数据因果一致性研究综述

田俊峰[1,2]，王彦鼛[1,2]，何欣枫[1,2]，张俊涛[1,2]，杨万贺[1,2]，庞亚南[1,2]
（1. 河北大学网络空间安全与计算机学院，河北 保定 071002；2. 河北省高可信信息系统重点实验室，河北 保定 071002）

摘 要：数据因果一致性是分布式存储中保障数据一致性的重要方案之一，目前的因果一致性方案研究重点包括时钟方法的优化、协议的设计以及操作事务序列的优化等方面。实际上云环境除了时钟漂移、查询放大等情况之外，还存在木马、不可信第三方等不安全因素，以致于破坏因果一致性元数据、用户操作结果的一致性，甚至影响存储环境的可用性。从一致性存储性能提升和安全保障角度出发，结合区块链等共识机制，对比分析了时钟同步方法、数据复制策略、服务端协议的分析设计以及操作事务序列化等研究方向，总结讨论了它们的的原理、优势、局限性以及安全约束方面的不同效用，进而指出未来的发展趋势和后续研究方向，期望对该领域的研究起到参考和帮助作用。

关键词：数据一致性；因果一致性；分布式存储；混合逻辑时钟；云存储
中图分类号：TP309
文献标识码：A
doi: 10.11959/j.issn.1000−436x.2020055

Survey on the causal consistency of data

TIAN Junfeng[1,2], WANG Yanbiao[1,2], HE Xinfeng[1,2], ZHANG Juntao[1,2], YANG Wanhe[1,2], PANG Ya'nan[1,2]
1. School of Cyber Security and Computer, Hebei University, Baoding 071002, China
2. Key Laboratory on High Trusted Information System in Hebei Province, Baoding 071002, China

Abstract: Causal consistency is one of the important projects to ensure data consistency in distributed storage. The current research focuses on causal consistency including optimization of clock method, design of the protocol and the optimization of operation transaction sequence. In the actual cloud environment, in addition to clock skew and query amplification, there are also insecure factors such as Trojans and untrusted third parties, which will destroy the causal consistency metadata stored by users, and the consistency of user's operating results, even affect the availability of the storage environment. From the perspective of performance improvement and security in distributed storage, the clock synchronization, data replications, analysis and design of server protocol, related research progress of the serialization of operational affairs were introduced combined with consensus mechanisms such as blockchain. At the same time, their principlest, advantages, limitations, and different utilities in terms of security constraints were discussed, and then the future development trends and follow-up research directions were point out at last, which would provide a reference and help for the research in this field.

Key words: data consistency, causal consistency, distributed storage, hybrid logical clock, cloud storage

1 引言

云存储是当今世界互联网服务的重要组成部分，云服务商将数据中心分别部署在不同的地方，为用户提供可扩展的计算、存储等服务[1]。分布式云存储的典型应用之一就是大数据存储，数据一致性则是大数据存储中的基本课题之一。

基于地理复制策略设计分布式云存储系统时，数据一致性协议的设计是基本课题之一[2]。数据一致性理论来自 Brewer[3]提出的 CAP 定理，其包括 3

收稿日期：2019−09−24；修回日期：2020−02−29
基金项目：国家自然科学基金资助项目（No.6180060654）
Foundation Item: The National Natural Science Foundation of China (No.6180060654)

个方面：一致性（consistency），可用性（availability），分区容忍性（partition tolerance）。其中，一致性指同一个数据在集群中的所有节点，同一时刻可供查询的值是否一致；可用性指集群中一部分节点故障后，集群整体是否还能继续处理客户端的更新请求；分区容忍性指系统应确保在部分网络出现异常的情况下仍能正常使用，除非网络整体瘫痪。

随着云存储的不断普及与应用，安全问题已成为制约其进一步发展的重要因素[4]。在云存储平台中，病毒、木马、黑客入侵等不安全因素的存在难免会对用户的操作结果一致性产生影响，例如，用户写入与查询数据过程中，服务器主副节点的数据可能会因节点存在不安全因素被篡改，影响数据一致性操作结果的可信性；节点间同步新数据过程中，节点与节点间、节点内分区间可能由于信道不可信造成同步数据被窃听、篡改、阻碍或时延，造成节点间数据无法正确同步。

当前数据一致性方面的研究都是基于理想环境的，对环境中的不安全因素疏于考虑。在可信云平台日渐成熟的情况下，如何分析研究可信环境下的数据一致性问题具有重要的理论意义和应用价值。对政企用户、云服务商来说，数据一致性是数据安全的基本保障之一，而科学技术部发布的“云计算和大数据”等重点专项 2018 年度项目的基础研究类的创新链也包含“大规模分布式可扩展的数据存储”和“数据副本一致性”，说明数据一致性对大数据安全、国民经济发展，乃至国防安全都是至关重要的。

数据一致性一般分为强一致性和弱一致性[5]。强一致性要求所有数据项的本地更新都立刻在其他副本中可见，对副本中数据的及时更新有着极高的要求[6]，而且该模型的实现要求极高的性能开销。弱一致性只要求不同节点中数据最终一致，对数据的及时更新没有任何要求，难以为用户提供最新的数据[7]。相比于强、弱一致性的局限性，因果一致性属于一种中间一致性模型，该模型不仅能提供数据的及时更新，在因果依赖性的影响可见时，也能保障用户事件的因果序。

在关系型数据库中，数据一致性是指事务执行的结果必须是使数据库从一个一致性状态变到另一个一致性状态，而数据因果一致性主要应用于判别非关系数据库中操作数据条目的前后顺序，即通过对 CAP 定理的权衡，在为用户提供数据存储的同时，保障同一客户端或进程对数据的操作有序。而不对同客户端或线程的数据操作顺序有任何要求。非关系型数据库，即 NoSQL（not only SQL）指的是基于 CAP 定理提供键值存储服务的键值存储数据库，例如 Redis、Mongo DB、MICA[8]等。

影响数据因果一致性的因素有很多。首先，云服务供应商（CSP, cloud service provider）通常将不同的存储节点分布在不同的地理位置，但节点间数据更新的共识则依赖于节点间时钟与状态的及时同步。此外，节点内部数据复制的策略、客户端与服务端对一致性元数据的操作以及用户操作事务的序列化等方面都对分布式云存储中数据因果一致性有着不容忽视的影响。数据因果一致性的 4 个主要方面如图 1 所示。

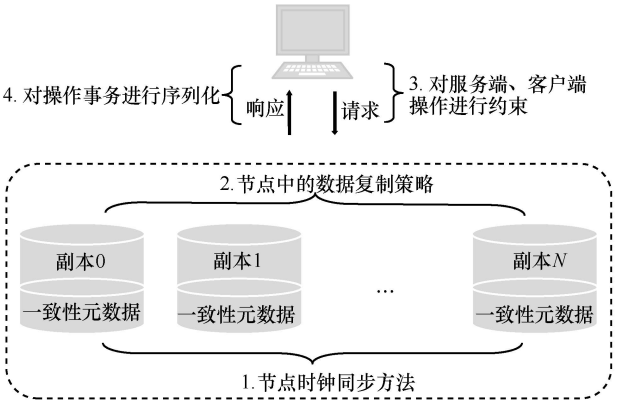

图 1　数据因果一致性的 4 个主要方面

本文通过介绍分布式云存储环境下因果一致性协议中节点间时钟同步方法、节点存储数据的存取与用户操作事务的序列化相关的研究进展，分析并对比经典方案的特点、适用范围，以及在及时更新和性能方面的权衡，对已有工作的优点和局限性进行讨论，进而指出数据因果一致性未来的发展趋势和后续研究方向。

2　副本间的时钟同步

传统的单点存储系统或网络存储系统等都无法同时满足大数据存储在性能、可扩展性和经济成本等方面的要求。目前，构建于大量廉价商用硬件之上的云存储系统一般通过增加存储副本来提升性能、降低成本，因此，副本云存储系统在大数据的存储管理中得到了极其广泛的应用[9]。CSP 通常将为用户提供服务的存储副本分布在世界各地，以满足不同地点用户的需求。各个数据中心之间的时

钟同步一般采用网络时间协议（NTP, network time protocol, ）同步本地物理时钟，这就面临着一个严峻的问题——时钟异常。

时钟异常指的是不同副本之间进行时钟同步时，不同地理位置的副本在通信过程中由于距离差异产生的一定通信时延，使发送请求到收到回复所用的响应时间有所不同，从而造成各个副本时间不同步、NTP 漂移等。而在分布式系统中，数据一致性服务对客户端和服务端间的低开销网络通信有着极高的要求。若用户在当前客户端发送了带有本地时间戳的查询请求，而数据中心的稳定时间小于请求中的时间戳，从而造成查询失败，影响因果一致性系统的可用性。

