(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 105359113 B
(45)授权公告日 2018.05.22

(21)申请号 201480026282.0
(22)申请日 2014.03.13
(65)同一申请的已公布的文献号
申请公布号 CN 105359113 A
(43)申请公布日 2016.02.24
(30)优先权数据
61/784,282 2013.03.14 US
(85)PCT国际申请进入国家阶段日
2015.11.09
(86)PCT国际申请的申请数据
PCT/US2014/026689 2014.03.13
(87)PCT国际申请的公布数据
WO2014/151928 EN 2014.09.25
(73)专利权人 加州理工学院
地址 美国加利福尼亚州
(72)发明人 昭·李 特蕾西·C·何
德雷克·莱昂 宏毅·姚
(74)专利代理机构 北京安信方达知识产权代理有限公司 11262
代理人 陆建萍 郑霞
(51)Int.Cl.
G06F 12/02(2006.01)
(56)对比文件
CN 101627650 A,2010.01.13,
US 2010115216 A1,2010.05.06,
US 2001044879 A1,2001.11.22,
US 2010235843 A1,2010.09.16,
审查员 陈恺

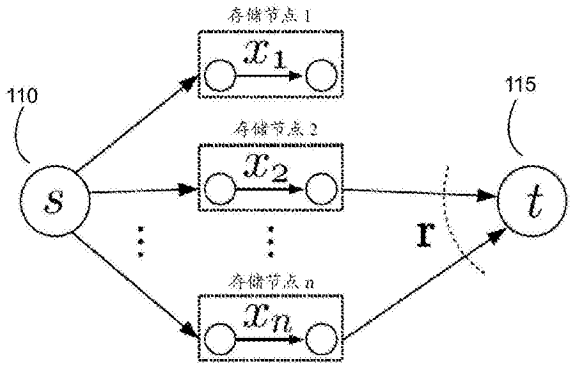
权利要求书3页 说明书13页 附图16页

(54)发明名称

用于异构系统的分布式存储分配

(57)摘要

描述了在基于计算机的分布式存储系统中存储预算的分配,其中相关的基于计算机的存储节点具有异构访问概率。问题是,跨越可用的基于计算机的节点,分配给定的存储预算,以便用更高的可靠性(例如,增加了存储预算被恢复的概率)存储单位大小的数据对象(例如,文件)。提出了用于优化一个或多个类别的分配的有效的算法。提出了基本的一级对称分配,其中存储预算均匀散布在适当选择的节点子集上。此外,提出了二级对称分配,其中预算被分为两部分,每个部分均匀散布在基于计算机的存储节点的不同子集上,使得分配到第一子集中的每个节点的量是第二子集中每个节点的量的两倍。以三级对称分配和推广的k级对称分配,提供了二级对称分配的进一步的延伸。



CN 105359113 B

1.一种基于计算机的方法，其用于在异构存储系统中分配存储，所述基于计算机的方法包括：

提供已知异构可靠性的一组硬件存储节点；

通过计算机，提供目标函数；

通过计算机，提供约束；

基于所述约束和所述目标函数，通过计算机，选择所述一组硬件存储节点中的一个或多个不相交的子集；以及

基于所述选择，通过计算机，将存储量分配到所述一个或多个不相交的子集，

其中所述分配是基于所述异构可靠性和所述约束的，并且所述分配是通过将所述存储量散布到所述一个或多个不相交的子集上获得的，使得所述一个或多个不相交的子集中的每个子集的每个硬件存储节点具有相同的分配存储量，所述相同的分配存储量不同于被分配到不同子集的硬件存储节点的量。

2.根据权利要求1所述的基于计算机的方法，其中所述目标函数对应于以下中的一个或多个：a)数据检索的可靠性，b)数据的可访问性以及c)存储的成本。

3.根据权利要求1或2所述的基于计算机的方法，其中所述约束对应于以下中的一个或多个：a)存储量小于或等于来自所述一组硬件存储节点的总可用存储，b)存储的成本，c)数据检索的可靠性以及d)数据的可访问性。

4.根据权利要求1或2所述的基于计算机的方法，还包括：

通过计算机，提供文件；

通过计算机，对所述文件编码；

基于所述编码，通过计算机，获得对应于预算的总大小的多个编码的文件；以及

基于所述获得，通过计算机，将所述编码的文件存储在所选择的一个或多个不相交的子集上，

其中，每个编码的文件的大小与由所述分配确定的分配的预算量相匹配。

5.根据权利要求4所述的基于计算机的方法，其中，所述编码是使用以下中的一个来实施的：a)最大距离可分(MDS)码和b)随机线性编码。

6.根据权利要求1-2和5中任一项所述的基于计算机的方法，其中所述一个或多个不相交的子集中的子集的硬件存储节点包括以下中的一个：a)硬盘，b)计算机系统，c)数据服务器，d)数据中心，e)因特网服务提供商以及f)虚拟专用服务器(VPS)。

7.根据权利要求1-2和5中任一项所述的基于计算机的方法，其中，所述选择和所述分配是通过基于计算机的算法来实施的，所述基于计算机的算法的步骤通过以下中的一个或多个来提供：a)单级对称分配算法，b)二级对称分配算法，c)三级对称分配算法以及d)k级对称分配算法。

8.根据权利要求7所述的基于计算机的方法，其中所述k级对称分配算法是所述二级对称分配算法和所述三级对称分配算法的延伸，并且其中相关的基本存储块S_k是通过：$S_k = \frac{T}{km_1+(k-1)m_2+\cdots+2m_{k-1}+m_k}$给出的，其中T是预算，并且$m_i$是分配到所述k级的第i级的每个硬件存储节点的预算量。

9.一种用于分布式存储分配的基于计算机的系统，其包括：

基于计算机的源，其被配置为通过一个或多个通信链路与已知异构可靠性的多个基于硬件的存储节点通信，其中所述基于计算机的源被配置为执行存储分配算法以获得在所述多个基于硬件的存储节点上被分配的存储，所述存储分配算法完成以下任务：

i)基于被提供的预算和提供的目标函数，选择所述多个基于硬件的存储节点中的一个或多个不相交的子集，其中所述预算指定所述多个基于硬件的存储节点的总可用存储大小的一部分；以及

ii)通过将所述预算散布在所述一个或多个不相交的子集上，将所述预算分配到所述一个或多个不相交的子集，使得所述一个或多个不相交的子集中的每个子集的每个基于硬件的存储节点具有相同的分配预算量，所述相同的分配预算量不同于分配到不同子集的基于硬件的存储节点的量，

其中，对所述预算的分配是基于所述异构可靠性和所述预算的。

10.根据权利要求9所述的基于计算机的系统，其中，所述目标函数对应于以下中的一个或多个：a)数据检索的可靠性，b)数据的可访问性，c)存储的成本以及d)可用存储大小限制。

11.根据权利要求9或权利要求10所述的基于计算机的系统，其中，所述存储分配算法还包括通过以下中的一个或多个提供的步骤：a)单级对称分配算法，b)二级对称分配算法，c)三级对称分配算法以及d)k级对称分配算法。

12.根据权利要求11所述的基于计算机的系统，其中所述k级对称分配算法是所述二级对称分配算法和所述三级对称分配算法的延伸，并且其中相关的基本存储块S_k是通过：$S_k=\frac{T}{km_1+(k-1)m_2+\cdots+2m_{k-1}+m_k}$给出的，其中T是所述预算，并且$m_i$是分配到所述k级的第i级的每个基于硬件的存储节点的预算量。

13.一种分布式存储系统，其包括：

如权利要求9所述的基于计算机的系统，其被配置为通过一个或多个通信链路与已知异构可靠性的多个基于硬件的存储节点通信；以及

已知异构可靠性的多个基于硬件的存储节点，其中所述基于计算机的系统被配置为，基于由所述存储分配算法提供的被分配的存储，将大小等于所述预算的编码的文件分布在所述多个基于硬件的存储节点上。

14.根据权利要求13所述的分布式存储系统，其中，所述编码的文件是通过在所述基于计算机的源上执行的基于计算机的编码算法且基于以下中的一个进行编码的：a)最大距离可分(MDS)码和b)随机线性编码。

15.根据权利要求13或权利要求14所述的分布式存储系统，其中，所述多个基于硬件的存储节点中的基于硬件的存储节点包括以下中的一个：a)硬盘，b)计算机系统，c)数据服务器，d)数据中心，e)互联网服务提供商以及f)虚拟专用服务器(VPS)。

16.一种基于计算机的方法，其用于通过通信链路对多个独立消息实时流传输，所述基于计算机的方法包括以下步骤：

i)通过计算机，提供所述多个独立消息的消息大小s；

ii)通过计算机，提供基于若干时间步长的消息创建间隔c，其中所述消息创建间隔限定两个连续的消息的创建时间之间的时间间隔；

iii)通过计算机，提供指定预算的约束，其中所述预算对应于在每个时间步长传输的编码的数据包的最大大小；

iv)通过计算机，提供为若干时间步长的固定的解码延迟d，其中所述固定的解码延迟限定关于所述多个独立消息中的消息的创建时间的延迟，在该延迟内，所述消息必须通过基于计算机的解码器基于一个或多个传输的数据包进行解码；

v)通过计算机，提供异构可靠性模型，所述异构可靠性模型限定通过所述通信链路传输的数据包的异构时滞相关损失概率；

vi)通过计算机，对所述多个独立消息中的消息进行编码；以及

vii)基于所述步骤i)至vi)，通过计算机，产生对应于所编码的消息的多个数据包，并且通过所述通信链路，顺序地传输所述多个数据包，

其中：

所述多个独立消息中在时间步长i创建的消息被分配了在根据所述异构可靠性模型选择的时间步长i，i+1，…i+d传输的数据包的空间中的部分；

在跨越时间步长i，i+1，…i+d的数据包的所述空间中的所分配的部分，使用纠删码对所述消息编码；以及

通过基于计算机的解码器，在距所述消息的创建时间的所述固定的解码延迟内对所述消息进行解码。

17.根据权利要求16所述的基于计算机的方法，其中，数据包的所述空间中的所分配的部分是通过基于计算机的分配算法获得的，所述基于计算机的分配算法的步骤是通过以下中的一个或多个来提供的：a)单级对称分配算法，b)二级对称分配算法，c)三级对称分配算法以及d)k级对称分配算法。

用于异构系统的分布式存储分配

[0001] 相关申请的交叉参考

[0002] 本申请要求2013年3月14日提交的美国临时专利申请序列号No.61/784,282、题为“异构系统的分布式存储分配(Distributed Storage Allocation for Heterogeneous Systems)”的优先权,通过引用其全部公开内容并入本文。

[0003] 政府资助声明

[0004] 本发明是由政府支持在空军授权的FA9550-10-10166下进行的。政府享有本发明的一定权利。

[0005] 背景

技术领域

[0006] 本公开涉及与如在内容分发或无线通信中所用的网络通信和分布式网络架构的领域相关的方法和算法。特别地,本公开提出了新的分布式存储分配技术,例如,该分布式存储分配技术可以被用于通过包括具有异构访问概率的存储节点的分布式存储的网络进行通信。目的是通过分布式存储来存储给定的数据对象(例如,文件),其具有成功恢复的最大概率。数据对象可以跨越多个存储节点进行拆分和编码。假设使用适当的编码(例如,最大距离可分(MDS)码、随机线性编码),如果访问的数据的总量是至少原始数据对象的大小,则原始数据对象可以被恢复。

[0007] 概述

[0008] 根据本公开的各种实施例,在分布式存储系统中的各种存储节点可以按照其访问概率的递减次序进行布置(例如,具有最高故障概率的节点是最后的)。在本公开中提出了基于一个或多个级别的对称分配的新方法。在基本的一级对称分配中,整个存储预算被均匀散布在第一m节点上,其中,例如,基于故障概率确定m的值。在二级对称分配中,预算被分为两部分,一部分均匀散布在第一m_1节点上,并且另一部分均匀散布在随后的m_2节点上,使得分配到第一子集中的每个节点的量是第二子集中的每个节点被分配的量的两倍。还提出了衍生自二层分配的k级对称分配,其中$k=2$。根据本公开的各种实施例,提供了给定访问概率的、找到良好的一级、二级和k级对称分配的有效算法。

[0009] 尽管它们的低复杂度,如本公开的各种图所呈现的,在数值实验中,在相同的参数设置下,根据本公开一级、二级和k级对称分配可以胜过基于大偏差不等式和凸优化的现有的方法。此外,根据本公开新的二级和k级对称分配可以实现比新的一级对称分配更高的恢复概率。此外且根据本公开,对于小数量的节点(例如,$n\leq 4$),可以穷举确定的精确最优分配可以在一级和二级对称分配中被找到。

[0010] 除了分布式存储以外,上述方法还可以应用于其他问题,诸如例如,应用于实时流传输的编码设计问题,其中消息可以顺序地到达源(例如,基于计算机的工作站),并且消息在源进行编码,以用于通过数据包删除通道传输到汇节点(例如,基于计算机的工作站),汇节点需要在指定延迟内对消息顺序地进行解码。在各种网络场景中,数据包延迟(例如,如被同步节点接收)可以表现出差异,导致数据包接收的概率随着延迟而增加。通过将实时流

传输问题中的传输的数据包视作存储分配问题中的节点，使用对应的异构时滞相关损失概率，异构存储分配问题的解决方案转化为对应的流传输问题的会话内编码。

[0011] 根据本公开的第一方面，提出了用于在异构存储系统中分配存储的基于计算机的方法，该基于计算机的方法包括：提供已知异构可靠性的一组硬件存储节点；通过计算机提供目标函数；通过计算机提供约束；基于所述约束和所述目标函数，通过计算机，选择所述一组硬件存储节点中的一个或多个不相交的子集；以及基于所述选择，通过计算机将存储量分配到所述一个或多个不相交的子集，其中所述分配基于所述异构可靠性和所述约束，并且是通过将所述存储量散布到所述一个或多个不相交的子集上而获得的，使得所述一个或多个不相交的子集中的每个子集的每个硬件存储节点具有相同的分配存储量，所述相同的分配存储量不同于分配到不同子集的硬件存储节点的量。

