(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 111935320 A
(43)申请公布日 2020.11.13

(21)申请号 202011044182.4

(22)申请日 2020.09.28

(71)申请人 腾讯科技(深圳)有限公司
地址 518057 广东省深圳市南山区高新区科技中一路腾讯大厦35层

(72)发明人 秦凯悦

(74)专利代理机构 深圳市深佳知识产权代理事务所(普通合伙) 44285
代理人 李杭

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

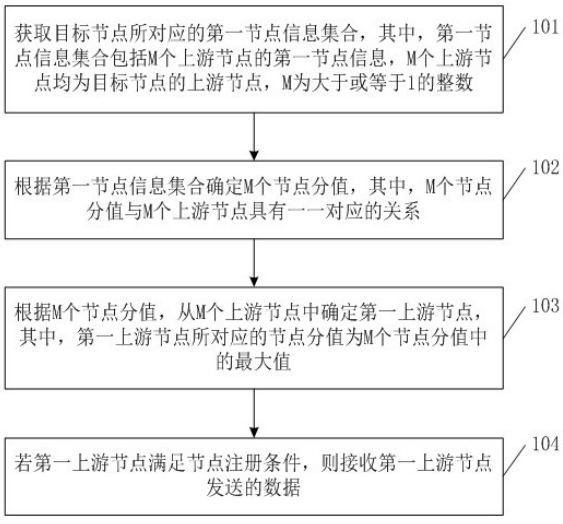
权利要求书3页 说明书28页 附图10页

(54)发明名称

一种数据同步的方法、相关装置、设备以及存储介质

(57)摘要

本申请公开了一种基于云技术实现的数据同步方法，该方法可应用于云存储领域，具体涉及到数据存储以及数据读取。本申请提供的方法包括：获取目标节点所对应的第一节点信息集合；根据第一节点信息集合确定M个节点分；根据M个节点分值，从M个上游节点中确定第一上游节点；若第一上游节点满足节点注册条件，则接收第一上游节点发送的数据。本申请还提供了相关装置、设备以及存储介质。在本申请中，对于learner而言，对每个上游节点进行打分，根据这些上游节点的节点分值，筛选出最优的上游节点，由该上游节点向learner同步数据，从而能够高效地完成数据同步。



CN 111935320 A

1.一种数据同步的方法,其特征在于,包括:

获取目标节点所对应的第一节点信息集合,其中,所述第一节点信息集合包括M个上游节点的第一节点信息,所述M个上游节点均为所述目标节点的上游节点,所述M为大于或等于1的整数;

根据所述第一节点信息集合确定M个节点分值,其中,所述M个节点分值与所述M个上游节点具有一一对应的关系;

根据所述M个节点分值,从所述M个上游节点中确定第一上游节点,其中,所述第一上游节点所对应的节点分值为所述M个节点分值中的最大值;

若所述第一上游节点满足节点注册条件,则接收所述第一上游节点发送的数据。

2.根据权利要求1所述的方法,其特征在于,所述获取目标节点所对应的第一节点信息集合,包括:

根据所述目标节点所对应的第一上游列表信息确定所述M个上游节点;

向所述M个上游节点中的每个上游节点发送信息采集请求,以使所述每个上游节点响应于所述信息采集请求,并获取所述每个上游节点的第一节点信息;

当接收到所述每个上游节点发送的第一节点信息时,获取所述第一节点信息集合。

3.根据权利要求1所述的方法,其特征在于,所述第一节点信息包括上游节点的状态信息以及所述上游节点的关联信息;

所述根据所述第一节点信息集合确定M个节点分值,包括:

针对所述第一节点信息集合中的每个第一节点信息,若所述上游节点的状态信息指示非正常状态,则确定所述上游节点的节点分值为预设值;

针对所述第一节点信息集合中的每个第一节点信息,若所述上游节点的状态信息指示正常状态,则根据所述上游节点的关联信息确定节点分值,其中,所述节点分值大于或等于所述预设值。

4.根据权利要求3所述的方法,其特征在于,所述上游节点的关联信息包括所述上游节点的挂载节点数量、所述上游节点的深度、所述上游节点的负载信息以及所述上游节点的已提交日志信息;

所述根据所述上游节点的关联信息确定节点分值,包括:

根据所述上游节点的挂载节点数量、所述上游节点的深度以及所述上游节点的负载信息,计算分值中间量;

根据所述分值中间量以及所述上游节点的已提交日志信息,计算得到所述上游节点的节点分值,其中,所述节点分值与所述上游节点的已提交日志信息呈正相关,所述节点分值与所述上游节点的挂载节点数量、所述上游节点的深度以及所述上游节点的负载信息均呈负相关。

5.根据权利要求1所述的方法,其特征在于,所述根据所述M个节点分值,从所述M个上游节点中确定第一上游节点,包括:

按照从大到小的次序,对所述M个节点分值进行排序处理,得到排序结果;

根据所述排序结果对第一上游列表信息中的节点进行排序处理,得到第二上游列表信息,其中,所述第二上游列表信息包括M个上游节点;

将所述第二上游列表信息中的首个上游节点确定为所述第一上游节点。

6.根据权利要求1所述的方法，其特征在于，所述根据所述M个节点分值，从所述M个上游节点中确定第一上游节点之后，所述方法还包括：

向所述第一上游节点发送节点注册请求，以使所述第一上游节点根据所述节点注册请求确定可挂载数量；

若所述可挂载数量大于或等于1，则确定所述第一上游节点满足所述节点注册条件；

若所述可挂载数量为0，则确定所述第一上游节点不满足所述节点注册条件。

7.根据权利要求1所述的方法，其特征在于，所述根据所述M个节点分值，从所述M个上游节点中确定第一上游节点之后，所述方法还包括：

向所述第一上游节点发送节点注册请求；

接收所述第一上游节点发送的节点注册响应，其中，所述节点注册响应携带所述第一上游节点的角色信息；

若所述角色信息指示所述第一上游节点为从节点或者学习节点，则确定所述第一上游节点满足所述节点注册条件；

若所述角色信息指示所述第一上游节点为主节点，则确定所述第一上游节点不满足所述节点注册条件。

8.根据权利要求1所述的方法，其特征在于，所述根据所述M个节点分值，从所述M个上游节点中确定第一上游节点之后，所述方法还包括：

若所述第一上游节点不满足所述节点注册条件，则根据所述M个节点分值，从所述M个上游节点中确定第二上游节点，其中，所述第二上游节点所对应的节点分值为所述M个节点分值中的次大值；

若所述第二上游节点满足所述节点注册条件，则接收所述第二上游节点发送的数据。

9.根据权利要求1至8中任意一项所述的方法，其特征在于，所述接收所述第一上游节点发送的数据之后，所述方法还包括：

接收网络设备发送的数据读取请求；

根据所述数据读取请求，向所述网络设备发送元数据信息。

10.根据权利要求1至8中任意一项所述的方法，其特征在于，所述接收所述第一上游节点发送的数据之后，所述方法还包括：

接收网络设备发送的第一数据读取请求；

根据所述第一数据读取请求，向所述网络设备发送第一元数据信息，其中，所述网络设备还用于向除所述目标节点以外的其他节点发送第二数据读取请求，所述第二数据读取请求用于获取第二元数据信息。

11.根据权利要求1至8中任意一项所述的方法，其特征在于，所述接收所述第一上游节点发送的数据，包括：

接收第一网络设备发送的初始查询请求；

根据所述初始查询请求，将所述第一上游节点中的目标元数据信息同步至所述目标节点；

所述接收所述第一上游节点发送的数据之后，所述方法还包括：

接收第二网络设备发送的数据查询请求；

根据所述数据查询请求，向所述第二网络设备发送所述目标元数据信息。

12.根据权利要求1至8中任意一项所述的方法，其特征在于，所述接收所述第一上游节点发送的数据之后，所述方法还包括：

当所述第一上游节点发生故障时，获取目标节点所对应的第二节点信息集合，其中，所述第二节点信息集合包括N个上游节点的第二节点信息，所述N个上游节点均为所述目标节点的上游节点，所述N为大于或等于1的整数；

根据所述第二节点信息集合确定N个节点分值，其中，所述N个节点分值与所述N个上游节点具有一一对应的关系；

根据所述N个节点分值，从所述N个上游节点中确定第三上游节点，其中，所述第三上游节点所对应的节点分值为所述N个节点分值中的最大值；

若所述第三上游节点满足所述节点注册条件，则将所述第三上游节点中的数据同步至所述目标节点。

13.一种数据同步装置，其特征在于，包括：

获取模块，用于获取目标节点所对应的第一节点信息集合，其中，所述第一节点信息集合包括M个上游节点的第一节点信息，所述M个上游节点均为所述目标节点的上游节点，所述M为大于或等于1的整数；

确定模块，用于根据所述第一节点信息集合确定M个节点分值，其中，所述M个节点分值与所述M个上游节点具有一一对应的关系；

所述确定模块，还用于根据所述M个节点分值，从所述M个上游节点中确定第一上游节点，其中，所述第一上游节点所对应的节点分值为所述M个节点分值中的最大值；

同步模块，用于若所述第一上游节点满足节点注册条件，则接收所述第一上游节点发送的数据。

14.一种服务器，其特征在于，包括：存储器、收发器、处理器以及总线系统；

其中，所述存储器用于存储程序；

所述处理器用于执行所述存储器中的程序，所述处理器用于根据程序代码中的指令执行权利要求1至12中任一项所述的方法；

所述总线系统用于连接所述存储器以及所述处理器，以使所述存储器以及所述处理器进行通信。

15.一种计算机可读存储介质，包括指令，当其在计算机上运行时，使得计算机执行如权利要求1至12中任一项所述的方法。

一种数据同步的方法、相关装置、设备以及存储介质

技术领域

[0001] 本申请涉及存储领域，尤其一种数据同步的方法、相关装置、服务器以及存储介质。

背景技术

[0002] 随着当前互联网的发展，云上客户对数据安全越来越重视，大量行业对数据存储有跨机房跨地域的需求。在分布式系统中，为了能够对抗故障情况，数据会被保存为多份副本，由复制组的每个上游节点保存副本。其中，复制组包括主节点(leader)和从节点(follower)，leader保存主副本，follower保存从副本。

[0003] 在数据读取请求量较大时，需要更多的节点来分担这些请求，如果在复制组中加入更多的follower来承担数据读取请求，则会给leader造成更大的负担。为了解决上述问题，目前，加入了一个新的节点角色，即学习节点(learner)，learner能够同步follower或者上游learner的最新数据，以此为数据读取请求提供相应服务。

[0004] 对于learner而言，可随机选择follower作为数据同步的节点，然而，该follower可能无法为learner提供高效的服务，导致数据同步效率较低。

发明内容

[0005] 本申请实施例提供了一种数据同步的方法、相关装置、设备以及存储介质，对于learner而言，对每个上游节点进行打分，根据这些上游节点的节点分值，筛选出最优的上游节点，由该上游节点向learner同步数据，从而能够高效地完成数据同步。

[0006] 有鉴于此，本申请一方面提供一种数据同步的方法，包括：

获取目标节点所对应的第一节点信息集合，其中，第一节点信息集合包括M个上游节点的第一节点信息，M个上游节点均为目标节点的上游节点，M为大于或等于1的整数；

根据第一节点信息集合确定M个节点分值，其中，M个节点分值与M个上游节点具有一一对应的关系；

根据M个节点分值，从M个上游节点中确定第一上游节点，其中，第一上游节点所对应的节点分值为M个节点分值中的最大值；

若第一上游节点满足节点注册条件，则接收第一上游节点发送的数据。

[0007] 在一种可能的设计中，在本申请实施例的一方面的另一种实现方式中，第一节点信息包括上游节点的角色信息以及上游节点的关联信息；

根据第一节点信息集合确定M个节点分值，包括：

针对第一节点信息集合中的每个第一节点信息，若上游节点的角色信息指示上游节点为主节点，则确定上游节点的节点分值为预设值；

针对第一节点信息集合中的每个第一节点信息，若上游节点的状态信息指示上游节点为从节点或者学习节点，则根据上游节点的关联信息确定节点分值，其中，节点分值大于或等于预设值。

[0008] 在一种可能的设计中，在本申请实施例的一方面的另一种实现方式中，第一节点信息包括上游节点的状态信息、角色信息以及上游节点的关联信息；

根据第一节点信息集合确定M个节点分值，包括：

针对第一节点信息集合中的每个第一节点信息，若上游节点的角色信息指示上游节点为主节点，或，上游节点的状态信息指示非正常状态，则确定上游节点的节点分值为预设值；

针对第一节点信息集合中的每个第一节点信息，若上游节点的状态信息指示上游节点为从节点或者学习节点，且，上游节点的状态信息指示正常状态，则根据上游节点的关联信息确定节点分值，其中，节点分值大于或等于预设值。

[0009] 本申请另一方面提供一种数据同步装置，包括：

获取模块，用于获取目标节点所对应的第一节点信息集合，其中，第一节点信息集合包括M个上游节点的第一节点信息，M个上游节点均为目标节点的上游节点，M为大于或等于1的整数；

确定模块，用于根据第一节点信息集合确定M个节点分值，其中，M个节点分值与M个上游节点具有一一对应的关系；

确定模块，还用于根据M个节点分值，从M个上游节点中确定第一上游节点，其中，第一上游节点所对应的节点分值为M个节点分值中的最大值；

同步模块，用于若第一上游节点满足节点注册条件，则接收第一上游节点发送的数据。

[0010] 在一种可能的设计中，在本申请实施例的另一方面的一种实现方式中，

获取模块，具体用于根据目标节点所对应的第一上游列表信息确定M个上游节点；

向M个上游节点中的每个上游节点发送信息采集请求，以使每个上游节点响应于信息采集请求，并获取每个上游节点的第一节点信息；

当接收到每个上游节点发送的第一节点信息时，获取第一节点信息集合。

[0011] 在一种可能的设计中，在本申请实施例的另一方面的另一种实现方式中，第一节点信息包括上游节点的状态信息以及上游节点的关联信息；

确定模块，具体用于针对第一节点信息集合中的每个第一节点信息，若上游节点的状态信息指示非正常状态，则确定上游节点的节点分值为预设值；

针对第一节点信息集合中的每个第一节点信息，若上游节点的状态信息指示正常状态，则根据上游节点的关联信息确定节点分值，其中，节点分值大于或等于预设值。

[0012] 在一种可能的设计中，在本申请实施例的另一方面的另一种实现方式中，第一节点信息包括上游节点的角色信息以及上游节点的关联信息；

确定模块，具体用于针对第一节点信息集合中的每个第一节点信息，若上游节点的角色信息指示上游节点为主节点，则确定上游节点的节点分值为预设值；

针对第一节点信息集合中的每个第一节点信息，若上游节点的状态信息指示上游节点为从节点或者学习节点，则根据上游节点的关联信息确定节点分值，其中，节点分值大于或等于预设值。

[0013] 在一种可能的设计中，在本申请实施例的另一方面的另一种实现方式中，第一节点信息包括上游节点的状态信息、角色信息以及上游节点的关联信息；

确定模块，具体用于针对第一节点信息集合中的每个第一节点信息，若上游节点的角

色信息指示上游节点为主节点，或，上游节点的状态信息指示非正常状态，则确定上游节点的节点分值为预设值；

针对第一节点信息集合中的每个第一节点信息，若上游节点的状态信息指示上游节点为从节点或者学习节点，且，上游节点的状态信息指示正常状态，则根据上游节点的关联信息确定节点分值，其中，节点分值大于或等于预设值。

[0014] 在一种可能的设计中，在本申请实施例的另一方面的另一种实现方式中，上游节点的关联信息包括上游节点的挂载节点数量、上游节点的深度、上游节点的负载信息以及上游节点的已提交日志信息；

确定模块，具体用于根据上游节点的挂载节点数量、上游节点的深度以及上游节点的负载信息，计算分值中间量；

根据分值中间量以及上游节点的已提交日志信息，计算得到上游节点的节点分值，其中，节点分值与上游节点的已提交日志信息呈正相关，节点分值与上游节点的挂载节点数量、上游节点的深度以及上游节点的负载信息均呈负相关。