在使用当前副本的时钟作为判断用户操作顺序的因果一致性方案中，依据副本同步的网络时间为条目分配的时间戳是系统判别用户操作因果序的唯一依据，目前有 2 种常见的时钟同步方法，一种是单纯地使用物理时钟，另一种是混合逻辑时钟。

2.1 物理时钟

物理时钟方案使用单向标量来检查依赖关系，通过提供因果一致性的快照，来保证用户写入事件满足原子性，从而避免写入等待冲突。当副本处理新的用户操作时，依据当前副本的物理时钟为该操作分配一个时间戳 t，若用户操作与其余操作间存在先后顺序，则前者的时间戳一定小于后者。并且当副本的物理时钟为 T 时，副本中存储的所有时间戳小于或等于 T 的数据条目对用户都是可查询的。

在使用物理时钟时还面临着一个严峻的问题——查询放大。单个最终用户请求在服务器中可能会转化为许多因果关系的内部查询，这种现象被称为查询放大[10]，尤其在服务提供商为成千上万个用户提供并发服务时，查询放大的现象更加明显。例如微博、Facebook 等社交系统中，查询放大可能导致多达数千个针对单个最终用户请求的内部查询，即使系统中出现极其轻微的时钟偏差问题，大量内部查询累积的查询时延也会极大地影响客户端查询结果中的因果一致性。

在某些情况下，在客户端写入新的条目时可能需要等待，如果客户端读取写一个时间戳 t 的键，则客户端调用的任何未来写入（PUT）操作都必须具有高于 t 的时间戳，因此，客户端需要将它已经读/写的最后值的时间戳 t 与 PUT 操作一起发送。在写入新值之前，接收此请求的分区将等待，直到其物理时钟高于 t。这种情况造成的时延即为客户端的操作等待时延，在时钟漂移或网络环境造成同步不及时的方案中，操作等待时延情况尤其明显。

在实际应用中，副本间同步的物理时钟只能单调递增，不能倒退回之前的某个状态。例如，在一个由 2 个数据中心 A 和 B 组成的系统中，假设 2 个数据中心的全局稳定时间（GST, global stable time）都是 3，这意味着 2 个数据中心时间戳小于 3 的所有值都是可供用户查询的。另一种情况是假设数据中心 A 中一台服务器的物理时钟后退到 2，此时，如果客户端在该服务器上写入新值，该写入操作无法在副本中进行处理，因为具有时间戳 2 的版本尚未到达数据中心 B，但其物理时钟为 3，高于 2，不同副本间就会产生矛盾，从而影响客户端的正常服务。

GentleRain[11]为构建具有因果一致的分布式数据存储提供了一种全新的设计模型，该模型消除了依赖项检查消息以提高吞吐量，在通过 NTP 服务器更新的物理时钟基础上，仅使用一个单向向量，即全局稳定时间，来满足用户对因果序的要求，减少了系统的存储和通信管理费用。但它的正确性都依赖于单调递增的物理时钟，分区之间的时钟偏移很容易造成 PUT 操作等待时间的增加，若客户端在 t 时刻写入一个键，则接收 PUT 请求的分区需一直等待至其物理时钟高于 t，才能写入最新值。此外，在服务器执行完本地更新后将其发送到远程数据中心进行异步复制的过程中，远程客户端需等待远程数据中心的 GST 大于本地时间戳才能查询到该更新，这导致了系统中更新可见性时延的增加。

2013 年，Du 等[12]对分区数据存储的快照隔离技术进行了研究与讨论，提出了 Clock-SI 模型。该模型并不是从当前系统中使用的集中式时间戳权限中获取快照和提交时间戳，而是基于单纯的物理时钟进行更新。用户通过读取其原始分区的时钟来获取其快照时间戳，并在所有分区上提供与之一致的快照。与使用集中式时间戳权限相比，Clock-SI 具有可用性和性能优势，避免了单点故障和系统潜在的性能瓶颈，提高了吞吐量。2016 年，Tomsic 等[13]提出的 PhysiCS-NMSI 方案解决了 SI 在系统可扩展性方面的瓶颈，该方案的创新之处在于使用松散同步的物理时钟来实现强一致性和因果一致的

快照，并使用单个标量来编码因果关系，其提交协议能有效保证用户操作的原子性，并能避免用户写入冲突。

2014 年，Du 等[14]提出的 Clock-RSM 是一种状态复制因果一致性协议，该协议通过在每个分区使用松散的物理时钟对系统中发出的请求进行完全排序，为跨数据中心提供了一种低时延的一致复制方案。它避免了 Multi-Paxos 机制中需要单一领导者及 Mencius 机制中存在时延提交的问题。但是，在同步过程中，分区之间若存在时钟偏移，物理时钟的特性会导致系统写入或更新时产生较长的响应时间。

在分布式虚拟系统（DVE, distributed virtual environment）中，维护事件执行时间的一致性是为系统中的所有副本提供统一视图的核心元素。然而，由于网络内消息传输时延的波动，一些事件不能在发送副本设定的预期时间内执行，特别是当用户事件之间因果关系较复杂时，会严重影响用户在虚拟环境中的交互体验。针对该场景，Lyu 等[15]于 2017 年设计了一种基于事件之间相关因果关系的新时间戳一致性控制方案。其原理是通过在传统的时间戳一致性控制方法中引入相关的因果事件检测机制，确保可以以正确的顺序处理具有因果关系的事件。通过实验表明，该方案可以大大降低因果违规的发生概率，提高虚拟环境系统的可用性。

2.2　逻辑时钟

由于不同机器上的物理时钟通常情况下很难同步，这会导致无法区分在分布式云存储系统中多个副本的事件时序。Lamport[16]在 1978 年率先给出了分布式云存储系统中事件发生时序问题的解决方案，提出了逻辑时钟的概念。对于任意事件 a 和 b，如果 a→b（“→”表示 a 先于 b 发生），那么 $C(a)<C(b)$，反之不然，因为有可能是并发事件。其中 $C(a)$和 $C(b)$分别表示事件 a 和 b 发生的时钟状态，即时间戳。如果 $C(a)=C(b)$，则事件 a 和 b 肯定不是因果关系，所以它们之间的先后关系其实并不会影响结果，只需要通过一种确定的方式来定义它们之间的先后关系就能得到全序关系。因此，Lamport 逻辑时钟只保证因果关系（偏序）的正确性，不保证绝对时序的正确性。该时钟方法解决的关键问题是约束副本间的交互要在事件的发生顺序上达成一致，而不是对于时间达成一致。逻辑时钟就是基于此思想提出的，指的是分布式系统中用于区分事件发生顺序的时间机制。

2.3　混合逻辑时钟

由于逻辑时钟不能通过 $C(a)$和 $C(b)$的大小推出事件 a 和 b 发生的先后顺序，从而在使用时导致很多困扰。针对此问题，Kulkarni 等[17]提出了一种时钟同步方法——混合逻辑时钟（HLC, hybrid logical clock）。HLC 结合了逻辑时钟和物理时钟，并充分发挥了两者的优势。HLC 在时间戳格式中设计了 2 个分量，第一个是本地物理时钟，第二个是逻辑关系，在提供写入实际时间的基础上也能提供因果序的分辨依据。

2016 年，为了改进读写操作响应时间过长的问题，Roohitavaf 等[18]对原 GentleRain 模型进行了改进，通过使用 HLC 代替物理时钟，降低时钟漂移对系统的影响，同时避免产生额外的处理开销，即 GentleRain+模型。然而，在数据中心之间的同步过程中，仍采用物理时钟作为当前最新条目的指示器，这导致在一些慢副本存在的情况下，系统会产生较高的更新可见性时延。为了解决上述方案的缺陷，随后 Roohitavaf 等[19]提出了 CausalSpartan 模型。该模型使用 HLC 作为客户端与服务端间读写操作、集群副本间同步状态的时钟依据，结合分区间的稳定状态，极好地解决了因时钟漂移与大量数据的重复读写造成的查询放大情况下的时延问题。此外，类似于 CausalSpartan 的 Okapi 模型[20]也使用混合逻辑时钟判别操作之间的因果序，并利用一个二元组——全局稳定向量，追踪当前分区最新条目与稳定状态。但是，该模型限制所有副本都同步更新完成后才允许查询，因此无法避免较高的更新等待时延。

