第36卷 第20期 计 算 机 工 程 2010年10月
Vol.36 No.20 Computer Engineering October 2010
·网络与通信· 文章编号：1000—3428(2010)20—0102—04 文献标识码：A 中图分类号：TP301.6

HPMR系统KV路由算法设计

郑启龙[1,2]，王向前[1,2]，王 昊[1,2]
(1. 中国科学技术大学计算机科学技术学院，合肥 230027；2. 安徽省高性能计算重点实验室，合肥 230026)

摘 要：提出一种针对HPMR系统的KV路由算法。HPMR系统是MapReduce模型的一个实现，改进了MapReduce模型以适应高性能计算的需求。HPMR的KV路由算法生成KV路由表，所有数据的收发动作将依据该路由表进行。KV路由算法产生的KV路由表直接决定HPMR系统在通信阶段的时间开销。实验结果表明，该KV路由算法产生的KV路由表可以提高HPMR系统的通信性能。

关键词：MapReduce模型；高性能计算；DSS-KV路由算法

Design on KV Routing Algorithm for HPMR System

ZHENG Qi-long[1,2], WANG Xiang-qian[1,2], WANG Hao[1,2]
(1. School of Computer Science and Technology, University of Science and Technology of China, Hefei 230027, China;
2. Anhui Provincial Key Laboratory of High Performance Computing, Hefei 230026, China)

【Abstract】 This paper presents a KV Routing Algorithm for High-Performance MapReduce(HPMR) system. HPMR system is an implementation of the MapReduce(MR) model, which accommodates the MR model to the requirements of high-performance computing. The KV routing algorithm is used to create KV routing table by which all the send or receive actions on data are done. The performance of the KV routing table generated by KV routing algorithm directly determines the time cost of the communication phase of HPMR system. Experimental results show that KV routing table created by the KV routing algorithm can improve the communication performance of HPMR system.

【Key words】 MapReduce(MR) model; high-performance computing; DSS-KV routing algorithm

1 概述

MapReduce[1-3]是由Google于2004年提出的并行编程模型，它被用在处理和产生海量数据集的实现中。MapReduce(MR)模型的优点有：简单易学，编写串行程序自动并行运行，高度抽象性等。

MR模型具有非常广泛的适用性。MR过程代表了“数据分组-数据收集-数据处理”一整套过程，任何并行程序都可以分解成一个或多个MR过程。

HPMR(High-Performance MapReduce)[4-5]是本实验室为了推广MR模型而开发的面向高性能计算的MapReduce系统，支持大规模计算的任务分配和自动并行。HPMR系统包括4个功能模块：数据管理模块，任务管理模块，通信管理模块，故障恢复模块。

MR模型的高度抽象性为程序员提供了方便，它把原本需要程序员编写的复杂的消息通信过程透明地屏蔽掉。程序员只需要编写用KV表达的高层通信语义，而不必去负责底层的繁琐的消息通信过程，这无疑减轻了程序员的负担。从KV表达的高层抽象通信语义转化为底层消息通信语义，则由HPMR系统的通信管理模块来负责。

HPMR系统的通信管理模块是支持MR模型的核心模块：因为正是它支持上层的KV通信语义到底层的消息通信语义的转换。这个转换过程称为KV路由过程。KV路由过程主要包括master上KV路由算法的执行。KV路由算法产生的KV路由表直接决定HPMR系统在通信阶段的时间开销，从而影响HPMR系统的执行效率。KV路由算法的设计对于HPMR系统来说，是至关重要的。

2 HPMR的通信管理模块

2.1 HPMR的通信管理模块的功能

HPMR的通信管理模块主要负责执行KV路由过程和KV传输。下面是通信管理中涉及的几个主要概念：

KV摘要：是worker从其拥有的<key1, key2, va-lue>中提取的基本信息。

KV路由过程：是指在Map阶段后，所有worker把自己的KV摘要发送给master，master调用KV路由算法生成KV路由表，然后把KV路由表回送给各个worker这一过程。

