位图索引的设计与实现*

许向阳　李明胜

(华中科技大学计算机学院　武汉　430074)

摘　要: 文章在分析了几种现有位图索引的基础上, 为国产数据库系统 DM 设计了分段范围编码位图索引。最后介绍了 DM 位图索引的建立以及查询方法。

关键词: 位图索引　分段编码　数据库管理系统

Bitmap Index Design and Implementation

XU Xiangyang, LI Mingsheng

(Department of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, 430074, China)

Abstract: Bitmap index is an index technology which support bit operation. It is an efficient index for OLAP database. After analyzing several existing bitmap indexes, we design segmented range encoded bitmap index on DM DBMS. The creation and query method of DM bitmap index were introduced at last.

Keywords: Bitmap Index, Segmented, Encoding, DBMS

位图索引是适合 OLAP 数据库的索引结构, 是提高数据库系统性能的重要措施。Oracle、Sysbase、Informix 等主流数据库都支持位图索引[1]。

DM DBMS 是华中科技大学计算机学院研制的具有自主知识产权的数据库管理系统。为了进一步扩大 DM 的应用范围, 提高 DM 的综合性能, 我们在 DM 的基础上展开位图索引的研究。

1　位图索引编码

位图索引编码有两种方式, 等值编码和范围编码[2,3]。以它们为基础可以衍变出多种编码方法。为了便于介绍, 我们作如下设定: 有 n 个元组的表 $T=\{t_1, t_2, \cdots, t_n\}$; A 表示 T 的一个列, A 列上有 m 个不同值, 不失一般性, 我们假设 m 个值是从 0 到 $m-1$ 的连续整数; B 表示一个长度为 n 的位向量 $(b_1, b_2, \cdots, b_n)$, 当中任一位 b_i 也记作 B[i]。

1.1　等值编码位图索引

等值编码位图索引是最基本的位图索引。它由 m 个位向量$\{B_{m-1}, \cdots B_j \cdots B_1, B_0\}$ 组成, 也可以看作一个 $n*m$ 的矩阵, B_j 是对应于属性值 j 的位向量, 当且仅当 $t_i.A=j$ 时, $B_j[i]=1$; 否则, $B_j[i]=0$。图 1(a) 表示一个有 10 个元组的表在 A 列上的投影($m=9$), 图 1(b) 为 A 列上的等值编码位图索引, 每一列代表一个位向量。

1.2　范围编码位图索引

范围编码位图索引由 $m-1$ 个位向量$\{B_{m-2}, \cdots B_j, \cdots B_1, B_0\}$ 组成, 其中 B_j 对应于属性值 j, 当且仅当 $t_i.A \leqslant j$ 时, $B_j[i]=1$。范围编码索引可以看作是等值编码位图索引的累积形式, 在等值编码位图索引中, 每一个位图向量对应于一个索引值, 而范围编码位图索引中, 每一位图向量代表所有小于等于对应值的索引值。图 1(c) 表示 A 上的范围编码位图索引。

1.3　分段位图索引

分段位图索引由 Chan 和 Ioannidis 提出[4], 它将属性值用 r 进制表示。在 r 进制中, 0 到 $m-1$ 之间的每一个值 x 可以用长度为 e 的数字序列($x_{e-1}, \cdots, x_1, x_0$) 表示, 其中 $e=\lceil \log_r^m \rceil$、$0 \leqslant x_e, \cdots, x_1, x_0 < r$、$x = x_{e-1} * r^{e-1} + \cdots, + x_1 * r + x_0$。$e$ 位数字序列可以看成是 e 个列的值组成的, 相应的 A 列可划分成 e 个子列 $A_{e-1}, \cdots A_1, A_0$。分段位图索引就是分别为每一个子列建立位图索引, 子列索引称为称为元索引, r 为索引的基。对于每一个元索引可以使用等值编码位图索引或范围编码位图索引。对于图 1(a) 中的数据, A 列可以分裂为 A_1、A_0 两个子列, 如图 1(d) 所示。图 1(e) 中的(1) (2) 分别表示 A_1、A_0 的等值编码位图索引, 其中列 B_p^k 表示第 k 个元索引中对应于值 p 的位向量。图 1(f) 表示 A 列上以 3 为基的范围编码位图索引。

