(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 111526186 A
(43)申请公布日 2020.08.11

(21)申请号 202010278020.0

(22)申请日 2020.04.10

(71)申请人 河海大学
地址 210000 江苏省南京市江宁区佛城西路8号

(72)发明人 许仁益 韩立新

(74)专利代理机构 南京经纬专利商标代理有限公司 32200
代理人 汤金燕

(51)Int.Cl.
*H04L 29/08*(2006.01)
*H04L 29/12*(2006.01)

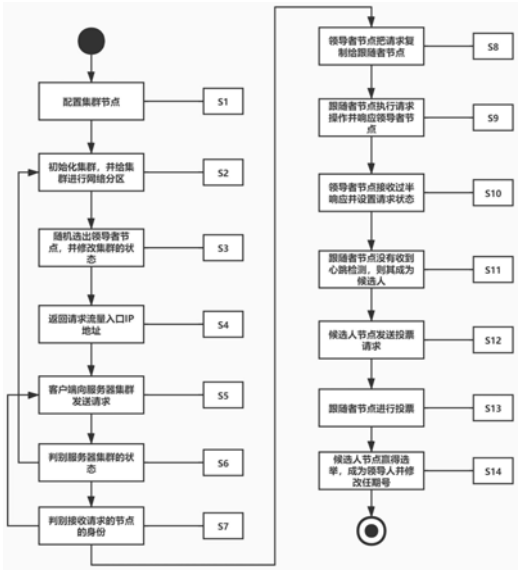
权利要求书2页 说明书8页 附图2页

(54)发明名称

基于Raft的分布式服务器集群配置方法

(57)摘要

本发明公开了一种基于Raft的分布式服务器集群配置方法，通过配置集群节点，对各个集群节点进行分区，修改整个服务器集群的状态，使网络分区的领导者节点将客户端的请求发送给该网络分区的其余跟随者节点，当跟随者节点将客户端的请求所包括的操作执行在本地数据库里面以后，向相应网络分区的领导者节点发送响应，当领导者节点收到相应网络分区跟随者节点的响应超过设定数量后，将客户端的请求设为提交状态，此外跟随者节点如果在选举超时时间之内没有接收到领导者节点发送的心跳检测信息，那么该跟随者节点成为候选人节点，按照相应规则使该候选人节点成为该网络分区的新任领导者节点，并更新任期号，以实现对相应分布式服务器集群的配置。




CN 111526186 A

1.一种基于Raft的分布式服务器集群配置方法,其特征在于,包括如下步骤:

S1,配置集群节点,读取管理员输入的各个集群节点IP地址;

S2,在初始化状态,服务器集群中的节点都是处于跟随者状态时,对各个集群节点进行分区,得到多个网络分区;

S3,在每个网络分区中随机选出一个领导者节点,等到所有网络分区的小集群都有了领导者节点以后,修改整个服务器集群的状态;

S4,系统返回一个请求流量入口的IP地址;

S5,客户端向服务器集群发送请求;

S6,若整个服务器集群处于初始化状态,则返回执行步骤S2,若服务器集群不全处于初始化状态,则执行步骤S7;

S7,在一个网络分区的集群节点中,如果客户端的请求发送到跟随者节点,则拒绝接收此请求,并重定向,返回执行步骤S5;如果客户端的请求发送到领导者节点,则接收此请求;

S8,每个网络分区的领导者节点将客户端的请求发送给该网络分区的其余跟随者节点;

S9,当跟随者节点将客户端的请求所包括的操作执行在本地数据库里面以后,向相应网络分区的领导者节点发送响应;

S10,当领导者节点收到相应网络分区跟随者节点的响应超过设定数量后,将客户端的请求设为提交状态;

S11,跟随者节点如果在选举超时时间之内没有接收到领导者节点发送的心跳检测信息,那么该跟随者节点成为候选人节点;

S12,候选人节点先投自己一票,并向所在网络分区的其余节点发送投票请求;

S13,收到投票请求的跟随者节点如果在该任期里没有将选票投出,则将选票投给该候选人节点;

S14,如果候选人节点获得的选票数超过设定票数,则该候选人节点成为该网络分区的新任领导者节点,并更新任期号。

2.根据权利要求1所述的基于Raft的分布式服务器集群配置方法,其特征在于,对各个集群节点进行分区包括:

以网络作为分区的依据对所有节点进行分区。

3.根据权利要求1所述的基于Raft的分布式服务器集群配置方法,其特征在于,修改整个服务器集群的状态包括:

在内存中定义一个整形变量,以记录服务器集群的状态;所述整形变量的初始值为0,0表示初始化状态;修改后的服务器集群的状态为1,1表示运行状态。

4.根据权利要求1所述的基于Raft的分布式服务器集群配置方法,其特征在于,重定向包括:

向客户端返回一个响应信息,使客户端再次客户端向服务器集群发送请求。

5.根据权利要求1所述的基于Raft的分布式服务器集群配置方法,其特征在于,在每个网络分区中随机选出一个领导者节点,等到所有网络分区的小集群都有了领导者节点以后,修改整个服务器集群的状态包括:

采用随机函数在各个网络分区所有节点ID的范围内进行选取,选出的ID对应的节点为

相应网络分区的领导者节点;待所有网络分区都有领导者节点后,把整个服务器集群的状态从初始状态修改为运行状态;其中初始化状态下,集群中的所有节点都是跟随者。

# 基于Raft的分布式服务器集群配置方法

## 技术领域

[0001] 本发明涉及分布式信号处理技术领域,尤其涉及一种基于Raft的分布式服务器集群配置方法。

## 背景技术

[0002] 在现代社会,很多时候处理一些事务需要使用很多台计算机,那么如何保证这些机器在一起运作的时候像一个整体而不相互分离隔绝就是分布式系统的一致性算法需要解决的问题。一致性并不意味着结果的正确性,而是分布式系统中的多个物理节点在经过处理后所展示的状态是否一致。现在人们普遍使用的分布式一致性算法是Paxos算法,Paxos算法从1990年提出到现在已经有三十年的时间了,期间有很多的研究者和团队试图改进Paxos算法,但是Paxos算法本身晦涩难懂,在实际系统的应用上也难以推进,必须要对其结构进行大幅度修改才能应用。所以我们迫切需要找到一种简便且易于理解的一致性算法来提升工作效率以及为以后的一致性算法的教学和推广打下基础,而本发明基于的Raft算法是在该种情况下由Stanford提出,意在取代Paxos算法,在Raft算法一经推出后立即得到一致性算法领域学者的重视,通过对Raft算法的不断研究和改进,越来越多的研究者和团队将一致性算法研究和应用的重点转移到了Raft算法上来。

[0003] 随着互联网技术的不断发展,各大互联网企业的业务量也呈现出爆发式的增长。故而,分布式架构就显得越发重要。目前,国内外的很多公司和研究机构都对分布式环境下的一致性处理方法进行了研究,现根据已有方案中应用的主要技术概括目前研究和水平如下:

[0004] Basic-Paxos算法

[0005] 基于Paxos协议构建的系统,只需要系统中超过半数的节点在线且相互通信正常即可正常对外提供服务。它的核心实现Paxos Instance主要包括两个阶段:准备阶段(prepare phase)和提议阶段(accept phase)。Basic-Paxos算法的基本流程如下:

[0006] 获取一个ProposalId,为了保证ProposalId递增,可以采用时间戳+serverId方式生成;

[0007] 提议者向所有节点广播prepare(n)请求;

[0008] 接收者比较n和minProposal,如果n>minProposal,表示有更新的提议,minProposal=n;否则将(acceptedProposal,acceptedValue)返回;

[0009] 提议者接收到过半数请求后,如果发现有acceptedValue返回,表示有更新的提议,保存acceptedValue到本地,然后跳转1,生成一个更高的提议;

[0010] 到这里表示在当前paxos instance内,没有优先级更高的提议,可以进入第二阶段,广播accept(n,value)到所有节点;

[0011] 接收者比较n和minProposal,如果n>=minProposal,则acceptedProposal=minProposal=n,acceptedValue=value,本地持久化后,返回;否则,返回minProposal。

[0012] 提议者接收到过半数请求后,如果发现有返回值>n,表示有更新的提议,跳转1;否

则value达成一致。

[0013] 两阶段必不可少,Prepare阶段的作用是阻塞旧的提议,并且返回已经接收到的acceptedProposal。

[0014] Multi-Paxos算法

[0015] Paxos算法是对一个值达成一致,Multi-Paxos是连续多个Paxos Instance来对多个值达成一致,这里最核心的原因是Multi-Paxos协议中有一个Leader。Leader是系统中唯一的Proposal,在Lease租约周期内所有提案都有相同的ProposalId,可以跳过Prepare阶段,议案只有Accept过程,一个ProposalId可以对应多个Value,所以称为Multi-Paxos。

