

基于预测极性动态变换的分支预测框架研究

陈晨 陈志坚* 孟建熠 严晓浪

(浙江大学超大规模集成电路设计研究所 杭州 310027)

摘要: 针对动态分支预测错误率在时间上分布不均匀且高错误率比较集中的特点, 该文提出一种可动态变换预测极性的分支预测方法。该方法对未经极性变换的原始动态分支预测错误率进行自适应监测, 筛选出原始动态分支预测错误率高于阈值的预测错误高峰期, 进而调整预测错误高峰期内分支预测器的预测极性, 使经过极性变换的最终动态分支预测错误率在程序运行过程中始终低于设定的阈值。该文同时研究了全局监测、按组监测和局部监测 3 种分支预测错误率监测方式。实验结果表明, 相同硬件资源下该方法比 Gshare 和 Bi-Mode 分支预测方法具有更高的分支预测精度。

关键词: 大规模集成电路; 嵌入式处理器; 分支预测; 预测错误高峰期; 预测极性动态变换

中图分类号: TP302.2; TN47

文献标识码: A

文章编号: 1009-5896(2013)04-1001-06

DOI: 10.3724/SP.J.1146.2012.00650

Branch Prediction Based on Dynamic Polarity Transformation

Chen Chen Chen Zhi-jian Meng Jian-yi Yan Xiao-lang

(Institute of VLSI Design, Zhejiang University, Hangzhou 310027, China)

Abstract: Periods with high branch misprediction rate tend to be uneven and concentrated during execution of programs. To address this problem, a new branch prediction strategy is proposed, which based on dynamic polarity transformation. This approach monitors original branch misprediction rate whose polarity has not been transformed, and detects the periods with original branch misprediction rate higher than a threshold. These periods are called as peaks of misprediction. The polarity of original prediction results will be transformed to make the final prediction during peaks of misprediction. As a result, the final branch misprediction rate whose polarity has been transformed will always be lower than the threshold during execution of programs. The prediction method can be divided into three categories according to the monitoring mechanism, which are global monitor, set monitor and per-address monitor. The experimental results show that this methodology gives better prediction accuracy than Gshare and Bi-Mode prediction schemes for the same cost.

Key words: VLSI; Embedded processor; Branch prediction; Peaks of misprediction; Dynamic polarity transformation

1 引言

随着高端嵌入式应用对处理器的性能要求越来越高, 超深流水线、超标量、多指令发射等技术纷纷应用于高端嵌入式处理器设计中。高效准确的分支预测框架是上述技术潜能得到发挥的前提和基础^[1]。基于神经网络的分支预测方法^[2,3]具有很高的分支预测精度, 但神经网络算法复杂度高, 资源消耗大, 不适合于嵌入式应用。当前高端嵌入式应用中广泛采用的分支预测方法为二级自适应分支预测方法^[4], 该方法将分支指令地址和分支历史信息进行组合, 从分支模式表中索引得到分支预测结果。分

支模式表为每个动态运行的分支指令分配一个分支预测器^[5], 预测器通过训练记录对应分支指令最近的分支跳转信息, 为分支指令再次执行提供预测信息。

分支预测错误的主要原因是破坏性分支重名^[6,7], 因此很多研究将重点放在减小破坏性分支重名概率上。文献[8]改变分支模式表中预测器的表征意义, 表征的内容由原本的分支跳转方向改变为分支跳转方向是否和分支目标缓存器(BTB)中的偏置比特一致, 该方法减小了破坏性分支重名概率, 其缺点是需要结合 BTB 协同工作, 设计复杂度高。文献[9]将分支模式表分为一个跳转型子表和一个非跳转型子表, 再增设一个选择模式表来决定使用哪个子表的预测结果, 该方法使每个子表中发生破坏性分支重名的概率降低, 缺点是该方法基于两次预测

