基于热度的 Hadoop 快速副本复制算法①

张 倩, 郑 烇, 王 嵩

(中国科学技术大学 自动化系, 合肥 230027)

摘 要: 在云存储中心, 由于节点失效带来的文件数据块副本丢失不仅会影响系统的可靠性, 还会影响文件的并发访问效率. 针对 Hadoop 中默认的副本复制方法存在的问题, 即副本复制过程某些节点数据传输过于集中, 负载不均衡, 磁盘 I/O 吞吐率低, 提出一种基于热度的快速副本复制算法. 该算法优先复制热度高的数据块, 合理选择数据块复制的源节点和目的节点. 仿真结果表明, 该算法平衡了系统的工作负载, 提高了磁盘 I/O 吞吐率, 显著降低用户请求平均响应时间.

关键词: 云存储; 节点失效; Hadoop; 副本复制; 热点

Rapid Replica Copy Algorithm Based on Popularity in Hadoop

ZHANG Qian, ZHENG Quan, WANG Song

(Department of Automation, University of Science and Technology of China, Hefei 230027, China)

Abstract: In cloud storage centers, replica of file may be lost because of the failure of nodes, which will affect the reliability of system, as well as the efficiency of file concurrent access. There are some deficiencies in the default replica copy algorithm in Hadoop, such as a concentration of data transfer process on a few DataNodes, load imbalance, low disk I/O throughput. To address this issue, this paper proposes a rapid replica copy algorithm based on popularity in Hadoop. It handles the popular block firstly, and chooses source and destination DataNodes properly. The simulation results show that the proposed algorithm improves the disk I/O throughput, load balance, and reduces average service response time significantly.

Key words: cloud storage; node failure; Hadoop; replica copy; popularity

随着云计算平台的不断普及以及大数据的出现, 如何有效地存储和管理数据以不断满足用户并发访问需求已成为一个重要的问题. 云存储架构中关键组成是分布式文件系统, 目前典型的分布式文件系统存储框架包括: Google 文件系统[1](GFS), Hadoop 分布式文件系统[2](HDFS)和 Amazon 存储服务[3](S3). 其中, HDFS 是一个开源的文件系统, HDFS 的工作机制类似于 GFS, 但它是轻量级的, 提供了更好的可扩展性, 并且适用于大规模分布式数据处理. 云存储系统中, 各节点性能大不相同, 大部分节点都是成本较低并且不够稳定. 此外, 由于网络连接的不可靠性以及带宽的限制, 节点很容易出现故障而失效. 一旦节点出现故障则存储在该节点上的数据将丢失. 节点失效的情况即使是在最先进的云供应商中也会出现, 例如, 2009 年 3 月微软云计算平台 Azure 中止服务 22 小时; 2009 年 6 月, Rackspace 遭受了严重的云服务中断故障, 供电设备跳闸, 备份发电机失效, 不少机架上服务器停机[4]. 在大规模存储系统中, 广泛使用块备份机制, 数据都是分布式并发访问的, 如果节点出现故障数据丢失, 则其他正常节点的访问量将大大提高, 从而增加这些节点的负担, 严重时可能导致请求阻塞或者被拒绝. 因此节点失效时快速复制丢失副本到指定数量对分布式并发访问至关重要.

目前, 在云存储平台中, 为了减少节点负担过重,

① 基金项目:国家自然科学基金(61174062)

收稿时间:2014-12-30;收到修改稿时间:2015-02-02

提高文件并发访问效率. 很多研究者提出副本管理和副本复制的算法, 这些算法主要考虑副本放置位置来平衡集群负载, 或者自适应改变副本数量提高系统性能. 文献[5]提出了一种快速自适应的块部署算法, 该算法会自适应改变副本数量, 提高系统容错性能. 文献[6]提出一种分布式系统中有效的容错算法, 该算法可以最小化故障的发生, 有效减小执行时间. 文献[7]提出一种分布式数据副本自适应算法(DARE), 该算法可以获得更好的数据放置位置. 文献[8]提出一种有效的动态副本管理算法(CDRM), 该方法根据工作负载的变化以及节点容量调整副本数量和位置, 提高系统性能以及负载平衡能力. 以上副本管理的算法中大部分算法集中研究副本的部署来提高系统性能, 并没有考虑节点故障, 对集群节点失效后副本数量恢复到指定副本因子数的研究很少. 文献[9]提出一种有效的节点故障后副本重部署算法, 该算法保证副本重部署过程中最优化节点之间块数量传输差异, 但是该方法没有考虑实际集群工作环境的影响.