[0016] 选举:首先我们需要有一个Leader,其实选举的实质也是一次Paxos算法的过程,只不过这次确定的“谁是Leader”这个值。由于任何一个节点都可以发起提议,在并发情况下,可能会出现多主的情况,比如A,B先后当选为Leader。为了避免频繁选举,当选Leader的节点要马上树立自己的Leader权威(让其它节点知道它是Leader),写一条特殊日志(Start-Working日志)确认其身份。根据多数派原则,只有一个Leader的StartWorking日志可以达成多数派。Leader确认身份后,可以通过了Lease机制(租约)维持自己的Leader身份,使得其它Proposal不再发起提案,这样就进入了Leader任期,由于没有并发冲突,因此可以跳过Prepare阶段,直接进入Accept阶段。通过分析可知,选出Leader后,Leader任期内的所有日志都只需要一个网络RTT(Round Trip Time)即可达成一致。

[0017] 新主恢复流程:由于Paxos中并没有限制,任何节点都可以参与选举并最终成为Leader,这就无法保证新选出的Leader包含了所有日志,可能存在空洞,因此在真正提供服务前,还存在一个获取所有已提交日志的恢复过程。领导者节点向所有成员查询最大logId的请求,收到多数节点响应后,选择最大的logId作为日志恢复结束点,这里多数节点的意义在于恢复结束点包含了所有达成一致的日志,当然也可能包含了没有达成多数节点的日志。获得logId后,从头开始对每个logId逐条进行Paxos协议,因为在新主获得所有日志之前,系统是无法提供服务的。为了优化,引入了Confirm机制,就是将已经达成一致的logId告诉其它Acceptor,Acceptor写一条Confirm日志到日志文件中。那么Leader在重启后,扫描本地日志,对于已经拥有Confirm日志的Log,就不会重新发起Paxos流程。同样的,在响应客户端请求时,对于没有Confirm日志的Log,需要重新发起一轮Paxos流程。由于没有严格要求Confirm日志的位置,可以批量发送。为了确保重启时,不需要对太多已提价的Log进行Paxos流程,需要将Confirm日志与最新提交的logId保持一定的距离。

[0018] 性能优化:Basic-Paxos过程的一次日志确认,需要至少两次磁盘写操作(Prepare,Promise)和两次网络RTT(Prepare,Promise)。Multi-Paxos利用一阶段提交(省去Prepare阶段),将一次日志确认缩短为一个RTT和一次磁盘写操作;通过Confirm机制,可以缩短新Leader的恢复时间。为了提高性能,我们还可以实现一批日志作为一个组提交,要么成功一批,要么都不成功,这点类似于Group-Commit,通过RT换取吞吐量。

[0019] 在过去的十几年里,Paxos算法一直是分布式一致性领域里面的标杆,但是由于其复杂性、难以理解,故而难以在实际的商业系统中实现,这就容易使分布式服务器集群的相关配置过程变得复杂,影响配置效率。

## 发明内容

[0020] 针对以上问题，本发明提出一种基于Raft的分布式服务器集群配置方法。

[0021] 为实现本发明的目的，提供一种基于Raft的分布式服务器集群配置方法，包括如下步骤：

[0022] S1，配置集群节点，读取管理员输入的各个集群节点IP地址；

[0023] S2，在初始化状态，服务器集群中的节点都是处于跟随者状态时，对各个集群节点进行分区，得到多个网络分区；

[0024] S3，在每个网络分区中随机选出一个领导者节点，等到所有网络分区的小集群都有了领导者节点以后，修改整个服务器集群的状态；

[0025] S4，系统返回一个请求流量入口的IP地址；

[0026] S5，客户端向服务器集群发送请求；

[0027] S6，若整个服务器集群处于初始化状态，则返回执行步骤S2，若服务器集群不全处于初始化状态，则执行步骤S7；

[0028] S7，在一个网络分区的集群节点中，如果客户端的请求发送到跟随者节点，则拒绝接收此请求，并重定向，返回执行步骤S5；如果客户端的请求发送到领导者节点，则接收此请求；