[0015] 在一种可能的设计中，在本申请实施例的另一方面的另一种实现方式中，

确定模块，具体用于按照从大到小的次序，对M个节点分值进行排序处理，得到排序结果；

根据排序结果对第一上游列表信息中的节点进行排序处理，得到第二上游列表信息，其中，第二上游列表信息包括M个上游节点；

将第二上游列表信息中的首个上游节点确定为第一上游节点。

[0016] 在一种可能的设计中，在本申请实施例的另一方面的另一种实现方式中，数据同步装置还包括发送模块；

发送模块，用于在确定模块根据M个节点分值，从M个上游节点中确定第一上游节点之后，向第一上游节点发送节点注册请求，以使第一上游节点根据节点注册请求确定可挂载数量；

确定模块，还用于若可挂载数量大于或等于1，则确定第一上游节点满足节点注册条件；

确定模块，还用于若可挂载数量为0，则确定第一上游节点不满足节点注册条件。

[0017] 在一种可能的设计中，在本申请实施例的另一方面的另一种实现方式中，数据同步装置还包括发送模块以及接收模块；

发送模块，用于在确定模块根据M个节点分值，从M个上游节点中确定第一上游节点之后，向第一上游节点发送节点注册请求；

接收模块，用于接收第一上游节点发送的节点注册响应，其中，节点注册响应携带第一上游节点的角色信息；

确定模块，还用于若角色信息指示第一上游节点为从节点或者学习节点，则确定第一上游节点满足节点注册条件；

确定模块，还用于若角色信息指示第一上游节点为主节点，则确定第一上游节点不满足节点注册条件。

[0018] 在一种可能的设计中，在本申请实施例的另一方面的另一种实现方式中，

确定模块，还用于根据M个节点分值，从M个上游节点中确定第一上游节点之后，若第一

上游节点不满足节点注册条件，则根据M个节点分值，从M个上游节点中确定第二上游节点，其中，第二上游节点所对应的节点分值为M个节点分值中的次大值；

同步模块，还用于若第二上游节点满足节点注册条件，则接收第二上游节点发送的数据。

[0019] 在一种可能的设计中，在本申请实施例的另一方面的另一种实现方式中，数据同步装置还包括发送模块以及接收模块；

接收模块，还用于在同步模块接收第一上游节点发送的数据之后，接收网络设备发送的数据读取请求；

发送模块，还用于根据数据读取请求，向网络设备发送元数据信息。

[0020] 在一种可能的设计中，在本申请实施例的另一方面的另一种实现方式中，数据同步装置还包括发送模块以及接收模块；

接收模块，还用于在同步模块接收第一上游节点发送的数据之后，接收网络设备发送的第一数据读取请求；

发送模块，还用于根据第一数据读取请求，向网络设备发送第一元数据信息，其中，网络设备还用于向除目标节点以外的其他节点发送第二数据读取请求，第二数据读取请求用于获取第二元数据信息。

[0021] 在一种可能的设计中，在本申请实施例的另一方面的另一种实现方式中，数据同步装置还包括发送模块以及接收模块；

同步模块，具体用于接收第一网络设备发送的初始查询请求；

根据初始查询请求，将第一上游节点中的目标元数据信息同步至目标节点；

接收模块，还用于在同步模块接收第一上游节点发送的数据之后，接收第二网络设备发送的数据查询请求；

发送模块，还用于根据数据查询请求，向第二网络设备发送目标元数据信息。

[0022] 在一种可能的设计中，在本申请实施例的另一方面的另一种实现方式中，

获取模块，还用于在同步模块接收第一上游节点发送的数据之后，当第一上游节点发生故障时，获取目标节点所对应的第二节点信息集合，其中，第二节点信息集合包括N个上游节点的第二节点信息，N个上游节点均为目标节点的上游节点，N为大于或等于1的整数；

确定模块，还用于根据第二节点信息集合确定N个节点分值，其中，N个节点分值与N个上游节点具有一一对应的关系；

确定模块，还用于根据N个节点分值，从N个上游节点中确定第三上游节点，其中，第三上游节点所对应的节点分值为N个节点分值中的最大值；

同步模块，还用于若第三上游节点满足节点注册条件，则将第三上游节点中的数据同步至目标节点。

[0023] 本申请另一方面提供一种计算机设备，包括：存储器、收发器、处理器以及总线系统；

其中，存储器用于存储程序；

处理器用于执行存储器中的程序，处理器用于根据程序代码中的指令执行上述各方面的方法；

总线系统用于连接存储器以及处理器，以使存储器以及处理器进行通信。

[0024] 本申请的另一方面提供了一种计算机可读存储介质，计算机可读存储介质中存储有指令，当其在计算机上运行时，使得计算机执行上述各方面的方法。

[0025] 本申请的另一个方面，提供了一种计算机程序产品或计算机程序，该计算机程序产品或计算机程序包括计算机指令，该计算机指令存储在计算机可读存储介质中。计算机设备的处理器从计算机可读存储介质读取该计算机指令，处理器执行该计算机指令，使得该计算机设备执行上述各方面所提供的方法。

[0026] 从以上技术方案可以看出，本申请实施例具有以下优点：

本申请实施例中，提供了一种数据同步的方法，首先，目标节点获取目标节点所对应的第一节点信息集合，然后根据第一节点信息集合确定M个节点分值，于是目标节点可以继续根据M个节点分值，从M个上游节点中确定第一上游节点，如果第一上游节点满足节点注册条件，则接收第一上游节点发送的数据。通过上述方式，对于learner而言，采用打分机制对每个上游节点进行打分，根据这些上游节点的节点分值，筛选出最优的上游节点，由该上游节点向learner同步数据，从而能够高效地完成数据同步。

附图说明

[0027] 图1为本申请实施例中数据同步系统的一个架构示意图；

图2为本申请实施例中基于动态打分机制实现数据同步的一个实施例示意图；

图3为本申请实施例中数据同步的方法一个实施例示意图；

图4为本申请实施例中目标节点采集第一节点信息的一个示意图；

图5为本申请实施例中目标节点选择用于同步数据的上游节点的一个示意图；

图6为本申请实施例中基于数据读取场景进行数据同步的一个示意图；

图7为本申请实施例中基于数据读取场景进行数据同步的另一个示意图；

图8为本申请实施例中基于数据查询场景进行数据同步的一个示意图；

图9为本申请实施例中动态调整打分策略的一个示意图；

图10为本申请实施例中数据同步装置的一个实施例示意图；

图11为本申请实施例中服务器的一个结构示意图。

具体实施方式

[0028] 本申请实施例提供了一种数据同步的方法、相关装置、设备以及存储介质，对于learner而言，对每个上游节点进行打分，根据这些上游节点的节点分值，筛选出最优的上游节点，由该上游节点向learner同步数据，从而能够高效地完成数据同步。

[0029] 本申请的说明书和权利要求书及上述附图中的术语“第一”、“第二”、“第三”、“第四”等(如果存在)是用于区别类似的对象，而不必用于描述特定的顺序或先后次序。应该理解这样使用的数据在适当情况下可以互换，以便这里描述的本申请的实施例例如能够以除了在这里图示或描述的那些以外的顺序实施。此外，术语“包括”和“对应于”以及他们的任何变形，意图在于覆盖不排他的包含，例如，包含了一系列步骤或单元的过程、方法、系统、产品或设备不必限于清楚地列出的那些步骤或单元，而是可包括没有清楚地列出的或对于这些过程、方法、产品或设备固有的其它步骤或单元。

[0030] 随着互联网企业的高速发展，对于数据存储的要求越来越高，而且模式各异。例

如,对于购物网站而言,涉及大量商品图片,其特点是文件较小,但数量巨大。又例如,对于视频服务网站而言,其后台存储着大量的视频文件,尺寸大多在数十兆到数吉字节不等。这些应用场景都是传统文件系统不能解决的。而分布式文件系统可以将数据存储在物理上分散的多个存储节点上,对这些节点的资源进行统一的管理与分配,并向用户提供文件系统访问接口,解决了本地文件系统在文件大小,文件数量,打开文件数等的限制问题。

[0031] 在分布式系统中,为了能够对抗故障情况,数据会被保存为多份副本,由复制组的每个上游节点保存副本。其中,复制组包括主节点(leader)和从节点(follower)。在数据读取请求量较大时,需要更多的节点来分担这些请求,如果在复制组中加入更多的follower来承担数据读取请求,则会给leader造成更大的负担。为了解决上述问题,目前,加入了一个新的节点角色,即学习节点(learner),learner能够同步follower或者上游learner的最新数据,以此为数据读取请求提供相应服务。

[0032] 基于此,本申请提供了一种基于云技术(Cloud technology)实现的数据同步方法,能够通过对每个上游节点进行打分后,筛选出最优的上游节点,由该上游节点向learner同步数据,从而能够高效地完成数据同步。本申请提供的数据同步方法具体涉及到云存储(cloud storage)技术。下面将分别对云存储和云技术进行介绍,其中,云存储是在云计算概念上延伸和发展出来的一个新的概念,分布式云存储系统 (以下简称存储系统)是指通过集群应用、网格技术以及分布存储文件系统等功能,将网络中大量各种不同类型的存储设备(存储设备也称之为存储节点)通过应用软件或应用接口集合起来协同工作,共同对外提供数据存储和业务访问功能的一个存储系统。

[0033] 目前,存储系统的存储方法为:创建逻辑卷,在创建逻辑卷时,就为每个逻辑卷分配物理存储空间,该物理存储空间可能是某个存储设备或者某几个存储设备的磁盘组成。客户端在某一逻辑卷上存储数据,也就是将数据存储在文件系统上,文件系统将数据分成许多部分,每一部分是一个对象,对象不仅包含数据而且还包含数据标识(ID entity,ID)等额外的信息,文件系统将每个对象分别写入该逻辑卷的物理存储空间,且文件系统会记录每个对象的存储位置信息,从而当客户端请求访问数据时,文件系统能够根据每个对象的存储位置信息让客户端对数据进行访问。存储系统为逻辑卷分配物理存储空间的过程,具体为:按照对存储于逻辑卷的对象的容量估量(该估量往往相对于实际要存储的对象的容量有很大余量)和独立冗余磁盘阵列(Redundant Array of Independent Disk,RAID)的组别,预先将物理存储空间划分成分条,一个逻辑卷可以理解为一个分条,从而为逻辑卷分配了物理存储空间。

[0034] 而云技术是指在广域网或局域网内将硬件、软件、网络等系列资源统一起来,实现数据的计算、储存、处理和共享的一种托管技术。云技术是基于云计算商业模式应用的网络技术、信息技术、整合技术、管理平台技术、应用技术等的总称,可以组成资源池,按需所用,灵活便利。云计算技术将变成重要支撑。技术网络系统的后台服务需要大量的计算、存储资源,如视频网站、图片类网站和更多的门户网站。伴随着互联网行业的高度发展和应用,将来每个物品都有可能存在自己的识别标志,都需要传输到后台系统进行逻辑处理,不同程度级别的数据将会分开处理,各类行业数据皆需要强大的系统后盾支撑,只能通过云计算来实现。

[0035] 为了便于理解,请参阅图1,图1为本申请实施例中数据同步系统的一个架构示意

图，如图所示，数据同步系统可以理解为是分布式系统中的一部分，数据同步系统包括多台服务器以及多个客户端。其中，这些服务器中有些是主设备，有些是从设备，简而言之，部署有主节点(leader)的设备为主设备，图1中的服务器1即为主设备。部署有从节点(follower)的设备为从设备，图1中的服务器2和服务器3均为服务器1的从设备。部署有学习节点(learner)的设备为从设备，图1中的服务器4和服务器5均为服务器3的从设备。客户端既可以部署于终端设备上，也可以部署于应用服务器上，此处不做限定。而服务器上部署的节点数量可以是一个或者多个，本申请以每个服务器部署一个节点为例进行介绍，然而这不应理解为对本申请的限定。

[0036] 在图1所示的数据同步系统中，由leader和follower构成复制组，在复制组中需要保证leader与follower之间的一致性，通常情况下，leader负责读写数据，follower仅负责读取数据。当leader遇到宕机的情况时，复制组可以快速选举出新的leader。learner作为一个新的副本角色，不参与投票，也不加入复制组，learner可以作为follower的下游节点，或者，作为其他learner的下游节点。

[0037] 在图1中，服务器2和服务器3从服务器1中同步数据，服务器4和服务器5可从服务器3中同步数据。客户端1和客户端2分别向服务器2发送数据读取请求，由服务器2分别向客户端1和客户端2反馈相应的元数据信息。客户端3可以向服务器1发送数据读取请求，由服务器1向客户端3反馈相应的元数据信息，而客户端4则是向服务器1发送数据写入请求，其中，数据可以并发写入至服务器1，也可以批量写入至服务器1，在数据写入完成之后，以回调函数的方式通知写入结果。客户端5和客户端6分别向服务器4发送数据读取请求，由服务器4分别向客户端5和客户端6反馈相应的元数据信息。可以理解的是，不同的follower和不同的learner上的数据可能存在延迟，这是因为数据还未完全同步，存在时间差的缘故，但是读取leader的数据是准确的。

[0038] 需要说明的是，本申请涉及的服务器可以是独立的物理服务器，也可以是多个物理服务器构成的服务器集群或者分布式系统，还可以是提供云服务、云数据库、云计算、云函数、云存储、网络服务、云通信、中间件服务、域名服务、安全服务、内容分发网络(Content Delivery Network，CDN)、以及大数据和人工智能平台等基础云计算服务的云服务器。终端设备可以是智能手机、平板电脑、笔记本电脑、掌上电脑、个人电脑、智能电视、智能手表等，但并不局限于此。终端设备以及服务器可以通过有线或无线通信方式进行直接或间接地连接，本申请在此不做限制。服务器和终端设备的数量也不做限制。

[0039] 下面将结合图2，对本申请提供的数据同步流程进行介绍，请参阅图2，图2为本申请实施例中基于动态打分机制实现数据同步的一个实施例示意图，如图所示，对于元数据存储模块为三个节点组成的复制组为例，这三个节点分别为主节点A(leader A)、从节点A(follower A)和从节点B(follower B)。然而需要访问它们的存储节点和接入节点可能非常多，为了避免元数据存储模块无法支持较高的每秒查询率(queries-per-second，QPS)，设计了元数据缓存层，使用简单一致性算法库(Simple Consensus Algorithm Library，SCAL)提供的learner技术从元数据节点同步数据，数据一致性由SCAL保障，元数据存储层可按照需求扩容。基于动态打分机制，使得元数据缓存层选取最优化的上游节点，以此保障提供更高效可靠的读取服务。

[0040] 具体地，下面将分别以新加入的学习节点D(learner D)和学习节点E(learner E)

为例,介绍如何选择最优化的上游节点,对于learner D而言,具体地:

在步骤A1中,learner D收集follower A的节点信息;

在步骤A2中,learner D收集follower B的节点信息;

在步骤A3中,learner D根据收集到的follower A的节点信息和follower B的节点信息,分别对各follower A和follower B进行打分;

在步骤A4中,learner D根据节点分值从高到低,依次向上游节点注册,假设follower B的分数大于follower A的分数,那么learner D向follower B注册,注册成功后learner D即可从follower B上同步数据,并且开始接收follower B的同步日志。

[0041] 对于learner E而言,具体地:

在步骤B1中,learner E收集learner A的节点信息;

在步骤B2中,learner E收集learner B的节点信息;

在步骤B3中,learner E根据收集到的learner A的节点信息和learner B的节点信息,分别对learner A和learner B进行打分;

在步骤B4中,learner E根据节点分值从高到低,依次向上游节点注册,假设learner A的分数大于learner B的分数,那么learner E向learner A注册,注册成功后learner E即可从learner A上同步数据,并且开始接收follower B的同步日志。

[0042] 鉴于本申请涉及到部分专业术语,为了便于更好地理解本申请提供的方案,下面将先对这些专业术语进行介绍。

[0043] 1、简单一致性算法库(Simple Consensus Algorithm Library,SCAL):SCAL是服务器之间用于针对某个问题达成一致所用的方法。

[0044] 2、副本(Replica):为了对抗(磁盘以及单机等)故障,分布式系统中数据会被保存多份,每一份被称为一个副本。

[0045] 3、leader:属于复制组中的主副本。

[0046] 4、follower:属于复制组中的从副本。

[0047] 5、learner:一种新的副本角色,不参与投票,也不加入复制组,可以挂在follower上或者另外的learner上。

[0048] 6、上游和下游:即为同步数据流的两端,其中,拥有最新数据的是上游,追赶数据的是下游。

[0049] 7、深度(depth):表示节点所处深度,其中,leader的depth为0,follower的depth为1,follower的下游learner的depth为2,第一层learner的下游learner的depth为3,以此类推。

[0050] 8、上游列表信息:表示learner的上游节点列表(SourceList)。

[0051] 9、任期(term):每当一个旧的主副本失效就进入一个新的term。通常每个term有选举阶段和对外服务阶段组成,但在某些极端情况下,如果无法选举出新的主副本,那么本term立即结束,进入下一个term。term是副本集的逻辑时钟。

