一种支持对象物理聚集的综合方法*

吴胜利 王能斌

(东南大学计算机系 南京 210096)

摘要 本文讨论了一种面向对象模型中同时支持按类查询和按复杂对象查询的新的对象物理聚集方法——综合法.该方法较常规聚集方法具有较佳的平均性能和最坏性能.

关键词 面向对象数据库,对象物理聚集,对象——关系数据库.

在面向对象数据库中,关于支持对象的物理聚集以提高其存取速度已有不少的讨论.[1~5]其中一些讨论[1,4]从有效支持复杂对象角度出发,探讨较佳的对象物理聚集方法.另一些则从更广的角度出发,假定已知系统中各对象(无视其类的差别)之间的互访可能性,从而形成所有对象为节点的图(各连接边标注互访概率或权重),在此基础上讨论最佳聚集方案.因这是一个NP完全问题,故一般只给出较佳答案.文献[2,5]代表了主要的2种方案.关于这2种方案和其它一些方案的性能模拟比较结果可在文献[6]中找到.而E. E. Chang则对聚集问题作了较全面的模拟实验.[3]总的说来,这些研究分别在理论和实现2方面取得了一些成果,对于支持面向对象数据库在一些领域(如CAD)的应用具有重要意义.但对类似关系数据库中的联想查询操作(如查找一个类中的所有对象)却未能给予有效支持,故进行这些操作的效率很低,这不能不说是一个缺点.

近年来,对面向对象数据库技术的研究已非常广泛、深入,一些学者认为面向对象数据库不应该撇开关系数据库,而只有将两者紧密结合为一体才能有效支持当今各种领域的数据管理和应用.[7]M. Stonebraker甚至提出了对象——关系数据库的概念.[8]本文作者对此亦深有同感.鉴于此,本文提出一种能同时支持常规关系操作(同类对象物理聚集)和复杂对象聚集(同一复杂对象的各组成部分物理聚集)的新方法,我们称之为综合法.这对于有效实现对象——关系数据库具有重要意义.

1 一个简单例子

下面是一个简单例子.

有4个复杂对象O_1,O_2,O_3和O_4如图1所示.其中每个复杂对象$O_i(1\leqslant i\leqslant 4)$均含有4个组成对象$a_i$,$b_i$,$c_i$和$d_i(1\leqslant i\leqslant 4)$,而它们分别属于相同的类$A$,$B$,$C$和$D$.假设一个物理

---

* 作者**吴胜利**,1963年生,博士生,主要研究领域为面向对象数据库系统等.**王能斌**,1929年生,教授,博士导师,主要研究领域为数据库及信息系统.

本文通讯联系人:吴胜利,南京 210096,东南大学计算机系

本文1995—04—07收到修改稿

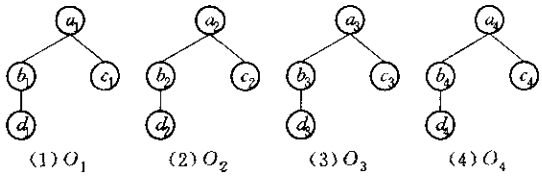

图1 4个对象 O_1, O_2, O_3 和 O_4

块可存放4个对象.这时有3种聚集方法:(1)按类聚集;(2)按复杂对象聚集;(3)综合法.其对象聚集分别如图2～图4所示.当查询某个类的所有对象时,方法1、3和2分别需读入1块、2块和4块.当查询某个复杂对象的全部组成对象时,方法2、3和1分别需读入1块、2块和4块.在此例中以下结论成立:

- 当查找某个类中所有对象时,方法1最优,3次之,2最劣;
- 当查找某个复杂对象的全部组成对象时,方法2最优,3次之,1最劣;
- 当按类查询和按复杂对象查询具有大致相当的概率时,同时考虑这两者,方法3具有最优的最坏性能(读2块)和平均性能(读2块),方法1和2相当,最坏情形需读4块,平均需读2.5块.

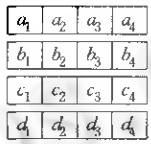

图2 按类聚集

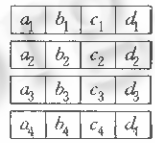

图3 按复杂对象聚集

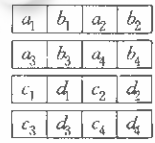

图4 综合法

在后面的讨论中,我们将看到这个结论是普遍成立的.

