软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2014,25(9):1937−1952 [doi: 10.13328/j.cnki.jos.004636]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

图数据表示与压缩技术综述*

张 宇[1,2], 刘燕兵[1], 熊 刚[1], 贾 焰[3], 刘 萍[1], 郭 莉[1]

[1](中国科学院 信息工程研究所,北京 100093)

[2](中国科学院大学,北京 100049)

[3](国防科学技术大学 计算机学院,湖南 长沙 410073)

通信作者: 熊刚, E-mail: xionggang@iie.ac.cn

摘 要: 对包含亿万个节点和边的图数据进行高效、紧凑的表示和压缩,是大规模图数据分析处理的基础.图数据压缩技术可以有效地降低图数据的存储空间,同时支持在压缩形式的图数据上进行快速访问.通过深入分析该技术的发展现状,将该技术分为基于传统存储结构的压缩技术、网页图压缩技术、社交网络图压缩技术、面向特定查询的图压缩技术 4 类.分别对每类技术详细分析了其代表方法并比较了它们之间的性能差异.最后对该技术进行了总结和展望.

关键词: 图数据管理;空间缩减;图数据压缩;网页图;社交网络

中图法分类号: TP311

中文引用格式: 张宇,刘燕兵,熊刚,贾焰,刘萍,郭莉.图数据表示与压缩技术综述.软件学报,2014,25(9):1937−1952. http://www.jos.org.cn/1000-9825/4636.htm

英文引用格式: Zhang Y, Liu YB, Xiong G, Jia Y, Liu P, Guo L. Survey on succinct representation of graph data. Ruan Jian Xue Bao/Journal of Software, 2014,25(9):1937−1952 (in Chinese). http://www.jos.org.cn/1000-9825/4636.htm

Survey on Succinct Representation of Graph Data

ZHANG Yu[1,2], LIU Yan-Bing[1], XIONG Gang[1], JIA Yan[3], LIU Ping[1], GUO Li[1]

[1](Institute of Information Engineering, The Chinese Academy of Sciences, Beijing 100093, China)

[2](University of Chinese Academy of Sciences, Beijing 100049, China)

[3](School of Computer Science, National University of Defense Technology, Changsha 410073, China)

Corresponding author: XIONG Gang, E-mail: xionggang@iie.ac.cn

Abstract: How to effectively compress and represent the large-scale graphic data becomes the fundamental issue for analysis and processing. Graphic data compression technology is an effective solution to significantly reduce the storage space while supporting fast access in the compressed form. An in-depth analysis is provided on the current development of the technologies, including compression technology based on the traditional storage structure, Web graph compression technology, social network compression technology and compression technology for a particular query. A detailed analysis and performance comparison about the representative methods of each technology is presented. Finally, the summary and prospect are listed.

Key words: graph data management; space reduction; graph data compression; Web graph; social network

随着移动互联网、物联网等技术的发展,众多新兴应用以前所未有的方式和速度产生并积累着大量数据,如何对这些数据进行分析并使用,已经成为许多领域面临的机遇与挑战,大数据(big data)时代已经到来.2010 年国际超级计算大会(Supercomputing Conference)为评估超级计算机对大数据的处理性能,定义了新的排名方法 Graph500[1],比较超级计算机在图数据(graph data)上的处理能力.在大数据分析的过程中,图作为一种有效描述

* 基金项目: 国家自然科学基金(61202477); 国家科技支撑计划(2012BAH46B02); 中国科学院战略性科技先导专项(XDA06030602)

收稿时间: 2014-01-26; 修改时间: 2014-04-29, 2014-06-09; 定稿时间: 2014-07-05

大数据的数据结构,扮演着越来越重要的角色,在互联网分析[2]、社交网络分析[3]、推荐网络分析[4]等领域,许多计算问题都可以转化为一个基于图的问题,并且使用图上的相关算法来解决.在大规模图数据分析处理应用中,对包含亿万个节点和边的图数据进行高效、紧凑的表示和压缩,是当前的研究热点之一.

在互联网分析中,将每一个页面对应图上的一个节点,将两个页面之间的链接对应图上的一条有向边,从而将互联网转换为一个有向的网页图(Web graph),通过对网页图的分析进行网页的排序.搜索引擎中使用的两种经典的网页排序算法 Pagerank[5]和 HITS[6],都是基于计算图上节点的出度和入度以及节点之间的连接关系等基本操作.在社交网络(social network)分析中,将社交网络中的实体和他们之间的关系转化为相应的图数据.在社交网络图的基础上,可以对社交网络进行相关研究,包括社区发现和重要角色检测[7,8],以及信息传播模式分析[9−11]等.文献[12]提出的垃圾邮件检测方法可以归结为寻找强连通分量、集团枚举和计算最小割等基于图的问题.一些常见的网络挖掘算法,比如网络结构和演化过程的发现都是根据基于图的深度优先搜索、宽度优先搜索、可达性、强连通性和弱连通性等基本算法和性质[13].

为了高效地支持图数据上的基本算法和操作,需要设计一种数据结构来存储这个图,并且可以快速地做一些图上的基本操作,比如查询给定的一个节点的所有邻居或者判断两个节点之间是否联通等.传统的存储方法是采用关联矩阵或者邻接表,为了支持快速的查询,通常将整个关联矩阵或邻接表加载到内存中.但是在实际应用中,这样的方法会面临存储空间过大的问题.以社交网络为例,根据 GlobalWebIndex[14]统计,2013 年 Facebook 用户量已经超过 11 亿,平均每个人的好友超过 100 位,使用邻接表来存储所有用户的关系信息,需要接近 1TB 的存储空间;以互联网为例,根据中国互联网络信息中心(CNNIC)发布的《第 29 次中国互联网络发展状况统计报告》[15],中国网页数量为 866 亿个,超链接数量据估计超过 10^{12},使用邻接表来存储网页直接的链接关系信息需要超过 16TB 的存储空间.同时随着用户量和信息量的快速增长,存储问题也只会变得越来越严峻.

针对大规模图数据存储空间过大的问题,当前主要从 3 个方面进行研究:(1) 硬盘的存储价格相对于内存是非常便宜的,可以使用外存储器存储图数据[16,17],但是由于硬盘的访问速度比内存访问速度慢 4~6 个数量级,导致查询产生较大的延时.可以通过优化图数据处理时的访问局部性,以减少磁盘的 I/O 次数,达到降低访问延时的目的.该技术适用于访问局部性较好的图数据.(2) 使用分布式系统是解决大规模数据的有效方法[18−20],将图数据分割为多个部分,分别存储在分布式系统中不同的计算机内存中,但是由于图数据的耦合性较强,导致分布式系统的通信代价较高,会使查询产生较大的延时.可以通过设计较好的图分割算法,使得分割后的不同子图规模均等并且子图之间的连通性较低,以降低通信代价,较少延时.该技术适用于易于分割的图数据.(3) 将图数据转换为占用空间较小的压缩形式存放在内存中[21,22],同时可以支持查询,查询的时间增长为数倍于不压缩的形式,但是延时远小于前两种方案.该技术适用于访问局部性较差或者耦合性较强不易分割的图数据.

对于上述 3 种解决大规模图数据存储空间过大的方法,本文主要讨论第 3 种,在保证查询时间的前提下压缩存储空间.虽然这种方法并不能解决所有的问题,有些规模特别大的图数据可能压缩后依然不能全部放到内存中,但是也可以通过压缩存储空间来改善另外两种方法的性能.对于硬盘存储的数据结构,如果可以在保持访问局部性的前提下压缩存储空间,那么就可以减少硬盘读取次数,以提高访问速度.对于分布式系统,压缩存储空间可以使用更少的处理节点来完成相同的任务,同时也可以减少处理节点之间的通信代价.因此,对图数据压缩技术的研究是一项非常有意义的工作.

本文第 1 节主要给出图数据压缩技术的问题描述、相关定义以及当前面临的主要问题.第 2 节~第 5 节依次介绍 4 种压缩技术,分别是基于传统存储结构的压缩技术、网页图压缩技术、社交网络图压缩技术和面向特定查询的图压缩技术.第 6 节总结全文并指出一些未来的研究方向.

1 问题描述

1.1 图的基本概念与定义

本文中使用 $G=(V,E)$表示一个图,其中 V 表示图中节点集合、E 表示图中边集合.使用 $n(n=|V|)$表示节点的个数,$m(m=|E|)$表示边的个数.图数据通常采用关联矩阵(adjacency matrix)和邻接表(adjacency list)作为存储结

构.按照图中的边是否有方向,图可分为有向图和无向图,图 1(a)为一个有向图的拓扑结构,图 1(b)和图 1(c)分别为该图对应的关联矩阵和邻接表,表 1 给出了有向图和无向图分别采用关联矩阵和邻接表两种存储结构所需要的存储空间复杂度.本文中将所有图都看作是有向图,图中的边都看作有向边,因为无向图中的无向边可以转化为这条边对应的两个节点之间的两条有向边.对于节点 $v\in V,u\in V$ 使用 $e(v,u)\in E$ 表示图中 v 指向 u 的一条边.使用 $out(v)$表示节点 v 指向的所有节点的集合,$out(v)=\{v|v\in V,e(v,u)\in E\}$,即节点 v 的外邻(out-neighbor).使用 $in(v)$表示指向节点 v 的所有节点的集合,$in(v)=\{v|v\in V,e(u,v)\in E\}$,即节点 v 的内邻(in-neighbor).