[0012] 根据本公开的第二方面，提出了用于分布式存储分配的基于计算机的系统，该基于计算机的系统包括：基于计算机的源，其被配置为通过一个或多个通信链路与多个已知异构可靠性的基于硬件的存储节点通信，其中所述基于计算机的源被配置为执行存储分配算法，以获得在所述多个基于硬件的存储节点上的分配的存储，所述算法执行以下任务：i)基于所提供的预算和提供的目标函数，选择所述多个基于硬件的存储节点中的一个或多个不相交的子集，其中所述预算指定所述多个基于硬件的存储节点的总可用存储大小的一部分；ii)以及，通过将所述预算散布到所述一个或多个不相交的子集上，将所述预算分配到所述一个或多个不相交的子集，使得所述一个或多个不相交的子集中的每个子集的每个基于硬件的存储节点具有分配预算的相同量，所述分配预算的相同量不同于分配到不同子集的基于硬件的存储节点的量，其中所述预算的分配是基于所述异构可靠性和所述预算。

[0013] 根据本公开的第三方面，提出了用于通过通信链路对多个独立消息实时流传输的基于计算机的方法，所述基于计算机的方法包括步骤：i)通过计算机提供所述多个独立消息的消息大小s；ii)通过计算机提供基于若干时间步长的消息创建间隔c，其中所述消息创建间隔限定两个连续的消息的创建时间之间的时间间隔；iii)通过计算机提供指定预算的约束，其中所述预算对应于在每个时间步长传输的编码的数据包的最大大小；iv)通过计算机提供为若干时间步长的固定的解码延迟d，其中所述固定的解码延迟限定关于所述多个独立消息的消息的创建时间的延迟，在所述延迟内，所述消息必须基于一个或多个传输的数据包，通过基于计算机的解码器进行解码；v)通过计算机提供限定通过所述通信链路传输的数据包的异构时滞相关损失概率的异构可靠性模型；vi)通过计算机对所述多个独立消息中的消息进行编码；以及vii)基于所述步骤i)-vi)，通过计算机产生对应于所编码的消息的多个数据包，并且通过所述通信链路顺序地传输所述多个数据包，其中：所述多个独立消息中在时间步长i创建的消息被分配了根据所述异构可靠性模型所选择的时间步长i、i+1、…、i+d传输的数据包的空间中的部分；使用纠删码，消息被跨越时间步长i、i+1、…i+d的数据包的所述空间中的所分配的部分编码；以及通过基于计算机的解码器，在距所述消息的创建时间的所述固定解码延迟内，对所述消息进行解码。

附图说明

[0014] 图1示出的是在示例性分布式存储系统中的信息流。源s具有将被编码和存储在n个存储节点上的归一化单位大小的单个数据对象。随后，数据采集器t，通过只访问存储在

节点的随机子集r中的数据，试图恢复原始数据对象。

[0015] 图2示出的是包括3个异构节点(n＝3)的系统的最优分配表I，其中$p_1 \geq p_2 \geq p_3$。

[0016] 图3示出的是包括4个异构节点(n＝4)的系统的最优分配表II，其中$p_1 \geq p_2 \geq p_3 \geq p_4$。

[0017] 图4示出的是比较精确最优解的、关于n＝4且(p_1,p_2,p_3,p_4)＝(0.8479,0.6907,0.6904,0.6725)的、分别通过本公开的算法1和算法3获得的一级对称分配和二级对称分配的表现。

[0018] 图5示出的是表示根据本公开内容和现有技术所提出的算法在n＝30的恢复故障的概率方面表现的各种曲线图。

[0019] 图6示出的是表示当总预算T＝2且n＝30时的不同方法的每个节点的分布量的各种曲线图。

[0020] 图7示出的是表示所提出的算法在n＝50情况下恢复故障的概率方面表现的各种曲线图。

[0021] 图8示出的是表示当总预算T＝1:7且n＝50时的不同方法的每个节点的分布量的各种曲线图。

[0022] 图9示出的是表示不同的算法针对节点数的时间成本的各种曲线图。

[0023] 图10示出的是表示针对其中消息率s＝1的从1到4的数据包率P的解码故障概率的各种曲线图。

[0024] 图11示出的是表示针对范围从0.1到1的消息率s且数据包率P＝1的解码故障概率的各种曲线图。

[0025] 图12示出的是根据本公开的实施例的一级对称分配算法。

[0026] 图13示出的是根据本公开的实施例的图12中提出的算法的简化版本的一级对称分配算法。

[0027] 图14示出的是根据本公开的实施例的二级对称分配算法。

[0028] 图15示出的是根据本公开的实施例的三级对称分配算法。

[0029] 图16示出的是示例性实时流传输系统，其中消息顺序地到达源，并且被编码用于通过数据包删除通道进行传输，以由汇节点在距它们的创建时间的指定的延迟内进行解码。

[0030] 图17A和图17B示出的是被分配用于在顺序地传输数据包的流传输数据包构造的情况下以两个连续时间步长进行流传输数据包的存储。

[0031] 图18示出的是用于本公开的各种分布式存储系统的各种存储节点的示例性目标硬件。

具体实施方式

[0032] 引言

[0033] 根据本公开的各个方面，考虑了下列问题：利用异构访问概率在一组存储节点(例如，具有硬件依赖性)上存储数据对象(例如，文件、视频流、蜂窝语音消息，等等)，以便使给定的总存储预算的成功恢复概率最大化。数据对象可以跨越多个存储节点进行拆分和编码。例如，通过利用最大距离可分(MDS)码，如果访问数据的总量是至少数据对象的大小，则

原始数据对象可以被恢复。目标是确定跨越该组存储节点的存储预算的最优分配，使得随后的恢复概率被最大化。

[0034] 存储分配的问题是由实际存储问题激发的，例如，在具有异构节点的对等云存储、内容分发网络、延迟容忍网络和无线传感器网络方面。例如，每个节点可以是虚拟专用服务器(VPS)，虚拟专用服务器(VPS)的正常运行时间/停机时间统计是已知的，从而可以被用于作为对应的访问概率的量度。在另一个示例中，用于存储服务器的数据的硬盘架可以包括存储硬件(例如，硬盘)，该存储硬件(例如，硬盘)具有不同程度的可靠性，该不同程度的可靠性，例如按照给定的存储硬件的年龄和/或对应制造商的可靠性数据(例如，故障前平均时间MTBF)而进行测量的，该不同程度的可靠性还可用于作为具有本公开中所提供的各种解决方案可以应用的异构存储的等效分布式存储建模的基础。

[0035] 存储分配是复杂的组合最优化问题，甚至是在节点具有同种(例如，相同)访问概率的特殊情况下，因为这里有可行分配的大空间，并且计算给定分配的恢复概率是#P-hard[1]。该问题的变体已经在若干不同的领域进行研究，包括P2P网络[2]、无线通信[3][4]和可靠性工程[5]。对于节点具有不同访问概率的异构的情况，Ntranos等人[6]提出了基于大偏差不等式和和凸优化的算法。

[0036] 如由本公开的各种实施例提供的，(基本的)对称分配将预算散布在适当选择的节点子集上，并且可以有效地进行构造，诸如，例如计算低廉，并且还可以胜过[6]中现有的方法。根据本公开的进一步的实施例，该基本的对称分配和对应的构造算法可以扩展到多k级对称分配，在多k级对称分配中，总预算被分为k部分，每个部分均匀散布在不同的节点子集上，使得分配到更高级中的每个节点的量是最后一级中每个节点的量的几倍。换句话说，在最后一级中每个节点的量作为单位量。

[0037] 在本公开接下来的部分中，提供了关于依照本公开的新的二级和三级对称分配的给定数量节点的性能分析。对于小数量$n \leq 4$的节点，可以穷举确定的(例如，穷举计算的)精确最优分配，可以在一级和二级对称分配(例如，如在表I和表II中给出的)中被找到。对于大量节点，如由本公开的各种实施例提供的这些对称分配可以胜过现有的非对称分配。此外，通过增加相关级别的数量可以进一步提高它们的性能，并且因此可以提高对称分配的灵活性。如在本公开的接下来的部分中所提出的，用于对称分配的各种技术可以应用于流传输编码的设计。

[0038] 问题描述

[0039] 假设原始数据对象被归一化为单位大小。数据对象可以利用适当编码进行存储在如图1所述的n个异构节点(1,2,…n)上，其中源节点S(110)通过n个链路(105)将数据对象供应给n个节点。针对给定的总存储预算T，每个节点i存储被编码的数据量x_i。例如，关于存储预算的约束可以与有限的传输带宽和/或有限的存储空间有关，或者甚至与关于存储成本的约束相关，因为在每个节点中反映其整体上的数据对象可能成本过高。在数据检索时，数据采集器(115)以概率p_i单独地访问每个节点，其中$0 < p_i < 1$。让r指示由数据采集器访问的节点的随机子集，其中，例如，r的分布概率可以由访问模型和/或故障模型(例如，在概率意义上，节点或链路可以出现故障)指定。可以如由以下表达式(1)所提供的、通过最大化恢复的目标函数和对可用存储施加限制的约束来说明该最优化问题的通用版本：

[0040]
$$\max_{x_1,\ldots,x_n} \sum_{r\in\mathcal{P}(\{1,\ldots,n\})} \left(\prod_{i\in r} p_i\right)\left(\prod_{i\notin r} 1-p_i\right)\cdot 1\left[\sum_{i\in r} x_i \geq 1\right]$$

服从于：

$$\sum_{i=1}^{n} x_i \leq T$$

$$x_i \geq 0 \qquad \forall i \in \{1,\ldots,n\} \qquad (1)$$

[0041] 其中1[.]代表指示函数，也被称为特征函数。每个节点可用概率p_i，并且得到的一组可用节点r(例如，可用被访问的节点)是随机的，并且是随机的大小。在表达式(1)中的目标函数可以被认为是恢复(或访问)概率，该恢复(或访问)概率被表示为对应于子集r的概率的和，其允许给定了由预算函数施加制约因素(例如，约束)的数据对象的成功恢复，在这种情况下，预算函数是限定总预算T的固定函数。换句话说，(1)中表示的最优化问题，试图使给定了关于可用存储预算施加了一定制约因素的系统的可靠性最大化。根据本公开且如在下面的部分中所提出的各种技术，对于该最优化问题提供了解决方案；以包括这些节点的系统的整体可靠性被最大化的方式，优化了在具有不同可靠性不同节点中输入(分配)多少存储。该解决方案还被提供于表示步进的获得最优化解决方案的算法的流程图中。本领域技术人员很熟悉最优化问题的这种公式化表述和对应的方法。虽然在最优化问题(1)中所考虑的约束是固定的，但是本领域技术人员将知道如何修改这样的约束以更好地符合一套施加的要求，并且仍然使用根据各种提出的实施例的教导以获得最优化的分配。例如，人们可能对使与预算(存储)相关的成本最小化感兴趣，所以使用越多的存储，系统的成本越多。在这种情况下，约束可以变成成本(目标)函数，并且可靠性目标可以变成可靠性约束，对于该可靠性约束，最优化问题将得到解决。换句话说，根据预期目标，上述目标函数和约束可以互换。可选地，目标函数可以是成本和可靠性的加权函数。

[0042] 为了简洁，假设$T<n$，这是因为$T\geq n$，最优分配是平凡解：$\{1,\cdots,1\}$，其中每个节点存储原始数据对象的一个副本。此外，可以假设$x_i\leq 1$，因为一个节点存储超过一个以上的数据副本是无益的。

[0043] 虽然由(1)限定的最优化问题对于大的n情况是难以计算的，但是通过对所有可行的子集的(例如，节点的)组合进行穷举搜索为小的n情况确定最优分配是可能的。在图2和图3的表I和表II中分别提供了通过对不同场景的穷举搜索获得的最优分配的列表。表I示出的是$n=3$且$p_1\geq p_2\geq p_3$的最优分配，而表II示出的是$n=4$且$p_1\geq p_2\geq p_3\geq p_4$的最优分配。

[0044] 一级对称分配

[0045] 根据本公开的实施例，通过将总预算T均匀散布在节点的子集上获得基本的一级对称分配。使用更高概率的节点增加系统的可靠性。因此，节点可以用递减次序的访问概率进行排序，使得$p_1\geq p_2\geq\cdots\geq p_n$，并且由(1)提供的最优化问题降低为选择子集的大小m，对应于分配：

[0046]
$$\left\{x_1=\frac{T}{m},\ldots,x_m=\frac{T}{m},\ x_{m+1}=0,\ldots,x_n=0\right\}$$

[0047] 成功的恢复要求访问m个非空节点中的至少$\left\lceil \frac{m}{T} \right\rceil$。访问M个非空节点的概率是：

[0048] $$P_M = \sum_{r \in \mathcal{P}(\{\binom{m}{M}\})} \left(\prod_{k \in r} p_k\right)\left(\prod_{l \notin r} (1-p_l)\right) \quad (2)$$

[0049] 其中$\mathcal{P}\left(\left\{\binom{m}{M}\right\}\right)$表示{1,…,m}的大小-M子集的集合。

[0050] 因此，由表达式(3)给出成功恢复的概率：

[0051] $$P_{success} = \sum_{M=\lceil \frac{n}{k} \rceil}^{m} P_M \quad (3)$$
$$= \sum_{M=\lceil \frac{n}{k} \rceil}^{m} \sum_{r \in \mathcal{P}(\{\binom{m}{M}\})} \left(\prod_{k \in r} p_k\right)\left(\prod_{l \notin r} (1-p_l)\right)$$

[0052] 根据本公开的实施例，如图4中所提出的算法1通过使用表达式(2)和(3)找到最优一级对称分配。如本文所使用的术语“级”可以指一种分配，在该分配中用非零预算分配的每个节点包含总预算的相同的量。

[0053] 代替考虑m中所有可能的值，可以考虑类似于[1，部分II-C]中提供的方法，并且因此将算法的范围限制于对应于$\left\lceil \frac{m}{T} \right\rceil$的每个不同值的最大的m，其产生候选值$m \in \{\lfloor T \rfloor, \lfloor 2T \rfloor, \ldots, \lfloor K_{max} T \rfloor\}$，其中Kmax是最大的整数k，使得$\lfloor KT \rfloor \leq n$。然而，算法1对于大值n仍然是难以计算的，因为相关的计算复杂度随着n的值迅速增长。

