http://bhxb.buaa.edu.cn jbuaa@buaa.edu.cn
DOI: 10.13700/j.bh.1001-5965.2024.0750

基于Gossip协议的高效集群数据同步方案

张宏海[1,2,*]，崔斌豪[1,2]，李一鸣[1,2]，田丰[1,2]，贾永强[1,2]，肖傲三[1,2]

(1. 中国民航信息网络股份有限公司 研发中心，北京 101318；2. 中国民航信息网络股份有限公司 北京市民航大数据工程技术研究中心，北京 101318)

摘　　要：随着民航客票运价搜索系统业务的快速发展，系统集群内部网络流量规模不断增长。为了在高负载网络流量场景下，实现集群内部数据同步，提出基于Gossip协议的集群数据同步方案。所提方案从网络协议的传输层和应用层着手设计，在传输层使用用户数据报协议（UDP）来减少集群中节点间的连接数量和交互次数，从而实现数据在网络传输过程中的低流量、低耗时。在应用层使用Gossip传播协议来实现数据的最终一致性，保证数据传输的可靠性。通过传输层UDP传输协议和应用层Gossip传播协议相结合，保证了集群监控过程中数据同步的高效性和可靠性。

关 键 词：Gossip协议；数据同步；一致性算法；高性能集群；去中心化
中图分类号：TP393
文献标志码：A　　**文章编号**：1001-5965(2025)05-1629-08

民航运价搜索系统作为互联网时代旅客自助购买机票的基础工具，承载了海量民航旅客用户的机票查询业务，为保障这一关键民航业务的安全运行，需要系统为用户提供低延迟、高稳定、高扩展的搜索服务。为保证系统的稳定运行，集群内部的数据同步方案，特别是监控数据的同步方案显得尤为重要[1]。

民航运价搜索系统的数据同步，面临2个方面的压力：①民航运价搜索系统涉及机票、航班等业务，业务复杂度高，系统存在交互报文大的特点；②系统承载了海量的用户查询量，导致集群内部面临巨大的网络流量压力。如何在高网络负载的情景下，保障系统的低延迟、高稳定、高扩展，是系统当前面临的主要技术障碍。针对上述技术障碍，本文提出一套采用基于用户数据报协议(user datagram protocol, UDP)的Gossip协议 的集群数据同步方案，可以在高负载网络流量的情况下，保障系统高可靠的服务能力。

传统集群数据同步方案通常基于传输控制协议(transmission control protocol, TCP)，在高并发和高网络负载的情况下，采用传统的TCP方式进行数据同步会面临一些瓶颈。TCP是一种面向连接的协议，其缺点包括连接的建立和维护成本较高、相对较重的头部开销及对可靠性的强调会导致较高的延迟。在高并发的场景中，这些问题可能进一步放大，导致系统的性能瓶颈和资源消耗增加。相比之下，基于UDP的Gossip协议在高并发、高网络负载的环境中具有一些优势。UDP是一种无连接、轻量级的传输协议，不涉及连接的建立和维护，具有较低的传输延迟。结合Gossip协议，通过节点之间的分布式信息传播，实现数据同步具有去中心化、容错性强的特点，使系统更具弹性。在大规模网络中，UDP的轻量级和低开销特性有助于降低通信开销，从而更有效地应对高并发和高网络负载的挑战。本文方案采用UDP-Based Gossip Protocol进行数据同步可以在高并发环境中更好地适应系统

收稿日期：2024-10-16；**录用日期**：2024-11-08；**网络出版时间**：2024-11-19 14:17
网络出版地址：link.cnki.net/urlid/11.2625.V.20241119.0932.001
基金项目：国家自然科学基金 (U2033203)
*通信作者. E-mail：hhzhang@travelsky.com.cn

引用格式：张宏海，崔斌豪，李一鸣，等. 基于Gossip协议的高效集群数据同步方案[J]. 北京航空航天大学学报，2025，51(5)：1629-1636. ZHANG H H，CUI B H，LI Y M，et al. An efficient cluster data synchronization scheme based on the Gossip protocol[J]. Journal of Beijing University of Aeronautics and Astronautics，2025，51(5)：1629-1636 (in Chinese).

