# 一致性hash算法 - consistent hashing



# 一致性hash算法（consistent hashing）

张亮

consistent hashing 算法早在 1997 年就在论文 **Consistent hashing and random trees** 中被提出，目前在 cache 系统中应用越来越广泛；

## 1 基本场景

比如你有 N 个 cache 服务器（后面简称 cache ），那么如何将一个对象 object 映射到 N 个 cache 上呢，你很可能会采用类似下面的通用方法计算 object 的 hash 值，然后均匀的映射到到 N 个 cache ；

hash(object)%N

一切都运行正常，再考虑如下的两种情况；

1 一个 cache 服务器 m down 掉了（在实际应用中必须要考虑这种情况），这样所有映射到 cache m 的对象都会失效，怎么办，需要把 cache m 从 cache 中移除，这时候 cache 是 N-1 台，映射公式变成了 hash(object)%(N-1) ；

2 由于访问加重，需要添加 cache ，这时候 cache 是 N+1 台，映射公式变成了 hash(object)%(N+1) ；

1 和 2 意味着什么？这意味着突然之间几乎所有的 cache 都失效了。对于服务器而言，这是一场灾难，洪水般的访问都会直接冲向后台服务器；

再来考虑第三个问题，由于硬件能力越来越强，你可能想让后面添加的节点多做点活，显然上面的 hash 算法也做不到。

有什么方法可以改变这个状况呢，这就是 consistent hashing...

## 2 hash 算法和单调性

Hash 算法的一个衡量指标是单调性（ Monotonicity ），定义如下：

单调性是指如果已经有一些内容通过哈希分派到了相应的缓冲中，又有新的缓冲加入到系统中。哈希的结果应能够保证原有已分配的内容可以被映射到新的缓冲中去，而不会被映射到旧的缓冲集合中的其他缓冲区。

容易看到，上面的简单 hash 算法 hash(object)%N 难以满足单调性要求。

## 3 consistent hashing 算法的原理

consistent hashing 是一种 hash 算法，简单的说，在移除 / 添加一个 cache 时，它能够尽可能小的改变已存在 key 映射关系，尽可能的满足单调性的要求。

下面就来按照 5 个步骤简单讲讲 consistent hashing 算法的基本原理。

### 3.1 环形hash 空间

考虑通常的 hash 算法都是将 value 映射到一个 32 为的 key 值，也即是 0~2^32-1 次方的数值空间；我们可以将这个空间想象成一个首（ 0 ）尾（ 2^32-1 ）相接的圆环，如下面图 1 所示的那样。

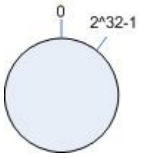


图 1 环形 hash 空间

### 3.2 把对象映射到hash 空间

接下来考虑 4 个对象 object1~object4 ，通过 hash 函数计算出的 hash 值 key 在环上的分布如图 2 所示。

hash(object1) = key1;

… …

hash(object4) = key4;

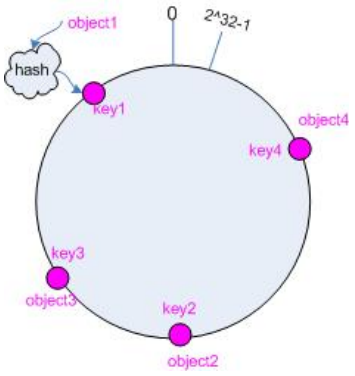


图 2 4 个对象的 key 值分布

### 3.3 把cache 映射到hash 空间

Consistent hashing 的基本思想就是将对象和 cache 都映射到同一个 hash 数值空间中，并且使用相同的 hash 算法。

假设当前有 A,B 和 C 共 3 台 cache ，那么其映射结果将如图 3 所示，他们在 hash 空间中，以对应的 hash 值排列。

hash(cache A) = key A;

… …

hash(cache C) = key C;

