实验总结

这次高级操作系统课程我选择的实验是 commuter。虽然这篇文章的很多形式化的证明我并没有看得很懂，文章中介绍的可交换规则的大体含义和核心思想能够理解。

一、 实验背景

实验主要基于文章：The Scalable Commutativity Rule: Designing Scalable Software for Multicore Processors

当前常见的评估多核软件可扩展性的方法是 :在软件上加上工作负载, differential profile, fix bottleneck, plot scalability 三个步骤循环往复进行。存在的问题有：新的工作负载会引发新的瓶颈；随着核数的增多，更多的瓶颈会暴露；然而真正的瓶颈可能在于接口设计上。作者顺势想到 :是否可以在任何实现存在之前，仅依靠接口规范就可以判断确定是否存在可扩展的机会。本文的主要内容就是解决这一问题, 作者首先在理论上提出了 :可扩展交换规则，无论何时接口操作可交换，这些操作一定可以以某种方式实现较好的扩展性。

这篇文章从理论上提出了可扩展的交换性规则，并且对这一规则做了形式化的证明，这一规则是状态依赖的，基于接口的。作者还将这一理论实现为了一个工具：commuter，这是一个自动的可扩展的测试工具，在 commuter 的指导下，对 sv6 操作系统进行了改进，开发了一个可扩展的文件和虚拟内存系统。

这一交换规则背后的直观理论是 :接口操作可交换，说明他们的执行顺序与最终的结果无关，操作之间的通信时不必要的，没有通信的话，也就是说这 2 个操作之间没有冲突。

Commuter 这个工具的主要组成部分及工作原理：

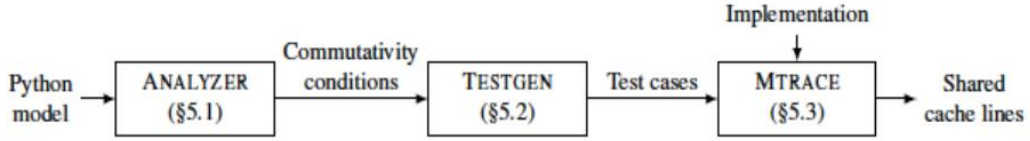


Figure 3: The components of COMMUTER.

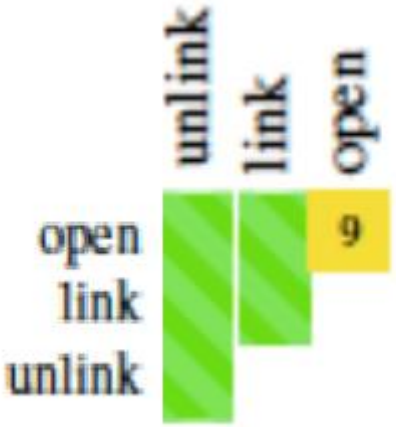
详细介绍见文章。

二、 实验内容

复现文章中第 6 部分的实验，作者已经开发好了一个关于 POSIX 文件系统和虚拟内存 API 的简化的模型，用来作为 commuter 的输入。这个模型覆盖了 18 个系统调用。鉴于文章中的实验在 8 核 10 线程, 256G 内存的机器上运行 8 分钟多, 本次复现只对其中的 open,link 和 unlink 三组系统调用进行实验。实验环境是主流台式机上的 vmware。

首先安装 commuter，参照 github 上的安装指令，逐步编译构建。运行 SV6。遇到的问题有 GCC 版本问题, 降级 GCC 版本到 4.7;在安装 sv6 的过程中遇到的问题有一个 page fault 的错误，不会解决，sv6 并没有运行起来。

对 SV6 的接口测试实验结果如下：



除此之外，想把 commuter 这个工具真正用在网络相关的研究上，我发现早在 2009 年就有 para-snort 的工作，文章题为：PARA-SNORT : A MULTI-THREAD SNORT ON MULTI-CORE IAPLATFORM。有了 commuter 之后，可以对这些已有的 snort 扩展的工作进行评估，在可交换原则的指导下对开发作出指导和改进。这部分工作需要了解已有代码，并对其作出 python 建模。

三、 实验结论

Commuter 的工作意义在于可以在实现之前就对软件的多核可扩展性作出评价，未雨绸缪, 只需要写出接口的规范, 对其进行 python 的建模就可以完成对软件扩展性实验的验证。经过实验，发现 commuter 简单易用，运行开销可接受。相比于文中进行的 linux 的接口可扩展性验证，可以发现在可交换原则下进行的修改可以提高软件的多核扩展性。