# 高等计算机体系结构，2019 年春季

# 作业 4： 流水线 2（参考答案）

主讲教师：栾钟治

助讲教师：杨海龙；助教：许崇杨，左佩璇

作业下发时间：2019 年 4 月 15 日

作业回收时间：2019 年 4 月 29 日

## 1 分支预测 35 分

考察以下高级语言代码段:

```
int array[1000] = { /* random values */ };
int sum1 = 0, sum2 = 0, sum3 = 0, sum4 = 0;
for (i = 0; i < 1000; i ++) // 分支 1: 循环分支
{
if (i % 4 == 0) // 分支 2: If 条件分支 1
sum1 += array[i]; // 发生分支的路径
else
sum2 += array[i]; // 不发生分支的路径
if (i % 2 == 0) // 分支 3: If 条件分支 2
sum3 += array[i]; // 发生分支的路径
else
sum4 += array[i]; // 不发生分支的路径
}
```

(a) 当使用 last-time 预测器时三个分支的预测准确率分别是多少？(假设每一个分支的 last-time 计数器起始状态是‘不发生’) 请写出你的计算过程和依据。

参考答案：

分支 1:

998/1000 = 99.8%. 分支第一次执行时预测错误

分支 2:

500/1000 = 50%:

分支 3:

0%. 每次执行时分支都改变方向.

(b) 当使用基于 2-bit 饱和计数器的预测器时三个分支的预测准确率分别是多少？(假设每一个分支的 2-bit 计数器起始状态是'强不发生') 请写出你的计算过程和依据。

参考答案：

分支 1:

997/1000 = 99.7%. 分支头两次执行和最后一次执行（退出循环时）时预测错误

分支 2:

750/1000 = 75%. 分支重复 T N N N T N N N 模式，饱和计数器在“强不发生”和“弱不发生”之间转换 (每四次预测发生一次，在分支真正发生之后)，预测结果永远是不发生。

分支 3:

500/1000 = 50%。 分支重复 T N T N 模式， 每次预测饱和计数器在“强不发生”和“弱不发生”之间转换，每次预测结果是不发生，只有一半是正确的。

(c) 当分支 2 和分支 3 的 2-bit 计数器起始状态分别是 (i)'弱不发生'；(ii)'弱发生' 时，预测准确率分别是多少？请写出你的计算过程和依据。

参考答案：

(i) 弱不发生

分支 2:

749/1000 = 74.9%

分支 3:

0%. 预测器在“弱不发生”和“弱发生”之间振荡

(ii) 弱发生

分支 2:

749/1000 = 74.9%. 分支的模式是 T N N N T N N N， 头四次迭代，预测器的计数器开始于“弱发生”，转换到“强发生”，再转换到“弱发生”，最后是“弱不发生”，因此它的预测是 T T T N；接下来是每组四个分支的循环，计数器状态变化为“强不发生”“弱不发生”“强不发生”“强不发生”，响应的预测为 N N N N 。因此一共有 249x3 + 2 = 749 次正确的预测。

分支 3:

500/1000 = 50%. 分支的模式是 T N T N，第一个分支计数器处于“弱发生”状态，转换为“强发生”，接下来是“弱发生”...，因此预测结果是 T T T T ... 所以预测器准确率为 50%。

注意：对于分支 3, 预测准确率强烈地依赖于分支预测器的初始状态。

(d) 当使用两层全局历史分支预测器（2bit 全局历史寄存器+每分支一张独立的模式历史表，模式历史表的每个表项是一个 2-bit 饱和计数器）时，三个分支的预测准确率分别是多少？假设全局历史寄存器每一位的初始状态都是‘不发生’，模式历史表中的 2-bit 饱和计数器初始状态都是‘强不发生’，计算预测准确率时，忽略前 500 次循环迭代。

参考答案：

分支 1:

499/500 = 99.8% 在 500 次循环迭代之后，预测器全局历史都进入“强发生”状态，它总是预测发生，只有最后一次循环迭代 (循环分支不发生) 的结果预测错误。