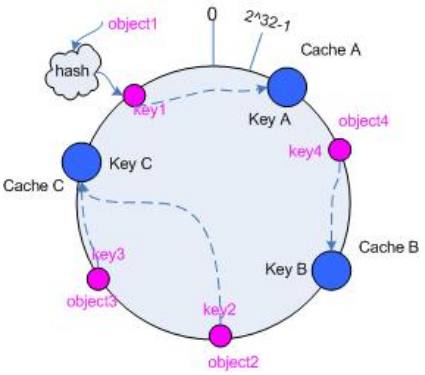

图 3 cache 和对象的 key 值分布

说到这里，顺便提一下 cache 的 hash 计算，一般的方法可以使用 cache 机器的 IP 地址或者机器名作为 hash 输入。

### 3.4 把对象映射到cache

现在 cache 和对象都已经通过同一个 hash 算法映射到 hash 数值空间中了，接下来要考虑的就是如何将对象映射到 cache 上面了。

在这个环形空间中，如果沿着顺时针方向从对象的 key 值出发，直到遇见一个 cache ，那么就将该对象存储在这个 cache 上，因为对象和 cache 的 hash 值是固定的，因此这个 cache 必然是唯一和确定的。这样不就找到了对象和 cache 的映射方法了吗？！

依然继续上面的例子（参见图 3 ），那么根据上面的方法，对象 object1 将被存储到 cache A 上； object2 和 object3 对应到 cache C ； object4 对应到 cache B ；

### 3.5 考察cache 的变动

前面讲过，通过 hash 然后求余的方法带来的最大问题就在于不能满足单调性，当 cache 有所变动时， cache 会失效，进而对后台服务器造成巨大的冲击，现在就来分析分析 consistent hashing 算法。

#### 3.5.1 移除 cache

考虑假设 cache B 去[illegible]面讲到的映射方法，这时受影响的将仅是那些沿 cache B 逆时针遍历直到下一个 cache （ cache C [illegible] 也即是本来映射到 cache B 上的那些对象。

因此这里仅需要变动对象 object4 ，将其重新映射到 cache C 上即可；参见图 4 。

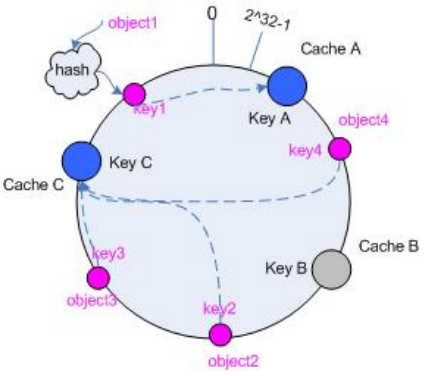


图 4 Cache B 被移除后的 cache 映射

#### 3.5.2 添加 cache

再考虑添加一台新的 cache D 的情况，假设在这个环形 hash 空间中， cache D 被映射在对象 object2 和 object3 之间。这时受影响的将仅是那些沿 cache D 逆时针遍历直到下一个 cache （ cache B ）之间的对象（它们是也本来映射到 cache C 上对象的一部分），将这些对象重新映射到 cache D 上即可。

因此这里仅需要变动对象 object2 ，将其重新映射到 cache D 上；参见图 5 。

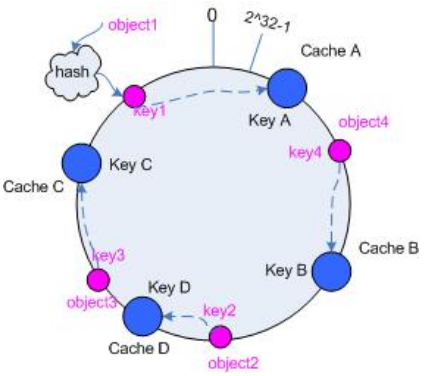


图 5 添加 cache D 后的映射关系

## 4 虚拟节点

考量 Hash 算法的另一个指标是平衡性 (Balance)，定义如下：

平衡性

平衡性是指哈希的结果能够尽可能分布到所有的缓冲中去，这样可以使得所有的缓冲空间都得到利用。

hash 算法并不是保证绝对的平衡，如果 cache 较少的话，对象并不能被均匀的映射到 cache 上，比如在上面的例子中，仅部署 cache A 和 cache C 的情况下，在 4 个对象中， cache A 仅存储了 object1 ，而 cache C 则存储了 object2 、 object3 和 object4 ；分布是很不均衡的。

为了解决这种情况， consistent hashing 引入了“虚拟节点”的概念，它可以如下定义：

“虚拟节点”（ virtual node ）是实际节点在 hash 空间的复制品（ replica ），一实际个节点对应了若干个“虚拟节点”，这个对应个数也成为“复制个数”，“虚拟节点”在 hash 空间中以 hash 值排列。

