Partial 关系上的聚合操作研究*)

On Aggregate Operations over Partial Relationship

袁晓洁　康义楠

(南开大学计算机科学与技术系　天津 300071)

Abstract　Arbee and Frank defined extended aggregate operations over partial relationship[1]. And they presented aggregate algorithms about COUNT, MIN and MAX. This paper extends their works, designs and implements algorithms of SUM and AVG based on a single attribute, and SUM based on groups. The paper analyses algorithm's complexity and verifies algorithms given by us are much better.

Keywords　Relational database, Partial relationship, Aggregate operation, Algorithm complexity

现实世界中存在着许多非精确数据，在数据库系统中一般用 Null 值表示[2]，表明目前不知道该值，显然这种表示丢失了许多信息。Grant 总结定义了部分值(partial values)的概念来取代空值，使得关系表中的属性值可以不局限于不可分的原子，而可以由一个非空的相应域值组成的子集来替代。一个部分值被定义为一个可能值的有限集，在该有限集中只有一个值为"true"值。

DeMichiel[3]在解决多媒体数据库系统中域匹配问题时使用部分值的概念，他提出了操作部分值的代数方法，Grant 和 Minker 研究了模糊数据库中查询机制，他们在关系模型的基础上利用一阶谓词逻辑公式解决了可能值的有限集合问题。Lipski 给出了操作非精确信息的一般讨论，其中包括部分值的处理。

实际上，关系代数或关系演算在涉及到统计信息或聚合操作的许多重要应用时是不适合的，因此现代查询语言(如 SQL)的检索能力已大大超过了关系完备性的标准，增添了一些有用的聚合操作[4]。然而，以往人们在研究非精确信息的操作时，通常将工作重点放在扩展的关系代数上，而忽略了扩展的关系聚合操作。为此，Arbee 和 Frank[1]提出了元组属性含有部分值的聚合操作定义，并实现了 count, min 和 max 算法。本文拓展了他们的工作，实现了单属性聚合操作 sum 和 avg 算法，并实现了基于部分属性分组的扩展 sum 算法，给出了算法复杂度分析，证明了该算法是较优的。

1. Partial 关系的基本概念

➢一个部分值，用 $\eta=[a_1,a_2,\cdots,a_p]$ 表示，它对应一个属性域中可能取值的有限集合 $\nu(\eta)=\{a_1,a_2,\cdots,a_p\}$，$p\geq 1$，在该集合中只有一个值是 η 的"true"值。部分值 η 的基数被定义为 $|\nu(\eta)|$。

➢包含带有部分值元组的关系称为 partial 关系。

➢若 T 是属性集 $[A_1,A_2,\cdots,A_m]$ 上的一个 *partial* 关系，它包含元组集合 $\{t_1,t_2,\cdots,t_n\}$，$t_i=\langle\eta_{i1},\eta_{i2},\cdots,\eta_{im}\rangle$，$1\leq i\leq n$，定义 $T[A_j]=\{\eta_{1j},\eta_{2j},\cdots,\eta_{nj}\}$，$\eta_{ij}=t_i\cdot A_j$，$t_i\in T$。设 ζ_k 是 t_i 的一个解，$\zeta_k(t_i)$ 定义为元组 $(a_{i1_k},a_{i2_k},\cdots,a_{im_k})$，$a_{ij_k}$ 表示解 ζ_k 对于元组 t_i 在 A_j 上的取值。t_i 表示的所有可能元组的集合为 $\{\zeta_k(t_i)\mid$ 对 t_i 的每一个解 $\zeta_k\}$，记作 $all(t_i)$。

➢令 T 表示 partial 关系，A_j 表示 T 的一个属性，A_j 上的单属性 sum 和 avg 聚合操作定义为：

$$\nu(sum(T.A_j))=\left\{\sum_{i_j\in R_k}a_{ij_k}\mid R_k\in all(T)\right\}$$

$$\nu(avg(T.A_j))=\left\{\frac{\sum_{i_j\in R_k}a_{ij_k}}{|R_k|}\mid R_k\in all(T)\right\}$$