分支 2:

75%. 分析分支 2 和分支 3 的相关性很有帮助。全局分支历史包括分支 3 的最后一次结果以及总是发生的循环分支结果，分别为偶数次和奇数次循环迭代设立独立的饱和计数器很有效。当最后一个分支 3 的分支不发生，分支 2 的模式是 N T N T ... 因为永远不会有连续两个 T 出现所以饱和计数器会在强不发生和弱不发生之间振荡，导致全部预测都是不发生以及 50% 准确率。当最后一个分支 3 的分支发生，分支 2 的模式是 N N N N ... 将导致 100%的精度，因此总的准确率为 75%。

分支 3:

75%. 与上述情况类似，分支 2 的分支历史与分支 3 相关，并且分支 3 使用两个饱和计数器，由 IF 条件分支 2 的结果决定用哪一个。当分支 2 不发生，分支 3 的模式是 N T N N T N N T N ... (这些分支结果来自于第 1, 2, 3, 5, 6, 7, 9, 10, 11, ...次迭代)，预测器将会在弱不发生和强不发生之间振荡，但总是预测 N；当分支 2 发生，分支 3 同样总是发生，所以预测器会 100%准确。所以，每 4 次迭代只有一次预测错误，总预测精度 75%。

## 2 两层分支预测器 25 分

假设一个两层全局预测器由全局历史寄存器和所有分支共享的一张模式历史表组成(称为预测器 A)

1）我们把分支预测器中不同的分支映射到相同位置的情况称为"分支干扰"。预测器 A 的结构中，不同分支会在哪里发生这种干扰?

参考答案：

全局历史寄存器 (GHR), 模式历史表 (PHT)

2）另一个两层全局预测器由全局历史寄存器和每个分支一张模式历史表组成 (称为预测器 B"),

(a) 什么情况下预测器 A 的预测准确率低于预测器 B? 请解释理由，并举例说明。可以通过代码来说明。

参考答案：

当映射到同一个 PHT 条目的两个分支转向相反方向时预测器 A 的预测精度会低。考虑一个分支 B1，它对于给定的全局历史总是发生，如果 B1 有自己的 PHT，它永远能预测正确。现在，考虑另一个分支 B2，对于同样的历史总是不发生，如果 B2 有自己的 PHT，它也总是能预测正确。但是，如果 B1 和 B2 共享一个 PHT，它们映射到同一个 PHT 条目，因此互相影响并降低了彼此的预测精度。

考虑一种情况，一个 3 位的全局历史寄存器索引到 8 条目的模式历史表，当执行以下代码段:

```
for (i = 0; i < 1000; i ++)
{
if (i % 2 == 0) // IF 条件 1
{
.....
}
if (i % 3 == 0) // IF 条件 2
{
.....
}
}
```

对于全局历史"NTN"， IF 条件 1 总是发生，而 IF 条件 2 总是不发生，这将导致 PHT 的相互干扰。

(b) 预测器 A 能获得比预测器 B 更高的预测准确率吗? 请解释理由，并举例说明。可以通过代码来说明。

参考答案：

如果一个分支 B1 在有另一个分支 B2 共享同一个 PHT 条目时比独有自己的 PHT 预测结果更准确，就有这种可能。 考虑这种场景，分支 B1 对于给定的全局历史(当 B1 有自己的 PHT)总是预测错误，原因是这段历史中它总是在发生和不发生之间振荡，现在有一个总是发生的分支 B2 映射到相同的 PHT 条目，这将会改进 B1 的预测精度，因为这个时候分支 B1 会由于 B2 总是发生而总是被预测发生。如果 B2 在相同的历史中比 B1 执行的更频繁的话，这也可能不会降低 B2 的预测精度。因此，总的预测精度将会得到改善。