动态变化，保障业务响应的低延迟，提高系统的扩展性和稳定性[2-3]。

1 Gossip协议实现原理

1.1 Gossip协议发展历史综述

Gossip协议的研究起源可追溯至1972年，Baker和Shostak对“电话问题”进行了描述，即探究最少的电话交流次数，使得每位妇女都能掌握所有流言[4]。1988年，Hedetniemi等对Gossip问题在计算机网络环境中进行了更为精确的概念界定，核心议题转变为信息如何在网络节点间传播[5]。Galton-Watson模型和Bailey的深入分析为Gossip算法的理论基础奠定了基石，而马尔可夫链的引入则进一步丰富了对其分析的方法论[6]。

自20世纪50年代至70年代，学者们致力于构建模型以描述Gossip机制并对其进行分析。随后十年间，该领域的研究热情有所减退[7]。随着新型网络技术的兴起和复杂网络研究的深化，Gossip协议的研究重新受到重视，近十年中取得了显著进展，其应用范围也扩展至数据库复制、故障检测、P2P网络的拓扑构建与维护，以及分布式环境下的聚合计算等多个领域[7]。

1.2 Gossip协议的定义

Gossip协议，又称为Epidemic协议[8]。该协议的核心思想是，当一个节点需要将信息同步到网络中的其他节点时，该节点可以定期地随机选择一些节点作为信息同步的目标。这些接收到信息的节点随后以相同的方式继续传播信息。Gossip协议模拟了信息的自发传播，类似于gossip(流言)和epidemic(流行病)的传播过程[9-10]。

Gossip协议以其简单高效、高扩展性、高容错性和去中心化等特点而著称。其设计追求一致性收敛，收敛速度较快，消息传播速度可达到对数级别的复杂度$Log_2(N)$。这使得Gossip协议在大规模分布式系统中表现出色。尽管Gossip协议在许多方面具有显著的优势，但需要注意的是，在实时性要求较高的场景中，由于需要进行多轮传播，可能会引入一定的消息延迟[11-12]。

Gossip协议的数据一致性和数据收敛性，是其最显著的优势，以下针对这2点进行着重论证。Gossip协议的收敛性主要体现在其时间复杂度：

$$T = O(\log_2 N) \tag{1}$$

式中：T为收敛所需的时间或轮次；N为网络中的节点数。式(1)表明Gossip协议的收敛时间随着网络规模的增长呈对数增加，能够在对数级别轮次中达到快速收敛。Gossip协议的一致性体现在其消息的传播达到一致性的概率：

$$P_{\text{consistent}} = 1 - e^{-\lambda N} \tag{2}$$

式中：λ为与Gossip协议参数相关的常数即传播概率；N为网络中的节点数。式(2)表明Gossip协议通过多轮信息交换，使得每个节点知道所有信息的概率逐渐增加，最终趋近于100%，确保了系统的一致性。同时，通过调整Gossip协议的参数，如每个周期内节点交换信息的频率和数量，可以提高达到一致性的概率和速度。

1.3 Gossip协议的通信方式

在Gossip协议下，2个节点之间通信方式主要有推模式、拉模式及推拉模式3种，这些模式在节点之间的通信中起着关键作用，决定了系统的效率和可靠性。推、拉和推拉这3种通信方式的特点，如图1所示。

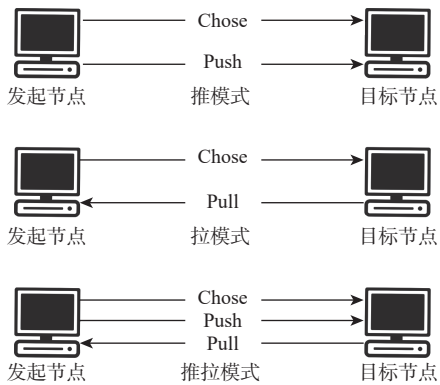

图1 Gossip协议的通信方式

Fig. 1 Communication method of Gossip protocol

1) 推模式。信息的发起节点会随机选择其邻近节点作为目标节点，并向其发送自己的消息。这一模式的特点是仅有拥有消息的节点才有资格成为发起节点。

2) 拉模式。信息的发起节点会随机选择邻近节点作为目标节点，但这次是从目标节点主动获取对方的消息。这一模式的优势在于，只有那些没有新消息的节点才会作为发起节点。

3) 推拉模式。是将推和拉两者结合起来，使消息的发起节点不仅向随机选择的目标节点发送消息，同时也从对方获取消息。这种综合模式综合了推和拉的优点，克服了各自的局限性。通过推拉模式，系统能够更加灵活地进行信息的传递，既能高效推送消息，又能在需要时主动拉取所需的信息，从而提高系统的整体性能。