例 1.1　考虑以下 partial 关系：

公司

部分	人数
D1	[8,12]
D2	[10,11]
D3	[8,14]

$\nu(sum(公司\cdot 人数))=\{26,32,27,33,30,36,31,37\}$

$\nu(avg(公司\cdot 人数))=\{26/3,32/3,27/3,33/3,30/3,36/3,31/3,37/3\}$

2. 单属性聚合操作 sum 和 avg 算法设计

由单属性聚合操作的定义可知，$|\nu(sum(T.A_j))|$

*)本文研究得到国家"863"CIMS 主题项目(编号:863-511-940-010)资助。

和$|v(avg(T.A_j))|$在最坏情况下有 p^n 个值($|T|=n$，$|\eta_{ij}|=p$)。因此不可能设计出多项式时间算法来实现 sum 和 avg 操作。而如果用组合算法实现 sum 和 avg，则时间复杂度为 $O(np^n)$。下面本文为 sum 和 avg 设计一个有效算法，并证明其算法复杂度是 $O(p^n)$，接近最优算法。

定义 2.1 已知 α 和 β 是两个集合，$\alpha=\{a_1,a_2,\cdots,a_n\}$，$\beta=\{b_1,b_2,\cdots,b_m\}$，$\alpha\oplus\beta$ 定义如下：

$\alpha\oplus\beta=\{a_i+b_j|a_i\in\alpha,b_j\in\beta,1\leqslant i\leqslant n,1\leqslant j\leqslant m\}$。

由该定义可得，$|\alpha\oplus\beta|\leqslant m\times n$，最坏情况 $|\alpha\oplus\beta|=m\times n$。

定义 2.2 已知 $\eta=\{a_1,a_2,\cdots,a_n\}$，定义 $\eta^{(i)}$ 是一个部分值，它由 η 集合中元素值与 a_i 的差组成，形式如下：

$\eta^{(i)}=[a_1-a_i,a_2-a_i,\cdots,a_n-a_i]$。

由定义 2.2 可知，$\eta^{(i)}$ 表示部分值 η 中每个元素减去 η 中第 i 个元素后得到的部分值，$|v(\eta^{(i)})|=n$。

定理 2.1 若令 $T[A_j]=\{\eta_{1j},\eta_{2j},\cdots,\eta_{nj}\}$，$\eta_{ij}=[a_{i_1j},a_{i_2j},\cdots,a_{i_{p_i}j}]$，$S_0=\{\sum_{i=1}^{n}a_{i_1j}\}$，则

$S_0\oplus v(\eta_{1j}^{(1)})\oplus v(\eta_{2j}^{(1)})\oplus\cdots\oplus v(\eta_{nj}^{(1)})$
$=v(sum(T.A_j))$

证明：为方便论述，定义符号如下：

$S_1=S_0\oplus v(\eta_{1j}^{(1)})$，$S_2=S_1\oplus v(\eta_{2j}^{(1)})$，$\cdots$，$Sn=S_{n-1}\oplus v(\eta_{nj}^{(1)})$。

首先采用数学归纳法证明 $S_n\subseteq v(sum(T.A_j))$，其次证明 $v(sum(T.A_j))\subseteq S_n$。

(1) $\because v(sum(T.A_j))=\{\sum_{i\in R_k}a_{ij}|R_k\in all(i)\}$

$\therefore\sum_{i=1}^{n}a_{ij}\in(sum(T.A_j))$，故 $S_0=\{\sum_{i=1}^{n}a_{ij}\}\subseteq v(sum(T.A_j))$。

若 $S_{n-1}\subseteq v(sum(T.A_j))$，下面证明 $S_n\subseteq v(sum(T.A_j))$。

任取 $x\in S_n$，但 $x\notin S_{n-1}$，可找到 $y\in S_{n-1}$，使得 $x=y-a_{nj_1}+a_{nj_l}$，$1\leqslant l\leqslant p_n$。

$\because\forall z\in v(sum(T.A_j))$，$z-a_{ij_k}+a_{ij_l}\in v(sum(T.A_j))$，$1\leqslant k,l\leqslant p_i$