2012-05-25 收到, 2012-12-20 改回

*通信作者: 陈志坚 chenzy@vlsi.zju.edu.cn

的叠加,一定程度上增加了预测不确定性。文献[10,11]在 TAGE 分支预测方法^[12]的基础上增加辅助预测器,弥补主预测器针对特定应用的不适应性,解决了包括破坏性分支重名在内的多种影响分支预测精度的因素,但该方法硬件资源耗费很大。文献[13]通过设立错误分支表来记录预测错误分支指令的地址和间距,通过计数机制探测预测错误分支指令并对其进行纠正,从而消除分支预测错误,但该方法同样基于两次预测的叠加,增加了预测不确定性。

本文结合程序局部性原理,从研究分支预测错误的行为特征出发,针对分支预测错误在时间上的局部性分布规律,提出一种通过动态调整分支预测器预测内容从而降低预测错误率的分支预测方法。该方法对未经极性变换的原始分支预测结果进行采样,并以分支预测错误率 50%为阈值进行研究,定义原始动态分支预测错误率高于该阈值的局部时间段为预测错误高峰期。动态监测预测错误高峰期的出现,并实时变换预测错误高峰期内分支预测器的预测极性,即原本预测跳转的变换为预测不跳转,反之亦然。本文通过实时监测预测错误高峰期并动态调整预测极性,降低了经过极性变换的最终动态分支预测错误率,提高了分支预测的稳定性与整体效率。

本文提出的方法与上述相关工作的区别在于:对比于文献[8,9],本文提出的方法不以减小破坏性分支重名的概率为目的,而是通过研究分支预测错误本身的行为特征进而提高分支预测精度,因而可以针对包括破坏性分支重名在内的所有导致分支预测错误的因素进行优化;对比于文献[10,11],本文提出的方法没有采用辅助预测器,因而在硬件资源上具有更高的利用效率;对比于文献[13],本文通过动态地监测分支预测错误率,采用自适应的监测机制探测预测错误高峰期,因而具有更好的实时性。

2 分支预测错误的时间局部性分析

根据程序局部性原理,一部分代码会在短时间内被反复执行,这部分代码称为程序热点。不同程序热点的交替执行占据了整个程序的大部分运行时间。一旦程序热点中存在破坏性分支重名等特殊的分支行为,则会导致程序在短时间内集中出现大量分支预测错误。本文以嵌入式测试基准程序 Powerstone^[14]作为目标程序对分支预测错误行为进行分析。表 1 是基准程序的简单介绍。

采用二级自适应分支预测方法对目标程序进行分支预测,分析程序运行过程中分支预测错误的行为特征。分支模式表共有 2^{13} 个分支预测器,以单个

表1 Powerstone测试基准程序简介

测试基准	指令数	动态分支数	功能描述
adpcm	124237	3801	音频编码
des	281233	2915	数据加密
fir	3055430	173054	信号滤波
g3fax	2775966	168251	G3fax 解码
posag	602670	38245	Pocsag 编码
ucbqsort	505895	29931	UCB 快速分类

分支预测器为研究对象,分别对 2^{13} 个预测器进行动态监测。对于每个预测器,选取时间上连续 10 次访问作为一个局部监测时间段,实时统计该时间段内的动态分支预测错误率;同时统计程序运行结束后该预测器整体的静态分支预测错误率。为了保证有足够的分支预测错误次数以便分析其行为特征,将每个目标程序运行过程中发生分支预测错误次数最多的预测器作为研究样本,并选取各个样本在程序运行过程中发生分支预测错误较多的一段时间进行预测错误行为分析。图 1 是每个研究样本在选取时间内的分支预测错误率,本文将图中所示未经极性变换的分支预测错误率定义为原始分支预测错误率(包括动态和静态),同时将原始动态分支预测错误率高于 50%(阈值)的时间段定义为预测错误高峰期。从图中可以看出各个目标程序中研究样本的动态分支预测错误率以静态分支预测错误率为界限上下抖动,无论静态分支预测错误率高或低,都会多次出现动态分支预测错误率高于 50%(阈值)的时间段,即预测错误高峰期。在图中未显示的其余大部分时间段中,分支预测错误次数相对较少,也较少出现预测错误高峰期。因此预测错误高峰期具有集中出现的特点。