2 综合聚集法

2.1 基本情况

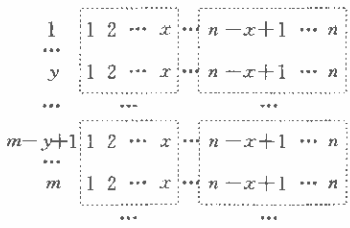

图5 综合聚集法分块示意图

假设有 n 个复杂对象 $O_1, O_2, \cdots, O_n$,每个复杂对象 $O_i(1 \leqslant i \leqslant n)$ 由 m 个对象组成,而这些组成对象分别属于 m 个类.所有对象大小相同,每个物理块(内存页)可存放 t 个对象,用户可按类或按复杂对象查询,问题是怎样在硬盘上存放对象才能使查询的平均性能和最坏性能最佳.不失一般性,可如图5所示方式存放.每个虚线框中的对象放在一物理块中.每块存放 y 个类中 x 个对象,亦即 $xy=t$,查询某个类的对象需读 $u=n/x$ 块,查询某个复杂对象需读 $v=m/y$ 块,则对这2类查询而言,最多需读 $\max(u,v)$ 块,平均需读 $(u+v)/2$ 块.根据上述条件,可得以下联立方程:

$$\begin{cases} t=x \cdot y \\ x \cdot u=n \\ y \cdot v=m \end{cases} \qquad \text{则}(u+v)/2=\frac{1}{2}\left(\frac{n}{x}+\frac{m}{y}\right)=\frac{1}{2}\left(\frac{n}{x}+\frac{mx}{t}\right)$$

$$\text{令 } f(x)=\frac{u+v}{2}$$

当 $x=\sqrt{\frac{tn}{m}}$ 时, $\quad \frac{df(x)}{dx}=\frac{n}{-2x^2}+\frac{m}{2t}=0$,

当 $x<\sqrt{\frac{tn}{m}}$ 时, $\quad \frac{df(x)}{dx}<0$,

当 $x>\sqrt{\frac{tn}{m}}$ 时, $\quad \frac{df(x)}{dx}>0$,

故 $x=\sqrt{\frac{tn}{m}}$ 为 $f(x)$ 的唯一极小值点.

此时 $$x=\sqrt{\frac{tn}{m}},y=\sqrt{\frac{tm}{n}},\frac{u+v}{2}=\max(u,v)=u=v=\sqrt{\frac{mn}{t}},$$

所以每块存放 $y=\sqrt{\frac{tm}{n}}$ 个类中 $x=\sqrt{\frac{tn}{m}}$ 个对象为最佳方案,此时最坏和平均性能都是 $\sqrt{\frac{mn}{t}}$ 次.

在以上讨论中需假定 $\sqrt{\frac{mn}{t}}$ 为整数,这在绝大多数情况下是不成立的. 若 $\sqrt{\frac{mn}{t}}$ 不为整数,则需在这一小节讨论的基础上找到较佳整数解. 详细情况下一节讨论.

2.2　算法及分析

若 $\sqrt{\frac{mn}{t}}$ 不为整数,则我们需寻找最佳整数解. 一种办法是比较所有可能点,还有一种办法是在最优解($x_0=\sqrt{\frac{nt}{m}},y_0=\sqrt{\frac{mt}{n}}$)附近寻找最佳解. 这 2 种办法分别由算法 1 和算法 2 描述.