KV路由算法：该算法用来生成KV路由表。KV路由表是收发动作序列(以下简称RS序列)，所有worker在获得KV路由表后，遵照其中的RS序列，执行相应的收发<key1, key2, value>的动作。

KV路由表：记录每个<key1, key2, value>应该从哪个worker发送给哪个worker。

KV传输：是指worker根据KV路由表把自己的<key1, key2, value>发送给其他worker的过程。

2.2 HPMR通信阶段的工作流程

worker执行一轮MR过程分为3个阶段：Map阶段，通信阶段，Reduce阶段。通信阶段的工作流程如图1所示，包

基金项目：国家自然科学基金资助重点项目(60533020)；安徽省自然科学基金资助项目(090412068)
作者简介：郑启龙(1969－)，男，副教授，主研方向：并行与分布式计算；王向前、王 昊，硕士研究生
收稿日期：2010-04-28 **E-mail：**forward@mail.ustc.edu.cn

括如下 3 个阶段：

(1)worker 把 KV 摘要发送给 master。

(2)master 执行 KV 路由算法，产生 KV 路由表，然后把 KV 路由表发送给每个 worker。

(3)每个 worker 依照 KV 路由表把各自的<key1, key2, value>发送给其他 worker。

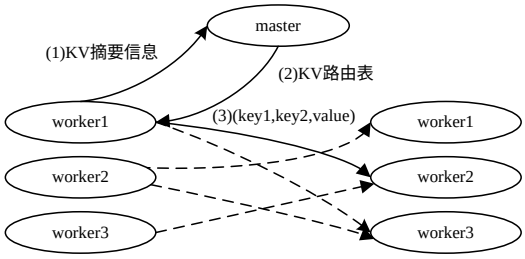


图 1　HPMR 通信阶段的工作流程

3　KV 路由算法的目标

HPMR 的 KV 路由算法有如下 3 点目标：

(1)避免通信死锁

HPMR 底层使用的 Boost MPI[6]还不支持安全的多线程通信，所以 HPMR 的每个 worker 只能使用一个线程执行通信任务，即在同一时刻只能执行一个发送(send)或接收(receive)动作。在这种情况下，不合理的 RS 序列可能会导致通信死锁。

HPMR 系统保证通信过程不会发生死锁。因为 KV 路由表是收发动作的线性顺序表，所有 worker 都获得相同的 KV 路由表，然后依照表中记录的顺序依次执行与自己相关的发送或接收动作，所有 worker 都使用阻塞式通信方式。当 worker1 向 worker2 发送数据的时候，worker2 必然要执行从 worker1 接收数据的动作，因此，不会出现上面的通信死锁现象。

(2)降低通信量

尽量降低通信量是缩短系统通信时间、降低通信开销的有效办法。KV 路由算法的目标之一就是降低系统通信量。方法是尽量使 size 大的 value“原地不动”，只“移动”那些 size 相对小的 value。

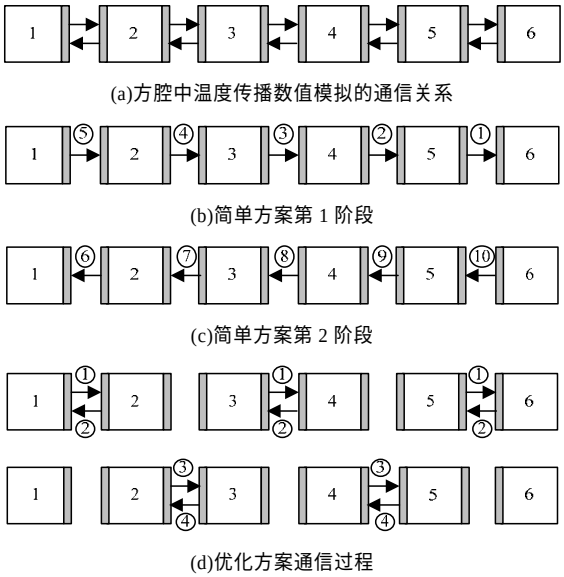
(3)减少通信阻塞

通信阻塞现象的产生主要是由于一个 worker 无法同时执行多个发送或接收动作而造成的。例如，当 worker2 和 worker3 同时向 worker1 发送数据时，worker1 只能一个一个地接收，如果 worker1 先接收 worker2 的数据，则 worker3 必然先处于阻塞状态，直到 worker1 可以接收它的数据为止。虽然表面上看 worker2 和 worker3 在同时发送数据，但实际上两者是串行的。