[0054] 为了进一步降低算法1的复杂度且根据本公开的实施例，每个概率p_i可以由通过$p_i' = p_{avg} = \frac{\sum_{i=1}^{m} p_i}{m}$给定的概率$p_i$的平均值代替，并且作为获得算法2的结果，如图5中所提到的，因此，算法2可以被认为是算法1的简化版本。该简化(例如，使用p_{avg})与原始算法1相比，可以大大减少该算法的运行时间(例如，在硬件处理器上执行的对应的程序代码)。如在本公开的接下来的部分提出的数值实验示出如在图5的算法2中描述的该简化的版本，尽管其简单性，但提供了非常好的性能。技术人员很容易理解如由图5中对应的步骤(131)-(138)所描述的算法2(以及在本公开中描述的其他新的算法)只是求解如在本公开的本节所提供的各种方程和相关简化的示例性实施。

[0055] 二级和三级对称分配

[0056] 基于对表I和表II中最优分配的观测，值得注意的是，对于给定的预算T，一些最优解具有两级分配量，发明人已经考虑了更一般类的二级对称分配，其中每级包括总可用节点的不相交的子集，其中分配到第一级中每个节点的量是分配到第二级中每个节点的量的两倍：

[0057] $$\left(\underbrace{x_1, \ldots, x_{m_1}}_{\text{级1}}, \underbrace{x_{m_1+1}, \ldots, x_{m_1+m_2}}_{\text{级2}}, x_{m_1+m_2+1}, \ldots, x_n\right), \quad (4)$$

[0058] 其中

[0059] $$x_1 = \cdots = x_{m_1} = S_1,$$

[0060] $$x_{m_1+1}=\cdots=x_{m_1+m_2}=S_2,$$

[0061] $$x_{m_1+m_2+1}=\cdots=x_n=0,$$

[0062] $$S_1=\frac{2T}{2m_1+m_2},$$

[0063] $$S_2=\frac{T}{2m_1+m_2}. \qquad (5)$$

[0064] 因为存储在每个节点中的数据的量可以表示为S_2的多倍，其中$S_2=\frac{T}{2m_1+m_2}$，S_2可用被视为量子或基本存储块。遵照先前部分的方法，发明人注意到当访问了$2m_1+m_2$个基本存储块中的至少$\left\lceil\frac{2m_1+m_2}{T}\right\rceil$时，发生成功恢复。让$m_{acc}\in\{0,\ldots,2m_1+m_2\}$是访问的基本存储块的总数，并且$m_{acc,L}\in\{0,\ldots,m_L\}$是在级L中访问的节点的数量。因此，$m_{acc}=2m_{acc,1}+m_{acc,2}$。其遵照所有对$(m_{acc,1},m_{acc,2})$的集合$\mathcal{M}_{m_1,m_2}$，使成功恢复可以由以下表达式(6)表示：

[0065] $$\left.\begin{aligned}\mathcal{M}_{m_1,m_2}=\{(m_{acc,1},m_{acc,2}):\quad & 2m_{acc,1}+m_{acc,2}\geq\left\lceil\frac{2m_1+m_2}{T}\right\rceil,\\ & m_{acc,1}\in\{0,\ldots,m_1\},\\ & m_{acc,2}\in\{0,\ldots,m_2\}\}.\end{aligned}\right\} \qquad (6)$$

[0066] 在该阶段，发明人作出简化假设：分配到二级对称分配的第一级(例如，级1)的节点的量$mOpt_1$，小于先前提出的一级对称分配中的$mOpt_{oneLevel}$。

[0067] 基于该简化假设，并且因此在一级对称分配算法中获得的值$mOpt_{oneLevel}$，并且根据本公开的进一步的实施例，提供了如图6中所描述的算法3。算法3通过以类似于算法2的方式大概地计算在二级(例如，级1、级2)中每个的概率，有效地找到由$(mOpt_1,mOpt_2)$给定的良好的二级对称分配。

[0068] 如在一级对称分配的情况下，算法3通过只考虑(m_1,m_2)的某些选择，减少搜索空间。特别地，对于给定的选择m_1，该算法将关注限制于对应于$\left\lceil\frac{2m_1+m_2}{T}\right\rceil$的每个不同值的最大的$m_2$，其产生候选值：

[0069] $$m_2\in\{\lfloor K_{min}T-2m_1\rfloor,\lfloor(K_{min}+1)T-2m_1\rfloor,\ldots,\lfloor K_{max}T-2m_1\rfloor\}$$

[0070] 其中K_{min}是最小的整数k，使得$\lfloor KT-2m_1\rfloor\geq 0$，并且$K_{max}$是最大的整数k，使得$\lfloor KT-2m_1\rfloor\leq n-m_1$。

[0071] 根据发明人的观测，$mOpt_2$的值在许多循环中很少改变，并且$mOpt_2$的两个不同的值变化通常在循环的三分之二大小的距离D内。所以在算法3中，当在内部循环中由算法的步骤(1402)-(1413)限定的$mOpt_2$对于三分之二的循环大小不改变时，该算法认为该$mOpt_2$为当前循环中最好的选择，并继续下一个循环，从而大大减少了该算法的运行时间(例如，如在硬件处理器中所执行的运行时间)，并且仍然达到几乎相同的性能。还应该注意的是，D可以被设置为较小的值，这使得该算法运行更快，但可能产生次优解。

[0072] 根据本公开的另一个实施例，在图7的算法4中以示例性形式描述和提出了支持三级的对称分配的进一步的推广。让三级的子集大小分别是m_1、m_2、m_3，其中在第一级中每个节

点的量是在第三级中每个节点的量的3倍，并且在第二级中每个节点的量是第三级中每个节点的量的2倍。所有三元组($m_{acc,1}$,$m_{acc,2}$,$m_{acc,3}$)的集合$\mathcal{M}_{m_1,m_2,m_3}$使成功恢复可以表示为：

[0073]
$$\left.\begin{aligned}\mathcal{M}_{m_1,m_2,m_3} = \{(m_{acc,1},m_{acc,2},m_{acc,3}):\\ 3m_{acc,1}+2m_{acc,2}+m_{acc,3} \geq \left\lfloor\frac{3m_1+2m_2+m_3}{T}\right\rfloor,\\ m_{acc,1} \in \{0,\ldots,m_1\},\\ m_{acc,2} \in \{0,\ldots,m_2\},\\ m_{acc,3} \in \{0,\ldots,m_3\}\}.\end{aligned}\right\} \quad (7)$$

[0074] 发明人作出简化假设：在三级对称分配中的m_1小于在本公开中先前描述的二级对称分配的$mOpt_{1,twoLevel}+mOpt_{2;twoLevel}$。基于该简化假设和本公开的每个实施例，提供了如图7中所描述的算法4。算法4为由表达式(1)表示的最优化问题有效地找到良好的三级对称分配。

[0075] 如在本公开的先前部分中提出的前述对称分配中，通过只考虑(m_1,m_2,m_3)的某些选择，可以减少搜索空间。特别地，对于给定的选择m_1和m_2，发明人将关注限制于对应于$\left\lfloor\frac{3m_1+2m_2+m_3}{T}\right\rfloor$的每个不同值的最大的$m_3$，其产生候选值：

[0076]
$$m_3 \in \{\lfloor K_{min}T-3m_1-2m_2\rfloor,\lfloor(K_{min}+1)T-3m_1-2m_2\rfloor,\ldots,\lfloor K_{max}T-3m_1-2m_2\rfloor\}$$

[0077] 其中K_{min}是最小的整数k，使得$\lfloor KT-3m_1-2m_2\rfloor \geq 0$，并且$K_{max}$是最大的整数k，使得$\lfloor KT-3m_1-2m_2\rfloor \leq n-m_1-m_2$。我们还将距离阈值D设置为由该算法的步骤(1503)-(1515)限定的(内部)循环的三分之二大小。

[0078] 按照本公开的各种实施例所提出的对称分配，可以被很容易地推广以支持k级，其中k＝4、5、6等等。在这样的k级对称分配中，存储预算T可以被分配，使得每个非空节点存储c，2c，…，或kc数据量中的任一个，其中c是由$S_k=\frac{T}{km_1+(k-1)m_2+\cdots+2m_{k-1}+m_k}$给出的对应的量子或基本存储块$S_k$的大小。

[0079] 数值实验

[0080] 在该部分中，我们依据故障恢复的概率评估以上算法的性能。

[0081] A.当n＝4时与表II的最优解进行比较

[0082] 对于n＝4，通过算法1和算法3获得的一级和二级对称分配的性能可以分别与从表II(图3)获得的精确最优分配进行比较，精确最优分配是穷举计算的。发明人考虑在0.5和1之间独立和均匀随机地得出的存储预算范围$1\leqslant T\leqslant 4$，以及节点访问概率p_i，其中$1\leqslant i\leqslant n$。图8示出的是访问概率典型实例的结果。按照本公开的实施例，发明人发现一级和二级分配对称一般良好地近似于精确最优分配。发明人还指出，对于T的某些有限的范围，为了最优性，需要二级对称分配(例如，在这种情况下，$m_1=1$，$m_2=3$)。

[0083] B.对于大的n值与其他现有算法进行比较

[0084] 在该部分中，提供了对于算法2、3和4的大的n值的性能与在[6]中提出的现有算法的比较，其中[6]即，最大扩散、切尔诺夫闭合形式和切尔诺夫迭代(Maximal spreading，Chernoff closed-form和Chernoff iterative)。用在0.5和1之间独立和均匀随机地得出的预算范围$1.2 \leqslant T \leqslant 2$，以及节点访问概率$p_i$来进行比较。使用数值蒙特卡洛(Monte Carlo)模拟实验获得(且绘制)故障概率曲线。

[0085] 在图9、图10、图11和图12中给出了关于不同问题规模$n=30$、50的结果。在图9和图11中，根据本公开的各种实施例，绘制由一级对称分配、二级对称分配和三级对称分配产生的故障概率曲线，以及由在[6]中提出的现有技术切尔诺夫迭代和切尔诺夫闭合形式方法产生的曲线。根据这些曲线，当预算T很小时，对称分配总是胜过切尔诺夫方法。当T很大(例如，对于$n=30$，$T=2$)时，即使切尔诺夫方法可以胜过一级对称分配和二级对称分配，但仍然存在的是三级对称分配可以比切尔诺夫方法表现更好。在图10和图12中分别提供了表示关于预算$T=2$和预算$T=1.7$的不同方法的量分布的曲线图。这些曲线图表明，如果对称方法变得更灵活且支持更多级，性能就可以进一步提高。

[0086] C.不同方法的时间成本

[0087] 图13示出针对不同算法的节点数的时间成本。根据该图，一级对称分配总是比其他分配更快，并且二级对称分配比$n<55$的切尔诺夫迭代算法更快。在大多数场景中，三级对称分配是最慢的，因此其可以被用作更好性能的折衷。

[0088] D.算法应用:对称时不变会话内编码

[0089] 除了分布式存储之外，对称分配算法还可以应用于实时流传输[7]、[8]的编码设计的问题，在图16中提供了示例性示意配置，其中消息顺序地到达源(1610)，并且被编码为通过数据包删除通道(1620)传输到汇节点(1630)(例如，通过如本领域已知的纠删码)，汇节点需要在指定的延迟内对消息顺序地进行解码。在各种网络场景中，数据包延迟可以表现出差异，导致数据包接收的概率随着延迟而增加。通过查看在流传输问题中消息创建时间和解码截止时间之间传输的数据包，如节点在存储分配问题中，利用对应的异构时滞相关损失概率(例如，异构可靠性模型)，异构存储分配问题的解决方案转化为对应的流传输问题的会话内编码，其中在存储分配问题中的存储预算转化为总可用数据包大小。

[0090] 作为例证且进一步参考图16，让我们假设，在源(1610)以$c=1$时间步长的规则间隔创建均匀大小$s>0$的独立消息$(1,2,3,\cdots)$，并且均匀大小$s>0$的独立消息$(1,2,3,\cdots)$必须在接收器处在距它们相应的创建时间$d=10$时间步长的延迟内进行解码。我们使用来自[9]中图5(a,b)的延迟界违规概率作为每个数据包的不同的故障概率。源被允许每个时间步长通过链路传输大小$P>s$的单个数据包。因此，对应的虚拟预算可以由$\lfloor \frac{P}{s} \rfloor$表示。

[0091] 给定$s=1$的固定消息率和在1到4范围中的数据包率P，可以获得图14中的故障概率曲线图。另外，给定$P=1$的固定数据包率和在0.1到1范围中的消息率s，可以获得图15中的故障概率曲线图。

[0092] 按照本公开的各种实施例，如与示例性流传输结构有关的，图17A和图17B示出的是在$c=1$和$d=4$的情况下的示例性数据包分布。在图17A和图17B的图中的列表示顺序传输的数据包，其中在给定时间(例如，t轴)处所有传输的数据包的总和，受预算(例如，总数据

包大小)的限制。图17A和图17B中阴影框表示在时间步长1创建的数据(x_1,x_2,x_3,x_4)的量,并且散布在一些传输的数据包(p_0,p_1,p_2,p_3)上。这些数据包根据延迟量进行传输,这些数据包在该延迟量内进行解码。图17B中的纹线区块表示在每个传输的数据包中来自时间步长2的(例如,在时间步长2创建的)数据的量。在时间步长i传输的数据包具有由时间步长j接收的概率p_{j-i},其中对于$\Delta = j-i$的延迟值,$p\Delta$是延迟Δ的递增函数。给定在每个传输可用的最大总数据包大小(例如,预算)和解码概率$p\Delta$,通过使用最优化问题(1)可以导出每个传输的数据包(p_0,p_1,p_2,p_3)的最优分配的数据包大小(x_1,x_2,x_3,x_4),并且因此相关的解决方案可以是根据关于异构存储分配问题所提供的解决方案。根据本公开和两个问题之间提供的相似之处给定这样的教导,本领域技术人员将能够将这样的教导应用于包括异构部件的其他系统。

