数据网格中一种填空式副本分配算法

陈 磊, 李三立
(清华大学计算机科学与技术系 北京 100084)

摘 要: 在数据网格应用中, 数据会由于性能和可用性等原因进行复制. 如何使数据复本合理分布以降低通信开销是数据网格系统需要解决的重要问题. 本文针对一种简化的数据网格环境, 考虑存储资源代理对数据的访问频率和代理间的网络性能, 提出一种填空式数据副本分配算法(CDRDA). 通过该算法得到的数据副本分配, 构成多级虚拟存储架构. 数据副本根据被存储资源代理访问的频率分布在访问开销小的节点上, 从而使系统的整体访问效率达到近似最优.

关键词: 数据网格; 数据副本; 通信开销; 填空式
中图分类号: TP393 文献标识码: A 文章编号: 0372-2112 (2006) 11-1951-04

A Calking Dynamic Replication Distribution Algorithm in Data Grid

CHEN Lei, LI San-li
(*Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China*)

Abstract: Data replication is a general mechanism to improve performance and availability for data grid applications. Distributing data replica reasonably in large scale data grid systems can decrease communication cost and improve its performances. This paper proposes a new Calking Dynamic Replication Distribution Algorithm (CDRDA) considering the communication cost among storage resource brokers and the characteristic of data accessing. By the algorithm, data replica will be distributed as a multi-level virtual cache and saved at the node in which applications can acquires the average lower communication cost. At last, the performance of the algorithm and its applications are also introduced.

Key words: data grid; data replica; cost of communication; calking

1 研究背景

数据网格[1]的概念来自网格(Grid)它是网格技术在数据管理方面的应用和实现, 即是为了建立网格环境下透明访问异构数据资源的新的体系结构. 网格技术的研究目标是实现网络虚拟环境下高性能资源的共享和协同工作以解决一致使用各种分散资源的问题. 数据网格是为了解决数据密集型计算应用中, 方便高效使用分布式数据资源的问题. 其研究内容主要集中在广域、异构、分布环境下如何对数据进行管理[2]; 如何从地理分布的各种异构数据资源中获取数据, 并通过地域分布的协作和处理从数据源中获取有用信息.

在数据网格应用中, 为保证资源的就近访问, 需要在存储资源代理上建立数据副本[6]. 当一个代理上的副本不能满足应用的需要时, 代理需要动态的从其他代理或者数据节点上获取副本. 比较典型的存储资源管理模型主要有 SDSC(San Diego Supercomputer Center)的 SRB[3] 和 LBNL(Lawrence Berkeley National Laboratory) 的 SRMs(The Storage Resource Managers). 存储资源代理如何合理分布复本直接影响数据访问效率. 当前, 数据复本分配方面的研究主要集中在缓存的构建方面, 典型的研究成果如: GASS[7], 该系统为数据存储提供了一个支持二级缓存的框架; VGRID[8], 该系统结合水下地图应用实现了动态分布数据存储; ADS[9]: 该系统实现了数据根据各存储系统的容量自动选择存储位置; HDDA[10], 该算法讨论了如何结合计算特征缓存数据集合; DSA-RM[11], 提出了软件体系结构驱动的动态自适应数据复本管理架构.

上述研究成果主要集中在缓存体系的构建方面, 没有考虑针对应用特征完成复本分配. 在实际的网格应用中, 对数据单元的访问频率决定了数据集访问特性. 作为应用的一个典型特点, 该特性通常可以在数据网格应用运营一段时间后获得, 并且由于应用的稳定性, 该特性将在一段时间内保持稳定. 因此, 定期根据数据单元的访问频率调整数据副本的分配, 提高存储资源代理对数据集的访问效率将提高数据网格的整体性能.

收稿日期: 2005-10-29; 修回日期: 2006-07-15
基金项目: 上海高校网格 e-研究院项目

2 问题描述

数据网格环境由网格节点和连接各节点的网络组成, 其中网格节点包括应用节点(计算节点)和存储节点(存储资源代理). 一个存储资源代理为一组通信费用较低的计算节点提供数据. 计算节点产生和存储原始数据, 存储资源代理存储数据副本.

