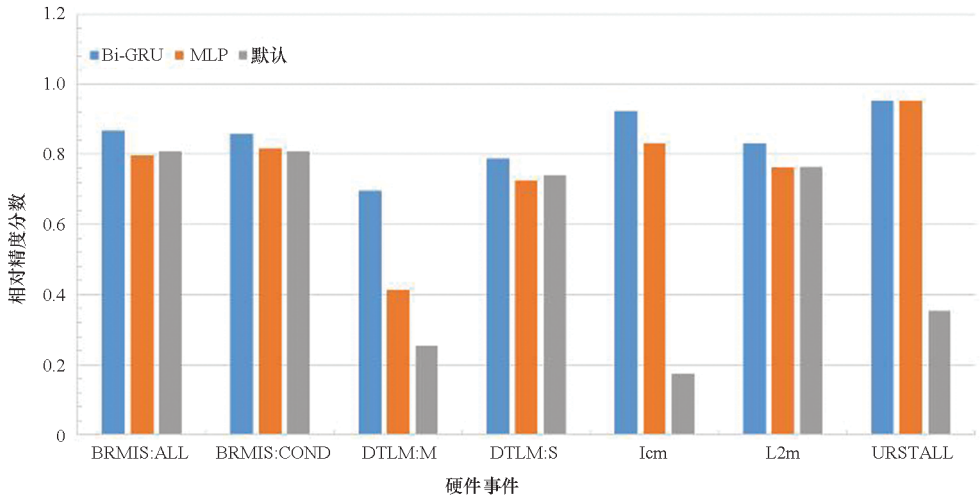


图 8　三种估计算法 RA 分数对比(越高越好)

Fig. 8　RA score comparison among three estimation methods(higher is better)

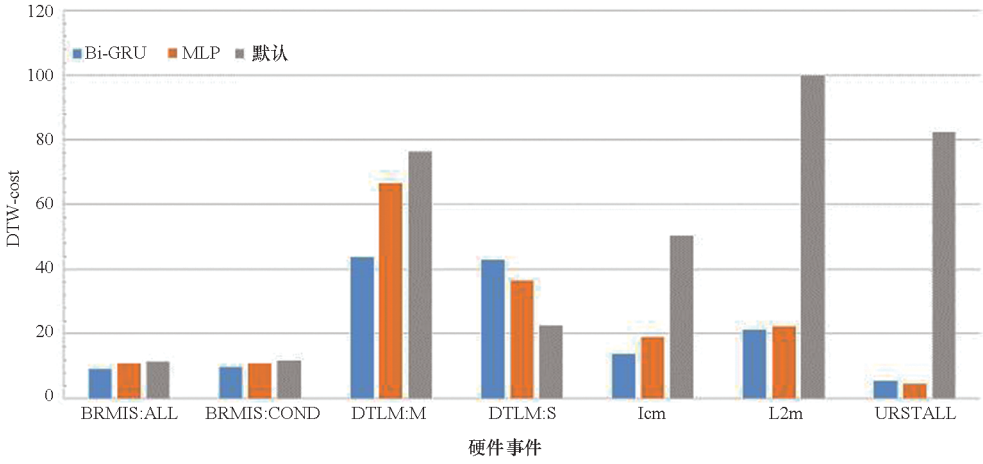


图9 3种方法的DTW开销对比(越低越好)

Fig. 9 DTW-cost among three methods (lower is better)

6 结论

为进一步提升精度,本文基于深度学习模型MLP和Bi-GRU,提出了2种新的MPX估计算法。本文的MPX实现旨在面向高性能计算应用,在处理器性能建模中,提升关键硬件事件的多事件采集精度,进而为性能建模提供更可靠的大数据基础。本文主要有以下3个贡献:

1)通过MPX结果与实际数据的相似性分析,证明多次运行之间得到的硬件计数值存在线性相关性;

2)基于MLP和Bi-GRU深度学习模型,对MPX结果估计进行数据拟合,提升了MPX结果精度;

3)基于时序分析提出评估MPX结果准确度的指标。

实验结果表明,基于MLP和Bi-GRU模型的MPX估计算法都能有效提升MPX结果精度。在Intel CPU上运行的13个基准测试程序中,在同时记录15个硬件事件时,MLP模型相比固定插值MPX,RA分数平均提升了0.10左右,最多能提高0.66。而在MLP模型提升有限的7个硬件事件中,Bi-GRU模型相比固定插值MPX提升了0.288,其中最大提升达0.745。使用DTW-cost进行评估时,两种基于深度学习的模型均比固定插值法降低了超过一半的DTW-cost。对比本文之前的工作,在更广泛的高性能计算应用中(由6个拓展到13个)取得了更好的表现。此外,实验结果表明,对于固定插值法下估计精度较差的硬件事件,采用MLP和Bi-GRU方法后,取得了更高的精度。部分硬件事件的精度在MLP和Bi-GRU方法下,精度仍未达到0.9,说明目前模型在面向不同硬件事件时,其模型的泛化能力仍有待评估与研究。