[0093] 根据本公开和在先前部分中讨论的各种实施例,提供了用于为由表达式(1)给出的异构分布存储分配最优化问题找到多级对称分配的各种算法。随着级数及因此对称分配的灵活性增加,算法的运行时间(例如,使用计算机处理器)也增加。然而,即使用很少的级(例如,一级、二级)和相对低的复杂度以及运行时间,所提出的数值实验示出,在很多场景中,根据本公开的各种实施例提出的对称分配仍然可以胜过现有的更复杂的分配方案。

[0094] 硬件考虑

[0095] 再次参考本公开的图1中所提出的和先前部分中所描述的分布式异构存储系统,本领域技术人员容易理解的是,由图1中对应的框(例如,标记的“存储节点i”)表示的存储节点可以采取硬件/固件/软件设备的各种不同组合的形式,以便创建图1中所表示的系统的异构性质,而不影响各种提供的算法的效率。每个存储节点可以被有效地认为是能够存储数据的实体,诸如,例如,单个硬盘驱动器、单个基于计算机的工作站、包括网络服务器的数据中心、存储阵列或单个互联网服务提供商。

[0096] 根据一个示例性实施,每个存储节点可以是不同的云服务提供商,其中相关的虚拟专用服务器(VPS)和对应的存储系统可以被用于与源(110)和数据对象x_i的部分的存储进行通信。每个这样的云服务提供商(例如,存储节点)还可以具有已知统计的正常运行时间/停机时间(例如,节点的80%或70%,或…的时间是可用的),该正常运行时间/停机时间可以进一步被用于通过数据处理器(115)为数据对象的恢复导出相关的访问/可靠性模型(例如,云服务以概率p_i可用)。

[0097] 根据另一个示例性实施,每个存储节点可以是在数据存储架中使用的物理硬盘,物理硬盘被用于作为源(110)的主要存储。例如,该源可以是服务器计算机。另外,例如,基于由硬盘制造商提供的故障前平均时间(MTBF)参数,每个硬盘可以具有相关的故障曲线。每个硬盘的这样的MTBF和硬盘的年龄可以被用于由数据采集器工作站导出用于数据对象恢复的对应的访问/可靠性模型。

[0098] 在某些情况下,存储节点的可靠性/可访问性可以是节点的各个组成部分的质量的函数。例如,便宜的硬盘驱动器可以比最先进的硬盘具有较低的可靠性,其中便宜的硬盘驱动器可能是翻新的驱动器或者是用低档次的部件制造出的硬盘驱动器。作为另一个示例,使用较旧的基础设施(例如,硬件、计算机、存储系统、布线和通信链路等等)的服务提供商可以成本低,但比使用最先进的基础设施条件的提供商具有更差的可靠性/可访问性。在这种情况下,由表达式(1)描述的最优化问题可以考虑与分布式异构存储空间相关的成本,

并且使这样的成本最小化，同时提供数据对象的恢复能力。因此，根据本公开，在各种算法中提供的所描述的方法，可以被用于降低整个系统的成本且保持可靠性。

[0099] 在图1的系统中所示的、连接各种存储节点和站(110、115)的各种通信链路，可以是使用导线的有线连接，诸如绞线、同轴电缆、以太网线缆、使用导线以携带电信号/光信号的任何类型的线缆，等等，和/或者是使用在空气中传输的RF信号的无线链路。一些这样的链路可以使用有线和无线链路的组合制成。

[0100] 在本公开的各种实施例中，源(例如，图1的110、图16的1610)可以是运行包括由本公开提供的各种算法(例如，图4-7)中的一个或多个的程序代码(例如，软件)的计算机处理器。这样的程序代码可以包括根据本公开的各种算法的各种步骤。这样的计算机处理器可以使用具有大量计算功率和存储器大小的任何目标硬件(如，之后所述的图18)来实施，其或者是从支架上下来的，诸如主机、微型计算机、台式机(PC、MAC，等等)、手提式电脑、笔记本电脑等等，或者是从专为特定任务而设计的且可以包括微处理器、数字信号处理器(DSP)、各种FPGA/CPLD等等的专有硬件。使用不同的操作系统(OS)和/或硬件配置，本领域技术人员容易发现的是，这样的计算机处理器的各种不同的配置。因此，包括各种算法中的一个或多个的程序代码可以适用于源的任何特定的目标硬件。

[0101] 可以在硬件、软件、固件或其组合中，实施本公开中描述的方法(例如，如在图4-7中所提供的单个级的对称分配算法和多个级的对称分配算法和相关的步进流程图)和对应的分布式存储通信系统。可以使用硬件、软件和/或固件的组合一起或单独地实施被描述为模块、节点或部件的特征。本公开的方法(例如，流程图、算法)的软件部分可以包括计算机可读介质，所述计算机可读介质包括指令(例如，可执行程序)，当执行该指令时至少部分实施所描述的方法，诸如根据本公开的各种实施例的对称分配算法的部分或全部结构。例如，所述计算机可读介质可以包括随机存取存储器(RAM)和/或只读存储器(ROM)。该指令可以由处理器(例如，数字信号处理器(DSP)、专用集成电路(ASIC)、现场可编程逻辑阵列(FPGA)或其组合，其可以集成在单个集成电路(IC)内)执行。

[0102] 图18是根据本公开的各种教导的、用于实施一个或多个代码/子图结构方法(例如，源节点)和/或这样的编码数据(例如，中继/汇节点)的解码的目标硬件(10)(例如，计算机系统)的示例性实施例。该目标硬件包括处理器(15)、存储器组(20)、本地接口总线(35)和一个或多个输入/输出设备(40)。基于存储在存储器(20)中的一些可执行程序，该处理器可以执行与各种提供的编码构造方法(例如，全部或部分)相关的且如由操作系统(25)提供的一个或多个指令。这些指令通过本地接口(35)被带到处理器(20)且通过一些数据接口协议，这些指令被专门用于本地接口和处理器(15)。应该注意的是，本地接口是若干元件的符号表示，所述若干元件诸如控制器、缓存器(高速缓存)、驱动器、中继器和接收器，这些一般针对，提供在基于处理器的系统的多个元件之间的地址、控制和/或数据连接。在一些实施例中，处理器(15)可以配备有一些本地存储器(高速缓存)，该本地存储器(高速缓存)可以存储用于一些附加的执行速度而要被实施的一些指令。由处理器对指令的执行，可能需要使用一些输入/输出设备(40)，诸如输入包括要被编码的和/或被解码的消息的比特流数据、来自键盘的输入命令、将数据输出到显示器或输出将要通过通信信道进行传输的编码的数据包或从通信通道输入数据包。在一些实施例中，操作系统(25)通过作为中心元件以集中程序执行所需的各种数据和指令，并且将这些提供给微处理器，来推进这些任务。在一

些实施例中，虽然目标硬件设备(10)的基本架构将与图18中所描述的是一样的，但操作系统可能不存在，并且所有的任务都在处理器(15)的直接控制下。在一些实施例中，为了增加执行速度，多个处理器可以被用于并行配置。在这种情况下，可执行的程序可以被特别定制成并行执行。另外，在一些实施例中，处理器(15)可以执行如在本公开中所提供的方法的一部分，并且可以使用放置在由目标硬件(10)通过本地接口(35)可访问的输入/输出位置的专用硬件/固件来实施另一部分。目标硬件(10)可以包括多个可执行程序(30)(例如，包括专用通信程序)，其中每个可以单独运行或者彼此可以组合运行。这些可执行的程序可以包括指令，当由处理器执行这些程序时，实施本公开中提出的方法(例如，构造算法、编码、解码)的至少一部分。

[0103] 可以在集成芯片(IC)中实施如图18所描述的这样的示例性计算机硬件。根据本公开的一些实施例，实施本公开的各种实施例(例如，算法)的对称分配模块，无论是全部还是部分，可以部分地或全部地在IC内实施。这样的IC可以被用于作为系统的一部分以提供根据本公开的各种实施例的异构系统的对称分配。例如，基于IC性能和/或其他系统要求，根据分配和/或算法执行速度的具体所需的性能，存储在IC的存储器(例如，可编程存储器)中的程序可以被升级，以便符合本公开的算法。技术人员可以想到的是，或者是一部分或者是整体上、并且使用所提供的对称分配方法和在图4-7中描述的流程图的算法的、图1和图16中提出的系统的各种其他可能的硬件/软件/固件实施。

[0104] 提供了以上所述示例，以给本领域技术人员完整的公开和如何制作和使用用于异构系统的分布式存储分配和相关的单个级别的对称分配算法和多个级别的对称分配算法的实施例的说明，并不旨在限制发明人所认定的其公开内容的保护范围。用于执行本公开的以上所述的模式的修改，可以被技术人员用于信息/编码/通信的理论和处理中，并且该修改被认为是落入以下权利要求书的保护范围内。本说明书中所提及的所有专利和出版物可以指示本公开所属领域技术人员的技术水平。本公开中所提及的所有引用，通过对其同程度的引用而被并入，这好比每个引用已经通过引用将其整体地分别并入。

[0105] 要理解的是，本公开并不局限于特定的方法或系统，当然，其方法或系统可以变更。还要理解的是，本文所使用的术语，仅用于描述特定的实施例的目的，并不旨在限制。除非该内容另外清楚地规定，否则如在本说明书和随附权利要求书中所使用的，单数形式“一种/个(a/an)”、以及“所述(the)”包括复数对象。除非该内容另外清楚地规定，否则术语“多个”包括两个或多个对象。除非另有定义，否则本文使用的所有技术术语和科学术语具有与本公开所属领域的普通技术人员中的一个通常理解的相同的含义。

[0106] 在本公开中所提及且以[.]所指示的以及在随后的参考列表中提供了其标题、作者和出版物的9个参考资料，其全部内容通过引用方式并入本文。

[0107] 已经描述了本公开的一些实施例。然而，应当理解的是，可以做出各种修改，将不偏离本公开的精神和保护范围。因此，其他实施例在以下权利要求的保护范围内。

[0108] 参考列表

[0109] D.Leong、A.G.Dimakis和T.Ho，“Distributed storage allocations”，IEEE Trans.Inf.Theory，2012年7月，第58卷第7期第4733页-第4752页。

[0110] W.K.Lin、D.M.Chiu和Y.B.Lee，“Erasure code replication revisited”，Proc.Int.Conf.Peer-to-Peer Comput.(P2P)，2004年9月。

[0111] A.Tsirigos和Z.J.Haas,“Analysis of multipath routing—Part I:The effect on the packet delivery ratio”,IEEE Trans.Wireless Commun.,2004年1月第3卷第1期第138页-第146页。

[0112] “Analysis of multipath routing,Part 2:Mitigation of the effects of frequently changing network topologies”,IEEE Trans.Wireless Commun,2004年3月第3卷第2期第500页-第511页。

[0113] J.-S.Wu和R.-J.Chen,“An algorithm for computing the reliability of weighted-k-out-of-n systems”,IEEE Trans.Rel1994年6月第43卷第2期第327页-第328页。

[0114] V.Ntranos、G.Caire、和A.G.Dimakis,“Allocations for heterogeneous distributed storage”,Proc.IEEE Int.Symp.Inf.Theory(ISIT),2012年7月。

[0115] D.Leong和T.Ho,“Erasure coding for real-time streaming”,Proc.IEEE Int.Symp.Inf.Theory(ISIT),美国,马萨诸塞州,剑桥,2012年7月。

[0116] D.Leong、A.Qureshi和T.Ho,“On coding for real-time streaming under packet erasures”,Proc.IEEE Int.Symp.Inf.Theory(ISIT),土耳其,伊斯坦布尔,2013年7月。

[0117] Y.Chen、J.Chen和Y.Yang,“Multi-hop delay performance in wireless mesh networks”,MOBILE NETW APPL,2008年4月,第13卷第160页-第168页。

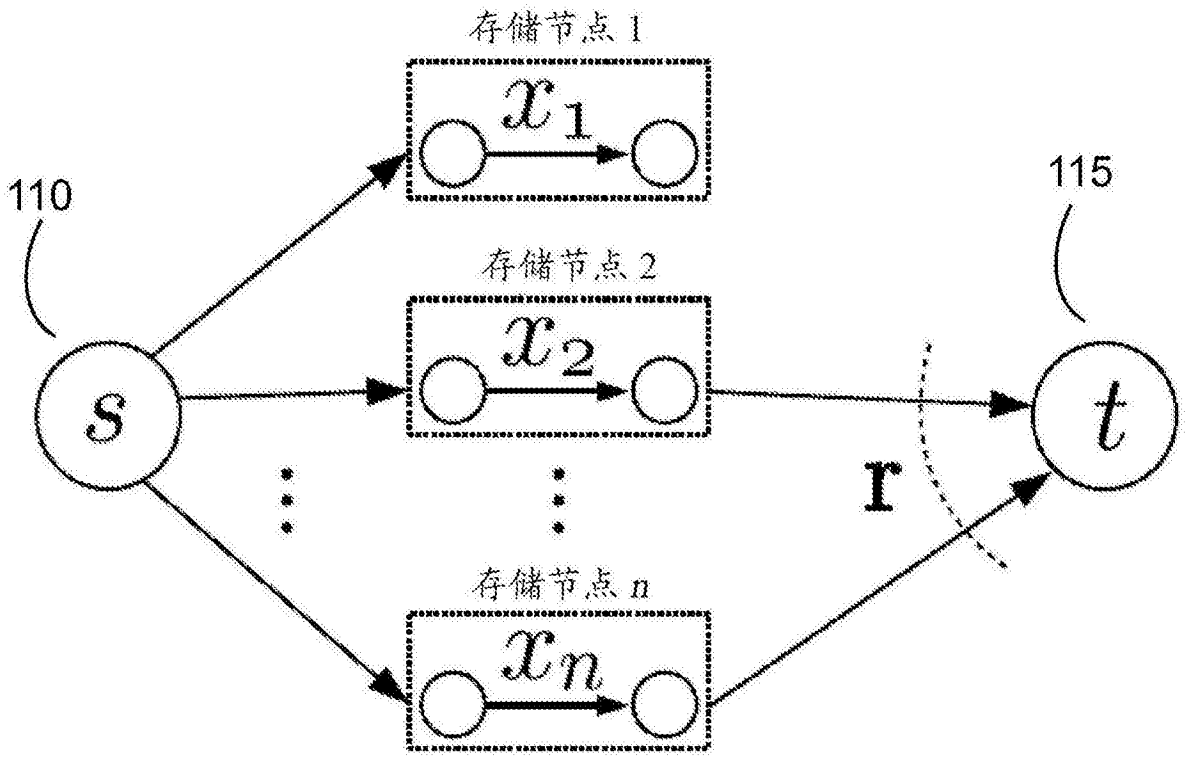


图1

表I

预算 T	最优分配 (x_1, x_2, x_3)
$1 \le T < \frac{3}{2}$	(1,0,0)
$\frac{3}{2} \le T < 2$	(1,0,0) if $-p_1 + p_1p_2 + p_1p_3 + p_2p_3 - 2p_1p_2p_3 \le 0$
	$(\frac{1}{2}, \frac{1}{2}, \frac{1}{2})$ if $-p_1 + p_1p_2 + p_1p_3 + p_2p_3 - 2p_1p_2p_3 \ge 0$
$2 \le T < 3$	(1,1,0)