算法 1. 输入:n,m,t,含义同前

输出:x,y,构成平均性能最佳整数解

$x=\sqrt{\frac{nt}{m}},y=\sqrt{\frac{mt}{n}}$,if x,y 均为整数 then end 算法 1

$u=+\infty,v=+\infty,x_1=1$

loop

$y_1=\left\lfloor\frac{t}{x_1}\right\rfloor,x_1=\left\lfloor\frac{t}{y_1}\right\rfloor,u_1=\left\lfloor\frac{n}{x}\right\rfloor,v_1=\left\lfloor\frac{m}{y}\right\rfloor$

if$(u_1+v_1)<(u+v)$ then $\{x=x_1,y=y_1,u=u_1,v=v_1\}$

$x_1=x_1+1$

if $x_1>t$ then exit

forever

end 算法 1

该算法主要包括一个执行 $2\lfloor\sqrt{t}\rfloor-1$ 或 $2\lfloor\sqrt{t}\rfloor$ 次的循环,故复杂性为 $O(t^{\frac{1}{2}})$. 该算法找出平均性能最优解,但不一定最坏性能最优. 因为这两者有时不等价,例如 $n=10,m=10,t=10$,则 $x=2,y=5$ 为平均性能最优解($u=5,v=2$),而 $x=3,y=3$ 为最坏性能最优解($u=v=4$). 可以很容易改写此算法找出最坏性能最优解.

算法 2. 输入:n,m,t,含义同前

输出:x,y,含义同前

$a=\sqrt{\frac{nt}{m}},b=\sqrt{\frac{mt}{n}}$

if a,b 均为整数 then $\{x=a,y=b$,end 算法 2$\}$

令 $a=a_1+a_2$,a_1 为整数,a_2 为小数

$b=b_1+b_2$,b_1 为整数,b_2 为小数

$ab=\max(a,b)$

L_0: if $a\leqslant b$ then $\{c_1=a_1,c_2=a_2,d_1=b_1,d_2=b_2\}$

else $\{c_1=b_1,c_2=b_2,d_1=a_1,d_2=a_2\}$

$i=1$

if $c_1=0$ then $\{t_1[i]=1, t_2[i]=t$, goto $L_2\}$

$t_1[i]=c_1, t_2[i]=\left\lfloor \frac{t}{t_1[i]} \right\rfloor, i=i+1$

if $t_1[i-1]=1$ or $t_2[i-1]>ab$ then goto L_1

$t_1[i]=t_1[i-1]-1, t_2[i]=\left\lfloor \frac{t}{t_1[i]} \right\rfloor, i=i+1$

L_1: $t_1[i]=c_1+1, t_2[i]=\left\lfloor \frac{t}{t_1[i]} \right\rfloor, t_1[i]=\left\lfloor \frac{t}{t_2[i]} \right\rfloor, i=i+1$

L_2: if $m<n$ then $\{p=m, m=n, n=p\}$

$uv=+\infty$

for $j=1$ to $i-1$ do

$uv_1=\left\lfloor \frac{n}{t_1[j]} \right\rfloor+\left\lfloor \frac{m}{t_2[j]} \right\rfloor$

if $uv_1<uv$ then $\{x=t_1[j], y=t_2[j], uv=uv_1\}$

end of do

if $a>b$ then $\{x_1=x, x=y, y=x_1\}$

end 算法 2

该算法在最优解近旁最多3个可能解中确定平均性能最优整数解,算法正确性的详细证明请参见附录.该算法的复杂性是$O(1)$.

根据以上2算法的输出,一般的块将存放$x \cdot y$个对象,这样的块我们称之为正常块,而有些存放不足$x \cdot y$个对象的块为边界块.$\left\lceil \frac{n}{x} \right\rceil \cdot \left\lceil \frac{m}{y} \right\rceil$为所需要的总块数.

若$\frac{n}{x}$和$\frac{m}{y}$均不为正整数,则有$\left\lceil \frac{n}{x} \right\rceil+\left\lceil \frac{m}{y} \right\rceil-1$个边界块.若$\frac{n}{x}$为正整数,$\frac{m}{y}$不为正整数,则有$\left\lceil \frac{m}{y} \right\rceil$个边界块.若$\frac{m}{y}$为正整数,$\frac{n}{x}$不为正整数,则有$\left\lceil \frac{n}{x} \right\rceil$个边界块.如果想充分利用存储空间和提高查询速度,可将2个或多个边界块合并为一.但在下一小节我们会看到,这样做一般是弊大于利.

下面讨论正常块的空间利用率.不妨假设$x \geqslant y \geqslant 2$(若$y=1$,则$x=t$,空间利用率为100%),利用率$\geqslant 1-\frac{y-1}{x \cdot y+(y-1)}$,当$x=y=2$时有最小值80%,一般情况下利用率比该下限值高许多,可达90%以上.