而 $S_{n-1}\subseteq v(sum(T.A_j))$，$y\in S_{n-1}$

$\therefore x=y-a_{nj_1}+a_{nj_l}\in v(sum(T.A_j))$，$1\leqslant l\leqslant p_n$

又有 $a_{nj_l}-a_{nj_1}\in v(\eta_{nj}^{(1)})$，$x=y+(a_{nj_l}-a_{nj_1})\in S_{n-1}\oplus v(\eta_{nj}^{(1)})$

故 $S_n=S_{n-1}\oplus v(\eta_{nj}^{(1)})\subseteq v(sum(T.A_j))$

根据数学归纳法可得 $S_0\oplus v(\eta_{1j}^{(1)})\oplus v(\eta_{2j}^{(1)})\oplus\cdots\oplus v(\eta_{nj}^{(1)})\subseteq v(sum(T[A_j]))$。

(2)取任意 $x\in v(sum(T.A_j))$

$\because v(sum(T.A_j))=\{\sum_{i\in R_k}a_{ij}|R_k\in all(T)\}$

$\therefore$ 可找到一组 $a_{1j_1},a_{2j_2},\cdots,a_{nj_n}$，使得

$a_{1j_1}+a_{2j_2}+\cdots+a_{nj_n}=x$，$1\leqslant k_i\leqslant p_i$，$1\leqslant i\leqslant n$。

$$x=a_{1j_1}+a_{2j_2}+\cdots+a_{nj_n}$$
$$=\sum_{i=1}^{n}a_{ij_1}+(a_{1j_1}-a_{1j_1})-(a_{2j_1}-a_{2j_1})+\cdots+(a_{nj_n}-a_{nj_1})$$

而 $\sum_{i=1}^{n}a_{ij_1}\in S_0$，$a_{1j_1}-a_{1j_1}\in v(\eta_{1j}^{(1)})$，$a_{2j_2}-a_{2j_1}\in v(\eta_{2j}^{(1)})$，$\cdots$，$a_{nj_n}-a_{nj_1}\in v(\eta_{nj}^{(1)})$

因此 $x\in S_n$，故 $v(sum(T.A_j))\subseteq S_0\oplus v(\eta_{1j}^{(1)})\oplus v(\eta_{2j}^{(1)})\oplus\cdots\oplus v(\eta_{nj}^{(1)})$。

下面以例 1.1 为背景说明实现过程。在例 1.1 中：

$S_0=\{26\}$，$\eta_{12}^{(1)}=[0,4]$，$\eta_{22}^{(1)}=[0,1]$，$\eta_{32}^{(1)}=[0,6]$

$S_1=\{26\}\oplus\{0,4\}=\{26,30\}$，

$S_2=\{26,30\}\oplus\{0,1\}=\{26,27,30,31\}$

$S_3=\{26,27,30,31\}\oplus\{0,6\}=\{26,27,30,31,32,33,36,37\}=v(sum(公司\cdot人数))$

依据定理 2.1，partial 关系上计算 sum 的算法设计如下。

设 T 是一个 partial 关系，A_j 是拟计算 sum 的属性列，$\{\eta_{1j},\eta_{2j},\cdots,\eta_{nj}\}$ 是在 A_j 上投影所得到的 n 个部分值集合 $\eta_{ij}=[a_{i_1j},a_{i_2j},\cdots,a_{i_{p_i}j}]$。

输入：要计算 sum 的属性列 A_j 上所有部分值，即 $T[A_j]=\{\eta_{1j},\eta_{2j},\cdots,\eta_{nj}\}$。

输出：在 A_j 的 sum 结果，即 $v(sum(T.A_j))$。

算法步骤：

```
V=Σ_{i=1}^{n} a_{i1};
S=V;
for i=1 to n
  { for m=2 to p_i
    { d=a_{im}-a_{i1};
      S=S∪(V⊕{d})
    }
    V=S;
}
output(S);
```

