一种大数据放置方法

张桂刚

（清华大学信息技术研究院 北京 100084）（首都经济贸易大学 北京 100070）

摘 要 数据密集型应用越来越多，如何将大数据在数据中心实现有效放置变得日益重要。研究了大数据的放置模型。影响大数据放置的因素主要有：能耗、异构节点的服务能力及具有关联计算的数据集。基于这3个因素设计了一种节能、充分考虑异构节点服务能力及提升MapReduce处理Join连接的效率的大数据放置模型。该模型将有效实现大数据的有效放置管理，同时也为未来软件定制数据中心奠定了基础。

关键词 大数据，数据放置，节能，异构节点，关联计算

中图法分类号 TP391.41 **文献标识码** A

A kind of Big Data Placement Method

ZHANG Gui-gang

(Research Institute of Information Technology, Tsinghua University, Beijing 100084, China)

(Capital University of Economics and Business, Beijing 100070, China)

Abstract More and more data-intensive applications have come into being. It is becoming more and more important for the big data' efficient placement in the data center. This paper proposed a kind of big data placement model. The major factors that influce the big data placement have the following three points: energy consumption, sevice capability of heterogeneous node and the data sets which have associated computing. Based on these three factors, our big data placement model considers the energy-saving, service capability of heterogeneous node and the complex Join query mapreduce computing so on. This model can implement the big data's efficient placement management efficiently. At the same time, it will estabilish a foundation for software customized data center in the future.

Keywords Big data, Data placement, Energy-saving, Heterogeneous nodes, Associated computing

1 引言

随着云计算技术的发展，各种数据密集型应用应运而生，数据中心的数据管理[1]变得越来越重要。如何让数据中心既能够充分考虑能耗[2]，同时又能够考虑到异构节点的服务能力及其处理复杂Join连接查询的实时性问题是一个非常紧迫的问题。

现有数据中心的数据放置策略仍然处在十分粗放的阶段，从而造成了大量的投资浪费和能源浪费，主要体现在：1)盲目采购大量的机器搭建数据中心，而实际上只存储了极少的数据和进行极少量的数据计算，导致了大量的投资浪费，同时将大量的机器投入到无价值的运转中，造成了极大的能源浪费。2)盲目地将所有利旧的机器全部放入到数据中心的集群中，而实际上只存储了极少的数据和进行极少量的数据计算，导致了大量能源浪费(陈旧的机器能耗更大)。3)由于数据放置方法的盲目性，使得大部分的机器进行了极少的计算，但是不得不时刻处在高耗能状态而不是休眠运行状态等节能运行状态 。

现有数据放置策略极少考虑数据节点的异构性[3,4]问题，从而导致数据处理效率低下，主要体现在：1)数据中心的数据节点，尤其是来自利旧的数据节点类型各异，而不同的数据节点的服务能力是大不相同的。如内存大的、CPU频率高的、多核的及存储容量大的数据节点明显比内存小的、CPU频率低的、单核的及存储容量小的数据节点服务能力要强。而Hadoop这种计算框架默认所有数据节点都具有同样的服务能力，数据主要按照平均主义的原则进行放置。如果按照Hadoop默认的数据放置方法进行数据放置，服务能力强的数据节点很快完成了分配的计算任务，而服务能力差的很慢才能完成任务。造成大量的等待或者需要将服务能力差的数据节点的数据迁移到服务能力强的数据节点，让服务能力强的节点来帮助完成计算。这样的数据迁移会给网络带来巨大负载，如属于MapReduce的计算的数据迁移会给Shuffle阶段带来巨大的I/O负载和网络负载。2)对于新购置的数据节点也和上述的一样，新购置的数据节点，尤其是不同批次采购的数据节点其服务能力可能大不一样。如果不采取智能的数据放置方法，同样会导致整个集群的处理效率十分低下。

现有的数据放置策略[5,6]在考虑复杂的Join连接查询方面有了一定的研究，主要的数据放置策略有Hadoop默认的放置方案、CoHadoop[7]、Hadoop++[8]以及CHMJ[9,10]。Hadoop默认的大数据放置策略就是实现最大程度上的负载均衡，将数据块平均地分配。一旦需要执行，MapReduce经常需要跨机器甚至机架进行数据远程传输，在Shuffle阶段浪