通过对比分析分支模式表中除研究样本以外的其他分支预测器,发现上述预测器研究样本的分支预测错误行为具有普遍意义,表现为:对于各个预测器,在未经极性变换的情况下,在程序运行过程中动态分支预测错误率在时间上分布不均匀,并以静态分支预测错误率为界限抖动,另外,无论该预测器静态分支预测错误率高或低,预测错误高峰期都容易在一段时间内集中出现。即分支预测错误具有时间上的局部性。

3 预测极性动态变换分支预测框架

根据分支预测错误的时间局部性,本文提出一种基于预测极性动态变换的分支预测方法。本方法的核心思想是:对未经极性变换的原始分支预测错误率进行自适应监测,筛选出程序运行过程中原始

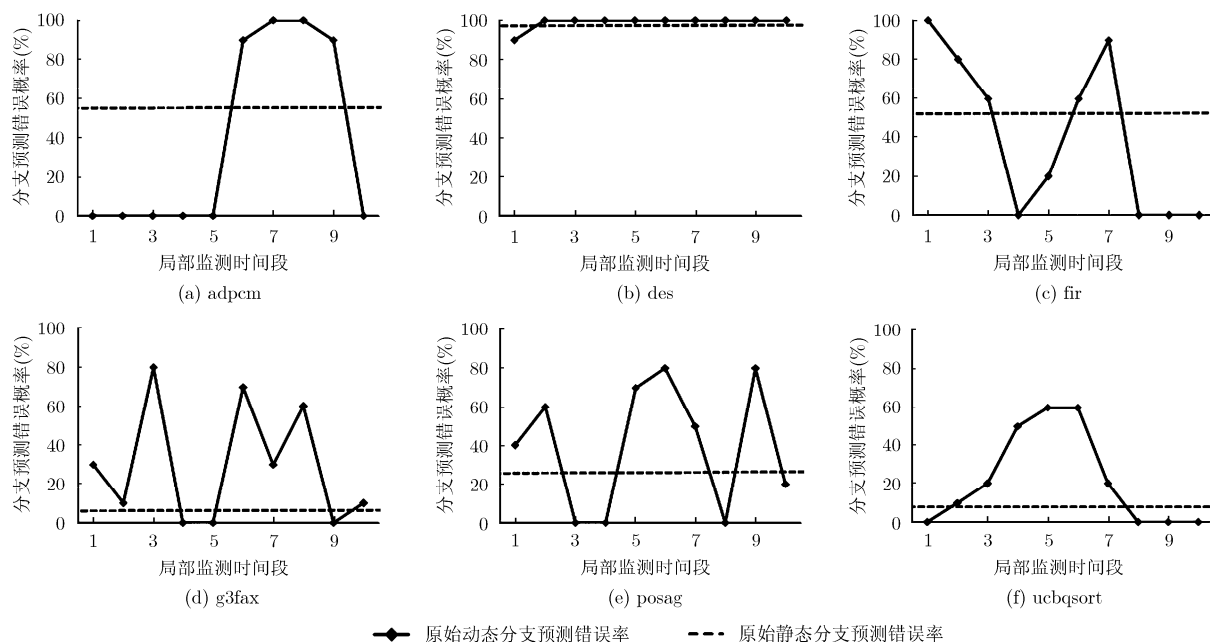


图1 分支预测错误的时间局部性

动态分支预测错误率高于 50% 的预测错误高峰期, 将预测错误高峰期内的预测极性进行变换, 即原本预测跳转的变换为预测不跳转, 反之亦然, 使得经过极性变换的最终动态分支预测错误率在程序运行过程中始终保持在 50% 以下, 减小最终动态分支预测错误率抖动幅度, 降低整体分支预测错误率。