最后我们对按类聚集、按复杂对象聚集和综合法聚集作一性能比较,结果如图6所示.

聚集方式	最坏性能	平均性能
按类聚集	$\max\left(\left\lceil \frac{n}{t} \right\rceil, m\right)$	$\frac{1}{2}\left(\left\lceil \frac{n}{t} \right\rceil+m\right)$
按复杂对象	$\max\left(\left\lceil \frac{m}{t} \right\rceil, n\right)$	$\frac{1}{2}\left(\left\lceil \frac{m}{t} \right\rceil+n\right)$
综 合 法	$\approx\sqrt{\frac{mn}{t}}$	$\approx\sqrt{\frac{mn}{t}}$

图6 3种聚集方式性能比较

2.3 动态特性

以上我们从静态角度讨论了综合聚集法.但在数据库系统的运行过程中,类的结构和类中对象数目都是不断变化的,显然在前面的讨论中我们未涉及该问题.

首先,可采用静态聚集方式,也就是说,一旦将对象写到硬盘,以后便不再重新移动其位置.则我们在定义组成复杂对象的类型(设为m个)时,需对对象的数目给出一个估计值n,然后就可按上一小节的方法进行聚集存放.

显然上述静态方法有较大缺陷，因为估计的对象数与实际情况出入往往相当大，所以性能可能不够理想. 解决的办法是采用动态聚集. 亦即每隔一定的时间或对象数目变化达到规定界限时重新聚集. 即将有关对象先卸载然后再加载，开销颇大，但能使系统保持在性能较佳的聚集状态，有时还是值得的. 如采用动态方式，重新聚集的条件设定较为重要，应根据系统运行状况确定一重聚集点.

当插入一复杂对象时，先查看有无相应的边界对象存在. 若有，则读入它们，将新对象分别插入其中，再写回. 若无，则需将内容写入$\left\lceil \frac{m}{y} \right\rceil$个新块中，并标识这些块为边界块.

删除一个复杂对象时，可查看硬盘中有无边界块. 若有，则将$\left\lceil \frac{m}{y} \right\rceil$个边界块调入，将其中一个对象移到被删除对象所在的块中，再将这$2\left\lceil \frac{m}{y} \right\rceil$个块写回(若此时边界块为空，则释放它们). 若无边界块，则定义已删除一复杂对象的这些块为边界块，将它们写回硬盘.

3　推广应用

3.1　不同概率情形

上一节我们假定用户对一类中所有对象的查询和复杂对象查询具有相同的可能性，但在实际应用中可能有不同的要求. 如用户觉得应以支持复杂对象查询为主，按类查询为辅，则上一节的方法需进一步扩充.

假定按类查询的概率为 p，按复杂对象查询的概率为 $1-p$，则定义 $x=\sqrt{\frac{t \cdot n \cdot p}{m \cdot (1-p)}}$，$y=\sqrt{\frac{t \cdot m \cdot (1-p)}{n \cdot p}}$，然后再用 2.2 节介绍的方法将 x,y 转换成整数解.

3.2　多维情形

在实际应用中，查询也有可能不只局限于按类查询和按复杂对象查询. 若系统支持版本功能，则一个复杂对象可能同时具有多个版本. 这时就需对查询对象版本提供有效支持. 这与前一节相比，又多一种聚集需求. 因涉及 3 种需求，我们不妨称之为三维情形. 上一节讨论的则为二维情形. 三维情形与二维情形处理方法亦类似.

假设一复杂对象一般有 h 个版本，则定义 $x=\sqrt[3]{\frac{t \cdot n}{m \cdot h}}$，$y=\sqrt[3]{\frac{t \cdot m}{n \cdot h}}$，$z=\sqrt[3]{\frac{t \cdot h}{m \cdot n}}$，然后再将 x,y,z 转换成较接近的整数，每个硬盘块存放 y 个类中 x 个对象的 z 个版本.

文献[1]中亦对二维、三维情形作了讨论，并给出了相应的实现方案. 但所提方案非常接近一维聚集，故只对其中一种查询有效，对其它的查询支持不够.