到稿日期：2013-08-19 返修日期：2013-12-30 本文受高等学校博士学科点专项科研基金课题(20100002110082)资助。

张桂刚(1978—)，男，博士后，副教授，CCF高级会员，主要研究方向为大数据关键技术、云计算技术研究，E-mail：zhangguigang@163.com。

费很多的 I/O 时间。针对静态数据硬编码的数据放置策略，为了提高数据 Join 效率，将数据进行处理，增加索引信息。典型的有 VLDB2010 年的论文 Hadoop＋＋. 数据本地化的聚合的数据放置策略。典型代表有 VLDB2011 年 IBM 的论文 CoHadoop 及中科院和腾迅公司合作的论文[9,10]里提及的数据 hash 聚合机制及基于它的 CHMJ 连接(Join)计算方法。

2 大数据放置模型体系结构

图 1 展示了大数据放置模型的体系架构。

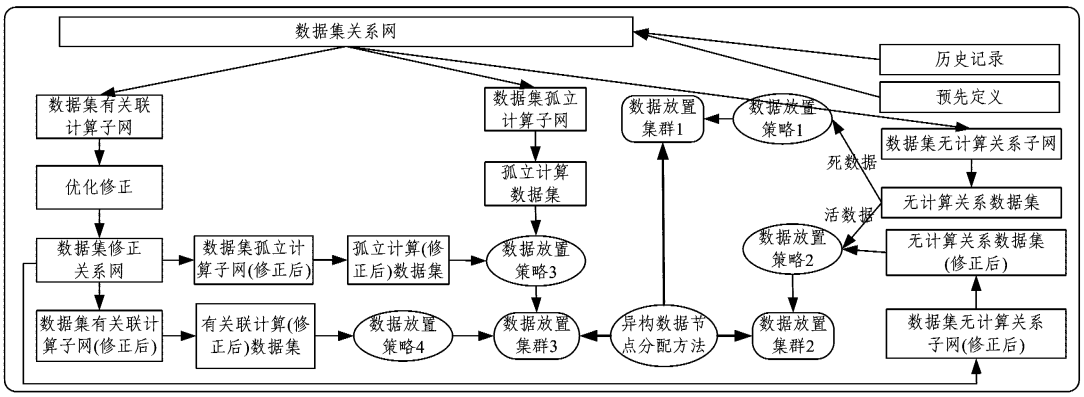

图 1　大数据放置模型的体系架构

该方法的基本实现原理为：根据数据集的历史处理记录或者根据预先的定义得到数据关系网，根据数据集关系网，可以得到数据集无计算关系子网、数据集孤立计算子网及数据集有关联计算子网 3 种。通过数据集无计算关系子网得到相应的无计算关系数据集，对于无计算关系数据集，需要进行一个判断，如果该数据集属于静态数据集(数据不会再改变)，则对它们采用数据放置策略 1，按照数据放置策略 1 的方法将它们放置到数据放置集群 1 中；如果该数据集属于动态数据集(数据会不断增加)，则对它们采用数据放置策略 2，按照数据放置策略 2 的方法将它们放置到数据放置集群 2 中。通过数据集孤立计算子网(该数据集只发生针对自身单个数据集的计算)得到的孤立计算数据集，按照数据放置策略 3 的方法将它们放置到数据放置集群 3 中。对于数据集有关联计算子网，需要进行相应的优化修正得到数据集修正关系网。根据数据集修正关系网，可以得到数据集无计算关系子网(修正后)、数据集孤立计算子网(修正后)及数据集有关联计算子网(修正后) 3 种。通过数据集无计算关系子网(修正后)得到相应的无计算关系数据集(修正后)，对于无计算关系数据集(修正后)，采用数据放置策略 2，按照数据放置策略 2 的方法将它们放置到数据放置集群 2 中。通过数据集孤立计算子网(修正后)得到的孤立计算(修正后)数据集，按照数据放置策略 3 的方法将它们放置到数据放置集群 3 中。通过数据集有关联计算子网(修正后)得到的有关联计算(修正后)数据集，按照数据放置策略 4 的方法将它们放置到数据放置集群 3 中。其中数据节点分配方法将决定数据放置集群 1、数据放置集群 2 及数据放置集群 3 的具体分配实施。

