第 4 卷第 2 期
1993 年 4 月

软 件 学 报
JOURNAL OF SOFTWARE

Vol. 4, No. 2
Apr., 1993

求解循环查询的一种新方法——标志位映射法

须 德　张 彤

(北方交通大学计算机系，北京 100044)

A NEW METHOD FOR SOLVING CYCLIC QUERY: MAPPING AMONG FLAG BITS (MAFB)

Xu De and Zhang Tong

(*Northern Jiaotong University, Beijing* 100044)

Abstract The semi-join relational operation is augmented and a new method for solving cyclic queries in distributed databases is described in the paper. This new method can fully reduce all the relations in a cyclic query. 5n-4 times of data transmission is required to carry out this new method, where n stands for the number of nodes in the query graph.

摘要 本文对半连接运算进行扩展，提出一个新的循环查询求解方法——标志位映射法，该方法能将循环查询中的所有关系完全化简，代价为 5n-4 次相邻结点间的数据传输，其中 n 为查询图中的结点数.

§1. 引　言

在远程分布式数据库系统中，减少数据传输量是查询优化的主要考虑因素. 理论和实践都已表明，半连接运算是减少数据传输量的有效手段. 文献[1]证明了对于树查询，存在一个长度取决于结点数的半连接序列能完全化简所有关系，但对于循环查询，则存在数据库的某种状态使得查询所涉及的关系不可能被任何半连接序列化简. 很多文献针对循环查询提出了各自的处理方案[3,4,7]，但都不够完善. 本文提出一个能直接用于求解循环查询的方案，使查询处理效率有明显的提高，不失一般性，我们假定本文所要处理的查询具有如下形式：

$$Q(R)=\{<t_1,\cdots,t_r>\in R_1\times\cdots\times R_r \mid \bigwedge_{i=1}^{r}(R_i\cdot X_i=R_{i+1}\cdot X_i)\}$$

其中关系 $R(1\leqslant i\leqslant r)$ 分布在 r 个结点中 t_i 为关系 R_i 的元组，X_i 和 X_{i+1} 是 R 的两个属性。

在条件限制部分，我们规定 $R_{r+1}=R_1$，这样，上述查询的查询图是一条基本回路(关于查询如何转换为查询图，文献[2]有详细的叙述).

本文 1990 年 12 月 18 日收到，1991 年 3 月 7 日定稿. 作者**须德**，副教授，主要研究领域为数据库、计算机网络、人工智能. **张彤**，1992 年硕士毕业于北方交通大学，主要研究领域为数据库、人工智能.

§2. 运算和子过程描述

首先研究一下一般意义下的半连接策略不能完全化简循环查询的原因. 从半连接的定义可以看出，经过半连接化简后的关系仅保证了同一属性的实际取值集合是相同的，实现了集合意义下的对应关系，精确度并未达到元组一级. 如执行完半连接运算 $R_1 \underset{A}{\propto} R_2$ 或 $R_2 \underset{A}{\propto} R_1$ 后，仅保证 R_1 和 R_2 在属性 A 上的取值集合是相同的，对于循环查询来说，这样的对应关系难以准确地确定哪些元组满足条件.

本文的方案是对每个属性值增加一个标志位. 把某关系中某属性的实际取值抽出来(不删除相同的值)并为每个属性值增加相应的标志位(为 0 或 1)，形成一个属性值向量，向量中标志位为 0 的属性值及紧跟其后的标志位连续为 1 的属性值形成一个向量片断(称为一个映射簇)，对应一个元组. 从下一个标志位为 0 的属性值开始，对应下一个元组，余类推.

求解方案执行时，把关系 R_i 的全部属性值的标志位都设置为 0(参见 §3 求解方案第(2),(5)步)，其余关系的属性值标志位的值将通过调用子过程 map 自动产生(参见 §3 求解方案第(3),(6),(7)步).

