基于聚类抽样的访存特征分析方法*

查中彬[1,2+]，崔泽汉[1,2]，黄永兵[1,2]，陈荔城[1,2]，包云岗[1]，陈明宇[1]

[1]（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，北京市 100190）

[2]（中国科学院大学，北京市 100190）

Memory Access Characteristics Analysis based on Clustering Sampling

ZHA Zhong-Bin[1+]，CUI Ze-Han[1]，HUANG Yong-Bing[1]，CHEN Li-Cheng[1]，BAO Yun-Gang[1]，CHEN Ming-Yu[1]

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

[2](University of Chinese Academy of Sciences, Beijing 100190, China)

+ Corresponding author: Phn: +86-15652917926, E-mail: zhazhongbin@ict.ac.cn

Abstract: The advent of the Big Data era makes it more obvious that the memory system is the bottleneck of the computer system performance. How to use the memory trace to analyze the memory access characteristics has become the hot topics in the study of memory system architecture. Because the size of complete memory trace is too big, it is urgent to find out a method which is fast and accurate to analysis the memory access characteristics. In this paper, we propose a method called EMAT (Extracting representative Memory Access Trace) which is based on clustering sampling to analyze the memory access characteristics. In EMAT, we use BPCV (Basic Performance Count Vector) to reflect the corresponding sequence fragment's memory access behavior. Then find out the memory access characteristics according to clustering result of BPCS. At the end of this paper, we use EMAT to analyze the benchmarks in the SPEC CPU2006 and PARSEC3.0 (include splash2x), the experiment results show that the EMAT can find out the memory access characteristics quickly and efficiently.

Key words: Memory access trace；Clustering Sampling；BPCV；Benchmark；EMAT

摘要： 大数据时代的来临，使得内存系统越来越成为制约系统整体性能的瓶颈，如何利用访存序列分析程序的访存特征已经成为内存系统结构研究的热点。针对完整访存序列数据量过大的问题，需要一种快速准确的访存特征分析方法。本文提出一种基于聚类抽样的访存特征分析方法 EMAT（Extracting representative Memory Access Trace）。在 EMAT 中，使用由体系结构相关性能指标构成的基本性能计数向量 BPCV（Basic Performance Count Vector）反映相应访存序列片段的访存行为，根据基本性能计数向量的聚类结果找出程序的访存特征。最后使用 EMAT 方法分析了 SPEC CPU2006 和 PARSEC3.0（含 splash2x）中标准测试程序的访存特征，实验结果显示 EMAT 方法可以快速有效的找出程序的访存特征。

关键词： 访存序列；聚类抽样；基本性能计数向量；标准测试程序；EMAT

* Supported by the National Natural Science Foundation of China under Grant No. 61272132; the Huawei Technologies Co. Ltd. under Grant No. YBCB2011030.

作者简介：查中彬(1991 –)，男，硕士研究生，学生，主要研究领域为计算机体系结构；崔泽汉(1989 –)，男，博士研究生，主要研究领域为计算机体系结构；黄永兵(1987 –)，男，博士研究生，主要研究领域为计算机体系结构；陈荔城(1986 –)，男，博士研究生，主要研究领域为计算机体系结构；包云岗(1980 –)，男，博士，副研究员，主要研究领域为计算机体系结构；陈明宇(1972 –)，男，博士，研究员，主要研究领域为计算机体系结构。

1. 引言

在电子商务，搜索引擎，数据挖掘等诸多计算机应用领域，数据量正在以极快的速度增长，例如 Google 在 2007 年每天需要处理的数据量就超过 20PB[3]。随着大数据时代的到来，需要处理的数据量越来越多，处理器与内存之间的性能差距越来越大。计算机内存系统是影响体系结构、系统软件和应用软件性能的重要因素之一。由于内存系统已经成为制约系统整体性能的瓶颈，所以改进内存系统的设计显得越来越重要。内存系统面临多核技术，复杂应用和 I/O 技术等多方面的影响。为了找到内存系统性能瓶颈存在的根源，需要更加具体的了解程序的访存特征，针对性的对内存系统进行改进，因此获取准确和完整的访存序列数据显得尤为重要。