根据数据访问的局部性原理, 当前被频繁访问的数据在将来一段时间内可能还会有较高的访问频率, 而当前很少被访问的数据在将来一段时间内可能仍然很少被访问. 且用户在数据访问高峰时期, 访问量可能以爆发形式出现, 此时系统负载相当繁重. 这种情况下节点失效, 文件副本数小于副本因子, 若文件并发访问量超过文件副本数, 则网络带宽和硬件资源竞争激烈, I/O 吞吐率会明显下降. 本文优先将最有可能被访问的文件恢复到指定的数量, 使文件访问分布到整个集群, 提高系统整体的传输效率和服务性能. 为了解决分布式并发访问环境下集群节点失效时将副本数量恢复到副本因子值的问题, 本文提出一种有效的基于热度的快速副本复制算法(HPRC).

1 Hadoop默认的副本复制技术及问题分析

1.1 节点失效副本复制方法

Hadoop 分布式文件系统[10](HDFS)是一个高容错性的系统, 并且部署在低廉的硬件上. HDFS 能提供高吞吐量的数据访问, 非常适合大规模数据集的应用. 整个 HDFS 系统由数百或上千存储着文件数据片段的服务器组成, 正是由于这些低廉的硬件设备加上系统本身组成部分复杂巨大, 所以 HDFS 硬件故障是常态而不是异常. 因此, 故障的检测和自动快速恢复是 HDFS 一个核心的设计目标.

HDFS 是基于主从架构的系统, 它由一个名字节点和多个数据节点组成. 名字节点存储文件元数据信息同时管理集群中的所有节点. 数据节点存储数据并且执行基于 MapReduce 的数据处理. 其中文件以块的形式进行存储, 为了保证存储的可靠性和可用性, 块以多副本形式进行放置. 当一个数据节点从集群中删除或失效时, Hadoop 为了保证指定的副本因子, 会从一个数据节点将丢失块复制到其他存活的节点上.

Hadoop 中默认的副本复制过程如图 1 所示:

① 名字节点检测到数据块备份数少于副本因子, 发起重新备份命令;

② 名字节点负责所有块复制的调度过程, 随机选择参与复制的源节点和目的节点;

③ 名字节点定期给源节点发送指令;

④ 源节点响应命令将数据块复制到目的节点;

⑤ 目的节点完成块的复制后将确认信号发送给源节点;

⑥ 源节点再发送确认信号给名字节点.

以上过程持续进行直到所有丢失的块全部复制成功.

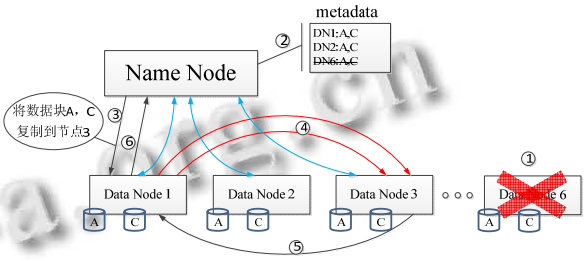

图 1 Hadoop 节点失效副本复制过程

1.2 问题分析

Hadoop 默认的副本复制过程中, 首先, 需备份的数据块添加到待备份列表时没有考虑优先级. Rabinovich 等人研究表明, 存储系统中的数据被访问的概率呈类 Zipf 分布[11], 因此, 当大量热点数据集中到某个或某些存储节点上, 一旦节点出现故障, 如果这些副本不能快速恢复到指定数量, 那么其他备份节点的访问量过大, 网络带宽和磁盘竞争加大, 最终将会影响文件的并发访问效率. 所以需要考虑数据块的优先级, 优先级高的先添加到待备份列表.

其次，副本复制过程中名字节点选择丢失数据块的复制源节点和目的节点是随机性的，即随机选择拥有数据块的一个节点作为源节点，随机选择不含该数据块的一个节点作为目的节点. 该策略会使得某些节点数据传输过于集中，文献[9]中给出的实验表明，每个数据节点复制块的数量大不相同且不稳定，某些时刻某节点接收的块数量很多，也就是副本复制操作过于集中在某个目的节点上，发送和接收数据块的过程是不平衡的，此时集群节点 I/O 吞吐率会下降，复制过程甚至出现停滞. 源节点和目的节点随机选择的策略也忽略了节点本身的性能，若节点本身负载很大，则仍然被选中，将带来复制过程延时，进一步降低了副本复制的效率.

