Perceptron-Based 分支预测 SimpleScalar 中的实现①

叶新栋 唐志强 涂时亮 (复旦大学 计算机学院 上海 200433)

摘 要: SimpleScalar 是目前国际上常用的一种超标量处理器的性能模拟器。首先分析了 SimpleScalar 模拟器的内部体系结构，并在此基础上深入剖析了其分支预测部件的实现机制。针对 SimpleScalar 模拟器分支预测部件只支持基于计数器预测器的局限性,通过深入研究 Perceptron-based 分支预测器的实现机制,提出并设计了如何在 SimpleScalar 模拟器中实现 Perceptron-based 分支预测器的方案。对超标量处理器的性能模拟和研究有着实际的意义。

关键词: 超标量处理器 模拟器 分支预测 乱序执行

Realization of Perceptron-Based Branch Predictors in SimpleScalar

YE Xin-Dong, TANG Zhi-Qiang, TU Shi-Liang

(Computer College, Fudan University, Shanghai 200433, China)

Abstract: SimpleScalar is a superscalar processor monitor which is used widely in performance analysis. This paper first analyzes the internal architecture of SimpleScalar, and on this basis, in-depth analyzes the mechanism to achieve branch prediction. Through an in-depth study of Perceptron-based branch predictor implementation mechanism, it proposes a design to let SimpleScalar support Perceptron-based branch predictor. And this has practical significance for superscalar processor performance modeling and research.

Keywords: SuperScalar; monitor; branch predictor; out-of-order

现代超标量体系结构处理器的设计研究是一项耗时漫长、复杂度极高的科研工作。科研人员往往需要借助于相关的模拟器对相应的体系结构进行性能上的模拟和方针，力求在硬件实现前评测出各种方案的性能及功耗，找出设计瓶颈并加以改进。Wisconsin-Madison 大学发布的 SimpleScalar 模拟器是一款源代码公开并且具有良好可扩展性的超标量体系结构的模拟器。SimpleScalar 既提供了简单的功能模拟器，也提供了模拟超标量处理器微体系结构的乱序性能模拟器，其乱序模拟器支持动态指令调度、指令乱序执行、指令预测执行、分支预测等现代微处理器的特性，而且还提供了一系列的工具，包括编译器、Benchmark、调试工具、流水线跟踪器等，为计算机体系结构的研究提供了全面的支持[1]。

分支预测器在超标量流水线的体系结构中起着极其重要的作用。SimpleScalar 的分支预测部件的设计具有极强的扩展性，目前支持：静态分支预测、动态两级分支预测模式、简单的两位直接映射预测模式以及两级和位预测联合的预测等众多模式。Perceptron-based 分支预测器是目前国内外科研的热点,因而利用 SimpleScalar 的分支预测部件的可扩展性为 Perceptron-based 分支预测器设计一种

① 收稿时间:2009-04-06

简单实用的模拟仿真环境具有实际的意义。

1　SimpleScalar体系结构简析

SimpleScalar 是一个易于修改扩充，性能效率较高的模拟器，因此在世界范围内被学术界广为采用。该模拟器模拟的是一个超标量，5 级流水的 RISC 体系结构的 CPU 模型，提供了从最简单到超标量乱序发射的不同的模拟程序。图 1 显示了 SimpleScalar 模拟的虚拟超标量处理器的结构框图。

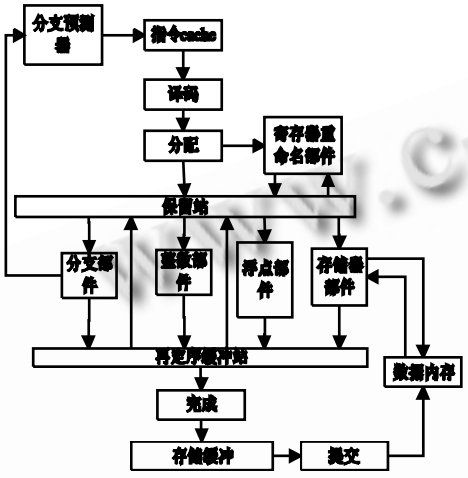

图 1　虚拟超标量处理器的结构框图

SimpleScalar 采用了分层模块化的组成结构。其自底向上依次可以分为：性能模块、功能模块、I/O 接口模块以及用户程序模块。其中性能模块是其模拟的核心部件，其主要包括了指令模拟器、存储体系模拟器、资源模拟器、分支模拟器以及流水线模拟器。其中占核心地位的是超标量乱序发射的流水线模拟器。