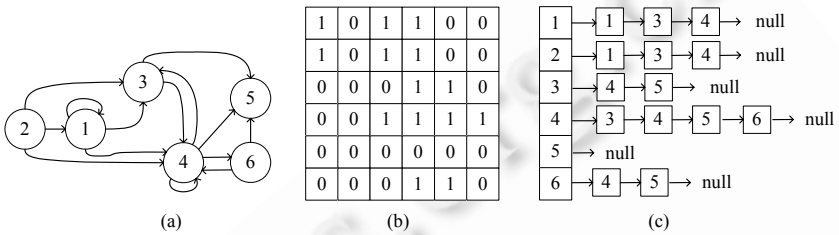


Fig.1 A directed graph and its adjacency matrix and adjacency list structure

图 1 有向图和其对应的关联矩阵与邻接表

Table 1 Comparison of directed and undirected graph space complexity using adjacency matrix and list matrix

表 1 有向图和无向图分别使用关联矩阵和邻接表时的空间复杂度对比

	有向图	无向图
关联矩阵	$O(n^2)$	$O(n(n-1)/2)$
邻接表	$O(n+m)$	$O(n+m)$

从表 1 中可以看出,这两种存储结构的存储代价非常高昂,前面提到的随着社交网络等数据规模的不断增加,存储空间过大问题表现得越来越严重.但是在实际应用中,图数据只需提供几种特定的查询就可以满足需求,并不需要真正存储如此耗费空间的存储结构.比如查询一个图上是否包含一条一个节点指向另一个节点的路径,以及最少经过几个节点就可以到达.对于这种查询,给定一个节点 $v\in V$,只要能查询出 $out(v)$的结果,就可以使用 dijkstra 算法计算出结果.对于图数据上的查询,可以对原始存储结构进行压缩或者构造一种占用存储空间更加小的结构来存储数据,在图上进行查询时,通过简单的计算就可以得到查询结果,这种技术被称为支持查询的图数据压缩技术.评价这种技术通常包括两个指标:压缩率和查询时间.压缩率表示压缩后的存储空间与图中边数的比,用 bpe(bit per edge)表示;查询时间表示判断给定两个节点之间是否存在边需要的时间,单位通常为μs (microsecond).

1.2 图数据压缩技术的发展历程及分类

已知的最早进行支持查询的图数据压缩技术方向研究的是文献[23],该文献中的方法将原图分割为若干子图,每个子图中的节点可以构造一个线性布局(linear layout)以满足子图中节点之间的边互不相交,该方法需要 $O(gn)$ bit 的存储空间(g 表示图中子图的个数),对节点的邻居查询可以在 $O(g+t\log(n))$时间完成(t 表示邻居的个数).之后,文献[24,25]对该算法进行了优化,将邻居查询的复杂度改进为 $O(g+t)$,同时可以支持查询节点的度,也可以在 $O(g)$时间内查询两个节点是否相连.这些早期的算法,更多的是在理论上对压缩问题进行分析,在实际应用中并没有特别好的效果.

从 2000 年开始,随着网络技术的飞速发展,大规模图数据压缩技术越来越受到人们的重视.文献[26–28]使用将原图分割的方法进行压缩,但均不能满足较快的查询.文献[2,29]提出了一种根据图中节点和链接具有相似性的特点进行压缩的方法,文献[21,22]在此基础上提出了 BV 算法,该方法通过对网页图中的 URL 按照字典顺序进行排序,由于顺序相近的 URL 往往包含比较相似的邻居,利用这个性质,BV 算法对网页图的压缩取得了非常好的效果.文献[30–32]先后对 BV 算法在存储空间和访问速度上作进一步优化.文献[33]提出了 VNM 算法,

可以在不影响查询结果的情况下将原图转换为边数更少的图,再结合 BV 算法,可以进一步减少存储空间.文献[34]提出了 AMN 算法,将排序后的 URL 按照域(domain)进行分块,利用域内节点间关系的冗余,获得了较高的压缩率.

网页图可以通过简单的节点排序对图数据进行压缩,但对于其他类型的图数据,例如社交网络对应的图中节点,并没有字典顺序的概念,因此无法通过简单的方法找到可以用以压缩的特性.文献[35]提出的 FRS 算法,定义了一种对图中的节点进行重新排序的方法,使排序后的节点也体现出和网页图类似性质,将重排序后的图再使用 BV 算法可以进行有效的压缩.文献[36]提出了 BFS 算法,同样采用节点重新排序的方法,利用排序后邻接表中相邻行的差异来存储图数据,以实现压缩的目的.文献[37,38]提出的 DSM 算法,通过发现图中的稠密图(dense graph),将原图分割为多个稠密图和一个稀疏图,对稠密图使用特殊的存储结构实现压缩.

通过发掘图数据的内在属性,可以对图数据进行压缩.但并不是所有的图数据都具有某种利于压缩的属性,而且发现这些属性也需要非常大的代价.因此直接压缩传统的图数据存储结构的方法也非常值得研究.传统的图数据存储结构主要包括关联矩阵和邻接表.文献[39]提出的 K^2 树(K^2-tree)算法,使用一种树形结构来存储关联矩阵,将全为 0 的子矩阵只用树中一个节点表示,由于关联矩阵中大多数元素为 0,这种方法有效地节省了存储空间.文献[40]根据图数据对应的邻接表构造一段字符序列,使用 Re-Pair[41]和 LZ78[42]算法对字符序列进行压缩,以达到压缩图数据的目的.

上面提到的压缩方法,在对数据进行压缩的时候,都没有考虑在压缩后的数据上使用哪种方式进行查询.文献[43]提出的 MP_k 算法针对需要同时满足内邻和外邻查询的需求设计了一种特殊的存储结构,已有的算法如果想要同时支持内邻和外邻查询,需要把原始图和其对应的转置同时进行压缩,而这一方法不需要处理原始图对应的转置,并且对内邻和外邻的查询可以达到亚线性时间复杂度.文献[44]提出的 QPGC 算法,针对可达性查询和图模式查询分别设计了不同的压缩方法,该算法通过特定的查询将原图转换为一个新的比原来规模更小的图,再使用已有的压缩技术进行压缩,可以进一步提高压缩效率.

本文将图数据压缩技术的研究分为如图 2 所示的 4 类:基于传统存储结构的压缩技术、网页图压缩技术、社交网络图压缩技术和面向特定查询的图压缩技术.

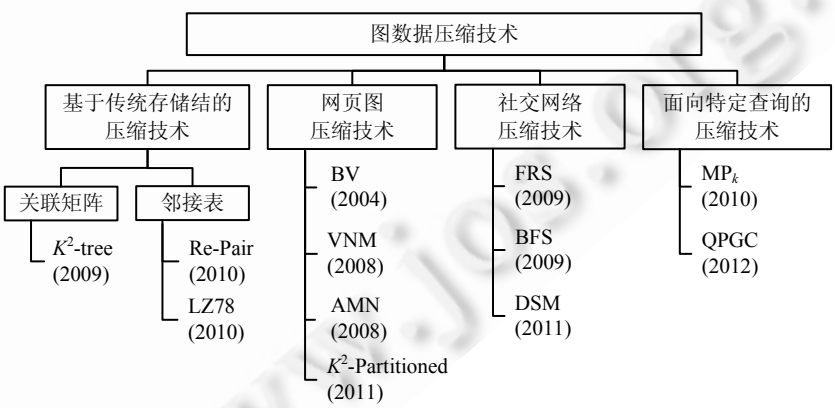


Fig.2 Classification and proposed time of graph data compression technology

图 2 图数据压缩技术分类与提出时间

2 基于传统存储结构的压缩技术

传统的图数据存储结构主要包括关联矩阵,如图 1(a)所示,邻接表如图 1(b)所示.关联矩阵使用矩阵表示图中各节点的关系,对于包含 n 个节点的图,图中的节点为 $V=\{v_1,v_2,...,v_n\}$,使用矩阵 $\{a_{ij}\}$ 表示图中边的信息.矩阵中的值用 0 或者 1 表示,如果 a_{ij} 为 1,则表示 E 包含一条节点 v_i 到节点 v_j 的边 $e(v_i,v_j)$,如果 a_{ij} 为 0,则表示没有这样的边.邻接表是一种链式存储结构,图中的每个节点 v_i 对应着一个链 $adj(v_i)=\{v_{i,1},v_{i,2},...,v_{i,ai}\}$,这个链中存储依次

存储着 v_i 指向的所有节点,其中 $v_{i,j}<v_{i,j+1}(1\leqslant i\leqslant n,1\leqslant j<a_i)$,$adj(v_i)$中节点的个数为 a_i.通常情况下,邻接表比关联矩阵占用更少的空间,但是关联矩阵的访问速度快于邻接表,所以针对不同的图数据和应用需求,这两种存储结构都有着广泛的应用.下面依次介绍基于关联矩阵和邻接表的压缩技术.

2.1 基于关联矩阵的压缩技术

在实际应用中,无论是网页图还是社交网络,使用关联矩阵存储时,产生的矩阵往往具有稀疏性.真实数据中边数远小于节点数,比如社交网络中平均好友数只有大约百人,相对于数以亿计的总人数而言是非常少的,网页图中更是如此.利用关联矩阵的稀疏性,文献[39]提出了一种基于 K^2 树的压缩技术,取得了较好的效果.