案例 1 展示了一个具体的数据中心云数据分配方法的实施例。

【案例 1】 数据中心云数据分配方法实施例

定义 1(数据集无计算关系子网)　指两个或者两个以上的数据集之间由于没有计算关系(如连接操作等)且这些数据集本身也没有计算关系(如针对数据集自身的查询计算)的数据集所组成的关系子网。

定义 2(数据集孤立计算子网)　指那些与其他数据集没有计算关系(如连接操作等)，但是自身有计算关系(如针对数据集自身的查询计算)的数据集所组成的关系子网。

定义 3(数据集有关联计算子网)　指那些与其他数据集有计算关系(如连接操作等)的数据集所组成的关系子网。

第 1 步：形成数据集关系网。根据数据集的历史处理记录或者根据预先的定义得到数据关系网。图 2 为一个具有 n 个数据集的数据集关系网。

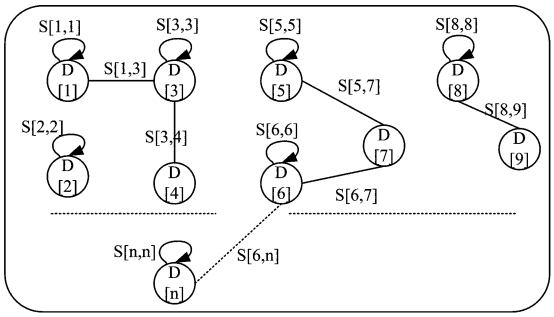

图 2　数据集关系图

图 2 的主要说明如下：

(1)其中在云计算中共使用了 n 个数据集合，分别为：$D[1]$，$D[2]$，$D[3]$，$D[4]$，$D[5]$，$D[6]$，$D[7]$，$D[8]$，$D[9]$，…，$D[n]$。

(2)$S[i,j]$表示数据集 $D[i]$与 $D[j]$之间的计算关联度，主要分为如下几种情况：

a)如果 $i=j$，并且 $S[i,j]=0$。$i=j$ 表明为同一个数据集。如果 $S[i,j]=0$，它表明针对该数据集自身没有任何计算操作(如查询等)。

b)如果 $i=j$，并且 $S[i,j]>0$。$i=j$ 表明为同一个数据集。如果 $S[i,j]>0$，它表明了针对该数据集自身有计算操作(如针对该单个数据集的查询等)。

c)如果 $i!=j$，并且 $S[i,j]=0$。$I!=j$ 表明涉及到两个不同的数据集。如果 $S[i,j]=0$，它表明这两个不同的数据集之间没有任何计算操作(如连接操作、联合操作及其笛卡尔积等)。

d)如果 $i!=j$，并且 $S[i,j]>0$。$I!=j$ 表明涉及到两个

不同的数据集。如果 $S[i,j]>0$，它表明这两个不同的数据集之间有计算操作(如连接操作、联合操作及其笛卡尔积等)。

(3)根据(2)及历史计算关系或者预先定义，得到相应的含数值的数据集历史计算关系图，如图 3 所示。其中：

$S[1,1]=200$；$S[1,3]=200$；$S[2,2]=3$；$S[3,3]=50$；$S[3,4]=2$；$S[5,5]=100$；$S[6,6]=80$；$S[5,7]=78$；$S[6,7]=88$；$S[8,8]=60$；$S[9,9]=0$；$S[6,n]=1$；$S[n,n]=120$。

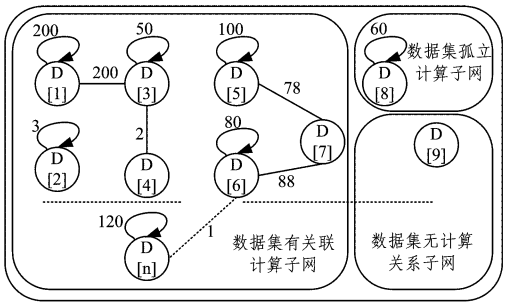

图 3　含数值的数据集关系网

(4)从图 3 可以得到图 2 中所提及的 3 个子网：数据集有关联计算子网、数据集孤立计算子网及数据集无计算关系子网。

(5)从图 3 可以得到图 2 中所提及的两个子网分别所对应的数据集：孤立计算数据集{$D(8)$}及无计算关系数据集{$D[9]$}。

第 2 步：形成数据集修正关系网。Hadoop 自身的数据放置策略的最大优势是通过分区函数让所有的数据块能够实现自由流动，从而达到一种较好的负载均衡。本文将对来自第 1 步的数据集有关联计算子网进行相应的修正，让一部分数据集的数据放置遵循 Hadoop 本身的数据放置策略，从而实现较好的负载均衡。其中最关键的是需要设定相应的修正因子(该修正因子可以由云数据中心管理人员自行编程设定)，然后对数据集有关联计算子网进行相应的修正得到一个数据集修正关系网。具体子步骤如下：