仍以仅部署 cache A 和 cache C 的情况为例，在图 4 中我们已经看到， cache 分布并不均匀。现在我们引入虚拟节点，并设置“复制个数”为 2 ，这就意味着一共会存在 4 个“虚拟节点”， cache A1, cache A2 代表了 cache A ； cache C1, cache C2 代表了 cache C ；假设一种比较理想的情况，参见图 6 。

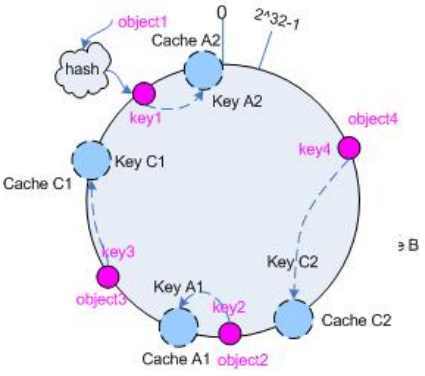


图 6 引入“虚拟节点”后的映射关系

此时，对象到“虚拟节点”的映射关系为：

objec1->cache A2 ； objec2->cache A1 ； objec3->cache C1 ； objec4->cache C2 ；

因此对象 object1 和 object2 都被映射到了 cache A 上，而 object3 和 object4 映射到了 cache C 上；平衡性有了很大提高。

引入“虚拟节点”后，映射关系就从 { 对象 -> 节点 } 转换到了 { 对象 -> 虚拟节点 } 。查询物体所在 cache 时的映射关系如图 7 所示。

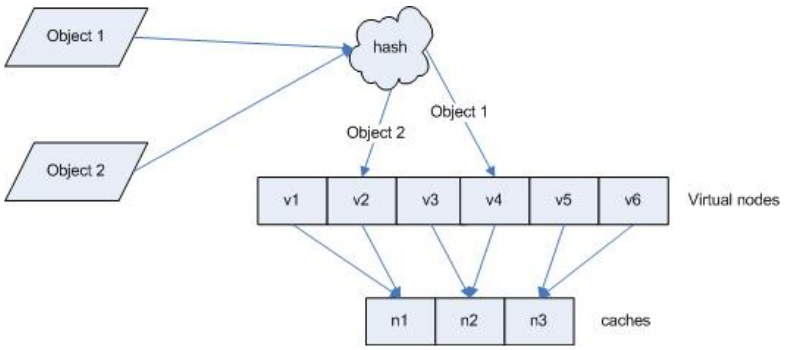


图 7 查询对象所在 cache

“虚拟节点”的 hash 计算可以采用对应节点的 IP 地址加数字后缀的方式。例如假设 cache A 的 IP 地址为 202.168.14.241 。

引入“虚拟节点”前，计算 cache A 的 hash 值：

Hash(“202.168.14.241”);

引入“虚拟节点”后，计算“虚拟节”点 cache A1 和 cache A2 的 hash 值：

Hash(“202.168.14.241#1”); // cache A1

Hash(“202.168.14.241#2”); // cache A2

## 5 小结

Consistent hashing 的基本原理就是这些，具体的分布性等理论分析应该是很复杂的，不过一般也用不到。

http://weblogs.java.net/blog/2007/11/27/consistent-hashing 上面有一个 java 版本的例子，可以参考。

http://blog.csdn.net/mayongzhan/archive/2009/06/25/4298834.aspx 转载了一个 PHP 版的实现代码。

http://www.codeproject.com/KB/recipes/lib-conhash.aspx C语言版本

一些参考资料地址：

http://portal.acm.org/citation.cfm?id=258660

http://en.wikipedia.org/wiki/Consistent_hashing

http://www.spiteful.com/2008/03/17/programmers-toolbox-part-3-consistent-hashing/

http://weblogs.java.net/blog/2007/11/27/consistent-hashing

http://tech.idv2.com/2008/07/24/memcached-004/

http://blog.csdn.net/mayongzhan/archive/2009/06/25/4298834.aspx



上一篇 Map Reduce – the Free Lunch is not over?