在涉及大容量数据处理及多用户并发访问的数据网格中, 存储资源代理对数据的访问效率将直接影响到数据网格的整体性能. 存储资源代理的数据访问效率通常由三个因素决定: (1)存储资源代理间的网络性能; (2)存储资源代理间的数据传输量; (3)数据副本的分布. 可以考虑通过合理的数据副本分布策略, 将存储资源代理最常访问的数据副本放置在访问效率最高的节点上. 一种极端的数据副本分布策略就是在所有的存储资源代理上放置全部数据的副本, 但由于存储资源代理存储容量的限制, 不可能这样实现. 我们给出一种填空式动态数据副本分配算法(Calking Dynamic Replication Distribution Algorithm, CDRDA), 以获得存储资源代理数据访问效率的近似最优值. 为简化讨论, 我们假定网格中各存储资源代理节点的计算和 I/O 性能相同.

一个数据网格存储环境的数据副本分配问题可以这样描述. 网格上的全体原始数据集合: $D=\{D_i \mid 1\leqslant i\leqslant d, d$ 为数据单元个数$\}$. 为简化讨论, 考虑将数据单元等分. 存储资源代理集合: $R=\{R_i \mid 1\leqslant i\leqslant n, n$ 为存储资源代理个数$\}$. 网格计算节点非本地数据的访问均通过存储资源代理完成, 存储资源代理对各个数据单元的访问频率用一个访问频率矩阵表示: $\boldsymbol{F}=\begin{pmatrix} f_{11}\cdots f_{1d} \\ \cdots\cdots \\ f_{n1}\cdots f_{nd} \end{pmatrix}$, f_{ij} 表示存储资源代理 R_i 对数据集 D_j 的访问频率. 存储资源代理间的数据单位访问开销可以用一个网络延时加权图表示: $\boldsymbol{C}=\begin{pmatrix} C_{11}\cdots C_{1n} \\ \cdots\cdots \\ C_{n1}\cdots C_{nm} \end{pmatrix}$, C_{ij} 表示存储资源代理 R_i 访问 R_j 上一个数据单元的开销. 各存储资源代理的存储容量可用向量 $\boldsymbol{Q}=(Q_1, Q_2, \cdots, Q_n)$ 表示, 其中, Q_i 为存储资源代理 R_i 的可用存储容量. 数据副本在存储资源代理上的分配可表示为一个聚集: $S=\{S_i \mid 1\leqslant i\leqslant n, n$ 为存储资源代理个数$\}$, S_i 由 D 中的数据单元组成, $\bigcup_{i=1}^{n} S_i = D$. 因此, 数据副本在存储资源代理上的分配问题即是求聚集 S, 使得存储资源代理集合 R 在数据访问频率 F 和网络延时 C 情况下对数据集 D 的平均访问时间最短, 该问题可以形式化描述如下:

问题: 要求一个分配策略 $g: D \rightarrow R$, 满足以下条件:

(1) R_i 是一个存储资源代理节点, 由 g 映射到 R_i 上的数据单元集合 $S_i(R_i)=\{S_i^k \mid S_i^k\in D, 1\leqslant k\leqslant p\}$, p 为数据单元个数. 对任意 S_i, $\sum_{k=1}^{p}\text{Quantity}(S_i^k)\leqslant Q_i$, 其中 Quantity$(x)$ 表示数据集 x 的大小. Quantity$(D)\leqslant\sum_{i=1}^{n} Q_i$. 数据单元到存储资源代理间的映射可用数据映射矩阵表示: $\boldsymbol{M}=\begin{pmatrix} M_{11}\cdots M_{1d} \\ \cdots\cdots \\ M_{n1}\cdots M_{nd} \end{pmatrix}$, $\begin{bmatrix} m_{ij}=1, d_j \text{ 在 } R_i \text{ 上存在} \\ m_{ij}=1/\infty, d_j \text{ 在 } R_i \text{ 上不存在} \end{bmatrix}$;

(2) 对于聚集 $S=\{S_i \mid 1\leqslant i\leqslant n, n$ 为存储资源代理个数$\}$, 图 C, 访问频率矩阵 $\boldsymbol{F}$, 由 g 确定系统总访问开销 $c=\sum_{i=1}^{n}\sum_{j=1}^{d} f_{ij}\times(\min(c_{i1}\times m_{1j}, \cdots, c_{ik}\times m_{kj}, \cdots, c_{in}\times m_{nj}))$ 应达到最小值.

上述问题类似于文献[4], 该文献已经证明 MDAP 是 NP 完全问题, 下面我们给出一种填空式, 多项式近似求解算法.