基于预测极性动态变换的分支预测方法中, 最基本的硬件单元是预测错误高峰期监测器。预测错误高峰期会根据具体应用和输入数据不同而动态出现。本文通过自适应的动态监测机制来判断预测错误高峰期, 监测器通过状态机控制。一种状态机的状态转换图如图 2 所示。图中 C(Correct)表示未经极性变换的原始分支预测结果(即预测器自身)和正确分支跳转结果一致, M(Mismatch)表示未经极性变换的原始分支预测结果和正确分支跳转结果不一致。初始状态为 IDLE, 每当确认 C 发生后, 直接越过相邻状态跳转到下一个状态, 例如从 M4 到 M2, 从 M2 到 IDLE 等; 每当确认 M 发生后, 跳转到相邻状态, 例如从 C3 到 C2, 从 M1 到 M2 等; 程序运行过程中, 处于图 2 虚线左侧状态(M3, M4)的时刻均为预测错误高峰期。此状态机的运行方式加强了 C 发生的影响权重, 这便于筛选出预测错误率较高(例如 90%)的预测错误高峰期。这样的设计更利于整体性能的提高, 因为对于预测错误率较低(例如 60%)的预测错误高峰期, 由于动态监测机制本身具有一定的误差性, 若强行对预测器进行极性变换, 往往达不到理想效果。

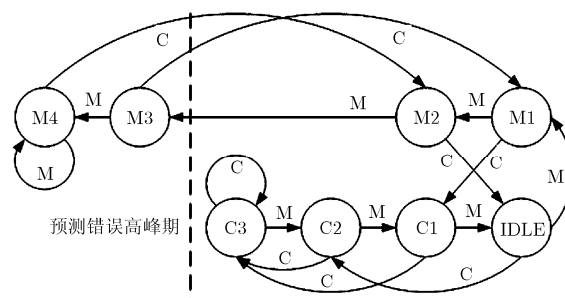


图2 监测器状态转换图

基于预测极性动态变换的分支预测框架的一种实现如图 3 所示, 预测电路由分支模式表、监测器表和数据选择器组成。分支模式表由分支预测器构成; 监测器表由图 2 所示的状态机构成。分支历史信息 and 分支指令地址进行异或操作后得到索引值, 从分支模式表中读出一个分支预测器的值作为原始分支预测结果; 同时用该索引值的一部分索引监测器表, 从中读出一个状态机的状态作为数据选择器的选通信号, 根据该状态决定最终分支预测结果: 若状态显示处于预测错误高峰期, 则将原始分支预测结果的极性变换后作为最终分支预测结果, 反之则直接将原始分支预测结果作为最终分支预测结果。

监测器表中的状态机在分支预测信息被后级流水线确认后被更新, 每次仅更新被确认分支指令所对应的状态机, 根据原始分支预测结果是否与正确分支跳转结果一致对状态进行转换, 状态转换方式如图 2 所示。分支模式表中的预测器仅当其输出的