2.4　小结

时钟同步方法基于 NTP 对副本中的时间进行同步，是分布式存储中实现数据一致性的重要依据。安全性方面，无论是单向标量物理时钟还是混合逻辑时钟，均无法避免非法第三方针对时间的攻击，例如中间人攻击等。

此外，在网络与信息传输领域中的数据一致性方面，Liu 等[21]近期设计出一种实用而高效的因果一致性网络领域实现策略——后缀因果一致性策略，采用 Lamport 时间戳标记每个数据分组，每个交换机都更新该时间戳，以反映与数据分组匹配的规则。该策略通过一种管理时间戳的新方法，限制每个网络更新必须涉及的交换机数量，减少了更新

过程中的分组丢失，在一定程度上解决了数据传输过程中难以保障因果一致性的问题。

最终一致性也是分布式地理复制系统中的一种流行一致性模型，虽然可以提供高性能和可用性，但它可能会导致异常。会话保证方法可以消除其中的一部分异常。2019 年，Roohitavaf 等[22]对会话保证进行了改进，能够避免在具有大量分区的系统中出现的级联减速问题，即慢副本产生的一系列副作用；同时配合使用混合逻辑时钟来消除写入操作的时延，以满足会话保证的需要。与最终一致性相比，此方案进行会话保证的开销几乎可以忽略不计。

3 基于地理位置的数据复制策略

分布在不同地理位置的存储副本除了需要依据 NTP 更新当前的副本时钟外，副本之间存储的数据也要定期进行同步。

数据复制策略包括完全地理复制和局部地理复制两类策略。完全地理复制策略将数据存储副本分布在其余地理位置，防止在某个副本出现宕机或自然灾害时无法提供服务等意外情况。局部地理复制策略则规定不同副本存储整体数据的任意子集，并且通过优化副本之间连接方式来提供保障及时更新的数据因果一致性存储服务。

3.1 完全地理复制策略

基于完全地理复制的策略在性能与成本之间的权衡上选择了性能，不仅能为数据提供容灾保障，更能提供高效、稳定的因果一致性元数据存取服务。

Almeida 等[23]在 2013 年对分布式数据库存储中性能、容错和可扩展性 3 个方面属性进行了研究，并提出利用链复制的新变体，实现了较强的因果一致性的方案，该方案能够利用多个副本的存在来均衡负载的读取请求。与最终一致性系统相比，该方案避免了线性化的瓶颈，同时提供了更有竞争力的性能。与其他的因果一致性解决方案相比，链复制的变体能够更有效地处理元数据信息编码操作之间的因果依赖关系。

2015 年，Agrawal 等[24]对分布式存储系统中的数据复制方法进行了详细分类，提出了地理复制模型中的数据一致性方法。例如张倩[25]对副本间复制数据的方式进行了分析，提出一种基于网格来更新副本时间戳的数据复制和选择策略，以及 2 种实现可靠多播的方法和错误处理方法。显式一致性是 Valter 等[26]为地理复制服务的应用程序提出的一种一致性方案，该方案基于应用程序属性定义，加强了最终的一致性，并保留应用程序定义的特定不变量，只要维护指定的不变量，就可以自由地对不同副本的执行操作重新排序。

刘鑫伟[27]在 2016 年提出一种基于读写比例的动态副本一致性策略。该方案使用定时器定时统计当前系统中的读写操作数，通过对写操作的异步更新约束可以大大降低写操作时延。

虽然基于完全地理复制策略能很大地简化因果一致性协议的设计难度，但是该策略要求很高的存储、带宽和硬件成本。为了降低成本，部分学者提出了局部地理复制策略。

3.2 局部地理复制策略

局部地理复制策略中，不同的副本均作为独立单元，每个副本存储用户数据的任意子集。但不同子集之间的数据如何定时同步更新，当前用户如何对其余位置的副本数据完成存取操作都是局部地理复制策略亟待解决的问题。

在线存储服务能够在处于不同地理位置的数据中心之间分发和复制状态，并将用户请求定向到最近或负载最少的站点。虽然有效地确保了低时延响应，但是这种方法与保持副本一致是相互矛盾的，2012 年提出的红蓝一致性模型[28]则解决了此问题，该模型允许强一致操作（红色）和最终一致操作（蓝色）共存，使用一种阴影概念使系统能够最大限度地使用蓝色操作以及标记方法，精确确定某些操作应该分配哪个一致性级别，因此在不牺牲一致性的前提下，该方案明显提升了地理复制策略的性能。

2012 年提出的显式因果一致性[29]概念则对现有因果一致性系统中数据复制策略的更新可见性时延和吞吐量之间的关系进行了权衡，该类方案并不对所有潜在的因果关系进行追踪，只主张追踪一些相对重要数据的因果关系。显式因果关系作为因果关系图的一个子集，大大降低了因果关系图的程度和深度，改善了系统可伸缩性的问题。采用显式因果关系追踪事件的一致性，减少了每次写入操作的依赖项数量，加快了更新传播复制的速度。

2015 年，Shen 等[30]对 Full-Track 算法中更新可见性时延与因果一致性之间的关系进行了讨论，并对 Full-Track 算法本地日志大小和随更新消息发送的控制消息数量进行了进一步的优化，得到

Algorithm Opt-Track 算法。同时给出了 Opt-Track 的一个特例，即完全复制共享内存系统中最优的算法 Opt-Track-crp。该算法不仅能最快地更新本地副本，而且在本地日志和更新消息中使用的控制信息的数量上也是最优的。但是由于使用的是物理时钟，可能会产生时钟漂移，从而影响系统的性能。

此外，Hsu 等[31]对 Approx-Opt-Track 算法进行了性能分析，使用仿真来分析初始 credits 值与元数据大小之间的权衡关系，得到了与 Opt-Track 相比，Approx-Opt-Track 在保证因果一致性的基础上能降低性能开销的结论。并在 2018 年提出了近似因果一致性[32]的概念，以一些可能违反因果一致性的行为为代价减小元数据存储规模。

Crain 等[33]首次提出支持部分复制的因果一致性协议，并在完全复制的情况下提供与完全复制协议相同的性能，同时最小化依赖项元数据并在部分复制情况下进行检查，这些机制对于支持因果一致性的许多协议都是通用的。尽管此协议有助于降低在先前协议中实现部分复制的一些成本，但它并不能完全解决问题。由于协议使用的依赖关系表示不够准确，可能导致错误的依赖关系，并且可能导致读取过时数据。

在基于异步的 SDN 传播更新规则时，属性的强度与由不同操作引起的复杂依赖关系之间的权衡问题，从根本上限制了 SDN 的更新传播速度。为解决此问题，Forster 等[34]于 2016 年提出了相应的解决方案，通过构建及应用所有更新的最小依赖结构，增加了允许并发更新的数量。同时，为满足高可用性要求，该方案避免了在链路状态路由协议收敛期间丢失数据分组，并允许在一次更新中更新来自不同等级的副本。

CausalSpartan[19]方案则将用户发出的更新请求定义为复制数据消息，与其余副本同步用户最新的写入/查询（PUT/GET）操作。若当前副本在一段时间内没有任何客户端的读写操作，则将当前副本的时钟以及最新数据封装为状态消息，与其余副本同步。2017 年提出的使用基于共享树的元数据传播技术的 Saturn 模型[35]，通过管理小块元数据即标签，不仅加快了消息在拓扑网络中的传播速度，也能充分利用部分地理复制的成本优势。

针对不同数据中心的副本同步造成的性能损失，Blotter 模型[36]对弱一致性进行了约束：读取操作只能在本地数据中心上运行，并通过精简的消息更新操作将事务更新到副本的一个子集。在相同代码的基础上，Blotter 模型在数据中心规模上具有较低的开销，同时降低了系统时延并提升了性能。

投机性事务复制（STR, speculative transaction replication）是一种用于部分复制地理分布数据存储的新型投机性事务协议[37]，通过使用松散同步时钟，避免了集中的问题。STR 通过遵循一个新的并发标准投机性快照隔离（SPSI, speculative snapshot isolation），避免了投机导致的并发异常。除了保证已提交事务的快照隔离（SI, snapshot isolation）之外，SPSI 还允许正在执行的事务读取在启动前保存的数据条目，并在原子性操作约束的前提下追踪同一副本在启动前预保存的非冲突事务间的关系。