若将2个节点数据同步一次定义为一个周期，推模式在一个周期内需要进行1次通信，拉模式需要2次通信，而推拉模式则需要3次通信。尽管消息数量增加，但从效果上来看，推拉模式表现出最

高的效率, 理论上推拉模式的收敛速度也是最快的, 能够在一个周期内使得 2 个节点完全一致。

2　系统设计

2.1　系统架构

本文方案的系统架构主要由部署在集群中各节点的协调器组成, 协调器之间的网络通信主要通过 UDP 传输协议实现, 协调器之间的数据同步功能基于 Gossip 协议实现。协调器的主要组件包括 3 部分: 接收器组件、发送器组件、缓存组件, 系统组成逻辑如图 2 所示。

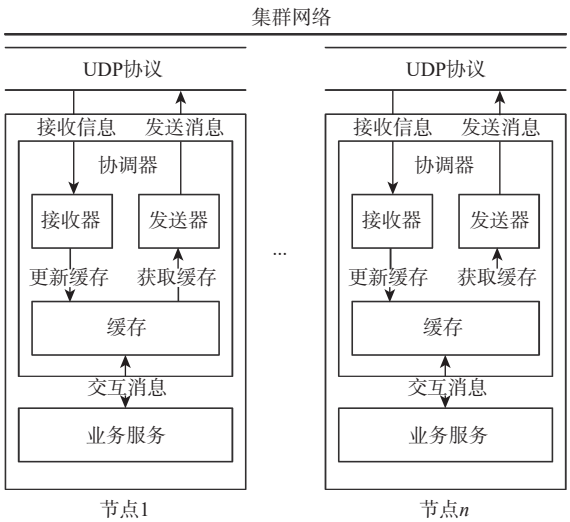

图 2　系统组成逻辑

Fig. 2　System composition logic

1) 发送器组件。负责对其他节点协调器分发请求信息, 将本节点的消息推送给集群中其他节点。

2) 接收器组件。接受其他协调器发来的请求信息, 目的是收集其他协调器发来的服务器信息。

3) 缓存组件。对协调器接受的服务器信息进行收集和存储, 对外提供查询和写入接口, 可使从外部获取服务器集群的相关信息, 节点中其他服务也可以通过接口和缓存之间交互信息。

2.2　传输协议的选择

本文方案根据民航运价搜索系统的业务特点, 最终选择 UDP 协议作为传输协议。选择 UDP 协议, 而放弃 TCP 协议主要出于 2 个方面的考虑: ①民航运价搜索系统的搜索业务对低延迟有强制性的要求, 而 TCP 协议是一种面向连接的协议, 连接的建立和维持成本较高, 存在较高的延迟; ②民航运价搜索业务的报文大并发高, 导致整体的网络负载较高, TCP 协议本身会占用大量网络连接数, 加之每次连接中间存在握手过程产生大量的数据包, 高网络负载的瓶颈会被进一步放大。综上两方面考虑, UDP 协议在本系统中更具优势[13-14]。

2.3　Gossip 传播模型

本节着重介绍 Gossip 协议节点间消息传播用到的传播模型。在讨论传播模型时, 将借鉴流行病学术语。根据术语, 集群中的每个节点可以处于以下 3 种状态之一, 即:

易感态: 节点未获取更新。

感染态: 节点已获取更新并传播。

愈除态: 节点已获取更新, 并且未发生新的更新, 不再参与传播过程。

Gossip 协议主要有 2 种传播模型: 简单流行病 (simple epidemics, SI) 模型和传染病 (susceptible infected recovered, SIR) 模型。

1) SI 模型。所有参与节点有 2 种状态, 分别为易感态和感染态。过程是种子节点会把所有的数据都跟其他节点共享, 以便消除节点之间数据的任何不一致, 其可以保证最终、完全的一致。通常只用于新加入节点的数据初始化[15]。

SI 模型的收敛速度可以通过下式近似:

$$E(S_{t+1}) = S_t\left(1-\frac{1}{N}\right)^{N(1-S_t)} \approx S_t e^{-(1-S_t)} \tag{3}$$