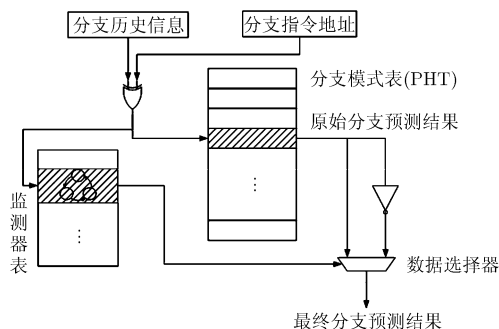


图3 基于预测极性动态变换的分支预测框架结构

原始分支预测结果与正确分支跳转结果不一致时被更新为正确分支跳转结果。

基于预测极性动态变换的分支预测方法按照监测方式不同可以分为：全局监测(GM)，按组监测(SM)和局部监测(PM)3种类型。各种类型的结构如图4所示。对于全局监测，整个分支模式表对应一个监测器，该监测器负责筛选出全局的预测错误高峰期。对于局部监测，分支模式表中每个预测器都对应一个监测器，各监测器负责筛选出对应于各预测器的预测错误高峰期。全局监测的监测粒度较粗，对预测错误高峰期的探测灵敏度低，但只需要一个监测器，硬件资源耗费极小，而局部监测的监测粒度更细，对预测错误高峰期的探测灵敏度更高，但监测器的数量需要和预测器一致，硬件资源耗费很大。按组监测是上述两种方式的折中，将分支模式表分为若干组，每组对应一个监测器，负责筛选出该组预测器对应的预测错误高峰期。此方式在监测灵敏度和硬件资源上相对比较平衡。

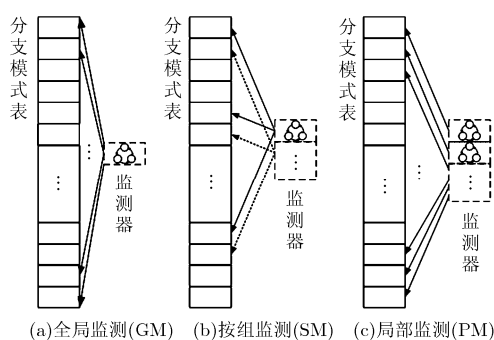


图4 3种监测方式结构

4 实验结果

本文以国产32位高性能嵌入式处理器CK610^[15]及其平台为基础，分别实现本文提出的基于预测极性动态变换的分支预测方法和现有的 Gshare 以及 Bi-Mode 分支预测方法，以 Powerstone 测试基准程序作为目标程序。先分析预测极性动态变换分支预

测方法自身几种监测方式的适用性，然后分析使用该方法后最终动态分支预测错误率的下降情况，最后给出该方法与 Gshare 以及 Bi-Mode 分支预测方法的综合预测错误率比较结果。

首先对预测极性动态变换分支预测方法自身的几种监测方式进行比较，图5表示3种监测方式在各种硬件资源配置下对于目标程序的平均预测错误率。当硬件资源很小时(对应于图5中小于4 kbit)，全局监测最优；当硬件资源增加后(对应于图5，在8~256 kbit之间)，按组监测最优；而当硬件资源进一步增加(对应于图5中大于256 kbit)，局部监测最优。其原因在于，对于不同的监测方式，监测器所消耗的资源不同，因此分配到分支模式表上的资源也不同。当硬件资源很小，分支模式表的容量问题成为影响分支预测精度的最主要因素，全局监测可用于分支模式表的硬件资源最多，因此可获得相对较好的分支预测结果；随着硬件资源的增加，分支模式表的容量问题得到缓解，但对于局部监测来说，每个预测器都需要一个对应的监测器，可用于分支模式表的硬件资源仍然有限，而按组监测则在分支模式表和监测器的硬件资源分配上取得了较好的平衡，因此可获得最好的分支预测结果；当硬件资源继续增加达到一定程度后，分支模式表的容量对分支预测精度的影响进一步减小，具有最高监测灵敏度的局部监测方式将获得最好的分支预测结果。现阶段嵌入式处理器中分支预测可接受的硬件资源约在64 kbit以内，从图5中可以看出按组监测方式在此区间内是最优的。

为了检验经过极性变换的最终动态分支预测错误率低于未经极性变换的原始动态分支预测错误率，采用预测极性动态变换分支预测方法对第2节所研究的分支预测错误率重新进行监测，研究样本和监测时间段均和图1保持一致，局部监测时间段也和图1保持一致，即选取时间上连续10次访问作为一个局部监测时间段。采用按组监测方式，分支