访存序列是用于记录由一系列的读写指令引起的内存访问的信息，主要包括时间信息、读写标识和访问的实际物理地址，能够用于反映指令级的体系结构信息。图 1 是一段常见访存序列的格式，数值均以 16 进制显示。其中第一列是两个访存行为之间的时间间隔，单位大小是一个指令周期；第二列是读写标识信息；第三列是内存被访问的实际物理地址。

```
a    r    1c33280
e    w    b6b05200
40   w    1c33280
14   r    9ca4ec00
```

Fig 1 The format of a conventional memory trace

图 1 一段常见访存序列的格式

目前收集访存序列的方法可以分为：基于软件（software-based）的方法和基于硬件（hardware-based）的方法。针对访存序列的收集工作，Uhlig 和 Mudge[8]提出了一个理想的访存序列收集工具应该具有：完全、详细、保真、便携、快速、低成本和易操作的特性。我们认为保真性和完全性是其中最重要的特性，因为缺少这两点特性的访存序列无法真实反映程序的访存行为。只有模拟器和硬件侦听机制能够满足这两点要求，但是由于硬件逻辑的复杂性和时间的限制，几乎所有的软件模拟器都无法完全模拟真实的硬件行为，所以我们认为硬件侦听是最理想的访存序列收集方法。

通过硬件探听方法能够准确收集程序的完整访存序列，但是大多数标准测试程序完整的访存序列数据较大，比如我们收集的 SPEC CPU2006 中 470 程序产生的访存序列数据量约 500GB。面对如此大数据量的访存序列，无法直接用于程序特征的分析，所以目前常用的方法都是从完整的访存序列中抽取出部分片段使用。在程序的整个过程中，由于每个程序都有不同的访存特征，并且同一个程序的不同时刻的访存特征也有可能相差很大。若只想分析部分数据就能获取程序完整的访存特征，则需要抽取出部分具有足够代表性的访存序列片段（Representative Memory Access Trace）。只有具有足够代表性，才能认为这些访存序列片段的访存行为是程序的访存特征。

为了准确并快速的抽取出代表性访存序列片段，本文提出一种基于聚类抽样的访存特征分析方法 EMAT（Extracting representative Memory Access Trace）对程序的访存特征进行分析。EMAT 方法的基本思想是在通过硬件探听方法获取程序完整的访存序列的同时，使用软件工具获取处理器上性能计数器（Performance Counter）值的变化，得到由体系结构相关性能指标构成的基本性能计数向量 BPCV（Basic Performance Count Vector）。使用基本性能计数向量反映相应访存序列片段的访存行为，从而利用基本性能计数向量的聚类结果找出代表性访存序列片段。

通过使用软硬件工具协同工作，EMAT 方法可以在真实机器上仅需执行一次标准测试程序，就能得到完整的访存序列数据和用于聚类分析的基本性能计数向量。最后，我们使用 EMAT 方法抽取了 SPEC CPU 2006 和 PARSEC3.0 程序集中所有程序的访存序列数据，通过实验分析，使用 EMAT 方法找出的代表性访存序列片段能够有效反映程序的访存特征，具有 96%的准确性。使用 EMAT 方法，从原始数据集（约 10TB，共 51 个程序）中只需抽取出 8.8GB 的数据，就能代表程序的访存特征，为进一步提高内存系统性能的研究提供了重要的参考信息。同时，本文将基本性能计数向量引入代表性访存片段的聚类分析的思路具有一定的通用性，易于移用于体系结构相关的其它研究方法中。

本文的组织结构如下：第 2 章介绍与本文有关的相关工作。第 3 章详细介绍 EMAT 方法的流程和具体实现中主要需要解决的问题。第 4 章中通过实验分析，验证了 EMAT 方法的有效性和准确性。第 5 章对 EMAT 方法进行总结。

2. 相关工作

目前在分析数据量较大的访存序列时，通过统计学方法找到一个原始数据的样本集合是一种比较直观的方法，比如 R. E. Wunderlich 等人中提出的 SMARTS[9]固定步长抽样方法。但是这种方法虽然简单，但是却无法保证抽取的片段能够反映程序所有的访存特征，往往只能反映出程序最明显的一类特征。我们认为尽管有些访存特征所占比例很小，但是依然是原始程序访存特征的一部分，需要在抽样结果中体现出来。通过后面的对比实验，证明了 EMAT 方法比 SMARTS 更能够全面的反映程序的访存特征。