本文于 2004 - 10 - 08 收到。

*基金项目: 国家 863 计划信息领域数据库重大专项资金项目(2002AA4Z3110)。

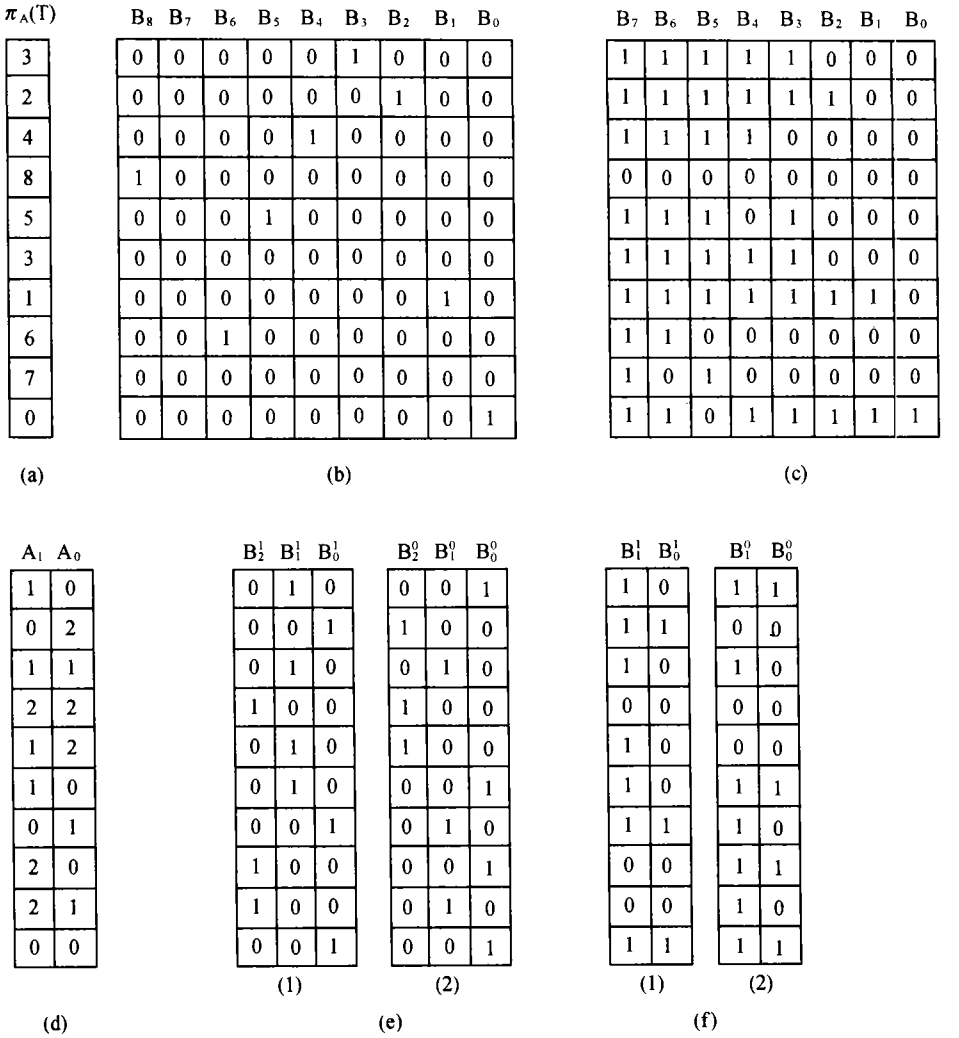

图 1　位图索引

(a) 索引列投影影　(b) 等值编码索引　(c) 范围编码索引

(d) A 列以 3 为基分裂成 A_1A_0 两列　(e) 分段等值编码位图索引　(f) 分段范围编码位图索引

1.4　几种位图索引的比较

等值编码位图索引和范围编码位图索引在一般情况下具有较好的查询效率, 在执行等值查询时, 等值编码位图索引只需要使用一个位向量, 而范围编码位索引也只需要两个位图向量。但是当 m 很大且 x 也很大的情况下, 等值编码位图索引的范围查询代价很大, 需要访问的位向量个数与查询范围成线性关系。另一方面, 随着列中不同值个数 m 的增加, 等值编码位图索引和范围编码位图索引所需要的空间越来越多, 当 m 超过元组长度的 8 倍的时, 存储位图索引的空间比数据空间还大, 严重影响了它们的可用性。分段位图索引有很强的灵活性。通过调节 r 的大小, 容易取得空间效率和时间效率上的平衡点。加大 r 可以减少查询位图向量访问量, 但增加存储开销, 反之, 查询位图向量访问量变大, 存储开销变小。当 $r= m$ 时, 就转化为不分段的位图向量。表 1 给出了几种编码的比较。