[0052] 10、已提交日志信息(LastCommitId):即已同步的日志编号由term和索引(index)组成,term和index都是单调递增的数字,不会后退。

[0053] 11、节点的状态信息:主要包括未初始化(uninitialized)状态、正常(normal)状态、进行数据备份和恢复数据(recovery)状态、存在问题(error)状态以及正在关闭

(shutting down)状态。

[0054] 结合上述介绍,下面将对本申请中数据同步的方法进行介绍,请参阅图3,本申请实施例中数据同步的方法一个实施例包括:

101、目标节点获取目标节点所对应的第一节点信息集合,其中,第一节点信息集合包括M个上游节点的第一节点信息,M个上游节点均为目标节点的上游节点,M为大于或等于1的整数;

本实施例中,目标节点首先需要获取第一节点信息集合,这里的第一节点信息集合包括M个上游节点的第一节点信息,且M个上游节点均为目标节点的上游节点。其中,目标节点可以为一个新加入的learner,而上游节点可以是其他的learner或者follower,此处不做限定。

[0055] 第一节点信息集合具有M个第一节点信息,每个第一节点信息对应于一个上游节点,例如,上游节点为follower A,则第一节点信息为follower A的节点信息。又例如,上游节点为learner A,则第一节点信息为learner A的节点信息。

[0056] 102、目标节点根据第一节点信息集合确定M个节点分值,其中,M个节点分值与M个上游节点具有一一对应的关系;

本实施例中,目标节点根据第一节点信息集合中的每个第一节点信息,计算得到每个上游节点的节点分值,从而得到M个节点分值。

[0057] 103、目标节点根据M个节点分值,从M个上游节点中确定第一上游节点,其中,第一上游节点所对应的节点分值为M个节点分值中的最大值;

本实施例中,目标节点在获取到每个上游节点对应的节点分值之后,先从中选择一个最大值,并将该节点分值对应的上游节点确定为第一上游节点。假设M个节点分值分别为“0”、“60”和“100”,由此确定M个节点分值的最大值为“100”,于是将节点分值为“100”的上游节点确定为第一上游节点。

[0058] 104、若第一上游节点满足节点注册条件,则目标节点接收第一上游节点发送的数据。

[0059] 本实施例中,由目标节点或者第一上游节点判断是否满足节点注册条件,如果满足节点注册条件,则表示该第一上游节点最适合为目标节点提供待同步的数据,因此,目标节点可以接收第一上游节点发送的数据,并将该数据存储于目标设备的本地,由此实现数据的同步。

[0060] 本申请实施例中,提供了一种数据同步的方法,首先,目标节点获取目标节点所对应的第一节点信息集合,然后根据第一节点信息集合确定M个节点分值,于是目标节点可以继续根据M个节点分值,从M个上游节点中确定第一上游节点,如果第一上游节点满足节点注册条件,则接收第一上游节点发送的数据。通过上述方式,对于learner而言,采用打分机制对每个上游节点进行打分,根据这些上游节点的节点分值,筛选出最优的上游节点,由该上游节点向learner同步数据,从而能够高效地完成数据同步。

[0061] 可选地,在上述图3对应的各个实施例的基础上,本申请实施例提供的另一个可选实施例中,目标节点获取目标节点所对应的第一节点信息集合,具体包括如下步骤:

目标节点根据目标节点所对应的第一上游列表信息确定M个上游节点;

目标节点向M个上游节点中的每个上游节点发送信息采集请求,以使每个上游节点响

应于信息采集请求，并获取每个上游节点的第一节点信息；

当目标节点接收到每个上游节点发送的第一节点信息时，目标节点获取第一节点信息集合。

[0062] 本实施例中，介绍了一种目标节点收集第一节点信息集合的方式。目标节点需要先获取第一上游列表信息。其中，第一上游列表信息是以配置文件的形式预先存储在目标节点中，当目标节点启动时，加载该配置文件，从而得到相应的第一上游列表信息。可以理解的是，目标节点还提供上游列表信息修改的接口，可以通过该接口修改第一上游列表信息中的相关信息。

[0063] 需要说明的第，第一上游列表信息可实时更新，例如，在一个时间段内，目标节点获取到的第一上游列表信息包括follower A和follower B，经过一段时间后，目标节点获取到的第一上游列表信息包括follower A、follower B以及follower C。本申请中，以第一上游列表信息为目标节点当前获取到的列表信息为例进行说明。

[0064] 为了便于理解，下面将结合图4，对目标节点获取第一节点信息的过程进行介绍。请参阅图4，图4为本申请实施例中目标节点采集第一节点信息的一个示意图，示例性地，请参阅图4中的(A)，假设目标节点为learner D，则learner D的第一上游列表信息如表1所示。

[0065] 表1

上游节点	深度
follower A	1
follower B	1

由表1可知，目标节点的M个上游节点分别为follower A和follower B，基于此，在步骤C1中，目标节点向follower A发送信息采集请求，在步骤C2中，follower A响应于该信息采集请求，获取自身的第一节点信息，再向目标节点发送follower A的第一节点信息。类似地，在步骤C3中，目标节点向follower B发送信息采集请求，在步骤C4中，follower B响应于该信息采集请求，获取自身的第一节点信息，再向目标节点发送follower B的第一节点信息。需要说明的是，步骤C1与步骤C3之间没有固定的执行顺序。在目标节点接收到follower A的第一节点信息以及follower B的第一节点信息之后，即目标节点已获取第一节点信息集合。示例性地，请参阅图4中的(B)，假设目标节点为learner E，则learner E的第一上游列表信息如表2所示。

[0066] 表2

上游节点	深度
learner A	2
learner B	2

由表2可知，目标节点的M个上游节点分别为learner A和learner B，基于此，在步骤D1中，目标节点向learner A发送信息采集请求，在步骤D2中，learner A响应于该信息采集请求，获取自身的第一节点信息，再向目标节点发送learner A的第一节点信息。类似地，在步骤D3中，目标节点向learner B发送信息采集请求，在步骤D4中，learner B响应于该信息采集请求，获取自身的第一节点信息，再向目标节点发送learner B的第一节点信息。需要说明的是，步骤D1与步骤D3之间没有固定的执行顺序。在目标节点接收到learner A的第一节

点信息以及learner B的第一节点信息之后，即目标节点已获取第一节点信息集合。

[0067] 其次，本申请实施例中，提供了一种目标节点收集第一节点信息集合的方式，通过上述方式，目标节点可以主动向第一上游列表信息中的各个上游节点请求相应的节点信息，由上游节点根据当前自身的情况下发节点信息，从而能够保证节点信息的实时性，有利于计算得到更准确的节点分值，由此，确保目标节点选取到最优的上游节点，保证同步数据的高效性。

[0068] 可选地，在上述图3对应的各个实施例的基础上，本申请实施例提供的另一个可选实施例中，第一节点信息包括上游节点的状态信息以及上游节点的关联信息；

目标节点根据第一节点信息集合确定M个节点分值，具体包括如下步骤：

针对第一节点信息集合中的每个第一节点信息，若上游节点的状态信息指示非正常状态，则目标节点确定上游节点的节点分值为预设值；

针对第一节点信息集合中的每个第一节点信息，若上游节点的状态信息指示正常状态，则目标节点根据上游节点的关联信息确定节点分值，其中，节点分值大于或等于预设值。

[0069] 本实施例中，介绍了一种基于状态信息确定节点分值的方式。每个上游节点的第一节点信息包括状态信息以及关联信息，其中，状态信息用于指示上游节点处于正常(normal)状态或者非normal状态，非normal状态包含但不限于uninitialized状态、recovery状态、error状态以及shutting down状态。有normal状态的上游节点可以正常同步数据，因此，如果处于非normal状态的上游节点作为目标节点的上游节点，则无法进行后续数据的同步。

[0070] 具体地，假设目标节点有两个上游节点，分别为follower A和follower B，基于follower A的第一节点信息确定该follower A的状态信息，基于follower B的第一节点信息确定该follower B的状态信息。假设follower A的状态信息为“error状态”，则直接将follower A的节点分值设定为预设值。假设follower B的状态信息为“normal状态”，则基于follower B的第一节点信息确定该follower B的关联信息，并且根据关联信息计算follower B的节点分值。上游节点的关联信息用于表示与该上游节点相关的信息，关联信息包含但不仅限于上游节点的挂载节点数量、深度、负载信息以及已提交日志信息等。

[0071] 可选地，在本申请中，还可以设定leader无法为learner提供数据同步服务的情况。基于此，每个上游节点的第一节点信息还可以包括上游节点所对应的角色信息，该角色信息用于上游节点所属的类型，例如，leader、follower或者learner。如果上游节点属于leader，则该上游节点的节点分值为预设值。如果上游节点属于follower或者learner，则根据上游节点的关联信息确定节点分值。

[0072] 需要说明的是，预设值可以为0，也可以为负数，且预设值小于或等于其他节点分值，节点分值的取值范围为uint64_t，即可采用8个字节来表示。

[0073] 其次，本申请实施例中，提供了一种基于状态信息确定节点分值的方式，通过上述方式，对于处于非正常状态的节点而言，无需采用关联信息计算其对应的节点分值，而是直接将预设值作为该节点的节点分值，从而减少了计算量，有利于提升节点分值的计算效率。

[0074] 可选地，在上述图3对应的各个实施例的基础上，本申请实施例提供的另一个可选实施例中，上游节点的关联信息包括上游节点的挂载节点数量、上游节点的深度、上游节点

的负载信息以及上游节点的已提交日志信息；

目标节点根据上游节点的关联信息确定节点分值，具体包括如下步骤：

目标节点根据上游节点的挂载节点数量、上游节点的深度以及上游节点的负载信息，计算分值中间量；

目标节点根据分值中间量以及上游节点的已提交日志信息，计算得到上游节点的节点分值，其中，节点分值与上游节点的已提交日志信息呈正相关，节点分值与上游节点的挂载节点数量、上游节点的深度以及上游节点的负载信息均呈负相关。

[0075] 本实施例中，介绍了一种基于节点关联信息计算节点分值的方式。如果上游节点为正常状态，那么目标节点可根据该上游节点的关联信息计算得到节点分值，其中，每个上游节点的关联信息包括挂载节点数量、深度、负载信息以及已提交日志信息。下面将介绍计算节点分值的方式。

[0076] 具体地，目标节点可以采用如下公式计算得到上游节点的节点分值：

Score =(10000- Depth * 100 - LearnerCount * 10- Load)+ LastCommitId;

其中，Score表示上游节点的节点分值，(10000- Depth * 100 - LearnerCount * 10- Load)表示上游节点的分值中间量，Depth表示上游节点的深度，LearnerCount表示上游节点的挂载节点数量，LastCommitId表示上游节点的已提交日志信息。由此可见，节点分值与上游节点的已提交日志信息呈正相关，节点分值与上游节点的挂载节点数量、上游节点的深度以及上游节点的负载信息均呈负相关。且上游节点的节点分值越高，则表示该上游节点越适合作为数据同步的节点。基于此，下面将分别介绍关联信息的内容以及定义。

[0077] 1、上游节点的深度，表示上游节点所处的层级，层级越深，表示同步到最新数据越慢，因此，节点分值与上游节点的深度呈负相关，即深度越大，节点分值越小。

[0078] 2、上游节点的挂载节点数量，表示上游节点下已经挂载的节点数目，一个上游节点挂载的下游节点越多，也就表示该上游节点承担的压力越大。因此，节点分值与上游节点的挂载节点数量呈负相关，即挂载节点数量越多，节点分值越小。

[0079] 3、上游节点的负载信息，表示上游节点的负载情况，上游节点的负载过高，会影响数据同步的效率。负载信息可表示为QPS，表示上游节点在一秒的时间内处理了多少个请求。因此，节点分值与上游节点的负载信息呈负相关，即负载信息越大，节点分值越小。

[0080] 4、上游节点的已提交日志信息，每一条日志都对应一个连续且递增的日志标识(log id)，这个已提交日志信息越大，表示该上游节点同步到的日志越新。已提交日志信息是一个从0开始连续递增的整数。示例性地，当上游节点为follower时，假设一个复制组中包括节点A、节点B和节点C，其中，节点A为leader A，节点B为follower B，节点C为follower C，假设leader A同步到的log id 为100，follower B同步到的log id为100，follower C同步到的log id 为97，由于大多数节点已同步到的log id为100，那么leader A的已提交日志信息为100，follower B的已提交日志信息为100，follower C的已提交日志信息为97。又假设leader A同步到的log id 为100，follower B同步到的log id 为99，follower C同步到的log id 为93，对于leader A而言，log id 为100是未提交的，因此，leader A的已提交日志信息为99。对于follower B而言，大多数节点已同步到的log id为99，因此，follower B的已提交日志信息为99。对于follower C而言，大多数节点已同步到的log id为93，因此，follower C的已提交日志信息为93。

[0081] 示例性地，当上游节点为learner时，已提交日志信息即为该上游节点最新的log id。因此，节点分值与上游节点的已提交日志信息呈正相关，即已提交日志信息越大，节点分值越大。

[0082] 需要说明的是，关联信息所包括的参数类型仅为一个示意，在实际应用中，关联信息还可以包括响应时间(Reaction Time，RT)、吞吐量(Throughput，TPS)以及并发用户数等。其中，RT是指系统对请求作出响应的时间。TPS是指系统在单位时间内处理请求的数量。可以理解的是，节点分值与RT、TPS以及并发用户数均呈负相关。

[0083] 再次，本申请实施例中，提供了一种基于节点关联信息计算节点分值的方式，通过上述方式，将挂载节点数量、深度、负载信息以及已提交日志信息，共同作为评估节点分值的依据，从不同的维度上反应上游节点的挂载能力，有利于计算得到更准确的节点分值，由此，确保目标节点选取到最优的上游节点，保证同步数据的高效性。

[0084] 可选地，在上述图3对应的各个实施例的基础上，本申请实施例提供的另一个可选实施例中，目标节点根据M个节点分值，从M个上游节点中确定第一上游节点，具体包括如下步骤：

目标节点按照从大到小的次序，对M个节点分值进行排序处理，得到排序结果；

目标节点根据排序结果对第一上游列表信息中的节点进行排序处理，得到第二上游列表信息，其中，第二上游列表信息包括M个上游节点；

目标节点将第二上游列表信息中的首个上游节点确定为第一上游节点。

[0085] 本实施例中，介绍了一种基于节点分值对M个上游节点进行排序的方式。首先，目标节点分别计算得到M个节点分值，然后，目标节点按照从大到小的次序，对M个节点分值进行排序处理，得到排序结果，排序结果用于表示节点分值的出现次序。最后，该目标节点根据排序结果对第一上游列表信息中的节点进行排序处理，得到第二上游列表信息。由此，目标节点在后续的节点注册过程中，直接使用排序好的第二上游列表信息就可以依次查找所需注册的上游节点。

[0086] 需要说明的是，本申请以从大到小的次序对M个节点分值进行排序为例进行介绍，然而，在实际情况下，也可以按照按照从小到大的次序，对M个节点分值进行排序处理，此处不做赘述。

[0087] 具体地，假设目标节点有5个上游节点，分别为follower A、follower B、follower C、follower D和follower E，目标节点的第一上游列表信息如表3所示。

[0088] 表3

上游节点	深度
follower A	1
follower B	1
follower C	1
follower D	1
follower E	1

其中，假设follower A的节点分值为280，follower B节点分值为520，follower C的节点分值为0，follower D的节点分值为440，follower E的节点分值为100。基于此，按照从大到小的次序，对M个节点分值进行排序处理后，得到目标节点的第二上游列表信息如表4所

示。

[0089] 表4

上游节点	深度
follower B	1
follower D	1
follower A	1
follower E	1
follower C	1

目标节点可以直接将第二上游列表信息中的首个上游节点确定为第一上游节点。以表4为例，则follower B为第一上游节点。

[0090] 其次，本申请实施例中，提供了一种基于节点分值对M个上游节点进行排序的方式，通过上述方式，目标节点可以对M个节点分值进行排序处理，从而更便于快速查找对应的上游节点，由此提升同步数据的效率。

[0091] 可选地，在上述图3对应的各个实施例的基础上，本申请实施例提供的另一个可选实施例中，目标节点根据M个节点分值，从M个上游节点中确定第一上游节点之后，还包括如下步骤：