1.3 解决策略

针对 Hadoop 默认的副本复制过程中在数据块加入待备份列表时没有考虑优先级的情况，本文提出了数据块热度的思想，周期性对热度进行计算，优先将访问率很高的热点文件进行复制.

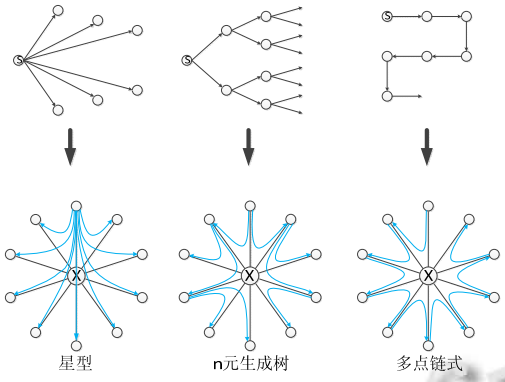

图 2　集群网络拓扑

文献[12]针对大规模集群研究了三种逻辑网络拓扑，如图 2 所示. 实验评估表明在集群节点增加时星型拓扑结构会出现严重链路拥塞；由于网络带宽的限制 n 元生成树拓扑结构无法将数据有效地复制到多个流；多点链式拓扑结构则在以上情况下都可以有效地复制数据. 因此本文集群采用单向环结构，如图 3 所示. 一个节点只能将数据传输到它的后继节点，该结构保证在源节点确定后目的节点为其后继节点，唯一确定，从而使得源节点的数据发送和目的节点的数据接收达到平衡，避免复制操作集中在某些节点上. 在选择源节点时，为了减少节点延时，优先选择并发访问率低的节点，从而平衡节点负载，提高副本复制效率.

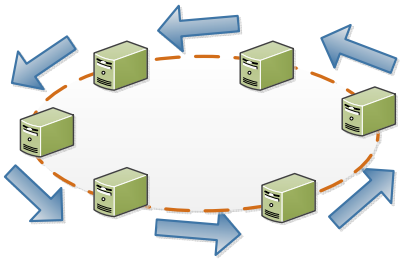

图 3　节点单向环结构

综上所述，本文副本复制方法在检测到节点失效时，首先将丢失块根据热度参数进行排序，优先将热度高的数据块信息添加到待备份列表中. 数据复制时在候选源节点中选择并发访问率最小的节点作为源节点，从而避免源节点负载过重，数据节点间以单向环结构进行数据传输，从而平衡源节点与目的节点间的数据传输量，最终提高副本复制效率，减少大量用户请求平均响应时间.

2　基于热度的副本复制算法

2.1 符号及定义

首先，假设集群中所有节点集合为 $A=\{D_1,D_2,...,D_n\}$，失效的节点集合为 $F=\{D_m,D_{m+1},...,D_n\}$，剩余存活的节点集合为 $L=A-F=\{D_1,D_2,...,D_{m-1}\}$. 总数据块集合为 $B=\{B_1,B_2,...,B_M\}$，失效节点上丢失的数据块集合为 $B'=\{B'_1,B'_2,...,B'_N\}$.

定义 1. 数据块热度值. 设采样周期为 T，定义如下:

$$H_T(B_j)=\alpha P_T(B_j)+(1-\alpha)H_{T-1}(B_j) \quad (1)$$

$$P_T(B_j)=\frac{Count_{B_j}}{\sum_{k=1}^{M}Count_{B_k}};\ j=1,2,...,M \quad (2)$$

其中，$H_T(B_j)$ 表示 T 时刻数据块 B_j 的热度值；$H_{T-1}(B_j)$ 表示 T-1 时刻数据块 B_j 的热度值；$\alpha(\alpha\in[0,1])$ 为热度调节因子；$P_T(B_j)$ 表示 T 时刻数据块 B_j 的访问频率(访问该数据块的次数与访问所有数据块次数的比值)；$Count_{B_j}$ 为第 j 个数据块 T-1 到 T 时刻被访问的次数；$\sum_{k=1}^{M}Count_{B_k}$ 为所有数据块被

访问的次数.