表 **1**　几种位图索引的性能对比

索引名称	占用空间 (向量数)	A= x 的查询代价 (访问向量数)	A≤x 的查询代价 (访问向量数)
等值编码位图索引	m	1	x
范围编码位图索引	$m-1$	2	1
以 r 为基的等值编码位图索引	$e*r$	e	$x_{e-1}+\cdots+x_1+x_0$
以 r 为基的范围编码位图索引	$e*(r-1)$	$e*2$	$2*(e-1)+1$

注: 分段情况下 x 分解为序列 $x_{e-1}\cdots, x_1x_0$

2 DM 的位图索引

在DM 中, 我们采用了分段的范围编码位图索引, 采用范围编码是因为它性能比较稳定, 虽然在等值查询的时候性能比不上等值编码位图索引, 但是在范围查询时却比等值编码快得多。DM 的位图索引主要由四个部分组成: 索引头块、映射表、位图向量组、物理地址索引表。每个部分又由一个和多个大小固定的块构成, 其结构如图 2。索引头块是位图索引的根, 存储位图索引的基本信息, 如位图索引的基 r、元索引数 e、最大索引值 $m-1$, 以及指向位图索引其他结构的指针。

映射表是一个从实际索引值到内部表示值的映射, 采用 B^+ 树存储。映射表的每个节点中除了含有实际索引值、内部表示值外, 还包括第一个和最后一个出现该索引值的行号, 以及该索引值的行数, 这些信息可以帮助查询优化器进行代价估计, 同时也可以在查询执行时减少索引数据读入量。

位图向量组是位图索引的主体, 它由多个位向量组成, 在 DM 中, 每一个位向量单独存储在一个块链中, 块链由一个头块和向量体组成。在头块里存有向量体中所有块的地址, 便于随机访问。向量体是一个双向链表, 每一个数据块里面存储位向量的一段。

物理地址索引表是行号和行所在块的地址的对应表。行号是元组的顺序编号, 与位向量中的位依次对应, 由于元组物理地址可由块地址和块内序号确定, 故在索引表中只需要存放各数据块的首行行号和块地址的对应信息。

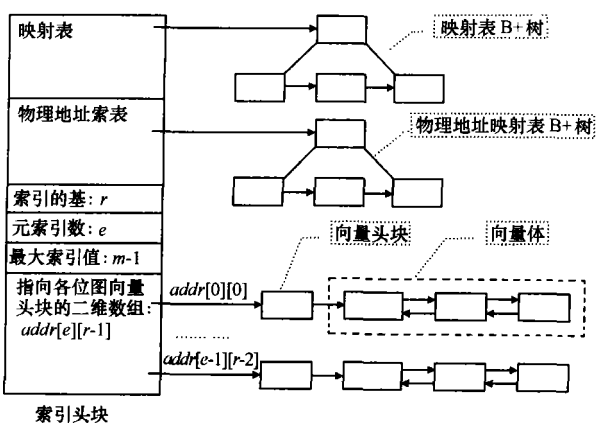

图 2 DM 位图索引的结构

3 DM 位图索引的实现

3.1 DM 位图索引的建立

DM 位图索引的建立分四步。①扫描全表, 建立物理地址索引 B+ 树和映射表的树结构。为了避免在建立位图向量时再次扫描全表, 在构造映射表时, 为每一个值建立其行号组块。②根据位图索引的基 r 和映射表中索引值的个数, 确定位图向量数目, 每一个位向量都一次性分配所需存储空间, 最大限度地保证其向量数据块物理地址的连续性, 以加快访问单个位向量的速度。③依次处理映射表中的各值, 根据行号组块构造分段等值编码位图索引(忽略各段中的最大值)。④在所有值处理完后, 采用逐级位图向量相加构造分段范围位图索引。