在 SimpleScalar 流水线模拟器中依次包括取指、分配、发射及执行、写回和提交等几个过程(如图 2 所示)。其各个阶段的功能分别如下所述：取指阶段首先通过分支预测器得出取址地址，并将取出的指令缓存在指令缓存队列 IFQ 中；分配阶段首先对指令进行译码并把指令放入保留站(RUU 和 LSQ)；发射及执行阶段将那些准备就绪的指令发射到相应的功能部件执行，其中 RQ 代表的是包括已经就绪的指令队列；写回阶段将执行阶段得到的结果通过 Tomasulo 机制广播到各个部件，其中包括了对分支预测器的验证工作，EQ 队列记录已发射指令何时执行完毕；提交过程将结果最终写回数据 cache 并更新相应的寄存器。

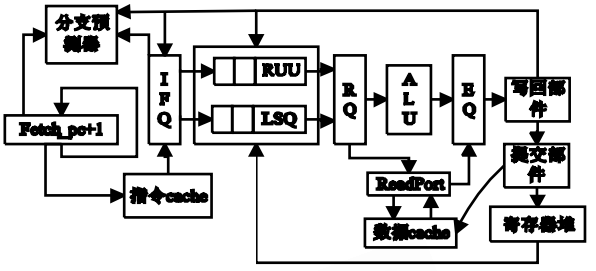

图 2　流水线模拟器框图[1]

2　基于计数器的分支预测方案

增加指令流通路的流量和增大持续取指的宽度在目的上是一致的，为此，必须减少取指段的停顿周期数。鉴于超标量处理器中的开销等于停顿周期数和流水线宽度的乘积，指令流技术主要目标是减少取指停顿或者利用这些停顿周期进行有用的工作。当前的主流技术是分支预测技术，利用动态分支预测技术可以有效的解决控制相关问题，极大地提高处理器的执行效率。

分支预测主要包括三大部分，即分支地址的推测、分支条件的推测和分支结果的验证。接下来我们以 SimpleScalar 的分支预测部件为基础就这三部分做详细的论述。

分支目标地址的推测主要是依靠设置分支目标缓冲 BTB 来实现。BTB 用以保存前几次分支执行的目标地址，它是一个较小的 cache 存储器并在取指时段通过 PC 访问。BTB 的每条记录包括两部分：分支指令地址和分支目标地址。BTB 的结构如图 3 所示。

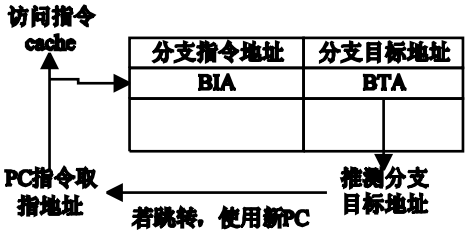

图 3　分支目标缓冲器框图

分支条件的推测目前绝大多数基于 Smith 于 1980 年提出的两位饱和计数器 FSM[2]。如图 4 所示，两位饱和计数器方案实际上是用一个有限状态机模型去预测分支发生的趋势。显而易见，当计数器的值大于或者等于 2 时预测分支发生，否则预测分支不发生。

当分支结果最终确定后，需要根据状态机进行相关的状态切换。

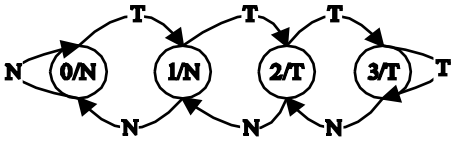

图 4 两位饱和计数器示意图

考虑到程序的实际运行过程中不仅仅与自身的历史信息相关联，同时与分支执行的上下文也有密切的关联。因而，现代超标量处理器的分支预测部件绝大多数采用了 Yeh 和 Patt 提出的两级自适应分支预测算法及其相关的变形算法如 (Scott McFarling 于 1993 年提出的关联分支预测器 gshare)[3]。两级自适应分支预测算法与简单的两位直接映射预测算法相比最大的改进之处在于其提供了分支历史信息移位寄存器(BHSR)用于保存和更新相应的分支信息。因此，在实际的预测中，首先会根据相关信息得到 BHSR 的值，然后综合 BHSR 和分支地址的信息并以此为索引获取模式历史信息表(PHT)中的两位饱和计数器的值，并利用图 4 所示的状态机得出预测信息。在分支验证阶段，需要根据实际分支跳转信息更新 PHT 和 BHSR 中的信息。