(1)获取来自第 1 步的数据集有关联计算子网，如图 4 所示。

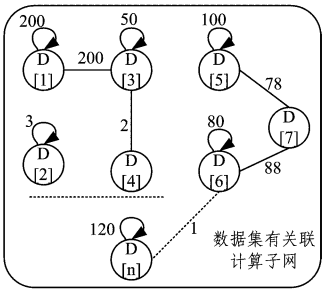

图 4　数据集有关联子网

(2)优化修正。进行优化修正的一个重要因素，是需要设置一个优化修正因子。优化修正因子的设置可以由数据中心管理人员设定(编程实现)。假设数据中心管理人员设定的修正因子为 5，即计算关系小于或者等于 5 的计算因子全部去掉，而保留那些计算关系大于 5 的计算因子。图 4 经过修正因子的修正之后得到的数据集修正关系网如图 5 所示。

(3)从图 5 可以得到图 2 中所提及的 3 个子网：数据集有关联计算子网(修正后)、数据集孤立计算子网(修正后)及数据集无计算关系子网(修正后)。

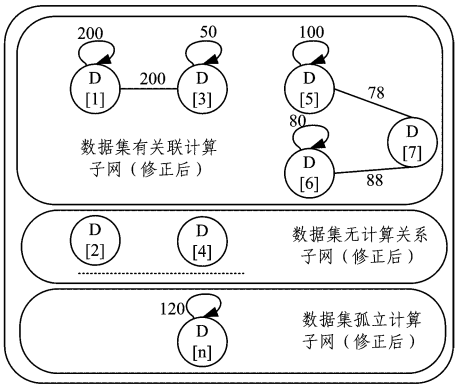

图 5　数据集修正关系网

(4)从图 5 可以得到图 2 中所提及的 3 个子网：数据集有关联计算子网(修正后)、数据集孤立计算子网(修正后)及数据集无计算关系子网(修正后)，分别所对应的数据集：有关联计算(修正后)数据集{$D[1]$，$D[3]$，$D[5]$，$D[6]$，$D[7]$}、孤立计算(修正后)数据集{$D[n]$}及无计算关系数据集(修正后){$D[2]$，$D[4]$}。

第 3 步实施数据集放置。主要根据第 1 步和第 2 步的结果，实施具体的数据集的放置，主要包含如下几个小步骤：

(1)对于第 1 步得到的无计算关系数据集的数据放置。对这部分数据集需要进行一个判断。

a)如果无计算关系数据集属于死数据(永远不会再被使用的数据，仅仅作为档案存储)，则将这部分数据按照数据放置策略 1 进行数据放置，将它们放入到数据放置集群 1 中。

b)如果无计算关系数据集属于活数据(该部分数据会继续增加，如来自云数据库表的数据，不断会有新数据增加)，则将这部分数据按照数据放置策略 2 进行数据放置，将它们放入到数据放置集群 2 中。

(2)对于第 1 步得到的孤立计算数据集的数据放置。对于孤立计算数据集，将这部分数据按照数据放置策略 3 进行数据放置，将它们放入到数据放置集群 3 中。

(3)对于第 2 步得到的无计算关系数据集(修正后)的数据放置。对于无计算关系数据集(修正后)，将这部分数据按照数据放置策略 2 进行数据放置，将它们放入到数据放置集群 2 中。

(4)对于第 2 步得到的孤立计算(修正后)数据集的数据放置。对于孤立计算(修正后)数据集，将这部分数据按照数据放置策略 3 进行数据放置，将它们放入到数据放置集群 3 中。

(5)对于第 2 步得到的有关联计算(修正后)数据集的数据放置。对于有关联计算(修正后)数据集，将这部分数据按照数据放置策略 4 进行数据放置，将它们放入到数据放置集群 3 中。