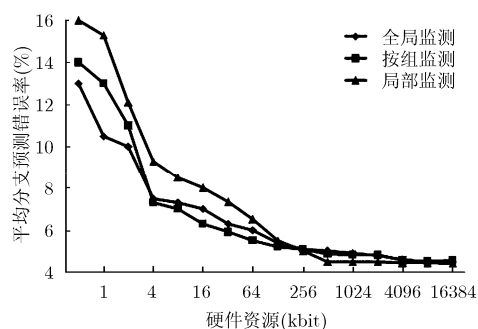


图5 3种监测方式预测错误率比较

模式表包含 2^{13} 个分支预测器, 使用 2^9 个监测器, 监测器状态转换采用图 2 所示方式。图 6 显示了各目标程序中原始动态分支预测错误率和最终动态分支预测错误率的比较情况, 可以看出, 通过极性变换, 最终动态分支预测错误率在整体上明显低于原始动态分支预测错误率。

图 7 显示了预测极性动态变换分支预测方法与 Gshare 以及 Bi-Mode 分支预测方法在各目标程序中分支预测错误率的比较情况。预测极性动态变换分支预测方法采用按组监测方式, 当硬件资源小于 1 kbyte 时, 分支模式表占用 7/8 的硬件资源, 监测器表占用 1/8 的硬件资源; 当硬件资源大于 1 kbyte

时, 分支模式表占用 3/4 的硬件资源, 监测器表占用 1/4 的硬件资源。监测器状态转换采用图 2 所示方式。从图中可以看出, 对于大部分目标程序, 预测极性动态变换分支预测方法均优于 Gshare 和 Bi-Mode 分支预测方法, 只有 ucbqsort 程序中, Bi-Mode 分支预测方法在特定硬件资源下产生最优的预测精度。另外, 预测极性动态变换分支预测方法的预测精度优势会在一定区间内达到最大值, 但随着硬件资源的增加, 精度优势会逐渐减弱, 对于 adpcm, des 这两个目标程序来说, 当达到一定硬件资源时, 3 种分支预测方法的预测精度基本一致。其原因是, 预测极性动态变换分支预测方法是以筛