3 填空式数据副本分配算法

考虑各存储资源代理的容量和网络延时不同, 我们提出了一种填空式数据副本分配算法——CDRDA, 在进行数据副本分配时, 尽量将访问频率高的副本放置在网络传输费用低的存储资源代理上. 下面是 CDRDA 算法的描述:

Step1: 将数据单位按照总访问频率排序, 放于表 A 中. 依次取出表 A 中的数据单元, 计算该数据单元放置在每个存储资源代理上时, 被全部存储资源代理访问的总开销. 选择总开销最小的且存储空间未满的存储资源代理, 将其放在该代理上.

Step2: 计算每个数据单元被未分配该数据单元的存储资源代理访问的总频率. 根据总频率排序将数据单元放在表 A 中. 依次取出表 A 中的数据单元, 计算该数据单元放置在每个存储资源代理上时, 被全部存储资源代理访问的总开销. 选择总开销最小的且存储空间未满的存储资源代理, 将其放在该代理上.

Step3: 重复 Step2 直到所有的存储资源代理的空间已满.

在实际分配策略中, 可能有如下情况需要处理:

(1) 在 Step1 和 Step2 中, 如果存在一个或多个数据单元的总访问频率相同, 考虑到数据访问的局部性, 则优先处理访问频率方差最大的数据单元.

(2) 在 Step1 和 Step2 中, 如果存在一个或多个存储资源代理, 数据单元放置在这些代理上时, 其被全部存储资源代理访问的总开销相同, 则优先选择对该数据访问频率高的存储资源代理.

(3) 在计算全部存储资源代理对一个数据单元的总访问开销时, 如果存在多个数据单元副本, 则按照网络开销最小的计算.

下面给出算法的形式化描述:

```
Procedure CDRDA(D, R, C, Q, F){
  for(each S_i ∈ S) S_i ∈ ∅
  while(R ≠ ∅){
    T = ∅; ∂ = ∅;
    for(each d_i ∈ D){
      t_i = Sum((R_i ∉ R || d_i ∈ S_i) ? 0: f_i; α_i = Variance
((R_i ∉ R || d_i ∈ S_i) ? 0: f_i;//求方差和频率
```

$T = T \cup \{t_i\}$; $\alpha = \alpha \cup \{\alpha_i\}$

}

$D_s = \mathrm{Sort}(D, T, \alpha)$;

for(each $d_i \in D_s$){　　//循环 1

$c_m = \infty$; $p = 0$;

for(each $R_j \in R$){　　//循环 2

$c = \sum_{k=1}^{n} \min(c_j f_{kl}, \bigcup_{m=1, m \neq j}^{n} \{((d_i \in S_m)\ ?\ 1 : \infty) c_m f_{kl}\})$;

$(c \leqslant c_m)\ ?\ \{c = c_m; p = (f_{jd} \leqslant f_{pd}\ ?\ p : j)\}$;

}

$S_p = S_p \cup \{d_i\}$; (Quantity(S_p) < Q_p) ?: $R = R - \{R_p\}$;

}

}//算法结束

上述算法的时间复杂度为 $O(d \times r^2)$, 其中, d 为数据单元数量, r 为存储资源代理的数量. 例如, 网格中存储资源代理集合 $R = \{R_1, R_2, R_3, R_4\}$; 数据单元集合 $D = \{D_1, D_2, D_3, D_4, D_5, D_6, D_7, D_8, D_9, D_{10}\}$, 每个数据单元容量均为 1; 各存储资源代理容量 $\boldsymbol{Q} = (4, 6, 5, 2)$, 代理间网络传输开销

$$\boldsymbol{C} = \begin{pmatrix} 0 & 3 & 2 & 4 \\ 3 & 0 & 5 & 1 \\ 2 & 5 & 0 & 12 \\ 4 & 1 & 12 & 0 \end{pmatrix};$$

数据访问频率

$$\boldsymbol{C} = \begin{pmatrix} 1 & 3 & 9 & 12 & 1 & 0 & 2 & 8 & 6 & 4 \\ 0 & 1 & 22 & 1 & 3 & 4 & 7 & 0 & 0 & 0 \\ 3 & 4 & 4 & 3 & 5 & 3 & 4 & 5 & 3 & 5 \\ 1 & 12 & 3 & 0 & 0 & 0 & 0 & 1 & 15 & 2 \end{pmatrix}.$$

对该实例, 利用 CDRDA 算法, 数据的分配结果为: $S_1 = \{d_4, d_8, d_9, d_{10}\}$, $S_2 = \{d_3, d_4, d_5, d_6, d_7, d_9\}$, $S_3 = \{d_1, d_2, d_5, d_8, d_{10}\}$, $S_4 = \{d_2, d_9\}$, . 系统总访问开销为: $\sum_{i=1}^{4} \mathrm{Consume}(R_i) = 43 + 1 + 55 + 39 = 138$.