由于 avg 结果可以通过 sum 结果除以元组个数得到，因此若 $|T|=n$，则将 $v(sum(T.A_j))$ 集合中的每个元素除以 n，即可得到 $(avg(T.A_j))$。

3. 算法复杂度分析

为简化分析该算法的时间复杂度，本文首先假设：1)η_{ij} 的基都相同，即 $|v(\eta_{ij})|=p$，其中 $i=1,\cdots,n$，$|T|=n$；2)以加、减运算作为分析算法复杂度的基本运算；3)得到的 sum 值都暂时不考虑重复问题。那么，欲得到 $sum(T.A_j)$，原始组合算法需要的基本运算次数为

$(n-1)p^n$，时间复杂度为 $O(np^n)$。下面来考察一下本文提出的改进算法：

(1) $\sum_{i=1}^{n} a_{x_i}$ 需要的基本运算次数为 $(n-1)$；

(2) 在二重循环中，需完成两个工作，一是求差，一是做 $\oplus$ 运算。我们分别考虑这两种运算。

◇求差需要的基本运算次数总共为 $n\times(p-1)$。

◇做 $\oplus$ 运算需要的基本运算总次数为 p^n-1，以下给出归纳证明。

命题 在双重循环中 $\oplus$ 运算涉及的基本运算次数为 p^n-1。

证明：当 $n=1$ 时，显然只有一重循环。此时 $|V|=1$，循环次数为 $p-1$ 次，故基本运算次数为 $(p-1)\times 1$，即 p^1-1，运算结果 $|S|=p$。

假设 $n=i-1$ 时，$\oplus$ 运算需要进行的基本运算次数等于 $p^{i-1}-1$。下证 $n=i$ 时，$\oplus$ 运算包含的基本运算次数等于 p^i-1。

显然 $n=i-1$ 时，运算结果 $|S|=p^{i-1}$，则 $|V|=|S|=P^{i-1}$。当 n 取到 i 时，内循环中做的加法运算次数为：

$(p-1)\times|V|=(p-1)\times p^{i-1}=p^i-p^{i-1}$

则 $\oplus$ 运算包含的基本运算次数共为：

$(p^{i-1}-1)+(p^i-p^{i-1})=p^i-1$。

故：做 $\oplus$ 运算需要的基本运算总次数为 p^n-1。

综上所述，本算法涉及的基本运算次数为：

$(n-1)+(p-1)\cdot n(p^n-1)=p^n+n\times p-2$。

故时间复杂度为 $O(p^n)$。

另外，在实际运算中还应考虑 sum 值中出现重复值的情况，此种情况下，改进算法更优于原始算法，因为在原始算法中，必须经过 $n-1$ 次加法运算求出 sum 值后，再去判断是否是重复值。而在本算法中，由于采用集合运算的思想，每次做集合并运算以及 $\oplus$ 运算时，就可以淘汰重复 sum 值，也就是说：当 $n=i$ 时，$|S|\leqslant p^i$。这样当 n 值继续增加，内循环中做的加法运算次数减少为 $(p-1)\times(p^i-\delta_i)$（$\delta_i$ 是一个变量，表示重复的次数），所以，为求 sum 值，最坏情况下（所有的 sum 值都不相同）基本运算次数为 $p^n+n\times p-2$。sum 值中重复值越多，减少的运算量也越多。

可以用 hash 法保存 sum 值，这样一般只需要一次定位运算，即可找到某一 sum 值。若重复，则淘汰。

若 $|v(\eta_i)|=p$，$|T|=n$，在不考虑重复值情况下，sum 的总个数为 p^n，为求出全部 sum 值，至少要进行 p^n+np-2 次基本运算，我们的算法可以说是较优的。

4. 基于分组的扩展 sum 算法

在实际运算中，聚合操作经常是在分组情况下进行的，也就是通常所说的 GROUP BY 操作。如果分组属性含有部分值，则应对部分值进行分组，然后再进行聚合操作。下面我们讨论分组的扩展聚合操作及其算法。

