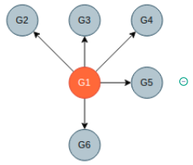
Raft(02)

网络分区

网络分区

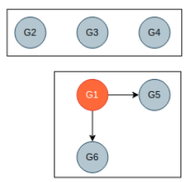
前置假设

假设当前集群节点如左图所示，且G1为当前任期的领导者，其余5个节点均为追随者



极端情况

当出现如上的网络分区时，无论两个子网络如何进行投票或者写入，均无法满足"大多数"节点这一条件，此时写入操作将会一直返回失败，直到网络分区被修复

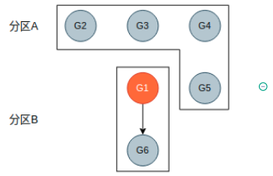


一般情况

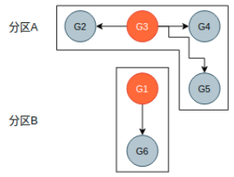
集群出现了不均匀网络分区

分区A

分区B



在此类情况下，分区A由于和领导者G1断开，将会导致分区A的节点达到心跳超时，从而选举出新的领导者

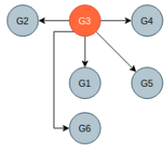


此时将存在多个领导者的情况

但是，Raft仍然处理多领导者的情况，因为在数据写入时，必须使得多数节点认可该写入，领导者才能真正将数据提交

数据写入G1 不满足"多数节点"的要求，写入将失败

数据写入G3 满足"多数节点"的要求，写入将会成功



在网络分区修复后，由于G3的任期编码大于G1，那么G1将会成为追随者，G3继续当任领导者，并且将所有的延迟数据同步至G1与G6节点之上

如果是主节点宕机下线的话，则追随者节点在心跳超时时间之后，选举出新的领导者。只不过在选举过程中服务短暂不可用

QA

Raft算法能保证强一致性吗？

可以，Raft在读写数据时相当于一个单机服务: 只有领导者才能够接受客户端的请求(读or写)，所以强一致性能够得到保证

Raft算法是通过牺牲性能的方式来达到副本数据的一致性的

Raft的性能如何？

可以看到，Raft是单领导者共识算法，所有数据的写入只能在领导者节点进行，相当于是一个单机服务

在并发量相当高的服务中，可能会成为性能瓶颈。所以在使用Raft算法的数据系统实现中，通常会对数据进行分区，并建立多个Raft分区集群以降低单机负载

基于Raft实现的集群，节点数是不是越多越好？

答案是否定的，如果集群采用Raft算法实现共识，那么节点数量最好为奇数个，且最佳节点数为3个或者是5个

奇数个节点

因为Raft采用"大多数"这一基础标准，那么奇数个节点更容易地得到

少量节点

节点数量越多就越容易出现网络分区，对集群的稳定性会有一定的影响

在实际应用中，基于Raft算法实现的分布式数据存储中，存储的并不是业务数据，而是元数据(Metadata)。特点是关键但少量

大量的数据写入但少量的数据读取，如何提高性能？

对数据进行分区，组成多网格Raft集群

更好地理解Raft

选举以及数据写入的动画演示 http://thesecretlivesofdata.com/raft/