3　数据集群划分

对于所有的异构节点(含利旧的数据节点及新购的数据节点)需要通过异构数据节点服务能力计算模块进行计算。数据节点服务能力的计算包括：数据节点实际计算能力、计算

能力、存储能力及使用年限等。当得到了所有异构数据节点服务能力后，使用异构数据节点分配算法将所有异构数据节点按逻辑划分为 4 个数据放置集群，分别为：数据放置集群 1、数据放置集群 2、数据放置集群 3 及数据放置集群 4。其中：数据放置集群 1 用于存储无计算关系数据集（死数据）；数据放置集群 2 用于存储无计算关系数据集（活数据）及无计算关系数据集（修正后）；数据放置集群 3 用于存储孤立计算数据集、孤立计算（修正后）数据集及有关联计算（修正后）数据集；数据放置集群 4 是那些备用异构数据节点所组成的一个数据放置逻辑集群。

4 异构数据节点服务能力计算方法

通过对异构节点的 CPU、内存、外存、I/O、使用年限等等进行分析，建立一个异构节点的能力计算模型。通过该模型计算出数据中心异构节点的服务能力。

我们将建立如下的计算公式来实现异构节点服务能力的计算。

$$ServiceCapability[i]=CPU.Capability[i]*Weights[CPU]+Memory.Capability[i]*Weights[Memory]+Storage.Capability[i]*Weights[Storage]+I/O.Capability[i]*Weights[I/O]+UsingYear.Capability[i]*Weights[UsingYear]+Others$$

其中：*ServiceCapability*[*i*]是指节点的整个服务能力。*CPU.Capability*[*i*]是指 CPU 的服务能力；*Weights*[*CPU*]是指 CPU 部分所占的权重。*Memory.Capability*[*i*]是指 Memory 的服务能力；*Weights*[*Memory*]是指 Memory 部分所占的权重。*Storage.Capability*[*i*]是指 Storage 的服务能力；*Weights*[*Storage*]是指 Storage 部分所占的权重。*I/O.Capability*[*i*]是指 I/O 的服务能力；*Weights*[*I/O*]是指 I/O 部分所占的权重。*UsingYear.Capability*[*i*]是指 UsingYear 的服务能力；*Weights*[*UsingYear*]是指 UsingYear 部分所占的权重（一般来说节点使用年限越久，服务能力越差，同时能耗也会越多）。*Others* 是指今后我们在不断研究中研究出的可能影响节点服务能力的其他因素。

另外，我们计算公式中的所有各项因素的权重大小将通过实验的不断比较，由大量的实验得出一个合理的值。

5 数据放置策略

本文采用了 4 种不同的数据放置策略，分别为：数据放置策略 1、数据放置策略 2、数据放置策略 3 及数据放置策略 4。分别描述如下：

(1)数据放置策略 1

使用 Hadoop 默认的数据放置方案（副本数为 2），一旦数据分配完成，立即关机，达到节省能源的目标。数据策略 1 主要是针对那些死数据的数据放置，这种数据直接使用 2 个副本可以确保安全，同时数据存储完成后，直接关机，节省能源。

(2)数据放置策略 2

使用 Hadoop 默认的数据放置方案（副本数为 3），让其处于节能运行状况。这里的让其处于节能运行状况，是指针对那些计算关系很少的数据集，一旦存储就让这些存储节点以休眠状态来节省能源，只有在偶尔需要进行计算的时候才激活，一旦计算完立即恢复到休眠状态。

(3)数据放置策略 3

使用 Hadoop 默认的数据放置方案（副本数为 3）。

(4)数据放置策略 4

基于 Hadoop 的一种改进的数据放置方案，其主要实现步骤描述如下：

第 1 步　将所有有数据关联的数据集形成一个数据关联子集。

第 2 步　对该数据关联子集进行数据划分。将每个数据集按照 Hadoop 的划分方式划分成每块 64M 的数据块。

第 3 步　将具有关联计算关系的所有数据块打上不同的语义标记号，如 SemanDFlag[1]、SemanDFlag[2]及 SemanDFlag[3]等。

第 4 步　将那些没有关联计算关系的所有数据块打上统一的语义标记号 SemanDFlag0。

第 5 步　将那些具有相同语义标记号（语义标记号为 SemanDFlag0 的除外）的所有数据块按照数据放置策略 4 的机制放到数据放置集群 3 中的同一个数据节点。放置原则可以描述如下：