目标节点向第一上游节点发送节点注册请求，以使第一上游节点根据节点注册请求确定可挂载数量；

若可挂载数量大于或等于1，则目标节点确定第一上游节点满足节点注册条件；

若可挂载数量为0，则目标节点确定第一上游节点不满足节点注册条件。

[0092] 本实施例中，介绍了一种基于上游节点的已注册数量判断上游节点是否满足节点注册条件的方式。目标节点按照节点分值对M个上游节点进行排序之后，可得到第二上游列表信息，于是，目标节点遍历第二上游列表信息中的各个上游节点，依次向上游节点注册，直至注册成功，选择出最优的上游节点，并开始从该上游节点接收同步的日志，以此提供高效的数据读取服务。

[0093] 具体地，第二上游列表信息为按照从大到小的次序，对M个节点分值进行排序后得到的，由此，将第二上游列表信息中的首个上游节点作为第一上游节点。基于此，目标节点向第一上游节点发送节点注册请求，第一上游节点根据节点注册请求确定当前的可挂载数量。由于第一上游节点可预先被设置一个最大可挂载数量，因此，第一上游节点根据自身的最大可挂载数量判断当前剩余的可挂载数量。如果当前剩余的可挂载数量大于或等于1，则表示还可以继续挂载新的learner，因此，目标节点确定第一上游节点满足节点注册条件，即目标节点向第一上游节点注册成功。反之，如果当前剩余的可挂载数量等于0，则表示无法继续挂载新的learner，因此，目标节点确定第一上游节点不满足节点注册条件，即目标节点向第一上游节点注册失败，于是，目标节点可从第二上游列表信息中选择次大的节点分值所对应的上游节点，并继续判断该上游节点是否满足节点注册条件，此处不再赘述。

[0094] 为了便于理解，请参阅图5，图5为本申请实施例中目标节点选择用于同步数据的上游节点的一个示意图，如图所示，以目标节点为learner D为例，假设learner D的上游节点为learner A和learner B，具体地；

在步骤E1中，learner D向learner B发送信息采集请求。

[0095] 在步骤E2中，learner D向learner A发送信息采集请求。需要说明的是，步骤E1与步骤E3之间没有固定的执行顺序。

[0096] 在步骤E3中，learner B响应于learner D发送的信息采集请求，获取自身的第一节点信息，再向learner D发送learner B的第一节点信息。

[0097] 在步骤E4中，learner A响应于learner D发送的信息采集请求，获取自身的第一节点信息，再向learner D发送learner A的第一节点信息。需要说明的是，步骤E3与步骤E4之间没有固定的执行顺序。

[0098] 在步骤E5中，learner D根据收集到的learner A的第一节点信息和learner B的第一节点信息，分别对learner A和learner B进行打分，分别得到learner A的节点分值和learner B的节点分值。假设learner A的节点分值大于learner B的节点分值，则learner D优先向learner A发送节点注册请求。

[0099] 在步骤E6中，如果learner A不满足节点注册条件，则learner D需要向learner B请求节点注册。

[0100] 在步骤E7中，learner D向learner B请求节点注册，如果learner B满足节点注册条件，则learner D可以从learner B中同步数据。

[0101] 其次，本申请实施例中，提供了一种基于上游节点的已注册数量判断上游节点是否满足节点注册条件的方式，通过上述方式，还可以限制上游节点的挂载数量，为了避免同一个上游节点挂载的learner数量过多，还可以预先设定最大可挂载数量，从而有利于维持数据同步系统的稳定性和可靠性。

[0102] 可选地，在上述图3对应的各个实施例的基础上，本申请实施例提供的另一个可选实施例中，目标节点根据M个节点分值，从M个上游节点中确定第一上游节点之后，还包括如下步骤：

目标节点向第一上游节点发送节点注册请求；

目标节点接收第一上游节点发送的节点注册响应，其中，节点注册响应携带第一上游节点的角色信息；

若角色信息指示第一上游节点为从节点或者学习节点，则目标节点确定第一上游节点满足节点注册条件；

若角色信息指示第一上游节点为主节点，则目标节点确定第一上游节点不满足节点注册条件。

[0103] 本实施例中，介绍了一种基于角色信息判断上游节点是否满足节点注册条件的方式。目标节点按照节点分值对M个上游节点进行排序之后，可得到第二上游列表信息，于是，目标节点遍历第二上游列表信息中的各个上游节点，依次向上游节点注册，直至注册成功，选择出最优的上游节点，并开始从该上游节点接收同步的日志，以此提供高效的数据读取服务。

[0104] 具体地，第二上游列表信息为按照从大到小的次序，对M个节点分值进行排序后得到的，由此，将第二上游列表信息中的首个上游节点作为第一上游节点。基于此，目标节点向第一上游节点发送节点注册请求，由第一上游节点向目标节点反馈节点注册响应，该节点注册响应中还携带有第一上游节点的角色信息。其中，角色信息有三类，分别为leader、follower和learner，挂载在leader上虽然能够更及时地同步数据，但是会对leader造成较

大的压力，考虑到利大于弊，因此，本申请中，可以预先规定不能将目标节点挂载在leader上。挂载在follower上的数据同步速率较慢，而挂载在learner上的数据同步效率更慢，但是能够避免对leader造成较大压力的情况。

[0105] 因此，目标节点根据第一上游节点的角色信息判断是否为leader，如果第一上游节点不是leader，而是follower或者learner，则目标节点确定第一上游节点满足节点注册条件，即目标节点向第一上游节点注册成功。反之，如果果第一上游节点是leader，则目标节点确定第一上游节点不满足节点注册条件，即目标节点向第一上游节点注册失败，于是，目标节点可从第二上游列表信息中选择次大的节点分值所对应的上游节点，并继续判断该上游节点是否满足节点注册条件，此处不再赘述。

[0106] 需要说明的是，目标节点选择用于同步数据的上游节点的流程如图5所示，可参阅图5以及图5相关的介绍，此处不做赘述。

[0107] 其次，本申请实施例中，提供了一种基于角色信息判断上游节点是否满足节点注册条件的方式，通过上述方式，还可以限制leader为learner提供挂载的情况，由于挂载在leader上会对该leader造成较大的压力，而leader本身就具有较高的负载，因此，不适合作为learner挂载的对象。基于角色信息判断上游节点是否满足节点注册条件，能够避免由leader直接为learner提供同步数据的情况，从而有利于维持数据同步系统的稳定性和可靠性。

[0108] 可选地，在上述图3对应的各个实施例的基础上，本申请实施例提供的另一个可选实施例中，目标节点根据M个节点分值，从M个上游节点中确定第一上游节点之后，还包括如下步骤：

若第一上游节点不满足节点注册条件，则目标节点根据M个节点分值，从M个上游节点中确定第二上游节点，其中，第二上游节点所对应的节点分值为M个节点分值中的次大值；

若第二上游节点满足节点注册条件，则目标节点接收第二上游节点发送的数据。

[0109] 本实施例中，介绍了一种对于上游节点不满足节点注册条件的处理方式。由目标节点或者第一上游节点判断该第一上游节点是否满足节点注册条件，如果第一上游节点满足节点注册条件，则表示该第一上游节点最适合为目标节点提供待同步的数据。反之，如果第一上游节点不满足节点注册条件，则继续从M个节点分值中确定次大值，

具体地，假设M个节点分值分别为“0”、“60”和“100”，由此，确定M个节点分值中的最大值为“100”，次大值为“60”，即最大值为“100”对应的上游节点为第一上游节点，而次大值为“60”对应的上游节点为第二上游节点。在确定第一上游节点不满足节点注册条件的情况下，由目标节点或者第二上游节点判断该第二上游节点是否满足节点注册条件，如果第二上游节点满足节点注册条件，则表示该第二上游节点最适合为目标节点提供待同步的数据。因此，目标节点可以接收第二上游节点发送的数据，并将该数据存储于目标设备的本地，由此实现数据的同步。

[0110] 其次，本申请实施例中，提供了一种对于上游节点不满足节点注册条件的处理方式，通过上述方式，对于learner而言，采用打分机制对每个上游节点进行打分，根据这些上游节点的节点分值，筛选出最优的上游节点，一旦目标节点无法注册到分值最高的上游节点，那么就会依次向节点分值次大的上游节点进行注册，直至完成注册。从而在保证方案可行性的同时，还可以高效地完成数据同步。

[0111] 可选地，在上述图3对应的各个实施例的基础上，本申请实施例提供的另一个可选实施例中，目标节点接收第一上游节点发送的数据之后，还包括如下步骤：

目标节点接收网络设备发送的数据读取请求；

目标节点根据数据读取请求，向网络设备发送元数据信息。

[0112] 本实施例中，介绍了一种在数据读取场景下实现数据读取的方式，网络设备通过向目标节点发送数据读取请求，实现数据的读取。基于SCAL设计的learner功能，可创建元数据存储模块的完整只读副本，该只读副本从上游节点的元数据存储模块同步到写入的元数据信息，并对外提供数据读取服务，以降低复制组的压力。

[0113] 具体地，网络设备可以直接从某个learner中读取数据，为了便于介绍，本申请以网络设备从目标节点中读取数据为例进行介绍，然而，这不应理解为对本申请的限定。请参阅图6，图6为本申请实施例中基于数据读取场景进行数据同步的一个示意图，如图所示，以复制组中包括三个元数据存储模块为例，其中，一个元数据存储模块为leader，另外两个元数据存储模块为follower，元数据缓存模块为learner。在启动选取上游节点的阶段，当上游节点出现故障或无法及时同步到元数据信息时，需要重新选择上游节点进行注册。以元数据缓存模块启动阶段选举上游节点为例。

[0114] 首先，在启动阶段，会配置目标节点可选的上游节点。然后，从其配置的上游列表信息中选取到最优的上游节点(即更利于及时同步到元数据信息)，并完成注册。此时，上游的节点元数据存储模块会知道其下游的元数据缓存模块，在元数据存储模块接收到数据写入请求之后，会进行复制组之间的同步。当数据同步到该数据缓存模块的上游后，上游会将数据同步到下游，此时，网络设备可以通过元数据缓存模块读取相应的元数据信息。

[0115] 本申请基于动态打分机制的上游节点选拔策略，能够保证选取到最利于同步数据的上游节点，以保证元数据信息同步的及时和高效性，提供更好的读服务。需要说明的是，网络设备可以是终端设备，也可以是服务器，此处不做限定。

[0116] 进一步地，本申请实施例中，提供了一种在数据读取场景下实现数据读取的方式，通过上述方式，能够在大规模数据读取的场景下，提供更高效可靠的读取服务，从而提升方案的可行性。

[0117] 可选地，在上述图3对应的各个实施例的基础上，本申请实施例提供的另一个可选实施例中，目标节点接收第一上游节点发送的数据之后，还包括如下步骤：

目标节点接收网络设备发送的第一数据读取请求；

目标节点根据第一数据读取请求，向网络设备发送第一元数据信息，其中，网络设备还用于向除目标节点以外的其他节点发送第二数据读取请求，第二数据读取请求用于获取第二元数据信息。

[0118] 本实施例中，介绍了另一种在数据读取场景下实现数据读取的方式，网络设备通过向目标节点发送数据读取请求，实现数据的读取。基于SCAL涉及的learner功能，可创建元数据存储模块的完整只读副本，该只读副本从上游节点的元数据存储模块同步到写入的元数据信息，并对外提供数据读取服务，以降低复制组的压力。

[0119] 具体地，网络设备可以直接从多个learner中读取数据，为了便于介绍，本申请以网络设备从两个learner中读取数据为例进行介绍，且其中一个learner为目标节点，然而，这不应理解为对本申请的限定。请参阅图7，图7为本申请实施例中基于数据读取场景进行

数据同步的另一个示意图，如图所示，以复制组中包括三个元数据存储模块为例，其中，一个元数据存储模块为leader，另外两个元数据存储模块为follower，元数据缓存模块为learner。在启动选取上游节点的阶段，当上游节点出现故障或无法及时同步到元数据信息时，需要重新选举上游节点。以元数据缓存模块启动阶段选举上游节点为例。

[0120] 首先，在启动阶段，会配置目标节点可选的上游节点。然后，从其配置的上游列表信息中选取到最优的上游节点(即更利于及时同步到元数据信息)，并完成注册。此时，上游的节点元数据存储模块会知道其下游的元数据缓存模块，在元数据存储模块接收到数据写入请求之后，会进行复制组之间的同步。当数据同步到该数据缓存模块的上游后，上游会将数据同步到下游。

[0121] 示例性地，网络设备可以向learner1发送第一数据读取请求，假设该learner为目标节点，即该目标节点向网络设备反馈第一元数据信息。如果网络设备获取到的第一元数据信息不完整，则该网络设备还可以向一个learner2发送第二数据读取请求，由另一个learner向网络设备反馈第二元数据信息，由此，网络设备获取到第一元数据信息和第二元数据信息。

[0122] 示例性地，网络设备可以同时向learner1发送第一数据读取请求，并且向learner2发送第二数据读取请求，由learner1基于第一数据读取请求向网络设备反馈第一元数据信息，由learner2基于第二数据读取请求向网络设备反馈第二元数据信息。需要说明的是，本实施例以2个learner向网络设备提供元数据信息为例进行说明，然而，在实际应用中，还可以由3个或3个以上的learner向同一个网络设备提供元数据信息，不应理解为对本申请的限定。

[0123] 本申请基于动态打分机制的上游节点选拔策略，能够保证选取到最利于同步数据的上游节点，以保证元数据信息同步的及时和高效性，提供更好的读服务。需要说明的是，网络设备可以是终端设备，也可以是服务器，此处不做限定。

[0124] 进一步地，本申请实施例中，提供了另一种在数据读取场景下实现数据读取的方式，通过上述方式，能够在大规模数据读取的场景下，提供更高效可靠的读取服务，从而提升方案的可行性。与此同时，将一个数据读取请求拆分为多个数据读取子请求，分散数据的读取量，即分别从不同的learner读取数据，从而提升数据读取效率，降低对learner的负荷。

[0125] 可选地，在上述图3对应的各个实施例的基础上，本申请实施例提供的另一个可选实施例中，目标节点接收第一上游节点发送的数据，具体包括如下步骤：

目标节点接收第一网络设备发送的初始查询请求；

目标节点根据初始查询请求，将第一上游节点中的目标元数据信息同步至目标节点；

目标节点接收第一上游节点发送的数据之后，还包括如下步骤：

目标节点接收第二网络设备发送的数据查询请求；

目标节点根据数据查询请求，向第二网络设备发送目标元数据信息。

[0126] 本实施例中，介绍了一种在数据查询场景下实现数据查询的方式，网络设备通过向目标节点发送数据读取请求，实现数据的读取。基于SCAL涉及的learner功能，可创建元数据存储模块的完整只读副本，该只读副本从上游节点的元数据存储模块同步到写入的元数据信息，并对外提供数据读取服务，以降低复制组的压力。

[0127] 具体地，网络设备可以直接从某个learner中查询数据，为了便于介绍，本申请以网络设备从目标节点中查询数据为例进行介绍，然而，这不应理解为对本申请的限定。请参阅图8，图8为本申请实施例中基于数据查询场景进行数据同步的一个示意图，如图所示，以复制组中包括三个元数据存储模块为例，其中，一个元数据存储模块为leader，另外两个元数据存储模块为follower，元数据缓存模块为learner。在启动选取上游节点的阶段，当上游节点出现故障或无法及时同步到元数据信息时，需要重新选择上游节点。以元数据缓存模块启动阶段选举上游节点为例。

[0128] 对于需要频繁获取元数据信息的元数据缓存模块而言，如果通过查询请求获取数据，可能会返回上吉字节(Gigabyte，GB)的数据，因此，可以将该元数据缓存模块作为元数据存储模块的目标节点(例如，learner1)，实时同步到写入的元数据信息。例如，首先，在启动阶段，会配置目标节点可选的上游节点。然后，从其配置的上游列表信息中选取到最优的上游节点(即更利于及时同步到元数据信息)，并完成注册。此时，上游的节点元数据存储模块会知道其下游的元数据缓存模块，在元数据存储模块接收到数据写入请求之后，会进行复制组之间的同步。当第一网络设备向目标节点发送初始查询请求时，目标节点根据初始查询请求，将第一上游节点(例如，follower1)中的目标元数据信息同步至目标节点。之后，当其他网络设备(例如，第二网络设备)还需要查询目标元数据信息时，直接目标节点发送数据查询请求，由目标节点根据数据查询请求，直接向第二网络设备发送该目标元数据信息，相当于仅访问了目标节点的内存，有利于提升数据查询效率。

[0129] 本申请基于动态打分机制的上游节点选拔策略，能够保证选取到最利于同步数据的上游节点，以保证元数据信息同步的及时和高效性，提供更好的读服务。需要说明的是，网络设备可以是终端设备，也可以是服务器，此处不做限定。

