1. 题目分析

本次实验要求用 C 语言实现软件层面的分支预测，通过在 PIN 中插桩，将分支指令出现的 PC 值作为输入参数，预测分支跳转方向，再根据后续指令的实际执行情况判断分支预测是否准确。依据实验要求，需要分别完成四种动态分支预测，分别是基于分支历史表的分支预测（BHT）、基于全局历史的分支预测（GHR）、基于局部历史的分支预测（LHR）、锦标赛分支预测。

2. 具体实现

四种方式都是在父类 BrchPrecdictor 的基础上完成 predict 与 update 函数来实现整个预测过程。

1) BHT

- 数据结构：

  设置一个 counter 数组用于记录分支历史。

- Predict 实现：

  通过截取 addr 的后 L 位查分支历史表（counter）即可得到 2bit 分支历史记录，再取分支历史记录的高位即为预测结果。

- Update 实现：

  BHT 的 update 函数实现如图 1 所示。

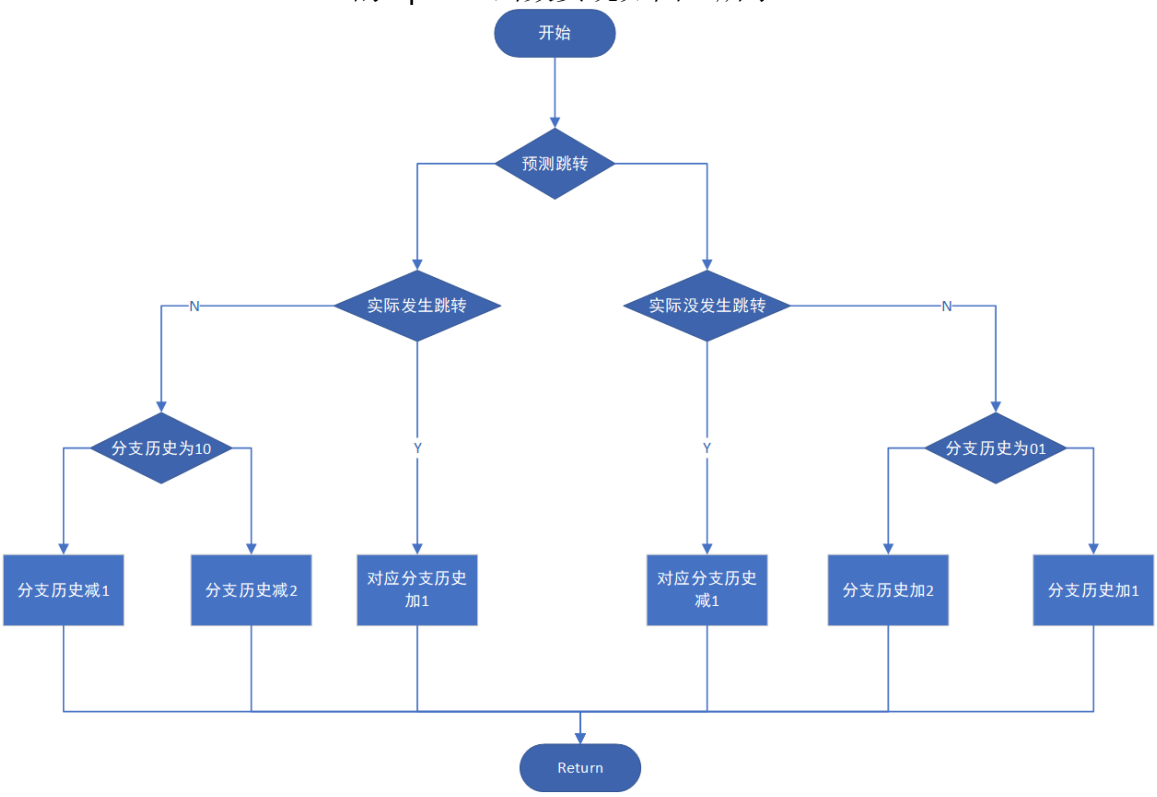

图 1 BHT 的 update 函数实现

2) GHR

- 数据结构

  在 BHT 的基础上引入一个 k 比特的 GHR 来记录所有最近 k 条分支指令的历史跳转方向。

  原来 counter 数组改名为 bhist。

- Predict 实现

  用 addr 与 GHR 异或的结果的低 L 位查分支历史表 bhist，然后取高 1 位作为预测结果返回。
- Update 实现

  逻辑同 BHT 实现，只是更新的表项修改为 addr 与 GHR 异或的结果的低 L 位。

  更新完 bhist 之后，需要进一步更新 GHR。具体修改为：将 GHR 左移 1 位，如果本次发生了跳转则移入 1，否则移入 0。

3) LHR
- 数据结构

  在 GHR 基础上将原来的 GHR 扩充为局部转移历史表 LHT，大小与分支历史表相同，页表项长度固定（6）。
- Predict 实现

  用 addr 的低 L 位取出局部转移历史，与 addr 进行异或，最后根据计算结果的低 L 位查分支历史表，取高 1 位作为预测结果返回。
- Update 实现

  逻辑同 GHR 实现，只是更新 GHR 的操作修改为更新 LHT 表的对应表项，更新逻辑同 GHR。

4) 锦标赛

选择实现基于全局历史的锦标赛分支预测器。
- 数据结构

  类似 GHR 设置一个 2bit 移位寄存器 GSHR，用于记录参与锦标赛预测的两个分支预测器的历史预测情况。

  此外类还需要接收两个分支预测器的类指针用于作为参加锦标赛预测的两个预测器。
- Predict 实现

  分别用 2 个预测器进行预测，然后根据 GSHR 的高 1 位确定返回哪个预测器的结果。
- Update 实现

  首先分别调用两个预测器的 update 函数对两个预测器进行更新，然后根据本次预测结果与跳转情况对 GSHR 进行更新，若只有预测器 1 预测正确，则对 GSHR 进行“减 1”操作；若只有预测器 2 预测正确，则对 GSHR 进行“加 1”操作；其余情况下不更新 GSHR。

3. 结果分析

实验结果如表 1 所示。可以看到同一预测器四个程序的预测准确率差距不大。四个预测其当中 BHT 效果最差，GHR 与 LHR 结果相近，相比 BHT 有较大提升，GHR 好于 LHR，而锦标赛预测在 GHR 与 LHR 的基础上有进一步的提升但提升效果远小于 GHR 相比 BHT 的提升效果。

	Bzip2	Sjeng	Wrf	Sphinx3
BHT	94.3541	94.3274	94.3763	94.3831
GHR	97.9263	97.932	97.9143	97.9328
LHR	96.7213	96.7412	96.7536	96.7287
锦标赛	98.0171	98.0088	98.0223	98.0176

表 1 分支预测器结果