2018 年，Xiang 等[38]提出的部分地理复制因果一致性方案对全局稳定方法进行了扩展，规定其中每个服务器可以存储数据的任意子集，并且允许客户端与服务器的任何子集进行通信，为用户元数据提供因果一致性保障。该方案中运用到的算法极大地提高了远程服务器中数据更新在本地可见的速度，降低了远程更新可见时延。但是，其时间戳使用物理时钟的方式会造成一些额外的写入与查询操作时延。

通过依赖时间戳对更新进行序列化的 Eunomia 方案[39]中，每个单独的分区都可以为每个更新分配时间戳，而不需要与其他的分区或者 Eunomia 进行同步协调，在实现了高并发性的同时降低了操作时延。该方案使用局部地理复制策略与局部稳定方法解决了吞吐量和可见性时延之间的矛盾，但是在使用 Eunomia 服务时，各个副本每次复制的都是完整的对象集，这就导致其不能充分利用局部地理复制策略的优势。

Fouto 等[40]提出基于局部地理复制策略实现的因果一致性方案，允许每个数据中心复制整个数据库的不同部分。该方案使执行操作获得最大并行性，提高了系统的性能。同时，该方案通过平衡本地操作和远程操作的方式权衡了系统吞吐量和数据的更新可见时延。

Karma 方案[41]的提出解决了由于扩展规模造成的可用性和时延问题，并且具有部分复制策略的成本优势。与之前静态地将客户端绑定到其关联 DC（或环）的因果一致性方案不同，Karma 允许基于可用性或时延为客户端选择任何副本提供服务。同时，Karma 是第一个在持久性 DC 级别存储缓存

和副本之间集成因果一致性的方法。通过实验表明，Karma 的成本远低于完全复制的因果存储方案，并且以相似的成本提供比上述部分复制扩展更高的可用性和更好的性能。

Xiang 等[42]受到复制共享内存系统（如时延复制）的启发提出以副本为中心的因果一致性模型，副本 i 发出的更新被认为与以前在该副本上应用的任何更新有因果关系，不论客户端是否读取之前更新的寄存器。该方案的提出解决了部分复制策略在共享寄存器的复制灵活性和跟踪因果关系的元数据开销之间的权衡问题。直观地说，每个副本维护一个边缘索引向量时间戳，该时间戳保存“共享图”中边缘子集的计数器，该图描述了寄存器在副本之间如何共享。Xiang 等证明了时间戳是最优的，分享的子集图边缘跟踪的正确性是必要的。

Kalavadia 等[43]使用一种自适应分区方法，通过在副本之间复制最频繁的模式，使系统适应工作负载的变化。当新三元组或属性被添加到系统中时，该方案还可以通过利用主−客体连接来确保新三元组在适当分区中的适当位置，从而适应这种情况。Kalavadia 等根据现有的静态分区方案，对该自适应分区方法的性能进行了评估，分析了系统对线性查询、星型查询、管理查询、随机查询等不同查询类型的性能。该方案提升了自适应分区方法的可扩展性，可根据不同的动态类别进行调整，并通过最小化副本间通信实现了更快的查询执行。虽然用于自适应分区的算法执行时间大于静态分区，但是对于缩放数据的静态分区，查询执行时间的增长速度要快得多。与静态分区相比，自适应分区在查询类型上的平均速度提高了 60%。

3.3 小结

数据复制策略是分布式存储系统维护数据正确性的重要手段，本节综述了地理复制方案中 2 种主要的策略，即完全地理复制策略和局部地理复制策略，随后介绍了使用该 2 种策略的研究方案，其中存在的一些局限性介绍如下。

1) 使用完全地理复制策略虽然有着较高的成本支出，但能提供稳定、高效的数据一致性存储。现有研究方案中副本间同步大多采用心跳机制，每次间隔固定的时间与其余副本同步当前副本最新状态。在实际的云环境中，高效的存取服务意味着更高的性能开销，所以心跳机制的间隔时间与性能之间的权衡问题是一个值得研究的问题。

2) 局部地理复制策略中每个副本仅存储整体数据的子集，节约了成本，但对高效的副本连接、状态更新方式有较高的要求，所以该策略对通信、时钟同步的性能以及系统吞吐量有着较高的要求。

3) 副本间复制策略往往是根据分布式集群的拓扑结构，基于心跳机制，定时向相邻副本发送更新。但该方法在副本拓扑结构非常复杂时容易造成较高的通信开销，尤其是完全地理复制策略中，若短时间内用户请求大量更新，在整体分布式环境中就会生成指数级的更新通信。

区块链技术是一种基于拜占庭容错（PBFT, practical Byzantine fault tolerance）机制的共识算法[44]，与数据一致性方案中副本数据复制策略方式类似，副本间对存储的一致性数据子集进行同步的目标是使分布存储环境内不同副本的状态能够达成共识。目前提出或改进的数据复制策略来满足数据一致性的方案均存在一定的性能开销或降低一致性约束的代价，因此借鉴优秀的共识机制，例如区块链、HashGraph 等改进副本间更新数据和达成共识的方式是数据一致性未来研究的重要方向之一。

4 客户端和服务端操作的约束

分布式副本中存储的因果一致性元数据，除了键值（K-V）条目之外，还应包含几个属性，例如操作时间戳、因果一致性的依赖集等。用户对数据的操作包含 2 个过程，即元数据的封装与分析。用户向服务副本写入数据过程即为元数据的封装过程，服务端对元数据进行处理后对该数据条目进行存储。用户向服务端查询数据即为元数据的分析过程，除了对元数据的因果一致性进行校验外，还要返回 K-V 条目及其依赖集。

部分协议除了处理用户的读写请求之外，还进行不同副本之间的状态更新，以期达成副本最新状态的共识，从而为如何处理用户读写请求作为判断依据。更新的结果为在每个副本维护的一组向量，即副本稳定状态，该向量包含副本最新时间戳或最新条目等分量，在副本稳定状态的时刻或之前，所有写入的数据条目对用户都是可查询的。目前针对客户端与服务端操作进行约束的方案包括两类：设置副本稳定状态方案，未设置副本稳定状态方案。

4.1　设置副本稳定状态

GentleRain+[18]基于 GentleRain，使用其中单调的物理时钟作为分配操作时间戳的依据，在存在时钟偏移的情况下依然能提供正确的因果一致性服务。该模型中副本间最新状态是依据副本中的物理时钟进行判别的，虽然能提供正确的存取服务，但无法避免时钟漂移等风险造成的写入、更新时延。同样使用物理时钟判别因果序的还有乐观的因果一致性（OCC, optimistic causal consistency）[2]协议，该协议中远程数据中心的更新可以立即对本地数据中心中的客户端可见。OCC 为每个服务器都配备了一个物理时钟用来提供单调增加的时间戳，以此来传递不同副本间的因果一致性。

为了提升在分区之间存在时钟时延和查询放大的情况下数据因果一致性方案的性能表现，2017 年提出的 CasusalSpartan 模型[19]定义了全新的分区稳定向量（DSV, data center stable vector），在分区中单个查询被放大成许多个内部查询的情况下，能为用户提供可靠的写入和查询服务，降低了客户端的更新等待时间。

图 2 和图 3 分别为 CausalSpartan 模型和 Okapi 模型中的 PUT 和 GET 操作。CasusalSpartan 模型中在客户端设计了一组客户端数据的依赖集时间向量，用户向服务端发送读写请求时都将该向量作为客户端依赖依据；同时，依据服务端响应消息更新本地数据时，更新该向量。此外，CasusalSpartan 模型中还对服务端与其余副本更新状态的消息进行了划分，即复制消息和状态消息。但该模型是在理想情况下实现的，并未提出对第三方篡改等风险的应对策略。

与 CasusalSpartan 模型类似，Okapi 模型[20]使用了 HLC 和全局稳定向量（GSV, global stable vector）的因果一致性协议，该协议设定一个 GSV 来追踪当前分区最新条目的时间戳。与 CasusalSpartan 模型相似的是，它们都对客户端、服务端中数据的处理步骤进行了约束，但 Okapi 模型中提出了一个满足多项查询的方法，允许用户在查询请求向量中发送多个键 $K\{k_1,k_2,\cdots,k_n\}$值，并且在服务端对其处理，同时返回 K 对应的值 $V\{v_1,v_2,\cdots,v_n\}$及其对应的依赖集。但该模型要求，只有所有副本都同步完用户更新后，才允许用户查询，这个约束造成 Okapi 的更新可见性时延变得很高。

