哈尔滨工业大学

<<计算机体系结构>>

实验报告

(2021 年度秋春季学期)

姓名：	胡聪
学号：	180110505
学院：	计算机科学与技术学院
教师：	董剑/汪东升/刘川意

（1）借助 Pin 的 API 文档，阅读 ManualExamples 目录下 inscount0、inscout1 和 inscout2 工具的源码，分析插桩工具的代码框架和执行过程；

instruction 是最低层次的，指的是汇编命令，如 mov、inc、loop 等，pintools 包含两个插件。桩：决定在哪里插入什么代码分析代码：插入点执行的代码 pintool 在 pin 中注册一些桩回调函数，每当 pin 生成新的代码时调用回调函数。

综迹插桩 TRACE_AddInstrumentFunction

指令插桩 INS_AddInstrumentFunction

镜像插桩 IMG_AddInstrumentFunction

函数插桩 RTN_AddInstrumentFunction

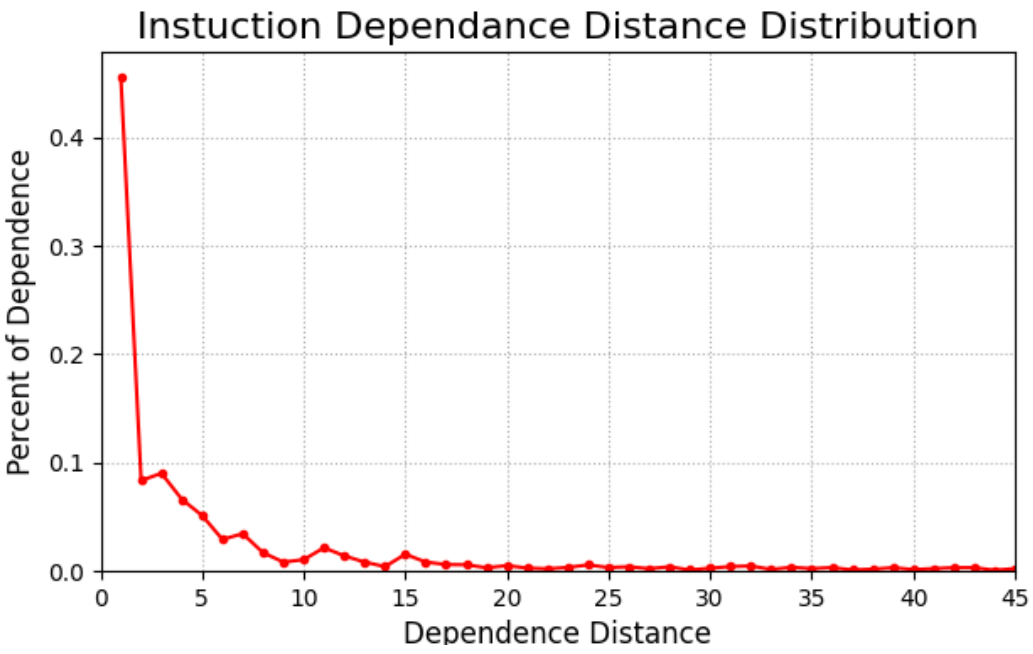
下面开始对于 inscount0、inscount1、inscount2 工具的源码分析。

inscount0：对指令进行插桩，分析代码的方式是把指令数加一，初始化，然后调用指令插桩函数，程序执行前，每条指令都会被加入该函数中，然后指令会被在之前计入回调函数，处理函数成功后启动程序；

inscount1：对基本块进行插桩，按照基本块统计指令的总数，首先进行初始化，调用 INS_AddInstrumentFunction 设置一个插桩函数，在基本块被调用之前，都会被该函数所处理，处理完毕后启动程序；

