海量遥感数据管理研究

李振举,李学军,杨 晟,罗 剑
(装备学院,北京 101416)

摘 要:随着遥感技术的发展,遥感数据的种类不断增加,遥感影像分辨率不断提高,遥感数据的量级呈现不断增加的趋势,大数据时代的遥感数据存储和处理都面临着严峻的挑战,传统的数据存储和处理已不能满足遥感数据管理的需求。以数字地球项目为背景,为有效地管理项目中的海量遥感数据,首先对遥感数据物理存储结构进行设计,从底层数据存储层对数据结构和编码方式进行了规划;其次为了提高数据的检索效率,在现有四叉树索引的技术上,提出了一种基于二叉树的最小包围盒索引结构,相比于其他索引结构具有设计简单、树结构平衡、检索效率高的特点。实验结果表明,提出的数据存储结构和索引结构可以满足数据入库、索引构建和数据查询的要求,适合于支持海量遥感数据的存储和管理。

关键词:遥感数据管理;存储结构;最小包围盒;二叉树
中图分类号:TP31 **文献标识码**:A **文章编号**:1673-629X(2015)11-0152-04
doi:10.3969/j.issn.1673-629X.2015.11.030

Research on Massive Remote Sensing Data Management

LI Zhen-ju, LI Xue-jun, YANG Sheng, LUO Jian
(Equipment Academy of PLA, Beijing 101416, China)

Abstract: With the development of remote sensing technology, there is a remarkable increase in the kind of remote sensing data and resolution of remote sensing image data, the traditional data storage and processing cannot meet the requirements of remote data management, which is a challenge for the traditional remote sensing data management mode. Based on digital earth project, in order to manage the massive remote sensing data effectively, firstly introduce a physical storage structure and design the data structure and coding mode. With the purpose to improve query efficiency, propose a index based on binary tree and minimum bounding box, which is more simple, better balanced, more effective than the other index. The experimental results illustrate the proposed data storage structure and index is suitable for managing massive remote sensing data, which can meet the requirements of data input, index building and data query.

Key words: remote sensing data management; storage structure; minimum bounding box; binary tree

0 引 言

随着遥感技术的发展,遥感数据在遥感测绘、航空航天和数字地球、数字城市等多个领域得到了应用。遥感数据的类型、数据来源、时相特征和分辨率不断增加[1],如何对海量的遥感数据进行高效的管理就成为了迫切需要研究的问题。

遥感数据是典型的非结构化数据,对于遥感数据的管理也可以看作是对非结构化数据的管理[2]。传统的非结构化数据管理主要包括两种方式。一是使用非结构化数据管理系统。比较典型的系统包括微软的 WinFS 系统、Google 的谷歌文件系统[3]、开源社区提供的 Hadoop 文件系统等;解决非结构化数据管理问题的另一个方法是以原有文件系统为基础,开发新型的索引以提高非结构化数据的管理效率。构建的索引方式包括 B-、B^+-树、二叉树、ISAM 索引和基于哈希的索引[4]等,这些索引技术设计的初衷都是解决一维属性数据的索引,需要通过降维将多维空间数据转化为一维数据。

1 遥感数据存储结构设计

存储结构对于遥感数据管理非常重要,决定系统的空间利用率、访问速度等性能。为有效管理海量数据[5],一方面要求存储空间利用率要高,另一方面要求支持快速读写。而且非结构化数据其数据长度不固

收稿日期:2015-01-20 **修回日期**:2015-04-22 **网络出版时间**:2015-09-23
基金项目:总装备部预研项目(513150701)
作者简介:李振举(1987-),男,博士生,CCF 会员,研究方向为云计算、海量遥感数据管理;李学军,教授,博士生导师,研究方向为计算机图形学、遥感图像处理、数字地球。
网络出版地址:http://www.cnki.net/kcms/detail/61.1450.TP.20150923.1503.018.html

定,这要求管理系统能够根据数据长度为每条记录分配合适的存储空间。这是文中遥感数据存储结构设计的两个主要目标。

为了有效地处理非结构化数据字段可变的特点[6-7],物理存储层将数据记录封装成特定格式的数据包进行存储。每个数据包的内容包括数据记录的长度、记录 ID 以及记录的各个字段,每个字段使用“字段定义—字段内容”的格式进行存储。其中,字段定义包括字段的语义信息和长度,语义信息保存在“数据字典”中,由系统统一管理。长度信息是每个具体的字段划分存储空间的依据。这种设计模式支持多种格式的非结构化数据,并且可以进行统一管理。