K^2 树的每一层将关联矩阵分为 K^2 个节点,树的高度为 $h=\log_k n$.每个节点包含 1bit 数据:除了最底层,1 表示内部节点,0 表示叶子节点;最底层中,0 或 1 表示该节点的值.最高层(标记为第 0 层)对应着根节点,它有 K^2 个孩子在第 1 层.每一个孩子对应着一个节点,标记为 0 或者 1.所有的内部节点(标记为 1)都有 K^2 个孩子,所有的标记为 0 的节点和标记为 1 的叶子节点都没有孩子.构造 K^2 树时,将原始矩阵分为 K 行 K 列个子矩阵后,每个子矩阵作为根节点的一个孩子,每个孩子是包含 n^2/K^2 个元素的矩阵.如果孩子中如果包含 1,则继续按照上述方法分为 K 行 K 列个子矩阵,作为这个孩子的孩子.如果孩子中不包含 1,全部为 0,则不再继续分,这个孩子作为叶子节点.如果子矩阵中只包含一个元素,则无论是 1 或者是 0,都停止继续分.每个节点的孩子按照它在父节点中的位置进行排列,从第 1 行开始,从左到右排列,然后是第 2 行,以此类推.对于生成的 K^2 树,第 0 层为根节点,第 1 层包含 K^2 个节点,依次表示原始矩阵对应的部分,非最底层的 0 表示该位置对应的原始矩阵的元素都为 0,非最底层的 1 表示该位置对应的原始矩阵的元素中必然包含 1,最底层的所有节点标记的 0 或 1 对应着原始矩阵中元素的值.更大的 K 值对应的 K^2 树的层数会更小,但是树中孩子的数量就会增多.

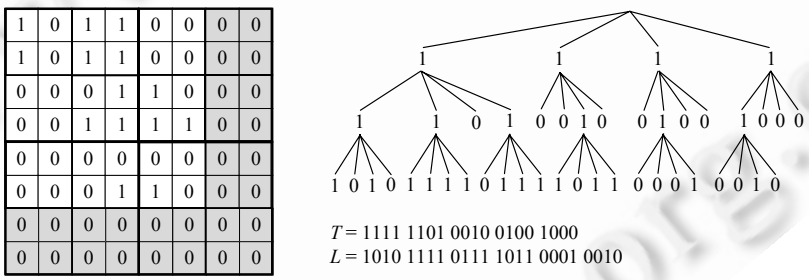


Fig.3 Adjacency matrix and K^2-tree when K=2

图 3 关联矩阵和当 K=2 时对应的 K^2 树

实际应用中,当关联矩阵中的 n 不等于 K 的若干次幂的时候,可以通过增加关联矩阵的行数和列数(新增的元素全都标记为 0)来满足条件.图 3 给出了根据图 1(b)中的关联矩阵在 K=2 时对应的 K^2 树,矩阵中灰色部分为增加的全为 0 的行和列.生成的 K^2 树可以使用两个位向量 T 和 L 来存储,位向量 T 依次存储除最底层外的所有节点,位向量 L 依次存储最底层的所有节点,如图 3 中的位向量 T 和 L.

使用 K^2 树进行查询时,位向量 T 需要支持两种操作 $rank(T,s)$和 $select(T,s)$.$rank(T,s)$表示向量中前 s 位中包含 1 的个数,$select(T,s)$表示向量中第 s 位的值;位向量 L 只需要支持 $select(L,s)$操作.查询关联矩阵中元素 a_{ij} 的值可以通过若干次 *rank* 和 *select* 操作完成.*select* 操作显然可以在常数时间内完成,文献[23]证明使用额外的亚线性存储空间,*rank* 操作也可以在常数时间内完成,文献[39]中使用文献[45]提出算法存储位向量 T,只需增加约 5%的存储空间即可保证快速的 *rank* 和 *select* 操作.故使用位向量来存储 K^2 树可以在减小存储空间的同时保证快速地查询.由于位向量只需要支持 *select* 操作,文献[46]对位向量 L 使用文献[47]中提出的可直接访问的变长编码技术 DAC(direct access to variable-length code),DAC 对位向量 L 进行压缩后 *select* 操作依然可以在常数时间内完成.实验结果表明,K^2 树对网页图的压缩率和查询时间达到 1.23bpe~3.22bpe 和 1.7μs~7.8μs,对社交网络的压缩率和查询时间达到 11.73bpe~17.24bpe 和 5.9μs~15.81μs.

2.2 基于邻接表的压缩技术

邻接表中的每一行都表示一个节点的邻居集合,这些节点按照特定的顺序排列,根据文献[2,29]的分析,图数据中许多节点的邻居集合具有相似性,这就使得邻接表中存储的信息产生了冗余,文献[40]提出了一种基于Re-Pair[41]的压缩算法,很好的利用这一特点对图数据进行压缩.

Re-Pair 是一种基于短语的压缩算法,并且可以支持快速的局部解压缩.该方法通过发现序列中的频繁字符对,并使用新的符号来替换这个字符对,直到没有可替换的字符对为止,已达到压缩的效果.序列 T 使用 Re-Pair 算法来压缩需要完成如下步骤:

(1) 发现在序列 T 中最频繁的模式 ab;

(2) 增加一条规则 $s\rightarrow ab$ 到字典 R 中,其中 s 是一个新的没有在 T 中出现过的符号;

(3) 使用 s 替换序列 T 中所有的 ab;

(4) 从第 1 步开始重复,直到每一对字符在 T 中都只出现 1 次.

将压缩完成的序列称为 C,使用字典 R 可以快速地还原出 C 中任意一个字符 s:首先在 R 中找到 $s\rightarrow s_1s_2$ 这条规则,使用 s_1s_2 替换 C 中的 s,并继续在 R 中寻找 s_1 和 s_2 对应的规则,直到在 R 中找不到对应的规则.

文献[40]将邻接表的存储结构进行了转变,以适用 Re-Pair 算法的压缩方式.对于图中的节点 v_i,与其相邻的节点记为 $adj(v_i)=\{v_{i,1},v_{i,2},...,v_{i,ai}\}$,并且按照节点的序号进行排序.对每个节点 v_i 增加一个符号 v_i' 表示邻接表中每一行的分隔,v_i' 的值记为 $-v_i$.那么就可以将邻接表表示成如下形式:

$$T=v_1'\,v_{1,1}v_{1,2}\ldots v_{1,a1}\,v_2'\,v_{2,1}v_{2,2}\ldots v_{2,a2}\ldots\,v_n'\,v_{n,1}v_{n,2}\ldots v_{n,an}(v_{i,j}<v_{i,j+1},1\leqslant i\leqslant n,1\leqslant j<ai).$$

图 4 给出了根据图 1(c)中的邻接表构造的 T 以及使用 Re-Pair 算法来对 T 进行压缩的过程.图 4 中第 1 行表示生成的 T,下面的每一行表示增加一条规则到 R 中并且替换 T 中的字符,最后一行除去增加的分隔符 v_i'.使用一个数组 $Ptrs$ 来记录每个节点对应的压缩后的数据中的起始位置.在对节点 v_i 进行邻居查询时,先通过数组 $Ptrs[i]$和 $Ptrs[i+1]$找到该节点对应的压缩数据的起始和终止位置$[Ptrs[i],Ptrs[i+1])$,将该区间的数据按照上述方法进行解压缩即可.

T=	1	1	3	4	2	1	3	4	3	4	5	4	3	4	5	6	5	6	4	5
7→4 5	1	1	3	4	2	1	3	4	3	7	4	3	7	6	5	6	7			
8→1 3	1	8	4	2	8	4	3	7	4	3	7	6	5	6	7					
9→8 4	1	9	2	9	3	7	4	3	7	6	5	6	7							
	9	9	7	3	7	6	7													

Fig.4 The process of compression using Re-Pair

图 4 使用 Re-Pair 压缩邻接表的过程

使用 Re-Pair 算法进行压缩后的数据也适于存放在外部磁盘上,因为每次在查询时就可以算出所需数据位于压缩后数据的位置,然后再将这部分数据读到内存中进行解压即可.另外,由于每次增加规则,都会产生额外的两个符号,也会使得 R 的规模越来越大,文献[48]中通过使用文献[49]提出了对 R 进行压缩的方法,可以使 R 的存储空间减少一半.实验结果表明,Re-Pair 算法对网页图的压缩率和查询时间达到 2.20bpe~3.93bpe 和 1.1μs~3.2μs,对社交网络的压缩率和查询时间达到 14.84bpe~22.53bpe 和 4.2μs~7.3μs.

文献[40]提出 Re-Pair 算法的同时,还提出了一种基于 LZ78[42]的压缩算法,LZ78 算法的主要思想是在压缩和解压缩的过程中维护一个字典:压缩时依次从输入的字符序列中发现前缀(prefix),如果前缀出现在字典中,则记录对应字典中的位置,如果没有出现在字典中,则输出上次记录的字典中位置和当前字符,并将当前前缀作为新的规则加入到字典中.