从式(1)可得, 随着时间的推移, 用户访问距离当前时间越久对文件的当前热度影响越小, 访问较近对文件的热度影响较大, 若近期访问量越大, 则文件的热度越高. 该热度计算公式避免了突发性访问带来的文件热度突增的情况, 从而使热度变化更加平稳.

定义 2. 节点并发访问率. 定义某时刻, 节点的并发访问率(concurrent access rate)为该节点当前所有任务数与最大任务数的比值. 定义如下:

$$CAR_{D_i}=\frac{TaskNum_{D_i}}{SumTaskNum_{D_i}};i=1,2,...,n \quad (3)$$

其中: CAR_{D_i} 为第 i 个节点并发访问率; $TaskNum_{D_i}$ 为第 i 个节点的当前任务数量; $SumTaskNum_{D_i}$ 为第 i 个节点最大的并行任务数量.

2.2 副本复制算法描述

Hadoop 中当数据节点由于某种原因发生故障时, 名字节点检测到故障信息, 同时会采用副本复制策略将副本快速恢复到指定数量以支持系统的可用性与可靠性. 具体副本复制步骤如下:

① 失效节点上丢失的块 B' 以数据块热度 H 为优先级, 热度高的数据块优先处理;

② 源节点选择(srcNode). 名字节点会检测到所有丢失块的其他副本所在的节点位置, 用 S_j 表示块 $B'_j\left(j=1,2,...,N\right)$ 的所有源节点的候选节点集合(若候选节点沿单向环结构的后继节点仍为候选节点, 则将该节点从候选节点中删除), 选择 S_j 中并发访问率最低的节点作为该数据块的复制源节点;

③ 目的节点选择(targetNode). 根据单向环结构将数据块从源节点复制到其后继节点.

副本复制算法的伪代码描述如下:

算法. 基于热度的副本快速复制算法(HPRC)

输入: 需要备份的 block 列表 $B'_j\left(j=1,2,...,N\right)$

输出: 待备份的任务队列 W

```
//将数据块 B' 按热度 H 进行降序排序
sort B' in descending order
for j from 1 to N
//源节点候选节点集合, K 为候选节点个数, K' 为实际候选节点个数
    K' = K
//删除不满足条件的候选节点
    for i from 1 to K-1
        if S_{j,i+1} next to S_{j,i} in ring
            delete S_{j,i} from S_j
            K' = K' - 1
        end if
    end for
    if S_1 next to S_{j,K} in ring
        delete S_{j,K} from S_j
        K' = K' - 1
    end if
//候选节点中选择并发访问率最小的节点作为源节点, min 初值为 1
    k = 1
    for i from 1 to K'
        if CAR_{S_{j,i}} < min then
            min = CAR_{S_{j,i}}
            k = i
        end if
    end for
    srcNode = S_{j,k}
//目的节点为源节点单向环方向的后继节点
    targetNode = S_{j,k+1}
//将数据块复制信息添加到任务队列
    W.add(B'_j, srcNode, targetNode)
end for
```

3 仿真实验及分析

为了评估本文提出的 HPRC 算法的性能, 下面将该算法与Hadoop默认的副本复制算法以及文献9提出的副本重建算法(ERRS)进行比较, 通过 MATLAB 仿真实验, 分析了算法的性能.

3.1 实验设置

采用 MATLAB 2014 仿真 Hadoop 环境. Hadoop 集群参数如表 1 所示.

表 1 集群参数表

DataNode 个数	6 个(包括失效节点)
block 大小	64M
副本因子	3
blocks 数量	600*3(副本因子)

我们测量副本复制过程中以下三个指标:

① 集群数据块负载平衡度. 计算公式如下:

$S_R=\sqrt{\frac{1}{n}\left[(r_1-\bar{r})^2+(r_2-\bar{r})^2+...+(r_n-\bar{r})^2\right]}$ 其中 $R=\{r_1,r_2,...,r_n\}$ 表示 n 个节点数据块数量的集合;

② 节点磁盘 I/O 吞吐率(单位 MB/s);

③ 文件请求平均响应时间(单位 s).

共设置六组实验, 依次为不同的 DataNode 失效, 每次实验前保证集群初始状态为平衡状态.

3.2 实验结果

(1) 实验一: 本实验目的是验证文中提出的算法是否有效解决 Hadoop 默认副本复制算法中数据块的复制集中在某些节点上的问题. 实验初始状态集群 block 块分布均匀, 采用六组实验, 每组实验分别为一个节点失效, 节点失效时将文中 HPRC 算法与 Hadoop 默认副本复制算法以及 ERRS 算法数据块负载均衡度进行比较, 如图 4 所示.