1.1　数据记录格式设计

数据库的主体是数据记录。数据记录由若干字段组成,因此可以根据字段结构设计数据记录的存储格式。在存储层,数据记录被封装成数据包,每个数据包由字段和若干标志信息组成。

每条数据记录由 int 型的 Size 字段开始,存储本条数据记录的长度。RID 字段负责记录在库内的 ID,数据库中每条数据记录都有唯一的记录 ID 作为身份标识,记录 ID 为 int 型数据。之后是若干条 FID-DATA 格式的字段,FID 为字段定义,DATA 为字段内容。最后是 4 位 0 作为记录的结束标志。数据包全部以二进制形式保存在文件中,这种文件称为“主数据文件”,扩展名称为. data。主数据文件由文件头和文件内容两部分构成,文件内容就是记录的数据包。文件的前 32 个字节为文件头。前 16 个字节为文件标志及版本号。内容为“CG. DATA 2010. 0”,其中“CG. DATA”为文件标志,“2010. 0”为版本号。后 16 个字节保留用于加密。从第 33 个字节开始为第一条数据记录。

在主数据文件中,一条数据记录由 Size 开始,到标志符 0000 结束。记录在文件中的位置用记录的开始字节,即 Size 距离文件头的偏移量来表示,因此第一条数据记录的地址为 33。位置信息存储在一个 int 型变量 m_addr 中。系统需要取一条数据记录时,首先找到该地址,然后可以根据 Size 中的信息向后取相应大小的字节数,也可以一直向后读取直到遇到结束符 0000 为止,可取得一条完整的数据记录。

1.2　编码结构设计

为有效进行遥感数据的管理,采用树形结构来表示数据存储的逻辑结果[8-9]。和传统的线性结构相对,树形结构可以更好地表示遥感数据之间的逻辑关系。文中只需要通过树形结构表示数据记录之间的逻辑关系,不需要对物理地址进行表示,因此可以使用记录的 ID 作为参数进行运算。系统为数据库中的每条记录分配一个 ID。记录 ID 是该条记录在库内的唯一标识。所有记录 ID 采取统一的编码规则和管理方式。ID 长度固定为 32 bit,因此可以表示 $2^{32}=4$ G 个条目,即单个数据库中数据记录的数量上限为 4 G。下面通过介绍 ID 的分配规则说明如何利用数据记录 ID 来表示树型逻辑结构。

每一个 ID 码包括类别码和顺序码两部分[10-11]。类别码表示该记录在树型目录结构中的层次关系,顺序码表示同层次叶子节点的兄弟关系。

记录 ID 从高位到低位分为 4 个部分。PN 部分用以表示该节点的父节点,BN 部分用以表示该节点和同层的兄弟节点,CN 部分用以表示该节点的子节点,剩余为顺序码,表示叶子节点之间的兄弟关系。对于以上四个部分来说,任意一部分的数据长度决定了该部分的数据容量。设某一部分的长度为 n,则其数据容量为 2^n,即该层次中节点数量的上限为 2^n。

在数据字典中保存有一个名为“编码结构”的字段,其功能是在产生新 ID 时,由用户指定 ID 各部分所占的位数。用户可以根据不同数据库的具体情况进行合理分配,保证每一部分有足够的容量,同时尽量减少地址空间浪费。

编码结构在数据字典中的定义如表 1 所示。编码结构是 int 型数据,采用 16 进制显示,共有 8 位。编码结构中每两位十六进制数据表示一个子项目,按从高到低的顺序依次表示在 ID 中为父节点、兄弟节点、子节点分配多少位。编码结构最后两位设置为 0。编码结构中的前两部分,即节点的父节点位数和兄弟节点位数分别是该节点父节点编码结构中的兄弟节点位数和子节点位数。因此确定一个节点的编码结构时,前两部分从父节点的编码结构中继承,第三部分即子节点位数由用户指定。

系统中还设置了一个名为“最大编码”的字段。最大编码包括该节点的最大子节点号和最大叶子节点号,用来快速产生新的 ID 及确定子节点范围,采用的数据结构与 ID 一致。最大编码在数据字典中的定义如表 2 所示。