未来的工作将对MPX模式下的异常值产生机制和性能波动原因深入分析,从而降低由于意外波动导致的误差。后续还将增加MPX同时采集的硬件事件数量,并将更多常用硬件事件纳入硬件事件组,从而验证基于深度学习的模型拟合方法在硬件事件更多、时间片更短的情况下,是否能够达到提升RA分数的效果,并最终在多核处理器上实现多线程并行抓取。

参考文献(References)

[1] BROWNE S, DONGARRA J, GARNER N, et al. A portable programming interface for performance evaluation on modern processors[J]. The International Journal of High Performance Computing Applications, 2000, 14(3): 189-204.

[2] ADHIANTO L, BANERJEE S, FAGAN M, et al. HPCTOOLKIT: tools for performance analysis of optimized parallel programs [J]. Concurrency and Computation: Practice and Experience, 2009, 22(6): 685-701.

[3] LYU Y R, SUN B, LUO Q Y, et al. CounterMiner: mining big performance data from hardware counters [C]// Proceedings of 51st Annual IEEE/ACM International Symposium on Microarchitecture (MICRO), 2018.

[4] MAY J M. MPX: software for multiplexing hardware performance counters in multithreaded programs [C]// Proceedings of 15th International Parallel and Distributed Processing Symposium, 2001.

[5] MATHUR W, COOK J. Improved estimation for software multiplexing of performance counters [C]//Proceedings of 13th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, 2005.

[6] 林新华, 王杰, 王一超, 等. 基于数据分布一致性的处理器硬件性能计数器复用估计方法[J]. 计算机研究与发展, 2022, 59(6): 1192-1201.
LIN X H, WANG J, WANG Y C, et al. A data distribution-consistency-based estimation method for multiplexing processor

hardware performance counters [J]. Journal of Computer Research and Development, 2022, 59(6): 1192 - 1201. (in Chinese)

[7] HORNIK K, STINCHCOMBE M, WHITE H. Multilayer feedforward networks are universal approximators[J]. Neural Networks, 1989, 2(5): 359 - 366.

[8] SCHUSTER M, PALIWALK K. Bidirectional recurrent neural networks[J]. IEEE Transactions on Signal Processing, 1997, 45(11): 2673 - 2681.

[9] CHE S, BOYER M, MENG J Y, et al. Rodinia: a benchmark suite for heterogeneous computing [C]// Proceedings of IEEE International Symposium on Workload Characterization, 2009.

[10] WEAVER V M, MCKEE S A. Can hardware performance counters be trusted? [C]//Proceedings of IEEE International Symposium on Workload Characterization, 2008.

[11] WANG Y C, WANG J, CHEN J K, et al. NeoMPX: characterizing and improving estimation of multiplexing hardware counters for PAPI [C]//Proceedings of IEEE International Conference on Cluster Computing, 2020.

[12] TUNCER O, ATES E, ZHANG Y J, et al. Online diagnosis of performance variation in HPC systems using machine learning[J]. IEEE Transactions on Parallel and Distributed Systems, 2019, 30(4): 883 - 896.

[13] CUI X W, FENG W C. IterML: iterative machine learning for intelligent parameter pruning and tuning in graphics processing units[J]. Journal of Signal Processing Systems, 2021, 93: 391 - 403.

[14] PEARSON K F R S. Notes on the history of correlation[J]. Biometrika, 1920, 13(1): 25 - 45.

[15] HOCHREITER S, SCHMIDHUBER J. Long short-term memory[J]. Neural Computation, 1997, 9(8): 1735 - 1780.

[16] CHO K, VAN MERRIENBOER B, GULCEHRE C, et al. Learning phrase representations using RNN encoder-decoder for statistical machine translation[EB/OL]. (2014 - 09 - 03)[2021 - 12 - 30]. https://arxiv.org/abs/1406.1078.

[17] WANG Y C, WANG J, CHEN J K. NeoMPX: characterizing and improving estimation of multiplexing hardware counters for PAPI [C]//Proceedings of 2020 IEEE International Conference on Cluster Computing, 2020.

[18] KEOGH E, RATANAMAHATANA C A. Exact indexing of dynamic time warping [J]. Knowledge and Information Systems, 2005, 7: 358 - 386.