下一篇 Hadoop分布式文件系统：架构和设计要点

我的同类文章

算法艺术（8）

- 从MySQL Bug#67718浅谈B… 2014-06-11 阅读 1882
- KMP算法真的很简单1 2010-02-03 阅读 8463
- CRC32算法-从bit到table-dr… 2010-01-24 阅读 2502
- 求1的个数问题 2009-08-18 阅读 1267
- C/C++语言实现动态数组 2010-03-09 阅读 8844
- 对ZZL字符串匹配算法的改 … 2010-01-28 阅读 5285
- ZZL字符串匹配算法 2009-12-31 阅读 4933
- 直线扫描算法 2008-12-14 阅读 2842

主题推荐 算法 hash

猜你在找

- 《C语言/C++学习指南》加密解密篇（安全相关算法）
- C语言系列之 递归算法示例与 Windows 趣味小项目
- C语言系列之 数组与算法实战
- C语言系列之 快速排序与全排列算法
- C语言系列之 字符串压缩算法与结构体初探
- 一致性hash算法consistent hashing
- 一致性hash算法 - consistent hashing
- 一致性hash算法 - consistent hashing
- 一致性hash算法 - consistent hashing
- 一致性hash算法 - consistent hashing



查看评论

103楼 bababi 2015-12-15 17:30发表

楼主威武，感谢分享

102楼 sunfeilong1993 2015-11-16 20:59发表


mark

101楼 funnyone 2015-10-22 11:11发表


这里写错啦。
3.5.1 移除 cache

考虑假设 cache B 挂掉了，根据上面讲到的映射方法，这时受影响的将仅是那些沿 cache B 逆时针遍历直到下一个 cache （ cache C ）之间的对象，

这里应该是遍历直到下一个 cache （ cache A ）

100楼 home_king 2015-08-15 12:39发表

通过添加后缀计算出来的虚拟节点的哈希值，怎么保证就是均匀分布的呢？我觉得只能增加被选中的概率吧？

99楼 _蟾宫客- 2015-07-28 18:25发表


图文并茂, 简洁易懂! 谢谢~

98楼 huagong_adu 2015-05-19 19:31发表


讲的通俗易懂，赞！

97楼 站在巨人的肩膀上奋斗 2015-04-03 12:45发表


楼主，请教个问题：假设我们已经在cache A中存储了一个object，然后不巧的是cacheA在之后挂掉了，这时候我们需要访问object，我们如何将该object从cacheA中读取出来？ 期待楼主回复！！！

Re: sparkliang 2015-04-07 13:09发表


回复站在巨人的肩膀上奋斗：对cache而言，最重要的是不要造成数据混乱，是允许数据丢失的；比如像你说的cacheA挂掉了，那么cacheA上的数据或者被迁移到了后继cacheB上；或者没有。无论如何，下次object应该定位到cacheB（如果检测不及时定位到A上，肯定是Get返回失败了），如果cacheB有数据就可以直接返回，否则就要

从数据源重新读取。

96楼 niezhongwei321 2015-03-06 18:30发表

只实现了部分功能，添加删除Node时，数据分流问题没解决。另外加入Node时 。strcat(str,num); 会导致虚拟节点是IP1 IP12 IP123 IP1234，用string= string+num. 添加删除Node时，红黑树怎么找第一个比自己小的节点呢？ 最好是list。毕竟服务器的高性能处理几百上千个节点并排序没任何压力。后台server启动通常都是一次性分配几G内存的。

Re: sparkliang 2015-04-07 13:13发表

回复niezhongwei321：其实删除节点时，不分流也没有关系，只是等同于一小部分数据cache丢失而已；
红黑树前驱后继节点都可以比较快的找到；
如果用list，每次根据key查询cache，都是线性的；恕我不了解，list如何做到logN的查找呢，除非是skiplist；

95楼 chinadocter 2014-12-24 22:51发表

请教个问题，hash说白了就是一个映射，那既然这样，为何不直接指定服务器的映射值呢？指定的话，我们完全可以让服务器的映射值均匀分布啊？

Re: sparkliang 2014-12-26 10:30发表

回复chinadocter：嗯，你可以考虑一下一致性hash是为了解决什么问题，如果把服务器节点的hash值固定分配，对这个解决方案有什么帮助或者阻碍。

94楼 zhongjiejack 2014-09-25 12:35发表

如何理解一致性哈希中的“一致性”?