Hamerly 等人提出的自动分析大规模程序行为的方法[6]是一种典型的抽取程序代表性片段的方法。这种方法首先通过 SimpleScalar[2]和 ATOM[1]对程序执行状态进行分析，得到程序的基本块向量 BBV[7]（Basic Block Vector），然后用 SimPoint[4]对基本块向量进行聚类分析从而找出其中的代表性代码片段。最后再次运行被测程序，将其中的代表性代码片段的访存序列记录下来。这种方法的主要不足之处在于需要利用软件模拟器收集基本块向量和访存序列，而且这两份数据的收集需要分两次进行。Jiaxin Li 等人提出的使用多种抽样粒度进行采样的方法[5]能够减少详细模拟的时间，但是依然需要收集基本块向量等信息。前面提到，由于硬件逻辑的复杂性和时间的限制，我们认为硬件侦听是最理想的访存序列收集方法，所以这种使用模拟器的方法时间效率上较差。

本文提出一种基于聚类抽样的访存特征分析方法 EMAT 对程序的访存特征进行分析。使用 EMAT 方法能够快速的抽取代表性访存序列片段，只需用很少的代表性片段就能够反映出原始程序所有的访存特征。为进一步提高内存系统性能的研究提供了重要参考信息。

3. 基于聚类抽样的访存特征分析方法 EMAT

为快速准确分析程序的访存特征，本文提出一种基于聚类抽样的访存特征分析方法 EMAT。只需要执行一次标准测试程序就能够获得用于聚类分析的基本性能计数向量 BPCV（Basic Performance Count Vector）并记录程序完整的访存序列。然后用聚类算法[10]离线对基本性能计数向量进行聚类分析找出其中的代表性片段，最后根据聚类结果从完整的访存序列中抽取出被选定的代表性访存片段，从而达到分析程序访存特征的目的。

3.1 EMAT 方法概述

EMAT 方法的基本思想就是同时收集程序完整的访存序列和用于聚类分析的基本性能计数向量，然后根据基本性能计数向量聚类的结果来决定最终需要抽取的代表性访存序列片段。为了实现 EMAT 方法，主要需要解决下面两个问题：

1）需要合理定义用于聚类分析的基本性能计数向量，使得每个基本性能计数向量能够反映出对应访存片段的访存特征；

2）如何区分出原始访存序列中的不同片段，并使其与每个基本性能计数向量形成一一对应的关系。

针对第一个问题，重点是解决如何定义基本性能计数向量。硬件性能事件被广泛使用于针对程序特征分析的微体系结构研究，本文同样采用硬件性能事件来进行程序的特征分析，从而得到用于聚类分析的基本性能计数向量。由于本文只关心程序的基本访存特征，所以只选出访存相关的六个性能指标来构成基本性能计数向量。图 2 给出了一个基本性能计数向量的具体格式，其中读/写非空周期数是指在一个固定时间间隔内，内存至少收到一次读/写请求的周期数，其余都是常用的体系结构相关的性能指标。这些性能指标可以从性能计数器值中直接或经过简单的计算获得。在我们的观点看来，这六个性能指标能够反映程序的绝大部分访存特征。

访存带宽	读写请求比例	DRAM Row Buffer 失效率	最底层缓存失效率	读非空周期数	写非空周期数

Fig 2 Format of Basic Performance Count Vector

图 2：基本性能计数向量格式

第二个问题主要是解决访存序列数据与基本性能计数向量之间的同步问题。我们利用软件每隔固定时间间隔收集一次性能计数器中的值，然后使用每个得到的基本性能计数向量反映该固定时间间隔内访存序列的访存行为，为此需要对原始访存序列进行相同时间间隔的分割。我们使用在访存序列中插桩的同步方式，达到基本性能计数向量与访存序列的同步。做法是在软件工具每次收集性能计数器中的值的同时，访问一个特殊的物理内存地址，称为 Tag 地址。这个 Tag 地址处在硬件预留的一段特殊物理内存区域内，用户程序和 OS 都不能访问。只需要通过简单的配置，就能使硬件探听工具能够侦听并记录这个 Tag 地址的访问操作。

