网站性能优化对于大型网站来说非常重要，一个网站的访问打开速度影响着用户体验度，网站访问速度慢会造成高跳出率，小网站很好解决，那对于大型网站由于栏目多，图片和图像都比较庞大，那该怎么进行整体性能优化呢？本文为你提供一份大型php网站性能和并发访问优化方案.

php.cn首页性能优化图

一、大型网站性能提高策略：

大型网站，比如门户网站，在面对大量用户访问、高并发请求方面，基本的解决方案集中在这样几个环节：使用高性能的服务器、高性能的数据库、高效率的编程语言、还有高性能的Web容器。这几个解决思路在一定程度上意味着更大的投入。

web容器是一种服务程序，在服务器一个端口就有一个提供相应服务的程序，而这个程序就是处理从客户端发出的请求，如JAVA中的Tomcat容器，ASP的IIS或PWS都是这样的容器。一个服务器可以多个容器。

1、HTML静态化

其实大家都知道，效率最高、消耗最小的就是纯静态化的html页面，所以我们尽可能使我们的网站上的页面采用静态页面来实现，这个最简单的方法其实也是最有效的方法。

2、图片服务器分离

大家知道，对于Web服务器来说，不管是Apache、IIS还是其他容器，图片是最消耗资源的，于是我们有必要将图片与页面进行分离，这是基本上大型网站都会采用的策略，他们都有独立的、甚至很多台的图片服务器。这样的架构可以降低提供页面访问请求的服务器系统压力，并且可以保证系统不会因为图片问题而崩溃。

在应用服务器和图片服务器上，可以进行不同的配置优化，比如apache在配置ContentType的时候可以尽量少支持、尽可能少的LoadModule，保证更高的系统消耗和执行效率。

3、数据库集群、库表散列

大型网站都有复杂的应用，这些应用必须使用数据库，那么在面对大量访问的时候，数据库的瓶颈很快就能显现出来，这时一台数据库将很快无法满足应用，于是我们需要使用数据库集群或者库表散列。

在数据库集群方面，很多数据库都有自己的解决方案，常用的MySQL提供的Master/Slave也是类似的方案。

集群通常会使用CDN与GSBL与DNS负载均衡技术，每个地区一组前台服务器群，例如：网易，百度使用了DNS负载均衡技术，每个频道一组前台服务器，一搜使用了DNS负载技术，所有频道共用一组前台服务器集群。

库表散列是常用并且最有效的解决方案。

我们在应用程序中安装业务和应用或者功能模块将数据库进行分离，不同的模块对应不同的数据库或者表，再按照一定的策略对某个页面或者功能进行更小的数据库散列，比如用户表，按照用户ID进行表散列，这样就能够低成本的提升系统的性能并且有很好的扩展性。

sohu的论坛就是采用了这样的架构，将论坛的用户、设置、帖子等信息进行数据库分离，然后对帖子、用户按照板块和ID进行散列数据库和表，最终可以在配置文件中进行简单的配置便能让系统随时增加一台低成本的数据库进来补充系统性能。

4、缓存

缓存一词搞技术的都接触过，很多地方用到缓存。网站架构和网站开发中的缓存也是非常重要。这里先讲述最基本的两种缓存。高级和分布式的缓存在后面讲述。

架构方面的缓存，对Apache比较熟悉的人都能知道Apache提供了自己的缓存模块，也可以使用外加的Squid模块进行缓存，这两种方式均可以有效的提高Apache的访问响应能力。

网站程序开发方面的缓存，Linux上提供的Memory Cache是常用的缓存接口，可以在web开发中使用，比如用Java开发的时候就可以调用MemoryCache对一些数据进行缓存和通讯共享，一些大型社区使用了这样的架构。另外，在使用web语言开发的时候，各种语言基本都有自己的缓存模块和方法，PHP有Pear的Cache模块，Java就更多了，.net不是很熟悉，相信也肯定有。

5、镜像

镜像是大型网站常采用的提高性能和数据安全性的方式，镜像的技术可以解决不同网络接入商和地域带来的用户访问速度差异，比如ChinaNet和EduNet之间的差异就促使了很多网站在教育网内搭建镜像站点，数据进行定时更新或者实时更新。