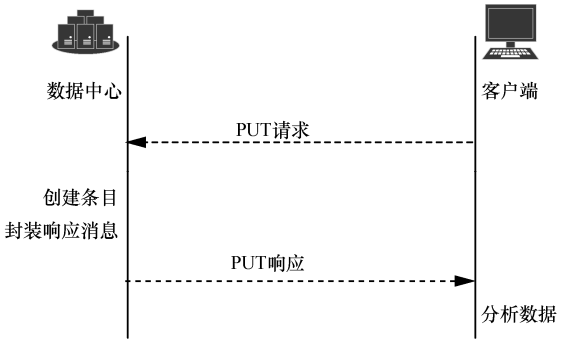

图 2　PUT 过程

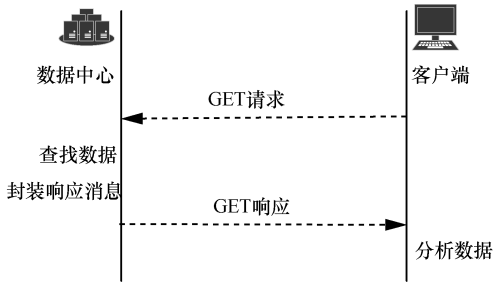

图 3　GET 过程

在分区间设置副本稳定状态的目的是为分布式环境中不同副本的数据一致性约束提供判别依据，即保证分布式系统集群中所有节点的数据同步一致并且能够对某个提案达成共识。区块链共识算法结合密码学技术，可以保证交易的可追溯性、不可篡改性、不可否认性和不可伪造性[45]，如工作量证明（PoW, proof of work）、权益证明（PoS, proof of stake）等均为传统的区块链数据一致性共识算法。2013 年 8 月，比特股（Bitshares）项目基于 PoS 算法提出了授权股份证明算法（DPoS, delegated proof of stake），该算法的基本思路类似于“董事会决策”，即系统中每个节点可以将其持有的股份权益作为选票授予一个代表，获得票数最多且愿意成为代表的前 N 个节点将进入“董事会”，按照既定的时间表轮流对交易进行打包结算并且签署（即生产）新区块。此外，2016 年提出的 Bitcoin-NG[46]将时间切分为不同的时间段，在每一个时间段上由一个领导者负责生成区块并将交易打包，并引入了 2 种不同的区块——用于选举领导者的关键区块和包含交易数据的微区块，进而保障不同角色区块之间的数据一致性共识。

基于拜占庭容错的共识算法也是共识算法研究中的热点问题，如 CDBFT（credit-delegated Byzantine fault tolerance）[47]依据信用情况为区块分配是否可信的身份，并定期执行身份检查。该算法

可以长期保持节点的良好信用状态，从而提高消除恶意节点、提高系统节点间达成数据一致性共识的效率。

近年来，针对区块链共识算法的改进研究越来越深入，其中较有代表性的是基于有向无环图（DAG, directed acyclic graph）、HashGraph 的分布式账本技术。2020 年，高政风等[48]对基于 DAG 的分布式账本共识机制进行了研究，并将其分为 3 类，1) 通过最大权重子树选举主链的主干链 DAG 共识协议；2) 每个节点均维护一条本地信息链的平行链 DAG 共识协议；3) 基于投票机制达成共识的朴素 DAG 共识协议。

采用心跳机制、副本状态时间戳更新等方法在分布式数据一致性方案中设置副本稳定状态，不仅能够降低时间开销，而且能为节点间共识提供判别依据。借鉴主链、平行链 DAG 分布式账本共识算法及区块链算法改进数据一致性共识机制、副本状态同步及数据更新过程是分布式数据一致性未来的重要研究方向之一。

4.2 未设置副本稳定状态

对于副本是否稳定的判别方法除了在每个副本维护稳定状态变量外，还可以持续跟踪用户读取的值来更新分区状态，如 2011 年提出的 COPS[49] 模型与 2013 年提出的 Orbe[50]模型，当客户端为某个键写入新的值时，模型就会将客户端读过的所有值都视为该客户端正在写入的新值的因果依赖关系。而每个副本中存在多个分区，所以每个分区都要发送同步请求到其他分区来检查依赖关系。这种明确的依赖关系跟踪对于同步消息中的消息复杂度要求极高，导致分区间通信开销很大。

为本地对象的读取和更新提供即时且一致的响应的典型方案还有 2015 年提出的 SwiftCloud 模型[51]，不仅能在保证因果一致性的前提下支持客户端的快速读写，而且能够保证服务器端有较高的吞吐量。此外，该模型还减小了元数据的规模，在 3 个数据中心的情况下，其连接的客户端可以很容易地扩展到上千台。但是该模型只适用于理想环境中，并未对实际应用中需要面对的身份认证、数据篡改等安全风险提出相应的保护机制。2018 年，Roohitavaf 等[52]对副本中键值条目的可查询状态与数据一致性的关系进行了研究，并提出一种对数据条目的向量与副本所在分组进行动态跟踪的方案，这种灵活性使应用程序能够在不同的目标之间自由权衡，并为不同的应用程序提供不同的视图，从而使每个应用程序获得最佳性能。

4.3 小结

对用户操作与副本中的同步时钟方法进行优化，是降低时延、解决时间同步风险的高效方法。而针对数据因果一致性中客户端、服务端处理数据的步骤进行约束，是提升性能、降低查询放大等情况造成的时延的重要手段。本节通过对是否标识副本稳定状态的不同研究方案进行综述，总结了现有经典方案中对客户端、服务端进行约束以保证因果一致性的几种方法，但其中存在一定的局限性，介绍如下。

1) 设计客户端、服务端保障因果一致性的操作步骤是降低操作时延、更新等待时间的必要手段。但在一致性元数据封装与分析过程中增加操作步骤无疑会带来更高的性能开销，结合身份认证、时钟方法优化客户端与服务端间的操作是一个值得研究的课题。

2) 服务器副本间同步最新状态过程中，除了最新写入的数据条目外，还应包含副本最新时钟信息，因此相比于单纯地使用时钟信息做额外副本状态信息的方案，同时处理数据与时钟状态的方案会有较高的性能消耗。如何在服务端之间同步数据过程中对副本状态进行标识是数据因果一致性研究中的热点课题。

3) 对客户端、服务端之间存取数据的步骤进行数据因果一致性约束，是针对数据处理过程中的细粒度操作。但该类方案在实际的云环境中存在数据篡改、身份冒用等风险时无法提供数据完整性保障，例如 CausalSpartan、Okapi 等方案中，客户端依据收到的响应消息更新自身依赖集，该响应消息作为唯一依据，无法保证传递消息内容是否完整，因此客户端、服务端也就不能提供准确可靠的因果一致性存取服务。

4) 与区块链共识算法在区块链接、同步的过程中采用不同机制进行同步以达成共识的过程相似，数据因果一致性模型中对数据副本进行同步为用户数据及其因果序提供一致性约束，也是为了在分布式集群中将不同副本同步为一个共识形态。但区块链共识机制往往是基于更高风险的复杂环境，需使用密码学加密并提供工作量或权益等证明机制以降低安全风险，而现有的数据一致性研究方案中鲜有对因果一致性提出安全保障的方案。但实际的

云环境中存在大量病毒、非法第三方等不安全因素。因此，应考虑风险环境、用户操作不可信等因素，对客户端、副本进行安全认证，例如结合密码学、可信认证、身份认证机制等设计安全约束。

目前的研究成果大多专注于降低性能开销与操作时延，但存在完整性、安全等方面的局限性，如用户一般将关键数据分散地存储在云存储节点上，而存储节点可能遭到破坏或者窃听，在实际应用中，由于云存储用户不会在本地保存副本，就失去了对数据的直接管理和控制。分布式存储数据主要面临数据破坏、云服务商恶意隐瞒以及用户隐私泄露等安全威胁，对用户数据、云服务商的一致性约束造成影响。对客户端、服务端的因果一致性操作进行序列化也是为处理用户请求的过程设计因果一致性约束的方法之一，具体在第 5 节中进行介绍。

5　操作事务的序列化

分布式存储系统中，用户的读写操作、客户端的读写操作必须由系统分配一个序列来执行。当一个客户端对一个数据条目进行了更新或者写入操作时，其余的客户端同时来读取这个条目，就会造成该条目的更新查询丢失问题。若数据条目在被处理的过程中分布式系统设置锁机制，只对当前客户端提供读写权限，即对分布式存储系统的操作事务进行序列化，就可避免丢失更新或脏数据等风险的出现。