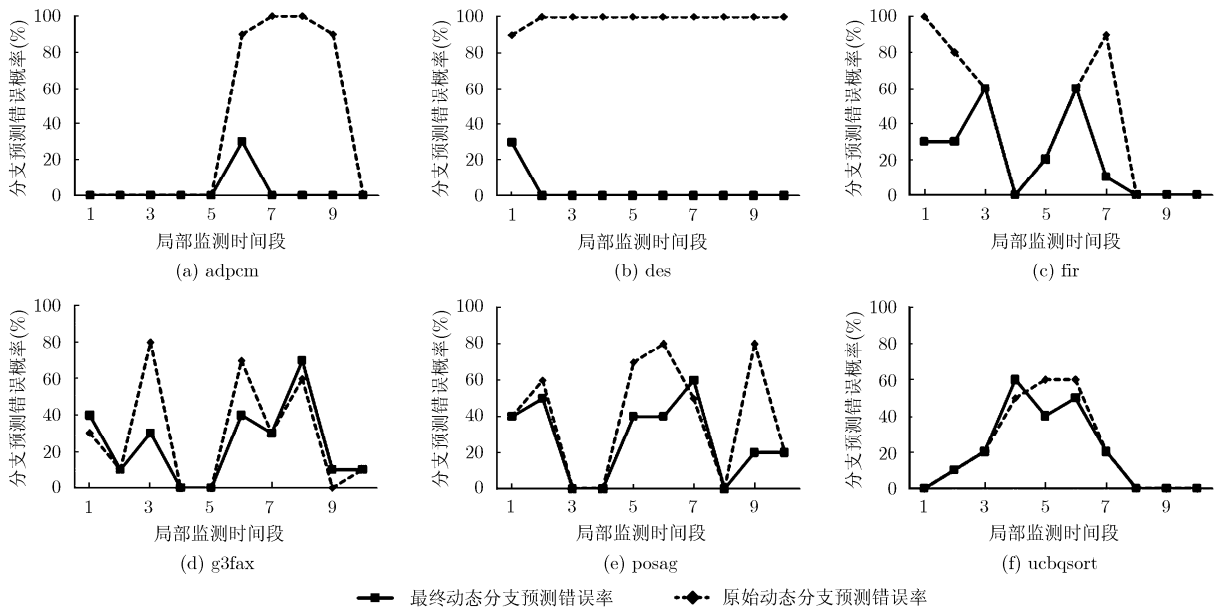


图6 最终动态分支预测错误率下降情况

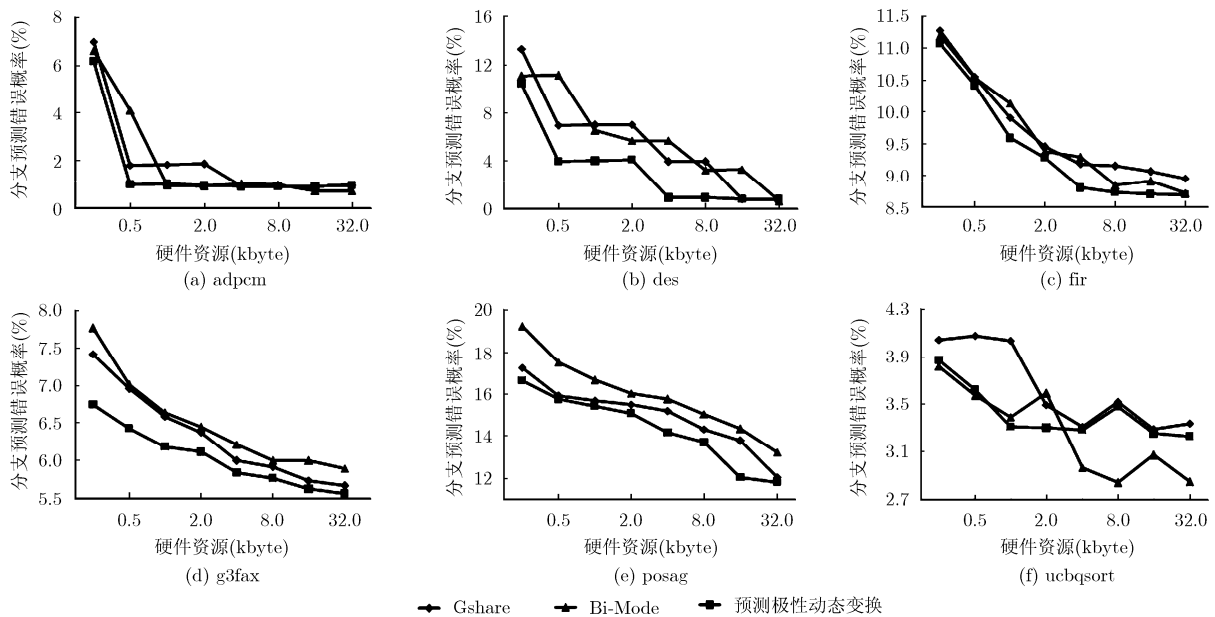


图7 预测极性动态变换分支预测方法和其他分支预测方法的预测错误率比较

选出预测错误高峰期为基础的,随着硬件资源增加,未经极性变换的原始分支预测错误率自身便会达到很低的水平,从而使得程序运行过程中出现的预测错误高峰期数量减少,各个预测错误高峰期自身的预测错误率也会降低,这给预测极性动态变换带来了困难。这也解释了为何 ucbqsort 程序中 Bi-Mode 分支预测方法产生最优的预测精度,因为该程序未经极性变换的原始分支预测错误率已经很低。