3.2 EMAT 方法实现

在 EMAT 方法中，我们使用硬件访存行为监测工具 HMTT[11,12]和一种轻量级的性能计数器监测工具 TopMC[13]（Performance Count Monitor Tool）分别收集完整的访存序列数据和性能计数器信息。

HMTT 系统是我们自主研发的一个独立于平台的硬件访存行为监测与分析系统，通过硬件侦听内存总线上的访存信号，实时获取全系统完整访存物理地址序列。HMTT 能够得到包括物理地址、读/写、时间戳、进程号、虚拟地址等信息。TopMC 是一个轻量级的性能计数器监测工具，用户只需简单的指定需要收集的性能计数器名称，就可以每隔一段时间收集一次相应的性能计数器中的值。同时 TopMC 支持许多不同的处理器架构，具有很好的移植性。

在具体实现中，我们在启动标准测试程序后，立即启动 HMTT 和 TopMC 进行相应数据的采集。为建立访存序列与基本性能技术向量之间的对应关系，首先需要对访存序列进行分割。图 3 描述了访存序列的分割原理。最终通过 HMTT 收集的访存序列中包含两个部分：物理内存的访存序列和用于同步的 Tag 地址。在两个连续的同步 Tag 地址之间的访存序列属于同一个时间间隔，时间间隔的长度为 2ms。通过 Tag 地址的同步，使得第 n 个基本性能计数向量与以第 n 个 Tag 地址开始的访存序列片段建立了一一对映的关系。

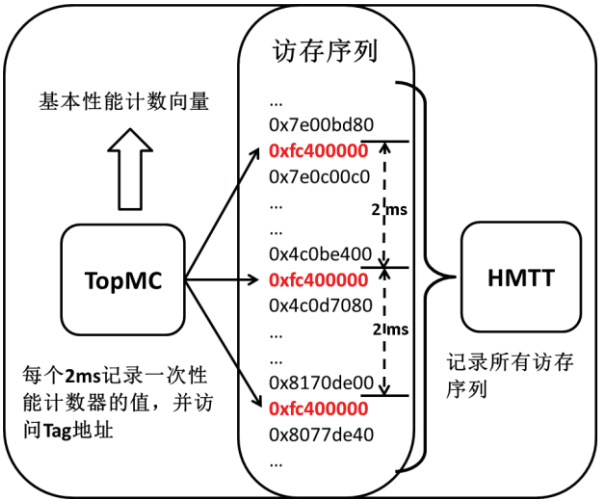


Fig 3 The principle of trimming Memory traces

图 3 访存序列分割原理

将访存序列与基本性能计数向量同步之后，就能够使用基本性能计数向量反映对应访存片段的访存特征，从而可以根据基本性能计数向量的聚类结果找出程序的访存特征。每一个聚类算法对于聚类向量的每一维都会有一个权重的分配，本文假定基本性能计数向量各维之间相互独立且权值相等。但从图 2 可以看到，基本性能计数向量各维的值之间差异较大，比如访存带宽可能是 2000（MB/s），而最底层缓存（Last Level Cache）失效率只能在 0,1 之间，所以不能直接用得到的性能计数向量来进行聚类分析，需要对各维进行归

一化处理。本文采用的策略是求出每一维的最大值，组成一个最大值基本性能计数向量。之后用每一个基本性能计数向量除以这个最大值性能计数向量进行归一化处理。

由于 SimPoint3.0 已经是一个功能完善的聚类工具，使用 K-means 算法进行聚类分析。本文同样使用 SimPoint3.0 的聚类功能对归一化处理后的基本性能计数向量进行聚类分析，主要得到每个性能计数向量的类型号和与该类中心点之间的距离。