LZ78 算法和 Re-Pair 算法不同的是,LZ78 算法在压缩过程中将字典的信息也输出到了结果中,字典可以在解压缩的过程生成,在存储压缩数据时并不需要存储字典,并且在解压时每次只需查询字典一次即可得到当前位置对应的原始数据,而 Re-Pair 算法需要查询字典多次才能解压,故解压速度快于 Re-Pair 算法.但是 LZ78 算

法在解压时必须从起始位置开始,否则不能生成字典,所以在使用LZ78算法对邻接表进行压缩时只能分别对每个节点的邻居集合进行压缩,这会导致每次压缩的数据规模变小,不能有效利用数据间的冗余,这也限制了LZ78算法的压缩效果.在实际应用中,LZ78算法的压缩率不如Re-Pair算法,但是查询时间略快于Re-Pair算法,实验结果表明该算法对网页图的压缩率和查询时间达到7.02bpe~12.97bpe和0.5μs~2.1μs,对社交网络的压缩率和查询时间达到18.29bpe~25.91bpe和4.0μs~7.1μs.

3 网页图压缩技术

第2节中介绍的基于传统存储结构的压缩技术主要利用图数据的外在特点,对关联矩阵的压缩主要利用矩阵的稀疏性,对链接表的压缩主要利用节点邻居集合的相似性.而在图数据的真实应用中,图数据表示的信息也包含内在的特点.例如网页图中同一个域内的网页之间的链接数比较多,而域之间的连接数就比较少.文献[21,22]通过分析网页图的链接信息,发现根据网页对应URL的字典顺序对网页图进行排序后,字典序相近的网页包含的邻居集合相似的可能性更大.网页图具备以下两个特征:

- 本地性:大多数链接都是域内(intra-domain)的,它们通常会指向字典序比较靠近的页面.
- 相似性:在字典序中邻近页面的邻居集合也是相似的.

文献[21,22]率先根据这两种特性设计了BV算法,该方法可以通过调节参数实现压缩率和查询时间之间的折中,以适应不同的需求.首先,BV算法将所有节点按照其对应URL的字典序进行排序,将排序后的节点从0开始标记,依次标记为0,1,2,…,$|E|-1$,节点v的序号表示为$index(v)$.其次,对任意节点对应的外邻,使用序号在该节点之前的某个节点对应的外邻来表示.具体方法引入了正整数参数W,W表示参考范围,任意节点v对应的$out(v)$使用排在v之前的W个节点中的某个节点u对应的$out(u)$来表示,其中,$|out(v)\cap out(u)|$最大并且满足$index(v)-W\leqslant index(u)\leqslant index(v)-1$,这时称$u$为$v$的参考节点.使用长度为$|out(u)|$的位向量来表示$out(v)$和$out(u)$之间的关系,第$i$个bit($1\leqslant i\leqslant|out(u)|$)如果为1则表示$out(v)$中包含$out(u)$中第$i$个节点,如果为0则表示不包含.对于$out(v)$中包含但是$out(u)$中不包含的节点序列$i_1<i_2<\ldots<i_k$,依次存储它们之间的差异$i_1,i_2-i_1,\ldots,i_k-i_{k-1}$.

BV算法允许节点多次参考,比如v_2为v_3的参考节点,v_1为v_2的参考节点等等.对于k个多次参考的节点序列$v_k,v_{k-1},\ldots,v_1,v_{i-1}$为$v_i$的参考节点,$k\geqslant i\geqslant 2$,这个节点序列称为参考序列,$k$为这个参考序列的长度.BV算法使用参数$\alpha$来限制参考序列的最长长度,当$\alpha$增大时,数据的压缩率就会提高,同时查询速度下降,当α取无穷大时,压缩率达到最高,同时查询速度最慢.文献[30,31]在存储空间上对BV算法进行了优化,实验结果表明当$\alpha=3$时BV算法对网页图的压缩效果和查询时间取得较好的平衡,压缩率达到2.04bpe~5.62bpe,同时查询时间为2.3μs~4.2μs.

文献[33]结合BV算法提出了VNM(virtual node mining)算法.图数据在使用BV算法压缩前,先进行预处理,使用文献[50]中提出的发现完全二部图的方法寻找图中的二部图,在发现的完全二部图中增加一种新的节点,称为虚拟节点(virtual node),这个完全二部图中出度大于0的节点均指向虚拟节点,并且虚拟节点均指向入度大于0的节点,并将二部图中原有的边删去.这样可以有效减少图中的边数,将转换后的图再使用BV算法进行压缩,实验结果表明,该算法对网页图的压缩率和查询时间分别达到1.95bpe~2.90bpe和3.8μs~4.5μs.

在发现通过本地性和相似性可以对网页图进行很好的压缩后,由于同一域内的URL更能体现出本地性和相似性,文献[34]提出的AMN算法将URL排序后按照域分为若干块,对每一块再利用这两种特性分割为6种类型的子块来存储,该方法使压缩率进一步提高.这6种类型的子块分别为:(1) 单元块(singleton block):由孤立的1组成;(2) 水平块(horizontal block):由水平方向上2个或2个以上连续的1组成;(3) 垂直块(vertical block):有垂直方向上2个或2个以上连续的1组成;(4) L型块(L-shaped block):由1个水平块和1个垂直块组成,它们共享位于左上角的1;(5) 矩形块(rectangular block):全部由1组成的子矩阵,包含2行或2行以上以及2列或2列以上;(6) 对角块(diagonal block):从左上方到右下方包含2个或2个以上连续的1.

图5中的关联矩阵被分为多个不同类型的子块.其中,B_1为水平块、B_3为对角块、B_5为矩形块、B_6为垂直块、B_8为L型块,剩余的子块B_2,B_4,B_7,B_9和B_{10}为单元块.在发现这些子块的过程中,当某个元素可以同时属于

多个子块时,选择包含 1 最多的子块,例如 B_6 中的 a_{40} 也可以和 B_7 组成对角块,但是这个对角块中包含的 1 的个数为 2,而 B_6 包含 1 的个数为 3.

按照上述方法发现所有子块后,由于有单元块的存在,这些子块可以覆盖关联矩阵中所有的 1,对每个子块使用开始位置、类型和纬度组成的一个三元组特征来表示:开始位置为子块左上角元素的位置;类型为上述 6 种类型之一;维度为子块的规模,对于水平块、垂直块和对角块,纬度使用 1 个数值来表示,*L* 型块和矩形块使用两个数值来表示,单元块忽略维度值.例如,B_1 使用((0,1),*horizontal*,5)表示,B_2 使用((0,9),*singleton*,1)表示,B_5 使用((2,4),*rectangular*,(3,5))表示,B_8 使用((7,1),*L-shaped*,(3,4))表示.这 6 类子块中除了单元块,均体现出了 Web 图的本地性和相似性,所以使用这些三元组来代替关联矩阵中的 1 进行存储可以有效减少存储空间,实验结果表明,该方法对网页图的压缩率可以达到 1.71bpe~2.78bpe,查询时间达到 2.38μs~28.72μs.

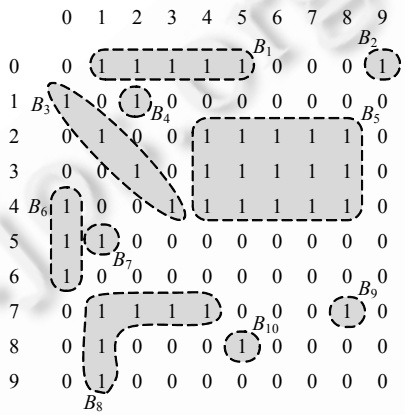


Fig.5 Different types of sub-blocks contained in the adjacent matrix

图 5 关联矩阵包含的不同类型的子块

AMN 算法虽然取得了很好的压缩效果,但是其压缩过程复杂,在实际应用中对于一些规模比较大的图数据需要较长的压缩时间.文献[51]提出的 K^2-Partitioned 算法使用分块与 K^2 树结合的思想,压缩时间、压缩率和访问速度在实际应用中均具有较好的效果.

K^2-Partitioned 算法使用了一种贪心策略将图分割为 t+1 个子图,分别记为 $G_1,G_2,...,G_{t+1}$,前 t 个子图节点和边互不相交,最后一个子图为原始图 G 删除这 t 个子图对应的边后的剩余图,然后再对这 t+1 个子图使用 K^2 树进行压缩.这样做会带来 3 方面的优势:

(1) 对于每一个子图对应的 K^2 树的高度会减小,这会提高查询的速度;

(2) 在查询速度变快的同时,并不会损失压缩率,因为原图中大多边都属于同一个域,而对图进行分割时,同一个域内的节点会尽量地分在同一个子图中;

(3) 剩余图会变成一个非常稀疏的图,也可以很好的发挥 K^2 树的性能,并不需要太大的空间就能够存储.

对图进行分块的方法采用一种贪心策略:将字典序中属于相同域的节点依次加入当前子图,直到当前子图规模大于 M(对于不同的数据 M 的取值也会不同)或者当前节点不属于当前子图中大多数节点所在的域,再接着创建新的子图.通过这种方法构建的子图,有 90%左右的边能够包含在前 t 个子图中,剩余图仅仅包含 10%左右的边.实验结果表明该算法对网页图的压缩率达到 2.11bpe~4.00bpe,查询时间达到 1.0μs~2.3μs.

4 社交网络图压缩技术

由于网页图按照 URL 字典序排序后自然地就可以体现出按域分类的特征,但是许多应用,比如社交网络对应的图数据,并不可以直接对其中的节点进行排序.但是社交网络也包含着类似于网页图体现的特性,例如同一个社区内的好友关系数量比较多,而社区之间的好友关系数量就比较少.那么如何根据这些特性对节点排序,使

得排序后的节点序列体现出类似网页图的相似性和本地性,便成为人们研究的重点.