[0130] 进一步地，本申请实施例中，提供了一种在数据查询场景下实现数据查询的方式，通过上述方式，对于需要频繁获取目标元数据信息的网络设备而言，可以预先将目标元数据信息存储在目标节点的本地，当网络设备再次请求目标元数据信息时，仅需要访问目标节点的内存即可，无需目标节点再次向上游节点请求相关的数据，从而提升数据查询效率。

[0131] 可选地，在上述图3对应的各个实施例的基础上，本申请实施例提供的另一个可选实施例中，目标节点接收第一上游节点发送的数据之后，还包括如下步骤：

当第一上游节点发生故障时，目标节点获取目标节点所对应的第二节点信息集合，其中，第二节点信息集合包括N个上游节点的第二节点信息，N个上游节点均为目标节点的上游节点，N为大于或等于1的整数；

目标节点根据第二节点信息集合确定N个节点分值，其中，N个节点分值与N个上游节点具有一一对应的关系；

目标节点根据N个节点分值，从N个上游节点中确定第三上游节点，其中，第三上游节点所对应的节点分值为N个节点分值中的最大值；

若第三上游节点满足节点注册条件，则目标节点将第三上游节点中的数据同步至目标节点。

[0132] 本实施例中，介绍了一种基于动态打分策略调整节点分值的方式。为了使目标节点能够在不同时间段内选择到最优的上游节点，还需要更新上游节点的节点分值。在本申请中，如果需要同步日志，则上游节点会发送日志，如果不需要同步日志，则上游节点会每

间隔一小段时间(如1秒)向下游节点发送心跳信号。如果下游节点在一段时间(如10秒)内没有收到上游节点发送的心跳信号或日志,则表示上游节点可能出现了故障,那么下游节点会自动发起重新计算各个上游节点的节点分值的流程,并向各个上游节点注册。

[0133] 具体地,当第一上游节点发生故障时,目标节点需要获取第二节点信息集合,其中,第二节点信息集合包括N个上游节点的第二节点信息。示例性地,在一种情况下,如果目标节点不更新第一上游列表信息,则目标节点还会向N个上游节点发送信息采集请求,其中,N等于M。这N个上游节点分别获取第二节点信息,并向目标节点反馈第二上游信息。当目标节点接收到N个上游节点发送的第二节点信息时,即得到第二节点信息集合。示例性地,在另一种情况下,如果目标节点更新第一上游列表信息,则目标节点根据更新后的上游列表信息向N个上游节点发送信息采集请求,其中,N与M可能相等,也可能不等。这N个上游节点分别获取第二节点信息,并向目标节点反馈第二上游信息。当目标节点接收到N个上游节点发送的第二节点信息时,即得到第二节点信息集合。

[0134] 基于此,目标节点根据第二节点信息集合中的每个第二节点信息,重新计算得到每个上游节点的节点分值,从而得到N个节点分值。目标节点在获取到每个上游节点对应的节点分值之后,先从中选择一个最大值,并将该节点分值对应的上游节点确定为第三上游节点。假设N个节点分值分别为“0”、“80”和“150”,由此确定N个节点分值的最大值为“150”,于是将节点分值为“150”的上游节点确定为第三上游节点。由目标节点或者第三上游节点判断是否满足节点注册条件,如果第三上游节点满足节点注册条件,则表示该第三上游节点最适合为目标节点提供待同步的数据,因此,目标节点可以接收第三上游节点发送的数据,并将该数据存储于目标设备的本地,由此实现数据的同步。

[0135] 为了便于理解,请参阅图9,图9为本申请实施例中动态调整打分策略的一个示意图,如图所示,假设目标节点为learner D,目标节点具有两个上游节点,分别为learner A和learner B,具体地:

在步骤F1中,learner D计算得到learner B发送信息采集请求,并收集到learner B的第一节点信息。

[0136] 在步骤F2中,learner D计算得到learner A发送信息采集请求,并收集到learner A的第一节点信息。

[0137] 在步骤F3中,learner D根据收集到的learner A的第一节点信息和learner B的第一节点信息,分别对learner A和learner B进行打分。

[0138] 在步骤F4中,learner D根据节点分值从高到低的顺序,依次向上游节点注册,假设learner A的分数大于learner B的分数,那么learner D向learner A注册,注册成功后learner D即可从learner A上同步数据,并且开始接收learner A的同步日志。

[0139] 基于此,在第一时间内,learner A即为第一上游节点,若当前满足分值更新条件,则learner D还可以获取第二节点信息集合,此时,第二节点信息集合所包括的第二节点信息可能与第一节点信息集合所包括的第一节点信息不一致,例如,learner A的第一节点信息所包括的挂载节点数量为2,而learner A的第而节点信息所包括的挂载节点数量为3。具体地:

在步骤G1中,learner D计算得到learner B发送信息采集请求,并收集到learner B的第二节点信息。

[0140] 在步骤G2中，learner D计算得到learner A发送信息采集请求，并收集到learner A的第二节点信息。

[0141] 在步骤G3中，learner D根据收集到的learner A的第二节点信息和learner B的第二节点信息，分别对learner A和learner B进行打分。

[0142] 在步骤G4中，learner D根据节点分值从高到低的顺序，依次向上游节点注册，假设learner B的分数大于learner A的分数，那么learner D向learner B注册，注册成功后learner D即可从learner B上同步数据，并且开始接收learner B的同步日志。

[0143] 基于此，在第二时间内，learner B即为第三上游节点。

[0144] 进一步地，本申请实施例中，提供了一种基于动态打分策略调整节点分值的方式，通过上述方式，目标节点可以周期性地调整根据上游节点的状态信息、角色信息以及关联信息来调整相应的节点分值，避免出现所选上游节点在一段时间之后不适合继续提供数据同步服务的情况，从而增加方案的可行性和可操作性，有利于提供更高效的数据同步服务。

[0145] 下面对本申请中的数据同步装置进行详细描述，请参阅图10，图10为本申请实施例中数据同步装置的一个实施例示意图，数据同步装置20包括：

获取模块201，用于获取目标节点所对应的第一节点信息集合，其中，第一节点信息集合包括M个上游节点的第一节点信息，M个上游节点均为目标节点的上游节点，M为大于或等于1的整数；

确定模块202，用于根据第一节点信息集合确定M个节点分值，其中，M个节点分值与M个上游节点具有一一对应的关系；

确定模块202，还用于根据M个节点分值，从M个上游节点中确定第一上游节点，其中，第一上游节点所对应的节点分值为M个节点分值中的最大值；

同步模块203，用于若第一上游节点满足节点注册条件，则接收第一上游节点发送的数据。

[0146] 本申请实施例中，提供了一种数据同步装置，首先，目标节点获取目标节点所对应的第一节点信息集合，然后根据第一节点信息集合确定M个节点分值，于是目标节点可以继续根据M个节点分值，从M个上游节点中确定第一上游节点，如果第一上游节点满足节点注册条件，则接收第一上游节点发送的数据。采用上述装置，对于learner而言，采用打分机制对每个上游节点进行打分，根据这些上游节点的节点分值，筛选出最优的上游节点，由该上游节点向learner同步数据，从而能够高效地完成数据同步。

[0147] 可选地，在上述图10所对应的实施例的基础上，本申请实施例提供的数据同步装置20的另一实施例中，

获取模块201，具体用于根据目标节点所对应的第一上游列表信息确定M个上游节点；

向M个上游节点中的每个上游节点发送信息采集请求，以使每个上游节点响应于信息采集请求，并获取每个上游节点的第一节点信息；

当接收到每个上游节点发送的第一节点信息时，获取第一节点信息集合。

[0148] 本申请实施例中，提供了一种数据同步装置，采用上述装置，目标节点可以主动向第一上游列表信息中的各个上游节点请求相应的节点信息，由上游节点根据当前自身的情况下发节点信息，从而能够保证节点信息的实时性，有利于计算得到更准确的节点分值，由此，确保目标节点选取到最优的上游节点，保证同步数据的高效性。

[0149] 可选地，在上述图10所对应的实施例的基础上，本申请实施例提供的数据同步装置20的另一实施例中，第一节点信息包括上游节点的状态信息以及上游节点的关联信息；

确定模块202，具体用于针对第一节点信息集合中的每个第一节点信息，若上游节点的状态信息指示非正常状态，则确定上游节点的节点分值为预设值；

针对第一节点信息集合中的每个第一节点信息，若上游节点的状态信息指示正常状态，则根据上游节点的关联信息确定节点分值，其中，节点分值大于或等于预设值。

[0150] 本申请实施例中，提供了一种数据同步装置，采用上述装置，对于处于非正常状态的节点而言，无需采用关联信息计算其对应的节点分值，而是直接将预设值作为该节点的节点分值，从而减少了计算量，有利于提升节点分值的计算效率。

[0151] 可选地，在上述图10所对应的实施例的基础上，本申请实施例提供的数据同步装置20的另一实施例中，第一节点信息包括上游节点的角色信息以及上游节点的关联信息；

确定模块202，具体用于针对第一节点信息集合中的每个第一节点信息，若上游节点的角色信息指示上游节点为主节点，则确定上游节点的节点分值为预设值；

针对第一节点信息集合中的每个第一节点信息，若上游节点的状态信息指示上游节点为从节点或者学习节点，则根据上游节点的关联信息确定节点分值，其中，节点分值大于或等于预设值。

[0152] 本申请实施例中，提供了一种数据同步装置，采用上述装置，对于上游节点为主节点的情况而言，无需采用关联信息计算其对应的节点分值，而是直接将预设值作为该节点的节点分值，从而减少了计算量，有利于提升节点分值的计算效率。

[0153] 可选地，在上述图10所对应的实施例的基础上，本申请实施例提供的数据同步装置20的另一实施例中，第一节点信息包括上游节点的状态信息、角色信息以及上游节点的关联信息；

确定模块202，具体用于针对第一节点信息集合中的每个第一节点信息，若上游节点的角色信息指示上游节点为主节点，或，上游节点的状态信息指示非正常状态，则确定上游节点的节点分值为预设值；

针对第一节点信息集合中的每个第一节点信息，若上游节点的状态信息指示上游节点为从节点或者学习节点，且，上游节点的状态信息指示正常状态，则根据上游节点的关联信息确定节点分值，其中，节点分值大于或等于预设值。

[0154] 本申请实施例中，提供了一种数据同步装置，采用上述装置，对于处于非正常状态的节点而言，或者，对于上游节点为主节点的情况而言，无需采用关联信息计算其对应的节点分值，而是直接将预设值作为该节点的节点分值，从而减少了计算量，有利于提升节点分值的计算效率。

[0155] 可选地，在上述图10所对应的实施例的基础上，本申请实施例提供的数据同步装置20的另一实施例中，上游节点的关联信息包括上游节点的挂载节点数量、上游节点的深度、上游节点的负载信息以及上游节点的已提交日志信息；

确定模块202，具体用于根据上游节点的挂载节点数量、上游节点的深度以及上游节点的负载信息，计算分值中间量；

根据分值中间量以及上游节点的已提交日志信息，计算得到上游节点的节点分值，其中，节点分值与上游节点的已提交日志信息呈正相关，节点分值与上游节点的挂载节点数

量、上游节点的深度以及上游节点的负载信息均呈负相关。

[0156] 本申请实施例中，提供了一种数据同步装置，采用上述装置，将挂载节点数量、深度、负载信息以及已提交日志信息，共同作为评估节点分值的依据，从不同的维度上反应上游节点的挂载能力，有利于计算得到更准确的节点分值，由此，确保目标节点选取到最优的上游节点，保证同步数据的高效性。

[0157] 可选地，在上述图10所对应的实施例的基础上，本申请实施例提供的数据同步装置20的另一实施例中，

确定模块202，具体用于按照从大到小的次序，对M个节点分值进行排序处理，得到排序结果；

根据排序结果对第一上游列表信息中的节点进行排序处理，得到第二上游列表信息，其中，第二上游列表信息包括M个上游节点；

将第二上游列表信息中的首个上游节点确定为第一上游节点。

[0158] 本申请实施例中，提供了一种数据同步装置，采用上述装置，目标节点可以对M个节点分值进行排序处理，从而更便于快速查找对应的上游节点，由此提升同步数据的效率。

[0159] 可选地，在上述图10所对应的实施例的基础上，本申请实施例提供的数据同步装置20的另一实施例中，数据同步装置20还包括发送模块204；

发送模块204，用于在确定模块根据M个节点分值，从M个上游节点中确定第一上游节点之后，向第一上游节点发送节点注册请求，以使第一上游节点根据节点注册请求确定可挂载数量；

确定模块202，还用于若可挂载数量大于或等于1，则确定第一上游节点满足节点注册条件；

确定模块202，还用于若可挂载数量为0，则确定第一上游节点不满足节点注册条件。

[0160] 本申请实施例中，提供了一种数据同步装置，采用上述装置，还可以限制上游节点的挂载数量，为了避免同一个上游节点挂载的learner数量过多，还可以预先设定最大可挂载数量，从而有利于维持数据同步系统的稳定性和可靠性。

[0161] 可选地，在上述图10所对应的实施例的基础上，本申请实施例提供的数据同步装置20的另一实施例中，数据同步装置20还包括发送模块204以及接收模块205；

发送模块204，用于在确定模块202根据M个节点分值，从M个上游节点中确定第一上游节点之后，向第一上游节点发送节点注册请求；

接收模块205，用于接收第一上游节点发送的节点注册响应，其中，节点注册响应携带第一上游节点的角色信息；

确定模块202，还用于若角色信息指示第一上游节点为从节点或者学习节点，则确定第一上游节点满足节点注册条件；

确定模块202，还用于若角色信息指示第一上游节点为主节点，则确定第一上游节点不满足节点注册条件。

[0162] 本申请实施例中，提供了一种数据同步装置，采用上述装置，还可以限制leader为learner提供挂载的情况，由于挂载在leader上会对该leader造成较大的压力，而leader本身就具有较高的负载，因此，不适合作为learner挂载的对象。基于角色信息判断上游节点是否满足节点注册条件，能够避免由leader直接为learner提供同步数据的情况，从而有利

于维持数据同步系统的稳定性和可靠性。

[0163] 可选地，在上述图10所对应的实施例的基础上，本申请实施例提供的数据同步装置20的另一实施例中，

确定模块202，还用于根据M个节点分值，从M个上游节点中确定第一上游节点之后，若第一上游节点不满足节点注册条件，则根据M个节点分值，从M个上游节点中确定第二上游节点，其中，第二上游节点所对应的节点分值为M个节点分值中的次大值；

同步模块203，还用于若第二上游节点满足节点注册条件，则接收第二上游节点发送的数据。

[0164] 本申请实施例中，提供了一种数据同步装置，采用上述装置，对于learner而言，采用打分机制对每个上游节点进行打分，根据这些上游节点的节点分值，筛选出最优的上游节点，一旦目标节点无法注册到分值最高的上游节点，那么就会依次向节点分值次大的上游节点进行注册，直至完成注册。从而在保证方案可行性的同时，还可以高效地完成数据同步。

[0165] 可选地，在上述图10所对应的实施例的基础上，本申请实施例提供的数据同步装置20的另一实施例中，数据同步装置20还包括发送模块204以及接收模块205；

接收模块205，还用于在同步模块203接收第一上游节点发送的数据之后，接收网络设备发送的数据读取请求；

发送模块204，还用于根据数据读取请求，向网络设备发送元数据信息。

[0166] 本申请实施例中，提供了一种数据同步装置，采用上述装置，能够在大规模数据读取的场景下，提供更高效可靠的读取服务，从而提升方案的可行性。

[0167] 可选地，在上述图10所对应的实施例的基础上，本申请实施例提供的数据同步装置20的另一实施例中，数据同步装置20还包括发送模块204以及接收模块205；

接收模块205，还用于在同步模块203接收第一上游节点发送的数据之后，接收网络设备发送的第一数据读取请求；

发送模块204，还用于根据第一数据读取请求，向网络设备发送第一元数据信息，其中，网络设备还用于向除目标节点以外的其他节点发送第二数据读取请求，第二数据读取请求用于获取第二元数据信息。

[0168] 本申请实施例中，提供了一种数据同步装置，采用上述装置，能够在大规模数据读取的场景下，提供更高效可靠的读取服务，从而提升方案的可行性。与此同时，将一个数据读取请求拆分为多个数据读取子请求，分散数据的读取量，即分别从不同的learner读取数据，从而提升数据读取效率，降低对learner的负荷。

[0169] 可选地，在上述图10所对应的实施例的基础上，本申请实施例提供的数据同步装置20的另一实施例中，数据同步装置20还包括发送模块204以及接收模块205；