3.2 DM 位图索引的查询

DM 位图索引的查询分为两个阶段: 查询重写和查询执行。查询重写将对整个位图索引的查询分解为对各个元位图索引的查询, 并且形成一个逻辑操作树, 查询执行阶段根据生成的操作树, 读入位图索引, 计算结果。

在查询重写阶段, 首先通过查询映射表将查询谓词中的实际值转化为内部表示值。然后将查询谓词转换为一种只含有"="或"≤"以及非运算的标准形式。转换的原则是:

(a) $A>x \Leftrightarrow \overline{A\leqslant x}$ (b) $A\geqslant x \Leftrightarrow \overline{A\leqslant x-1}$ (c) $A<x \Leftrightarrow A\leqslant x-1$

最后将查询分布到各个索引段, 并构造位图向量运算表达式。对于基为 r, 元索引个数为 e 的位图索引, x 为 e 个数字的序列 $x_{e-1}\cdots, x_1x_0$, 等值查询 $A=x_{e-1}\cdots, x_1x_0$, 可以分解为对各元索引的查询的合取: $A_{e-1}=x_{e-1}$ AND…, AND $A_1=x_1$ AND $A_0=x_0$, 而 $A_j=x_j$ 又可以表示成 $A_j\leqslant x_j$ AND NOT $A_j\leqslant x_j-1$, 也即 $A_j\leqslant x_j$ XOR $A_j\leqslant x_j-1$。查询 $A\leqslant x_{e-1}\cdots, x_1x_0$ 可以转化为: $(A_{e-1}=x_{e-1}$ AND $A'\leqslant x_{e-2}\cdots, x_1x_0)$ OR $A_{e-1}\leqslant(x_{e-1}-1)$。类似的, 对 $A'\leqslant x_{e-2}\cdots, x_1x_0$ 进行转换, 直到查询分解到所有的段中。

图 3 给出了 $r=8$ 时, 查询 $A=421(645_8)$ 和 $A\leqslant 421$ 的操作树。为了便于执行, 我们分别用 ∧、∨、⊕来表示 AND、OR、XOR, 用 $A_{\leqslant j}$ 表示 $A\leqslant j$ 对应的位图向量, 将位图元算表达式转换成操作树。

在查询执行阶段, 对于等值查询可以利用映射表中保存的每个值的第一行行号和最后一行行号来减少向量块的读入, 操作树中的每一个位向量只需要取出映射表第一行行号和最后一行行号之间的块, 而不是所有向量块, 就可以得到结果。操作树的执行过程采用流水线方式, 即以位图块为单位, 首先每一个位向量取出各自的第一块, 然后操作树执行一次, 得到一个结果块, 之后分别取出第二块, 得到第二个结果块, 依次执行得到最终的结果向量。这种执行方法与自底向上完全计算出每个操作的结果相比, 可以利用仍在缓冲区的位图块, 避免树中多次出现的同一位图向量的换入换出, 有效的减少了 I/O。另外流水线操作还可以提高响应时间, 它的结果向量逐步产生, 而不是等到下层操作完全完成才开始计算。

4 结束语

本文介绍了位图索引的等值编码、范围编码以及分段索引, 分析比较了这几种位图索引的性能, 在此基础上, 设计了

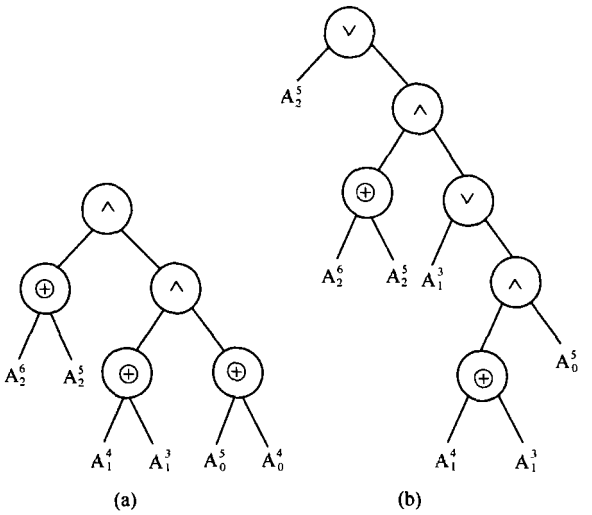

图 3　一个 $r=8$, $m=512$ 的位图索引

(a) A= 421 的操作树　(b) A≤421 的操作树

DM 位图索引的结构。文章最后介绍了 DM 位图索引的建立以及查询方法。

参考文献

1　Chee – Yong Chan, Yannis E Ioannidis. An Efficient Bitmap Encoding Scheme for Selection Query. In Proceedings of SIGMOD 1999. ACM Press, 1999.