设 $T[\cdots,A_x,A_y,\cdots]$ 是一个 partial 关系，A_x 和 A_y 是包含部分值的属性，GROUP BY 的含义是按属性 A_x 分组，对 A_y 进行聚合操作，它的结果是一个 partial 关系，该关系的元组形如 (a_x,η_y)，其中 a_x 是属性 A_x 中某个元素的一个值，η_y 是基于 a_x 的 A_y 属性上的聚合操作，Arbee[3] 定义它为扩展的聚合操作，扩展 sum 聚合操作描述如下：

对一个 partial 关系 $T[\cdots,A_x,A_y,\cdots]$，符号 $sum(T.A_y\ by\ T.A_x)$ 表示对 $T.A_x$ 进行分组，在 $T.A_y$ 上进行扩展 sum 运算，其定义为：

$$sum(T.A_y\ by\ T.A_x)=\{(a_x,\eta_y)\mid a_x\in\bigcup_{t\in T}v(t.A_x)\wedge v(\eta_y)=\{\sum_{t\in S_k}t.A_y\mid R_i\in all(T)\wedge S_k\neq\Phi\}\},S_k=\{t\mid t\in R_i\wedge t.A_x=a_x\}$$

下面举例说明扩展的 sum 聚合操作。

例3.1 考虑下列 partial 关系

SALE

ID	Product	Amount
1	A	[40,45]
2	[A,B]	32
3	B	[20,32]

$sum(SALE.Amount\ by\ SALE.Product)=$

Product	Amount
A	{72,77,40,45}
B	{20,32,52,64}

由于本文已经实现了 partial 关系上的单属性聚合算法，因此可以通过分解 partial 关系，实现基于分组的扩展 sum 聚合操作。为方便描述，我们采用扩展的关系代数运算，定义如下符号：

设 T 是一个 partial 关系 $T[\cdots,A_x,A_y,\cdots]$，则

$T[A_x,A_y]=\pi_{A_x,A_y}(T)$

// 在关系 T 上对 A_x、A_y 属性进行投影

$T_{a_x}[A_x,A_y]=\sigma_{a_x\in\eta_x\wedge\eta_y\in A_y}(T[A_x,A_y])$

// 选择 A_x 属性中含有 a_x 元素的元组

$T^1_{a_x}[A_x,A_y]=\sigma_{(a_x)=\eta_x\wedge\eta_y\in A_y}(T_{a_x}[A_x,A_y])$

// 选择 A_x 属性中只有 a_x 一个元素的元组

$T^2_{a_x}[A_x,A_y]=T_{a_x}[A_x,A_y]-T^1_{a_x}[A_x,A_y]$

// 选择 A_x 属性中除 a_x 外还有其它元素的元组

在例3.1中：

$T_A[Product,Amount]=\{(A,[40,45])\}$，

$T^2_A[Product,Amount]=\{([A,B],32)\}$

$T^1_B[Product,Amount]=\{(B,[20,32])\}$，

$T_p^1[Product, Amount]=\{\langle[A,B],32\rangle\}$

由于分组扩展聚合操作必须区分每一个属于$\bigcup_{t\in T}v(t.A_x)$的元素，并对每一$\bigcup_{t\in T}v(t.A_x)$中元素进行一次聚合操作，从而得出$m=|\bigcup_{t\in T}v(t.A_x)|$个结果，因此以下算法只针对聚合结果中的一行元组进行。

扩展的sum算法须引入空元素的概念。设Φ是一个空元素，a是一个数值，则定义$\Phi-a=a+\Phi=a$，$a-\Phi=a$，$\Phi+\Phi=\Phi$。

对$T_x^1[A_x,A_y]$做如下处理，在A_y属性上的每个partial值中，添加一个Φ元素，并使其位于第一个元素的位置，形成$T_x^{2\Phi}[A_x,A_y]$关系，即：

$T_x^{2\Phi}[A_x,A_y]=\{\langle\eta_{x_i},\{\Phi,a_{i_{y_1}},a_{i_{y_2}},\cdots,a_{i_{y_m}}\}\rangle|\eta_{x_i}\in A_x,\{a_{i_{y_1}},a_{i_{y_2}},\cdots,a_{i_{y_m}}\}\in A_y,\langle A_x,A_y\rangle\in T_x^2[A_x,A_y],1\leqslant i\leqslant n\}$