6、负载均衡

负载均衡将是大型网站解决高负荷访问和大量并发请求采用的高端解决办法。

负载均衡技术发展了多年，有很多专业的服务提供商和产品可以选择，我个人接触过一些解决方法，其中有两个架构可以给大家做参考

二、Php代码编写优化：

1. echo比print要快很多。

两个方法都会在页面上打印东西，不过echo不返回任何值，print会在成功或失败的时候返回0或1。

2. 　include\_once比include更加耗时。

因为它需要去检查你要包含的class是否已经被包含进来。

3. 对于长段落的字符串一定要使用单引号，而不是双引号。

因为双引号会去搜索字符串中的变量。比如：echo ‘This is long string’.$name就比echo “This is long string $name”要快很多。

4. 不要再循环中嵌套使用for循环

5. 如果能将函数定义为静态的

那么就不要定义成为成员函数，静态函数比成员函数快33%。

6. 如果你可以不通过正则表达式就能解决问题

那么就别用正则。正则表达式比PHP原生的函数要慢一些。

例如使用str\_replace取代preg\_replae。

7. 尽量不要使用相对路径来包含文件

在相对路径中查找文件，会现在当前目录中查找，然后依次再查找。这样查找文件就很 慢。最好是先定义WEB\_ROOT这样的常量，然后使用这个常量来包含文件。

8. 全等符号===比相等==要快

而且if(1 == ’1′)会返回true，if(0 == ”)也会返回true，而当你使用全等符号的时候if(1 ===’1′)和if(0===”)都会返回false。所以当你在程序中需要检测一些布尔变量 的时候最好使用全等符号。

三、针对thinkphp 有以下几种方式

1. 关闭调试模式

由于关闭调试模式之后，系统会自动生成项目编译缓存以及关闭日志写 入，这样可以减少很多的IO加载和日志写入的开销。

2. 开启页面压缩输出

3.1版本开始，增加了OUTPUT\_ENCODE配置参数，用于控制页面压缩输出。

3. 开启缓存

在网站部署环境安装APC或者Xcache缓存能够有效提升网站运行性能和内存占用

XCache 是一个开源的 opcode 缓存器/优化器, 这意味着他能够提高您服务器上 的 PHP 性能. 他通过把编译 PHP 后的数据缓冲到共享内存从而避免重复的编译 过程, 能够直接使用缓冲区已编译的代码从而提高速度. 通常能够提高您的页面生 成速率 2 到5 倍, 降低服务器负载.

Alternative PHP Cache (APC)是一种对PHP有效的开放源高速缓冲储存器工具，它能够缓存opcode的php中间码。

4. 字段缓存

默认情况下，字段缓存是自动生成的，在开发完成之后，基本上数据库的变动变得 比较少，因此可以考虑合并字段缓存到对应的模型类，这样能够减少每次读取字段 缓存的IO开销。合并的方法是在Runtime/Data/\_fields下面找到对应的字段缓存文件

四、 数据库优化

1、选择正确的存储引擎

以 MySQL为例，包括有两个存储引擎 MyISAM 和 InnoDB，每个引擎都有利有弊。

MyISAM 适合于一些需要大量查询的应用。InnoDB 的趋势会是一个非常复杂的存储引擎，对于一些小的应用，它会比 MyISAM 还慢。但是它支持“行锁” ，支持事务。

2、优化字段的数据类型

记住一个原则，越小的列会越快。对于大多数的数据库引擎来说，硬盘操作可能是最重大的瓶颈。所以，把你的数据变得紧凑会对这种情况非常有帮助，因为这减少了对硬盘的访问。

如果一个表只会有几列罢了（比如说字典表，配置表），那么，我们就没有理由使用 INT 来做主键，使用 MEDIUMINT, SMALLINT 或是更小的 TINYINT 会更经济一些。如果你不需要记录时间，使用 DATE 要比 DATETIME 好得多。当然，你也需要留够足够的扩展空间。