2　HKT Wong, H – F Liu, et al. Bit Transposed Files. In Proceedings of the Intl Conference on Very Large Data Bases, pages 448 – 457, Stockhokm, 1985.

3　HKT Wong, JZ Li, et al. Bit Transposition for Very Large Scientific and Statistical Databases. Algorithmica, pages 289 – 309, 1986.

4　C Chan, YE Ioannidis. Bitmap Index Design and Evaluation. Proceedings ACM SIGMOD International Conference on Management of Data, June 1998, Seatle, Washington, USA.

作者简介

许向阳, 男, (1967 年生), 华中科技大学副教授, 博士, 主要研究方向为数据库系统查询优化。

李明胜, 男, (1980 年生), 华中科技大学计算机系研究生。

简报

Ipbtables 与 linux 透明代理

透明代理的意思是客户端根本不需要知道有代理服务器的存在, 其基本原理是代理服务器截取内网主机与外网通信, 由代理服务器本身完成与外网主机通信, 然后把结果传回给内网主机。在这个过程中, 无论内网主机还是外网主机都意识不到它们其实是在和代理服务器通信。而从外网只能看到代理服务器, 隐藏了内部网络, 提高了安全性。假设客户端向网站"http: // www. gdsspt. net"请求网页, 首先客户端向 DNS 请求 IP 地址解析, 代理服务器把这个请求传给 DNS 服务器, 并把结果返回给客房端。客房端根据 IP 地址向网站请求网页, 代理服务器把请求传给网站, 并把结果返回给客户端, 客户端的浏览器显示该页面。代理服务器不需要分析客户端的请求, 也不需要存储任何内容, 响应速度非常快。

在透明代理服务器上安装两块网卡, 把网络划分为两个区域: 内部网和外部网。对外提供服务的 Internet 服务器比如 WWW 服务器、FTP 服务器也放在内部网, 和局域网其它主机使用同一网段地址。Linux 透明代理器负责带动局域网内的用户主机访问 Internet, Internet 服务器也能给外部网提供服务, 并保护局域网内的用户主机和 Internet 服务器不受来自外部网的攻击。代理服务器有两块网卡 eth0 和 eth1, eth0 具有合法 IP 地址 218. 15. 56. 51, 与外部网络相连。与 eth1 相连的是内部网络, 内部网络除了用户主机外, 还有向外部网和内部网都提供服务的 WWW 服务器和 FTP 服务器。

对网卡 eth0 绑定 3 个 IP 地址, 218. 15. 56. 51、218. 15. 56. 52、218. 15. 56. 53, 子网掩码 255. 255. 255. 240, 网关地址 218. 15. 56. 63, DNS 服务器 202. 96. 128. 134。对网卡 eth1 也绑定 3 个 IP 地址, 192. 168. 1. 254、192. 168. 2. 254、192. 168. 3. 254, 可以满足 700 多用户主机通过透明代理上网。WWW 服务器的 IP 地址是 192. 168. 3. 251, FTP 服务器的 IP 地址是 192. 168. 3. 252, 子网掩码 255. 255. 255. 0, 网关地址 192. 168. 3. 254。

(1) 内部用户透明代理的 iptables 配置。由于内部的 IP 地址都不是合法的 IP 地址, 所以在出 eth0 之前, 用合法地址对数据包进行封装。用 kedit 编辑/etc/rc. d/rc. local 文件, 加上下面的命令行。

iptables – t nat – A POSTROUTING – o eth0 – s 192. 168. 1. 0/24 – j SNAT – – to – source 218. 15. 56. 51

(2) 内部 Internet 服务器反向透明代理的实现。外部网络与内部网络被透明代理服务器的两块网卡隔开, 外部网络只能访问到 eth0, 所以内部的 Internet 服务器必须反向代理到 eth0, 才能被外部网络访问。

(下转 202 页)