式中: S_t 为在第 t 个周期结束时易感节点的比例; N 为网络中节点的总数; $(1-1/N)$ 为固定感染节点不会感染某个固定易感节点的概率。显然, 如果一个节点在第 t+1 个周期是易感的, 那么其在第 t 个周期是易感的, 并且所有感染节点都选择了其他节点。实际上, 这个近似模型相当准确, 本文将期望值 $E(S_{t+1})$ 作为 S_{t+1} 的良好近似。由式(3)可以看出, 如果等待足够长的时间, 那么最终所有节点都会收到更新。换句话说, 一个特定节点永远不会收到更新的概率为 0。让每个节点都获得更新(被感染)所需的周期数为

$$S_N = \log_2 N + \ln N + O(1) \tag{4}$$

式中: S_N 为传播更新至所有节点所需的周期数。由式(4)可以看出, SI 模型传播效率相当高, 只需要 $O(\log_2 N)$ 个周期就可以让每个节点都获得更新, 这表明 SI 模型具有非常好的可扩展性。

SI 模型算法的伪代码如算法 1 所示。

算法 1　SI 模型算法。

```
loop
    wait(Δ)
    p ← random peer
    if push and in state I then
        send update to p
    end if
```

```
        if pull then
            send update-request to p
        end if
    end loop
    procedure ONUPDATE(m)
        store m.update
    end procedure
    procedure ONUPDATEREQUEST(m)
        if in state I then
            send update to m.sender
        end if
    end procedure
```

2）SIR模型。所有参与节点有3种状态：易感态、感染态和愈除态。过程是消息只包含最新update数据，谣言消息在某个时间点之后会被标记为“removed”，并且不再被传播。通常用于节点间数据增量同步[15]。

SIR模型相较于SI模型引入了“移除”状态，运行协议在所有节点都已接收更新后进行终止，提高了效率。终止条件如下：

$$S^* = \exp[-(k+1)(1-S^*)] \tag{5}$$

式中：S^*为当传播终止时未获得更新的节点比例，理想情况下，S^*趋近于0，意味着所有节点都已接收到更新；k表示平均每个感染节点在停止传播前传播信息的次数，该参数直接影响传播的效率和终止条件。式(5)表明，随着k的增加，S^*的值会迅速减小，因为指数函数的衰减速度非常快。这意味着，增加每个感染节点传播的次数可以显著提高传播的覆盖率。在实际应用中，选择合适的k值是一个权衡过程，需要考虑网络的规模、更新的紧急性及系统的资源限制的。例如，在需要快速传播更新的场景中，可能会选择较大的k值以确保更新能够迅速覆盖大部分节点。而在资源受限的环境中，可能会选择较小的k值以减少网络负载。

SIR模型算法的伪代码如算法2所示。

算法2 SIR模型算法。

```
    loop
        wait(Δ)
        p ← random peer
        if push and in state I then
            send update to p
        end if
        if pull then
            send update-request to p
        end if
    end loop
    procedure ONFEEDBACK(m)
        switch to state R with prob. 1/k
    end procedure
    procedure ONUPDATE(m)
        if in state I or R then
            send feedback to m.sender
        else
            store m.update
        end if
    end procedure
    procedure ONUPDATEREQUEST(m)
        if in state I then
            send update to m.sender
        end if
    end procedure
```

2.4 自研传播协议原理

本文方案在应用层采用Gossip Push和Gossip Pull相结合的方式分别发送增量消息和全量消息，在网络传输层采用UDP单播和UDP多播相结合的方式传输网络报文，有效的减少了网络中的网络负载压力，同时保障了主业务服务的低延迟[16-18]，自研传播协议的时序图如图3所示。

具体步骤如下：

步骤1 启动协调器主进程。集群中的每个节点都部署有协调器服务，依次启动每台节点协调器服务进程。协调器进入初始化状态，并会持续监听节点上的业务服务进程，对业务接口进行实时监听，并将业务服务状态(api_state)写入缓存组件，如果监听到业务服务接口的状态更新，会将更新的数据写入缓存，如果接口状态没有变化则会持续监听。

步骤2 启动监听线程。协调器主进程启动的同时，监听线程随之启动，监听线程主要负责Gossip消息的监听，每个节点都有两类监听进程：Gossip主动监听线程和Gossip被动监听进程。

Gossip主动监听线程：

1）每隔固定时间t触发监听。

2）如果业务服务状态api_state的值为Join，表示业务服务为新加入服务，采用Push-Gossip方式将节点本地数据(LD)加入AckMsg消息当中，通过UDP广播的方式推送给集群中的其他目标节点。如果业务服务状态api_state的值为Update，表示业务服务的数据发生更新，首先随机选取邻居节点，同样采用Gossip push方式将节点本地数据(LD)加入AckMsg消息当中，通过UDP单播的方式推送给选中邻居节点发送消息。