[0029] S8，每个网络分区的领导者节点将客户端的请求发送给该网络分区的其余跟随者节点；

[0030] S9，当跟随者节点将客户端的请求所包括的操作执行在本地数据库里面以后，向相应网络分区的领导者节点发送响应；

[0031] S10，当领导者节点收到相应网络分区跟随者节点的响应超过设定数量后，将客户端的请求设为提交状态；

[0032] S11，跟随者节点如果在选举超时时间之内没有接收到领导者节点发送的心跳检测信息，那么该跟随者节点成为候选人节点；

[0033] S12，候选人节点先投自己一票，并向所在网络分区的其余节点发送投票请求；

[0034] S13，收到投票请求的跟随者节点如果在该任期里没有将选票投出，则将选票投给该候选人节点；

[0035] S14，如果候选人节点获得的选票数超过设定票数，则该候选人节点成为该网络分区的新任领导者节点，并更新任期号。

[0036] 在一个实施例中，对各个集群节点进行分区包括：

[0037] 以网络作为分区的依据对所有节点进行分区。

[0038] 在一个实施例中，修改整个服务器集群的状态包括：

[0039] 在内存中定义一个整形变量，以记录服务器集群的状态；所述整形变量的初始值为0，0表示初始化状态；修改后的服务器集群的状态为1，1表示运行状态。

[0040] 在一个实施例中，重定向包括：

[0041] 向客户端返回一个响应信息，使客户端再次客户端向服务器集群发送请求。

[0042] 在一个实施例中，在每个网络分区中随机选出一个领导者节点，等到所有网络分区的小集群都有了领导者节点以后，修改整个服务器集群的状态包括：

[0043] 采用随机函数在各个网络分区所有节点ID的范围内进行选取，选出的ID对应的节

点为相应网络分区的领导者节点;待所有网络分区都有领导者节点后,把整个服务器集群的状态从初始状态修改为运行状态;其中初始化状态下,集群中的所有节点都是跟随者。

[0044] 上述基于Raft的分布式服务器集群配置方法,通过配置集群节点,对各个集群节点进行分区,得到多个网络分区,修改整个服务器集群的状态,使每个网络分区的领导者节点将客户端的请求发送给该网络分区的其余跟随者节点,当跟随者节点将客户端的请求所包括的操作执行在本地数据库里面以后,向相应网络分区的领导者节点发送响应,当领导者节点收到相应网络分区跟随者节点的响应超过设定数量后,将客户端的请求设为提交状态,此外跟随者节点如果在选举超时时间之内没有接收到领导者节点发送的心跳检测信息,那么该跟随者节点成为候选人节点,按照相应规则使该候选人节点成为该网络分区的新任领导者节点,并更新任期号,以实现对相应分布式服务器集群的配置。其中采用了强领导者的方式来简化数据一致性处理方法,赋予了领导者节点更多的功能和责任,在心跳检测信息的基础上增加了随机选举超时时间,以此来进行领导者节点选举,在可理解性上大大增强,易于实现,且相应系统操作简洁,只需进行相应节点的配置,无需其余复杂操作。

## 附图说明

[0045] 图1是一个实施例的基于Raft的分布式数据一致性处理系统的模块组成图;

[0046] 图2是一个实施例的基于Raft的分布式服务器集群配置方法流程图。

## 具体实施方式

[0047] 为了使本申请的目的、技术方案及优点更加清楚明白,以下结合附图及实施例,对本申请进行进一步详细说明。应当理解,此处描述的具体实施例仅仅用以解释本申请,并不用于限定本申请。

[0048] 在本文中提及“实施例”意味着,结合实施例描述的特定特征、结构或特性可以包含在本申请的至少一个实施例中。在说明书中的各个位置出现该短语并不一定均是指相同的实施例,也不是与其它实施例互斥的独立的或备选的实施例。本领域技术人员显式地和隐式地理解的是,本文所描述的实施例可以与其它实施例相结合。

[0049] 参考图1所示,图1为一个实施例的基于Raft的分布式服务器集群配置方法流程图,包括如下步骤:

[0050] S1,配置集群节点,读取管理员输入的各个集群节点IP地址。

[0051] S2,在初始化状态,服务器集群中的节点都是处于跟随者状态时,对各个集群节点进行分区,得到多个网络分区。

[0052] 具体地,对各个集群节点进行分区包括:

[0053] 以网络作为分区的依据对所有节点进行分区。

[0054] 进一步地,确定网络分区的具体过程及其作用为:根据不同的网络地址给集群中的节点进行划分,把合适数量的服务器节点划分到同一个网络区域中;网络分区的作用在于能更好地管理服务器节点,也能把客户端发送的请求通过负载均衡的方法更好地分配给合适的节点进行处理。

[0055] S3,在每个网络分区中随机选出一个领导者节点,等到所有网络分区的小集群都有了领导者节点以后,修改整个服务器集群的状态。

[0056] 具体地，修改整个服务器集群的状态包括：

[0057] 在内存中定义一个整形变量，以记录服务器集群的状态；所述整形变量的初始值为0，0表示初始化状态；修改后的服务器集群的状态为1，1表示运行状态；即整形变量的初始值为0；修改整个服务器集群的状态就是把该整型变量从0改为1。

[0058] 进一步地，在每个网络分区中随机选出一个领导者节点，等到所有网络分区的小集群都有了领导者节点以后，修改整个服务器集群的状态包括：

[0059] 采用随机函数在各个网络分区所有节点ID的范围内进行选取，选出的ID对应的节点为相应网络分区的领导者节点；待所有网络分区都有领导者节点后，把整个服务器集群的状态从初始状态修改为运行状态；其中初始化状态下，集群中的所有节点都是跟随者。即，初始化状态下，集群中的所有节点都是跟随者，在每个网络分区中随机选出一个领导人节点，采用随机函数在该分区所有节点ID的范围内进行选取，选出的ID对应的节点为该分区的领导人(领导者节点)；待所有分区都有领导人后，把整个服务器集群的状态从初始状态修改为运行状态。