考虑一个 2 位全局历史寄存器和以下代码段：

```
if (cond1) { }
if (cond2) { }
if ((a % 4) == 0) {} //分支 1
if (cond1) { }
if (cond2) { }
if ((a % 2) == 0) {} //分支 2
```

分支 2 强烈相关于分支 1，因为当分支 1 发生时分支 2 总是发生，同时，两个分支有相同的历史。所以，分支 2 可以根据分支 1 的结果被准确预测。

(c) 分支干扰是否总是影响预测器的预测准确率? 请解释理由，并举例说明。可以通过代码来说明。

参考答案：

如果映射到同一个 PHT 条目的分支转向相同，那么预测器 A 和 B 会获得同样的预测精度。在这种情况下，分支之间的干扰不会影响预测精度。考虑两个分支 B1 和 B2 在某段固定的全局历史中总是发生，那么不管是有自己的 PHT 还是共享 PHT 条目，预测精度是相同的。

考虑这种情况，3 位的全局历史寄存器，检索到 8 条目模式历史表，执行以下代码段：

```
for (i = 0; i < 1000; i += 2) //循环分支
{
if (i % 2 == 0) //IF 条件
{
.....
}
}
```

循环分支和 IF 条件都总是发生，全局历史为"TTT"，所以两个分支映射到模式历史表的相同位置，两者的干扰也不会影响预测精度。

## 3 分支预测和推断 30 分

考察两台具有 15 段流水线的机器 A 和 B，流水段分布如下：

取指 (1 个阶段)

译码 (8 个阶段)

执行 (5 个阶段)

写回 (1 个阶段)

两台机器都会在发生流相关时采用数据转发。在译码的最后一个阶段检测是否有流相关，指令会在停顿在这个阶段等待检测结果。

机器 A 有一个预测准确率为 P%的分支预测器，分支方向和目标在执行的最后一个阶段产生。

机器 B 采用推断执行，类似咱们在课堂上讲的方式。

1）考察以下在机器 A 上执行的代码段:

```
Add   r3  r1, r2
sub    r5  r6, r7
beq    r3, r5, X
addi   r10  r1, 5
```

```
    add   r12  r7, r2
    add   r1  r11, r9
X:  addi  r15  r2, 10
    .....
```

我们把这段代码转化为在机器 B 上执行的推断代码，大概是下面这个样子:

```
    add      r3  r1, r2
    sub      r5  r6, r7
    cmp      r3, r5
    addi.ne  r10  r1, 5
    add.ne   r12  r7, r2
    add.ne   r14  r11, r9
    addi     r15  r2, 10
    .....
```

(假设条件结果由'cmp'指令计算，推断由'.ne'指令根据条件结果执行。条件结果在执行的最后一个阶段计算并且可以像其他值一样被转发)

这一段代码会重复执行上百次，分支 40%可能性发生，60%可能性不发生，平均而言，P 取什么样的值会使机器 A 比机器 B 具有更高的指令吞吐量?

参考答案：

这个问题显示了在分支预测机器上预测错误的惩罚和在推断执行机器上执行无用指令浪费的时钟周期之间的折衷。

我们在这里假设:

- 机器 A 和 B 具有分离的（流水）分支/比较和加法执行单元，即，在分支/比较指令停顿时可以执行加法指令
- 按序写回
- 当推断执行的指令被发现是无用的(在比较指令结果之后)，它仍然会继续剩余的流水段，相当于插入空操作。