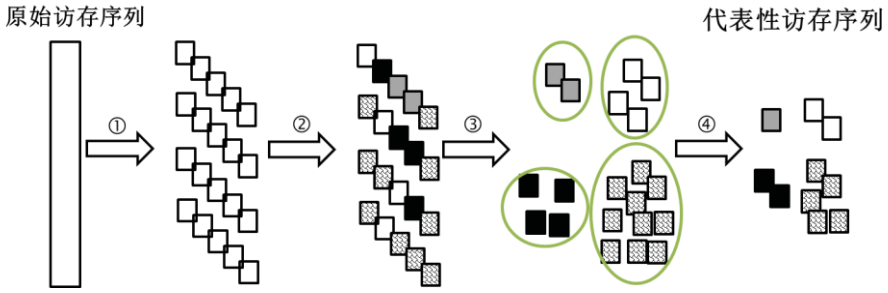


Fig 4 The workflow of trimming memory traces

图 4 切割访存序列的流程

在选定抽取的数量后，就可以按不同类别基本性能计数向量集合所占比例确定需要从中抽取出的片段个数，并按每个性能计数向量与该类中心点之间的距离的大小排序决定最终选取的访存序列片段代表程序的访存特征。最后从完整的访存序列数据中切割出被选取的代表性访存序列片段。图 4 中描述了详细的流程：

① 通过 Tag 地址的同步，将原始访存序列分成时间间隔为 2ms 的访存片段；

② 建立每个访存片段与其对应的基本性能计数向量的一一对映关系；

③ 根据基本性能计数向量的聚类结果将访存片段进行分类；

④ 从各类访存片段中按比例抽取出反映程序访存特征的代表性片段。

在我们的实验中，得到的原始数据集约为 10TB（共 51 个程序），为方便存储，我们分别从每个程序原始数据中抽取 20000 个代表性片段(约 416GB)放入硬盘和 400 个代表性片段(约 8.8GB)刻入 2 张 DVD。

本文提出的 EMAT 方法的核心思想在于利用体系结构相关指标构成的基本性能计数向量来反映对应访存片段的行为，从而达到根据基本性能计数向量的聚类结果分析程序访存特征的目的。在接下来的实验分析章节中，将证明 EMAT 方法的高效性和准确性。

4. EMAT 方法性能评测

4.1 实验平台和标准测试程序集

我们在一台拥有 2 个 Xeon E5504 处理器的机器上使用 TopMC v3 和 HMTT v3 进行实验，机器具体参数如表 1 所示。我们为 HMTT 保留了 0.25GB 的物理内存，所以操作系统和应用程序实际可用的大小为 3.75GB。最终将 EMAT 方法用于分析了 PARSEC3.0（包括 SPLASH2x）和 SPEC CPU 2006 中的所有程序，其中 PARSEC3.0 中每个程序跑满全部的八个核。

Table 1 The detail configurations of Machine

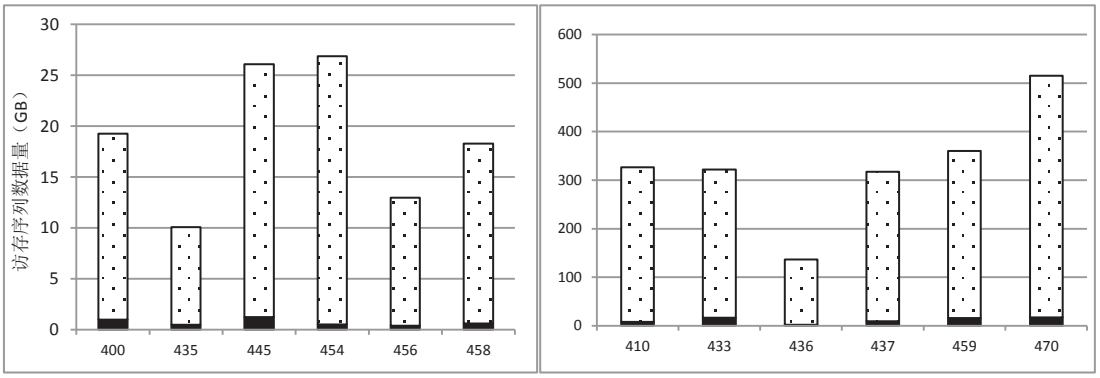
表 1 实验机器的具体配置

参数名称	详细配置
处理器	2.00GHz Intel Xeon E5504（四核）
物理内存	4GB
双列直插式存储模块	DDR3-800 RDIMM
峰值带宽	6.4GB/s
操作系统	CentOS 5.3
内核版本	2.6.32.18