[0060] S4，系统返回一个请求流量入口的IP地址。

[0061] 上述系统可以指数据一致性处理系统。在一个示例中，这个系统用于处理服务器集群的数据一致性的，故而与服务器有关。此时系统与服务器的关系是“管理”与“被管理”的关系。

[0062] S5，客户端向服务器集群发送请求。

[0063] S6，若整个服务器集群处于初始化状态，则返回执行步骤S2，若服务器集群不全处于初始化状态，则执行步骤S7。

[0064] S7，在一个网络分区的集群节点中，如果客户端的请求发送到跟随者节点，则拒绝接收此请求，并重定向，返回执行步骤S5；如果客户端的请求发送到领导者节点，则接收此请求。

[0065] 具体地，重定向包括：

[0066] 向客户端返回一个响应信息，使客户端再次客户端向服务器集群发送请求。

[0067] 具体地，判别接收请求的节点的状态的具体流程可以包括：请求到达服务器集群后，若收到此请求的节点是跟随者，则拒绝处理此请求，并让客户端重定向，客户端需再次发送该请求到集群；若收到请求的节点是领导人，则把该请求复制给该分区的其余跟随者节点去执行操作；此过程是强领导者机制的重要体现，所有客户端发送的请求都需要经过领导人节点。

[0068] S8，每个网络分区的领导者节点将客户端的请求发送给该网络分区的其余跟随者节点。

[0069] S9，当跟随者节点将客户端的请求所包括的操作执行在本地数据库里面以后，向相应网络分区的领导者节点发送响应。

[0070] S10，当领导者节点收到相应网络分区跟随者节点的响应超过设定数量后，将客户端的请求设为提交状态。

[0071] 上述设定数量可以依据相应网络分区中的跟随者节点数设置，比如设为超过相应网络分区中的跟随者节点数一半的值等等。

[0072] 上述步骤明确了请求需要经过集群中的大多数节点操作后才能真正提交到集群

中。

[0073] S11,跟随者节点如果在选举超时时间之内没有接收到领导者节点发送的心跳检测信息,那么该跟随者节点成为候选人节点。

[0074] S12,候选人节点先投自己一票,并向所在网络分区的其余节点发送投票请求。

[0075] S13,收到投票请求的跟随者节点如果在该任期里没有将选票投出,则将选票投给该候选人节点。

[0076] S14,如果候选人节点获得的选票数超过设定票数,则该候选人节点成为该网络分区的新任领导者节点,并更新任期号。

[0077] 上述设定票数可以依据相应网络分区中的跟随者节点数设置,比如设为超过相应网络分区中的跟随者节点数一半的值等等。

[0078] 本实施例在Raft算法的基础上提出了一种强领导者、更加容易理解、更易于实现的分布式一致性处理方法及系统。该方法通过选举出一个领导者节点,给予领导者更多的功能和责任,从而降低算法的复杂性,并更易于实现;该系统用于配置集群节点、分发请求流量以及处理节点的数据一致性。

[0079] 上述基于Raft的分布式服务器集群配置方法,通过配置集群节点,对各个集群节点进行分区,得到多个网络分区,修改整个服务器集群的状态,使每个网络分区的领导者节点将客户端的请求发送给该网络分区的其余跟随者节点,当跟随者节点将客户端的请求所包括的操作执行在本地数据库里面以后,向相应网络分区的领导者节点发送响应,当领导者节点收到相应网络分区跟随者节点的响应超过设定数量后,将客户端的请求设为提交状态,此外跟随者节点如果在选举超时时间之内没有接收到领导者节点发送的心跳检测信息,那么该跟随者节点成为候选人节点,按照相应规则使该候选人节点成为该网络分区的新任领导者节点,并更新任期号,以实现对相应分布式服务器集群的配置。其中采用了强领导者的方式来简化数据一致性处理方法,赋予了领导者节点更多的功能和责任,在心跳检测信息的基础上增加了随机选举超时时间,以此来进行领导者节点选举,在可理解性上大大增强,易于实现,且相应系统操作简洁,只需进行相应节点的配置,无需其余复杂操作。

[0080] 在一个实施例中,运行上述基于Raft的分布式服务器集群配置方法的分布式数据一致性处理系统采用了多模块的结构,具体可以包括:

[0081] 集群节点配置模块:系统管理员在该模块中对集群节点的IP地址进行配置,以便系统对集群节点的数据一致性进行处理;

