分布式数据库查询执行代价模型的研究

Cost Model of Query Plan in Distribute DataBase System

帅训波 周相广
（中国石油勘探开发研究院廊坊分院地球物理与信息研究所 河北廊坊 065007）
李树铁 （中国科学研究院渗流力学研究所 河北廊坊 065007）
陈 东 （中国石油勘探开发研究院 北京 100083）
龚 安 （中国石油大学计算机与通信工程学院 山东东营 257061）

摘要：本文在收益半连接的研究基础上，对查询执行代价模型进行研究，提出一种包括数据副本选择、多连接查询次序、操作站点选择、数据传输及局部数据处理等因素的代价模型，能准确地描述当前分布式数据库查询执行计划的代价，更具有实际意义。

关键词：网络传输代价 查询时间 局部处理代价 代价函数

1 引言

由于网络模型不同，数据分布、通信延迟和操作间的资源冲突等因素影响，使得分布式数据库查询执行代价描述比较复杂[1,2]。然而，随着“海量信息”出现，数据大量冗余存储，站点局部数据处理能力的差异明显，是当前分布式数据库不可回避的问题。因此，对于分布式数据库的查询，评估其查询执行代价的模型要求应该更加完善。在收益半连接的研究基础上[3]，对查询执行代价模型进行研究，它包括数据副本选择、多连接查询次序、操作站点选择、数据传输及局部数据处理等因素，能准确地描述当前分布式数据库查询执行计划的代价，更具有实际意义。

2 查询执行计划表示

数据库查询执行计划用一棵查询二叉树 P = (V, E) 表示，V 是查询树的结点，表示查询关系序列，E 是查询树的边，表示连接操作，根结点表示查询的最终结点，中间结点代表中间连接操作，叶子结点表示基本关系。同一个查询语句可有不同的查询执行计划，如图 1 和图 2 所示。

SQL 语句 Select A1, … from BF0, BF1, BF2, BF3 where BF0. A0 = BF1. A0 and BF1. A1 = BF2. A1 and BF2. A2 = BF3. A2

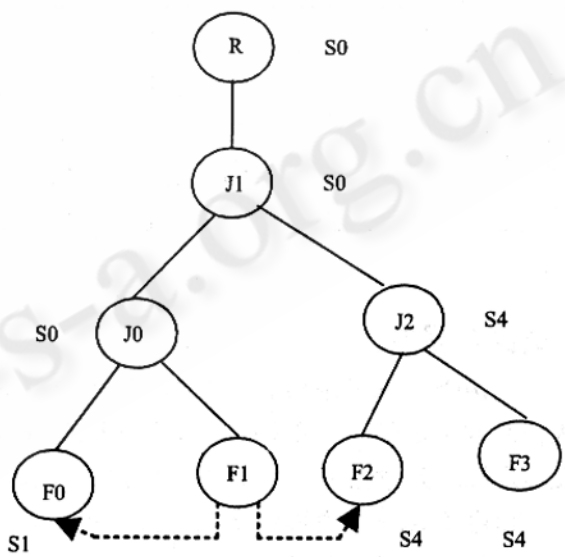

图 1 查询执行计划 A

假设关系 BF0 副本存储的站点为 S1 和 S2，BF1 存储在 S3，BF2 存储在 S4，BF3 副本存储在 S3 和 S4。其对应的两个查询执行计划分别为图 1 和图 2 所示。