2015 年，事务存储器的概念由 Dziuma 等[53]首次提出，规定用户只能读取已处理事务写完的数据条目。数据一致性一直是分布式存储系统中的一个热点话题，设计一致性约束下的事务序列化模型尤其重要。Viotti 等[54]划分和总结了事务型数据库和非事务型数据库中的数据一致性定义。RedBlue[55]一致性模型将操作集分为强一致性和弱一致性这 2 种情形，约束强操作、增加弱操作空间，并使不同类型的操作选择不同的一致性级别，从而降低系统时延。Andrea 等[56]通过设计一个保证一致性操作的原子可见性的框架，规定某一事务的全部更新必须对其他副本明显可见。

为使事务性存储系统开销更小，2018 年，Zhang 等[57]提出一个消除复制协议中一致性的新方法，同时仍然为应用程序提供分布式事务的强一致性，即不一致复制的事务应用程序协议（TAPIR, transactional application protocol for inconsistent replication），这是第一个使用新型复制协议的事务协议，称为不一致复制，它提供了非一致性下的容错性。该方案通过仅在事务协议中强制执行强一致性的约束，可以在单次查询操作中提交分布式事务并在没有中央协调者的情况下对其进行排序。与传统系统相比，TAPIR 能提供更低的时延和更好的吞吐量。

为在上述所提方法的基础上实现一种高可用的事务协议，Cure 模型[58]通过一个交互式的事务接口来保证因果一致性和支持 CRDT（conflictfree replicated data type），在保持高度可用性的同时实现了最强大的语义。此外，它采用了存有每个数据中心条目的矢量时钟，实现了对因果一致性更准确的追踪，虽降低了更新可见性时延，但是由于需要管理大量元数据而增加了计算和存储开销，进一步牺牲了吞吐量。为确保事务的原子可见性，Bailis 等[59]提出了一种新的数据一致性事务读取模型——原子读取隔离模型。该模型保证事务更新完成后才对其余事务可见，但在系统中存在较高争用请求的情况下，模型会造成额外的性能开销。

为解决只读事务算法的性能和可见性时延之间存在的权衡问题，SNOW（Strict serializability, Non-blocking operations, One-response from each shard, and compatibility with conflicting Write transactions）定理[60]指出，只读事务算法不可能提供所有 4 个理想的属性：严格的串行性、非阻塞操作、每个切分只有一次响应以及与冲突写事务的兼容性。其中与性能相关的属性是严格的串行性（一致性的最强形式）和与冲突写事务的兼容性（表明系统中还有哪些其他类型的事务）。该方案虽然使用其他 3 个属性不可能实现严格的串行化，但是可以实现一个稍微弱一些的一致性模型，称为流程顺序串行化，并将其定义为 SNOW 最优性，作为检查现有系统并确定其只读事务的时延是否可以改进的强有力依据。Saturn[35]是一种依靠元数据来传递因果序的数据一致性协议，该协议向每个数据中心提供一个追踪因果一致性的序列化标签。此外，它还使用了一组合作的序列化器件来加快标签在拓扑网络中的传播速度，使在地理上彼此接近的 2 个数据中心之间快速地建立起元路径传播路径。此协议在保证吞吐量的前提下也能实现良好的更新可见性，同时可以充分利用部分地理复制的成本优势。

为允许应用程序通过分片有效地在复制站点

内实现扩展，Wren[61]模型在保证因果一致性的同时，通过为每个数据中心安装一个快照实现了事务的非阻塞读操作，该快照是由本地数据中心的每个分区安装的一个新的因果快照和一个客户端缓存组成。该模型不考虑系统的规模，而仅仅使用 2 个标量时间戳来跟踪因果性，一个时间戳跟踪本地项的依赖关系，另一个时间戳跟踪远程项的依赖关系。与之前的设计相比，2 个标量时间戳大大提高了存取效率和可扩展性。但是，由于时间戳的选取问题，增加了更新可见时延。

2018 年，Didona 等[62]指出“时延最优”只读事务会在写操作上带来额外的开销，这种开销非常高，甚至在以读为主的工作负载中也降低性能表现。并且提出了实现“近似时延最优”的只读事务协议，但不会增加由时延最优协议引起的任何开销。该协议是非阻塞的，并且针对单条数据条目、需要两轮客户端和服务器之间的通信。但该协议中时延最优的只读事务对写施加的额外开销是固有的不可避免的，并且这种开销随着客户端数量的增加而呈线性增长趋势。

为减少时钟偏移对现有的事务模型造成的影响，Roohitavaf 等[63]在 CausalSpartan 模型的基础上设计了 CausalSpartanX 模型，该模型提供了一种非阻塞的因果一致性只读事务，只需要在客户端和服务器之间进行一轮通信，并且与事务无关分区的慢化也不会影响事务的响应时间。无论读写事务是否涉及慢副本，这种改进都非常有意义，能有效地降低响应时间和时延。

操作事务的序列化方案参考了关系型数据库中事务的原子性，在用户批量的请求到达服务端请求处理时，对操作进行原子性约束，并进行序列化。本节对不同方案中将事务序列化来保障数据因果一致性的方法进行了综述，与在客户端、服务端设计满足因果一致性约束操作的方案类似，在事务的序列化研究方案中大多集中于对数据一致性与性能之间的权衡问题上，对实际环境中的安全风险鲜有考虑。

在相关技术方面也有商业化产品面市，如满足数据一致性约束的商业化数据库产品之一 MongoDB，其中设计了读写锁机制来保障，即通过写关注（write concern）、读关注（read concern）和读优先（read preference）机制来为用户操作的事务提供序列化约束。MongoDB 客户端利用会话的概念捕获因果关系，并将操作时间戳作为判别因果序的依据，不同客户端也可以共享其余客户端的因果序令牌，从而为不同线程中的副本和用户提供高效的因果一致性约束。

6 未来研究展望

本文主要围绕数据因果一致性的最新研究内容展开综述，介绍了近年来具有代表性的因果一致性模型。通过分析可以看出，现有数据存储中对因果一致性的保障仍然存在一些不足，未来的科研工作可以更多地关注以下几点。

1) 因果一致性时钟同步方法中，使用物理时钟容易受到时钟异常的影响，但混合逻辑时钟结合了物理时钟和逻辑时钟两者，主要的局限性介绍如下。

① 单纯的物理时钟同步方法仅需对时间戳标量进行校验，混合逻辑时钟除了对物理时钟进行校验外还需要对逻辑时钟进行校验，造成一定的性能开销。

② 若因果一致性协议单纯依靠时钟来分辨因果序，在系统中存在病毒、身份验证信息漏洞等风险时，NTP 可能会被破坏，从而影响系统可用性。针对时钟同步过程中的安全风险，结合密码学设计时间戳信息可信传输、时钟信息的安全验证等方法应是未来研究工作的重要方向之一。

③ 虽然 HLC 减小了与真实时钟间的差距，但混合逻辑时钟中单纯依据逻辑时钟修正物理环境中的漂移具有一定的不确定性。结合更多时钟依赖向量、更安全的时钟信息约束来保证分区间时钟同步的准确性应是未来研究的关键方向之一。

2) 基于地理位置的复制策略中，完全地理复制要求较高的成本，而局部地理复制策略则对通信、性能开销有着极高的要求，两者的局限性介绍如下。

① 局部地理复制策略中每个副本仅存储整体数据的子集，副本之间只能通过优化的拓扑结构及与高频率的更新和转发来保证数据因果一致性。因此，对局部复制策略的优化研究还有很大的研究空间，如对副本数据子集更新规模与数据更新间隔之间的权衡，结合数据操作事务的序列化方案优化事务的原子性约束。

② 心跳机制的间隔、副本数据复制消息的内容关乎分布式存储系统的更新可见速度，因此，基于实际环境中不同用户的需求，对副本数据复制消

息内容和心跳机制进行优化，为分布式存储设计更高效、低性能的因果一致性约束，应是未来研究的关键。

3) 对客户端操作、服务端处理操作的过程进行约束的方案是从通信与元数据格式转换的角度对数据因果一致性提供操作保障的。但该类方案通常从发送请求与处理请求的角度出发，存在以下几个方面的局限性。

① 结合时钟同步方法对副本的最新状态进行标识、用户发送请求与处理请求过程中都对依赖集进行更新均有较大的性能开销，因此未来研究重点要着重于设计低时延、低开销的因果一致性协议。

② 现有研究方案极少对实际环境中的安全风险进行讨论，后续研究的重点应集中在结合密码技术、可信认证机制对副本、客户端的操作设计安全约束等方面。

③ 在安全风险方面，尚未提出对因果一致性存储的身份认证、权限控制以及结合数据完整性约束的方案，复杂的云环境导致目前的数据因果一致性方案实用性较差，未来工作应多关注于安全约束下的因果一致性研究。