93楼 doniyou 2014-09-17 18:20发表

mark，简单易懂，蛮好的。楼主威武

92楼 yudylaw 2014-08-18 12:07发表

nice，简单易懂。

91楼 xucuiping_ 2014-08-13 11:14发表

资讯一个问题：
（1）新增cache节点后， 原来cache的数据分流一部分给这个新cache节点， 这个分流过程是后端copy过去吗（性能和一致性无法保证）？ 还是新增一个obj就顺序存放到后面2~3个cache中，不是有数据冗余

90楼 周小虎_ 2014-08-07 22:28发表

好久没有看到这么眼前一亮的博文了，写的真好！

89楼 奋力向上游 2014-08-07 21:08发表

写得不错，这里移除一个节点时，“这时受影响的将仅是那些沿 cache B 逆时针遍历直到下一个 cache （ cache C ）之间的对象”，逆时针的cache应该是A，是不是弄错了？

88楼 hujun1717 2014-08-07 16:05发表

个人觉得楼主提供的这个一致性hash存在一定的问题，不知道是不是我理解的问题，现在提出来，望指教。问题如下：
1.理想状态下，0到2^32的环形hash空间里，各个节点所持有的数值域应该是连续的，假设共三台服务器，节点1和节点2，节点3，例如节点1控制0到2^16数值域，节点2控制2^16到2^22数值域，节点3控制2^22到2^32数值域.当然节点1和节点2、节点3占有的数值域可能不是一样大的，但是一定是连续不中断的。这样情况下，增加一个节点4，如果节点4所占据的数值域也是连续不中断的，为2^27到2^30，那么原来在节点3上的一部分数据[hash之后值在2^27到2^30之中]就会漂移至节点4上。而其他节点的数据则不受影响，继续存在原来所在的节点上。
2.但是现实情况是，楼主采用的这个hash初始化方法，各个节点所持有的数值域并不连续，即可能会出现，节点1持有0到2^10数值域，还持有2^18至2^24数值域，节点2持有2^10到2^18数值域，还持有2^25至2^27数值域，节点3等等，总之就是每个节点所持有的数值域是可能会不连续的。这种情况下，新增加一个节点4，假设其持有的数值域为也不连续，为2^19到2^20,2^12到2^14，2^30到2^32。那么这种情况下，可能会导致节点1，节点2，节点3的原有的一部分数据都漂移至节点4上，满足不了单调性。
综上所述，我觉得各个节点的初始hash值域必须是连续非中断的，而非中断的。

87楼 我叫夏阳多多指教 2014-07-24 10:37发表

讲的真心不错！

86楼 AllTheWayKeep 2014-07-10 13:39发表

楼主 图文并茂 值得称赞

85楼 AllTheWayKeep 2014-07-10 13:36发表

个人理解：
1：待缓存信息Key 取Hash值 假设为 objHK1,ObjHK2,....objHKn
2：缓存服务器Key 取Hash值 假设为 svrHk1,svrHk2,...svrHkn
3：一致性原理：objHKn 和 svrHKn 肯定在相同取值空间内0~2^(32-1)

4：objHKx对应的缓存信息存储到 svrHKx中，其中svrHkx满足条件 svrHkx>=objHkx 且 svrHkx是 svrHk1 到 svrHkn中最小的一个；

5：优化方案，增加虚拟节点，原理是对物理缓存服务器Key虚拟出多个Key出来，尽量均摊存储压力即可；

84楼 cuteOpenHeart 2014-06-29 09:49发表

感觉图画的有问题，如果是顺时针，2^32-1应该在0的左边，2^32应该跟0重合。
3.5.1 移除 cache那一节，“考虑假设 cache B 挂掉了，根据上面讲到的映射方法，这时受影响的将仅是那些沿 cache B 逆时针遍历直到下一个 cache （ cache C ）之间的对象”，这里有问题，下一个cache不是cache C，而是cache A。

83楼 凝霜 2014-06-08 13:31发表

楼主，看了下你的libconhash，发现一个问题：
conhash.c -> conhash_init函数中，最后的
free(conhash);
return NULL;
这两句永远不会被执行到，另外do {...} while(0)这个感觉没必要加把，毕竟这段逻辑无论如何都要被执行，且不会被用作宏；

Re: sparkliang 2014-06-09 09:23发表

回复凝霜：谢谢，你看的很仔细。确实不需要，这个函数不会出现失败的情况。

82楼 dangercheng 2014-05-31 20:07发表

原文：再考虑添加一台新的 cache D 的情况，假设在这个环形 hash 空间中， cache D 被映射在对象 object2 和 object3 之间。这时受影响的将仅是那些沿 cache D 逆时针遍历直到下一个 cache （ cache B ）之间的对象（它们是也本来映射到 cache C 上对象的一部分），将这些对象重新映射到 cache D 上即可。
问题：楼主假设cache D 被映射在对象 object2 和 object3 之间，那么也存在一种可能cacheD被映射到object2与cacheB之间，那样的话，新添加的cacheD就不能有效实现负载吧，楼主能详细说说吗？