后续章节将通过三个对比实验，验证 EMAT 方法不仅能够高效的找出程序的访存特征，同时具有很好的准确性，能够全面的反映原始程序的访存特征。首先在 4.2 节中通过实验数据说明 EMAT 方法只需使用很少的访存序列片段就能够反映程序的访存特征，接着在 4.3 节中比较不同抽样规模的 EMAT 方法抽样结果，验证 EMAT 方法的准确性。最后结合具体程序，对比 EMAT 方法、固定步长抽样方法和连续抽样方法的抽样结果，进一步验证 EMAT 方法的准确性。

4.2 EMAT 方法抽样效果

我们使用 EMAT 方法分析了 SPEC CPU2006 和 PARSEC3.0 所有程序的访存。正如 3.2 节中所说，如果设定从每个程序中挑选出 20000 个代表性片段，只需从 10TB 的数据中抽取 416GB 的数据就能够代表原始数据的访存特征。并且在 4.3 中将进一步证明选取更少数量的代表性片段，依然能代表原始数据的访存特征并保持基本性能计数向量中各维体系结构指标的均值误差不超过 4%。由于篇幅有限，图 5 中只展示了 SPEC CPU2006 中部分程序抽取 20000 个代表性片段的抽样效果。横坐标是标准测试程序的名称，纵坐标是每个程序对应原始访存序列数据大小和抽样之后的大小，单位为 GB。可以看出 EMAT 方法只需抽取原始数据中的很少的数据量就能够反映程序的访存特征。其中对于 SPEC436 的效果最好，原始数据是抽取出数据大小的 90 倍，如果只抽取出 400 个代表性片段，则原始数据大小是抽取出数据大小的 4600 倍。这样的抽样效果将大大提高利用访存序列分析程序访存特征的效率。



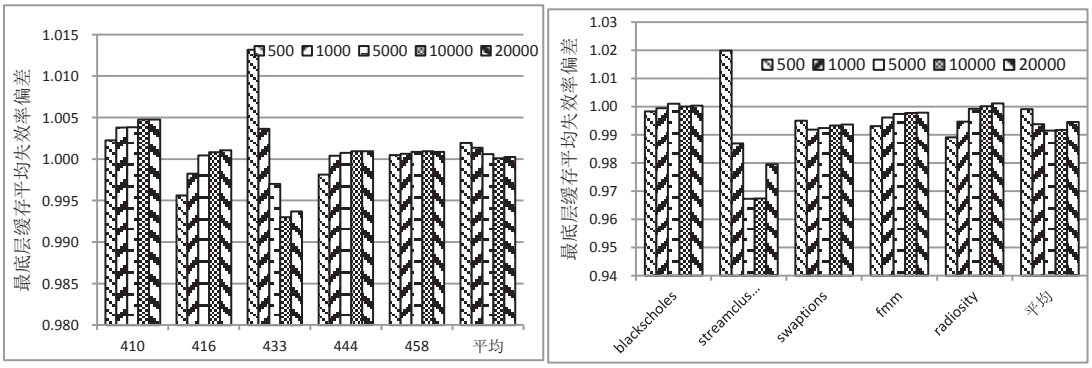
（a）访存稀疏型程序　　（b）访存密集型程序

Fig 5 Sampling results of some SPEC CPU2006 programs

图 5 SPEC CPU2006 部分程序抽样效果

4.3 准确性测试

为了验证EMAT方法的准确性，我们使用EMAT方法从原始访存序列数据中按不同规模分别抽取出500、1000、5000、10000 和 20000 个片段进行对比实验。



