

（深圳）

实验报告

开课学期： 2021春季

课程名称： 计算机体系结构(实验)

实验名称： 分支预测器设计

实验性质： 综合设计型

实验时间： 6 地点： T2210

学生班级：18计算机科学与技术7班

学生学号： 180110711

学生姓名： 郑漫莎

评阅教师：

报告成绩：

实验与创新实践教育中心印制

2021年4月

# 1. 实验内容

*简述实验内容*

本实验将基于Pin插桩分析的工作机理，使用C/C++实现分支预测器的软件模型，从而在进一步熟悉插桩工具使用的同时，加深对分支预测原理的理解。

实验包含基础实验和附加实验。基础实验实现三种动态分支预测方法，分别为基于BHT的分支预测，基于全局历史的分支预测，基于局部历史的分支预测。基础实验还包括两种锦标赛分支预测，分别为基于全局历史的选择方法和基于局部历史的选择方法。

附加实验实现Bi-mode动态分支预测和分支目标的检测。

文件brchPredict.cpp中为各预测器的实现代码，brchPredict\_addr.cpp为实现分支目标地址检测的代码。

# 2. 设计与实现

## 2.1 题目分析

*描述个人对实验题目、实验内容和实验要求的分析和理解，并得出相应的结论，或根据分析得出设计方案。*

和静态分支预测相比较，动态分支预测能根据分支指令过去的表现来预测将来的行为。在分支行为发生变化，那么分支预测的结果也相应的发生改变。因此，动态分支预测具有更高的预测准确率和不同条件下的适应性。分支预测分为两部分，预测和更新。

基于BHT的分支预测是根据分支历史表来记录分支指令的历史行为。如图2-1所示，Tag字段是指令地址的一部分，分支历史由2bit饱和计数器组成。在预测过程中，在一定程度上，Tag字段可以近似为一个指令地址，随着分支的跳转，每个Tag也即每条指令地址维护一个分支历史。在分支历史的基础上，进行跳转的判断。分支历史采用2bit饱和计数器，如图2-2所示，00为强不跳转，01为弱不跳转，10为弱跳转，11为强跳转。对于一直跳转的指令，即11，当突然不跳转时，转为10，给一次机会，预测下一次还是为跳转，如果再次为跳转，则认为下次不跳转，转为01，否则转为11，反之亦然。



图 2‑1 基础BHT的结构

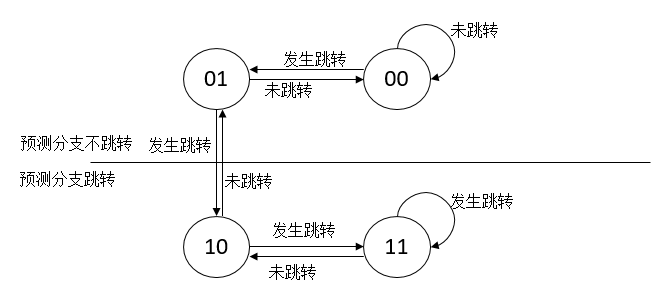


图 2‑2 基于2bit饱和计数器的分支预测

基于全局历史的分支预测是在BHT的基础上增加了GHR，将所有分支指令关联起来。如图2-3，k比特的GHR记录最近k条分支指令的历史跳转方向，PHT的Tag字段由分支指令地址和GHR异或得到。在一定程度上，PHT的每一行可以对应为在过去k条分支指令的跳转历史为GHR的情况下，分支指令对应的分支历史，很大程度上，减少了同一指令由于跳转历史不同而造成的的结果冲突。例如，A->C，B->C为A调用C和B调用C，对于C而言的分支指令可能由不同的跳转结果，但对应的分支指令地址相同。GHR通过左移位的方式进行更新，1为跳转，0为不跳转。

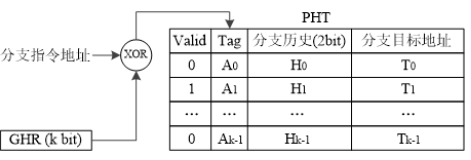


图 2-3 基于全局历史的分支预测

基于局部历史的分支预测用LHT代替全局历史预测中的GHR，因为不是所有分支指令都具有关联性。如图2-4所示，LHT每条记录包含Tag和局部转移历史，分支指令地址和对应的局部转移历史进行异或，得到的结果作为PHT的Tag。即近似为每条分支指令地址都维护一个局部转移历史，每条分支指令在不同的局部转移历史下维护一行分支历史。

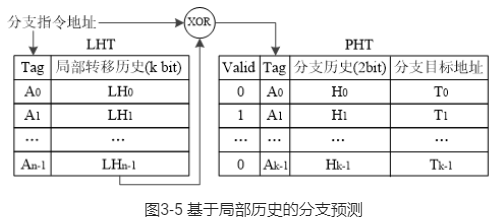


图 2-4 基于局部历史的分支预测

基于全局选择历史的锦标赛分支预测含两个子预测器和一个全局选择历史寄存器GSHR。GSHR的更新规则如图2-5，只有一个预测期结果正确时进行更新操作。子预测器按照跳转结果和预测结果进行预测。