同步模块203，具体用于接收第一网络设备发送的初始查询请求；

根据初始查询请求，将第一上游节点中的目标元数据信息同步至目标节点；

接收模块205，还用于在同步模块203接收第一上游节点发送的数据之后，接收第二网络设备发送的数据查询请求；

发送模块204，还用于根据数据查询请求，向第二网络设备发送目标元数据信息。

[0170] 本申请实施例中，提供了一种数据同步装置，采用上述装置，对于需要频繁获取目

标元数据信息的网络设备而言，可以预先将目标元数据信息存储在目标节点的本地，当网络设备再次请求目标元数据信息时，仅需要访问目标节点的内存即可，无需目标节点再次向上游节点请求相关的数据，从而提升数据查询效率。

[0171] 可选地，在上述图10所对应的实施例的基础上，本申请实施例提供的数据同步装置20的另一实施例中，

获取模块201，还用于在同步模块203接收第一上游节点发送的数据之后，当第一上游节点发生故障时，获取目标节点所对应的第二节点信息集合，其中，第二节点信息集合包括N个上游节点的第二节点信息，N个上游节点均为目标节点的上游节点，N为大于或等于1的整数；

确定模块202，还用于根据第二节点信息集合确定N个节点分值，其中，N个节点分值与N个上游节点具有一一对应的关系；

确定模块202，还用于根据N个节点分值，从N个上游节点中确定第三上游节点，其中，第三上游节点所对应的节点分值为N个节点分值中的最大值；

同步模块203，还用于若第三上游节点满足节点注册条件，则将第三上游节点中的数据同步至目标节点。

[0172] 本申请实施例中，提供了一种数据同步装置，采用上述装置，目标节点可以周期性地调整根据上游节点的状态信息、角色信息以及关联信息来调整相应的节点分值，避免出现所选上游节点在一段时间之后不适合继续提供数据同步服务的情况，从而增加方案的可行性和可操作性，有利于提供更高效的数据同步服务。

[0173] 图11是本申请实施例提供的一种服务器结构示意图，该服务器300可因配置或性能不同而产生比较大的差异，可以包括一个或一个以上中央处理器(central processing units，CPU)322(例如，一个或一个以上处理器)和存储器332，一个或一个以上存储应用程序342或数据344的存储介质330(例如一个或一个以上海量存储设备)。其中，存储器332和存储介质330可以是短暂存储或持久存储。存储在存储介质330的程序可以包括一个或一个以上模块(图示没标出)，每个模块可以包括对服务器中的一系列指令操作。更进一步地，中央处理器322可以设置为与存储介质330通信，在服务器300上执行存储介质330中的一系列指令操作。

[0174] 服务器300还可以包括一个或一个以上电源326，一个或一个以上有线或无线网络接口350，一个或一个以上输入输出接口358，和/或，一个或一个以上操作系统341，例如Windows Server™，Mac OS X™，Unix™，Linux™，FreeBSD™等等。

[0175] 上述实施例中由服务器所执行的步骤可以基于该图11所示的服务器结构。

[0176] 在本申请实施例中，该服务器所包括的CPU 322还具有以下功能：

获取目标节点所对应的第一节点信息集合，其中，第一节点信息集合包括M个上游节点的第一节点信息，M个上游节点均为目标节点的上游节点，M为大于或等于1的整数；

根据第一节点信息集合确定M个节点分值，其中，M个节点分值与M个上游节点具有一一对应的关系；

根据M个节点分值，从M个上游节点中确定第一上游节点，其中，第一上游节点所对应的节点分值为M个节点分值中的最大值；

若第一上游节点满足节点注册条件，则接收第一上游节点发送的数据。

[0177] 可选地,CPU 322具体用于执行如下步骤:

根据目标节点所对应的第一上游列表信息确定M个上游节点;

向M个上游节点中的每个上游节点发送信息采集请求,以使每个上游节点响应于信息采集请求,并获取每个上游节点的第一节点信息;

当接收到每个上游节点发送的第一节点信息时,获取第一节点信息集合。

[0178] 可选地,CPU 322具体用于执行如下步骤:

针对第一节点信息集合中的每个第一节点信息,若上游节点的状态信息指示非正常状态,则确定上游节点的节点分值为预设值;

针对第一节点信息集合中的每个第一节点信息,若上游节点的状态信息指示正常状态,则根据上游节点的关联信息确定节点分值,其中,节点分值大于或等于预设值。

[0179] 可选地,CPU 322具体用于执行如下步骤:

根据上游节点的挂载节点数量、上游节点的深度以及上游节点的负载信息,计算分值中间量;

根据分值中间量以及上游节点的已提交日志信息,计算得到上游节点的节点分值,其中,节点分值与上游节点的已提交日志信息呈正相关,节点分值与上游节点的挂载节点数量、上游节点的深度以及上游节点的负载信息均呈负相关。

[0180] 可选地,CPU 322具体用于执行如下步骤:

按照从大到小的次序,对M个节点分值进行排序处理,得到排序结果;

根据排序结果对第一上游列表信息中的节点进行排序处理,得到第二上游列表信息,其中,第二上游列表信息包括M个上游节点;

将第二上游列表信息中的首个上游节点确定为第一上游节点。

[0181] 可选地,CPU 322还用于执行如下步骤:

向第一上游节点发送节点注册请求,以使第一上游节点根据节点注册请求确定可挂载数量;

若可挂载数量大于或等于1,则确定第一上游节点满足节点注册条件;

若可挂载数量为0,则确定第一上游节点不满足节点注册条件。

[0182] 可选地,CPU 322还用于执行如下步骤:

向第一上游节点发送节点注册请求;

接收第一上游节点发送的节点注册响应,其中,节点注册响应携带第一上游节点的角色信息;

若角色信息指示第一上游节点为从节点或者学习节点,则确定第一上游节点满足节点注册条件;

若角色信息指示第一上游节点为主节点,则确定第一上游节点不满足节点注册条件。

[0183] 可选地,CPU 322还用于执行如下步骤:

若第一上游节点不满足节点注册条件,则根据M个节点分值,从M个上游节点中确定第二上游节点,其中,第二上游节点所对应的节点分值为M个节点分值中的次大值;

若第二上游节点满足节点注册条件,则接收第二上游节点发送的数据。

[0184] 可选地,CPU 322还用于执行如下步骤:

接收网络设备发送的数据读取请求;

根据数据读取请求，向网络设备发送元数据信息。

[0185] 可选地，CPU 322还用于执行如下步骤：

接收网络设备发送的第一数据读取请求；

根据第一数据读取请求，向网络设备发送第一元数据信息，其中，网络设备还用于向除目标节点以外的其他节点发送第二数据读取请求，第二数据读取请求用于获取第二元数据信息。

[0186] 可选地，CPU 322具体用于执行如下步骤：

接收第一网络设备发送的初始查询请求；

根据初始查询请求，将第一上游节点中的目标元数据信息同步至目标节点；

CPU 322还用于执行如下步骤：

接收第二网络设备发送的数据查询请求；

根据数据查询请求，向第二网络设备发送目标元数据信息。

[0187] 可选地，CPU 322还用于执行如下步骤：

当第一上游节点发生故障时，获取目标节点所对应的第二节点信息集合，其中，第二节点信息集合包括N个上游节点的第二节点信息，N个上游节点均为目标节点的上游节点，N为大于或等于1的整数；

根据第二节点信息集合确定N个节点分值，其中，N个节点分值与N个上游节点具有一一对应的关系；

根据N个节点分值，从N个上游节点中确定第三上游节点，其中，第三上游节点所对应的节点分值为N个节点分值中的最大值；

若第三上游节点满足节点注册条件，则将第三上游节点中的数据同步至目标节点。

[0188] 本申请实施例中还提供一种计算机可读存储介质，该计算机可读存储介质中存储有计算机程序，当其在计算机上运行时，使得计算机执行如前述各个实施例描述的方法。

[0189] 本申请实施例中还提供一种包括程序的计算机程序产品，当其在计算机上运行时，使得计算机执行前述各个实施例描述的方法。

[0190] 所属领域的技术人员可以清楚地了解到，为描述的方便和简洁，上述描述的系统，装置和单元的具体工作过程，可以参考前述方法实施例中的对应过程，在此不再赘述。

[0191] 在本申请所提供的几个实施例中，应该理解到，所揭露的系统，装置和方法，可以通过其它的方式实现。例如，以上所描述的装置实施例仅仅是示意性的，例如，所述单元的划分，仅仅为一种逻辑功能划分，实际实现时可以有另外的划分方式，例如多个单元或组件可以结合或者可以集成到另一个系统，或一些特征可以忽略，或不执行。另一点，所显示或讨论的相互之间的耦合或直接耦合或通信连接可以是通过一些接口，装置或单元的间接耦合或通信连接，可以是电性，机械或其它的形式。

[0192] 所述作为分离部件说明的单元可以是或者也可以不是物理上分开的，作为单元显示的部件可以是或者也可以不是物理单元，即可以位于一个地方，或者也可以分布到多个网络单元上。可以根据实际的需要选择其中的部分或者全部单元来实现本实施例方案的目的。

[0193] 另外，在本申请各个实施例中的各功能单元可以集成在一个处理单元中，也可以是各个单元单独物理存在，也可以两个或两个以上单元集成在一个单元中。上述集成的单

元既可以采用硬件的形式实现，也可以采用软件功能单元的形式实现。

[0194] 所述集成的单元如果以软件功能单元的形式实现并作为独立的产品销售或使用时，可以存储在一个计算机可读取存储介质中。基于这样的理解，本申请的技术方案本质上或者说对现有技术做出贡献的部分或者该技术方案的全部或部分可以以软件产品的形式体现出来，该计算机软件产品存储在一个存储介质中，包括若干指令用以使得一台计算机设备(可以是个人计算机，服务器，或者网络设备等)执行本申请各个实施例所述方法的全部或部分步骤。而前述的存储介质包括：U盘、移动硬盘、只读存储器(read-only memory，ROM)、随机存取存储器(random access memory，RAM)、磁碟或者光盘等各种可以存储程序代码的介质。

[0195] 以上所述，以上实施例仅用以说明本申请的技术方案，而非对其限制；尽管参照前述实施例对本申请进行了详细的说明，本领域的普通技术人员应当理解：其依然可以对前述各实施例所记载的技术方案进行修改，或者对其中部分技术特征进行等同替换；而这些修改或者替换，并不使相应技术方案的本质脱离本申请各实施例技术方案的精神和范围。

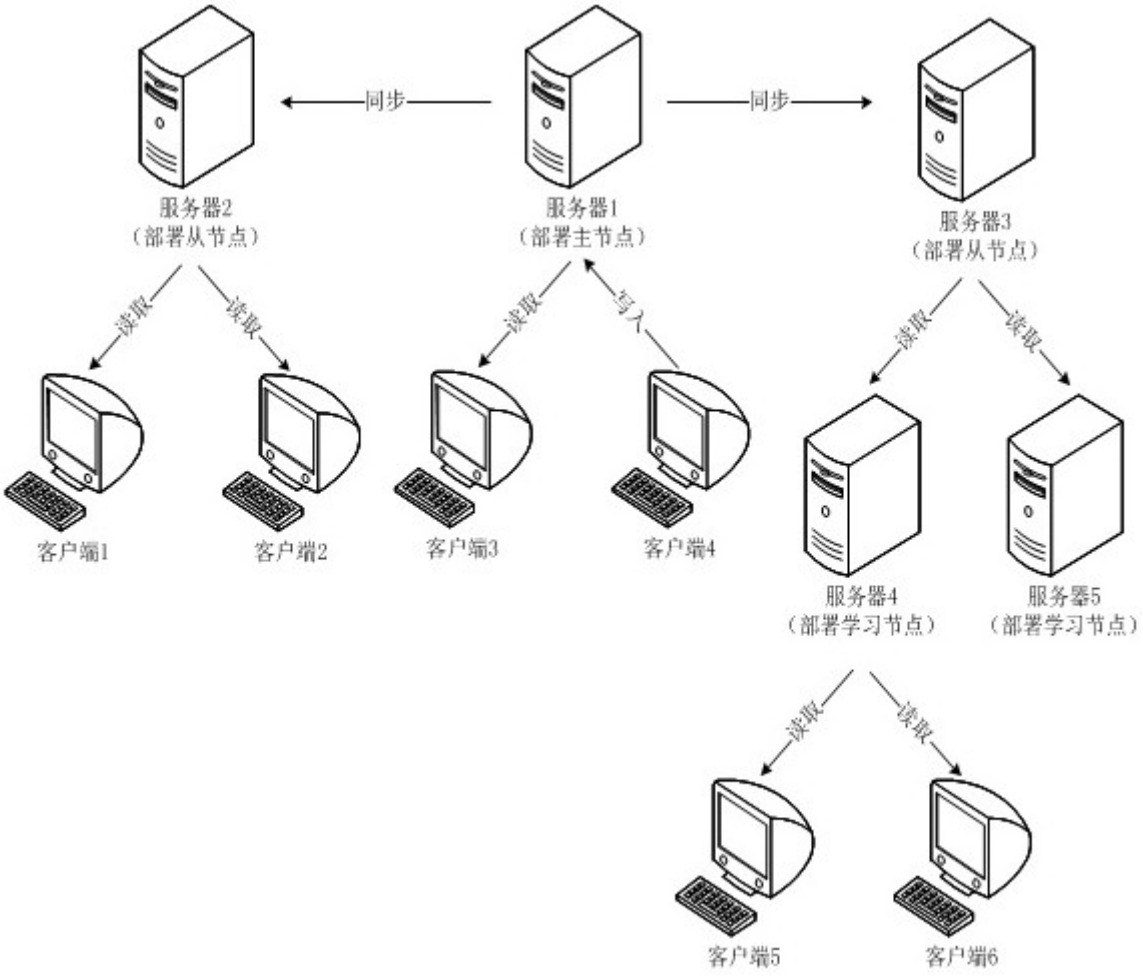

同步
同步
服务器2
（部署从节点）
服务器1
（部署主节点）
服务器3
（部署从节点）
读取
读取
读取
写入
读取
读取
客户端1
客户端2
客户端3
客户端4
服务器4
（部署学习节点）
服务器5
（部署学习节点）
读取
读取
客户端5
客户端6

图1

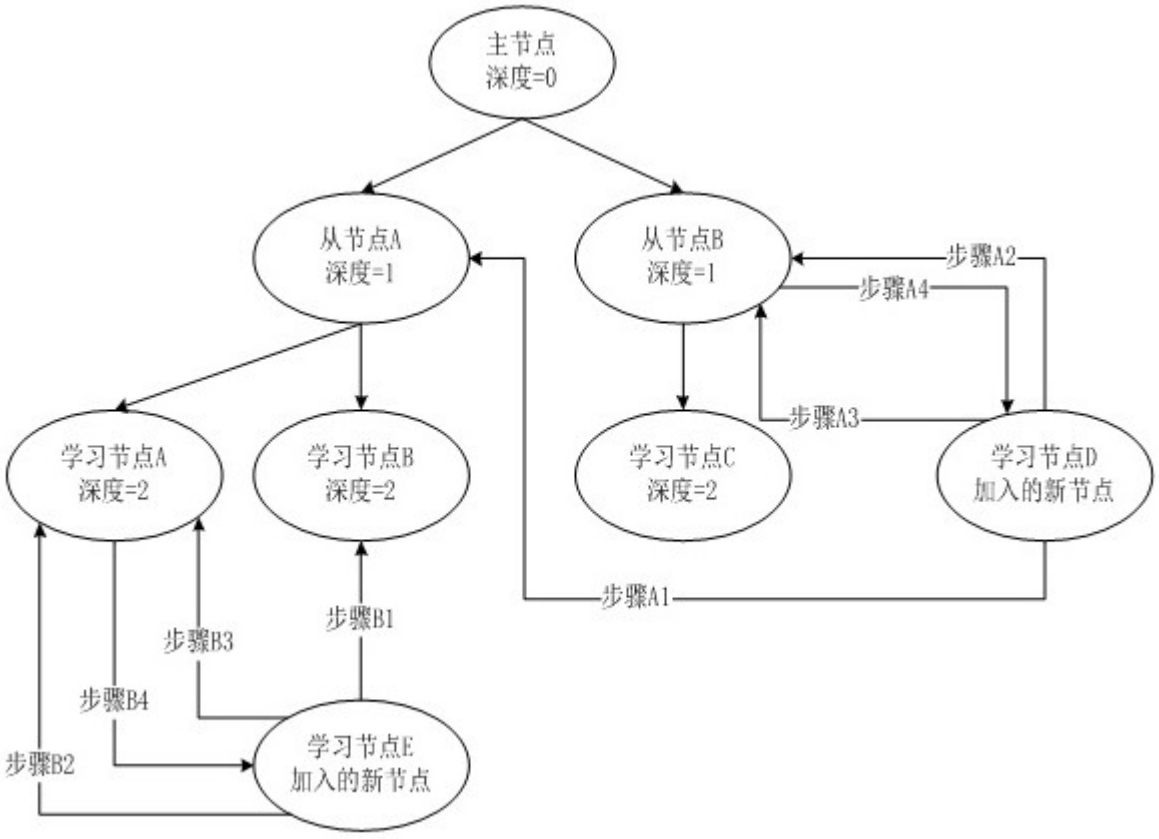

主节点
深度=0
从节点A
深度=1
从节点B
深度=1
步骤A2
步骤A4
步骤A3
学习节点A
深度=2
学习节点B
深度=2
学习节点C
深度=2
学习节点D
加入的新节点
步骤A1
步骤B1
步骤B3
步骤B4
步骤B2
学习节点E
加入的新节点