对于不同的假设，这个问题的解答会不同，都是可能的正确答案。

在机器 A，当 beq r3, r5, X 分支不发生并且预测正确，执行的过程如下：

```
add r3 <- r1, r2    F|D1|D2|D3|D4|D5|D6|D7|D8|E1|E2|E3|E4|E5|WB|
sub r5 <- r6, r7       F|D1|D2|D3|D4|D5|D6|D7|D8|E1|E2|E3|E4|E5|WB|
beq r3, r5, X             F|D1|D2|D3|D4|D5|D6|D7|D8|- |- |- |- |E1|E2|E3|E4|E5|WB|
addi r10 <- r1, 5            F|D1|D2|D3|D4|D5|D6|D7|D8|E1|E2|E3|E4|E5|- |- |- |- |WB|
add r12 <- r7, r2               F|D1|D2|D3|D4|D5|D6|D7|D8|E1|E2|E3|E4|E5|- |- |- |- |WB|
add r1 <- r11, r9                  F|D1|D2|D3|D4|D5|D6|D7|D8|E1|E2|E3|E4|E5|- |- |- |- |WB|
X: addi r15 <- r2, 10                 F|D1|D2|D3|D4|D5|D6|D7|D8|E1|E2|E3|E4|E5|- |- |- |- |WB|
.....
```

当分支发生并且预测正确，执行过程如下:

```
add r3 <- r1, r2    F|D1|D2|D3|D4|D5|D6|D7|D8|E1|E2|E3|E4|E5|WB|
sub r5 <- r6, r7       F|D1|D2|D3|D4|D5|D6|D7|D8|E1|E2|E3|E4|E5|WB|
beq r3, r5, X             F|D1|D2|D3|D4|D5|D6|D7|D8|- |- |- |- |E1|E2|E3|E4|E5|WB|
X: addi r15 <- r2, 10        F|D1|D2|D3|D4|D5|D6|D7|D8|E1|E2|E3|E4|E5|- |- |- |- |WB|
.....
```

机器 A 因为分支预测错误(不管是发生还是未发生)受到 17 个时钟周期(8 个译码+5 个执行+4 个停顿)的惩罚。

机器 B 在分支不发生且预测正确时的执行跟机器 A 完全一样，但是，当分支发生(比较结果相等)时机器 B 浪费 3 个时钟周期，如下图所示。

```
add r3 <- r1, r2    F|D1|D2|D3|D4|D5|D6|D7|D8|E1|E2|E3|E4|E5|WB|
sub r5 <- r6, r7       F|D1|D2|D3|D4|D5|D6|D7|D8|E1|E2|E3|E4|E5|WB|
cmp r3, r5                F|D1|D2|D3|D4|D5|D6|D7|D8|- |- |- |- |E1|E2|E3|E4|E5|WB|
addi.ne r10 <- r1, 5         F|D1|D2|D3|D4|D5|D6|D7|D8|E1|E2|E3|E4|E5|- |- |- |- |WB|
add.ne r12 <- r7, r2            F|D1|D2|D3|D4|D5|D6|D7|D8|E1|E2|E3|E4|E5|- |- |- |- |WB|
add.ne r14 <- r11, r9              F|D1|D2|D3|D4|D5|D6|D7|D8|E1|E2|E3|E4|E5|- |- |- |- |WB|
addi r15 <- r2, 10                    F|D1|D2|D3|D4|D5|D6|D7|D8|E1|E2|E3|E4|E5|- |- |- |- |WB|
.....
```

所以，如果预测错误的代价低于执行无用指令浪费的周期，机器 A 比机器 B 的指令吞吐高。

(1- P) x 17 < 3 x 0.4

因此，P > 0.9294 时，机器 A 比机器 B 有更高的指令吞吐量。

2）考察在机器 A 上执行的另一段代码:

```
    add   r3  r1, r2
    sub   r5  r6, r7
    beq   r3, r5, X
    addi  r10  r1, 5
    add   r12  r10, r2
    add   r14  r12, r9
X: addi   r15  r14, 10
```

.....

我们把这段代码转化为在机器 B 上执行的推断代码，大概是下面这个样子:

```
add      r3  r1, r2
sub      r5  r6, r7
cmp      r3, r5
addi.ne  r10  r1, 5
add.ne   r12  r10, r2
add.ne   r14  r12, r9
addi     r15  r14, 10
.....
```

(假设条件结果由‘cmp’指令计算，推断由‘.ne’指令根据条件结果执行。条件结果在执行的最后一个阶段计算并且可以像其他值一样被转发)

这一段代码会重复执行上百次，分支 40%可能性发生，60%可能性不发生，平均而言，P 取什么样的值会使机器 A 比机器 B 具有更高的指令吞吐量?