4　算法性能分析

对于 NP 完全问题, 我们只能给出近似最优解. 通过对 CDRDA 算法的分析, 可以给出该算法的上界.

该算法的 Step1 可以保证系统中最常访问的数据单元最优先分配存储资源代理上的空间, 且在分配每个数据单元时, 保证整个系统对该数据单元的访问效率最高. Step2 继续按照该模式将数据单元填到还有存储空间的存储资源代理上, 直到所有存储资源代理都被填满. 因此, 相比通过该算法得到的数据副本分布聚集 S, 和通过前 $k = \lfloor (\sum_{i=1}^{n} Q_i) / Quantity(D) \rfloor$ 轮数据副本填空后得到的数据副本分布 S^k, 系统的总访问开销 $\mathrm{Consume}(S) \leqslant \mathrm{Consume}(S^k) \leqslant \mathrm{Consume}(S^u)$. 其中, $u = \min((\min(Q)/\mathrm{Quantity}(d_i)), k)$.

在进行完第 m 轮 ($m \leqslant u$) 数据副本分配时, 系统总访问开销降低的总量: $(\mathrm{Consume}(S) - \mathrm{Consume}(S^k)) \geqslant \sum_{k=1}^{m} \sum_{i=1}^{n} d_p f_{ip}$ (公式 1), 其中 d_p 指存储资源代理 R_i 访问频率第 k 高的数据单元. 使用数学归纳法证明公式 1 如下: (1) $m = 1$ 时, $\forall d_i \in D$, CDRDA 算法的循环 2 保证了该数据副本单元被分配后开销降低总量 $\geqslant d_p f_{ip}$; 故可以得到 $(\mathrm{Consume}(S) - \mathrm{Consume}(S^1)) \geqslant \sum_{k=1}^{1} \sum_{i=1}^{n} d_p f_{ip}$. (2) 假设 $m = g$ 时公式 1 成立, 即 $(\mathrm{Consume}(S) - \mathrm{Consume}(S^g)) \geqslant \sum_{k=1}^{g} \sum_{i=1}^{n} d_p f_{ip}$. (3) 在 $m = g + 1$ 时, $\forall d_i \in D$, CDRDA 算法的循环 2 找到开销最低的分配方式. 这时有两种情况: (1) 该存储资源代理的空间已满; (2) 该存储资源代理仍然可以继续存储数据单元副本. 第 (2) 种情况, 显然可以保证该数据副本单元被分配后开销降低总量 $\geqslant d_p f_{ip}$. 对于第 (1) 种情况, 在循环 2 的上一级循环 1 中或者 $m = g$ 的循环中已经保证了数据副本单元被分配后开销降低总量 $\geqslant d_p f_{ip}$. 因此可以得到 $(\mathrm{Consume}(S) - \mathrm{Consume}(S^{g+1})) \geqslant \sum_{k=1}^{g+1} \sum_{i=1}^{n} d_p f_{ip}$. 公式 1 得证.

综上, CDRDA 算法保证了数据副本分配后, 系统的总体访问开销的上界是每个存储资源代理只存储访问频率高的数据单元副本的系统总体访问开销. 在第 3 节给出的示例中, 可以计算出, 当每个存储资源代理存储访问频率在前 u 名的数据单元副本其他副本随机分配时, 总体访问开销为 180 ⩾138. 从我们的模拟数据看出, CDRDA 算法得到的数据副本分布, 实际上形成了一个多级虚拟存储架构. 数据副本总是可以根据被存储资源代理访问的频率分布在访问开销小的节点上.

此外, 考虑到网格中资源的不稳定性, 各存储资源代理对数据单元的访问频率需定期按照新的资源分布情况进行更新. 此时, 数据更新的开销即算法初始化开销和重新分配数据的开销之和. 初始化算法时间复杂度为 $O(d \times r^2)$, 其中, d 为数据单元数量, r 为存储资源代理的数量. 数据分配可在数据访问过程中完成, 只对数据首次访问的性能有所影响.