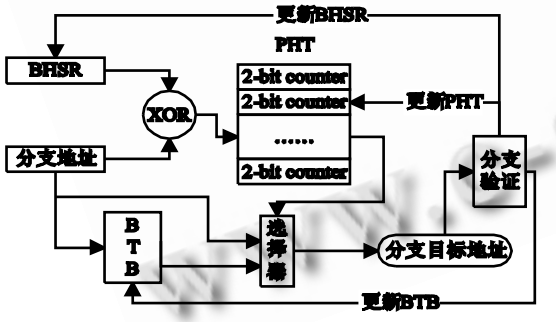

图 5 gshare 算法在模拟器中的实现框图

SimpleScalar 的分支预测部件具有极强的可扩展性。目前支持的动态预测算法都是基于两位计数器有限状态机模型，其中包括：可配置的直接映射预测算法、可配置的两级自适应算法、gshare 关联分支预测算法以及联合分支预测算法等。图 5 显示了在 SimpleScalar 中利用 gshare 关联分支预测算法实现分支预测的整个流程，需要注意的是实际运行中分支地址的推测、分支条件的推测和分支结果的验证这三个过程是在流水线的不同阶段完成的，图 5 中简化了这一过程。

3 Perceptron-Based算法理论基础

Perceptron-based 分支预测算法由 D.A.Jimenez 和 C.Lin 于 2001 年首度提出[4]。该算法的更高精确度是通过利用更长的分支历史信息来获取，同时该算法还克服了基于计数器的预测器硬件复杂度和历史信息长度呈指数性相关这一弊端，在 Perceptron-based 分支预测算法中硬件复杂度和历史信息长度仅存在线性相关性。

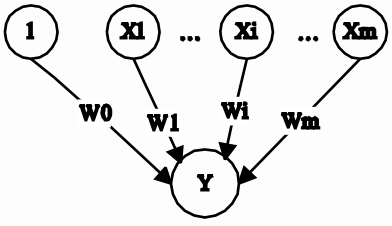

图 6 感知器示意图

如图 6 所示，1 和 X 向量为输入部分，X 向量中的值只能取 1 或 −1，分别代表该历史分支是否发生转移。同时每个输入值都与一个权重值相关联，记为 W 向量。Y 表示输出量，其值的正负关系分别代表预测分支是否发生。Y 的计算公式如下所示。

$$Y = W_0 + \sum_{i=1}^{m} X_i W_i$$

同基于计数器的预测器一样，Perceptron-based 分支预测算法在分支方向最终确定后，也必须对相关的记录信息进行更新操作。假设 t 为实际分支的输出，为门限值，其表示该感知器是否需要进一步更新操作[5]。

if sign(Y) ≠ t or $|Y| \le \theta$ then
  for i in 0...m do
    $Wi := Wi + tXi$;
  end for
end if

通过上述代码片段，我们可得仅当分支预测发生错误或者输出值不超过门限值的情况下才需要进行权值的更新操作。

4　Perceptron-based 算法的模拟实现

在这一部分，我们将以 SimpleScalar 中分支部件的数据结构为基础，将 Jimenez 和 Lin[6]提出的 Perceptron-based 分支预测算法扩充进 SimpleScalar 的分支预测部件中。图 7 显示的是 Perceptron-based 分支预测算法在 SimpleScalar 中的实现框架。

SimpleScalar 中分支部件的程序采用了 swich 语句来选择实现不同的预测器功能，因而，我们只需要在 swich 结构中加入相应的 case 语句便可以选择实现我们所需的 Perceptron-based 分支预测器。

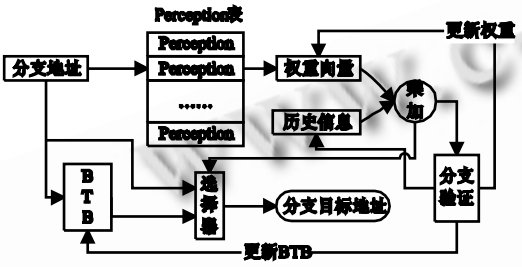

图 7　Perceptron-based 算法在 SimpleScalar 中的实现框图

SimpleScalar 源代码结构中涉及分支预测的文件只有 Bpred.h、Bpred.c 和 sim-outorder.c 文件。其中 Bpred.h 记录了分支部件的数据结构，Bpred.c 实现了分支部件的所有功能，而 sim-outorder.c 流水线模拟器文件只负责调用 Bpred.c 实现的相关功能函数。其中 Bpred.c 文件中的 bpred_lookup 和 bpred_update 函数实现了分支地址的推测、分支条件的推测和分支结果的验证等一系列功能。为了让模拟器能正常的工作，在流水线开始执行之前我们必须建立相应的分支预测器。建立分支预测器的函数名为 bpred_create，针对 Perceptron-based 分支预测器，我们只需要在 switch 语句中增加相应的功能代码以完成申请并初始化 Perceptron 表、权重向量及分支历史信息向量的工作便可。

