Commuter

这篇主要讨论了为多核处理器设计可扩展的软件的方法。并介绍了COMMUTER工具。

作者首先提出了一项规则：如果接口操作是可交换的，那么它们就可以以可扩展的方式实现。这项规则使得开发者可以从接口设计阶段设计可扩展的程序。

传统的方法评估多核系统软件的可扩展性，是选择一定的负载，画出性能随着核数的变化，使用工具来分析可扩展性瓶颈。这有一些缺点，不同的负载在较大的核数量时会产生新的瓶颈，哪一个瓶颈时根本性的并不明确。

本文认为，如果接口实现的内存访问没有冲突，那么可以说这些操作是可扩展的。因为当一个核写cache时，没有其他的核读或写这个cache。当内存访问没有冲突是，增加更多的核会使性能线性增加。

可交换规则有着直觉性的认识：当操作可交换时，它们的结果是不随操作顺序而改变的。 在可交换操作之间的通信是不必要的，消除这些通信将会产生不冲突的实现。

可交换原则是很严格的，几乎不能用在复杂的，状态多的系统软件上。作者提出的SIM交换性原则，则是依赖状态的基于接口的并且单调的原则。这使得实现可扩展性软件有更大的空间。

可交换性原则导致了新的设计方法来设计可扩展软件，1分析接口的可交换性，2在可交换情况下设计可扩展的实现。复杂的接口会导致分析和发现可交换情况变得更难。因此，作者介绍了成为COMMUTER的工具来辅助推理。

开发者通常按照一种迭代的方式来提高可扩展性：设计，实现，测试，重复。这种方法已经是的Linux具有较好的扩展性，能够适应各种负载情况。然而，Linux系统仍然具有可扩展性瓶颈，并且缺少工具来分析接口层次的可扩展性，至于那些瓶颈是系统调用接口导致的好不明确。多核处理器的多内核通过避免在内核中共享数据结构来增强可扩展性。

测试用例的产生。COMMUTER工具使用符号和共鸣执行产生测试的组合。COMMUTER的测试用例产生针对完成冲突的覆盖。

作者继续提出了可扩展的可交换原则。由于可交换原则不是单调的，因此应用起来不方便。作者进一步提出了SIM可交换原则。SIM可交换原则具有单调性，基于接口的并且基于状态的，实用性很好。

对实现的建模。S是状态集合，I是调用集合，R是响应集合。则实现是SI到SR的映射。

设计可交换的接口。可交换原则将扩展性推理放在了接口层次上，并且SIM可交换原则使得我们可以将规则应用在复杂接口上。1分解组合操作。很多POSIX的API将集合操作结合成一个，限制了组合操作的可交换性。2接受规则的不确定性。POSIX的最低可获得FD规则造成了较差的可扩展性。因为这个规则，同一个进程的相同的打开操作不可交换，因为执行的顺序决定了返回的FD。3允许弱要求。这是限制可交换性的另一点。4不同步的释放资源。