表 1　编码结构定义

编号	字段名	中文名称	英文名称	数据长度	操作标记	备注
3006	IDSTRUCT	编码结构	IDStruct	4	HX	ID 结构

表 2　最大编码定义

编号	字段名	中文名称	英文名称	数据长度	操作标记	备注
3007	MAXCODE	最大编码	IDMax	4	HR	最大子类及叶子编码

2 遥感数据管理索引构建

空间索引负责描述存储在介质上的数据位置信息,可以提高系统对数据获取的效率。使得传统的 B 树索引并不适用于遥感数据的多维性,因为 B 树所针对的字符、数字等传统数据类型都在一个维度上,集合中任给两个元素,都可以在单独维度上确定其关系,而空间数据的多维性,在任何方向上并不存在优先级问题。目前最常见的空间数据索引有 R 树及其变种,四叉树等算法。本节采用最小包围盒四叉树算法来构建索引。

2.1 最小包围盒二叉树算法

最小包围盒二叉树算法以数据的包围盒表示其空间位置信息,将所有记录的包围盒信息保存在一棵二叉树中。每条数据记录的索引对应树中的一个叶子节点,非叶子节点由叶子节点或其他非叶子节点合并而成,保存两个子节点的公共包围盒。

建树算法可以描述为:叶子节点内保存该节点的包围盒以及该条数据记录在主数据文件中的记录 ID;非叶子节点内保存两个叶子节点的公共包围盒以及指向叶子节点的指针;三个节点以上时需要选取其中两个并进行合并,选择合并节点时以合并后的中间节点包围盒最小为优先原则。加入新节点时,自顶向下进行合并,直到将新节点插入到叶子节点的位置为止。具体算法描述如下:

Step1:对库中所有数据记录计算包围盒信息,每个叶子节点存储三方面内容:数据记录的包围盒、数据记录在主数据文件中的存储地址、指向父节点的指针,初始值为空。

Step2:树中加入第一个节点 A,以该节点为根节点。

Step3:加入第二个节点 B,将两个节点的合并作为根节点 AB,以两个节点分别为左右叶子节点。合并后的 AB 中保存两个子节点的公共包围盒以及指向两个子节点的指针。

Step4:当加入第三个节点时,则将其中两个节点合并。此时有三种合并方案,比较合并后产生的中间节点的包围盒 Bound_{AB}、Bound_{AC}、Bound_{BC},选择方案中包围盒面积最小的包围盒作为最终合并方案,$\mathrm{Bound}_{Result} = \mathrm{Min}\{S_{\mathrm{Bound}_{AB}}, S_{\mathrm{Bound}_{AC}}, S_{\mathrm{Bound}_{BC}}\}$。如图 1 所示,假设图中 AB 在三者中面积最小,故将 AB 合并,节点 A、B 作为其子节点。中间节点 AB 再与节点 C 分别作为根节点的左、右子节点。

Step5:当加入三个以上节点时,将新加入的节点与原树每层的左、右子节点进行比较,在此三个节点中按 Step4 选择合并方案,直到将新加入的节点插入到叶子节点位置为止。

通过上述方法,将数据文件中的所有记录保存在一棵二叉树中,将二叉树索引以二进制形式保存在文件中,文件名为库 ID,后缀名为. tree。

查询时,通过传入空间坐标范围,自顶向下从树中查找包围盒与坐标范围相交的节点,如果节点与调用坐标范围相交,则在该节点的子树内继续查询,直到查找到叶子节点为止。查找到叶子节点后,在叶子节点的数据结构中得到数据记录的 ID,完成查找操作。对于包括 N 条数据记录的文件,如果不使用该索引,需要在文件中顺序查找每个节点的包围盒并判断是否需要调入,查找效率为 $O(N)$。建立二叉树后,树中叶子节点数量为 N。最差情况下,节点的包围盒之间互相叠加,查找效率为 $O(N)$。在实际应用中,节点的包围盒相交情况较少时,查找效率接近于 $O(\log N)$。

2.2 索引性能比较

最小包围盒二叉树与 R 树原理[12]相似,但也存在很大差异。与 R 树相比,最小包围盒二叉树有如下特点:

(1)最小包围盒二叉树形状结构与插入顺序无关。R 树在建树时,节点不同的插入顺序会使得产生不同形状的 R 树,为获得性能较好的 R 树,需要将空间位置相邻的节点尽量集中在一个父节点下。最小包围盒二叉树在插入新节点时,会逐次比较合并后的包围盒大小,选择合并后公共包围盒最小的方案,这样就自动保证了空间位置上相近的节点属于同一棵子树,即使节点插入顺序改变,也不会影响树的结构形状。