图2

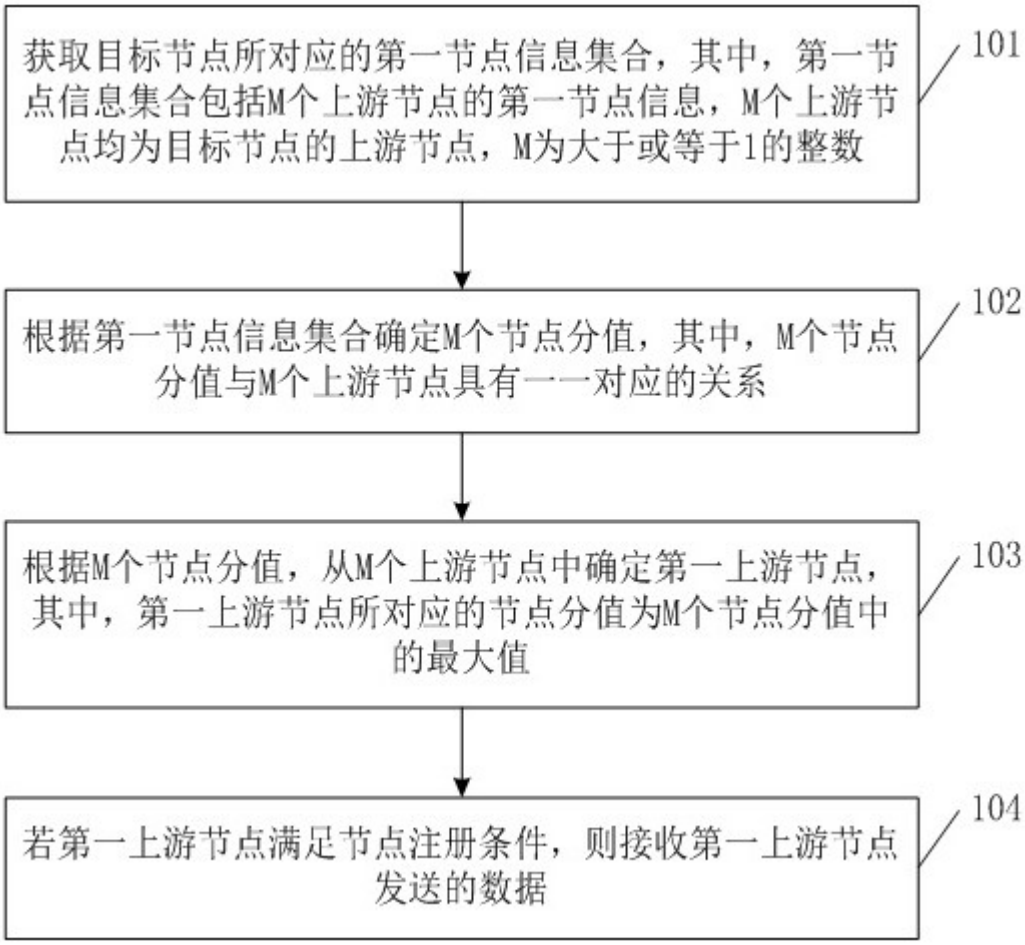

获取目标节点所对应的第一节点信息集合，其中，第一节点信息集合包括M个上游节点的第一节点信息，M个上游节点均为目标节点的上游节点，M为大于或等于1的整数
101
根据第一节点信息集合确定M个节点分值，其中，M个节点分值与M个上游节点具有一一对应的关系
102
根据M个节点分值，从M个上游节点中确定第一上游节点，其中，第一上游节点所对应的节点分值为M个节点分值中的最大值
103
若第一上游节点满足节点注册条件，则接收第一上游节点发送的数据
104

图3

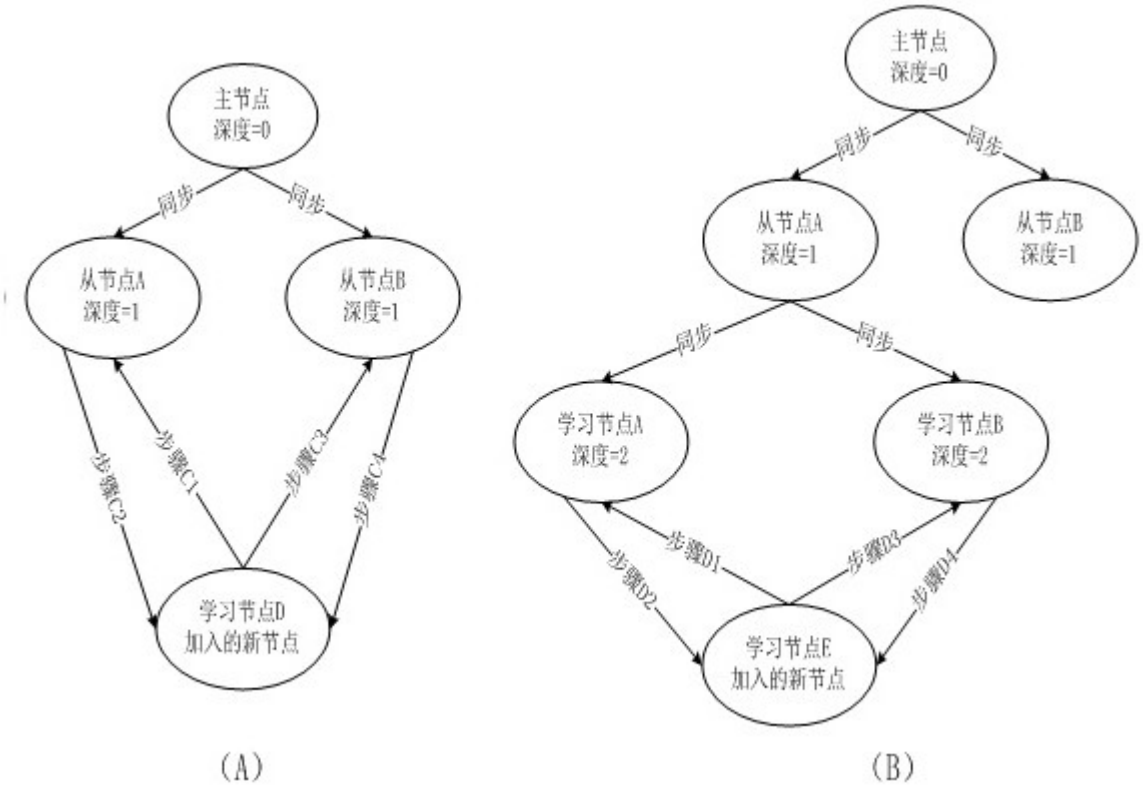

主节点
深度=0
同步
同步
从节点A
深度=1
从节点B
深度=1
步骤C1
步骤C2
步骤C3
步骤C4
学习节点D
加入的新节点
(A)
主节点
深度=0
同步
同步
从节点A
深度=1
从节点B
深度=1
同步
同步
学习节点A
深度=2
学习节点B
深度=2
步骤D1
步骤D2
步骤D3
步骤D4
学习节点E
加入的新节点
(B)

图4

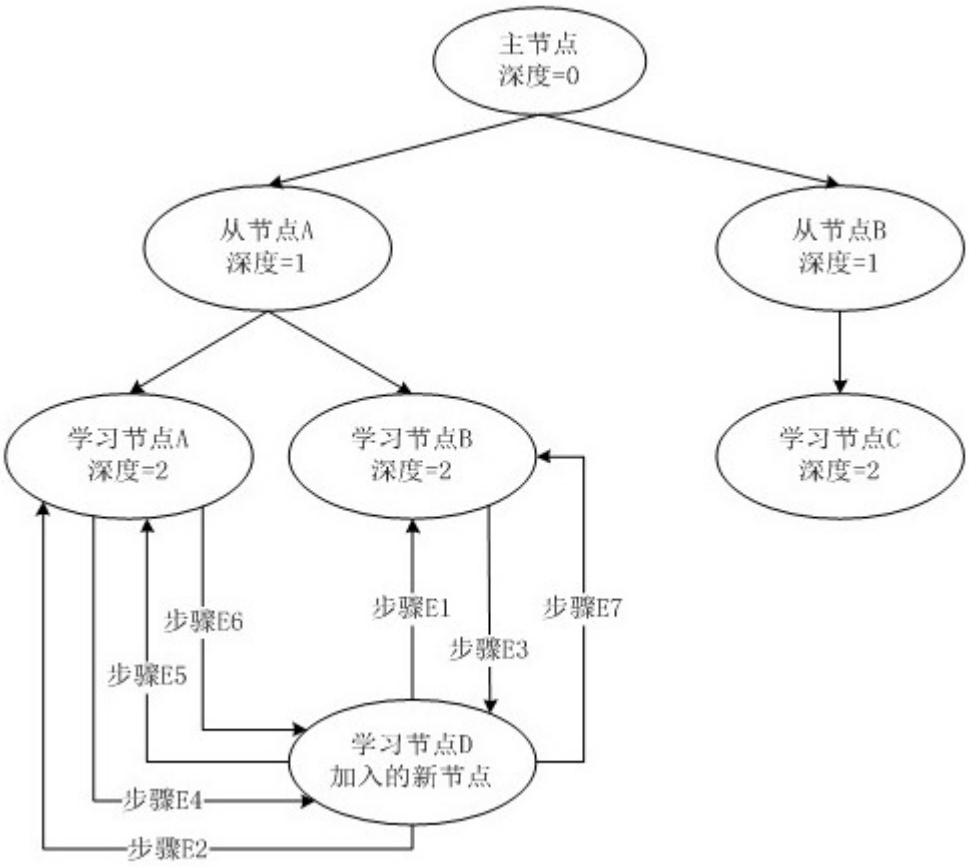

主节点
深度=0
从节点A
深度=1
从节点B
深度=1
学习节点A
深度=2
学习节点B
深度=2
学习节点C
深度=2
步骤E6
步骤E1
步骤E7
步骤E3
步骤E5
学习节点D
加入的新节点
步骤E4
步骤E2

图5

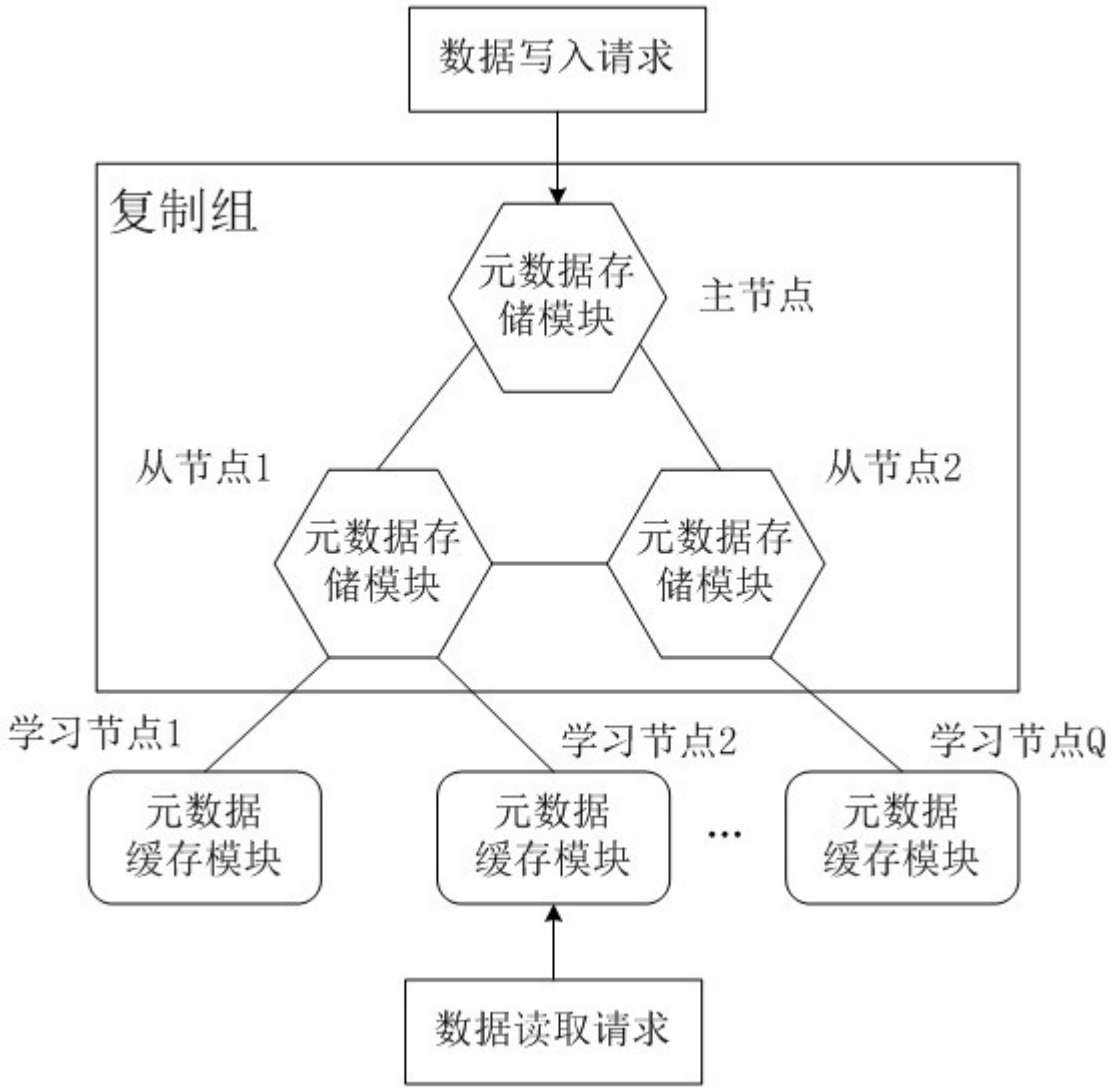

数据写入请求
复制组
元数据存储模块
主节点
从节点1
从节点2
元数据存储模块
元数据存储模块
学习节点1
学习节点2
学习节点Q
元数据缓存模块
元数据缓存模块
...
元数据缓存模块
数据读取请求