4) 依据 CAP 定理对事务进行序列化的方案通过对数据库操作事务进行约束，为分布式存储系统中副本与客户端的操作提供了数据因果一致性的保障，但对事务的完整验证、完全和部分地理复制策略中数据操作事务提供不同程度的一致性约束等工作有待进一步深入。

综上，为全面降低分布式存储中的操作时延、更新时延、成本和性能开销，提升因果一致性存储的安全程度，应该将优化时钟同步方法，为数据同步、元数据存取事务和副本状态同步方法设计安全约束等方面的研究方法进行结合。目前研究通常专注于数据因果一致性中某个特定的需求，例如性能、成本，并为此提出大量的优化方案。然而，针对每个需求分别提出的优化方案在实际应用中的可行性较差，而且在单独部署的同时也极有可能引入额外的安全风险。因此，未来研究的首要任务是设计全面完整的安全解决方案，以满足实际云存储面临的多样需求。

7　结束语

低廉的存储成本与高效的通信传输促使大数据存储产业蓬勃发展给大数据存储环境中的时钟同步、数据复制及事务序列化技术也带来了特有的性能和安全问题。特别是时钟同步方法、元数据存取间的因果一致性约束，引起了人们对数据因果一致性的普遍关注。通过深入分析与对比可知，数据因果一致性工作在国内外已取得较好的研究成果，但是仍有许多遗留问题尚待探讨，需要综合考虑多种因素，不断优化数据存取、同步策略，加强安全约束。本文重点介绍了分布式数据存储中 4 类影响数据因果一致性的影响因素及其特点，然后分别围绕时钟同步、数据复制、元数据存取和事务序列化展开综述，以期能够为数据因果一致性的未来研究做出一些有益的探索。

参考文献：

[1] 崔勇, 宋健, 缪葱葱, 等. 移动云计算研究进展与趋势[J]. 计算机学报, 2017, 40(2): 273-295.
CUI Y, SONG J, MIAO C C, et al. Mobile cloud computing research progress and trends[J]. Chinese Journal of Computers, 2017, 40(2): 273-295.

[2] SPIROVSKA K, DIDONA D, ZWAENEPOEL W. Optimistic causal consistency for geo-replicated key-value stores[C]//International Conference on Distributed Computing Systems. Piscataway: IEEE Press, 2017: 2626-2629.

[3] BREWER E A. Towards robust distributed systems[C]//The Nineteenth Annual ACM Symposium on Principles of Distributed Computing. New York: ACM Press, 2000: 477-502.

[4] 张玉清, 王晓菲, 刘雪峰, 等. 云计算环境安全综述. 软件学报, 2016, 27(6): 1328-1348.
ZHANG Y Q, WANG X F, LIU X F, et al. Survey on cloud computing security[J]. Journal of Software, 2016, 27(6): 1328-1348.

[5] AGUILERA M, TERRY B. The many faces of consistency[J]. Bulletin of the IEEE Computer Society Technical Committee on Data Engineering, 2016, 39(1): 3-13.

[6] BALEGAS V, LI C, NAJAFZADEH M, et al. Geo-replication: fast if possible, consistent if necessary[J]. IEEE Data Engineering Bulletin, Special Issue on Data Consistency across Research Communities, 2016, 39(1): 12.

[7] BAILIS P, GHODSI A. Eventual consistency today: limitations, extensions, and beyond[J]. Communications of the Association for Computing Machinery, 2013, 56(5): 55-63.

[8] LI M H, HAN D, DAVID G, et al. MICA: a holistic approach to fast in-memory key-value storage[C]//Usenix Conference on Networked Systems Design & Implementation. Berkeley: USENIX Association, 2014: 429-444.

[9] 王意洁, 许方亮, 裴晓强. 分布式存储中的纠删码容错技术研究[J]. 计算机学报, 2017, 40(1): 273-295.
WANG Y J, XU F L, PEI X Q. Research on code-based fault-tolerant technology for distributed storage[J]. Chinese Journal of Computers, 2017, 40(1): 273-295.

[10] AJOUX P, BRONSON N, KUMAR S, et al. Challenges to adopting stronger consistency at scale[C]//Usenix Conference on Hot Topics in

Operating Systems. Berkeley: USENIX Association, 2015: 1-13.

[11] DU J Q, IORGULESCU C, ROY A, et al. Gentlerain: cheap and scalable causal consistency with physical clocks[C]//The ACM Symposium on Cloud Computing. New York: ACM Press, 2014: 1-13.

[12] DU J Q, ELNIKETY S, ZWAENEPOEL W. Clock-SI: snapshot isolation for partitioned data stores using loosely synchronized clocks[C]//2013 IEEE 32nd International Symposium on Reliable Distributed Systems. Piscataway: IEEE Press, 2013: 173-184.

[13] TOMSIC A, CRAIN T, SHAPIRO M. PhysiCS-NMSI: efficient consistent snapshots for scalable snapshot isolation[C]//The 2nd Workshop on Principles and Practice of Consistency for Distributed Data. New York: ACM Press, 2016: 1-4.

[14] DU J Q, SCIASCIA D, ELNIKETY S, et al. Clock-RSM: low-latency inter-datacenter state machine replication using loosely synchronized physical clocks[C]//The 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks. Piscataway: IEEE Press, 2014: 343-354.

[15] LYU Y X, ZHANG W. A relevant casual relation based timestamp order consistency control method in DVE systems[C]//IEEE 3rd Information Technology and Mechatronics Engineering Conference. Piscataway: IEEE Press, 2017: 27-31.

[16] LAMPORT L. Time, clocks, and the orderingof events in a distributed system[J]. Communications of the Association for Computing Machinery, 1978, 21(7): 558-565.

[17] KULKARNI S, DEMIRBAS M, MADAPPA D, et al. Logical physical clocks[C]//International Conference on Principles of Distributed Systems. Berlin: Springer, 2014: 17-32.

[18] ROOHITAVAF M, KULKARNI S. GentleRain+: making gentlerainrobust on clock anomalies[J]. arXiv Preprint, abs/1612. 05205, 2016.

[19] ROOHITAVAF M, DEMIRBAS M, KULKARNI S. CausalSpartan: causal consistency for distributed data stores using hybrid logical clocks[C]//Reliable Distributed Systems. Piscataway: IEEE Press, 2017: 184-193.

[20] DIDONA D, SPIROVSKA K, ZWAENEPOEL W. Okapi: causally consistent geo-replication made faster, cheaper and more available[J]. arXiv Preprint, abs/1702.04263, 2017.

[21] LIU S, BENSON T, REITER M. Efficient and safe network updates with suffix causal consistency[C]//European Conference on Computer Systems. New York: ACM Press, 2019: 1-15.

[22] ROOHITAVAF M, AHN J S, KANG W, et al. Session guarantees with raft and hybrid logical clocks[C]// International Conference on Distributed Computing Systems. Piscataway: IEEE Press, 2019: 100-109.

[23] ALMEIDA S, LEITAO J, LUÍS E T. et al. ChainReaction: a causal+ consistent datastore based on chain replication[C]//European Conference on Computer Systems. New York: ACM Press, 2013: 85-98.

[24] AGRAWAL D, ABBADI A E, SALEM K. A taxonomy of partitioned replicated cloud-based database systems[J]. Bulletin of the IEEE Computer Society Technical Committee on Data Engineering, 2015, 38(1): 4-9.

[25] 张倩. 分布式存储系统中用户数据一致性分析及研究[D]. 西安: 西安工业大学, 2015.
ZHANG Q. Analysis and research of user data consistency in distributed storage system[D]. Xi'an: Xi'an Technological University, 2015.

[26] VALTER B, NAJAFZADEH, DUARTE M, et al. Putting the consistency back into eventual consistency[C]//European Conference on Computer Systems. New York: ACM Press, 2015: 1-16.

[27] 刘鑫伟. 基于 Ceph 分布式存储系统副本一致性研究[D]. 武汉: 华中科技大学, 2016.
LIU X W. Research on replica consistency based on ceph distributed storage system[D]. Wuhan: Huazhong University of Science and Technology, 2016.

[28] LI C, PORTO D, CLEMENT A, et al. Making geo-replicated systems fast as possible, consistent when necessary[C]//The 10th USENIX conference on Operating Systems Design and Implementation. New York: ACM Press, 2012: 265-278.