(2)搜索执行情况不同。R 树是 n 叉树,每层需要进行 n 次比较;在 R 树中搜索时,因为叶子节点全部出现在同一层,搜索时需要到达叶子节点层;进入叶子节点后执行线性查找操作,最终找到所需记录。最小包围盒二叉树每层只需进行两次比较;叶子节点可以出现在树中任意层次,因此搜索时可以不到达树的最底层;每个叶子节点内只存储一条数据记录,节点内无须执行线性查找。但是由于 R 树每层内容纳的节点数量较多,并且每个叶子节点内可以容纳多条数据记录,因此当数据记录数目相同时,R 树的深度会比最小包围盒二叉树小。

(3)数据结构简单。最小包围盒二叉树中每个节点只需保存包围盒信息,叶子节点内存储记录的 ID。索引文件中按树的先根顺序排列。

与四叉树相比,最小包围盒二叉树从数据的空间分布情况考虑,更适合处理数据分布不均匀的情况[13]。

图 1 展示的是数据记录空间位置的分布和四叉树[14]划分情况。图(b)为四叉树算法所建立的索引树结构,图(c)为按最小包围盒二叉树所建立的索引树结构。对比两棵索引树的结构可以看出,当数据记录

分布不均匀时,四叉树索引中会产生多余节点(如一级子节点 NW),并且当数据被分割到多个节点内时(如数据记录 D),树中会用多个节点保存同一条记录;而根据最小包围盒二叉树算法所建立的索引树则不会出现这个问题。与四叉树相比,最小包围盒二叉树的优点在于树中没有冗余节点,并且不存在单条记录被保存在多个节点的问题。最小包围盒二叉树更适合处理数据记录空间位置分布不均匀的情况,因而更加适合处理海量遥感数据。

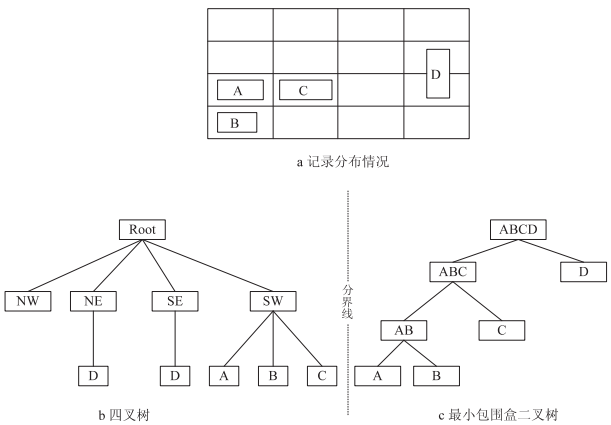

图 1　与四叉树的索引算法对比图

3　实验结果与分析

本节对系统进行性能测试和分析。主要从数据入库、空间索引建立和记录查询三个方面进行测试。测试使用的计算机硬件配置:开发环境为 VS2010 的 VC++,PC 机内存 2.0 G,主频 3.0 GHz。以日本月亮女神拍摄的月图影像的 DEM 作为数据源,程序读取文件夹中的数据,对影像进行批量入库。

3.1　数据入库性能测试

首先对系统数据入库性能进行测试。入库时,系统需要磁盘读入数据,生成主数据文件、空闲空间文件,并同时为记录分配 ID、建立索引。按数据量从小到大的顺序进行测试,并记录系统运行时间。测试结果如图 2 所示。

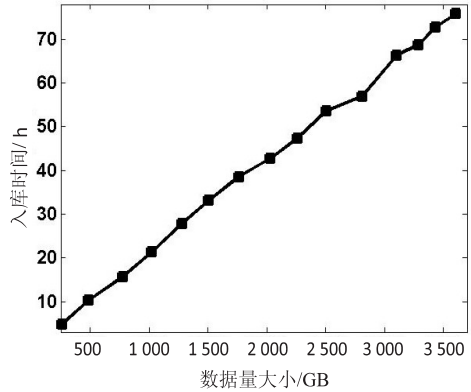

图 2　遥感数据入库时间测试

由于 DEM 影像数据块的大小不一,难以保证入库测试时的数据量成整数倍递增,但从图中仍然可以看出数据量与入库时间大致呈线性关系,每小时可以入库的数据量为 50 GB 左右。