（a）SPEC CPU2006 部分程序　　（b）PARSEC 3.0 部分程序

Fig 6 The accuracy of LLC average miss rate with different sampling sizes

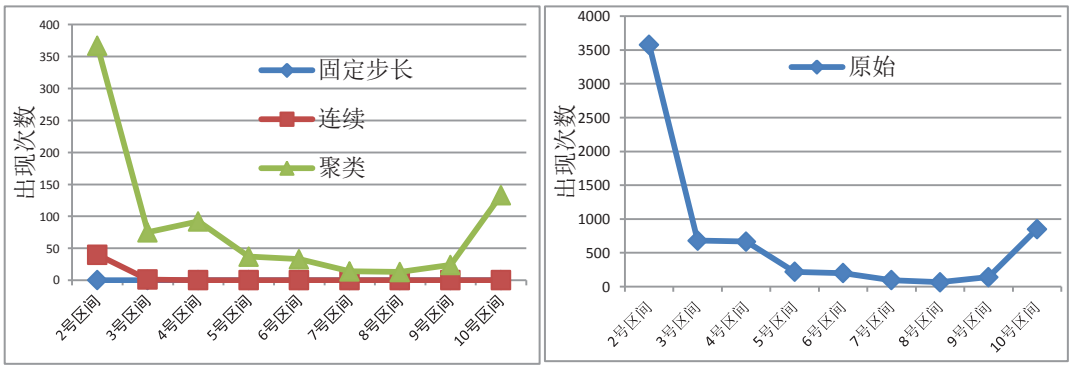
图 6 不同抽样规模最底层缓存平均失效率准确性

以抽样规模为 500 为例，对于基本性能向量的各维，如带宽，计算出 500 个代表性片段的均值，最终得到一个均值性能向量。不同的抽样规模会得到不同的均值性能向量，最终与原始数据的均值性能向量进行对比验证 EMAT 方法的准确性。图 6 中显示了抽取不同规模代表性片段最底层缓存平均失效率的准确性对比，由于篇幅有限，图 6 中只选取了部分程序的结果。其中横坐标是程序名称，纵坐标是代表性片段最底层缓存平均失效率占原始平均失效率的百分比，抽样规模分别对应 500、1000、5000、10000 和 20000。可以看出不管是对于最底层缓存失效率在 0.12 的程序 Streamcluster 和最底层缓存失效率高达 0.98 的 SPEC CPU 2006 的程序 416，代表性片段均值的偏差都控制在 4%以下。同时对于 SPEC CPU2006 程序集和 PARSEC3.0 程序集的平均准确率更是在 99%以上。通过图 6 的数据可以看出，本文提出的 EMAT 方法具有很好的准确性。

4.4 抽样方法对比

为了更进一步的验证 EMAT 方法的准确性，我们将 EMAT 方法与其他常用抽样方法进行对比，如 R. E. Wunderlich 等人中提出的 SMARTS 固定步长抽样方法和我们之前分析访存序列使用的连续抽样方法。其中固定步长抽样的步长根据程序执行时间长短和抽样规模确定，而连续抽样则是抽取从程序某一时刻开始的一段连续访存序列。最终通过实验数据显示只有 EMAT 方法抽样结果的访存特征分布与程序原始访存特征分布相同，进一步证明 EMAT 方法的准确性。

为了便于说明对比方法和结果，我们以 splash2x.radiosity 程序的带宽分布举例。首先找到 splash2x.radiosity 程序原始数据中带宽的分布区间，即找到最小带宽和最大带宽值。然后将整体带宽区间等分成 10 份，分别统计带宽值处在某一区间内的片段个数。最后对比使用三种不同抽样方法得到抽样结果的带宽分布，抽取规模为 20000 个。