[0082] 请求流量入口模块:客户端对于整个集群的请求都需要经过该入口模块,该模块用于请求流量的分发;

[0083] 集群节点数据一致性处理模块:该模块是整个数据一致性处理系统的核心,该模块基于Raft算法处理集群节点的数据一致性。

[0084] 在一个实施例中,上述基于Raft的分布式服务器集群配置方法也可以通过如下过程表述:

[0085] S1、如附图1所示,本分布式数据一致性处理系统中包含集群节点配置模块,管理员需在该模块中输入集群中所有节点的IP地址进行配置并提交至系统中的集群节点数据一致性处理模块;

[0086] S2、如附图2所示,初始化状态下,服务器集群中的节点都应处于跟随者状态,所以

给整个服务器集群进行初始化操作;一般在实际的商业系统环境中,服务器集群中的节点数量是很多的,所以应该以网络为依据对整个集群中的节点进行分区,分区的目的是为了更方便、更灵活地管理节点,也为了更好地对外提供服务,因为分区以后对于客户端的请求可以更方便地进行负载均衡。

[0087] S3、在每个网络分区的服务器集群中随机选出一个领导人节点,可以把该分区的所有节点的ID作为范围,采用随机函数从该范围中选出一个数字,选出的数字对应的ID的节点作为领导人。待所有分区的服务器集群都有领导人节点后,把整个集群的状态从“初始状态”修改为“运行状态”。

[0088] S4、如附图1所示,在系统对集群节点配置完毕后,分布式数据一致性处理系统返回一个请求流量入口的IP地址至请求流量入口模块,该模块用于接收客户端发送的所有请求,并将这些请求进行分流至集群节点。

[0089] S5、客户端向服务器集群发送请求。在集群接收请求之前需要对客户端的请求进行负载均衡,通过综合判断每个分区的节点的负载情况(包括节点处理任务数、内存利用率、磁盘占有率等)来选择合适的分区里面的节点接收该请求。

[0090] S6、判断整个服务器集群的状态,如果集群处于“初始状态”,则拒绝接收该请求,并回到S1;如果集群处于“运行状态”,则进行S7;

[0091] S7、判断接收到请求的节点的状态,如果该节点是跟随者,则拒绝接收此请求,并响应客户端,让客户端重定向,再次发送该请求;如果该节点是领导人,那么就处理请求。

[0092] 在本方法中,相比于跟随者节点,领导人节点有更多的功能和责任,客户端发送给服务器集群的所有请求都必须经过领导人接收处理,跟随者无权接收客户端的请求,只能让客户端重定向;这是一种强领导者机制,可以在降低可理解性的基础上保证集群的数据一致性。

[0093] S8、接收了客户端的请求后,每个分区里的领导人节点需要把该请求发送给该分区里面的所有跟随者节点,这是维护集群数据一致性的开始步骤。

[0094] S9、跟随者接收到领导人发送的请求后,需要执行请求里面对于数据库的操作,对于数据库进行操作以后,跟随者会接收到执行结果,根据执行结果来响应该分区的领导人节点;如果执行成功,则返回执行成功的标志给领导人节点,否则返回执行失败的标志给领导人节点。

[0095] S10、当领导人节点收到该分区里面的大多数(超过一半)跟随者节点执行成功的信息时,则发送该请求的执行成功的标志给一台专门管理请求提交状态的专有节点,如果该专有节点收到大多数(超过一半)分区的领导人节点发送的该请求执行成功的信息,那么该专有节点会响应所有分区的领导人节点,通知各领导人节点该请求可以提交,故而,该请求处于提交状态;否则,该请求就不能真正提交,在该请求不能真正提交的状态下,需要各跟随者节点回滚该请求中对于数据库的操作。

[0096] S11、跟随者节点如果在选举超时时间结束之前没有收到该分区里面的领导人的心跳检测信息,那么该跟随者就成为候选人,使任期号在当前基础上加一,并发起选举。

[0097] 在某个节点成为领导人后,为了维护其领导人的地位,也为了让跟随者节点知道其存活状态,需要领导人节点每隔固定的时间(心跳检测时间)给其分区的跟随者节点发送心跳检测信息,跟随者节点在收到心跳检测信息后就将其选举超时时间归零,使其重新开

始计时。

[0098] 为了选举的成功率，在设置节点的选举超时时间时，是采用一个区域范围内的随机数进行设置，一般这个范围是在150毫秒到300毫秒之间；这样做以后，每个节点的选举超时时间在很大概率上是不同的，所以在如果在选举超时时间结束时没有收到领导人的心跳检测信息，那么一般是选举超时时间最短的跟随者节点成为候选人，而且很大概率上是只有一个节点成为候选人，这样就避免了多个候选人进行选举，而其获得的票数一样继而无法选举出领导人的现象出现。