图2

表II

预算 T	最优分配 (x_1, x_2, x_3, x_4)
$1 \le T < \frac{4}{3}$	$(1, 0, 0, 0)$
$\frac{4}{3} \le T < \frac{3}{2}$	$(1, 0, 0, 0)$ if $-p_1 + p_1p_2p_3 + p_1p_2p_4 + p_1p_3p_4 + p_2p_3p_4 - 3p_1p_2p_3p_4 \le 0$
	$\left(\frac{1}{3}, \frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right)$ if $-p_1 + p_1p_2p_3 + p_1p_2p_4 + p_1p_3p_4 + p_2p_3p_4 - 3p_1p_2p_3p_4 \ge 0$
$\frac{3}{2} \le T < \frac{5}{3}$	$(1, 0, 0, 0)$ if $-p_1 + p_1p_2 + p_1p_3 + p_2p_3 - 2p_1p_2p_3 \le 0$
	$\left(\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, 0\right)$ if $-p_1 + p_1p_2 + p_1p_3 + p_2p_3 - 2p_1p_2p_3 \ge 0$
$\frac{4}{3} \le T < 2$	$(1, 0, 0, 0)$ if $-p_1 + p_1p_2 + p_1p_3 + p_2p_3 - 2p_1p_2p_3 \le 0$ and $-p_1 + p_1p_2 + p_1p_3 - p_1p_2p_3 + p_1p_4 - p_1p_3p_4 - p_1p_3p_4 + p_2p_3p_4 \le 0$
	$\left(\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, 0\right)$ if $-p_1 + p_1p_2 + p_1p_3 + p_2p_3 - 2p_1p_2p_3 \ge 0$ and $-p_2p_3 + p_1p_2p_3 + p_1p_4 - p_1p_2p_4 - p_1p_3p_4 + p_2p_3p_4 \le 0$
	$\left(\frac{2}{3}, \frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right)$ if $-p_1 + p_1p_2 + p_1p_3 - p_1p_2p_3 + p_1p_4 - p_1p_3p_4 - p_1p_3p_4 + p_2p_3p_4 \ge 0$ and $-p_2p_3 + p_1p_2p_3 + p_1p_4 - p_1p_2p_4 - p_1p_3p_4 + p_2p_3p_4 \ge 0$
$2 \le T < \frac{5}{2}$	$(1, 1, 0, 0)$ if $-p_1 - p_2 + p_1p_2 + p_1p_3 + p_2p_3 - 2p_1p_2p_3 + p_1p_4 + p_2p_4 - 2p_1p_2p_4 + p_3p_4 - 2p_1p_3p_4 - 2p_2p_3p_4 + 3p_1p_2p_3p_4 \le 0$
	$\left(\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}\right)$ if $(p_2p_3 + p_2p_4 + p_3p_4 - 2p_2p_3p_4)(1 - p1) + p_1 \le p_1 + p_2 - p_1p_2$
$\frac{5}{2} \le T < 3$	$(1, 1, 0, 0)$ if $p_2p_3 + p_2p_4 + p_3p_4 - 2p_2p_3p_4 + p_1 - p1p2 \ge p_1 + p_2 - p_1p_2$
	$\left(1, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}\right)$ if $-p_2 + p_2p_3 + p_2p_4 + p_3p_4 - 2p_2p_3p_4 \ge 0$
$3 \le T < 4$	$(1, 1, 1, 0)$

图3

算法1（一级对称分配）

121: **for** each subset size choice $m = 1, \ldots, n$ **do**
122: **for** each $M = \lceil \frac{m}{T} \rceil, \ldots, m$ **do**
123: Compute the probability of accessing exactly M out of the m nonempty nodes: $P_M = \sum_{r \in \mathcal{P}(\{(\frac{m}{M})\})} (\prod_{k \in r} p_k) (\prod_{l \notin r} (1 - p_l))$.
124: **end for**
125: Compute the success probability for subset size m: $P_{\text{succ},m} = \sum_{M=\lceil \frac{m}{T} \rceil}^{m} P_M$.
126: **end for**
127: Obtain the optimal subset size choice m with the largest $P_{\text{succ},m}$.

图4

算法2（算法1的简化版）

131: **for** each subset size choice $m = \lfloor T \rfloor, \ldots, \lfloor K_{\max} T \rfloor$ **do**
132: Compute the average access probability for the m nodes: $p_{\text{avg}} = \frac{\sum_{i=1}^{m} p_i}{m}$.
133: **for** each $M = \lceil \frac{m}{T} \rceil, \ldots, m$ **do**
134: Compute $P'_M = \binom{m}{M} \cdot p_{\text{avg}}^M \cdot (1 - p_{\text{avg}})^{(m-M)}$.
135: **end for**
136: Compute the success probability for subset size m: $P_{\text{succ},m} = \sum_{M=\lceil \frac{m}{T} \rceil}^{m} P'_M$.
137: **end for**
138: Obtain the optimal subset size choice $mOpt$ with the largest $P_{\text{succ},mOpt}$.

图5

算法3（二级对称分配）

1401: **for** each subset size choice $m_1 = 1, \ldots, mOpt_{\text{oneLevel}}$ **do**

1402: Compute the average access probability of the m_1 nodes in the 1st level: $p_{\text{avg},1} = \frac{\sum_{i=1}^{m_1} p_i}{m_1}$.

1403: **for** each subset size choice $m_2 = \lfloor K_{\min} T - 2m_1 \rfloor$, $\lfloor (K_{\min}+1)T - 2m_1 \rfloor, \ldots, \lfloor K_{\max} T - 2m_1 \rfloor$ **do**

1404: Compute the average access probability of the m_2 nodes in the 2nd level: $p_{\text{avg},2} = \frac{\sum_{i=m_1+1}^{m_1+m_2} p_i}{m_2}$.

1405: Compute the set $\mathcal{M}_{m_1,m_2}$ of all pairs $(m_{\text{acc},1}, m_{\text{acc},2})$ that allow successful recovery, as given by (6).

1406: Compute the success probability corresponding to the current choice of (m_1, m_2):

$$P_{\text{succ},m_1,m_2} = \sum_{(m_{\text{acc},1},m_{\text{acc},2}) \in \mathcal{M}_{m_1,m_2}} P_{\text{acc},1} \cdot P_{\text{acc},2},$$

where

$$P_{\text{acc},1} = \binom{m_1}{m_{\text{acc},1}} \cdot p_{\text{avg},1}^{m_{\text{acc},1}} \cdot (1 - p_{\text{avg},1})^{(m_1 - m_{\text{acc},1})},$$

$$P_{\text{acc},2} = \binom{m_2}{m_{\text{acc},2}} \cdot p_{\text{avg},2}^{m_{\text{acc},2}} \cdot (1 - p_{\text{avg},2})^{(m_2 - m_{\text{acc},2})}.$$

1407: Compare P_{succ,m_1,m_2} with the current optimal $P_{\text{succ},mOpt_1,mOpt_2}$.

1408: **if** $P_{\text{succ},m_1,m_2} > P_{\text{succ},mOpt_1,mOpt_2}$ **then**

1409: Set $P_{\text{succ},mOpt_1,mOpt_2} = P_{\text{succ},m_1,m_2}$.

1410: Set $mOpt_1 = m_1$, and $mOpt_2 = m_2$.

1411: **end if**

1412: If $mOpt_2$ keeps the same value for $D = \frac{2(n-m_1)}{3}$ steps, then we consider this value of $mOpt_2$ to be the best choice of m_2 for the current choice of m_1, and continue to the next value of m_1.

1413: **end for**

1414: **end for**

1415: Obtain the optimal choice of $(mOpt_1, mOpt_2)$ with the largest value of $P_{\text{succ},mOpt_1,mOpt_2}$.

图6

算法4（三级对称分配）

1501: **for** each subset size choice $m_1 = 1, \ldots, mOpt_{1,\text{twoLevel}} + mOpt_{2,\text{twoLevel}}$ **do**

1502: Compute the average access probability of the m_1 nodes in the 1st level: $p_{\text{avg},1} = \frac{\sum_{i=1}^{m_1} p_i}{m_1}$.

1503: **for** each subset size choice $m_2 = 1, \ldots, n - m_1 - 1$ **do**

1504: Compute the average access probability of the m_2 nodes in the 2nd level: $p_{\text{avg},2} = \frac{\sum_{i=m_1+1}^{m_1+m_2} p_i}{m_2}$.

1505: **for** each subset size choice $m_3 = \lfloor K_{\min}T - 3m_1 - 2m_2 \rfloor, \lfloor (K_{\min}+1)T - 3m_1 - 2m_2 \rfloor, \ldots, \lfloor K_{\max}T - 3m_1 - 2m_2 \rfloor$ **do**

1506: Compute the average access probability of the m_3 nodes in the 3rd level: $p_{\text{avg},3} = \frac{\sum_{i=m_1+m_2+1}^{m_3+m_2+m_3} p_i}{m_3}$.

1507: Compute the set $\mathcal{M}_{m_1,m_2,m_3}$ of all triples $(m_{\text{acc},1}, m_{\text{acc},2}, m_{\text{acc},3})$ that allow successful recovery, as given by (7).

1508: Compute the success probability corresponding to the current choice of (m_1, m_2, m_3):

$$P_{\text{succ},m_1,m_2,m_3} = \sum_{(m_{\text{acc},1},m_{\text{acc},2},m_{\text{acc},3}) \in \mathcal{M}_{m_1,m_2,m_3}} P_{\text{acc},1} \cdot P_{\text{acc},2} \cdot P_{\text{acc},3},$$

where

$$P_{\text{acc},L} = \binom{m_L}{m_{\text{acc},L}} \cdot p_{\text{avg},L}^{m_{\text{acc},L}} \cdot (1 - p_{\text{avg},L})^{(m_L - m_{\text{acc},L})}$$

for $L \in \{1, 2, 3\}$.

1509: Compare $P_{\text{succ},m_1,m_2,m_3}$ with the current optimal $P_{\text{succ},mOpt_1,mOpt_2,mOpt_3}$.

1510: **if** $P_{\text{succ},m_1,m_2,m_3} > P_{\text{succ},mOpt_1,mOpt_2,mOpt_3}$ **then**

1511: Set $P_{\text{succ},mOpt_1,mOpt_2,mOpt_3} = P_{\text{succ},m_1,m_2,m_3}$.

1512: Set $mOpt_1 = m_1$, $mOpt_2 = m_2$, and $mOpt_3 = m_3$.

1513: **end if**

1514: If $mOpt_3$ keeps the same value for $D = \frac{2(n-m_1-m_2)}{3}$ steps, then we consider this value of $mOpt_3$ to be the best choice of m_3 for the current choice of (m_1, m_2), and continue to the next value of m_2.

1515: **end for**

1516: ...

1517: **end for**

1518: **end for**

1519: Obtain the optimal choice of $(mOpt_1, mOpt_2, mOpt_3)$ with the largest value of $P_{\text{succ},mOpt_1,mOpt_2,mOpt_3}$.