4　结　论

本文提出了一种面向对象数据库中同时支持按类查询和按复杂对象查询的聚集方法. 该方法从本质上说，是一种折衷方法. 在局部作出了一些牺牲，但从全局观点来看，可显著改善系统查询的最坏性能和平均性能. 性能平稳(最坏性能与平均性能大体相同)可能是最值得称道的地方. 此外实现也较简单.

我们认为,在面向对象数据库中有效支持关系操作是一种必然趋势,本文提出的聚集方法是支持面向对象数据库或对象——关系数据库的有效技术.

参考文献

1 Cheng J R, Hurson A R. Effective clustering of complex objects in object-oriented databases. Proceedings of the 1991 ACM SIGMOD International Conference on Management of Data, 1991.

2 Drew P, King R, Hudson S. The performance and utility of the cactis implementation algorithms. VLDB. 1990.

3 Chang E E, Katz R H. Exploiting inheritance and structure semantics for effective clustering and buffering in an object-oriented DBMS. Proceedings of the 1989 ACM SIGMOD International Conference of Management of Data, 1989.

4 Banerjee J *et al*. Clustering a DAG for CAD databases. IEEE Transactions on Software Engineering, Nov. 1988, **14** (**11**).

5 Tsangaris M M, Naughton J F. A stochastic approach for clustering in object bases. Proceedings of the 1991 ACM SIGMOD International Conference on Management of Data, 1991.

6 Tsangaris M M, Naughton J F. On the performance of object clustering techniques. Proceedings of the 1992 ACM SIGMOD International Conference on Management of Data, 1992.

7 Ananthanaraganan R *et al*. Using the coexistence approach to achieve combined functionality of object-oriented and relational systems. Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, 1993.

8 Stonebraker M. Object-relational database systems. OODB Journal.

附录. 算法 2 正确性证明

以下讨论中均假定 3 个正整数 m,n,t 已给定.

定义 1. (x,y)称为可能点,如果 $x\cdot y\leqslant t\wedge(x+1)\cdot y>t\wedge x\cdot(y+1)>t\wedge x,y$ 均为正整数.

引理 1. 所有可能点可组成一有序集合.

证明:我们按下法给它们排序.对 2 个可能点(x_1,y_1)和(x_2,y_2),$x_1<x_2$ 且 $y_1>y_2$,则(x_1,y_1)排在(x_2,y_2)前面.

这里条件 $x_1<x_2$ 与 $y_1>y_2$ 是相当的,亦即当 $x_1<x_2$,则一定有 $y_1>y_2$,反之亦然.

定义 2. (x_0,y_0)称为标准点,这里 $x_0=\sqrt{\frac{tn}{m}}$,$y_0=\sqrt{\frac{tm}{n}}$.若 x_0,y_0 均为正整数,则称为整数标准点,否则为非整数标准点.

在下面我们只考虑非整数标准点,并简称为标准点.亦即对 m,n,t 的取值有所限制.

此时若令 $u_0=v_0=\sqrt{\frac{n\cdot m}{t}}$,则 $x_0\cdot u_0=n$,$y_0\cdot v_0=m$.

引理 2. 将标准点加入可能点集合,则它们亦可如引理 1 那样定义次序,但有时存在一个可能点(x,y),它与标准点无法确定次序.我们称这个可能点为奇异点.

证明:对标准点(x_0,y_0)可有以下 3 种不同情况:

① x_0 为整数,y_0 不为整数;

② x_0 不为整数,y_0 为整数;

③ x_0,y_0 均不为整数.

对于情形①,$(x_0,\left\lfloor\frac{t}{x_0}\right\rfloor)$是唯一的奇异点.假设$(x_1,y_1)$是另一奇异点,则 $x_1>x_0\wedge y_1>y_0$,或者 $x_1<x_0\wedge y_1<y_0$ 成立.若 $x_1>x_0\wedge y_1>y_0$,则 $x_1\cdot y_1>x_0\cdot y_0=t$,不可.若 $x_1<x_0\wedge y_1<y_0$,则 $x_1\leqslant x_0-1$,y_1

$\leqslant\left\lfloor\frac{t}{x_0}\right\rfloor,(x_1+1)\cdot y_1\leqslant(x_0-1+1)\cdot\left\lfloor\frac{t}{x_0}\right\rfloor=x_0\cdot\left\lfloor\frac{t}{x_0}\right\rfloor\leqslant t$,所以$(x_1,y_1)$不是可能点.