3、为搜索字段添加索引

索引并不一定就是给主键或是唯一的字段。如果在你的表中，有某个字段你总要会经常用来做搜索，那么最好是为其建立索引，除非你要搜索的字段是大的文本字段，那应该建立全文索引。

4、避免使用Select \*从数据库里读出越多的数据，那么查询就会变得越慢。并且，如果你的数据库服务器和WEB服务器是两台独立的服务器的话，这还会增加网络传输的负载。即使你要查询数据表的所有字段，也尽量不要用\*通配符，善用内置提供的字段排除定义也许能给带来更多的便利。

5、使用 ENUM 而不是 VARCHAR

ENUM 类型是非常快和紧凑的。在实际上，其保存的是 TINYINT，但其外表上显示为字符串。这样一来，用这个字段来做一些选项列表变得相当的完美。例如，性别、民族、部门和状态之类的这些字段的取值是有限而且固定的，那么，你应该使用 ENUM 而不是 VARCHAR。

6、尽可能的使用 NOT NULL

除非你有一个很特别的原因去使用 NULL 值，你应该总是让你的字段保持 NOT NULL。 NULL其实需要额外的空间，并且，在你进行比较的时候，你的程序会更复杂。 当然，这里并不是说你就不能使用NULL了，现实情况是很复杂的，依然会有些情况下，你需要使用NULL值。

7、固定长度的表会更快

如果表中的所有字段都是“固定长度”的，整个表会被认为是 “static” 或 “fixed-length”。 例如，表中没有如下类型的字段： VARCHAR，TEXT，BLOB。只要你包括了其中一个这些字段，那么这个表就不是“固定长度静态表”了，这样，MySQL 引擎会用另一种方法来处理。

固定长度的表会提高性能，因为MySQL搜寻得会更快一些，因为这些固定的长度是很容易计算下一个数据的偏移量的，所以读取的自然也会很快。而如果字段不是定长的，那么，每一次要找下一条的话，需要程序找到主键。

并且，固定长度的表也更容易被缓存和重建。不过，唯一的副作用是，固定长度的字段会浪费一些空间，因为定长的字段无论你用不用，他都是要分配那么多的空间。

使用“垂直分割”技术，你可以分割你的表成为两个一个是定长的，一个则是不定长的。

8、垂直分割 “垂直分割”是一种把数据库中的表按列变成几张表的方法，这样可以降低表的复杂度和字段的数目，从而达到优化的目的。

例如：在User表中有一个字段是家庭地址，这个字段是可选字段，相比起，而且你在数据库操作的时候除了个人信息外，你并不需要经常读取或是改写这个字段。那么，为什么不把他放到另外一张表中呢？ 这样会让你的表有更好的性能，大家想想是不是，大量的时候，我对于用户表来说，只有用户ID，用户名，口令，用户角色等会被经常使用。小一点的表总是会有好的性能。

另外，你需要注意的是，这些被分出去的字段所形成的表，你不会经常性地去Join他们，不然的话，这样的性能会比不分割时还要差，而且，会是极数级的下降。

9、EXPLAIN 你的 SELECT 查询；

使用 EXPLAIN 关键字可以让你知道MySQL是如何处理你的SQL语句的。这可以帮你分析你的查询语句或是表结构的性能瓶颈。EXPLAIN 的查询结果还会告诉你你的索引主键被如何利用的，你的数据表是如何被搜索和排序的……等等，等等。

通常我们可以对比较复杂的尤其是涉及到多表的SELECT语句，把关键字EXPLAIN加到前面。你可以使用phpmyadmin来做这个事。详情见附件explain.doc.

五、前端优化

优化完后端和数据库之后，我们紧接着要做的就是针对输出的页面优化你的前端页面和资源文件，主要包括对图片、JS和样式文件的优化。

我们建议采用下列网页性能测试工具进行检测和分析，会给出相关的优化建议：

PageSpeed 谷歌开发的工具

网站管理员和网络开发人员可以使用PageSpeed来评估他们网页的性能，并获得有关如何改进性能的建议。

yslow YSlow

可以对网站的页面进行分析，并告诉你为了提高网站性能，如何基于某些规则而进行优化。