KV 路由算法决定了产生怎样的 RS 序列，好的 RS 序列可以减少通信阻塞。图 2 说明了不同的 RS 序列可产生高低不同的通信效率。图 2 以方腔中温度传播的数值模拟[6]的通信模型为例，如图 2(a)所示，6 个 worker 各有 1 块数据(编号 1~编号 6)，在通信阶段时，邻近的数据块彼此交换边界信息。

一种简单易行的 RS 序列产生的收发动作如下：

每个 worker 先向它的右侧 worker 发送数据，然后接收左侧 worker 的数据，如图 2(b)所示。动作顺序如箭头上标号所示，因为存在通信阻塞，所以需要 5 个通信步才能完成。接下来，每个 worker 再向它的左侧 worker 发送数据，然后接收右侧 worker 的数据，如图 2(c)所示。同样也需要 5 个通信步。所以，整个通信过程共需要 10 个通信步。



(a)方腔中温度传播数值模拟的通信关系

(b)简单方案第 1 阶段

(c)简单方案第 2 阶段

(d)优化方案通信过程

图 2　收发动作序列的优化举例

这种通信方式的优点是简单，容易实现；缺点是由于通信阻塞过多，造成通信耗时过大。如果用这种方式处理 n 块数据，则一轮通信需要 $2(n-1)$个通信步。

图 2(d)展示了一种优化的 RS 序列是如何工作的：把 worker 分成两两一组，两两互发，这样大大减少了通信阻塞。整个通信过程只需要 4 个通信步，而且这种通信方式的通信耗时不会随着数据块数量的增加而增加，始终都是 4 个通信步。

由此可见，RS 序列的好坏直接影响通信效率的高低，好的 RS 序列可以提高 worker 之间通信的并发度，减少通信阻塞。HPMR 的 KV 路由算法虽然无法彻底消除通信阻塞，但它采用了一种简单易行的方法——对角线跳跃选择法(Diagonal-Skip Selection, DSS)，可以有效地减少通信阻塞现象的发生，且算法的时间复杂度也不高。

4　DSS-KV 路由算法

4.1　算法的输入、输出数据

master 根据 KV 摘要信息，执行 DSS-KV 路由算法，生成 worker 所需的 KV 路由表。故 DSS-KV 算法的输入输出如下：

(1)算法的输入数据

KV 路由算法的输入数据是各个 worker 的 KV 摘要，其中每一项的结构如下：

worker	key1	value_count	value_size

worker 是指拥有本摘要信息的所有者编号，value_count 是相同 key1 的 value 的个数，value_size 是相同 key1 的 value 的 size 大小之和。

(2)算法的输出数据

KV 路由算法的输出数据是 KV 路由表，其中每一项的结构如下：

sender	receiver	key1_list

sender 是发送方 worker 的编号，receiver 是接收方 worker 的编号，key1_list 是 sender 要发送给 receiver 的 key1 的列表。在 KV 传输过程中，sender 需要把与 key1_list 中 key1 相关的<key1, key2, value>打包，一次性发给 receiver。

4.2　算法的步骤

DSS-KV 路由算法的主要步骤如下：

步骤 1 确定每个 key1 的 owner

相同 key1 的所有<key1, key2, value>在通信阶段要“汇聚”到唯一一个 worker 手中，该 worker 就是该 key1 的拥有者(owner)。确定 key1 的 owner 的过程如图 3 所示。