3）更新节点状态node_state为I，表示当前节点易感染。

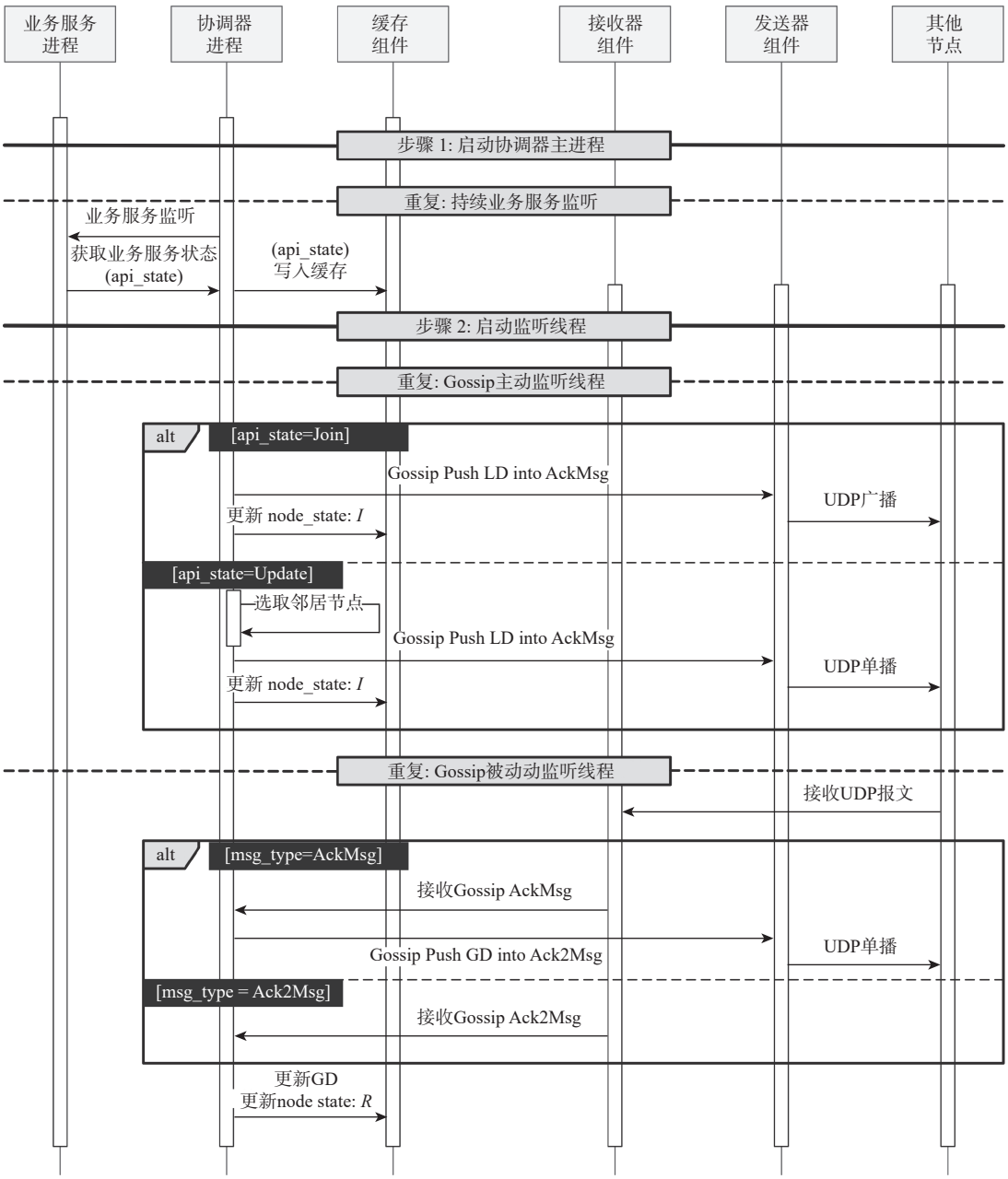

图 3　自研传播协议时序图

Fig. 3　Timing diagram of self-developed propagation protocol

4) 当前节点的被动监听进程进入监听状态, 等待和-Gossip push 消息对应的 Gossip pull 消息的返回。

Gossip 被动监听线程:

1) 每隔固定时间 t 触发监听, 接收网络中 UDP 报文。

2) 如果当前节点是目标节点, 则会接收到源节点发送过来的 AckMsg 消息, 将 AckMsg 中的本地数据(LD)和目标节点自身节点存储的全局数据(GD)进行合并, 并将合并后的全局数据(GD)加入到 Ack2Msg 消息当中, 通过 UDP 单播的方式返回给源节点。