对于情形②,$\left(\left\lfloor\frac{t}{y_0}\right\rfloor,y_0\right)$为唯一奇异点,证明同情形①完全类似.

对于情形③,可能有一个奇异点$(\lfloor x_0\rfloor,\lfloor y_0\rfloor)$,也可能没有. □

引理 3. 设 P 为所有可能点集合,(x_1,y_1)和(x_2,y_2)为 P 中两元素.若

① $x_2<x_1<x_0$ 或②$x_2>x_1>x_0$

则$|u_1-v_1|\leqslant|u_2-v_2|$成立,其中 $u_1=\left\lceil\frac{n}{x_1}\right\rceil,v_1=\left\lceil\frac{m}{y_1}\right\rceil,u_2=\left\lceil\frac{n}{x_2}\right\rceil,v_2=\left\lceil\frac{m}{y_2}\right\rceil$.

证明:①若 $x_2<x_1<x_0$,则 $y_2>y_1>y_0$ (引理 1)

$u_0\leqslant u_1=\left\lceil\frac{n}{x_1}\right\rceil\leqslant\left\lceil\frac{n}{x_2}\right\rceil=u_2$

$v_0\geqslant v_1=\left\lceil\frac{m}{y_1}\right\rceil\geqslant\left\lceil\frac{m}{y_2}\right\rceil=v_2,u_0=v_0$

所以$|u_1-v_1|\leqslant|u_2-v_2|$

②的证明同①类似. □

引理 4. 若 $u_1\cdot v_1\geqslant\left\lceil\frac{mn}{t}\right\rceil,u_2\cdot v_2\geqslant\left\lceil\frac{mn}{t}\right\rceil$,

$u_1\cdot(v_1-1)<\left\lceil\frac{mn}{t}\right\rceil\wedge(u_1-1)\cdot v_1<\left\lceil\frac{mn}{t}\right\rceil,u_2\cdot(v_2-1)<\left\lceil\frac{mn}{t}\right\rceil\wedge(u_2-1)\cdot v_2<\left\lceil\frac{mn}{t}\right\rceil$,

$|u_1-v_1|<|u_2-v_2|$,则 $u_1+v_1\leqslant u_2+v_2$

证明:因 u_1 与 v_1,u_2 与 v_2 地位相同,不妨令 $u_1\geqslant v_1,u_2\geqslant v_2$.假设引理 4 结论不成立,则存在满足条件的 u_1,v_1 和 u_2,v_2,但 $u_1+v_1>u_2+v_2$.

根据$|u_1-v_1|<|u_2-v_2|$可分为以下 3 种情况:

① $u_1=u_2,v_1>v_2$;②$u_1<u_2,v_1=v_2$;③$u_1<u_2,v_1>v_2$.

分别讨论如下:

① 从 $u_1=u_2,v_1>v_2$ 和 $u_1\geqslant v_1$(假定)可得

$u_2=u_1,v_2\leqslant v_1-1$

$u_2\cdot v_2\leqslant u_1\cdot(v_1-1)<\left\lceil\frac{mn}{t}\right\rceil$,与条件 $u_2\cdot v_2\geqslant\left\lceil\frac{mn}{t}\right\rceil$矛盾,故不可.

② 从 $u_1<u_2$ 可得 $u_2\geqslant u_1+1$

$u_2+v_2\geqslant u_1+v_1+1$,亦不可取.

③ 令 $u_2=u_1+k$ (k 为正整数且 $k<u_2$)

由 $u_1+v_1>u_2+v_2$ 得 $v_2\leqslant v_1-k-1$

$u_2\cdot v_2\leqslant(u_1+k)\cdot(v_1-k-1)=u_1v_1-ku_1-u_1+kv_1-k^2-k$

$\leqslant u_1v_1-u_1-k^2-k(u_1\geqslant v_1,ku_1\geqslant kv_1)$

$<u_1v_1-u_1=u_1\cdot(v_1-1)<\left\lceil\frac{mn}{t}\right\rceil$,与条件 $u_2\cdot v_2\geqslant\left\lceil\frac{mn}{t}\right\rceil$矛盾,故不可.所以引理 4 成立. □

定理 1. 至多需比较 3 个可能点,可得平均性能最佳者($\frac{u+v}{2}$亦即$(u+v)$最小者).我们称之为平均性能最优整数解.