下面用类 PASCAL 描述求解方案要用到的三个局部处理过程. R_k 表示某关系，D、D_1 和 D_2 为 R_k 的属性，V、V_1 和 V_2 表示与 D、D_1 和 D_2 相对应的属性值向量，P。VE 表示 VE 的读写指针(VE 为某向量)，flag(x) 表示 x 的标志位.

i) map(R_k,V_1,V_2,D_1,D_2)

功能：根据属性值向量 V 和 R 中属性 D 与 D 的对应关系，产生新的属性值向量 V，使得 V_1 和 V_2 之间的映射簇一一对应.

```
PROCEDURE MAP (Rk,V1,V2,D1,D2)
  BEGIN V2 := Φ; SET P.V1 TO TOP;
    WHILE NOT (BOTTOM(P.V1)) DO
      BEGIN S := FLAG(P.V1);
        SET P.Rk TO TOP;
        WHILE NOT (BOTTOM(P.Rk)) DO
            BEGIN
              IF P.Rk.D1 = P.V1 THEN
                BEGIN
                    X := p.Rk.D2; FLAG(X) := S;
                    S := 1; APPEND(V2,X);
                END;
              P.RK := P.RK + 1
            END;
          P.V1 := P.V1 + 1
        END
  END
```

例 2.1：设关系 R_k 及属性值向量 V_1 的赋值如图 2.1(a) 所示，其中方括号所括为标志位，花括号所括为一个映射簇. 调用 map 过程后，V_2 及其与 V_1 的对应关系如图 2.1(b) 所示.

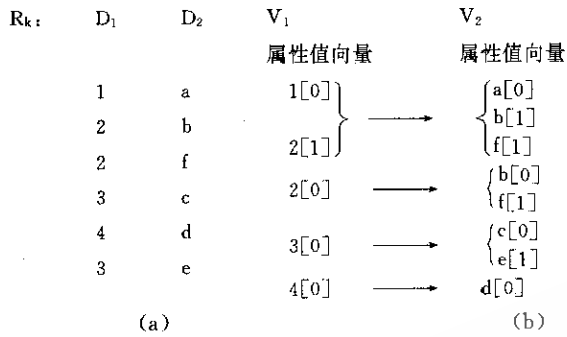

图 2.1

ii) delete(R_k,V,D)

功能：若 R_k 的第 i 个元组的属性值 D 不出现在向量 V 的第 i 个映射簇中，则删除该元组.

```
PROCEDURE DELETE(Rk,V,D)
  BEGIN SET P.Rk TO TOP;
        SET P.V TO TOP;
    WHILE NOT(BOTTOM(P.Rk)) DO
    BEGIN TAG:=FALSE;
      REPEAT
        IF P.Rk.D=P.V THEN TAG:=TRUE;
         P.V:=P.V+1
      UNTIL FLAG(P.V)=0 OR BOTTOM(P.V);
        IF TAG=FALSE THEN DELETE THE CURRENT TUPLE;
         P.Rk:=P.R+1
    END
  END
```

iii) select(R_k,V_1,V_2,D_1,D_2)

功能：若存在 j 使得 $t_i.D_1$ 出现在 V_1 的第 j 个映射簇中且 $t_i.D_2$ 出现在 V_2 的第 j 个映射簇中，则把 t_i 选出来存放于 R'_k. 其中 $t_i \in R_k$，$t_i.D$ 表示元组 t_i 的属性 D.

```
PROCEDURE SELECT(Rk,V1,V2,D1,D2)
  BEGIN R'k:=Φ;
    SET P.V1 TO TOP; SET P.V2 TO TOP;
    WHILE NOT(BOTTOM(P.V1)) DO
        BEGIN
          FOR ALL(t∈Rk AND t.D1=P.V1) DO
            BEGIN Q:=P.V2;
              REPEAT
                IF t.D2=P.V2 THEN APPEND(R'k,t);
                P.V2:=P.V2+1
              UNTIL FLAG(P.V2)=0 OR BOTTOM(P.V2)
            END;
          P.V1:=P.V1+1;
          IF FLAG(P.V1)=1 THEN P.V:=Q
        END
  END
```

§3. 求解方案

在上节基础上，循环查询求解方案叙述如下：

(1) 删除查询图中的边 $R_r.X_r=R_1.X_r$，原查询变为树查询，用树查询的半连接简化策略对全部关系进行化简.

这一步的作用是保证在以后调用 map(R_k,V_1,V_2,D_1,D_2) 过程时，V_1中没有哪个映射簇在V_2中的映射为空，从而保证了V_1和V_2中的映射簇的一一对应关系.

(2) 对R_1作X_1和X_r投影得R'_1，再取出R'_1的X_1属性值赋给A_1^*(保留相同的属性值)，并置所有标志位为 0，这样，R'_1中每个元组自成一个映射簇.

(3) 对 K=2,3,…,r，按顺序在结点R_k调用过程 map(R_k,A_{k-1}^*,A_k^*,X_{k-1},X_k)生成A_k^*并传送到结点R_{k+1}，最后A_r^*传送回结点R_1.