（a）三种抽样方法结果带宽分布　　（b）原始带宽分布

Fig 7 The bandwidth distribution results of different methods

图 7　抽样结果带宽和原始带宽分布图

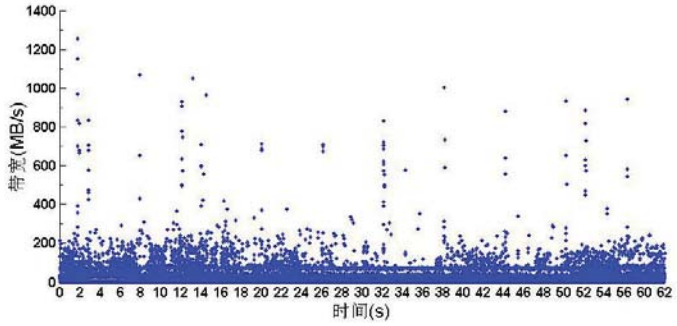


Fig 8 The bandwidth of splash2x.radiosity

图 8 splash2x.radiosity 的带宽变化

图 7 显示了三种不同抽样方法抽样结果的带宽分布图和原始带宽分布图。其中横坐标是 9 个等长区间的编号，纵坐标是带宽值出现在某区间内的片段个数。由于 splash2x.radiosity 程序有 96%左右的时间带宽都处在最小值区间[0.03051, 561.935)内，单位 MB/s，我们按递增顺序画出了后 9 个区间的分布情况，其中 10 号区间为最大值区间[5057.17，5619.08)。我们认为尽管这些区间所占比例很小，但是依然是原始程序访存特征的一部分，需要在抽样结果中体现出来。从图 7 中可以看出，EMAT 方法得到的抽样结果与原始程序的带宽分布很接近。图 8 是 splash2x.radiosity 程序一段时间内的带宽变化图，横坐标是执行时间，纵坐标是对应时刻的带宽值，单位是 MB/s。图 8 中可以看出，虽然程序大部分的时间带宽值都处在较低水平，但是每隔一段时间会出现若干带宽较高的时刻，这样的访存特征分布通过固定步长抽样和连续抽样是很难反映的。

综上分析，EMAT 方法比固定步长抽样和连续抽样方法能更准确的反映出原始程序的访存特征，更近一步的验证了 EMAT 方法抽取结果的具有很好的代表性和准确性。

5. 总结

本文提出了一种基于聚类抽样的访存特征分析方法 EMAT，并通过对 EMAT 方法有效性和准确性进行评测，得出 EMAT 方法是一种快速准确分析程序访存特征的方法。通过使用 EMAT 方法找出的代表性片段代替原始访存序列，只需使用很少的数据就能够反映程序的访存特征，提高了存储和移动的便利性。更重要的是，通过对于代表性访存序列片段的进一步分析，可以为提高内存系统性能的研究提供重要参考信息。同时将基本性能计数向量引入程序访存特征的聚类分析的思路具有一定的通用性，易于用在体系结构相关的其它的研究方法中。

致谢 在此，我们向所有给予本文建议和支持的老师和同学表示感谢。

Reference:

[1] A. Srivastava and A. Eustace. ATOM: A system for building customized program analysis tools. In Proceedings of the Conference on Programming Language Design and Implementation, pages 196-205.ACM,1994

[2] D. C. Burger and T. M. Austin, “The SimpleScalar tool set, version 2.0,” Technical Report CS-TR-97-1342, University of Wisconsin, Madison, June 1997.

[3] Dean, Jeffrey, and Sanjay Ghemawat. "MapReduce: simplified data processing on large clusters." Communications of the ACM 51.1 (2008): 107-113.

[4] Hamerly, Greg, et al. "Simpoint 3.0: Faster and more flexible program phase analysis." Journal of Instruction Level Parallelism 7.4 (2005): 1-28.

[5] Li J, Zhang W, Chen H, et al. Multi-level phase analysis for sampling simulation[C]//Design, Automation & Test in Europe Conference & Exhibition (DATE), 2013. IEEE, 2013: 649-654.

[6] Sherwood, Timothy, et al. "Automatically characterizing large scale program behavior." ACM SIGARCH Computer Architecture News. Vol. 30. No. 5. ACM, 2002.

[7] T. Sherwood, E. Perelman, and B. Calder. Basic block distribution analysis to find periodic behavior and simulation points in applications. In International Conference on Parallel Architectures and Compilation Techniques, September 2001.

[8] Uhlig, Richard A., and Trevor N. Mudge. "Trace-driven memory simulation: A survey." ACM Computing Surveys (CSUR) 29.2 (1997): 128-170.

[9] Wunderlich, Roland E., et al. "Statistical sampling of microarchitecture simulation." ACM Transactions on Modeling and Computer Simulation (TOMACS) 16.3 (2006): 197-224.

[10] Wu, Xindong, et al. "Top 10 algorithms in data mining." Knowledge and Information Systems 14.1 (2008): 1-37.

[11] Yungang Bao, Mingyu Chen, Yuan Ruan, et al. HMTT: a platform independent full-system memory trace monitoring system. in ACM SIGMETRICS, 2008.

[12] The homepage of HMTT: http://asg.ict.ac.cn/hmtt/

[13] The homepage of TopMC: http://asg.ict.ac.cn/projects/topmc/。