图7

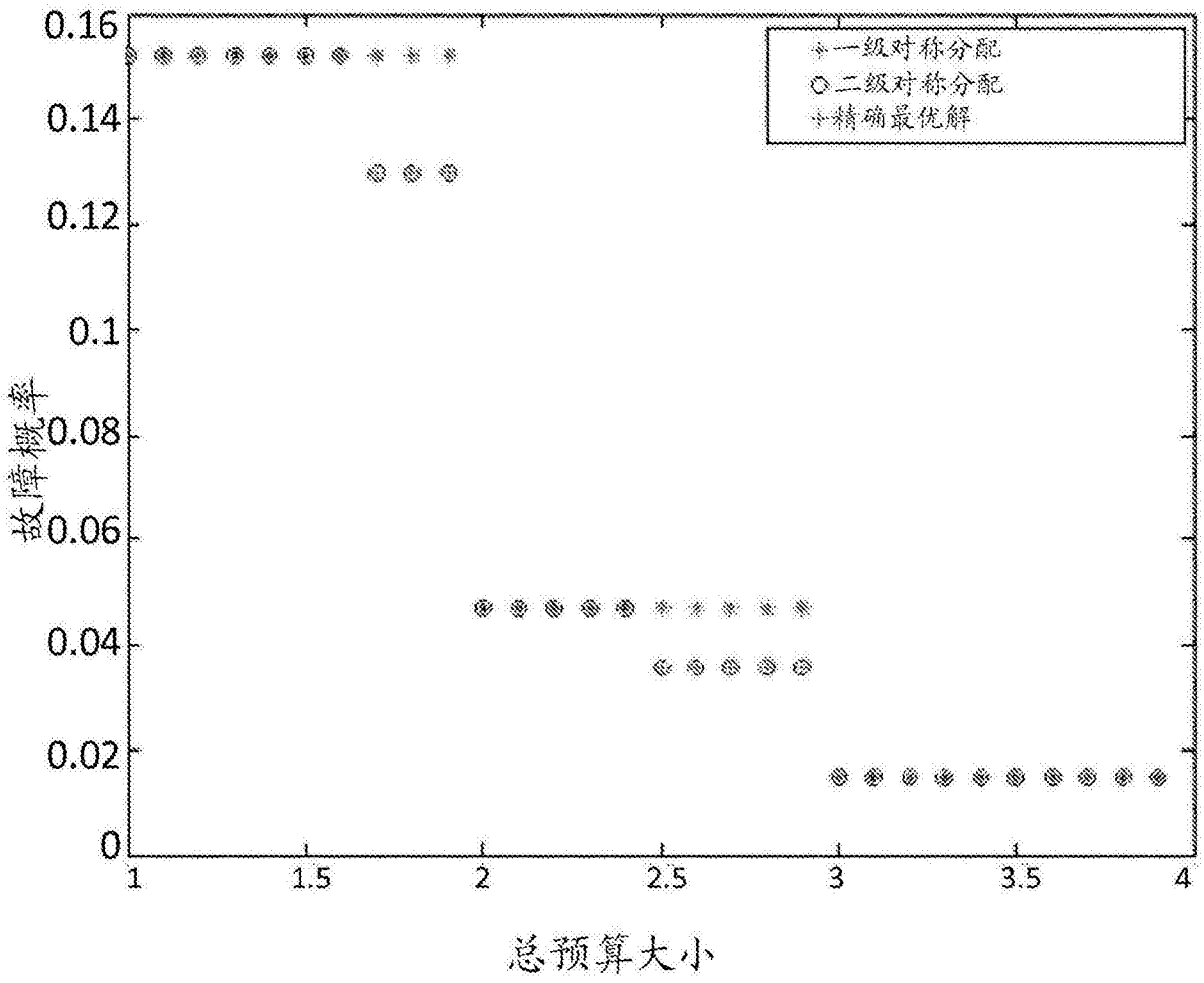

一级对称分配
二级对称分配
精确最优解
故障概率
总预算大小
0.16
0.14
0.12
0.1
0.08
0.06
0.04
0.02
0
1
1.5
2
2.5
3
3.5
4

图8

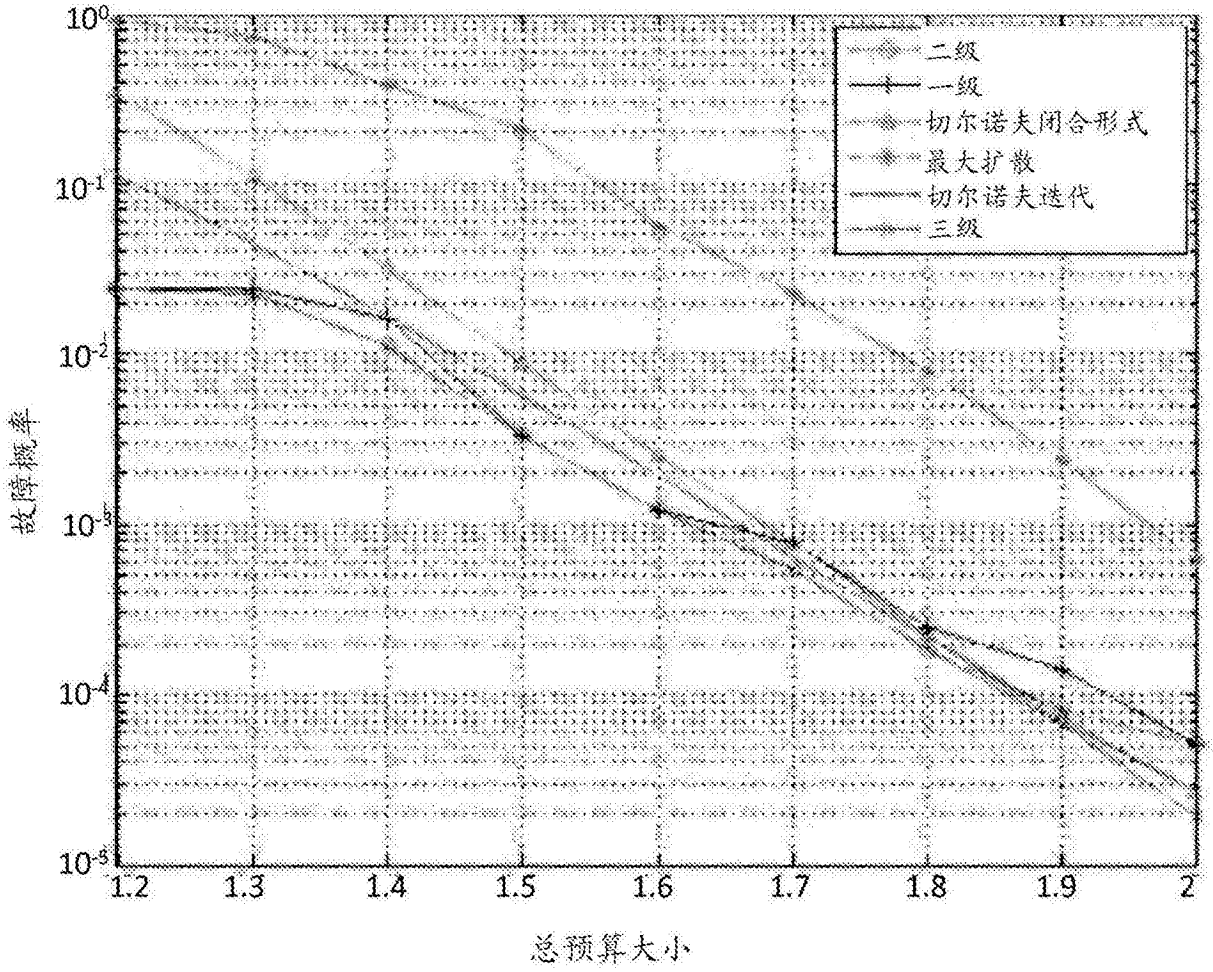

二级
一级
切尔诺夫闭合形式
最大扩散
切尔诺夫迭代
三级
故障概率
总预算大小
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
1.2
1.3
1.4
1.5
1.6
1.7
1.8
1.9
2

图9

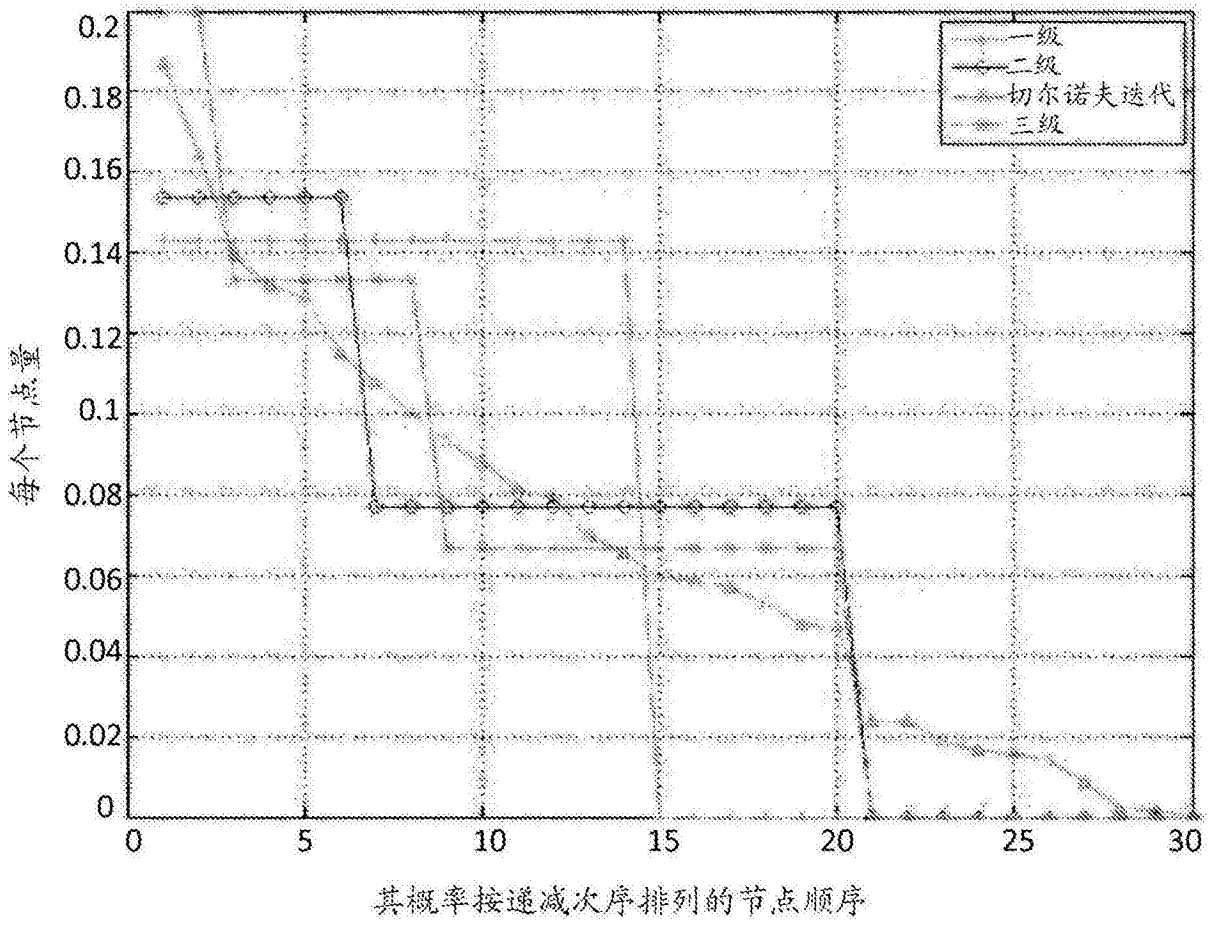

0.2
0.18
0.16
0.14
0.12
0.1
0.08
0.06
0.04
0.02
0
每个节点量
0
5
10
15
20
25
30
其概率按递减次序排列的节点顺序
一级
二级
切尔诺夫迭代
三级

图10

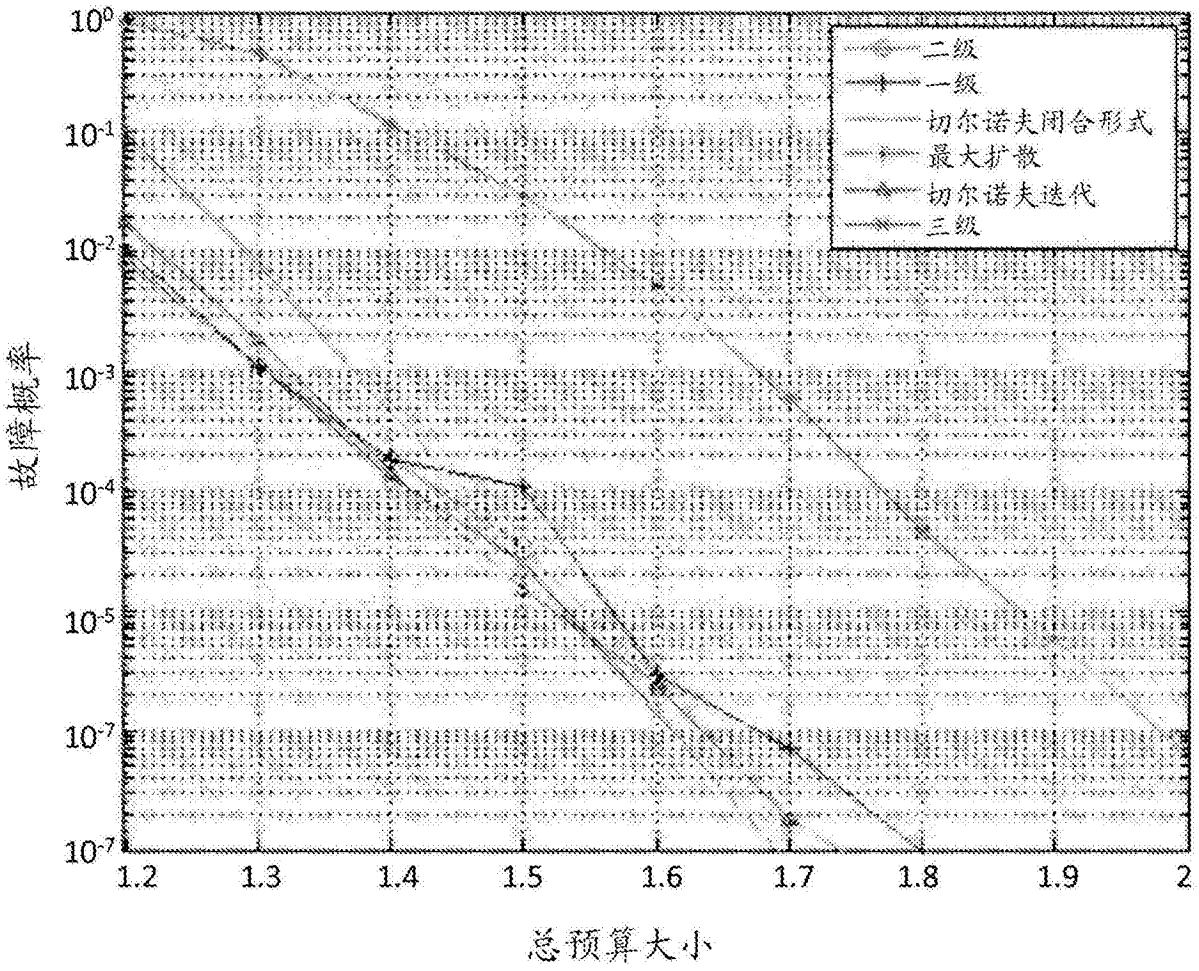

二级
一级
切尔诺夫闭合形式
最大扩散
切尔诺夫迭代
三级
故障概率
总预算大小
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
10^{-7}
10^{-7}
1.2
1.3
1.4
1.5
1.6
1.7
1.8
1.9
2