a)将具有相同语义标记号的数据块形成一个语义表（语义标记号为 SemanDFlag0 的除外），如表 1 所列。

表 1　数据语义标记表

语义标记号	数据块	数据块数量
SemanDFlag[1]	D[i].j,……	Num[1]
SemanDFlag[2]	D[k].j,……	Num[2]
……	……	……
SemanDFlag[m]	D[p].k,……	Num[m]

b)从表 1 中找出数据块数量最大的语义标记号。

c)从数据放置集群 3 中找出服务能力最好的数据节点。

d)将 b)步骤找到的语义标记号所对应的全部数据块放到 c)步骤所找到的服务能力最好的数据节点中。

e)将 b)步骤找到的语义标记号在表 1 中对应的行删除，得到新的表 1。

f)重复步骤 b)到步骤 e)，直到所有的语义标记号所对应的数据块全部分配到数据放置集群 3 中。

第 6 步　将所有语义标记号为 SemanDFlag0 的数据块按照数据放置策略 3 的机制放置到数据放置集群 3 中。这些语义标记号为 SemanDFlag0 的数据块其实不与任何其他数据块发生计算关系（如连接操作、联合操作及其笛卡尔积等），这样我们可以按照 Hadoop 提供的数据放置机制进行放置即可。

结束语　本文提出了一种新的大数据放置模型，该模型充分考虑了能耗、异构节点的服务能力及具有关联计算的数据集。本文基于这 3 个因素设计了一种节能、充分考虑了异构节点的服务能力及提升 MapReduce 处理 Join 连接的效率的大数据放置模型。该模型将有效实现大数据的有效放置管理。大数据的有效放置是大数据能够实现实时处理和分析的关键所在。

为了实现未来对大数据的实时分析和处理，达到“1 秒定律”，即：在 1 秒内返回用户的对大数据计算的结果，未来的数据放置应该更多考虑复杂计算需求的大数据放置策略。本文虽然提出了一种模型来对有关联数据块进行语义标记，将具

（下转第 36 页）

并使用权重感知的遗传算法优化启发式算法所得解。算法采用实际测量值建模信号穿透障碍物所受到的衰减，从而更高效地利用无线传感器节点的感知能力与通信能力；使用划分区域权重的方式根据不同的需求决定不同区域使用节点的数量和位置，加强重要区域的监控力度。实验验证了所提出的算法的优越性。但是，我们所设计的部署方案只考虑了静态障碍物，下一步我们将研究如何更具体地考虑动态障碍物对传感器节点信号的影响。

参 考 文 献

[1] 刘丽萍，王智，孙优贤. 无线传感器网络部署及其覆盖问题研究[J]. 电子与信息学报，2006，28(9)：1752-1757

[2] 莺池，粱奕，周晓峰. 一种能量异构自适应的无线传感器网络覆盖控制协议[J]. 计算机科学，2009，36(5)：39-44

[3] Xiong S，Yu L，Shen H，et al. Efficient algorithms for sensor deployment and routing in sensor networks for network-structured environment monitoring[C]// INFOCOM，2012 Proceedings IEEE. IEEE，2012：1008-1016

[4] Bai X，Xuan D，Yun Z，et al. Complete optimal deployment patterns for full-coverage and k-connectivity (k≤ 6) wireless sensor networks[C]// Proceedings of the 9th ACM International Symposium on Mobile Ad hoc Networking and Computing. ACM，2008：401-410

[5] Bai X，Yun Z，Xuan D，et al. Optimal patterns for four-connectivity and full coverage in wireless sensor networks[J]. IEEE Transactions on Mobile Computing，2010，9(3)：435-448

[6] Oh Y，Schmidt A，Woo W. Designing，developing，and evaluating context-aware systems[C]// International Conference on Multimedia and Ubiquitous Engineering，2007 (MUE' 07) IEEE，2007：1158-1163

[7] Alam S M，Haas Z J. Coverage and connectivity in three-dimensional networks[C]// Proceedings of the 12th Annual International Conference on Mobile Computing and Networking. ACM，2006：346-357

[8] Kouakou M T，Yasumoto K，Yamamoto S，et al. Cost-efficient sensor deployment in indoor space with obstacles[C]// 2012 IEEE International Symposium on World of Wireless，Mobile and Multimedia Networks (WoWMoM). IEEE，2012：1-9