接下来我们将详细论述如何在 SimpleScalar 中利用程序来实现 Perceptron-based 分支预测算法，以完成分支地址的推测、分支条件的推测和分支结果的验证等一系列功能。

在流水线的取指段，SimpleScalar 会进行分支地址和分支条件的推测，以得出下条指令的地址。其程序流程如下所示(序号代表执行的不同步骤)：A0. 解析分支指令，查看指令是否为无条件转移 jmp 指令。A1. 将分支指令地址左移相应位并与 Perceptron 表的大小值取余操作。以该值为索引获取 Perceptron 表相应表项的值。A2. 将获取的权重向量与分支历史信息向量做乘加操作，查看结果以决定是否跳转。A3. 置跳转发生标志并以指令地址为索引到 BTB 中取出分支跳转地址。A4. 置分支不跳转标志并返回该信息。A5. 分支地址预测及条件预测完毕，返回相关数值。

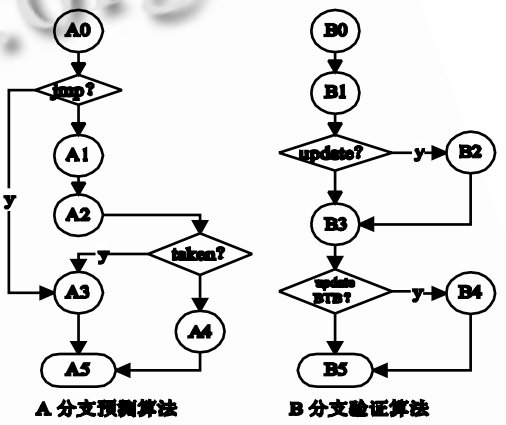

图 8　Perceptron-based 分支预测算法流程图

分支的验证和更新操作根据指令的不同，其所发生在的流水线部件也不尽相同。其程序流程如下所示：B0.根据预测结果更新相关的统计信息。B1.根据预测的结果及门限值[1.93m+14]判断是否需要更新相应的权重向量。B2.对相应的权重向量进行更新操作。遍历该分支历史信息向量，若某一元素与分支实际输出信息相等则相应权重向量中的元素值加 1，否则进行减 1 操作。B3.更新分支历史信息向量。B4.若预测有误或该分支指令不在当前 BTB 中，则执行更新 BTB 操作。B5.分支验证及更新操作完成。

5　总结

Perceptron-based 分支预测算法是目前超标量体系结构中分支预测器研究的热点和难点。通过在超标量模拟器 SimpleScalar 分支预测部件中扩充相关的部件，可以有效地模拟基于感知器分支预测算法的性能。对现代超标量处理的科研工作具有实际的意义。

(下转第 94 页)

(上接第 54 页)

参考文献

1 陈剑龙,傅忠传,崔刚. SimpleScalar 模拟器内核分析及应用.哈尔滨工业大学学报, 2004,36 (5):652 – 653.

2 Smith JE. A study of branch prediction strategies. Proc. of the 8th Annual International Symposium on Computer Architecture. 1981.135 – 148.

3 Yeh TY, Patt YN. A comparison of dynamic branch predictors that use two levels of branch history. Proc. of the 20th Annual International Symposium on Computer Architecture. 1993.257 – 266.

4 Jimenez DA, Lin C. Neural methods for dynamic branch prediction. ACM Transactions on Computer Systems, 2002,20(4):369 – 397.

5 Ho CY, Chng KF, Yau CH, Anthony SS. Fong. A study of dynamic branch predictors: Counter versus perceptron. Proc. of the International Conference on Information Technology. 2007.528 – 563.

6 Jimenez DA, Lin C. Dynamic branch prediction with perceptron. Proc. of the 7th Int'l Symposium on High Performance Computer Architecture. 2001.197 – 206.