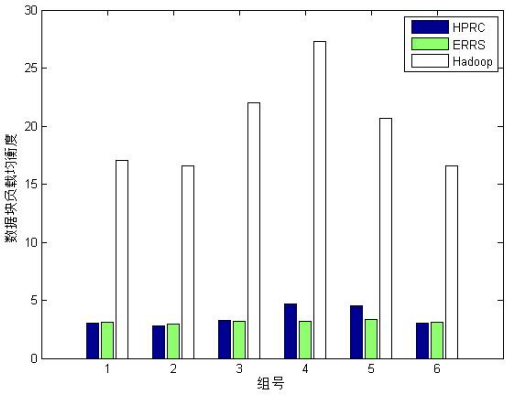

图 4　数据块负载均衡度

从图 4 中可以看出, 本文 HPRC 算法和 ERRS 算法的负载均衡度均远小于 Hadoop 默认副本复制算法的负载均衡度. 由于Hadoop默认算法在选择源节点和目的节点时采用随机化方法, block 块分布很不均匀, 某些节点工作负载很大. 本文利用单向环结构, 在源节点候选节点中选择并发访问率最低的节点作为源节点, 减少了源节点的负载, 源节点选定后目标节点唯一确定, 平衡了数据接收和发送过程, 块重新备份后分布较均匀, 减轻了某一节点过载情况下的不利影响, 负载均衡性能大大提高. ERRS 算法以最优化块数量传输差异为目的, 其负载均衡度有时甚至比本文算法更低, 但是 ERRS 算法没有考虑集群实际工作环境, 若选择的源节点实际负载很高, 则严重影响了副本复制效率.

(2) 实验二: 本实验目的是验证文中提出的算法选择并发访问率低的节点作为源节点是否有效减少副本复制时间, 同时保证系统工作负载均衡. 在节点 6 失效时计算副本复制过程 10 个采样点的平均磁盘 I/O 吞吐率. 实验分别将本文 HPRC 算法与 Hadoop 默认的副本复制算法以及 ERRS 算法进行比较, 如图 5 所示.

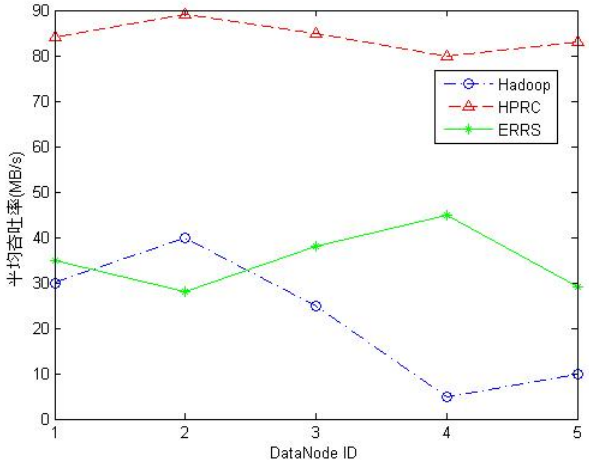

图 5　节点磁盘 I/O 吞吐率

从图5可以看出, 在集群DataNode6失效后, 副本复制过程中, 由于节点接收数据块过于集中, 此时所有节点的磁盘 I/O 吞吐率下降; 本文提出的算法在节点失效后, 优先选择访问率较低的节点作为源节点, 平衡了源节点的工作负载提高了副本复制效率, 使得节点吞吐率较高且比较稳定, 从而有效减少副本复制时间. 而 ERRS 算法未考虑实际工作环境, 平均吞吐率明显小于本文算法, 且不稳定. 但是由于其数据块分布较均匀, 因此吞吐率高于 Hadoop 默认算法.

(3) 实验三: 本实验目的是验证文中提出的算法是否会提高用户请求(主要是对热点数据块的请求)的平均响应时间. 实验中在节点 6 失效时分别执行不同数量的用户请求数, 将本文 HPRC 算法与默认的 Hadoop 副本复制算法以及 ERRS 算法比较, 如图 6 所示.