图6

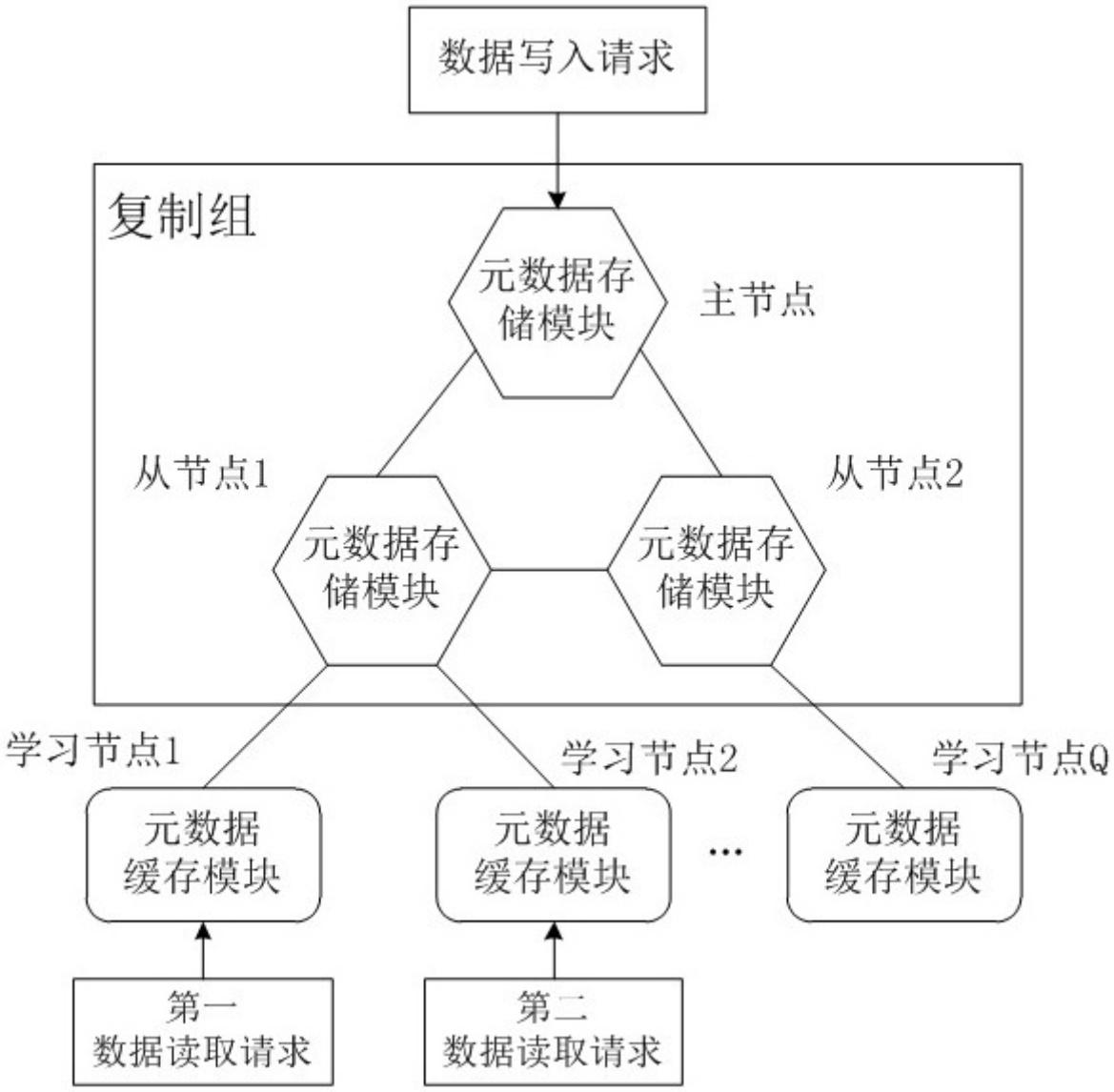

数据写入请求
复制组
元数据存储模块
主节点
从节点1
从节点2
元数据存储模块
元数据存储模块
学习节点1
学习节点2
学习节点Q
元数据缓存模块
元数据缓存模块
...
元数据缓存模块
第一数据读取请求
第二数据读取请求

图7

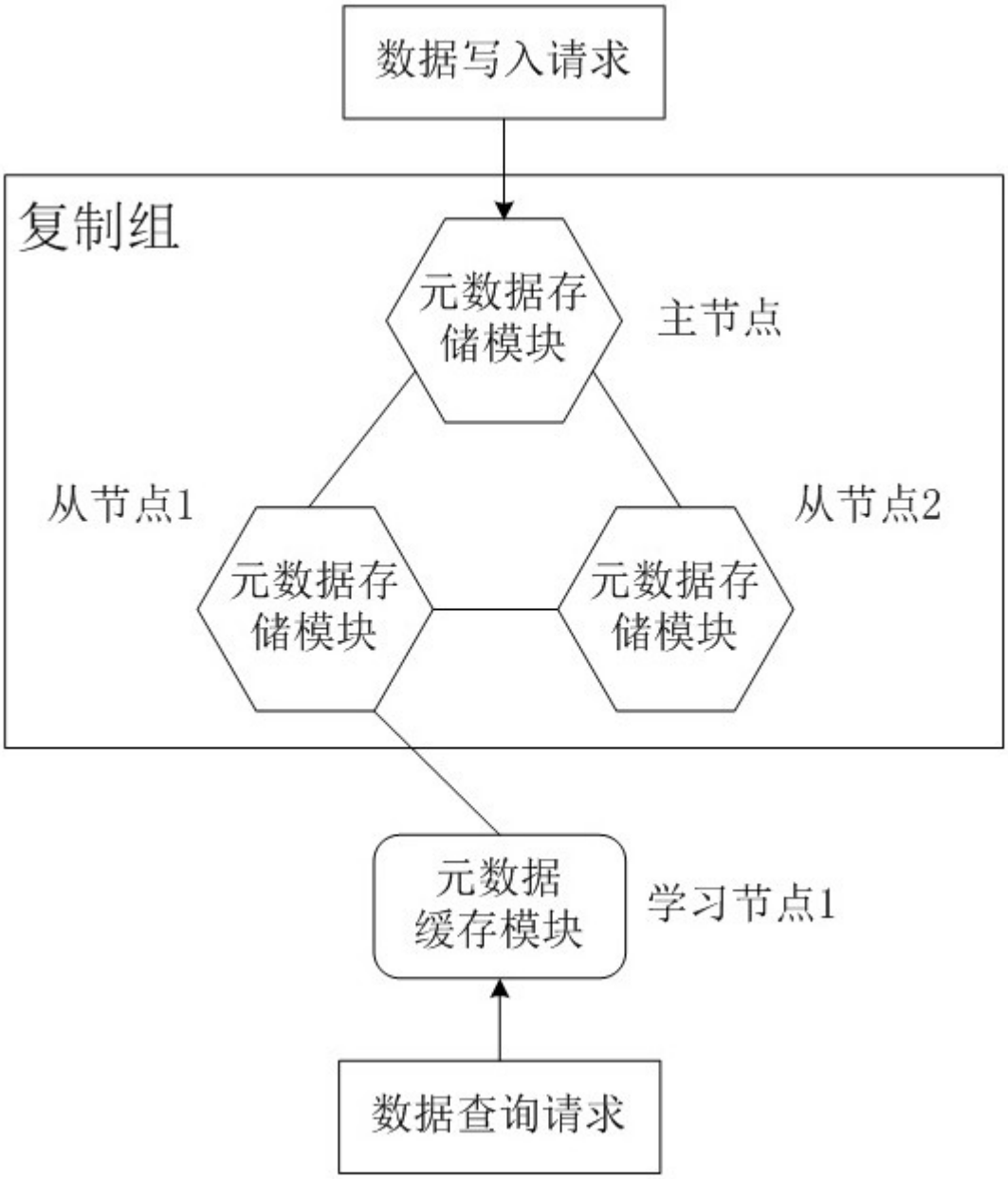

数据写入请求
复制组
元数据存储模块
主节点
从节点1
从节点2
元数据存储模块
元数据存储模块
元数据缓存模块
学习节点1
数据查询请求

图8

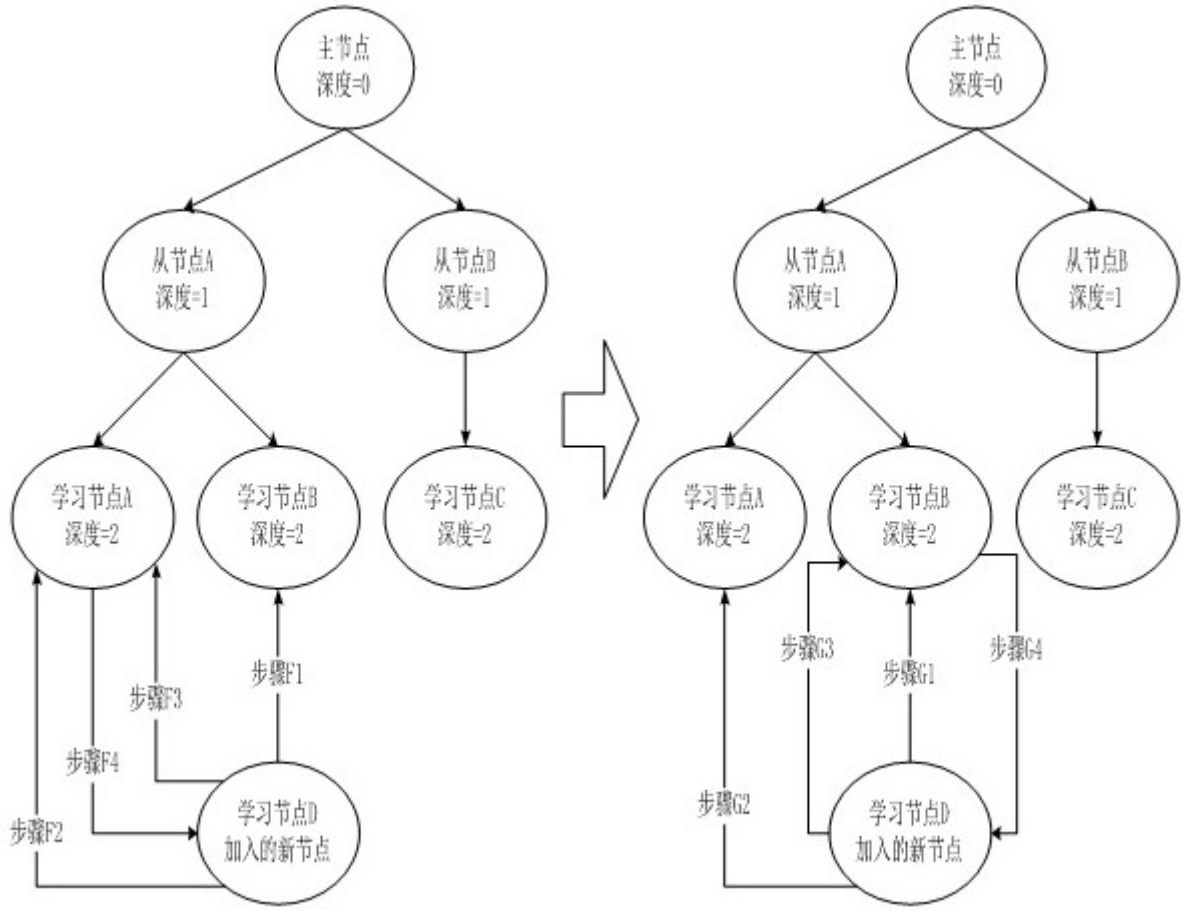

主节点
深度=0
从节点A
深度=1
从节点B
深度=1
学习节点A
深度=2
学习节点B
深度=2
学习节点C
深度=2
步骤F1
步骤F2
步骤F3
步骤F4
学习节点D
加入的新节点
主节点
深度=0
从节点A
深度=1
从节点B
深度=1
学习节点A
深度=2
学习节点B
深度=2
学习节点C
深度=2
步骤G1
步骤G2
步骤G3
步骤G4
学习节点D
加入的新节点

图9

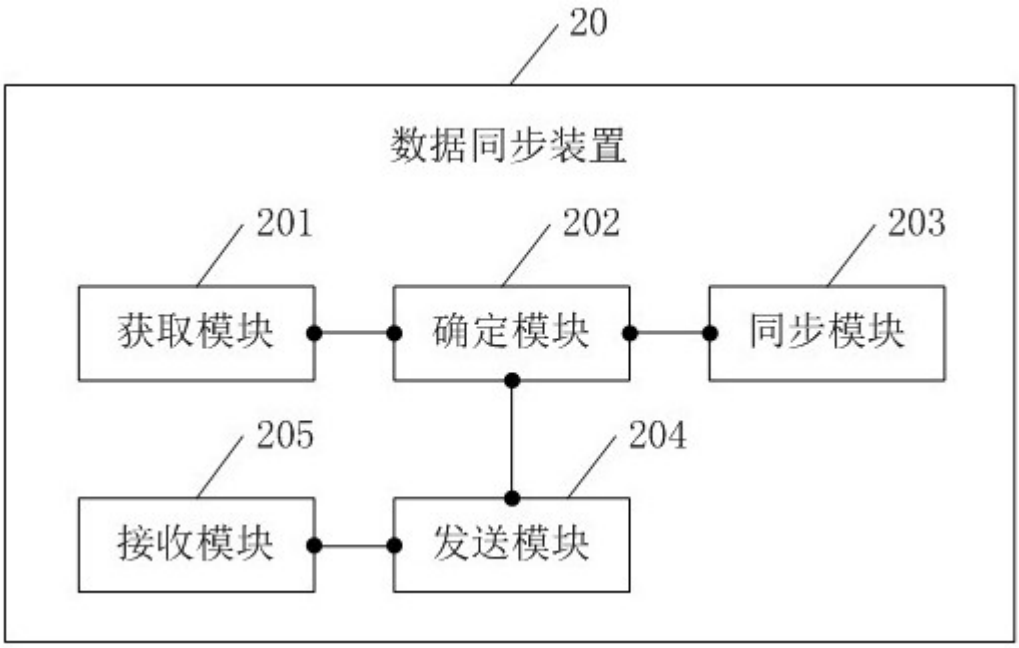

20
数据同步装置
201
202
203
获取模块
确定模块
同步模块
205
204
接收模块
发送模块

图10

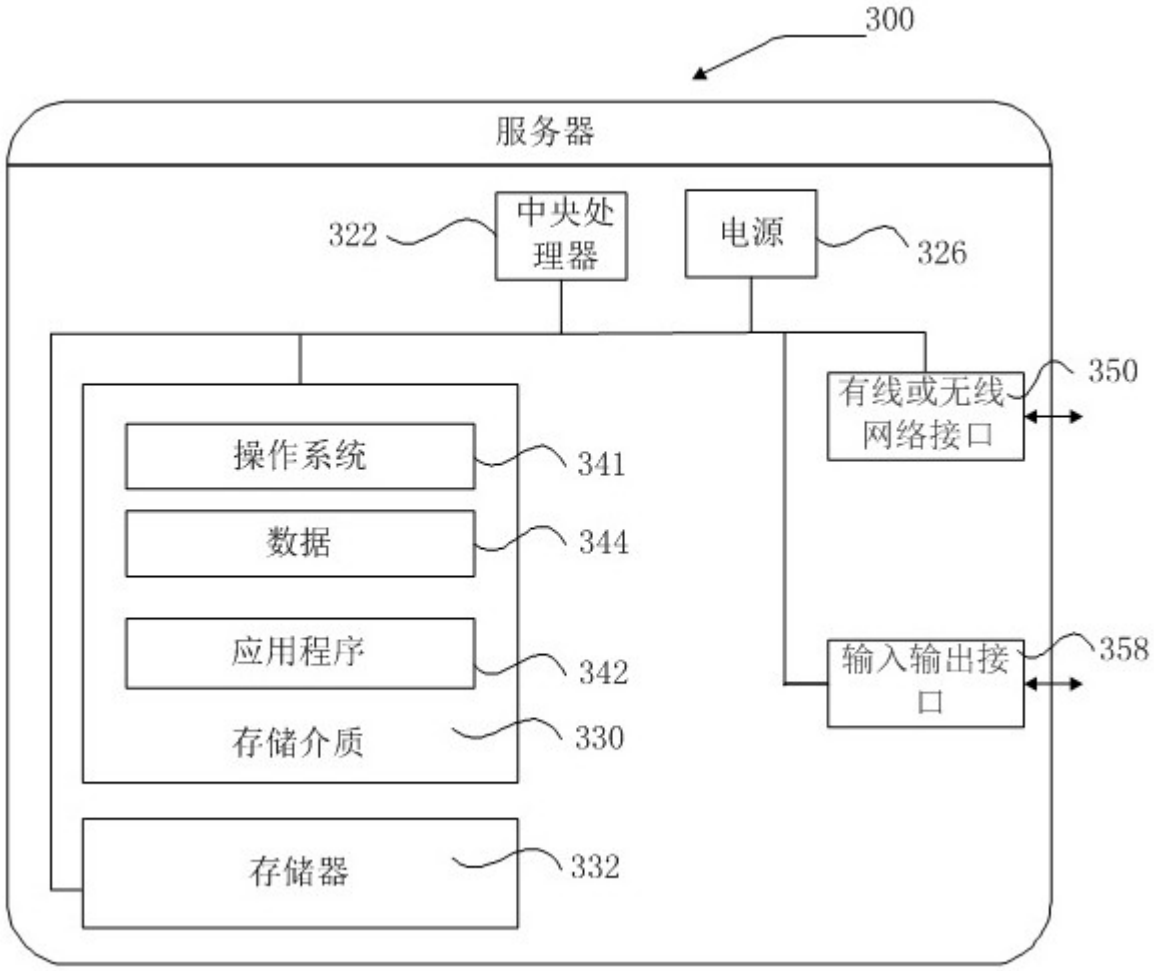

300
服务器
322
中央处理器
电源
326
有线或无线网络接口
350
操作系统
341
数据
344
应用程序
342
存储介质
330
输入输出接口
358
存储器
332

图11