3.2　索引构建性能测试

同样以月亮女神拍摄的月图影像的 DEM 作为数据源,对建立索引及查询时间进行测试,实验测试结果如图 3 所示。

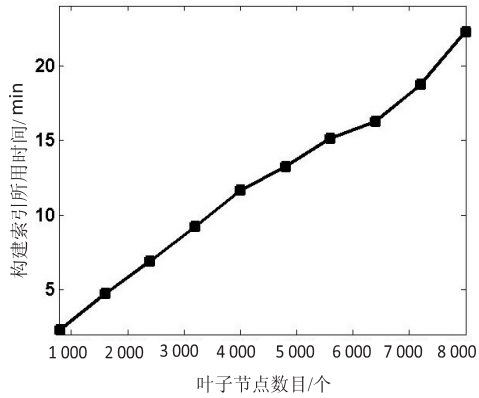

图 3　索引构建时间随数据量变化曲线

3.3　索引查询性能测试

查询时间是系统的重要性能之一,以不同叶子节点数目 N 对应的查询时间以及叶子节点数目的对数 $\log_2 N$ 作为测试指标,结果如图 4 所示。

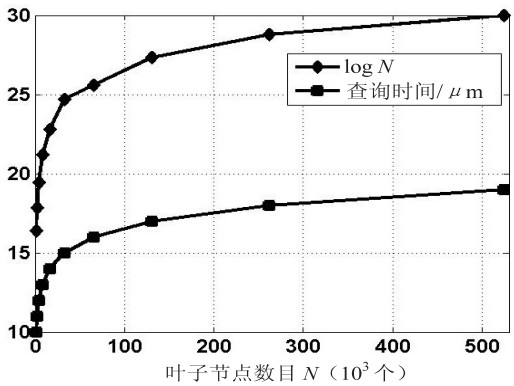

图 4　查询时间和叶子节点数目对数随叶子节点数目变化曲线

对于叶子节点数目 N 而言,其对应的查询时间为 T,设 $S = T/\log_2 N$,经对上表结果计算可知,S 的值处于 17 左右,符合二叉树查找的时间复杂 $O(\log_2 N)$。

4　结束语

为有效管理海量遥感数据,文中设计了统一的物理存储结构,将不同种类的非结构化数据按相同的格式存储在二进制文件中,实现了数据的统一录入和统一管理。通过设计树型结构使得系统可以读数据一致存放,通过库 ID 和记录 ID 实现对数据的引用,避免了

(下转第 162 页)

vent processing over streams[C]//Proceedings of ACM conference on management of data. Chicago: ACM, 2006: 407-418.
[5] 陈 琳,彭商濂,尹方鸣,等. 多维度的 RFID 复杂事件处理优化算法研究[J]. 计算机仿真,2009,26(8):360-364.
[6] 陈 远,李战怀,陈 群. 不可靠 RFID 数据上的复杂事件处理研究[J]. 计算机应用研究,2009,26(7):2537-2539.
[7] Tao K, Zhu Y L, Hu K Y, et al. A novel distributed complex event processing for RFID application[C]//Proceedings of third international conference on convergence & hybrid information technology. [s. l.]: IEEE, 2008: 1113-1117.
[8] Wang Yongheng, Yang Shenghong. High-performance complex event processing for large-scale RFID applications[C]//Proceedings of 2nd international conference on signal processing systems. [s. l.]: [s. n.], 2010: 127-131.
[9] Liu Y, Wang D. Complex event processing engine for large volume of RFID data[C]//Proceedings of second international workshop on education technology and computer science. [s. l.]: [s. n.], 2010: 429-432.
[10] Bok K S, Yeo M H, Lee B Y, et al. Efficient complex event processing over RFID streams[J]. International Journal of Distributed Sensor Networks, 2012, 2012: 71-81.
[11] Nishimura N, Kawashima H, Kitagawa H. A high throughput complex event detection technique with bulk evaluation[C]//Proceedings of the 2013 eighth international conference on P2P, parallel, grid, cloud and internet computing. [s. l.]: [s. n.], 2013: 624-629.
[12] 汤 新. Ceper:一个高性能复杂事件处理引擎[J]. 电脑与信息技术,2013,21(4):32-37.
[13] 阳建坤,祖向荣. 一种基于 CEP 发布订阅中间件应用研究[J]. 中国科技信息,2014(24):103-104.
[14] Wang F S, Liu S R, Liu P, et al. Bridge physical and virtual worlds: complex event processing for RFID data streams[C]//Proc of EDBT. [s. l.]: [s. n.], 2006: 588-607.
[15] Fuhrer P, Guinard D, Liechti O. RFID: from concepts to concrete implementation[C]//Proceedings of the international conference on advances in the internet, processing, systems and interdisciplinary research. [s. l.]: [s. n.], 2006: 1-12.
[16] 彭小娟,刘世安,熊春如,等. 复杂事件处理在大规模 RFID 数据通信中的应用研究[J]. 化工自动化及仪表,2009,36(4):76-79.
[17] Chakravarthy S, Krishnaprasad V, Anwar E, et al. Composite events for active databases: semantics, contexts and detection[C]//Proceedings of international conference on very large data bases. [s. l.]: [s. n.], 1994: 606-617.
[18] 谷 峪,于 戈,张天成. RFID 复杂事件处理技术[J]. 计算机科学与探索,2007(3):255-267.
[19] Akdere M, Cetintemel U, Tatbul N. Plan-based complex event detection across distributed sources[C]//Proceedings of international conference on very large data bases. [s. l.]: [s. n.], 2008: 66-67.