(4) 结点 R 调用 delete(R'_1,A_r^*,X_r)，化简关系R_1.

由第(3)步可知，A_k^{*1}和A_1^{*1}中的映射簇是一一对应的(K=2,3,…,r)，那么A_1^{*1}和A_r^{*2}的映射簇也是一一对应的. 如果R'_1的第 i 个元组t_i的X_r属性值不出现在A_r^{*2}的第 i 个映射簇中，说明不存在一个映射从$R_1,R_2,\cdots,R_r$再回到R_1使t_i满足查询条件，应将其删去；反之，则应保留.

所以，调用 delete(R'_1,A_r^*,X_r) 后，R_1被完全化简，R'_1为最简关系.

(5) 取出R'_1的属性X_1和X_r的值(不删除重复值)作为A_1^{*1}和A_r^{*2}，标志位全部置为 0.

(6) 对 k=2,3,…,r−1，在结点R_k调用 map(R_k,A_{k-1}^{*1},A_k^{*1},X_{k-1},X_k) 生成A_k^{*1}传递给结点R_{k+1}.

(7)对 k=2,3…,r−1,在结点R_k 调用 map(R_k,A_{k-1}^{*1},A_k^{*1},X_k,X_{k-1})生成A_{k-1}^{*2}传送给结点R_{k-1}.

第(6)(7)步执行完后，结点R_k(k=2,3,…,r) 保存着两个属性值向量：A_{k-1}^{*1}和A_k^{*2}. 如图 3.1 所示：

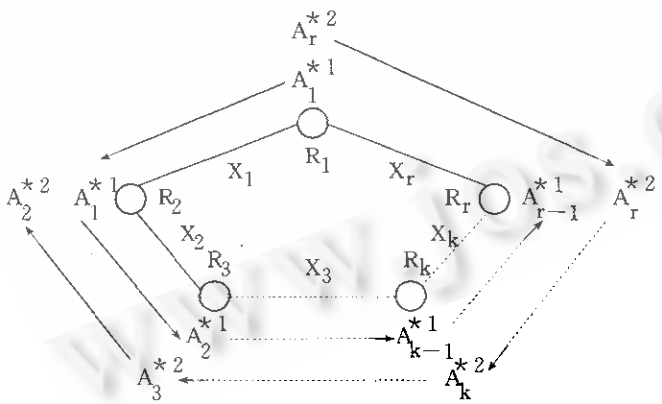

图 3.1

(8) 结点R_k(K=2,3,…,r) 调用 select(R_k,A_k^{*1},A_k^{*2},X_{k-1},X_k) 生成关系R'_k.

由第(5)步可知，A_1^{*1}与A_r^{*2}的映射簇一一对应，由第(6)(7)步可知A_{k-1}^{*1}与A_k^{*1}(K=2,3,…,r−1)、A_k^{*2}与A_{k-1}^{*2}(K=r,r−1,…,3)的映射簇也是一一对应的，因此A_{k-1}^{*1}与A_k^{*2}的映射簇具有一一对应关系. 如果存在$t_p\in R_k$且$t_p.X_{k-1}$出现在A_k^{*1}的第 i 个映射簇中，$t_p.X_k$出现在A_k^{*2}的第 j 个映射簇中并且 i=j，则说明存在一条映射回路使t_p满足查询条件，选出所有这样的元组形成关系R'_k，则R'_k为最简关系(K=2,3,…,r).

至此，循环查询中的全部关系都被完全化简.

例 3.1：一个数据库中有R_1、R_2、R_3三个关系，其赋值情况示于图 3.2.

查询条件为 $R_1.X_1=R_2.X_1\wedge R_2.X_2=R_3.X_2\wedge R_3.X_3=R_1.X_3$

R_1		R_2		R_3	
X_3	X_1	X_1	X_2	X_2	X_3
1	a	*a	i	r	2
2	b	a	n	*i	2
3	g	b	j	j	1
*3	c	*b	l	*k	3
1	h	d	p	s	5
*2	a	*c	k	r	4
4	d	d	l	*n	3
*3	f	e	q	*l	5
2	h	e	m	l	7
6	f	*f	n	m	6
5	e	b	p	m	3
*5	b	d	q	n	7

(图中打 * 号的为满足查询条件的元组)

图 3.2

R_1		R_2		R_3	
X_3	X_1	X_1	X_2	X_2	X_3
1	a	*a	i	*i	2
2	b	a	n	j	1
*3	c	b	j	*k	3
*2	a	*b	l	*n	3
4	d	*c	k	*l	5
*3	f	d	l	l	7
6	f	e	m	m	6
5	e	*f	n	m	3
*5	b			n	7

图 3.3

执行完第(1)步后，数据库状态示于图 3.3.

