知乎

^{自 反 †} 分布式一致性与高可用实践



А		A,E	3,C	A,B,D				
	1	2	3	(change)	5	6	•••	
		A,I	B,C	A,B, <mark>D</mark>				
В	7	2	3	4 (change)	5	6		
	A,B,C				A,B, <mark>D</mark>			
		A,E	3,C			A,B,D		
С	1	A,E 2	3,C 3	4 (change)	5	A,B,D 6		
С	1				5			
C	1	2			5			

Paxos理论介绍(4): 动态成员变更



已关注

29 人赞同了该文章

发布于 2016-08-23 23:58 , 编辑于 2016-10-26 17:06

多数派的本质

在讲解成员变更之前,我们先回顾一下前文介绍的Paxos理论第一篇文章 Paxos理论介绍(1): 朴素 Paxos算法理论推导与证明,(仔细回顾数学定义和投票约束章节)文中提到Bqrm为一轮成功投票所需要的投票者集合,而Paxos算法理论第二条约束要求任意两个Bqrm的交集不为空,于是乎我们可以理解为Bqrm就是一个多数派的意思,因为在一个固定的投票者集合里面,取多数派作为Bqrm,肯定是满足条件的。

而所有的理论介绍,都是基于投票者集合是固定的。一旦投票者集合出现变化,Bqrm的定义将不再是多数派,Bqrm的取值将变得异常困难,而无法定义Bqrm,Paxos算法的约束就无法达成一致性。也就是说,固定的成员是Paxos算法的根基。

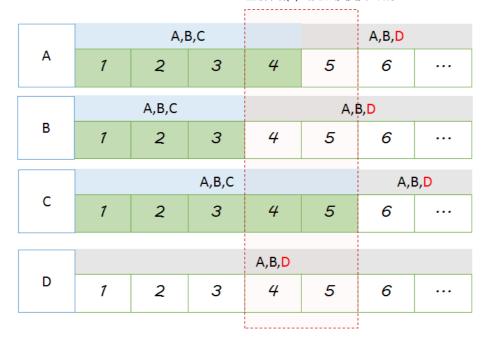
人肉配置进行成员变更?

我们再进行第二篇文章 Paxos理论介绍(2): Multi-Paxos与Leader 的回顾,通过文章我们知道 Paxos是以独立的实例的方式推进,从而产生一个一致的有序的系列,而每个实例都是单独运作的 Paxos算法。再根据上文,我们得出一个要求,在相同的实例上,我们要求各个成员所认为的成员 集合必须是一致的,也就是在一次完整的Paxos算法里面,成员其实还是固定的。

每个成员如何得知这个成员集合是什么?通常我们是通过配置文件。在通过配置的变更能否满足以上的要求呢?我们知道Multi-Paxos在推进的过程中是允许少数派落后的,而在同一个实例里面,获知Value被chosen也是有先后的,那么配置的变更可能出现在以上任何的先后夹缝内,下图演示一个更换节点C为D的样例。

https://zhuanlan.zhihu.com/p/22148265





注: 绿色代表已经获知chosen value的实例

可以观察到4这个实例,已经出现了成员混乱,(A,C),(B,D)都可以被认为是Bqrm,但明显这两个Bqrm没有交集,已经违反Paxos协议。

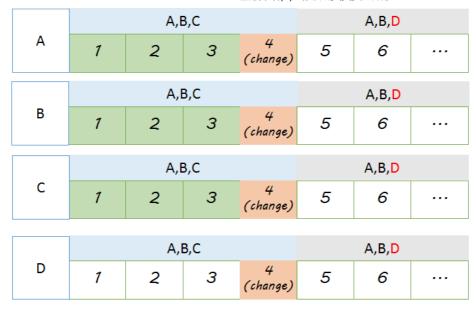
事实上我们追求的是找到一个切入时机,使得Paxos的运作程序都在这相同的时刻完成配置的原子切换,但明显在分布式环境里面能做原子切换的只有一致性算法,所以配置更新不靠谱。

题外话,如果真的要使用人肉配置更新,在工程上是有一些办法,通过一些工具加人肉的细微观察来无限逼近这个正确性,但终究只能逼近。在理论层面我们会放大任何现实中可能不会出现的细微错误,比如时间的不同步,网络包在交换机无限停留,操作系统调度导致的代码段卡壳等等,这些都会导致这些人肉方法不能上升到理论层面。况且,我们接下来要介绍的动态成员变更算法也是非常的简单。所以这些细致的问题就不展开来聊了。

Paxos动态成员变更算法

这个算法在 Paxos Made Simple 的最后一段被一句话带过,可能作者认为这个是水到渠成的事情,根本不值一提。

Multi-Paxos决议出的有序系列,一般被用来作为状态机的状态转移输入,一致的状态转移得出一致的状态,这是Paxos的基本应用。那么非常水到渠成的事情就是,成员(投票者集合)本身也是一个状态,我们通过Paxos来决议出成员变更的操作系列,那么各台机器就能获得一致的成员状态。如下图。



在4这个实例,我们通过Paxos算法来决议一个成员变更操作,所有的节点在实例4之后都能获取到成员从A,B,C变成了A,B,D,在理论上达到了原子变更的要求。

延缓变更生效

通常Paxos的工程化为了性能和进展性都会由一个Leader节点进行数据写入,而成员变更往往可能会导致Leader节点发生变化。那么变化的瞬间可能会出现大量当前Leader节点堆积请求的失败。

另外如果采用多个实例基于窗口滑动并行提交(这里暂时不展开讲,而且我觉得这个在实际工程实现中必要性也不是很大)的话,在成员变更操作被chosen后,之后的实例可能还在Paxos算法运行当中,那么这些实例的Bqrm可能已经被确定下来,所以我们不能再去改动这些实例的Bqrm。如上图例子,4的时候进行了成员变更,但是由于并行提交的关系,5和6可能都已经在提交当中了,那么他的Bqrm还是被确定下来为A,B,C,这时候我们不能去改这些实例的Bqrm为A,B,D。

为了解决这个问题,我们可以将成员生效时间延缓一下,在这个期间将请求引导到新的写入节点。

						7	
А	A,B,C						A,B,D
	1	2	3	4 (change)	5	6	
В	A,B,C						A,B,D
	1	2	3	4 (change)	5	6	
С	A,B,C						A,B,D
	1	2	3	4 (change)	5	6	
D	A,B,C						A,B,D
	1	2	3	(change)	5	6	

我们可以定义一个延缓窗口为a,成员变更点为l,则生效点为l + a。这个a根据实际情况进行调节。如上图,成员变更点在实例4,但是生效点可以在实例6之后。我们规定旧的Leader在l + a之前仍然能正常写入数据,而新的写入节点必须从l + a开始写入数据,这样可以完成一个平滑过渡。

检测到网页存在资源 资源存夸克网盘在线查看

查看资源 转存网盘

收起

Paxos理论介绍(4): 动态成员变更 - 知乎

*说的再多不如阅读源码,猛击进入我们的开源Paxos类库实现:*github.com/tencent-wech...,我们完整的实现成员变更算法。

如果觉得文章对你有帮助或者启发,麻烦点一波关注,谢谢各位。

发布于 2016-08-23 23:58, 编辑于 2016-10-26 17:06

「真诚赞赏,手留余香」

特常

还没有人赞赏, 快来当第一个赞赏的人吧!

分布式系统 Paxos 微信



检测到网页存在资源 资源存夸克网盘在线查看

查看资源

转存网盘

(4

收起



文章被以下专栏收录



分布式一致性与高可用实践

微信后台在分布式一致性与高可用上的实践经验介绍

推荐阅读



收起