5　总结

本文基于一个多存储资源代理的数据网格环境, 提出一种基于数据集访问频率的填空式数据副本分配算法 CDRDA. 通过该算法获得的数据副本分配聚集使得整个系统的数据访问开销达到近似最优解. 该算法适用于多存储资源代理数据网格环境, 计划在上海医学网格[5]中应用.

CDRDA 算法理想化了数据网格环境, 仅考虑了存储资源代理存储性能和网络通信开销的不同, 但是没有考虑存储资源代理节点计算和吞吐能力以及网络通信开销的变化, 也没有考虑新加入数据单元的分配问题, 这些都将是我们进一步需要完善 CDRDA 算法的内容.

参考文献:

[1] A Chervenak, I Foster, C Kesselman, C Salisbury, S Tuecke. The data grid: towards an architecture for the distributed marr agement and analysis of large scientific data sets[J]. Journal of Network and Computer Applications, 2001, 23: 187– 200.

[2] W Hoschek, F J Jaen-Martinez, A Samar, H Stockinger, K Stockinger. Data management in an international data grid project[A]. Proceedings of the First IEEE/ACM International Workshop on Grid Computing[C]. London, UK Springer-Verlag. 2000. 77-90.

[3] C Baru, R Moore, A Rajasekar, M Wan. The SDSC storage resource broker[A]. Proceedings of the 1998 Conference of the IBM Centre for Advanced Studies on Collaborative Research (CASCON'98)[C]. Toronto, Canada. IBM Press. 1998. 5-16.

[4] O Frieder, H T Siegelmann. Multiprocessor document allocation: A genetic algorithm approach[J]. IEEE Transactions on Knowledge and Data Engineering, 1997, 9(4): 640-642.

[5] 陈磊, 顾雷, 李三立. 医学数据网格中数据一致性问题研究及实现[J]. 小型微型计算机系统, 2006, 27(5): 813-817.
Chen Lei, Gu Lei, Li Sanli. Research on data consistency in medical data grid[J]. Mini-Micro Systems 2006, 27(5): 813-817. (in Chinese)

[6] I Foster, C Kesselman. A data grid reference architecture[R]. GriPhyN-2001-12, http://www.griphyn.org, 2001.

[7] J Bester, I Foster, C Kesselman, J Tedesco, S Tuecke. GASS: a data movement and access service for wide area computing systems[A]. Proceedings of the Sixth Workshop on I/O in Parallel and Distributed Systems[C]. Atlanta, Georgia, United States: ACM Press. 1999. 78-88.

[8] C A Steed. J E Braud, K A Koehler. VGRID: a generic, dynamic HDF5 storage model for geo referenced grid data[A]. Proc of MTS/IEEE OCEANS2002[C]. Biloxi, Mississippi, USA: Marine Technology Society. 2002. 900-907.

[9] L C Hu. S X Sun. ADS: a handle-based storage architecture under grid computing[A]. Proc of the IEEE 18th Annual Workshop on Computer Communications (CCW 2003)[C]. Dana Point, California, USA: IEEE Communications Society, 2003. 187-193.

[10] 王新军, 洪晓光, 王海洋, 孟祥旭. 网格计算中一种启发式数据分配算法的讨论[J]. 电子学报, 2004, 32(4): 648-650.
Wang Xin-jun, Hong Xiao-guang, Wang Hai-yang, Meng Xiang-xu. Discussion on heuristic algorithm of data distribution in grid computing[J]. ACTA Electronica Sinica 2004, 32(4): 648-650. (in Chinese)

[11] 陈磊, 李三立. 网格数据复本管理的动态自适应软件体系结构[J]. 软件学报, 2006, 17(6): 1436-1447.
Chen Lei, Li Sanli. Dynamic self-adapting software architecture for replica management in grids[J]. Journal of Software, 2006, 17(6): 1436-1447. (in Chinese)

作者简介:

陈 磊 男, 1976 年生于河南信阳, 工程师, 博士研究生, 主要研究方向为网格计算技术和高性能计算等. E-mail: cL03@mails.tsinghua.edu.cn

李三立 男, 1935 年生于上海, 中国工程院院士, 博士生导师, 研究方向为网格计算技术和高性能计算技术等.