81楼 SvenCheng 2014-05-27 15:13发表

说的不错，通俗易懂

80楼 猪-哥-靓 2014-05-05 20:18发表

讲解的非常好，swift里面ring也是利用了一致性hash。结合这篇文字理解起来更容易

79楼 rainysky 2014-05-01 21:19发表

通俗易懂，学习了

78楼 nevermore2614 2014-04-14 16:13发表

讲的太好了

77楼 coderchenjingui 2013-12-10 16:17发表

如果现在有10台服务器构成集群，每个服务器被虚拟200次。则相当于与2000个结点。

2^32 / 2000 = 2147483
是不是说每个key最坏要查找2147483次才能确定cache服务器呢？求指点。

谢谢。

Re: 鸟样年华 2014-04-28 18:36发表

回复coderchenjingui：既然是hash，那就一定是常数级的时间复杂度

Re: lixia0417lixia0417 2014-01-10 16:57发表

回复coderchenjingui：最多查找2000次就可以找到cache了啊。如何cache排好序的话，log2000=11次就可以确定cache了啊。个人理解。

76楼 OsBelief 2013-11-18 22:17发表

学习了

75楼 Elvin_C_L 2013-10-10 15:40发表

简单易懂，顶一个

74楼 阿仆来耶 2013-10-08 15:02发表

讲的挺好的，期待有更加精彩的博文~~

73楼 leecyz 2013-09-28 22:41发表

不错哟

72楼 Evil_Moon 2013-09-12 16:24发表

写的很好，一下子就看明白了。

71楼 skwen 2013-09-12 11:47发表

顶，很好！！！
以前学习过，好久后又忘了。现在听你这样一说，更深刻了。希望以后不要再忘了！

70楼 leonzhouwei 2013-09-11 15:22发表

第二次看了，写的非常务实详尽，收藏了，感谢博主

69楼 潇潇洒洒_007 2013-08-15 11:52发表

谢谢楼主很好

68楼 weiihaisuiyueran 2013-08-02 19:06发表

good

67楼 Spark-zh 2013-07-09 20:27发表

感谢楼主的讲解，很明了。

66楼 thomastianqq 2013-07-09 17:26发表

单调性是指如果已经有一些内容通过哈希分派到了相应的缓冲中，又有新的缓冲加入到系统中。哈希的结果应能够保证原有已分配的内容可以被映射到新的缓冲中去，而不会被映射到旧的缓冲集合中的其他缓冲区。

Hi,LZ，文中关于单调性的阐述是错误的。正确的应该是：哈希的结果应能够保证原有已分配的内容可以被映射到原有的或者新的缓冲中去，而不会被映射到旧的缓冲集合中的其他缓冲区。

Re: 站在巨人的肩膀上奋斗 2015-04-03 12:41发表

回复thomastianqq：关于单调性，你和楼主的理解是一样的，人家怎么错了？

Re: lixia0417lixia0417 2014-01-10 16:49发表

回复thomastianqq：我说看那个单调性的定义这么抽象，原来是写错了。你这个才是对的啊。楼主文章不错。确实通俗易懂

65楼 lgfeng218 2013-06-26 11:59发表


通俗易懂

64楼 midstr 2013-06-18 14:54发表

通俗易懂，顶一个哈

63楼 zxh87 2013-06-03 15:18发表

很好，又看了一次！

62楼 fanjiabing 2013-05-22 16:00发表

写的很好!

61楼 mydreamongo 2013-05-20 16:27发表

讲的很好。。

60楼 bingmo01 2013-05-17 00:31发表

很给力。基本上看懂

59楼 perfectshark 2013-05-11 13:26发表

浅显易懂!