图11

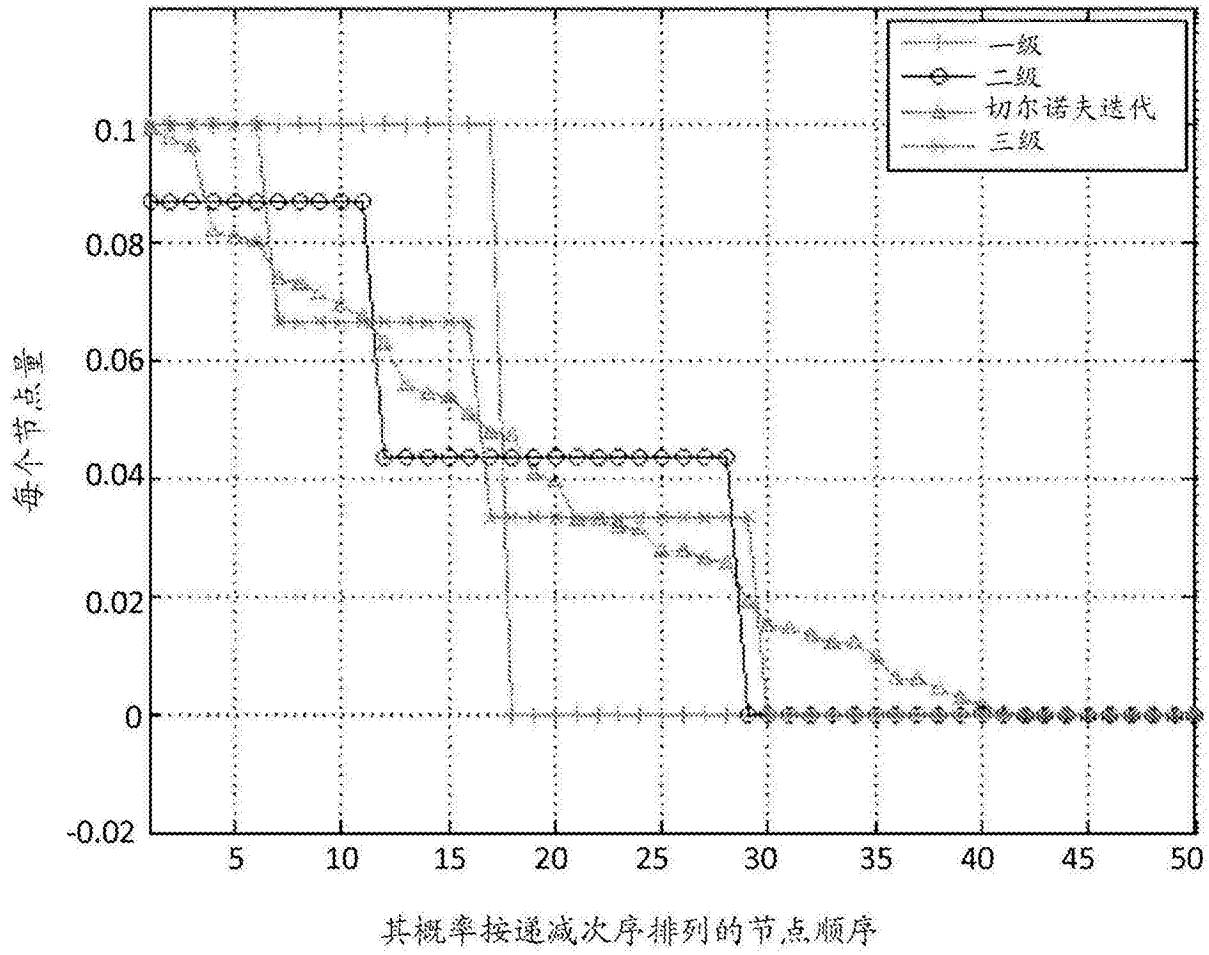

一级
二级
切尔诺夫迭代
三级
0.1
0.08
0.06
0.04
0.02
0
-0.02
5
10
15
20
25
30
35
40
45
50
每个节点重
其概率按递减次序排列的节点顺序

图12

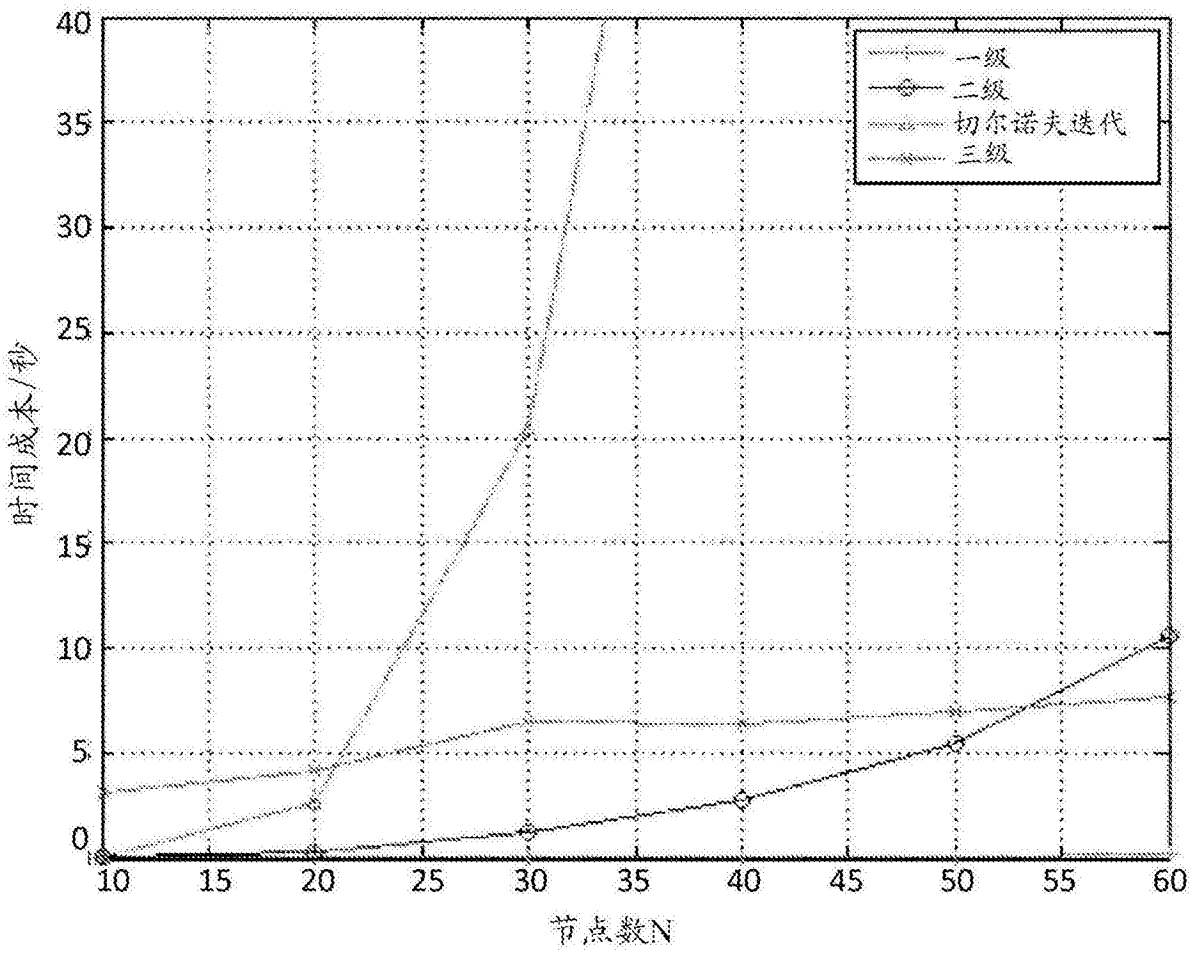

一级
二级
切尔诺夫迭代
三级
时间成本/秒
节点数N
0
5
10
15
20
25
30
35
40
10
15
20
25
30
35
40
45
50
55
60

图13

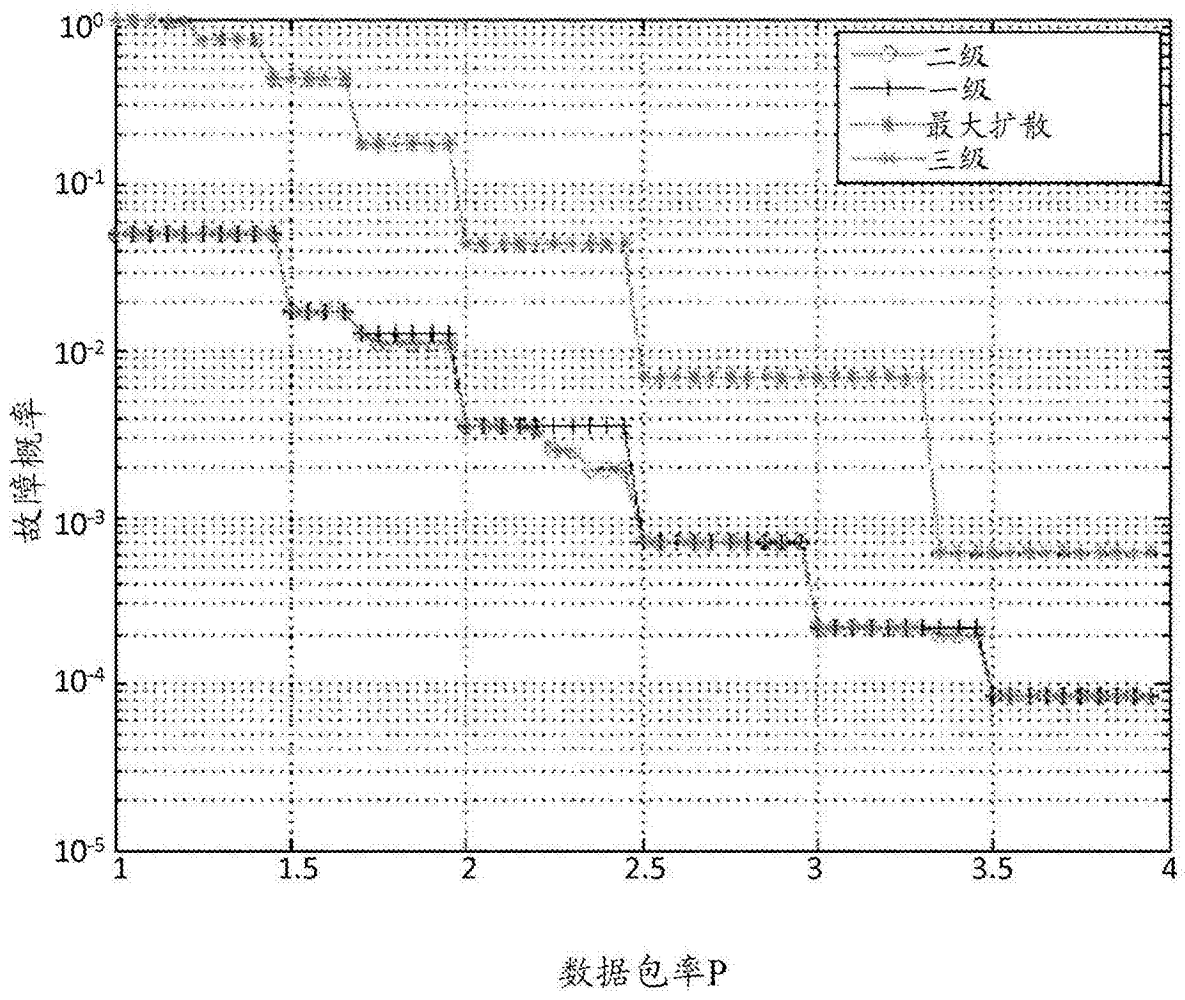


图14

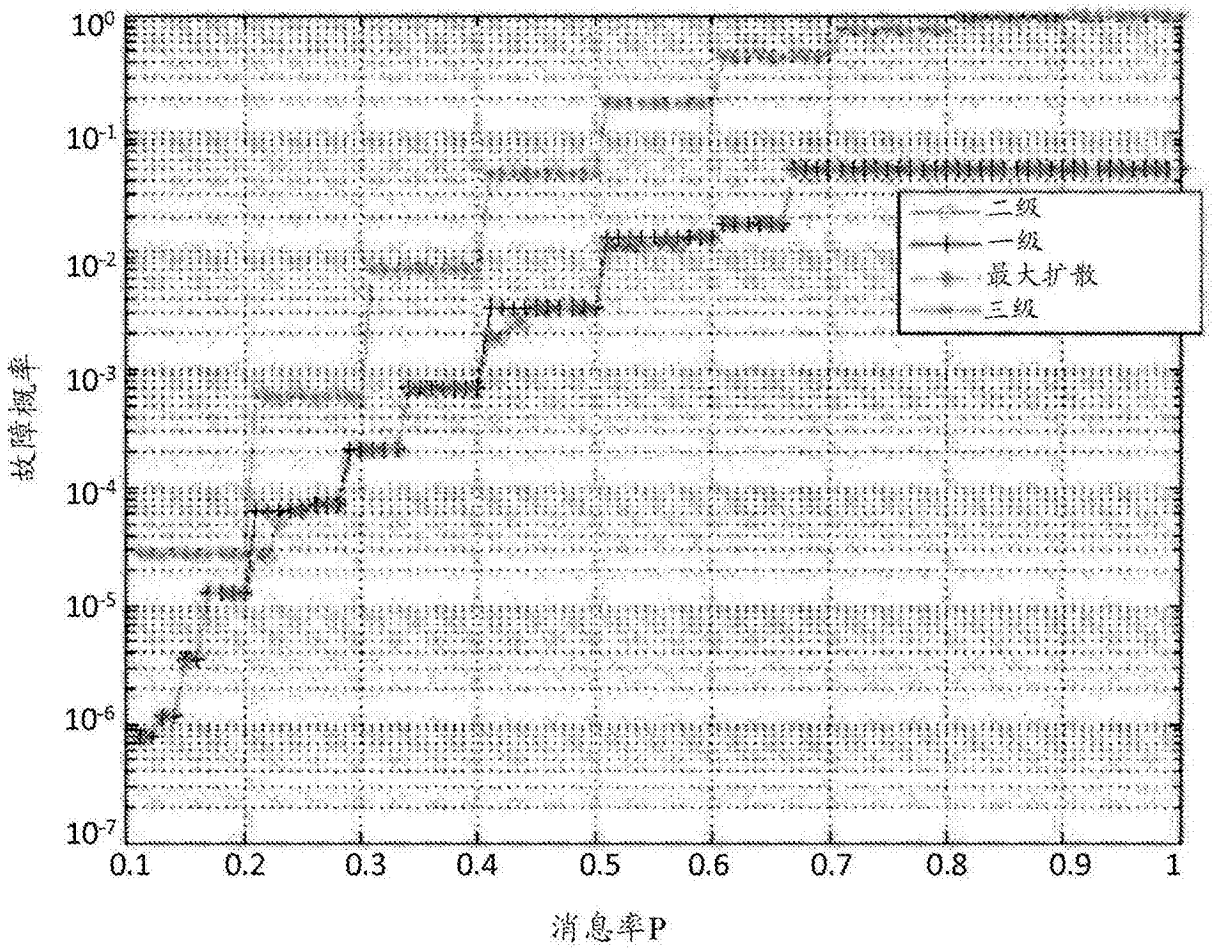

10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
10^{-6}
10^{-7}
故障概率
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
消息率P
二级
一级
最大扩散
三级