参考答案：

在机器 A 上， 当 beq r3, r5, X 分支未发生且预测正确，执行过程如下：

```
add r3 <- r1, r2  F|D1|D2|D3|D4|D5|D6|D7|D8|E1|E2|E3|E4|E5|WB|
sub r5 <- r6, r7     F|D1|D2|D3|D4|D5|D6|D7|D8|E1|E2|E3|E4|E5|WB|
beq r3, r5, X           F|D1|D2|D3|D4|D5|D6|D7|D8|- |- |- |E1|E2|E3|E4|E5|WB|
addi r10 <- r1, 5          F|D1|D2|D3|D4|D5|D6|D7|D8|E1|E2|E3|E4|E5|- |- |- |WB|
add r12 <- r10, r2            F|D1|D2|D3|D4|D5|D6|D7|D8|- |- |- |- |E1|E2|E3|E4|E5|WB|
add r14 <- r12, r9               F|D1|D2|D3|D4|D5|D6|D7|D8|- |- |- |- |- |- |- |- |E1|E2|E3|E4|E5|WB|
X: addi r15 <- r14, 10              F|D1|D2|D3|D4|D5|D6|D7|D8|- |- |- |- |- |- |- |- |- |- |- |- |E1|E2|E3|E4|E5|WB|
```

当分支发生且预测正确，执行过程如下：

```
add r3 <- r1, r2  F|D1|D2|D3|D4|D5|D6|D7|D8|E1|E2|E3|E4|E5|WB|
sub r5 <- r6, r7     F|D1|D2|D3|D4|D5|D6|D7|D8|E1|E2|E3|E4|E5|WB|
cmp r3, r5              F|D1|D2|D3|D4|D5|D6|D7|D8|- |- |- |E1|E2|E3|E4|E5|WB|
addi r15 <- r14, 10        F|D1|D2|D3|D4|D5|D6|D7|D8|E1|E2|E3|E4|E5|- |- |- |WB|
.....
```

机器 A 因为分支预测错误(不管是发生还是未发生)受到 17 个时钟周期(8 个译码+5 个执行+4 个停顿)的惩罚。

机器 B 在分支不发生且预测正确时的执行跟机器 A 完全一样，但是，当分支发生(比较结果相等)时机器 B 浪费 11 个时钟周期，如下图所示。

```
add r3 <- r1, r2  F|D1|D2|D3|D4|D5|D6|D7|D8|E1|E2|E3|E4|E5|WB|
sub r5 <- r6, r7    F |D1|D2|D3|D4|D5|D6|D7|D8|E1|E2|E3|E4|E5|WB|
cmp r3, r5            F |D1|D2|D3|D4|D5|D6|D7|D8|- |- |- |E1|E2|E3|E4|E5|WB|
addi.ne r10 <- r1, 5      F |D1|D2|D3|D4|D5|D6|D7|D8|E1|E2|E3|E4|E5|- |- |- |WB|
add.ne r12 <- r10, r2        F |D1|D2|D3|D4|D5|D6|D7|D8|- |- |- |E1|E2|E3|E4|E5|WB|
add.ne r14 <- r12, r9           F |D1|D2|D3|D4|D5|D6|D7|- |- |- |- |D8|- |- |- |- |E1|E2|E3|E4|E5|WB|
addi r15 <- r14, 10               F |D1|D2|D3|D4|D5|D6|- |- |- |- |D7|- |- |- |- |D8|- |- |- |- |E1|E2|E3|E4|E5|WB|
```

所以，如果预测错误的代价低于执行无用指令浪费的周期，机器 A 比机器 B 的指令吞吐高。

$(1- P) \times 17 < 11 \times 0.4$

因此，$P > 0.7411$ 时，机器 A 比机器 B 有更高的指令吞吐量。