证明:设 P 为所有可能点加上标准点(若有奇异点则除去)之集合.又设在如引理 1 证明所规定之序列中,(x_1,y_1)为标准点紧左邻,(x_2,y_2)为标准点紧右邻.则根据引理 3 和引理 4,由 $x<x_1$ 所组成的可能点其平均性能($\frac{u+v}{2}$)均不优于(x_1,y_1).由 $x>x_2$ 所组成的可能点其平均性能($\frac{u+v}{2}$)均不优于(x_2,y_2).所以我们只需比较(x_1,y_1),(x_2,y_2)和奇异点这 3 个可能点的平均性能,取其最佳者即可.在某些条件下,这 3 个可能点中的一个或 2 个不存在,则需比较的就不足 3 个可能点. □

引理 5. 设标准点为(x_0,y_0),$x_0\leqslant y_0$,$x_0\geqslant 2$.

① 对于所有的整数x_1,$1\leqslant x_1\leqslant\lfloor x_0\rfloor$,$(x_1,\left\lfloor\frac{t}{x_1}\right\rfloor)$均为可能点;

② 令$x_1=\lfloor x_0\rfloor$,$y_1=\left\lfloor\frac{t}{x_1}\right\rfloor$,若$y_1<y_0$,则$(x_1,y_1)$为奇异点,此时$(x_2=x_1-1,y_2=\left\lfloor\frac{t}{x_2}\right\rfloor)$为标准点之紧左邻.若$y_1>y_0$,则$(x_1,y_1)$为标准点之紧左邻.

对于$y_0<x_0$的情形亦有类似结论.

证明: ① 对于$1\leqslant x_1\leqslant\lfloor x_0\rfloor$,$y_1=\left\lfloor\frac{t}{x_1}\right\rfloor$

有$x_1\cdot y_1\leqslant t\wedge(x_1+1)\cdot y_1>t\wedge x_1\cdot(y_1+1)>t$成立,故$(x,\left\lfloor\frac{t}{x}\right\rfloor)$为可能点.

② 可从定义直接推出. □

引理 6. 设(x_0,y_0)为标准点,$x_0\leqslant y_0$,则(x_0,y_0)的紧右邻(x,y)可由下列3步求出:

① $x=\lfloor x_0\rfloor+1$;②$y=\left\lfloor\frac{t}{x}\right\rfloor$;③$x=\left\lfloor\frac{t}{y}\right\rfloor$

对于$y_0<x_0$有相似的结论.

证明: (x_0,y_0)的紧右邻其x值至少为$\lfloor x_0\rfloor+1$,第2步$y=\left\lfloor\frac{t}{x}\right\rfloor$,必然有$y<y_0$,而且是所有满足$y<y_0$条件的可能点中$y$值最大的一个.此时进行第3步是必要的.因第1步中所得x值可能并不能使$(x+1)\cdot y>t$成立,这时进行第3步会增大x的值. □

定理 2. 算法2是正确的.

证明: 根据引理5和引理6,算法2在L_1语句前6行形成奇异点和紧左邻(当$x_c\leqslant y_0$),从L_1语句起2行形成紧右邻,其后的循环语句与它们比较,并取平均性能最优者为解(定理1).

若$y_0<x_0$,处理大致相同.注意第L_6号语句作了一点变换,使得和$x_0\leqslant y_0$用相同的语句处理. □

A HYBRID METHOD FOR OBJECT CLUSTERING

Wu Shengli Wang Nengbin

(*Department of Computer Science Southeast University Nanjing* 210096)

Abstract This paper proposes a new object clustering method-hybrid method for effectively supporting both query of objects within a class and components of a complex objects in object oriented databases. The method has better performance in both average and the worst case than usual clustering methods.

Key words Object-oriented database, object clustering, object-relational database.