**[Web 站点的水平扩展和垂直扩展](http://www.cnblogs.com/wisdo/p/4515049.html)**

当一个开发人员提升计算机系统负荷时，通常会考虑两种方式垂直扩展和水平扩展。选 用哪种策略主要依赖于要解决的问题以及系统资源的限制。在这篇文章中我们将讲述这两种策略并讨论每种策越的优缺点。如果你已经有一个软件系统需要不断成 长，那么你将有意或者无意中选择这两种策略中的一种。

举个例子

例如，假设你现在负责一批木材采伐的操作。

在这个例子中，我们假设有3辆卡车，每辆车一次可以运25根木材，计算花费1小时的情况下可以运送到指定地点等待处理的木材数量。通过这些数字我们可以算出我们系统最大的负荷量：

3辆卡车 \* 25根木材 \* 1小时=75根木材／小时

**垂直扩展**

在垂直扩展模型中，想要增加系统负荷就意味着要在系统现有的部件上下工夫，即通过提高系统部件的能力来实现。

如果我们选择垂直扩展模型，那么我们将怎么做来使我们每小时可以处理150根木材？我们需要至少做以下两件事中的一件：

使每辆卡车的运输量增加一倍（50棵树每小时），或者使每辆卡车的运输时间减半（每辆卡车30分钟）。

3辆卡车 \* **50棵树** \* 1小时 = 150棵树／每小时

或者

3辆卡车 \* 25棵树 \* **30分钟** = 150棵树／每小时

我们没有增加系统的成员数，但是**我们通过增加系统成员的生产效率来获得期望的负荷量**。

**水平扩展**

在水平扩展模型中，我们不是通过增加单个系统成员的负荷而是简单的通过增加更多的系统成员来实现。也就是说，在以上运送木材的例子中，通过增加卡车的数量来运送木材。因此，当我们需要将负荷从75棵树每小时增加到150棵树每小时，**那么只需要增加3辆卡车。**

6辆卡车 \* 25棵树 \* 1小时 = 150棵树／每小时

在这个例子中，系统每个成员的生产力依然没变，我们通过增加更多的卡车来提高系统的能力。

**扩展你的web数据库**

通过对水平扩展和垂直扩展的基本了解，下面让我们来关注web系统的扩展。**一个网站通常有很多组件都需要去考虑它们的扩展性，**但是我通常喜欢关注处在最边缘的一个：数据库。为什么数据库是最边缘的？因为数据库通常是共享资源，是几乎所有请求最终的连接点。

**你的系统是什么类型的？**

在扩展你的数据库时，你必须要问的一个重要问题是：“我所面对的系统是什么类型的？”你所面对的是一个读操作多还是写操作多的系统？读操作多的网站 一般包括：在线商城，在商城里用户大部时间是在浏览(读操作)，只有少数时间在付款(写操作)、或者博客，在博客上人们大部分时间是在浏览博文(读操 作)，只有少数时间是在评论或者发表博文(写操作)。相反的，关于写操作非常多的很好的例子包括：信用卡交易处理器，这个系统的主要负载时在处理记录交易 (写操作)，偶尔会查找交易(读操作)、或者Google分析，主要工作实在记录业务数据(写操作)，偶尔会展示分析图(读操作)。

了解你所创建的网站是什么类型的，可以在网站成长过程中帮助你选择正确的技术。

**读操作扩展（网站）**

如果你的系统读操作非常多，那么通过关系型数据库如mysql或者PostgreSql来**垂直扩展数据存储是一个不错的选择**。结合你的关系型数据库 通过使用memcached或者CDN来**构建一个健壮的缓存系统，那么你的系统将非常容易扩展。在这种模式中，如果数据库超负荷运行，那么将更多的数据放入缓存中来缓解系统的读压力。**当没有更多的数据往缓存中放时，可以更换更快的数据存储硬件或者买更多核的处理器来获取更多的运行通道。摩尔定律使通过这种 方法来垂直扩展变得和购买更好的硬件一样简单。

**写操作扩展（银行卡）**

如果你的系统写操作非常多，那么**你可能更希望考虑使用可水平扩展的数据存储方式**，比如Riak，Cassandra或者HBase。和大多数关系型 数据管理系统不同，这种数据存储随着增长增加更多的节点。由于你的系统大部分时间是在写入，所以缓存曾并不能像在读操作比较频繁的系统中起到那么大作用。 很多写频繁的系统一开始使用垂直扩展的方式，但是很快发现并不能根本解决问题。为什么？因为硬盘数和处理器数在某一点达到平衡，在这个边界上再增加一个处 理器或者一个硬盘都会是每秒钟的I/O操作数成指数性增长。相反，如果对写频繁的系统采取水平扩展策略，那么你将达到一个拐点，在这个拐点之后如果在增加 一个节点都远比使用更多的硬盘来的实惠。

**其他注意事项**

另一件事需要记住的是每种扩展策略下预想不到的开销。采用垂直扩展的系统将开销凡在单独的组件上。当我们去提升系统负荷时，这些单独的组件需要在管 理上花费更多。拿我们运送木材的例子来说，如果需要使每辆卡车的货运量翻倍，那么我们需要更宽、更长、或者更高的车厢。也许有的路因为桥的高度对车辆高度 有要求，或者基于巷子宽度车宽不能太大，又或者由于机动车安全驾驶要求车厢不能太长。这里的限制就是对单个卡车做垂直扩展做的什么程度。同样的概念延伸到 **服务器垂直扩展：更多的处理器要求更多的空间，进而要求更多的服务器存储架。（就是加大处理器的架构）**

相反的，采用水平扩展的系统将额外的开销放在**系统中连接起来的共享组件上**。当我们去提升系统负荷时，共享的开销和新增加的成员之间的协调性有关。在 我们运送木材的例子中，当我们在路上增加更多卡车时，那么路就是共享资源也就成了约束条件。这条路上适合同时跑多少量卡车？我们是否有足够的安全缓冲区使 得所有的车可以同时装运木材？如果再来看我们水平扩展的数据库系统，那么经常被**忽略的开销就是服务器同时连接时的网络开销**(译者注：网络为各个系统的共享 资源)。当**你为系统增加更多的节点时，共享资源的负荷也就越来越重（就是增加服务器的个数，越多个服务器，网络开销越大）**，通常呈非线性改变。

**综合说明**

和计算机的大多数东西一样，好的解决办法通常并不像我这里列出来的这么简单。而我在这里尝试简化这种思想用来来说明这中概念而不是讲具体的解决办 法。**扩展是个困难的问题，**这是个需要在实际处理的每个步骤中都要思考的问题。扩展策略没有魔法，也没有魔法般的软件帮你建立一个完整可靠的可扩展系统。就 像扩展中的其他问题一样，一个大的解决方案通常是由很多个一起协调工作的小的解决办法组成的。这需对每一个中解决方案进行精心正确的设计和开发。