文献[35]中的 FRS 算法率先提出了对社交网络中的节点进行排序的方法,利用杰卡德相似系数(Jaccard similarity coefficient)将排序问题定义为寻找一种排列方式π,使得 $\sum_{e(u,v)\in E}\lg|\pi(u)-\pi(v)|$ 的值最小,其中$\pi(v)$表示节点 v 在排序后的序列位置. 文献[52]证明求解这种排序的最优解是 NP 难的,根据文献[53]给出了近似解的求解方法,最后将排序后的节点结合 BV 算法对网页图压缩的方法对社交网络进行压缩.实验结果表明该算法对社交网络的压缩率和查询时间分别达到 13.48bpe~15.82bpe 和 2.4μs~4.4μs.

文献[36]根据对社交网络中的节点进行重新排序的思想提出了 BFS(breadth first search)算法,使用宽度优先搜索对节点进行排序,利用排序后邻接表中相邻行之间的数据冗余进行压缩.BFS 算法包含参数 l,将邻接表相邻的每 l 项划分为一块,每块只记录第 1 项的邻居集合,其余项记录与上 1 项的邻居集合的差异,这样可以通过调节 l 的大小来实现访问速度与存储空间的调节,通过实验说明,当 l=4 时压缩效果和查询时间取得较好的平衡,对社交网络的压缩率达到 11.23bpe~17.84bpe,同时查询时间为 6.4μs~40.8μs.

文献[54]也提出了一种对节点进行排序的方法,定义了结构相似度的概念,认为两个节点的结构相似度为这两个节点邻居集合的交集和并集元素个数的比值,并通过深度优先搜索每次优先遍历与当前节点结构相似度最高的节点,最终按照遍历的顺序对节点重排序.重排序之后的图对应的关联矩阵中 1 元素的位置会更加集中,再使用 K^2 树对关联矩阵进行压缩,更多的 1 元素会被分割到同一个 K^2 树节点中.相比不进行重排序直接使用 K^2 树,重排序之后可以使 K^2 树节点减少 30%~40%.

对节点重排序是为了将关系密切的节点集合排到靠近的位置,再利用排序后的序列体现出的本地性和相似性进行压缩.如果可以直接找到关系密切的节点集合,根据本地性和相似性,这些节点和它们之间的边组成的图会是一个稠密图,对这个稠密图进行特殊的压缩处理,同样可以实现有效的压缩.

文献[37]提出的 DSM(dense subgraph mining)算法,使用与第 3 节中 VNM 算法相似的发现完全二部图的方法.但是在社交网络实际应用中发现图中大量节点并不包含在某些完全二部图中,如图 6(a)中 4、5 和 6、7 节点可以组成完全二部图,而 1、2、3 节点却不在任何完全二部图中.于是该算法在图中的每个节点上加入自环(loop),再使用 VNM 算法中发现完全二部图的方法,最后将未包含在完全二部图中的节点的自环删去,就可以将图 6(a)转化为图 6(b)的形式,将原图中的所有边都包含在这个完全二部图中,同时该算法将这个完全二部图使用一个位向量 B 和一个节点序列 X 来表示其中节点的关系,如图 6(c)所示.

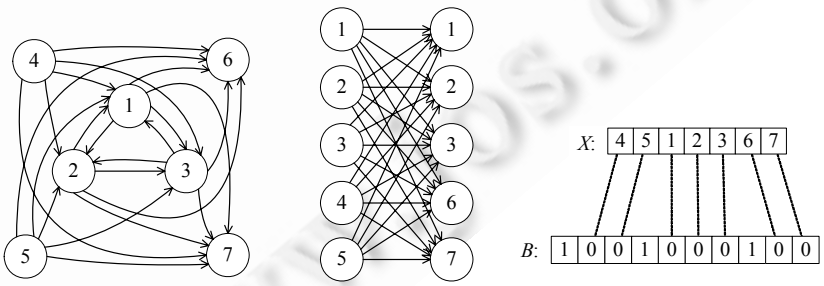


Fig.6 The complete bipartite graph generated by adding loops and its storage coding methods

图 6 通过增加自环生成的完全二部图和其存储方式

在实际应用时,DSM 算法首先从图中发现尽量多的完全二部图,通常可以将图中 90%以上的边包含其中,再将第 i 个完全二部图对应位向量 B_i,与节点序列 X_i 拼接为描述图所有完全二部图的位向量 $B=B_1B_2\ldots B_N$ 和节点序列 $X=X_1X_2\ldots X_N$.对位向量 B 和节点序列 X 使用文献[55,56]提出的压缩方法存储并提供查询.对图中没有被包含在完全二部图中边组成的剩余图,使用 K^2 树来压缩存储.文献[38]对 DSM 算法在完全二部图发现以及存储结构方面做了进一步优化,并通过实验说明 DSM 算法对社交网络的压缩率达到 8.41bpe~13.04bpe,同时查询时

间为 7.6μs~11.8μs.

5 面向特定查询的图压缩技术

许多应用在使用图数据时,有时只使用很少的一种或者几种查询方式,比如内邻或外邻查询、可达性(reachability)查询和图模式(graph pattern)查询等.针对这些不同查询的特点,可以使用特殊的压缩方法,以使查询更加快速高效.文献[43]针对内邻和外邻查询设计了 MP_k(multi-position linearization of degree k)算法,文献[44]提出的 QPGC(query preserving graph compression)算法针对可达性查询和图模式查询将原图简化为规模更小的图,在这个图的基础上依然可以采用已有的压缩技术.

已知的大多算法往往只能支持外邻查询,如果需要查询内邻,则需要更多的时间或空间代价,文献[43]提出的 MP_k 算法很好地解决了这一问题,在不需要额外存储空间的情况下同时支持快速的内邻和外邻查询.该方法采用了一种类似于欧拉路径的方法,如果原图中存在一条欧拉路径,即为图中存在一条每个边遍历一次且仅遍历一次的路径,如果不存在,增加尽量少的边以完成欧拉路径,然后将这条路径上的节点依次排序,这个序列中部分节点可能会出现多次,对于每个节点使用两个 bit 来表示这个节点与两侧节点的关系.

图 7 为 MP_k 算法根据图 1(a)中的图构造的存储结构,图中还存在着一些边,它们指向相同节点的下一个位置,如果节点没有出现重复,则指向该节点本身.在该结构的基础上,还需要额外存储每个节点对应在该结构中的起始位置.使用该结构进行内邻或者外邻查询时,首先找到要查询节点的起始位置,查询该节点和两侧节点的关系,再通过指向该节点下次出现位置的边继续遍历同时查询和两侧节点的关系,直到跳转回起始位置,即可得到该节点所有的内邻和外邻.

在实际应用时,每个节点不仅可以记录与两侧相邻的 1 个节点的关系,也可以使用 $2k$ 个 bit 记录与两侧最靠近的 k 个节点的关系,这样虽然增加了每个节点的存储空间,但是可以表示更多的边以减少节点的数量,所以对于不同的图数据需要选取不同的 k 值.实验结果表明 MP_k 算法对社交网络进行压缩时,k 通常选取 10~40,压缩率达到 8.48bpe~17.02bpe,同时查询时间为 8.0μs~12.9μs.

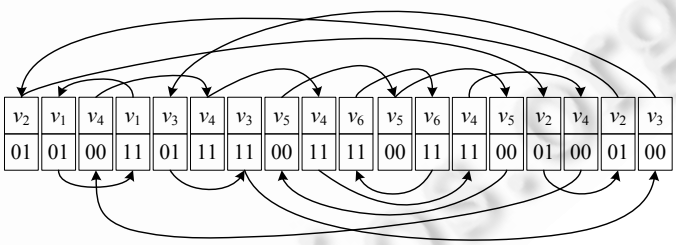


Fig.7 Data structure constructed by MP_k

图 7 MP_k 构造的数据结构

随着新应用的发展,尤其是社交网络的兴起,对图的查询不仅仅有邻居查询或者判断边是否存在,还产生了一些比较复杂的查询方式,比如可达性查询或者图模式查询.如果依旧使用原来的压缩方法,在压缩数据的基础上再附加算法来计算,会导致速度下降.文献[44]中提出的 QPGC 算法针对这两种查询方式,对图 G 进行化简生成 G'并且在查询时并不需要将 G'还原为 G,只需要将查询和查询结果进行转变即可.

对于可达性查询,QPGC 算法将图 G 中所有具有到达该节点的集合和该节点可以到达的节点集合相同的节点归为一类,并且在 G'中使用 1 个节点来表示.图 8(a)中给出了图 G 以及根据可达性查询转换后的图 G',图 G 中节点 A 和节点 E 可以到达的节点集合都为{B,C,D},可以到达节点 A 和节点 E 的集合都为空集,所以可以将节点 A 和节点 E 归为一类.在查询一对节点是否可达时,只需查询这对节点在 G'中对应的节点是否可达即可.

对于图模式查询,使用文献[57]提出的节点互模拟关系(bisimulation relation),对于节点类型相同的两个节点 u 和 v,如果 u 和 v 符合以下两个条件之一,那么 u 和 v 是互模拟关系.这两个条件分别是:

(1) u 和 v 的类型相同并且 $out(u)$和 $out(v)$都为空集;

(2) 对于每一个节点 $u'(u' \in out(u))$,都存在节点 $v'(v' \in out(v))$,满足 u'和 v'为互模拟关系,反之亦然.