58楼 pymqqq 2013-05-04 20:18发表

非常好，谢谢楼主！学习了！

57楼 草原面朝大海 2013-04-30 22:59发表

假设我们已经在cache A中存储了一个object 1，然后不巧的是cacheA在之后被删除了，这时候我们需要访问object1，难道再一次进行hash，将object1存到下一个cache中吗？
如果上述成立，问题是我们如何将object1从cacheA中读取出来，因为在实际情况中，很有可能是cacheA是一下子坏掉了。

Re: 总版主 2013-05-07 17:08发表

回复草原面朝大海：我觉得可以把OBJ存三份，分别再顺时针往后三个cache中，这样就算cache瞬间崩溃了，那么重新复制一份备份保证有三个备份即可，加CACHE也不是问题，剪切一部分obj到新增的cache上即可。

56楼 云_腾 2013-04-07 13:46发表

讲的太好了，清晰明了，浅显易懂

55楼 轻舞飘扬 2013-03-30 17:09发表

非常好

54楼 scncpb 2013-03-27 15:13发表

作者写的和老外写的一样通俗易懂

53楼 mail_f5 2013-02-21 17:55发表

图文并茂，不错

52楼 zhouxiaoqingelse 2013-02-20 18:18发表

非常清晰，感谢。

51楼 fantisGod 2013-01-22 11:08发表

楼主，有个问题请教一下,这个算法是如何保证后面添加的服务器能承担更多的任务，我应该如何控制"更多"这个量。例如我的服务器是分等级的，如何根据这个等级来控制hash到它上面的任务。

Re: sparkliang 2013-01-22 13:57发表

回复fantisGod：首先新加入的节点，将会把后继节点的部分数据分流给自己；后面的新数据将会根据hash结果，无区别的分配到各节点上；从我了解的来看，hash本身解决不了你说的分等级问题；hash环上的所有节点都是无差别对待的。不过对于引入虚拟节点的hash环来说，一个方法就是根据优先级来设置vnode的个数，这样也可以做到能力强的节点承担更多数据的结果，从统计意义上，实际上可能略有偏差。

Re: 白俊鸿---小白 2013-09-27 11:17发表

回复sparkliang：说的没粗，如果要添加新的node，那hash值会移植到新的cache中，至于说的等级，在hash根本不存在，在环中，所有node都是平等的。

50楼 林大虫 2012-12-24 13:44发表

讲得很清晰啊

49楼 dream1baby 2012-12-18 11:32发表

赞，学习了！

48楼 blueskyltt 2012-12-03 18:30发表

楼主讲的很浅显易懂，赞

47楼 syzcch 2012-11-12 09:11发表

讲的浅显易懂

46楼 jixianwu2009 2012-11-05 10:03发表

清晰明了

45楼 月亮床 2012-10-23 18:27发表

受益非浅

44楼 ju136 2012-10-16 09:32发表

good

43楼 yangqisheng 2012-10-07 13:42发表

楼主写的很好，一看就懂。我非常喜欢图文并茂的说明。谢谢楼主。

42楼 lcw918110 2012-09-05 10:14发表

3.5.1中，这时受影响的将仅是那些沿 cache B 逆时针遍历直到下一个 cache （ cache C ）之间的对象。其中的cache C如图所示应为A。请楼主检查下。

谢谢楼主的博文。

Re: sparkliang 2012-09-05 15:42发表

回复lcw918110：赞，看的很仔细；逆时针是在A上，顺时针落在C上。

Re: lcw918110 2012-09-07 11:40发表

回复sparkliang：楼主，关于cache中使用的hash算法，楼主知道的还有哪些？学习学习。

41楼 renjunyi6666 2012-08-23 11:38发表

受益!谢谢

40楼 macrotea-cn 2012-08-17 21:05发表


学习了

39楼 skywhsq1987 2012-08-02 10:35发表

ding

38楼 m_vptr 2012-08-02 09:25发表

好文章

37楼 boYwell 2012-08-01 09:08发表

文笔很好，赞一个

36楼 大城小爱 2012-07-06 19:02发表

讲得很清晰，不错不错

35楼 brucest0078 2012-06-18 11:44发表

关于virtualNode 有一些问题，因为你的key在存储的时候可以这样，但是读取的时候只有原生的key去相关的内容，而不会把vNode的ip之类的带进去，所以这里只是理论呢还是已经有实践过？表示怀疑。