执行完第(2)(3)步后，A_1^*、A_2^*、A_3^*示于图 3.4.

执行完第(4)(5)步后，所得结果示于图 3.5.

执行完第(6)(7)步后，所得结果示于图 3.6.

执行完第(8)步后，数据库状态示于图 3.7.

A_1^*	A_2^*	A_3^*
a[0]	i[0]	2[0]
b[0]	n[1]	3[1]
c[0]	j[0]	7[1]
a[0]	l[1]	1[0]
d[0]	k[0]	5[1]
f[0]	i[0]	7[1]
f[0]	n[1]	3[0]
e[0]	l[0]	2[0]
b[0]	n[0]	3[1]
	n[0]	7[1]
	m[0]	5[0]
	j[0]	7[1]
	l[1]	3[0]
		7[1]
		3[0]
		7[1]
		6[0]
		3[1]
		1[0]
		5[1]
		7[1]

图 3.4

R'_1:

X_3	X_1	A_1^{*1}	A_3^{*2}
3	c	c[0]	3[0]
2	a	a[0]	2[0]
3	f	f[0]	3[0]
5	b	b[0]	5[0]

图 3.5

A_2^{*1}	A_2^{*2}
k[0]	k[0]
i[1]	n[1]
n[1]	m[1]
n[0]	i[0]
j[0]	k[0]
l[1]	n[1]
	m[1]
	l[0]

图 3.6

R_1		R_2		R_3	
X_3	X_1	X_1	X_2	X_2	X_3
3	c	a	i	i	2
2	a	b	l	k	3
3	f	c	k	n	3
5	b	f	n	l	5

图 3.7

至此，循环查询中的全部关系都被完全化简.

§4. 求解方案的正确性

在下面的证明中，我们用$[V]_i$表示属性值向量 V 的第 i 个映射簇，t_1、t_2、…、t_r分别表示R_1、R_2、…、R_r的任意一个元组，R'_k表示R_k中满足查询条件的全部元组的集合. 显然，R_k中满

足条件的元组与其它关系中的元组存在如下关系：

$t_r \in R'_k \Leftrightarrow \langle t_1,\cdots,t_{k-1},t_{k+1},\cdots,t_r\rangle \in R_1\times\cdots\times R_{k-1}\times R_{k+1}\times\cdots\times R_r$

使得 $\bigwedge_{i=1}^{r} t_i.X_i = t_{i+1}.X_i$ 为真（其中 $t_{r+1}=t_1$）

下面用一个引理形式化地表述 map 过程的功能：

引理 4.1：map 过程被执行后，向量 V_1,V_2 的映射簇与 R_k 中的元组存在如下联系：

$u\in[V_2]_i \Leftrightarrow \exists\ t_k\in R_k$ 使得 $t_k.D_2 = u \wedge t_k.D_1\in[V_1]_i$

这里 u 表示属性 D_2 的某个值.

定理 4.1：若 $t_1\in R'_1$ 且 $t_1.X_1\in[A_1^*]_i$，则 $t_1.X_r\in[A_r^*]_i$

证明：因为 $t_1\in R'_1$（即 t_1 满足查询条件）

所以 $\exists\ \langle t_2,\cdots,t_r\rangle\in R_2\times\cdots\times R_r$ 使得

$t_1.X_1=t_2.X_1\wedge t_2.X_2=t_3.X_2\wedge\cdots\wedge t_{r-1}.X_{r-1}=t_r.X_{r-1}\wedge t_r.X_r=t_1.X_r$

$t_1.X_1=t_2.X_1\wedge t_1.X_1\in[A_1^*]_i \Rightarrow t_2.X_1\in[A_1^*]_i$

$t_2.X_1\in[A_1^*]_i \Rightarrow t_2.X_2\in[A_2^*]_i$（由引理 4.1）

即如果 $t_1.X_1\in[A_1^*]_i$ 则 $t_2.X_2\in[A_2^*]_i$

同理，若 $t_k.X_k\in[A_k^*]_i$ 则 $t_{k+1}.X_{k+1}\in[A_{k+1}^*]_i (K=1,2,\cdots,r-1)$

递推的结果便是：$t_r.X_r\in[A_r^*]_i$.