其中关系用表示，表示关系的选择副本和选择的操作站点。表示连接操作。半连接操作的表示，由“缩减关系”指向“被缩减关系”。

不同查询执行计划，在副本的选择、关系连接的次序、操作站点的选择等方面，有较大的差别。具体哪个

查询执行计划较优，本文的研究主要是从网络的传输代价和查询时间两个方面考虑。其中查询时间是查询树中查询时间代价最长的查询分支。

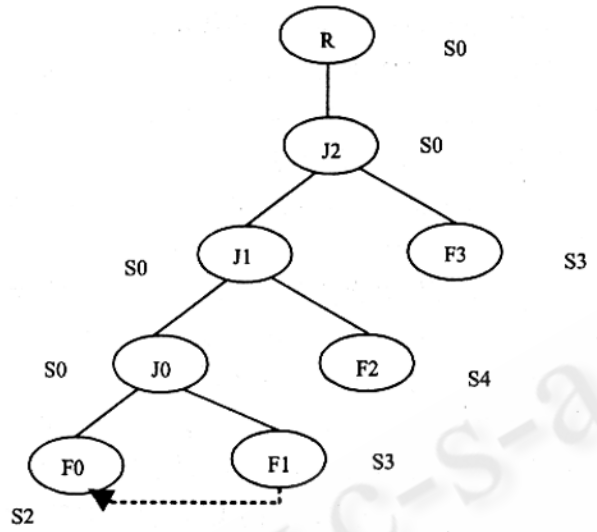

图 2　查询执行计划 B

3　模型设计

3.1　叶子连接代价函数

在查询树中，叶子结点是基本连接关系表，查询连接过程代价包括：局部处理代价和执行半连接过程时的投影操作代价、网络传输代价和关系的连接操作代价等。对于一个查询执行计划，其叶子结点的代价用关于所有叶子结点的代价函数表示如下：

$$cost(v)=\sum_{i=1}^{n}\{IO_{project}(site(u_i),baseRelation(u_i),joinAttribute(v,u_i))+IO_{send}(site(u_i),attributeSize(baseRelation(u_i),joinAttribute(v,u_i)))+COM(site(u_i),site(v),attributeSize(baseRelation(u_i),joinAttribue(v,u_i)))+IO_{receive}(site(v),attributeSize(baseRelation(u_i),joinAttribute(v,u_i)))+IO_{join}(site(v),attribueSize(baseRelation(u_i),joinAttribute(v,u_i)),size_{i-1}(v),size_i(v))\}. \quad (1)$$

公式中，v 表示叶子结点；n 表示查询计划中叶子结点 v 的总个数；base Relation(v) 表示叶子结点 v 所代表的基本关系；site(v) 表示 v 所选的副本站点位置；joinAttribute(v,u) 表示关系 v 和 u 的连接属性；attributeSize(v,a) 表示关系 v 在半连接操作中缩减属性的传输量；$size_i(v)$ 表示关系 v 被第 i 关系缩减，半连接操作后的临时结果，其大小估计在第 4 小节再做详细分析。

3.2　中间连接代价函数

在查询计划树中，中间结点表示每个操作步骤，其孩子结点的操作结果作为输入，进行连接操作。在连接操作过程中，当连接关系处在不同站点时，收益半连接操作包括网络传输的代价。因此，中间结点所表示的连接操作过程可描述为：孩子结点对应的关系，执行有益半连接时进行读写、数据传输、数据的接收，然后执行连接操作，得到临时结果，并作为输入传向该中间结点，再与其兄弟结点执行操作，得到的结果作为输入传向上一层结点。综合分析中间结点所表示的连接操作，代价函数表示如下：

$$cost(v)=\sum_{u\in children(v)\wedge sites(u)\neq site(v)}\{IO_{send}(site(u),size(u))+COM(site(u).size(v),size(7))+IO_{receive}(site(v),size(u))\}+IO_{join}(site(v),size(leftchild(v)),size(rightchild(v)),size(v)) \quad (2)$$

在公式中，leftchild(v) 和 rightchild(v) 分别表示中间结点 v 的左右孩子。由构造此函数的模型过程可见，中间结果代价评估比较复杂。

3.3　结果操作代价函数

在查询树中，根结点表示查询计划的最后完成。在有些情况下，最后的连接操作结点与查询结果站点是不同的，因此，需要做查询执行计划的最后处理，即把关系操作的最终结点传输到最终的结果站点处。其代价函数模型表示如下：

$$cost(v)=\{IO_{send}(site(u),size(u))+COM(site(u),site(v),size(u))+IO_{receive}(site(v),size(u))\}\ if\ site(v)\neq site(u),\ =0\ otherwise \quad (3)$$

在公式中 u 为关系操作的最终结点，根结点 v 是查询计划的最终结果。

3.4　传输计量代价函数

在上述函数模型中，临时连接结果大小的估算非常重要。我们用[4][5]中的方法，对查询执行中产生的临时关系结果大小进行估算。首先，假定属性值在表的元组中是均匀分布的，并且不同属性间的取值无关。