[0099] S12、候选人节点先投自己一票，然后给其所属的分区里的其余跟随者节点发送投票请求。

[0100] S13、收到投票请求的跟随者节点如果在当前任期中没有将自己的票投出，那么就把选票投给请求的候选人，对于单个跟随者节点而言，在有多个(大于一个)候选人的情况下，投票采用先来先得的原则；如果跟随者节点已经在当前任期中投出了选票，则不能再次投票。

[0101] S14、如果候选人获得了其所属的分区里的大多数跟随者节点的投票，则其成为该分区里的新任领导者，并真正地更新任期号；如果出现S9所述的多个候选人票数相等而无法选出领导人的情况，那么此次选举作废，需要重新选举，直到选举出领导人为止。

[0102] 以上实施例的各技术特征可以进行任意的组合，为使描述简洁，未对上述实施例中的各个技术特征所有可能的组合都进行描述，然而，只要这些技术特征的组合不存在矛盾，都应当认为是本说明书记载的范围。

[0103] 需要说明的是，本申请实施例所涉及的术语“第一\第二\第三”仅仅是区别类似的对象，不代表针对对象的特定排序，可以理解地，“第一\第二\第三”在允许的情况下可以互换特定的顺序或先后次序。应该理解“第一\第二\第三”区分的对象在适当情况下可以互换，以使这里描述的本申请的实施例能够以除了在这里图示或描述的那些以外的顺序实施。

[0104] 本申请实施例的术语“包括”和“具有”以及它们任何变形，意图在于覆盖不排他的包含。例如包含了一系列步骤或模块的过程、方法、装置、产品或设备没有限定于已列出的步骤或模块，而是可选地还包括没有列出的步骤或模块，或可选地还包括对于这些过程、方法、产品或设备固有的其它步骤或模块。

[0105] 以上所述实施例仅表达了本申请的几种实施方式，其描述较为具体和详细，但并不能因此而理解为对发明专利范围的限制。应当指出的是，对于本领域的普通技术人员来说，在不脱离本申请构思的前提下，还可以做出若干变形和改进，这些都属于本申请的保护范围。因此，本申请专利的保护范围应以所附权利要求为准。