[9] Watfa M K，Commuri S. A coverage algorithm in 3D wireless sensor networks[C]// Proc. of the 1st Int'l. Symp. on Wireless Pervasive Computing(ISWPC 2006). 2006：10-16

[10] Bai X，Zhang C，Xuan D，et al. Low-connectivity and full-coverage three dimensional wireless sensor networks[C]// Proceedings of the Tenth ACM International Symposium on Mobile Ad hoc Networking and Computing. ACM，2009：145-154

[11] Bai X，Zhang C，Xuan D，et al. Full-coverage and k-connectivity (k=14，6) three dimensional networks[C]// INFOCOM 2009，IEEE. IEEE，2009：388-396

[12] Benkic K，Malajner M，Planinsic P，et al. Using RSSI value for distance estimation in wireless sensor networks based on ZigBee[C]// 15th International Conference on Systems，Signals and Image Processing，2008(IWSSIP 2008). IEEE，2008：303-306

[13] Xu J，Liu W，Lang F，et al. Distance measurement model based on RSSI in WSN[J]. Wireless Sensor Network，2010，2(8)：606-611

[14] 郭山红，孙锦涛，谢仁宏，等. 电磁波穿透墙体的衰减特性[J]. 强激光与粒子束，2009，21(1)：113-117

[15] 王召召，邹澎，王瑶. 建筑物对通信信号衰减的定量研究[J]. 通信技术，2010，5：23

(上接第 4 页)

有语义标记的大数据放置在同一节点，但是其仍然有许多不足。未来要着重解决如下几个问题。

(1)基于时间段的大数据语义标记。未来对大数据的实时分析，不一定需要对整个大数据集进行全部分析，可能只需要分析从某个时间段到另外一个时间段的大数据。对于这种需求，我们针对大数据的未来标记应该设计出一种基于时间段的大数据语义标记机制和方法。

(2)可编程大数据放置策略。云计算的发展终态是可编程数据中心。未来对于大数据的放置，也将朝着可编程的方向发展，因此未来有用户编程能够控制的可编程大数据放置策略应该为未来的重点研究内容。

(3)未来将以 Hadoop 为基础搭建一个大的数据中心平台，以便对不同的数据集进行测试得到实验对比数据。

参 考 文 献

[1] 覃雄派，王会举，李芙蓉，等. 数据管理技术的新格局[J]. 软件学报，2013，24(2)：175-197

[2] Kaushik R T，Bhandarkar M. GreenHDFS：towards an energy-conserving，storage-efficient，hybrid Hadoop compute cluster[C]// HotPower'10 Proceedings of the 2010 International Conference on Power Aware Computing and Systems. 2010：1-5

[3] Arasanal R M，Rumani D U. Improving MapReduce Performance through Complexity and Performance Based Data Placement in Heterogeneous Hadoop Clusters[C]// Distributed Computing and Internet Technology，Lecture Notes in Computer Science. 2013，7753：115-125

[4] Xie J，Yin S，Ruan X，et al. Improving MapReduce performance through data placement in heterogeneous Hadoop clusters[C]// Proceedings of IPDPS Workshops. 2010：1-9

[5] 王俊伟. 大规模多媒体存储系统中数据放置与调度策略的研究[D]. 长沙：国防科学技术大学，2005

[6] 林伟伟. 一种改进的 Hadoop 数据放置策略[J]. 华南理工大学学报：自然科学版，2012(01)：152-158

[7] Mohamed Y，Tian Yuan-yuan，Özcan F，et al. CoHadoop：flexible data placement and its exploitation in Hadoop[J]. Proceedings of the VLDB Endowment VLDB2011，2011，4(9)：575-585

[8] Dittrich J，Quian'e-Ruiz J A，Jindal A，et al. Hadoop＋＋：Making a yellow elephant run like a cheetah (without it even noticing)[J]. PVLDB2010，2010，3(1/2)：518-529

[9] Zhao Yan-rong，Wang Wei-ping，Meng Dan，et al. A data locality optimization algorithm for large-scale data processing in Hadoop[C]// ISCC 2012. 2010：655-661

[10] 赵彦荣，王伟平，孟丹，等. 基于 Hadoop 的高效连接查询处理算法 CHMJ[J]. 软件学报，2012，23(8)：2032-2041