[29] BAILIS P, FEKETE A, GHODSI A, et al. The potential dangers of causal consistency and an explicit solution[C]//Proceedings of the Third ACM Symposium on Cloud Computing. New York: ACM Press, 2012: 1-7.

[30] SHEN M, KSHEMKALYANI A D, HSU T Y. Causal consistency for geo-replicated cloud storage under partial replication[C]//The 2015 IEEE International Parallel and Distributed Processing Symposium Workshop. Los Alamitos: IEEE Computer Society, 2015: 509-518.

[31] HSU T Y, KSHEMKALYANI A D. Performance of approximate causal consistency for partially replicated systems[C]//International Workshop on Adaptive Resource Management & Scheduling for Cloud Computing. New York: ACM Press, 2016.

[32] HSU T Y, KSHEMKALYANI A D. Value the recent past: approximate causal consistency for partially replicated systems[J]. IEEE Transactions on Parallel and Distributed Systems, 2018, 29(1): 212-225.

[33] CRAIN T, SHAPIRO M. Designing a causally consistent protocol for geo-distributed partial replication[C]//European Conference on Computer Systems. New York: ACM Press, 2015: 1-4.

[34] FORSTER K, MAHAJAN R, WATTENHOFER R. Consistent updates in software defined networks: On dependencies, loop freedom, and blackholes[C]//2016 IFIP Networking Conference and Workshops. Piscataway: IEEE Press, 2016: 1-9.

[35] BRAVO M, RODRIGUES L. Saturn: a distributed metadata service for causal consistency[C]//European Conference on Computer Systems. New York: ACM Press, 2017: 111-126.

[36] MONIZ H, LEITAO J, DIAS R J, et al. Blotter: low latency transactions for geo-replicated storage[C]//The 26th International Conference. International World Wide Web Conferences Steering Committee. New York: ACM Press, 2017: 263-272.

[37] LI Z M, ROY P, ROMANO P. Exploiting speculation in partially replicated transactional data stores[C]//Proceedings of the 2017 Symposium on Cloud Computing. New York: ACM Press, 2017: 640.

[38] XIANG Z, VAIDYA N H. Global stabilization for causally consistent partial replication[J]. avXiv Preprint, abs/1803.05575, 2018.

[39] GUNAWARDHANA C, BRAVO M, RODRIGUES L. Unobtrusive deferred update stabilization for efficient geo-replication[C]//USENIX Annual Technical Conference. Berkeley: USENIX Association, 2017: 83-95.

[40] FOUTO P, LEITAO J, PREGUICA N. Practical and fast causal consistent partial geo-replication[C]//17th International Symposium on Network Computing and Applications. Piscataway: IEEE Press, 2018: 1-10.

[41] TARIQ M, SHANKARANARAYANAN P N, SANJAY R, et al. Karma: cost-effective geo-replicated cloud storage with dynamic en-

forcement of causal consistency[J]. IEEE Transactions on Cloud Computing, 2018(99): 1.

[42] XIANG Z L, VAIDYA N. Brief announcement: partially replicated causally consistent shared memory[C]//Proceedings of the 2018 ACM Symposium on Principles of Distributed Computing. New York: ACM Press, 2018: 273-275.

[43] KALAVADIA B, BHATIA T, PADIYA T, et al. Adaptive partitioning using partial replication for sensor data[C]//International Conference on Distributed Computing and Internet Technology. Berlin: Springer, . 2019: 260-269.

[44] 贺海武, 延安, 陈泽华. 基于区块链的智能合约技术与应用综述[J]. 计算机研究与发展, 2018, 55(11): 2452-2466.
HE H W, YAN A, CHEN Z H. Survey of smart contract technology and application based on blockchain[J]. Journal of Computer Research and Development, 2018, 55(11): 2452-2466.

[45] 韩璇, 袁勇, 王飞跃. 区块链安全问题: 研究现状与展望[J]. 自动化学报, 2019, 45(1): 206-225.
HAN X, YUAN Y, WANG F Y. Security problems on blockchain: the state of the art and future trends[J]. Acta Automatica Sinica, 2019, 45(1): 206-225.

[46] EYAL I, GENCER A E, SIRER E G, et al. BitcoinNG: a scalable blockchain protocol[C]//The 13th USENIX Conference on Networked Systems Design and Implementation. Berkeley: USENIX Association, 2016: 45-59.

[47] WANG Y H, CAI S B, LIN C L, et al. Study of blockchains's consensus mechanism based on credit[J]. IEEE Access, 2019(7): 1024-1023.

[48] 高政风, 郑继来, 汤舒扬, 等. 基于DAG的分布式账本共识机制研究[J]. 软件学报, 2020, 31(4): 1-20.
GAO Z F, ZHENG J L, TANG S Y, et al. State-of-the-art survey of consensus mechanisms on DAG-based distributed ledger[J]. Journal of Software, 2020, 31(4):1-20.

[49] LLOYD W, FREEDMAN M, KAMINSKY M, et al. Don't settle for eventual: Scalable causal consistency for wide-area storage with cops[C]//The Twenty-Third ACM Symposium on Operating Systems Principles. New York: ACM Press, 2011: 401-416.

[50] DU J Q, ELNIKETY S, ROY A, et al. Orbe: scalable causal consistency using dependency matrices and physical clocks[C]//The 4th Annual Symposium on Cloud Computing. New York: ACM Press, 2013: 1-14.

[51] ZAWIRSKI M, PREGUICA N M, DUARTE S, et al. Write fast, read in the past: causal consistency for client-side applications[C]//Proceedings of the 16th Annual Middleware Conference. New York: ACM Press, 2015: 75-87.

[52] ROOHITAVAF M, KULKARNI S. Toward adaptive causal consistency for replicated data stores[J]. avXiv Preprint, abs/1803.08609, 2018.

[53] DZIUMA D, FATOUROU P, KANELLOU E. Consistency for transactional memory computing[M]. Berlin: Springer International Publishing, 2015.

[54] VIOTTI P, VUKOLIC M. Consistency in non-transactional distributed storage systems[J]. ACM Computing Surveys, 2016, 49(1): 1-34.

[55] ANDREA C, BERNARDI G, GOTSMAN A. A framework for transactional consistency models with atomicvisibility[C]//The 26th International Conference on Concurrency Theory. Berlin: Springer, 2015: 58-71.

[56] ZHANG I, SHARMA N K, SZEKERES A D, et al. Building consistent transactions with inconsistent replication[J]. ACM Transactions on Computer Systems, 2018, 35(4): 1-37.

[57] AKKOORATH D D, TOMSIC A Z, BRAVO M, et al. Cure: strong semantics meets high availability and low latency[C]//36th International Conference on Distributed Computing Systems. Piscataway: IEEE Press, 2016: 405-414.

[58] BAILIS P, FEKETE A, GHODSI A, et al. Scalable atomic visibility with RAMP transactions[J]. ACM Transactions on Database Systems, 2016, 41(3): 1-45.

[59] LU H, HODSDON C, NGO K, et al. The SNOW theorem and latency-optimal read-only transactions[C]//Usenix Conference on Operating Systems Design & Implementation. Berkeley: USENIX Association, 2016: 133-150.

[60] SPIROVSKA K, DIDONA D, ZWAENEPOEL W. Wren: nonblocking reads in a partitioned transactional causally consistent data store[C]//The 48th International Conference on Dependable Systems and Networks. Piscataway: IEEE Press, 2018: 1-2.

[61] DIDONA D, GUERRAOUI R, WANG J J, et al. Causal consistency and latency optimality: friend or foe?[C]//Proceedings of the VLDB Endowment. New York: ACM Press, 2018: 1618-1632.

[62] ROOHITAVAF M, DEMIRBAS M, KULKARNI S. CausalSpartanX: causal consistency and non-blocking read-only transactions[J]. avXiv Preprint, abs/1812.07123, 2018.

[作者简介]

田俊峰（1965− ），男，河北保定人，博士，河北大学教授、博士生导师，主要研究方向为信息安全与分布式计算。

王彦翾（1994− ），男，河北邢台人，河北大学硕士生，主要研究方向为信息安全与数据一致性。

何欣枫（1976− ），男，河北保定人，河北大学副教授，主要研究方向为云计算、可信计算。

张俊涛（1995− ），男，河北保定人，河北大学硕士生，主要研究方向为信息安全、数据一致性。

杨万贺（1996− ），男，河北保定人，河北大学硕士生，主要研究方向为信息安全、数据一致性。

庞亚南（1995− ），女，河北衡水人，河北大学硕士生，主要研究方向为信息安全、数据一致性。