Re: sparkliang 2012-06-18 13:52发表

回复brucest0078：读取的时候，关vnode什么事呢。读取者事先是不知道key存在那个节点上的。

34楼 Honeybb 2012-06-12 09:44发表

最近在看这方面的文章，很迷茫。 楼主，分布式网络关于资源定位与一致性哈希算法和paxos算法的关系，请楼主明示啊！

Re: sparkliang 2012-06-18 13:52发表

回复Honeybb：一致性哈希算法和paxos算法，没什么关联

33楼 laigege 2012-06-02 09:48发表

楼主是好人

32楼 dengjm_2011 2012-05-23 16:41发表

很不错的文章，简单明了！

31楼 zcl198715 2012-04-29 21:34发表

分析的很好！赞一个！

30楼 leonkyd 2012-04-21 10:28发表

非常好的技术文章，赞一个

29楼 wumucheng 2012-04-06 20:04发表

讲得非常清晰，赞！

28楼 无籽-西瓜 2012-03-30 00:51发表

分析得很好

27楼 TCCaiWQ 2012-03-15 16:54发表

简洁明了

26楼 apprentice89 2012-03-05 17:22发表

赞，我看英文维基百科一直没看懂，博主的东西真是简洁明了。赞赞赞！

25楼 libobo5954451 2012-02-10 18:02发表

非常好

24楼 humingchun 2012-01-02 19:44发表

偶竟然看明白了 说明写的很清楚 呵呵

23楼 jmx_zhidai 2011-11-24 16:31发表

写的非常明白，赞一个！！！

22楼 yfk 2011-11-16 15:27发表

很赞的文章
谢谢IZ

21楼 南山道人 2011-11-16 12:47发表

顶，非常感谢！

20楼 zhaopeinow 2011-10-20 17:52发表

顶

19楼 lbqBraveheart 2011-10-12 15:03发表

顶

18楼 向良玉 2011-10-11 13:59发表

好文章

17楼 xgtogd 2011-09-25 23:26发表

很好

16楼 fairywell 2011-09-12 21:54发表

好文，图文并茂，让我一下就看明白了：）
Mark并收藏~~

15楼 parakpurple 2011-08-29 17:09发表

写的太棒了 深入浅出

14楼 bokee 2011-08-06 16:12发表

很清晰啊，想问下你的图是用什么工具画的？

Re: sparkliang 2011-08-30 09:19发表

回复bokee：visio啊

13楼 aldohou 2011-07-16 17:12发表


不得了。

12楼 wayon_yang 2011-05-30 11:49发表

[e01]
英语不好 之前没看懂 现在清晰多了

11楼 csdn程序猿 2011-05-13 20:45发表

这么好的一篇文章居然没有推荐，csdn的编辑干嘛吃去了

10楼 chemila 2011-03-31 17:45发表

[e01]

9楼 xunujNext 2011-02-17 22:28发表

不得不[e01]

8楼 printfabcd 2010-12-17 14:51发表

请问虚拟节点，怎么保证均匀分布在，那个环上呢？？O(∩_∩)O~

Re: sparkliang 2011-03-11 15:31发表

回复printfabcd：这就要由hash算法来保证了，均匀分布是概率上的均匀，当虚拟节点足够时，就能保证大概均匀了。

7楼 ccnuliu 2010-06-23 00:01发表

假如cache通过hash函数计算出的值 和 object通过hash函数计算出来的值是同一个hash值怎么办？

那object应该指向哪个cache?

Re: sparkliang 2010-06-24 09:41发表

回复ccnuliu：如果碰巧相同，就是指向具有相同hash值的cache。选择的原则是cache.hash >= object.hash；（当然在环的衔接处是例外）

Re: a43350860 2011-03-05 18:27发表

回复sparkliang：假设我给hash环开启一扇大门，所有的值进来都必须经过这一扇门，那么我只要在这一扇大门里加一个hash关卡，我不关心你传进来的值是什么类型，是什么形态，我都是统一做hash运算，然后再分配到相应的节点上面去。

6楼 ccnuliu 2010-06-23 00:01发表

问你一个问题

加入cache通过hash函数计算出的值 和 object通过hash函数计算出来的值是同一个hash值怎么办？

那object应该指向哪个cache?

5楼 chengqianl 2010-06-02 16:53发表

收益了

4楼 specific_intel 2010-05-12 16:26发表

[e01] 非常清晰，谢谢！

3楼 yayaaishenghuo 2010-03-09 13:46发表

讲得太好了

2楼 匿名用户 2010-02-22 10:06发表

[e10]

1楼 匿名用户 2010-02-22 10:05发表

[e01]

Angular Cloud Foundry Redis Scala Django Bootstrap