扩展sum聚合操作算法设计如下：

$S=\bigcup_{t\in T}v(t.A_y)$；

对所有$\omega_x\in S$，执行如下操作：

{
构造$T_x[A_x,A_y]$；
构造$T_x^1[A_x,A_y]$和$T_x^2[A_x,A_y]$；
构造$T_x^{2\Phi}[A_x,A_y]$；
$T_x[A_x,A_y]=T_x^1[A_x,A_y]\cup T_x^{2\Phi}[A_x,A_y]$；
根据单属性sum聚合操作算法，计算$S_y=v(sum(T'_x[A_x,A_y].A_y))$；
$S_y=S_y-\Phi$；
输出$\langle a_x,S_y\rangle$
}

在例3.1中：

$T_A[Product,Amount]=\{\langle A,[40,45]\rangle,\langle[A,B],32\rangle\}$

$T_A^1[Product,Amount]=\{\langle A,[40,45]\rangle\}$

$T_A^{2\Phi}[Product,Amount]=\{\langle[A,B],[\Phi,32]\rangle\}$

$T_A[Product,Amount]=T_A^1\cup T_A^{2\Phi}=\{\langle A,[40,45]\rangle\langle A,B,[\Phi,32]\rangle\}$

$v(sum(T_A[Product,Amount].Amount))=\{40,45,72,77\}$

所以$\langle A,\{40,45,72,77\}\rangle\in v(sum(SALE.Amount\ by\ SALE.Product))$。

结束语 partial关系上的聚合操作广泛应用于模糊数据库和多媒体数据库，本文在所承担的国家863计划"制造类型企业的自我诊断与评价方法研究"项目中，利用值不确定关系数据模型上的聚合操作算法，解决了企业诊断中不完全信息处理的问题，完成了值不确定情况下的诊断分析、趋势预测以及导航库性能评价等问题。

参考文献

1 Chen A L P, Tseng F S C. Evaluating Aggregate Operations Over Imprecise Data. IEEE Trans. Knowledge and Data Engineering, 1996, 8(2): 273～283

2 Chen Arbee L P, Tseng Frank S C. Evaluating Aggregate Operations Over Imprecise Data. IEEE Trans. Knowledge and Data Engineering, 1996, 8(2): 273～283

3 Codd E F. Missing Information (Applicable and Inapplicable) in Relational Databases. ACM SIGMOD Record, 1986, 15(4): 53～78

4 DeMichiel L G. Resolving Database Incompatibility: An Approach to Performing Relational Operations over Mismatched Domains. IEEE Trans Knowledge and Data Engineering, 1989, 1(4): 485～493

5 Date C J. A Guide to the SQL Standard, Reading, Mass: AddisonWesley, 1989

（上接第16页）

2.4 配送

物流配送体系是电子商务实现过程中的一个重要环节，收到商家发送的配送信息后，配送公司到生产厂家取货，并在规定的时间内将货送往用户，以保证良好的信用。以下是目前常用的几种配送系统：EMS特快专递服务；铁路方面有中铁快运等方式；公路方面可实现近郊的配送；航空配送实现对时间要求苛刻，而且有经济实力的客户的配送；全国各大中心城市配送系统。

根据用户对时间的要求及其经济实力可采取个性化的服务，时间要求紧的可加收额外的加快费等。

2.5 厂家

厂家根据配送公司是否能够提供商家有效授权凭证来确定是否让配送公司提货，若合法，则根据商家提供的订单提取相应商品。

结束语 电子商务是21世纪的经济增长点，所以本文所述的电子商务系统模型将对各商家电子商务的开展产生积极的指导意义，从而降低各商家的开发成本，提高劳动生产率。

参考文献

1 邵维忠，杨芙清 面向对象的系统分析. 清华大学出版社，广西科学技术出版社，1998，12

21 周建鹏. 电子商务技术 电信科学，1999，1(1)