图15

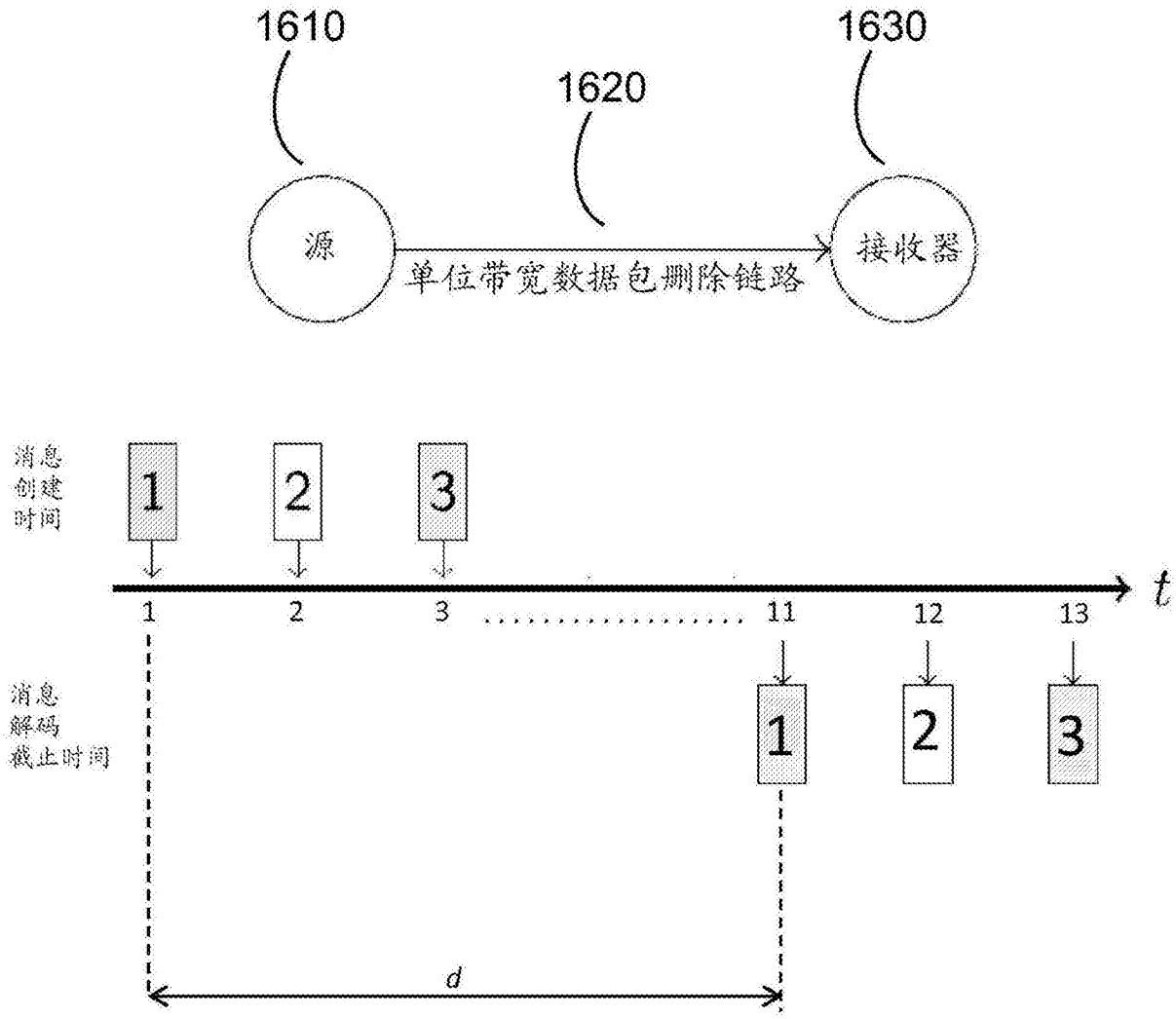

1610
1620
1630
源
单位带宽数据包删除链路
接收器
消息
创建
时间
1
2
3
t
1
2
3
11
12
13
消息
解码
截止时间
1
2
3
d

图16

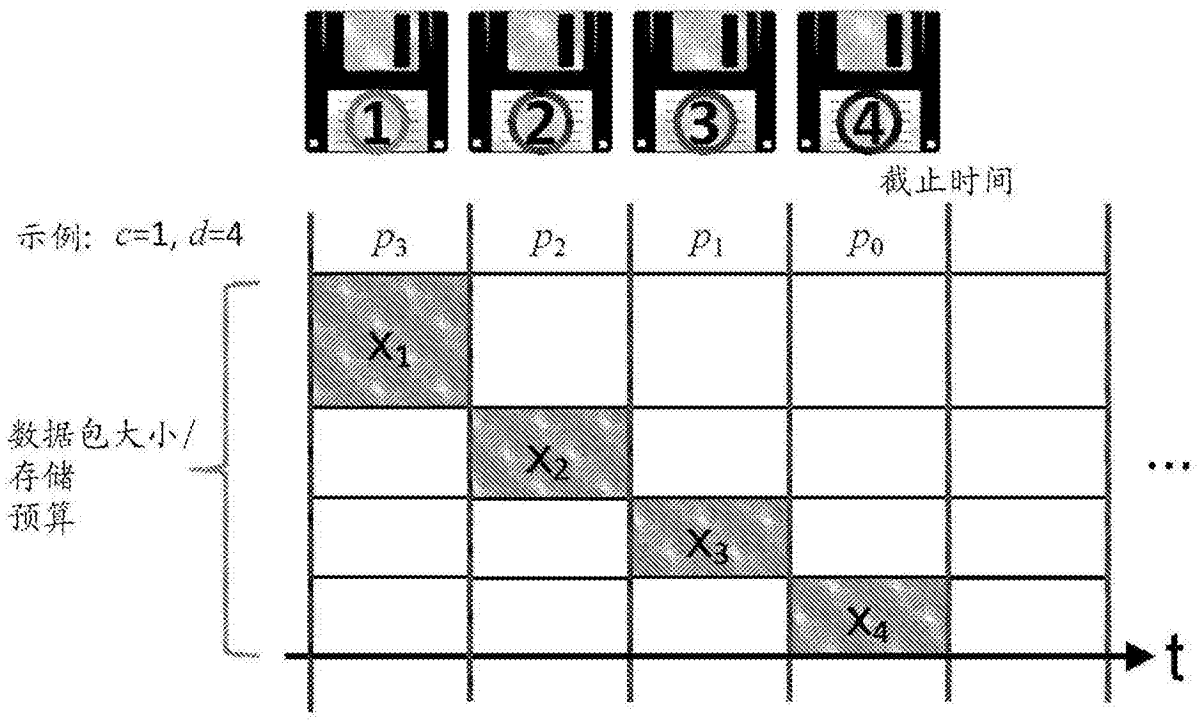

1
2
3
4
截止时间
示例: c=1, d=4
p_3
p_2
p_1
p_0
X_1
X_2
X_3
X_4
数据包大小/
存储
预算
...
t

图17A

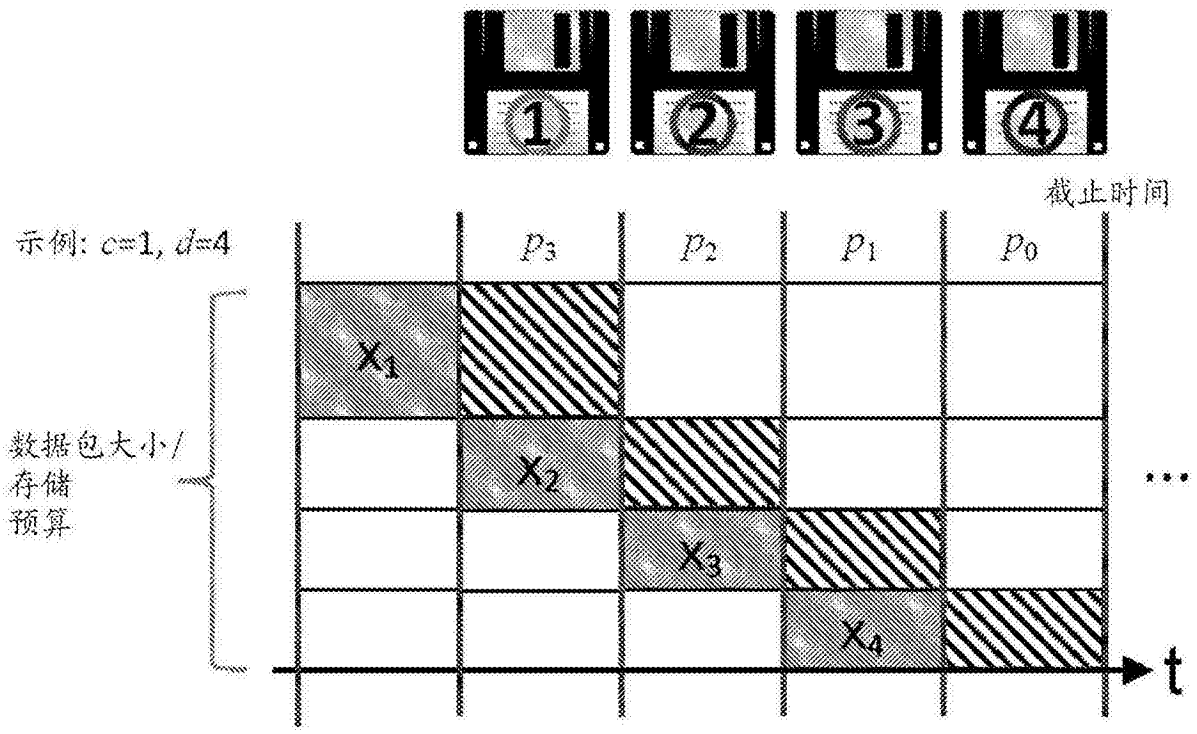

1
2
3
4
截止时间
示例: c=1, d=4
p_3
p_2
p_1
p_0
X_1
X_2
X_3
X_4
数据包大小/
存储
预算
...
t

图17B

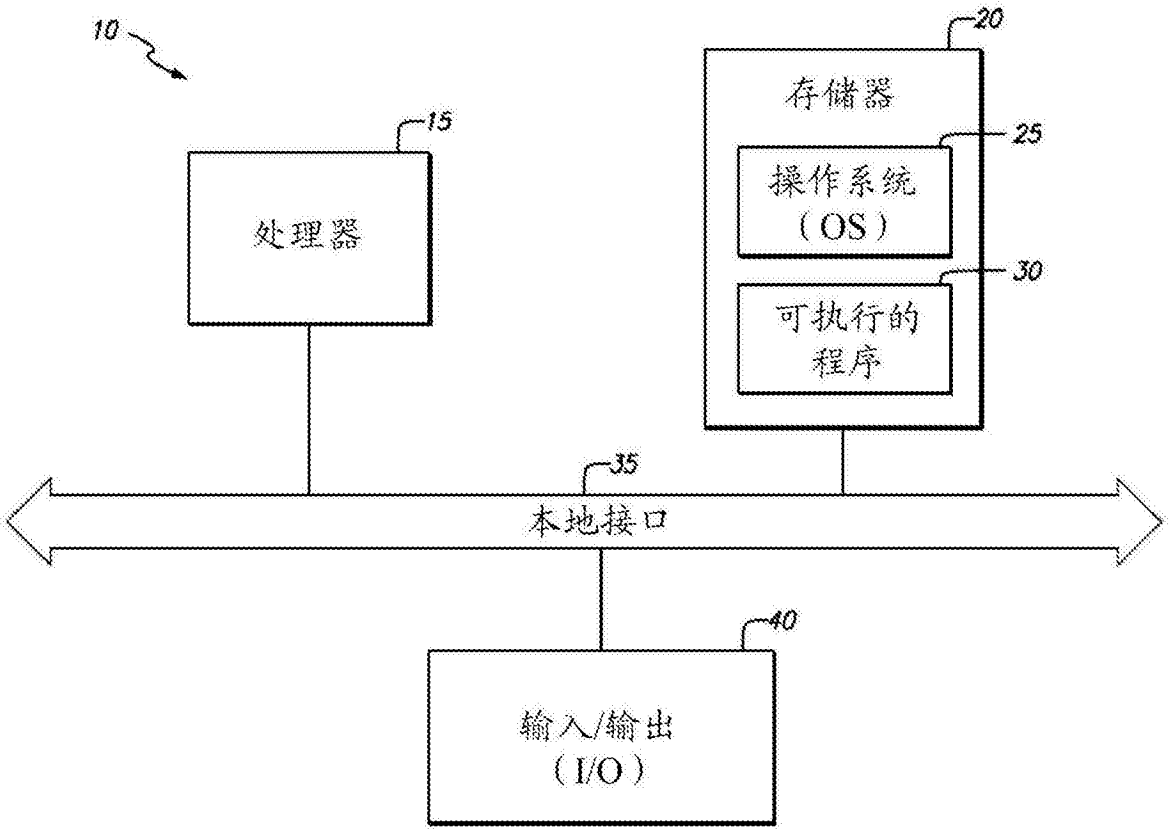


图18