3) 如果当前节点为源节点, 则会接收到目标节点返回的 Ack2Msg 消息, 将 Ack2Msg 中的全局数据(GD)和源节点自身存储的本地数据(LD)进行合并。

4) 重复第 2, 第 3 步, 如果持续 k 次收到同一消息, 将合并后的全局数据(GD)写入缓存组件, 同时将节点状态(node_state)更新为 R, 表示消息传播结束, 终止传播。

3　工程验证及分析

传统数据同步方法包括中心化同步、周期性轮询、时间戳同步、基于版本控制的同步和事务性同步, 其在实现简单性、一致性维护和系统复杂度之间存在权衡。与之相比, 基于 Gossip 协议的 SI 模型和 SIR 模型提供了去中心化、动态自适应性、概

率一致性保证、轻量级设计和系统健壮性等优势。Gossip协议通过优化收敛时间 $O(\log_2 n)$、传输率、通信负载、网络负载率和消息延迟等关键性能指标，在大规模分布式系统中展现出高效性。其概率性传播机制和轻量级通信策略，使得在保持数据一致性的同时，减少了消息冗余和网络负载，提高了数据同步的效率。

基于上述理论依据，系统使用Java语言在工程上进行了编码实现和实践验证，目前系统已部署到生产和开发环境，生产环境的机器和网络配置如表1所示。

表1 验证环境配置

Table 1 Verify environment configuration

参数	配置
编程语言	Java
操作系统	Linux Redhat v6.5
主机类型	X86服务器
CPU核数	16Core
内存	512 GB
硬盘	3T SSD
集群网络	万兆局域网
Node数量	100台

实验针对Gossip协议的数据一致概率、收敛周期、容错率3项关键评估指标进行验证分析。记录了10 000轮的数据同步结果，每1 000轮结果针对3项关键指标求算数平均值。实验设定的传播概率为0.8，即每个节点在每个周期内有80%的概率向随机选择的邻居节点传播信息。信息更新间隔设置为1轮，即每个节点在每个周期都会尝试更新其信息状态。每1 000轮随机故障率设为(0%～0.2%)，实验的统计分析数据如表2所示。

以上实验结果显示，一致概率平均值为99.84%，表明在绝大多数情况下，网络能够在极短时间轮次内达到高度的数据一致性。收敛周期的平均值为2.84轮，这意味着信息在网络中传播并达到一致性状态的速度非常快。容错率的平均值为99.62%，表明即使在部分节点出现故障的情况下，Gossip协议仍能保持极高的一致性，说明该协议具有很强的容错能力。从1 000轮到10 000轮，容错率略有波动，但总体上保持在99%以上，显示出Gossip协议在不同轮次和可能的故障条件下的稳定性。综上所述，Gossip协议在仿真测试中展现出了卓越的性能，尤其是在容错性和快速收敛方面的表现，使其成为分布式系统中数据同步的有力候选方案。

表2 Gossip协议关键评估指标

Table 2 Gossip Protocol Key Performance Metrics

轮次	一致概率/%	收敛周期/轮	容错率/%
1 000	98.8	3.2	99.8
2 000	99.9	2.9	100
3 000	100	2.8	99.9
4 000	99.9	3.1	98.7
5 000	99.6	2.2	99.3
6 000	99.7	2.4	98.9
7 000	100	2.5	98.8
8 000	100	3.0	100
9 000	99.9	2.7	99.9
10 000	100	2.6	100

为验证在高负载网络流量的集群中进行数据同步方案的可行性和有效性，选择了10台与生产机器同等配置的机器进行方案验证。每台机器的网络负载为100 TPS，每次请求的所有报文大小平均为150 kB。通过持续压测这些机器，持续时间达到了24 h，并详细记录了整个过程中每台机器使用传统数据同步协议和UDP-Based Gossip协议时的网络负载率和业务服务耗时。

在实验结束后，对这10台机器的数据进行了平均值计算，得到了2组对比实验数据，如图4和图5所示。通过2组对比数据，可以看到自研的UDP-Based Gossip协议相较于传统数据同步方案在网络负载率和业务耗时两方面均有显著提高，可以