设 C_T 是表 T 的基数（即关系表 T 中不同元组的个

数)；CD_A 是属性 A 的所有不同取值的个数；W_A 为属性 A 的宽度；并假设所有属性的宽度都是固定的。因此，表 T 的宽度 W_T 可用公式计算：$W_T = \sum_{A_i \in T} W_{A_i}$. (4)

再定义表 T 中属性 A 的选择率为：$selectivity(T,A) = U(T,A)/CD_A$. (5)

其中 U(T,A) 是表 T 中属性 A 的不同取值个数。代价模型中其它元素：

表 T 中属性 A 的大小 $attributeSize(T,A) = U(T,A) * W_A$. (6)

则节点 V 连接后，输出结果大小：$size(v) = C_v \times W_v$. (7)

在中间操作过程中，对于收益半连接操作结果代价估算是，设结点 V 被“缩减关系”U 在属性 A 上缩减，则缩减后关系表大小：$C_T = C_v \times selectivity(u,A)$. (8)

推而广之，如果结点 V 不是被一个表的半连接所缩减，而是被多个表的半连接缩减，那么缩减后的结果中不同元组个数为：

$$C_T = C_v \times \prod_{u \in Reducer(v)} selectivity(baseRelation(U), JoinAttribute(u,A)). \quad (9)$$

其中 Reducer(v) 是缩减关系 baseRelation(u) 所组成的集合，JoinAttribute(u,A) 表示减缩属性。

因此，综上分析可得，操作结果关系表的大小为：

$$C_T = \prod_{u \in children(v)} / [CD_{joinAttribute(v)} \prod_{u \in leaves(v)} \prod_{w \in reducers(v)} selectivity(baseRelation(w), jionAttribute(u,w))] \quad (10)$$

这里 joinAttribute(v) 是中间结点 V 和孩子结点 children(v) 或叶子结点 leave(v) 的连接属性。

下面举例说明公式(11)的应用，假设有连接 T_1 和 T_2，连接属性为 A，

结果关系表为 T_3，那么结果就有：$C_{T_3} = C_{T_2} \times C_{T_1}/CD_A$. (11)

对于关系连接中，每次操作连接执行两个关系，如果连接属性不是纯连接属性，则从结果中去掉多余的连接属性，只保留一个；如果连接属性是纯连接属性，则从结果关系中把该连接所有的属性去掉。于是，对于每次连接操作的结果，关系表 T 的宽度可以这样估算：

$$W_{T_3} = W_{T_1} + W_{T_2} - 2W_{joinAttribute(T_1,T_2}$$ 若 $joinAttribute(T_1, T_2)$ 为纯连接属性；

$$= W_{T_1} + W_{T_2} - W_{joinAttribute(T_1,T_2}$$　否则；(12)

在关系操作中，具体时间代价的计算，本文采用[4]的估算方法。

4　模型实现

4.1　网络性能与站点计算性能差异表示

对于分布式数据库系统，当大量的数据需要传输时，网络速度是影响查询响应时间的“瓶颈”。不同站点的计算能力也影响着查询时间，特别在当今数据挖掘、知识发现等智能领域需要海量数据时，局部处理代价已经是不可忽略的问题。当系统中各站点的计算能力相同时，局部处理代价直接由将要处理的数据量来决定；当系统的网络特性、各站点的计算能力相同时，模型代价将由数据的传输量来决定。

研究模型目的是有效评估代价较小的最优查询执行计划。对于系统中的网络特性、站点的计算能力等存在差异时，可不必将各站点的实际性能标量精确化，而是将这种差异用反映该性能比的数据参数直观表示。在代价计算中，既能直观表示网络性能差异与站点计算性能差异，又易于数据管理与代价计算。例如：站点间的网络性能比 2:1 表示，前者站点间的网络传输速度是后者的两倍。因此，用尽可能精确的性能比参数表示站点计算性能差异和网络间的性能差异。

4.2　全局数据字典

研究过程中，通过建立全局数据字典，读取各站点和关系的静态特性，包括关系存储属性大小、元组数目等。

全局数据字典是对分布式数据库全局结构和信息的描述。一个详细准确的全局数据字典，对数据库来说非常重要。它主要包括：

(1) 字段名；

(2) 字段类型(表示字段的数据类型，如：字符，日期，数字等)；

(3) 字段数据的长度(表示此数据所占的空间大小)；

(4) 字段数据允许取值的个数(表示此字段上的数据可以取多少不同的值)；

(5) 字段数据的取值(最大取值和最小取值)；

(6) 表名称(此字段所在表的名字)；

(7) 表的记录数(此表所含的记录的个数)；

(8) 站点号(此表所在的站点处)；

(9) 数据库名称(建立此表的数据库);