5 结束语

本文通过研究程序运行过程中分支预测错误的行为特点,提出一种基于预测极性动态变换的分支预测方法。该方法监测未经极性变换的原始动态分支预测错误率,筛选出程序运行过程中的预测错误高峰期,将高峰期内预测器的预测极性进行变换后作为最终分支预测结果,从而降低经过极性变换的最终分支预测错误率。根据预测错误率监测方式不同可进一步分为全局监测、按组监测和局部监测 3 种类型,在分支预测精度和硬件成本之间取得平衡。该方法可以满足高性能嵌入式处理器在可控成本内实现高精度分支预测的要求。

参 考 文 献

- [1] Hennessy J and Patterson D. Computer Architecture: A Quantitative Approach[M]. Fifth Edition, Beijing: China Machine Press, 2012: 162-167.
- [2] Jiménez D. An optimized scaled neural branch predictor[C]. 2011 IEEE 29th International Conference on Computer Design (ICCD), Amherst, 2011: 113-118.
- [3] Jiménez D. Oh-snap: optimized hybrid scaled neural analog predictor[C]. Proceedings of the 3rd Championship on Branch Prediction, Detroit, 2011: 9-12.
- [4] Yeh T and Patt Y. Two-level adaptive training branch prediction[C]. Proceedings of the 24th Annual International Symposium on Microarchitecture, New York, 1991: 51-61.
- [5] Zhang Long, Tao Fang, and Xiang Jin-feng. Researches on design and implementations of two 2-bit predictors[J]. *Advanced Engineering Forum*, 2011, 1(1): 241-246.
- [6] Young C, Gloy N, and Smith M. A comparative analysis of schemes for correlated branch prediction[C]. Proceedings of the 22nd Annual International Symposium on Computer Architecture, New York, 1995: 276-286.
- [7] Talcott A, Nemirovsky M, and Wood R. The influence of branch prediction table interference on branch prediction scheme performance[C]. Proceedings of the 3rd Annual International Conference on Parallel Architectures and Compilation Techniques, Manchester, 1995: 89-98.
- [8] Sprangle E, Chappell R, Alsup M, *et al.* The agree predictor: a mechanism for reducing negative branch history interference[C]. Proceedings of the 24th Annual International Symposium on Computer Architecture, New York, 1997: 284-291.
- [9] Lee C, Chen I, and Mudge T. The bi-mode branch predictor [C]. 30th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO'97), Research Triangle Park, NC, 1997: 4-13.
- [10] Seznec A. A 64 kbytes ISL-TAGE branch predictor[C]. Proceedings of the 3rd Championship Branch Prediction, Detroit, 2011: 13-16.
- [11] Seznec A. A new case for the TAGE branch predictor [C]. Proceedings of the 44th Annual IEEE/ACM International Symposium on Microarchitecture, New York, 2011: 117-127.
- [12] Seznec A and Michaud P. A case for (partially)-tagged geometric history length predictors[J]. *Journal of Instruction Level Parallelism*, 2006, 8(1): 1-23.
- [13] Sendag R, Yi J, and Chuang Peng-fei. Branch misprediction prediction: complementary branch predictors[J]. *Computer Architecture Letters*, 2007, 6(2): 49-52.
- [14] Scott J, Lea H, Arends J, *et al.* Designing the low-power M*CoreTM architecture[C]. Proceedings of IEEE Power Driven Microarchitecture Workshop, Barcelona, 1998: 145-150.
- [15] Meng Jian-yi. C-SKY microsystems, 32-bit high performance and low power embedded processor[OL]. <http://www.c-sky.com>, 2012, 5.

陈 晨: 男, 1985年生, 博士生, 研究方向为嵌入式处理器体系架构及其高性能和低功耗技术。

陈志坚: 男, 1984年生, 博士后, 研究方向为高性能嵌入式处理器设计、可重构处理器设计。

孟建熠: 男, 1982年生, 博士, 讲师, 研究方向为高性能低功耗嵌入式处理器设计研究和超大规模集成电路设计技术。

严晓浪: 男, 1947年生, 教授, 博士生导师, 研究方向为超大规模集成电路设计、嵌入式处理器设计、VLSI设计自动化技术。