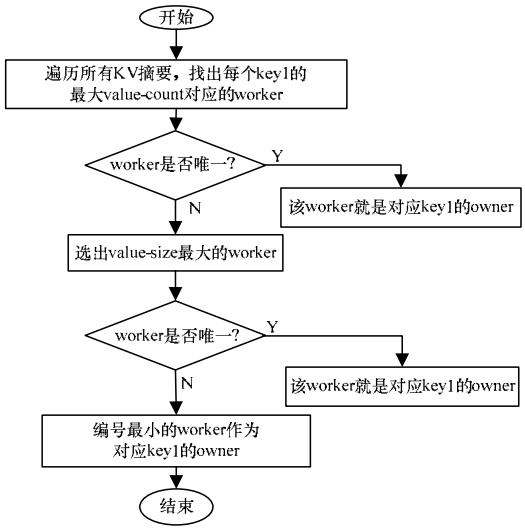


图 3　确定 key1 的 owner 的过程

步骤 2 确定 KV 路由表中的每一项

如果 workeri 是某 key1 的 owner，则 workeri 作为接收方，所有其他产生该 key1 的 worker 分别作为发送方，并把该 key1 添加到每个发送方的 key1_list 中。

步骤 3 确定 KV 路由表中收发动作的顺序，即 RS 序列

KV 路由算法在产生 RS 序列时，通过调整收发动作的顺序关系来减少通信阻塞现象。基本思想是让 KV 路由表上临近的(sender, receiver, value_list)尽量避免出现 senders 之间、receivers 之间或 sender 与 receiver 之间名字相同的现象，有如下 3 种情况：

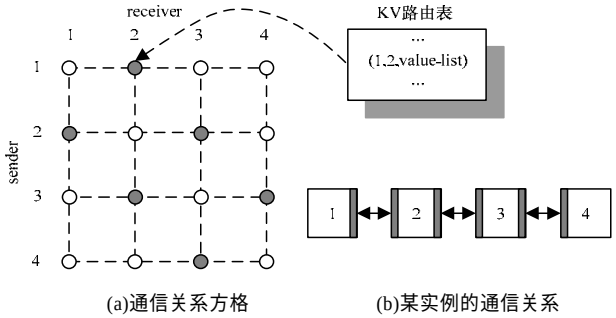
(1)(1, 2, value-list)、(1, 3, value-list)

(2)(1, 3, value-list)、(2, 3, value-list)

(3)(1, 2, value-list)、(2, 3, value-list)

为此，KV 路由算法设计了对角线跳跃选择法(Diagonal-Skip Selection, DSS)来对收发动作进行排序。

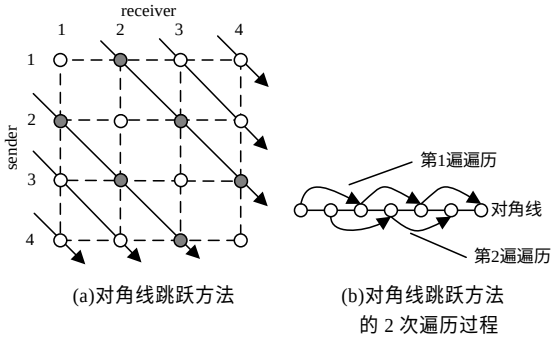
DSS 方法基于 RS 方格，格式如图 4(a)所示。RS 方格的行和列都是 worker 的编号，方格上每一个点对应一个编号(*i*,*j*)。如果 KV 路由表中存在表项(*i*,*j*,value_list)，说明存在 *i* 和 *j* 的收发动作，(*i*,*j*)点指向该表项，(*i*,*j*)点称为有效点。否则(*i*,*j*)点为空，称为无效点。图 4(a)是图 4(b)的 RS 方格。



(a)通信关系方格　　(b)某实例的通信关系

图 4　4×4 RS 方格

所谓 DSS 方法，是对 RS 方格的每条对角线从右向左依次进行跳跃遍历，如图 5(a)所示，箭头方向标明遍历的顺序。由于采用跳跃遍历方法，因此需要 2 遍才能完整遍历一条对角线，过程如图 5(b)所示。