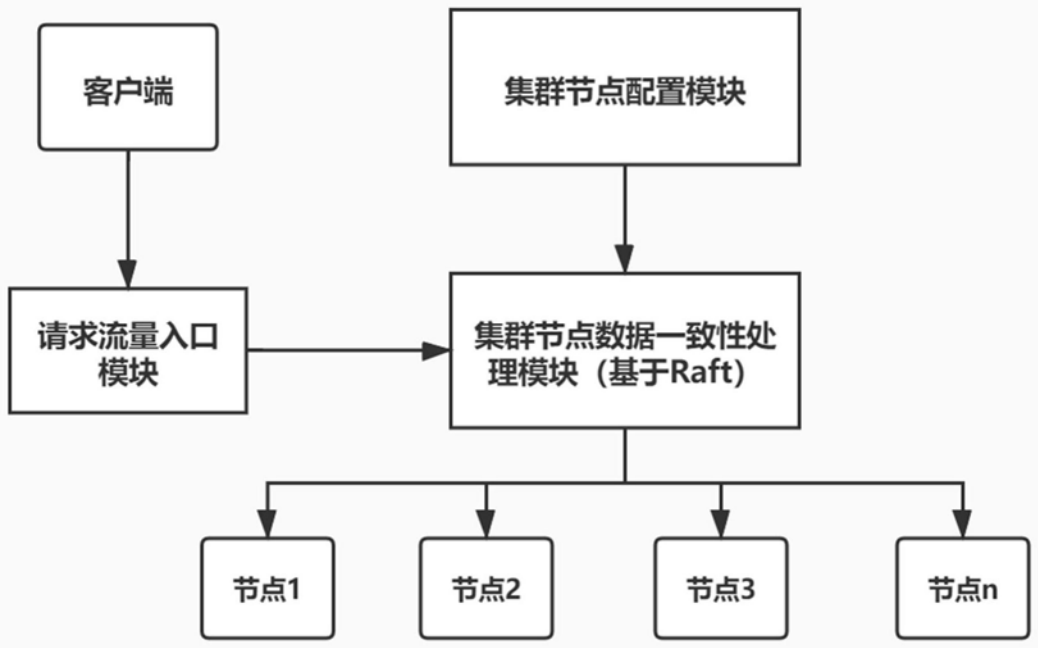

客户端
集群节点配置模块
请求流量入口模块
集群节点数据一致性处理模块（基于Raft）
节点1
节点2
节点3
节点n

图1

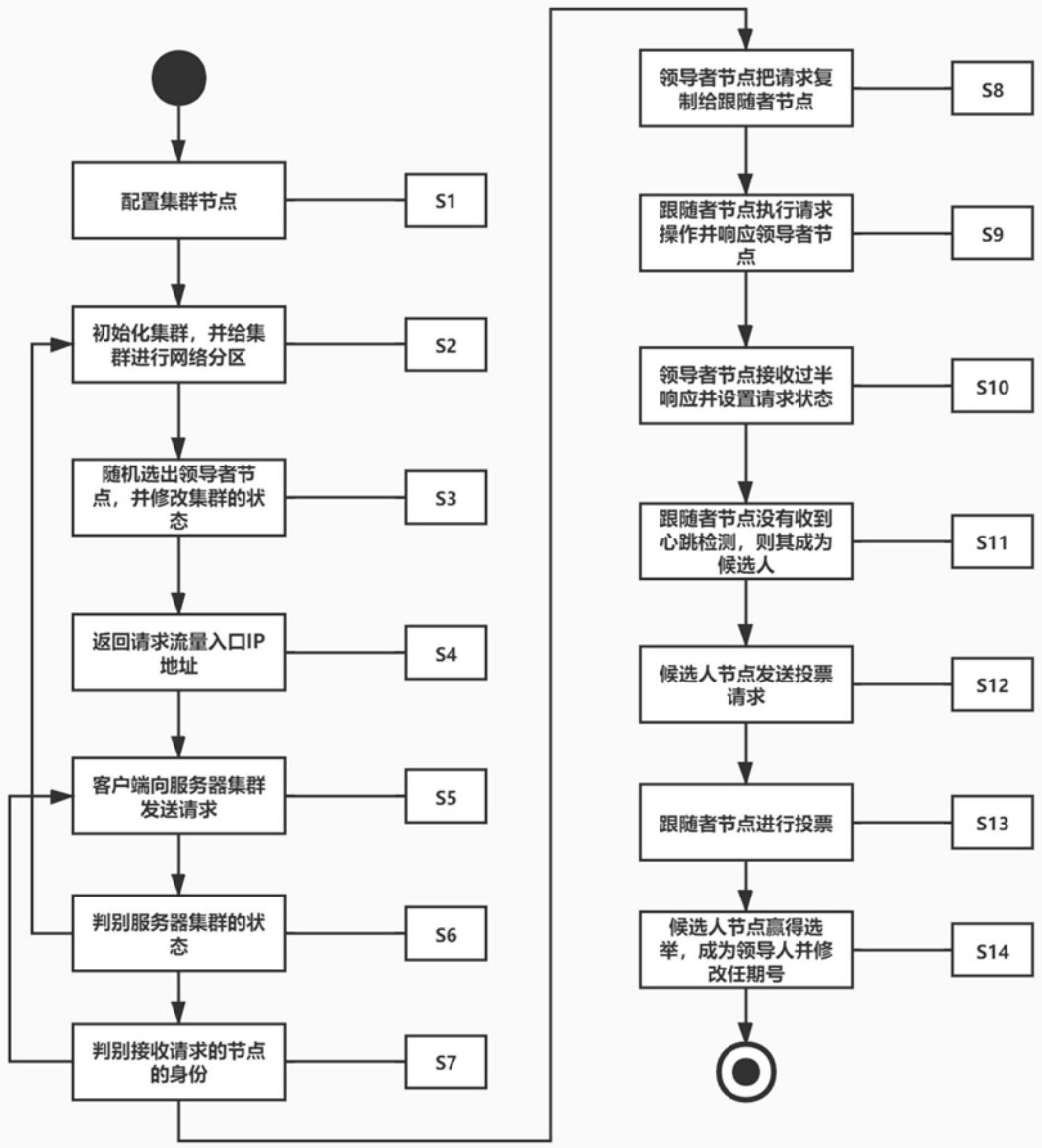

配置集群节点
S1
初始化集群，并给集群进行网络分区
S2
随机选出领导者节点，并修改集群的状态
S3
返回请求流量入口IP地址
S4
客户端向服务器集群发送请求
S5
判别服务器集群的状态
S6
判别接收请求的节点的身份
S7
领导者节点把请求复制给跟随者节点
S8
跟随者节点执行请求操作并响应领导者节点
S9
领导者节点接收过半响应并设置请求状态
S10
跟随者节点没有收到心跳检测，则其成为候选人
S11
候选人节点发送投票请求
S12
跟随者节点进行投票
S13
候选人节点赢得选举，成为领导人并修改任期号
S14

图2