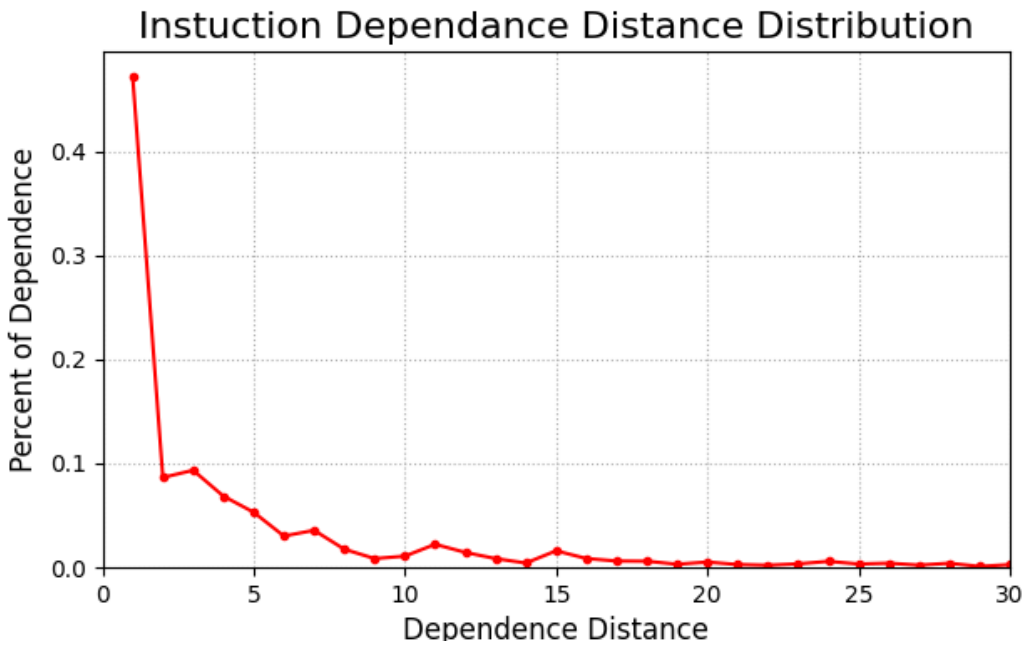
inscount2：在综合 inscount1 中的综迹插桩后，在回调函数中加入了参数，可以更快速调用回调函数。

（2）列出必做部分实验结果的指令依赖距离分析统计图/分布图，并对分布图进行必要的分析讨论。注意，需要对 pi 和 memtester 两个测试程序进行插桩测试和统计；