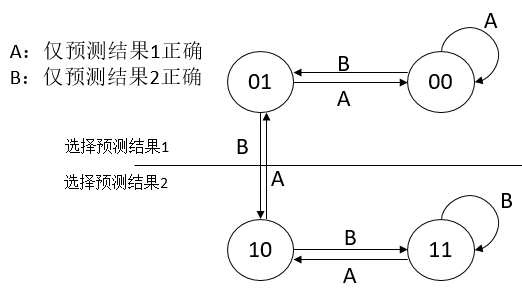


图 2-5 GSHR的更新规则

基于局部选择历史的锦标赛预测分支预测含两个子预测表和一个局部选择历史表LSHT，LSHT的结构如图2-1，LSHT每个相应的选择历史LSHT[i]的更新策略如图2-5。预测器按照跳转结果和预测结果进行预测。

基于Bi-mode的分支预测在全局历史的分支预测的基础上，一定程度上减少了分支别名干扰，即分支预测过程中PHT表项的冲突问题，不同的地址分支访问同一个表项，这是由于哈希的冲突。Bi-Mode如图2-6所示，将PHT表分为两层，分别为方向PHT和选择PHT。方向PHT分为跳转表PHTT和不跳转表PHTNT。和全局历史分支预测类似，利用PC和GHR的异或进行方向PHT的索引，用PC直接进行选择PHT的索引。根据选择PHT进行选择使用哪个方向表的结果。对于方向表而言，仅被选择的方向PHT会进行更新，更新策略与图2-2类似。对于选择表而言，当结果跳转时，选择表更倾向于跳转表，否则倾向于不跳转表。但是，当选择PHT选择方向和结果不一致，但选择的方向表的结果和跳转结果一致，即预测正确，则不更新选择表。

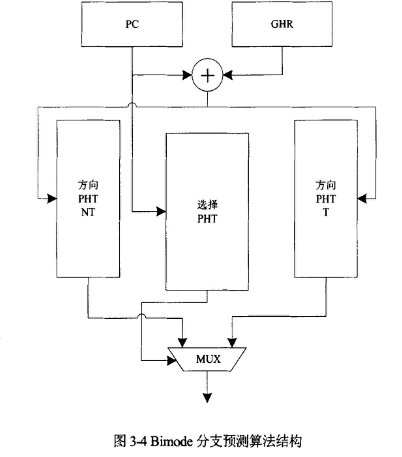


图 2-6 Bi-mode的分支预测算法结构

上述为各分支预测器的分析和设计方案。以下补充分支目标地址的预测和更新，分支地址和GHR等进行异或选择操作两点。

为预测分支目标地址，将BHT改进为图2-7，增加了有效位和分支目标地址字段。分支目标地址按照下述方案进行独立更新。当前行无效时，且分支指令不跳转，则不进行分支目标地址预测。当前行无效，但分支指令跳转，则将当前行标记为有效，复位饱和计数器，同时更新饱和计数器，并将分支指令的目标地址更新到BHT中。如果当前行有效，则用BHT的分支目标地址进行预测，预测失败时，当前行标记为无效。

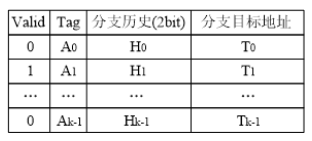


图 2-7 改进的BHT结构

分支指令地址和GHR等类似结构进行异或操作，将地址和历史信息进行结合。相对于与、或等其他操作，异或更加的中性化，不会偏向于0或者1，导致Tag分布不均匀。与操作容易使得Tag中的0偏多，相对而言1多的行就较少占用，容易加重冲突。或操作亦然。

## 2.2 实验过程

*图文并茂地展示实验分析、优化和求解的过程。对于关键实现，可用图表叙述（必要时可贴关键代码，但禁止大段粘贴代码）。*

基本过程：

预测：

BOOL predict(ADDRINT addr)

更新：

void update(BOOL takenActually, BOOL takenPredicted, ADDRINT addr)

### 2.2.1 基于BHT的分支预测

**变量：**

SaturatingCnt<2> counter[1 << L]

：维护Tag为L位的BHT表，每行含2bit的分支历史。

**预测：**根据addr低L位为索引的counter对应的分支历史，返回最高位，即0不跳转，1跳转

**更新**：根据addr低L位索引，counter对应的分支历史进行更新，如果结果跳转，则分支历史加一，否则减一。其中加一结果不超过11，减一结果不小于00。

**优化**：

1. 指导书中更新策略如图2-8，采用更新策略如图2-2。相对而言，图2-8的更新策略

*\*如果有做附加题，需按照指导书要求，在报告中添加相应的分析、设计过程和运行结果截图和对比等内容。*

## 2.3 实验结果及分析

*描述测试方法，列出实验结果，并进行相应的分析。*

*如果进行了优化，需给出优化前后的结果对比截图及文字分析（如果优化没有效果，也需分析原因）。*

# 3. 总结和感想

*请填写实验过程中遇到的问题及解决方案、学习本课程的收获和反思，并对课程提出合理的建议。*