(a)对角线跳跃方法　　(b)对角线跳跃方法的 2 次遍历过程

图 5　对角线跳跃选择法

图 5 实例产生的遍历顺序如下：

(1,4) (1,3) (2,4) (1,2) (3,4) (2,3) (2,1) (4,3) (3,2) (3,1) (4,2) (4,1)

如果在遍历过程中发现有效点，则把与之关联的表项添加到 KV 路由表的表尾。图 4 实例生成的 KV 路由表如表 1 所示，表项中的数据依次为 sender、receiver、通信数据列表。

表 1　KV 路由表实例

KV 路由表项
(1, 2, value_list)
(3, 4, value_list)
(2, 3, value_list)
(2, 1, value_list)
(4, 3, value_list)
(3, 2, value_list)

分析 DSS 方法的思想可知，该方法可以有效地减少 KV 路由表中相邻表项中 sender/receiver 相同的现象，从而减少通信阻塞，提高通信并行性。但 DSS 方法还无法完全避免通信阻塞。

4.3　KV 路由算法的时间复杂度

KV 路由算法由 3 个步骤组成。步骤 1 的时间复杂度是遍历所有 key1 的时间 $O(K)$，K 是所有 worker 产生的 key1 的个数。步骤 2 的时间复杂度也是 $O(K)$。步骤 3 的时间复杂度是遍历 RS 方格的时间 $O(N^2)$，N 是 worker 的个数，也等于参与计算的节点的个数。所以，KV 路由算法的时间复杂度是 $O(2K+N^2)$。

在用 MapReduce 程序解决科学计算问题时，往往 worker 产生的 key1 的个数等于初始输入数据块个数的常数倍。设程序初始输入数据块的个数是 B，则

$K=cB$　　(c 是一个常数)

所以，在解决科学计算问题时，KV 路由算法的时间复杂度是 $O(cB+N^2)$，c 是一个常数。可见当参与计算的节点个数固定时，KV 路由算法的时间复杂度取决于初始输入数据块的个数，个数越多算法的运行时间越长。

5　KV 路由算法性能

5.1　测试平台

HPMR 使用国家高性能计算中心(合肥)的曙光 TC4000A 作为硬件测试平台，这台高性能计算机由 42 个计算节点、1 个登录节点和 1 个控制节点组成，它的理论峰值达到 26 752 亿 flops。每个计算节点的配置如下：

CPU：双 AMD Opteron 2347(4 核，主频 1.9 GHz)

内存：4 GB

硬盘：160 GB SATA

计算网络：1 000 Mb/s 快速以太网

操作系统：Red Hat AS 4.6 for Linux

MPI 系统：MPICH2-1.0.8

5.2 测试

以方腔中温度传播的数值模拟为例，比较未经优化的 KV 路由表(简称 UOPTI 路由表，产生方法如图 2(b)和图 2(c)所示)与 KV 路由算法产生的经过优化的 KV 路由表(简称 DSS 路由表)对 KV 传输时间的影响。测试结果如图 6 和图 7 所示。