将图 G 中所有具有互模拟关系的节点归为一类,在 G'中使用一个节点来表示,图 8(b)中给出图 G 以及根据图模式查询转换后的图 G'.图 G 中 C_1 与 C_2 显然为互模拟关系;由于 $out(B_1)$和 $out(B_2)$分别为$\{C_1\}$和$\{C_2\}$,故 B_1 与 B_2 也为互模拟关系;$out(A_2)$和 $out(A_3)$分别为$\{B_1,C_1\}$和$\{B_2,C_1\}$,相同的节点自然满足互模拟关系,所以 A_2 与 A_3 为互模拟关系;但是 $out(A_1)$为$\{B_1\}$,$out(A_2)$和 $out(A_3)$中的 C_1 无法在 $out(A_1)$中找节点和其形成互模拟关系,故 A_1 无法与其他节点形成互模拟关系.

文献[44]对以上两种转换方式的正确性给出了严格的证明,并说明这种压缩方法与以往的压缩方法存在的不同之处在于,已有的压缩技术都可以和该算法同时使用,因为可以使用这些已有的压缩技术对转换后的图 G'进行压缩.实验结果表明,根据可达性查询进行转换后的图相对于原图规模大约减小 95%,查询时间相对于原图大约减小 98%,根据图匹配查询进行转换后的图相对于原图规模大约减小 57%,查询时间相对于原图大约减小 70%.

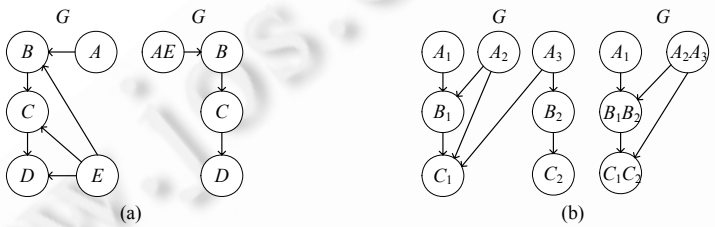


Fig.8 Graph conversion methods for reachability query and pattern matching query

图 8 面向可达性查询和图模式查询的图转换方法

除了网页图和社交网络,越来越多的新领域使用图来描述和分析数据,例如语义挖掘(semantic mining)中的语义网(semantic Web)[58],以及生物信息学中的转录网络(transcription network)[59]、蛋白质相互作用网络(protein-protein interaction network)[60]和代谢网络(metabolic network)[61]等.这些新的应用场景对图数据进行分析时产生了许多新的并且比较复杂的查询方式,不同的查询方式之间的差异也比较大,对于这些需求通常需要设计有针对性的压缩方法以达到较好的压缩效果.文献[62]对语义挖掘中两种典型的应用场景设计了两种压缩策略:等效压缩(equivalent compression)和依赖压缩(dependent compression),取得了较好的效果.文献[63]根据转录网络中节点之间的相互关系,将多个节点转化为一个强节点(power node),多条边转化为一条强边(power node),减少了图中节点和边的数量,可以有效节约存储空间.

6 图数据压缩技术比较与未来发展

6.1 图数据压缩技术比较

在实际使用中,对图数据进行压缩时算法在压缩率和查询时间方面表现出不同效果,本节根据公开的真实数据集测试上述代表性算法,并分析、对比其性能.

表 2 列举了测试中使用的真实网页图数据和社交网络数据,其中 LJ-1 数据来自于斯坦福大学 SNAP[64](Stanford Network Analysis Package Project),其余数据均来自于米兰大学 LAW[65](Laboratory for Web Algorithmics).表 2 分别描述了这些数据集的节点数、边数、节点数与边数的比值和该数据集在其网站的文件名.表 3 和表 4 分别列举了不同算法对网页图数据和社交网络数据的压缩率和查询时间.对于可以获得源代码的算法,列举了在统一的实验环境下进行的测试结果,其中 K^2-tree,Re-Pair,LZ78,K^2-Partitioned 和 MP_k 算法的源代码来自智利大学 FCWGR[66](Fast Compact Web Graph Representations Project),BV 和 FRS 算法的源代码来自 WebGraph 框架[67](WebGraph framework),BFS 算法的源代码来自提出该算法的文献[36]作者 Guido Drovandi 的个人网站[68],算法中的参数均使用在压缩和访问之间取得较好折中时的配置.测试使用的实验环境为 Red Hat

Enterprise Linux 6.0 Server 64 位操作系统,系统配置为 Intel(R) Core(TM) i7-3820 处理器,32GB 内存.对于无法获得源代码的算法,列举了相关文献中在相同数据集下的测试结果,VNM 算法的测试结果来自文献[33],AMN 算法的测试结果来自文献[34],DSM 算法的测试结果来自文献[69].表 3 中的"–"表示无法获得该算法在当前数据下的测试结果.

Table 2 Description of testing real graphs

表 2 测试使用的真实数据集

分类	数据集	节点数	边数	边数/节点数	文件名
网页图	cnr	325 577	3 216 152	9.88	cnr-2000
	in	1 382 908	16 917 053	12.23	in-2004
	eu	862 646	19 235 140	22.30	eu-2005
	uk	18 520 487	298 113 762	16.10	uk-2002
社交网络	enron	69 244	276 143	3.99	enron
	dblp	986 324	6 707 236	6.80	dblp-2011
	LJ-1	4 847 571	68 993 773	14.23	soc-LiveJournal1
	LJ-2	5 363 260	79 023 142	14.73	LiveJournal-2008

Table 3 Comparison of different compression algorithms over Web graph

表 3 不同算法对网页图的压缩效果对比

数据集	压缩率(bpe)				查询时间(μs)			
	cnr	in	eu	uk	cnr	in	eu	uk
K^2-tree	2.68	1.23	3.22	2.09	1.9	1.7	3.1	7.8
Re-Pair	3.02	2.20	3.93	2.70	1.3	1.1	2.5	3.2
LZ78	10.35	7.02	12.97	9.86	0.7	0.5	1.4	2.1
BV	2.53	2.04	5.62	2.44	2.7	2.3	3.6	4.2
VNM	–	–	2.90	1.95	–	–	3.8	4.5
AMN	1.99	1.71	2.78	–	2.43	2.38	28.72	–
K^2-Partitioned	3.53	2.11	4.00	3.12	1.2	1.0	1.6	2.3

Table 4 Comparison of different compression algorithms over social network

表 4 不同算法对社交网络的压缩效果对比

数据集	压缩率(bpe)				查询时间(μs)			
	enron	dblp	LJ-1	LJ-2	enron	dblp	LJ-1	LJ-2
K^2-tree	13.78	11.73	17.24	17.19	6.5	5.9	14.62	15.81
Re-Pair	16.28	14.84	22.53	20.66	4.2	4.5	6.9	7.3
LZ78	20.33	18.29	25.91	24.20	4.0	4.3	6.4	7.1
FRS	14.36	13.48	15.73	15.82	2.9	2.4	3.7	4.4
BFS	13.98	11.23	17.69	17.84	15.8	6.4	36.4	40.8
DSM	10.07	8.41	13.02	13.04	9.5	7.6	11.8	11.3
MP_k	17.02	8.48	13.25	13.35	8.5	8.0	12.9	11.0

由于未能获得 QPGC 算法的源代码,并且也未能在相关文献中找到该算法使用表 2 中数据的测试结果,故未列举该算法与其他算法在相同测试数据下的性能对比.提出该算法的文献[44]分别使用网页图、社交网络、论文引证关系图(citation map)和 P2P 网络(peer-to-peer network)等多组图数据测试该算法在进行可达性查询和图匹配查询时的压缩率和查询时间.文献[44]中的实验表明根据可达性查询进行转换后的图所占的存储空间平均相比原图空间减少约 95%,同时查询时间减少约 98%,根据图匹配查询进行转换后的图所占的存储空间相对于原图规模减小约 57%,同时查询时间减少约 70%,对于同一类型的测试数据,压缩后的查询时间通常随着图数据的规模增加而变慢.

通过对比不同算法表现出的压缩率与查询时间,对每类技术进行总结分析得到如下结论:

(1) 基于传统存储结构的压缩技术不需要对图中的节点进行排序,K^2 树算法的压缩效果好于 Re-Pair 算法和 LZ78 算法,但 Re-Pair 算法和 LZ78 算法的查询时间较快.K^2 树算法不但可以支持外邻查询还可以支持内邻查询,而 Re-Pair 算法和 LZ78 算法只能支持外邻查询.Re-Pair 算法和 LZ78 算法支持局部解压缩,压缩后的数据适于存放在外部磁盘上.

(2) 网页图压缩技术中,BV 可以调节压缩率和查询时间的关系,在压缩率和查询时间方面都表现出较好的

性能.当图中包含较多完全二部图时,使用 VMN 算法对图数据进行处理后,压缩率可以进一步提高.AMN 算法的压缩率较高,但是初始化过程复杂,规模较大的图需要的预处理时间较长,对有些数据集查询时间较长.K^2-Partitioned 算法的预处理过程简单,可以较快地进行数据压缩,同时查询时间较短.

(3) 由于社交网络与网页图数据特点的不同,对社交网络进行压缩的效果差于对网页图的压缩效果.在社交网络图压缩技术中,FRS 算法对社交网络中节点排序后,使用 BV 算法进行压缩,在压缩率和查询时间两方面的性能都较好.DSM 算法在图中包含边较多的稠密子图时压缩率较高,但是查询时间慢于 FRS 算法.