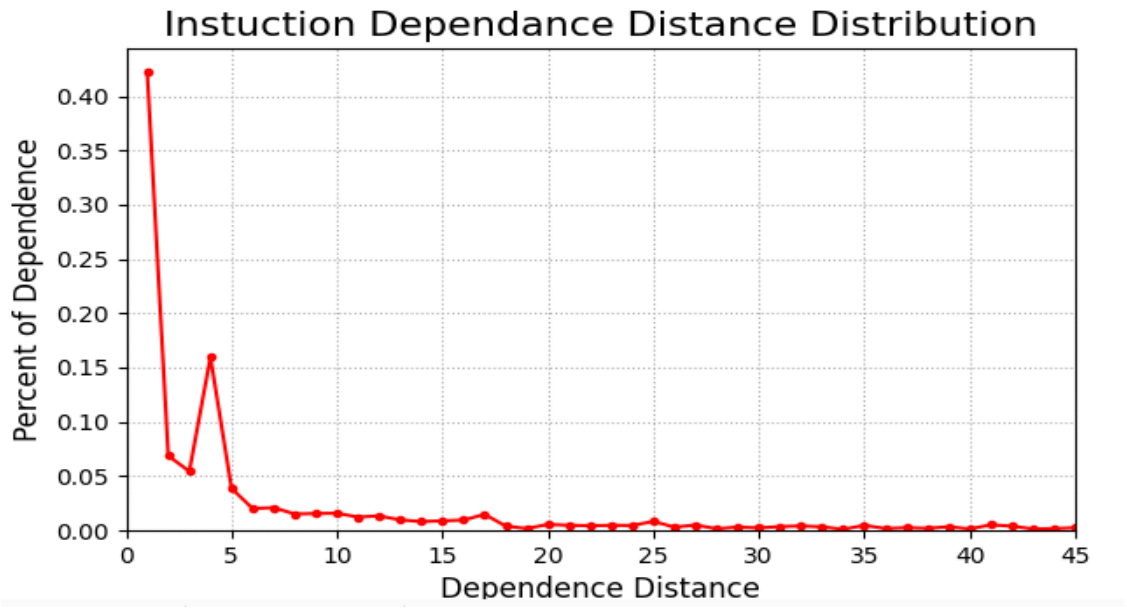
ls 命令的依赖距离分析

可以看到，ls 的大部分指令的依赖距离在 5 以内，说明该指令对于寄存器的调用大多是短期近距离使用。



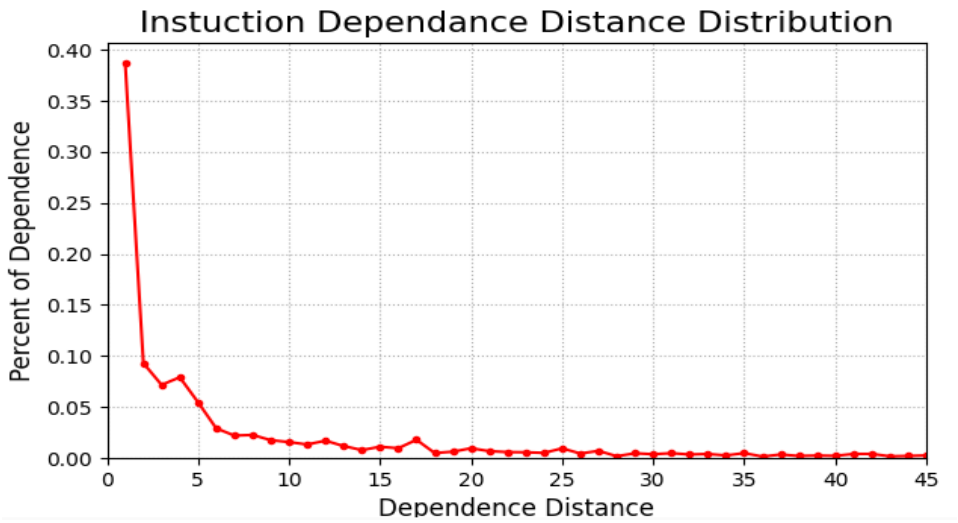
pwd 命令的依赖距离分析

可以看到，pwd 的大部分指令的依赖距离在 5 以内，说明该指令对于寄存器的调用大多是短期近距离使用。



pi 命令的依赖距离分析

可以看到，大部分指令的依赖距离在 5 以内，说明使用的是多次通用寄存器进行 pi 的计算。



memtester 命令的依赖距离分析

大部分指令的依赖距离在 5 以内，使用多次通用寄存器进行的内存分析。

（3）思考并回答以下问题，将答案以及分析过程写入实验报告。

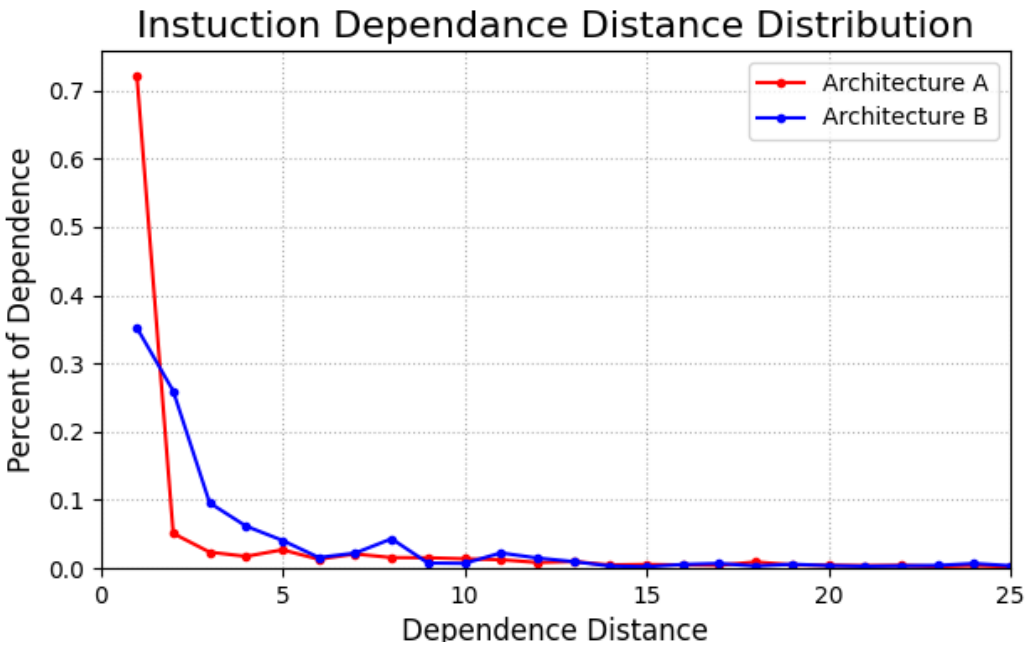
a. 可以发现，指令依赖距离分布图存在很长的尾巴。试分析尾巴可能是由哪些寄存器引起的？为什么？

可能是数据寄存器引起，计算结果没有使用，但是需要进行计算，这样会导致运算频率低、依赖距离长的情况，所以尾巴较长；

也可能是由于段寄存器引起，段寄存器访问内存，但是由于段寄存器内的内容很少改变，导致依赖距离长。

b. 设有 2 个不同架构的处理器平台，现分别于其上运行由相同版本编译器所编译的测试程序，并分别在这两个平台上使用 insDependDist 工具对该测试程序进行插桩分析，得到如图所示的指令依赖距离分布图。请问架构 A、B 之中谁具有更多的寄存器？为什么？

B 有着更多的寄存器，A 的指令依赖距离低于 5 的可能性相对于 B 来说高很多，这是因为寄存器的数量原因，所以此时选择依赖距离较高的寄存器，所以 A 的计算器数据变动较快，依赖距离低。



2 种架构下，同一程序的指令依赖距离分布图

c.现有基于相同 ISA(Instruction Set Architecture)设计的架构 A 和架构 B。若架构 A 采用停顿法解决流水线数据冲突，架构 B 则采用数据转发法，当二者执行相同的测试程序时，它们的指令依赖距离分布图是否相同？为什么？

不同，A 不改变相对位置关系，B 改变指令的相对位置关系，所以 A 和 B 的分布图不会相同。