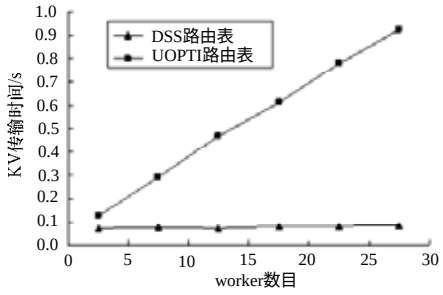


图 6　KV 传输时间随 worker 数量的变化

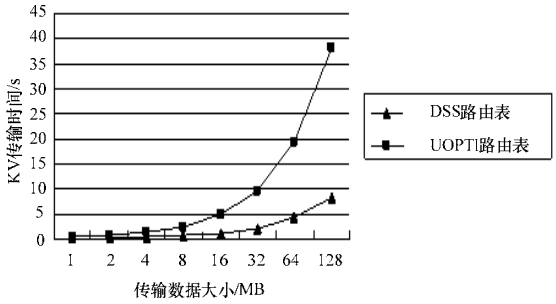


图 7　KV 传输时间随传输数据量的变化

从图 6 和图 7 可以看出，KV 路由算法产生的 DSS 路由表相对 UOPTI 路由表可以有效地减少通信阻塞，缩短 KV 传输时间。这点在图 6 中反映得更加明显，由 UOPTI 路由表指导的 KV 传输的时间随着 worker 数目的增加呈线性递增趋势，而 DSS 路由表指导的 KV 传输的时间变化微小。其中的原因如第 3 节所述，UOPTI 路由表指导下的 KV 传输在每轮需要 2(n-1)个通信步(n 为 worker 数目)，而 DSS 路由表指导下的 KV 传输在每轮所需的通信步基本上不会随着 worker 数目的增加发生大的变化。

6　结束语

基于 MR 模型的 DSS-KV 路由算法，在保证从高层 KV 通信语义转化为底层消息通信语义的同时，降低了 KV 传输时间，可见，DSS-KV 路由算法对 HPMR 系统通信效率的提高是显著的。

参考文献

[1] Dean J, Ghemawat S. MapReduce: Simplified Data Processing on Large Clusters[C]//Proc. of the 6th Symposium on Operating Systems Design and Implementation. San Francisco, USA: ACM Press, 2004: 137-150.

[2] Lämmel R. Google's MapReduce Programming Model-Revisited[J]. Science of Computer Programming Journal, 2007, 68(3): 208-237.

[3] Ranger C. Evaluating MapReduce for Multi-core and Multiprocessor Systems[C]//Proc. of the 13th International Symposium on High Performance Computer Architecture. Phoenix, Arizona, USA: IEEE Press, 2007: 13-24.

[4] 郑启龙, 王　昊, 吴晓伟, 等. HPMR: 多核集群上的高性能计算支撑平台[J]. 微电子学与计算机, 2008, 25(9): 21-23.

[5] 郑启龙, 吴晓伟, 房　明, 等. HPMR 在并行矩阵计算中的应用[J]. 计算机工程, 2010, 36(8): 49-51.

[6] Boost Org. Boost MPI Home Page[EB/OL]. (2009-09-12). http://www.boost.org/doc/libs/1_38_0/doc/html/mpi.html.

编辑　任吉慧

(上接第 92 页)

扩展性。(2)初始负载分布状态互不相同的 3 条线几乎重合。因此，对于普通的贪心线性推移平衡算法，初始负载分布状态对平衡时间性能几乎没有影响。

4　结束语

本文针对具有环或线性阵列结构(或子结构)的网络提出贪心线性推移平衡算法。对常用的不同静态互连网络即线性阵列(环)、二维网状网结构进行了详细的算法设计及性能分析。分析与实验表明，对一般的互连网络结构，该算法的平衡步数都不高于处理器节点的数目 n。贪心线性推移平衡算法是一种迁移量较小、时间开销较少的算法，较适用于随机任务负载分布的负载平衡情况。

参考文献

[1] Imani N, Sarbazi-Azad H, Akl S G. Perfect Load Balancing on the Star Interconnection Network[J]. Journal of Supercomputing, 2007, 41(3): 269-286.

[2] 王　俊, 郑　笛, 吴泉源. 分布式高可扩展负载平衡中间件技术研究[J]. 计算机工程, 2008, 34(5): 39-41.

[3] Elsasser R, Monien B, Preis R. Diffusion Schemes for Load Balancing on Heterogeneous Networks[J]. Theory of Computing Systems, 2002, 35(3): 305-320.

[4] Rotaru T, Nageli H H. Dynamic Load Balancing by Diffusion in Heterogeneous Systems[J]. Journal of Parallel and Distributed Computing, 2004, 64(4): 481-497.

[5] Arndt H. Load Balancing: Dimension Exchange on Product Graphs[C]//Proc. of the 18th International Parallel and Distributed Processing Symposium. New Mexico, USA: IEEE Press, 2004.

[6] Chau Siu-cheung. Load Balancing Between Heterogeneous Computing Clusters[C]//Proc. of International Workshop on Grid and Cooperative Computing. Wuhan, China: [s. n.], 2003.

编辑　张正兴