(4) 面向特定查询的图压缩技术中,QPGC 会改变图的结构,并且改变之后的图无法还原,只有对于特定的查询比如可达性查询和图模式查询,才能使用 QPGC.而 MP_k 不改变图的结构,当外邻查询和内邻查询的速度同时要求较高时比较适合使用这种方法.

6.2 未来发展

对处理图数据时面临的存储空间过大的问题,可以通过外部存储器存储、分布式处理以及使用构造支持查询的压缩结构等方式来解决,但是由于磁盘访问速度以及图数据的耦合性较强等原因导致前两种方案容易产生查询延时较大等问题,而采用第 3 种方案即图数据表示与技术可以在保证查询速度的同时缩减存储空间.

本文分别从基于传统存储结构的压缩技术、网页图压缩技术、社交网络图压缩技术和面向特定查询的图压缩技术 4 个方面入手,分析了图数据压缩技术的研究现状并详细介绍了不同方面在实际应用中取得效果较好的典型算法.通过分析发现,针对不同的图数据,采用不同的技术或者多种技术相结合才能产生较好的效果,通常先将图数据进行预处理,减少图中的数据冗余或者调整图的结构,再对处理后的图采用合适的压缩结构,如果有特殊的查询需求,还可以针对这种需求进行优化.

对网页图进行压缩的研究已经比较成熟,在压缩率和查询时间两方面都取得了较好的效果.现阶段研究比较火热的是针对社交网络进行压缩的技术,未来更多的研究可能会采用模式识别以及机器学习等方面的技术来更好地分析社交网络中的信息冗余,以实现更好的压缩效果.面向特定查询的图压缩技术最近才引起人们的关注,但是随着图数据应用越来越广,会产生许多特殊的查询方式,对于这些查询,传统的方法可能效果并不理想,而采用特殊的处理方式则会产生非常好的效果,所以对该技术的研究也会更加受到人们的重视.

References:

[1] The Graph 500 List. http://www.graph500.org

[2] Randall KH, Stata R, Wickremesinghe RG, Wiener, JL. The link database: Fast access to graphs of the Web. In: Proc. of the 2002 Data Compression Conf. (DCC). Piscataway: IEEE, 2002. 122−131. [doi: 10.1109/DCC.2002.999950]

[3] Kumar R, Novak J, Tomkins A. Structure and evolution of online social networks. In: Proc. of the 12th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining (KDD). New York: ACM, 2006. 611−617. [doi: 10.1145/1150402.1150476]

[4] Perugini S, Gonçalves MA, Fox EA. Recommender systems research: A connection-centric survey. Journal of Intelligent Information Systems, 2004,23(2):107−143. [doi: 10.1023/B:JIIS.0000039532.05533.99]

[5] Brin S, Page L. The anatomy of a large-scale hypertextual Web search engine. Computer Networks and ISDN Systems, 1998,30(1): 107−117. [doi: 10.1016/S0169-7552(98)00110-X]

[6] Kleinberg JM. Authoritative sources in a hyperlinked environment. Journal of the ACM (JACM), 1999,46(5):604−632. [doi: 10.1145/324133.324140]

[7] Saito K, Kimura M, Ohara K, Motoda H. Efficient discovery of influential nodes for SIS models in social networks. Knowledge and Information Systems, 2012,30(3):613−635. [doi: 10.1007/s10115-011-0396-2]

[8] Zhuge H. Communities and emerging semantics in semantic link network: Discovery and learning. IEEE Trans. on Knowledge and Data Engineering, 2009,21(6):785−799. [doi: 10.1109/TKDE.2008.141]

[9] Cha M, Mislove A, Gummadi KP. A measurement-driven analysis of information propagation in the flickr social network. In: Proc. of the 18th Int'l Conf. on World Wide Web (WWW). New York: ACM, 2009. 721−730. [doi: 10.1145/1526709.1526806]

[10] Musiał K, Kazienko P, Bródka P. User position measures in social networks. In: Proc. of the 3rd Workshop on Social Network Mining and Analysis. New York: ACM, 2009. 1−9. [doi: 10.1145/1731011.1731017]

[11] Mislove A, Marcon M, Gummadi KP, Druschel P, Bhattacharjee B. Measurement and analysis of online social networks. In: Proc. of the 7th ACM SIGCOMM Conf. on Internet Measurement (IMC). New York: ACM, 2007. 29−42. [doi: 10.1145/1298306.1298311]

[12] Saito H, Toyoda M, Kitsuregawa M, Aihara K. A large-scale study of link SPAM detection by graph algorithms. In: Proc. of the 3rd Int'l Workshop on Adversarial Information Retrieval on the Web. New York: ACM, 2007. 45−48. [doi: 10.1145/1244408.1244417]

[13] Donato D, Laura L, Leonardi S, Millozzi S. Algorithms and experiments for the Webgraph. Journal of Graph Algorithms and Applications, 2006,10(2):219−236. [doi: 10.7155/jgaa.00125]

[14] GlobalWebIndex. 2013. http://www.globalwebindex.net/products/report/stream-report-social-q1-2013

[15] China Internet Network Information Center. 2012. http://www.cnnic.net.cn/research/bgxz/tjbg/201201/t20120116_23668.html

[16] Vitter JS. External memory algorithms and data structures: Dealing with massive data. ACM Computing Surveys (CsUR), 2001, 33(2):209−271. [doi: 10.1145/384192.384193]

[17] Vitter JS. Algorithms and data structures for external memory. Foundations and Trends in Theoretical Computer Science, 2008, 2(4):305−474. [doi: 10.1561/0400000014]

[18] Badue C, Ribeiro-Neto B, Baeza-Yates R, Ziviani N. Distributed query processing using partitioned inverted files. In: Proc. of the 8th Symp. on String Processing and Information Retrieval (SPIRE). Piscataway: IEEE, 2001. 10−20. [doi: 10.1109/SPIRE.2001.989733]

[19] Tomasic A, Garcia-Molina H. Query processing and inverted indices in shared-nothing text document information retrieval systems. The Int'l Journal on Very Large Data Bases-Parallelism in Database Systems, 1993,2(3):8−17. [doi: 10.1007/BF01228671]

[20] Yu G, Gu Y, Bao YB, Wang ZG. Large scale graph data processing on cloud computing environments. Chinese Journal of Computers, 2011,34(10):1753−1767 (in Chinese with English abstract). [doi: 10.3724/SP.J.1016.2011.01753]

[21] Boldi P, Vigna S. The Webgraph Framework I: Compression techniques. In: Proc. of the 13th Int'l Conf. on World Wide Web (WWW). New York: ACM, 2004. 595−602. [doi: 10.1145/988672.988752]

[22] Boldi P, Vigna S. The WebGraph Framework II: Codes for the World-Wide Web. In: Proc. of the Conf. on Data Compression (DCC). Piscataway: IEEE, 2004. [doi: 10.1109/DCC.2004.1281504]

[23] Jacobson G. Space-Efficient static trees and graphs. In: Proc. of the 30th Annual Symp. on Foundations of Computer Science. Piscataway: IEEE, 1989. 549−554. [doi: 10.1109/SFCS.1989.63533]

[24] Munro JI, Raman V. Succinct representation of balanced parentheses, static trees and planar graphs. In: Proc. of the 54th Annual Symp. on Foundations of Computer Science. Piscataway: IEEE, 1997. 118−126. [doi: 10.1109/SFCS.1997.646100]

[25] Chuang RC, Garg A, He X, Kao M, Lu H. Compact encodings of planar graphs via canonical orderings and multiple parentheses. Automata, Languages and Programming, 1998,1443(1):118−129. [doi: 10.1007/BFb0055046]

[26] Deo N, Litow B. A structural approach to graph compression. In: Proc. of the 23th MFCS Workshop on Communications. 1998. 91−101.

[27] He X, Kao M, Lu H. A fast general methodology for information-theoretically optimal encodings of graphs. SIAM Journal on Computing, 2000,30(3):838−846. [doi: 10.1137/S0097539799359117]

[28] Chakrabarti D, Papadimitriou S, Modha DS, Faloutsos C. Fully automatic cross-associations. In: Proc. of the 10th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining (KDD). New York: ACM, 2004. 79−88. [doi: 10.1145/1014052.1014064]

[29] Adler M, Mitzenmacher M. Towards compressing web graphs. In: Proc. of the Conf. on Data Compression (DCC). Piscataway: IEEE, 2001. 203−212. [doi: 10.1109/DCC.2001.917151]

[30] Boldi P, Santini M, Vigna S. A large time-aware Web graph. ACM SIGIR Forum, 2008,42(2):33−38. [doi: 10.1145/1480506.1480511]

[31] Boldi P, Santini M, Vigna S. Permuting Web graphs. In: Algorithms and Models for the Web-Graph. Heidelberg: Springer-Verlag, 2009. 116−126. [doi: 10.1007/978-3-540-95995-3_10]

[32] Boldi P, Rosa M, Santini M, Vigna S. Layered label propagation: A multiresolution coordinate-free ordering for compressing social networks. In: Proc. of the 20th Int'l Conf. on World Wide Web (WWW). New York: ACM, 2011. 587−596. [doi: 10.1145/1963405.1963488]

[33] Buehrer G, Chellapilla K. A scalable pattern mining approach to Web graph compression with communities. In: Proc. of the 2008 Int'l Conf. on Web Search and Data Mining (WSDM). New York: ACM, 2008. 95−106. [doi: 10.1145/1341531.1341547]

[34] Asano Y, Miyawaki Y, Nishizeki T. Computing and Combinatorics. Heidelberg: Springer-Verlag, 2008. 1−11. [doi: 10.1007/978-3-540-69733-6_1]

[35] Chierichetti F, Kumar R, Lattanzi S, Mitzenmacher M, Panconesi A, Raghavan P. On compressing social networks. In: Proc. of the 15th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining (KDD). New York: ACM, 2009. 219−228. [doi: 10.1145/1557019.1557049]

[36] Apostolico A, Drovandi G. Graph compression by BFS. Algorithms, 2009,2(3):1031−1044. [doi: 10.3390/a2031031]

[37] Hernández C, Navarro G. Compression of Web and social graphs supporting neighbor and community queries. In: Proc. of the 6th ACM Workshop on Social Network Mining and Analysis (SNAKDD). New York: ACM, 2011. 1−10.