(上接第 155 页)

数据的冗余;提出了最小包围盒二叉树算法,用以建立空间数据索引。该算法具有数据结构简单、树结构平衡、索引性能好等优点。实验最终结果表明,提出的管理机制可以有效管理海量遥感数据。

参考文献:

[1] Nativi S, Mazzetti P, Santoro M, et al. Big data challenges in building the global earth observation system of systems[J]. Environmental Modelling & Software, 2015, 68: 1-26.
[2] Yang C, Goodchild M, Huang Q, et al. Spatial cloud computing: how can the geospatial sciences use and help shape cloud computing? [J]. International Journal of Digital Earth, 2011, 4(4): 305-329.
[3] Chang F, Dean J, Ghemawat W C, et al. Bigtable: a distributed storage system for structured data[J]. ACM Transactions on Computer Systems, 2008, 26(2): 1-14.
[4] 李国斌. 空间数据库技术[M]. 北京:电子工业出版社,2010.
[5] 黄 杰,刘仁义,刘 南,等. 海量遥感影像管理与可视化系统的研究与实现[J]. 浙江大学学报:理学版,2008,35(6):701-706.
[6] 陈金水,王 崟. 非结构化数据存储管理的实用化方法[J]. 计算机与现代化,2006(8):25-28.
[7] van Oosterom P. Scaleless topological data structures suitable for progressive transfer: the gap-face tree and gap-edge forest[J]. Cartography and Geographical Information Science, 2005, 32(4): 331-346.
[8] 李东军,曾国荪. 一种基于四叉树的空间数据缓存策略[J]. 计算机工程与应用,2008,44(22):162-165.
[9] 唐立文,廖学军,汪荣峰. 基于四叉树的海量空间数据模型研究[J]. 装备指挥技术学院学报,2007,18(2):70-74.
[10] 王晨星. 海量遥感影像数据管理系统的设计与实现[D]. 北京:装备学院,2012.
[11] 李文琦. 数字地球中的非结构化数据存储与管理方案研究[D]. 北京:装备学院,2010.
[12] 邓红艳,武 芳,翟仁健,等. 一种用于空间数据多尺度表达的 R 树索引结构[J]. 计算机学报,2009,32(1):177-184.
[13] 艾廷华,帅 赟,李精忠. 基于形状相似性识别的空间查询[J]. 测绘学报,2009,38(4):356-362.
[14] 董 鹏,杨崇俊,芮小平,等. 一种基于改进四叉树的 GIS 空间选择查询算法—以 ESRI SHAPE 格式文件为例[J]. 计算机工程与应用,2013,39(13):58-61.

海量遥感数据管理研究

作者： 李振举， 李学军， 杨晟， 罗剑， LI Zhen-ju， LI Xue-jun， YANG Sheng， LUO Jian
作者单位： 装备学院,北京,101416
刊名： 计算机技术与发展 ISTIC
英文刊名： Computer Technology and Development
年，卷(期)： 2015,25(11)