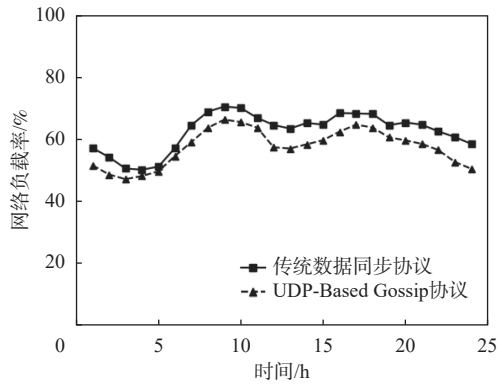

图4 网络负载率对比实验

Fig. 4 Comparison experiment of network load rate

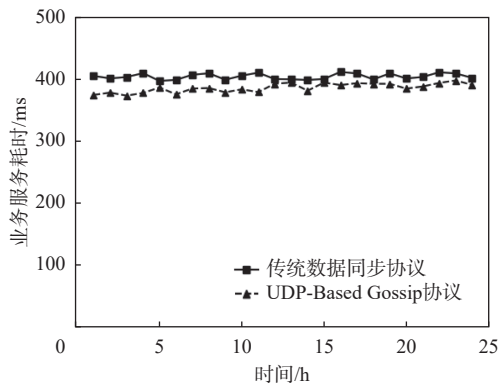

图5 业务服务耗时对比实验

Fig. 5 Comparison experiment of business service time consumption

更好的提升民航运价搜索系统的性能需求[19]。

4 结　论

1) 本文方案设计了一种基于 UDP 传输协议和 Gossip 传播协议的集群状态监控方法。解决了在高负载网络流量的场景下, 集群内业务服务状态的监控和数据同步的问题。

2) 本文方案从网络协议的传输层和应用层着手设计, 在传输层使用 UDP 传输协议来减少集群中节点间的连接数量和交互次数, 从而实现数据在网络传输过程中的低流量、低耗时。在应用层使用 Gossip 传播协议来实现数据的最终一致性, 保证数据传输的可靠性。

3) 通过传输层 UDP 传输协议和应用层 Gossip 传播协议的两方面的结合, 最终保证了集群监控过程中数据同步的高效性和可靠性, 在工程实践领域具有一定的典型性和通用性。

参考文献（References）

[1] 张宏海, 彭明田, 刘开胜. 基于价格导向的民航运价搜索方案设计[J]. 中国民航大学学报, 2020, 38(3): 40-44.
ZHANG H H, PENG M T, LIU K S. Design of price-based airfare search method[J]. Journal of Civil Aviation University of China, 2020, 38(3): 40-44(in Chinese).

[2] 王成全. 基于 UDP 的消息即时传输系统的研究与设计[J]. 信息技术, 2008(7): 162-164.
WANG C Q. Study and design of real-time message transmission system based on UDP[J]. Information Technology, 2008(7): 162-164(in Chinese).

[3] 刘锦江, 范洪博, 高志伟. 基于 Gossip 网络协议的前置消息验证[J]. 信息技术, 2022(7): 109-113.
LIU J J, FAN H B, GAO Z W. Gossip network protocol-based pre-message authentication[J]. JInformation Technology, 2022(7): 109-113(in Chinese).

[4] BAKER B, SHOSTAK R. Gossips and telephones[J]. Discrete Mathematics, 1972, 2(3): 191-193.

[5] HEDETNIEMI S M, HEDETNIEMI S T, LIESTMAN A L. A survey of gossiping and broadcasting in communication networks[J]. Networks, 1988, 18(4): 319-349.

[6] BOYD S, GHOSH A, PRABHAKAR B, et al. Analysis and optimization of randomized gossip algorithms[C]//Proceedings of the 43rd IEEE Conference on Decision and Control. Piscataway: IEEE Press, 2004: 5310-5315.

[7] WEATHERSPOON H, MIRANDA H, IWANICKI K, et al. Gossiping over storage systems is practical[J]. ACM SIGOPS Operating Systems Review, 2007: 41(5): 75-81.

[8] DEMERS A, GREENE D, HOUSER C, et al. Epidemic algorithms for replicated database maintenance[J]. Acm Sigops Operating Systems Review, 1988, 22(1): 8-32.

[9] MARK J, ALBERTO M, OZALP B. T-Man: Gossip-based fast overlay topology construction[J]. Computer Networks, 2009, 53(13): 2257-2258.

[10] NAOHIRO H, MAKOTO T. Design of the notification system for failure detectors[J]. International Journal of High Performance Computing and Networking, 2009, 6(1): 25-34.

[11] BOYD S, GHOSH A, PRABHAKAR B, et al. Randomized Gossip algorithms[J]. IEEE Transactions on Information Theory, 2006, 52(6): 2508-2530.