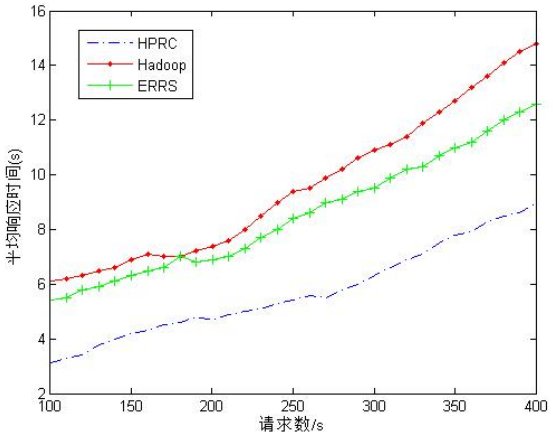

图 6 请求平均响应时间

从图 6 可以看出, 节点失效时, 本文提出的 HPRC 方法明显比 Hadoop 默认副本复制方法以及 ERRS 算法平均响应时间少, 因为热点数据块优先复制使得数据块能够并发执行, 且集群副本复制过程中吞吐率较高, 所以平均响应时间明显降低. ERRS 算法平均响应时间略低于 Hadoop 默认方法, 因为其数据块分布均匀, 系统负载较均衡.

综上所述, 本文提出的节点失效时基于热度的快速副本复制算法在合理选择源节点和目的节点的情况下, 优先复制热度高的数据块, 有效地提高了用户请求的平均响应时间, 从而提高用户服务质量.

4 结语

本文提出一种基于热度的快速副本复制算法, 该算法在集群节点失效时能快速将所有数据块重新恢复到指定的数量. 该算法基于单向环结构且考虑到节点的访问率合理选择源备份节点和目的节点, 避免数据块过于集中在某个节点接收. 考虑到副本复制过程中可能会有大量的用户数据访问, 该算法对热点数据块优先进行备份. 为了对算法性能进行测试搭建了实验平台并进行了仿真实验. 实验结果表明, 该算法有效减少了副本复制的时间且整个集群负载比较均衡, 吞吐率较高, 同时大大提高了用户请求平均响应时间. 下一步工作是基于本文的算法, 在实际集群环境下进行实验, 同时考虑数据节点失效后新数据节点加入的情况, 进一步对算法进行优化.

参考文献

1 Ghemawat S, Gobioff H, Leung ST. The Google file system. Proc. of 19th ACM Symposium on Operating Systems Principles (SOSP 2003). New York, USA. October, 2003. 29–43.

2 The Apache Software Foundation. Hadoop. http://hadoop.apache.org/core/.

3 Amazon-S3.Amazon simple storage service. http://aws.amazon.com/s3/.

4 http://www.ciotimes.com/cloud/cjs/68259.html

5 Bansal S, Sharma S, Trivedi I. A fast adaptive replication placement for ultiple failures in distributed system. Proc. Commun. Comput. Info. Sci. Tirunelveli, Tamil Nadu, India, 2011. 495–502.

6 Al-Jaroodi J, Mohamed N, Nuaimi KA. An efficient fault-tolerant algorithm for distributed cloud services. Proc. IEEE Symp. Netw. Cloud Comput. Appl. NCCA, London, United Kingdom. 2012. 1–8.

7 Abad CL, Lu Y, Campbell RH. DARE: Adaptive data replication for efficient cluster scheduling. Proc. IEEE Int. Conf. Cluster Comput. ICCC. Austin, TX,United Sstates. 2011. 159–168.

8 Wei QS, Veeravalli B, Gong BZ, et al. CDRM: a cost-effective dynamic replication management scheme for cloud storage cluster. Proc. IEEE International Conference on Cluster Computing .2010. 188–196.

9 Higai A, Takefusa A, Nakada H, et al. A study of effective replica reconstruction schemes at node deletion for HDFS. Proc. of the 14th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing. 2014. 512–521.

10 Shvachko K, Kuang H, Radia S, et al. The hadoop distributed file system. IEEE Mass Storage Systems and Technologies(MSST). 2010. 1–10.

11 Rabinovich M, Rabinovich I, Rajaraman R, et al. A dynamic object replication and migration protocol for an Internet hosting service. Proc. of the 19th IEEE International Conference on Distributed Computing Systems. 1999. 101–113.

12 Rauch F, Kurmann C, Stricker TM. Partition cast - modelling and optimizing the distribution of large data sets in PC clusters. Euro-Par 2000. LNCS 1900. 2000. 1118–1131.