[38] Hernández C, Navarro G. Compressed representation of Web and social networks via dense subgraphs. In: Proc. of 19th Int'l Symp. on String Processing and Information Retrieval (SPIRE). Heidelberg: Springer-Verlag, 2012. 264−276. [doi: 10.1007/978-3-642-34109-0_28]

[39] Brisaboa N R, Ladra S, Navarro G. k^2-Trees for compact Web graph representation. In: Proc. of the 16th Int'l Symp. on String Processing and Information Retrieval (SPIRE). Heidelberg: Springer-Verlag, 2009. 18−30. [doi: 10.1007/978-3-642-03784-9_3]

[40] Claude F, Navarro G. A Fast and compact Web graph representations. ACM Trans. on the Web (TWEB), 2010,4(4):1−31. [doi: 10.1145/1841909.1841913]

[41] Larsson NJ, Moffat A. Off-Line dictionary-based compression. Proc. of the IEEE, 2000,88(11):1722−1732. [doi: 10.1109/5.892708]

[42] Ziv J, Lempel A. Compression of individual sequences via variable-rate coding. IEEE Trans. on Information Theory, 1978,24(5):530−536. [doi: 10.1109/TIT.1978.1055934]

[43] Maserrat H, Pei J. Neighbor query friendly compression of social networks. In: Proc. of the 16th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining (KDD). New York: ACM, 2010. 533−542. [doi: 10.1145/1835804.1835873]

[44] Fan W, Li J, Wang X, Wu Y. Query preserving graph compression. In: Proc. of the 2012 ACM SIGMOD Int'l Conf. on Management of Data (SIGMOD). New York: ACM, 2012. 157−168. [doi: 10.1145/2213836.2213855]

[45] González R, Grabowski S, Mäkinen V, Navarro G. Practical implementation of rank and select queries. In: Proc. of the 4th Workshop on Efficient and Experimental Algorithms (WEA). CTI Press and Ellinika Grammata, 2005. 27−38.

[46] Brisaboa NR, Ladra S, Navarro G. Compact representation of Web graphs with extended functionality. Information Systems, 2014, 39:152−174. [doi: 10.1016/j.is.2013.08.003]

[47] Brisaboa NR, Ladra S, Navarro G. DACs: Bringing direct access to variable-length codes. Information Processing and Management: an International Journal, 2013,49(1):392−404. [doi: 10.1016/j.ipm.2012.08.003]

[48] Claude F, Navarro G. Extended compact Web graph representations. In: Algorithms and Applications. Heidelberg: Springer-Verlag, 2010. 77−91. [doi: 10.1007/978-3-642-12476-1_5]

[49] He M, Munro JI. Succinct representations of dynamic strings. In: Proc. of the 17th Int'l Symp. on String Processing and Information Retrieval (SPIRE). Heidelberg: Springer-Verlag, 2010. 334−346. [doi: 10.1007/978-3-642-16321-0_35]

[50] Feder T, Motwani R. Clique partitions, graph compression and speeding-up algorithms. In: Proc. of the 23rd Annual ACM Symp. on Theory of Computing. New York: ACM, 1991. 123−133. [doi: 10.1145/103418.103424]

[51] Claude F, Ladra S. Practical representations for Web and social graphs. In: Proc. of the 20th ACM Int'l Conf. on Information and Knowledge Management (CIKM). New York: ACM, 2011. 1185−1190. [doi: 10.1145/2063576.2063747]

[52] Garey MR, Johnson DS, Stockmeyer L. Some simplified NP-complete problems. In: Proc. of the 6th Annual ACM Symp. on Theory of Computing. New York: ACM, 1974. 47−63. [doi: 10.1145/800119.803884]

[53] Broder AZ, Charikar M, Frieze AM, Mitzenmacher M. Min-Wise independent permutations. Journal of Computer and System Sciences, 2000,60(3):630−659. [doi: 10.1006/jcss.1999.1690]

[54] Shi Q, Xiao Y, Bessis N, Lu Y, Chen Y, Hill R. Optimizing K^2 trees: A case for validating the maturity of network of practices. Computers & Mathematics with Applications, 2012,63(2):427−436. [doi: 10.1016/j.camwa.2011.07.060]

[55] Raman R, Raman V, Rao SS. Succinct indexable dictionaries with applications to encoding k-ary trees and multisets. In: Proc. of the 30th Annual ACM-SIAM Symp. on Discrete Algorithms (SODA). Philadelphia: Society for Industrial and Applied Mathematics, 2002. 233−242.

[56] Grossi R, Gupta A, Vitter JS. High-Order entropy-compressed text indexes. In: Proc. of the 14th Annual ACM-SIAM Symp. on Discrete Algorithms. Philadelphia: Society for Industrial and Applied Mathematics, 2003. 841−850.

[57] Dovier A, Piazza C, Policriti A. A fast bisimulation algorithm. In: Proc. of the 13th Int'l Conf., Computer Aided Verification. Heidelberg: Springer-Verlag, 2001. 79−90. [doi: 10.1007/3-540-44585-4_8]

[58] Zhang X, Zhao C, Wang P, Zhou F. Mining link patterns in linked data. In: Web-Age Information Management. Heidelberg: Springer-Verlag, 2012. 83−94. [doi: 10.1007/978-3-642-32281-5_9]

[59] Babu MM, Teichmann SA. Evolution of transcription factors and the gene regulatory network in Escherichia coli. Nucleic Acids Research, 2003,31(4):1234−1244. [doi: 10.1093/nar/gkg210]

[60] Jeong H, Mason SP, Barabási AL, Oltvai ZN. Lethality and centrality in protein networks. Nature, 2001,411(6833):41−42. [doi: 10.1038/35075138]

[61] Ravasz E, Somera AL, Mongru DA, Oltvai ZN, Barabási AL. Hierarchical organization of modularity in metabolic networks. Science, 2002,297(5586):1551−1555. [doi: 10.1126/science.1073374]

[62] Jiang X, Zhang X, Gao F, Pu C, Wang P. Graph Compression Strategies for Instance-Focused Semantic Mining, Linked Data and Knowledge Graph. Heidelberg: Springer-Verlag, 2013. 50−61. [doi: 10.1007/978-3-642-54025-7_5]

[63] Ahnert SE. Power graph compression reveals dominant relationships in genetic transcription networks. Molecular BioSystems, 2013,9(11):2681−2685. [doi: 10.1039/C3MB70236G]

[64] Stanford Large Network Dataset Collection. 2009. http://snap.stanford.edu/data

[65] Laboratory for Web Algorithmics. 2011. http://law.di.unimi.it/datasets.php

[66] WebGraphs on recoded.cl. 2010. http://webgraphs.recoded.cl/

[67] WebGraph homepage. 2009. http://webgraph.di.unimi.it/

[68] Drovandi G. PhD Web Site. 2012. http://www.dia.uniroma3.it/~drovandi/software.php

[69] Hernandez C, Navarro G. Compressed representations for Web and social graphs. Knowledge and Information Systems, 2013, 40(1):1−35. [doi: 10.1007/s10115-013-0648-4]

附中文参考文献:

[20] 于戈,谷峪,鲍玉斌,王志刚.云计算环境下的大规模图数据处理技术.计算机学报,2011,34(10):1753−1767. [doi: 10.3724/SP.J.1016.2011.01753]

张宇(1987－),男,河南新乡人,博士生,CCF学生会员,主要研究领域为图数据管理,算法设计.

E-mail: zhangyu@iie.ac.cn

贾焰(1960－),女,教授,博士生导师,CCF高级会员,主要研究领域为分布式计算,超大规模数据库,并行数据库.

E-mail: jiayanjy@vip.sina.com

刘燕兵(1981－),男,副研究员,CCF 会员,主要研究领域为模式匹配算法,图数据计算,信息内容安全.

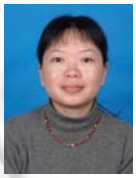
E-mail: liuyanbing@iie.ac.cn

刘萍(1972－),女,副研究员,CCF 会员,主要研究领域为信息安全,数据流处理,字符串匹配算法.

E-mail: liuping@iie.ac.cn

熊刚(1977－),男,博士,高级工程师,博士生导师,主要研究领域为网络测量,网络攻防与信息对抗,信息安全.

E-mail: xionggang@iie.ac.cn



郭莉(1969－),女,高级工程师,博士生导师,CCF 高级会员,主要研究领域为信息安全,数据流处理.

E-mail: guoli@iie.ac.cn