在建立全局数据字典时,当一个字段在多个表中出现时,应建立多个记录;一个字段所在的一个表分布在多个站点,也需要建立多个记录。

4.3 状态动态数据字典

状态动态数据字典的功能是对分布式数据库全局各站点及关系表的动态状态进行描述,它主要包括:

(1) 站点号;

(2) 站点状态(表示站点是否工作正常);

(3) 数据库名;

(4) 表的个数;

(5) 表名称;

(6) 与其他各站点的网络连接 (网络传输速度的性能比);

(7) 计算性能(全局网络中计算性能比);

状态动态数据字典中对每个站点及其存放的关系表的状态进行动态详细描述和定义。站点正常工作时,与其他各站点的网络性能和计算性能才被识别有效;否则,将以无穷大或负数表示此站点非正常工作。

网络传输速度的性能比和全局网络的计算性能比是模型实现的关键因素。在研究过程中,应用网络性能矩阵表示站点间的网络性能,应用计算性能数组表示各站点的计算性差异。

网络性能矩阵直观表示了各站点间的网络性能,是一个对称矩阵。当进行代价评估时,访问性能矩阵,得到网络性能参数。当网络状态发生改变时,修改性能矩阵,反应当前状态的网络性能。例如站点数为 N 的分布式数据库,其动态数据字典的网络性能矩阵用 N×N 表示,每个元素值,表示该行站点与该列站点间的网络性能在当前全局网络性能的比。由实际各站点间的网络性能矩阵,得到网络性能矩阵的方法是:站点自身网络性能用 -1 表示,以站点网络性能最小的量为基准,网络性能除以最小网络性能量的所得值,是其在整个网络中的相对性能比。例如有 4 个站点的网络性能矩阵表示为:

$$M=\begin{bmatrix} -1 & 1.0 & 1.7 & 3.1 \\ 1.0 & -1 & 0 & 2.8 \\ 1.7 & 0 & -1 & 1.3 \\ 3.1 & 2.8 & 1.3 & -1 \end{bmatrix}$$

性能矩阵用 M 表示,站点 1 和 4 之间的网络性能最好,相对比为 3.1,站点 1 和 2 间的网络性能最差,比值为 1.0,站点 3 和站点 2 之间无法通信。

在状态动态数据字典中,计算性能数组反应各站点的计算性能的数据结构。站点数 N 的分布式数据库系统,计算性能数组用 Compute[N] 存储,以性能最差的站点为基准,站点计算性能除以基准,计算结果存储在 Compute[i] 中,表示该站点处理数据性能在全局的相对性能比。

网络性能矩阵和计算性能数组是状态动态数据字典的重要部分,存储在一个站点或者以副本形式存储在多个站点,全局状态发生变化时,实时触发更新数据字典。查询计划首先访问全局数据字典,状态动态数据字典被查询计划访问,获取相应参数,并分别计算网络传输代价、局部处理代价、操作代价等,逐步实现查询执行代价。

5 结论

本文在对收益半连接的研究基础上,详细阐述了一种新的分布式数据库查询计划的执行代价模型及其实现,它包括数据副本选择、多连接查询次序、操作站点的选择、数据的传输及局部数据处理等因素。并对其进行了详细设计、分析和实现。

参考文献

1 Chen M. S, Yu P. S. Interleaving a join sequence with semi – join in distributed query Processing. IEEE Trans. Parallel and Distributed System, 1992,3(6).

2 Chen, M. S., Yu, P. S. Using combining join and semi – join operations for distributed query processing. IEEE Trans on Knowledge and Data Engineering. 1993,5 (3).

3 Chen M. S, Yu P. S, Wu, K. L. Optimization of parallel execution for multi – join queries. IEEE Transaction on Knowledge and Data Engineering, 1996,8 (3).

4 于红、王秀坤,基于值的分布式查询优化算法,大连理工大学学报, 2005,45(3).

5 芦金石、李红星、李晓,基于遗传算法的分布式异构数据库查询优化,计算机应用与软件,2003,20 (11).