[12] GIUSEPPE D, DENIZ H, MADAN J, et al. Dynamo: amazon's highly available key-value store[J]. Symposium on Operating Systems Principles, 2007, 41(6): 205-220.

[13] 李明, 贾智平. 基于 TCP/IP 的分布式监控系统的研究与开发[J]. 计算机工程与应用, 2002, 38(16): 150-153.
LI M, JIA Z P. Research and development of distributed monitoring and control system based on TCP/IP protocol[J]. Computer Engineering and Applications, 2002, 38(16): 150-153(in Chinese).

[14] 王继刚, 顾国昌, 徐立峰, 等. 可靠 UDP 数据传输协议的研究与设计[J]. 计算机工程与应用, 2006, 42(15): 113-116.
WANG J G, GU G C, XUN L F, et al. Research and design of reliable data transfer based on UDP[J]. Computer Engineering and Applications, 2006, 42(15): 113-116(in Chinese).

[15] 刘德辉, 尹刚, 王怀民, 等. 分布环境下的 Gossip 算法综述[J]. 计算机科学, 2010, 37(11): 24-28.
LIU D H, YIN G, WANG H M, et al. Overview of Gossip algorithm in distribute system[J]. Computer Science, 2010, 37(11): 24-28(in Chinese).

[16] 张国印, 李军, 王向辉, 等. 一种基于移动 P2P 改进的 Gossip 算法[J]. 计算机科学, 2013, 40(9): 103-105.
ZHANG G Y, LI J, WANG X H, et al. Improved Gossip algorithm based on mobile P2P networks[J]. Computer Science, 2013, 40(9): 103-105(in Chinese).

[17] 李燚, 顾乃杰, 黄增士, 等. Redis 集群可靠性的研究与优化[J]. 计算机工程, 2018, 44(5): 40-46.
LI Y, GU N J, HUANG Z S, et al. Research and optimization of redis cluster reliability[J]. Computer Engineering, 2018, 44(5): 40-46(in Chinese).

[18] 王峰, 李立新, 曹景源, 等. 发布/订阅系统中的缓存副本一致性研究[J]. 计算机应用, 2016, 36(6): 1510-1514.
WANG F, LI L X, CAO J Y, et al. Research on replication consistency of cache in publish/subscribe systems[J]. Journal of Computer Applications, 2016, 36(6): 1510-1514(in Chinese).

[19] 杨永凯, 彭明田, 王炜东. 集群内高效可靠的数据文件分发方案的设计[J]. 计算机技术与发展, 2019, 29(11): 163-167.
YANG Y K, PENG M T, WANG W D. Design of effcient and reliable data file distribution solution in cluster[J]. Journal of Computer Applications, 2019, 29(11): 163-167(in Chinese).

An efficient cluster data synchronization scheme based on the Gossip protocol

ZHANG Honghai[1, 2, *], CUI Binhao[1, 2], LI Yiming[1, 2], TIAN Feng[1, 2], JIA Yongqiang[1, 2], XIAO Aosan[1, 2]

(1. Research and Development Center, TravelSky Technology Limited, Beijing 101318, China;
2. Beijing Engineering Technology Research Center of Civil Aviation Big Data, TravelSky Technology Limited, Beijing 101318, China)

Abstract: With the rapid development of the civil aviation ticket fare search system business, the internal network traffic scale of the search system cluster continues to grow. Under high-load network traffic, a cluster data synchronization scheme based on the Gossip protocol was developed. This solution is designed from the transport layer and application layer of the network protocol and uses the user datagram protocol (UDP) transmission protocol in the transport layer to reduce the number of connections and interactions between cluster nodes in order to achieve low traffic and low data consumption during network transmission. At the application layer, the Gossip propagation protocol is used to achieve eventual consistency of data and ensure the reliability of data transmission. In the cluster monitoring process, the effectiveness and dependability of data synchronization are guaranteed by the combination of the Gossip propagation protocol at the application layer and the UDP transmission protocol at the transport layer.

Keywords: gossip protocol; data synchronization; consistency algorithm; high-performance cluster; decentralization

Received: 2024-10-16; **Accepted**: 2024-11-08; **Published Online**: 2024-11-19 14:17

URL: link.cnki.net/urlid/11.2625.V.20241119.0932.001

Foundation item: National Natural Science Foundation of China (U2033203)

* **Corresponding author**. E-mail: hhzhang@travelsky.com.cn