又因为　　$t_r.X_r=t_1.X_r$　　所以 $t_1.X_r\in[A_r^*]_i$　　【证毕】

定理 4.2：若存在 i 使得 $t_1.X_1\in[A_1^*]_i$ 且 $t_1.X_r\in[A_r^*]_i$，则 $t_1\in R'_1$.

证明：因为 A_r^* 是由 R_r 中属性 X_r 的一部分值构成的

所以 $t_1.X_r\in[A_r^*]_i \Rightarrow \exists\ t_r\in R_r$ 使得 $t_r.X_r = t_1.X_r$

$t_r.X_r\in[A_r^*]_i \Rightarrow t_r.X_{r-1}\in[A_{r-1}^*]_i$（由引理 4.1）

同理：$t_r.X_{r-1}\in[A_{r-1}^*]_i \Rightarrow \exists\ t_{r-1}\in R_{r-1}$ 使得 $t_{r-1}.X_{r-1} = t_r.X_{r-1}$

$t_{r-1}.X_{r-1}\in[A_{r-1}^*]_i \Rightarrow t_{r-1}.X_{r-1}\in[A_{r-2}^*]_i$

按上述过程递推，可有：

$\exists\ \langle t_2,\cdots,t_r\rangle\in R_2\times\cdots\times R_r$ 使得

$$t_2.X_2=t_3.X_2\wedge t_3.X_3=t_4.X_3\wedge\cdots\wedge t_{r-1}.X_{r-1}=t_r.X_{r-1}\wedge t_r.X_r=t_1.X_r \quad (1)$$

并且 $t_2.X_1[A_1^*]_i$

由于 $[A_1^*]_i$ 只有一个元素 $t_1.X_1$

所以 $t_2.X_1=t_1.X_1$ (2)

由 (1)(2) 可得：$t_1\in R'_1$.　　【证毕】

由定理 4.1、4.2 和 delete 过程的功能，可得：

求解方案执行到第(4)步时，R_1 被完全化简.

定理 4.3：若存在 i 使得 $t_m.X_{m-1}\in[A_{m-1}^*]_i$ 且 $t_m.X_m\in[A_m^{*2}]_i$，

则 $t_m\in R'_m (m=2,3,\cdots,r)$.

定理 4.4：若 $t_m\in R'_m$，则存在 i 使得 $t_m.X_{m-1}\in[A_{m-1}^{*1}]_i$，$t_m.X_m\in[A_m^{*2}]_i$，$(m=2,3,\cdots,r)$.

限于篇幅，以上两个定理的证明从略，读者可仿照定理 4.1、4.2 的证明自行给出.

由定理 4.3、4.4 和 select 过程的功能，可得：求解方案执行完后，$R_m(2\leqslant m\leqslant r)$ 被完全化简.

因此，本文提出的循环查询求解方案是正确的.

结论：本文提出的循环查询求解方案所需的相邻结点通信次数为 5n－4 次(n 为查询图中的结点数)：执行第(1)步需 2n－2 次半连接，第(3)步需 n 次数据传送，第(6)(7)步各需 n－1 次数据传送，共需 5n－4 数据传送.

文献[1]证明了对循环查询，在最坏情况下，半连接策略需 mn 次数据传送，其中 m 是某关系的元组数，本文的方案所需数据传送次数与元组数无关，使查询优化的效率明显提高.

从局部处理代价来看，局部处理过程 map 和 delete 的时间复杂度为 O(MN)，其中 M,N 分别是该过程所搜索的两个向量的长度. 在半连接简化策略中，也需要搜索两个向量(属性向量及本地关系)，时间复杂度也是 O(MN)，因此 map 和 delete 过程并没有增加局部处理代价. select 过程需要搜索三个向量：一个本地关系和两个属性关系，时间复杂度为 $O(MN_1N_2)$，其中 M,N_1,N_2 分别为三个向量的长度.

本文提出的循环查询求解方案，在增加不多的局部处理代价的情况下，用 5n－4 次相邻结点的数据传送，能完全化简循环查询中的全部关系，查询优化的效果是突出的

参考文献

[1]P. A. Bernstein and D. W. Chiu ,Using Semi－joins to Solve Relational Queries, JACM Jaunary 1981.

[2] 须德、徐悦，分布式查询处理，《计算机研究与发展》，No. 1，1989.

[3]N. Goodman and O. Shmueli ,Transforming Cyclic Schemas into Trees, Proc. of 1st ACM SIGACT－SIGMOD Symp. on PODS, 1982.

[4]Y. Kambayashi and M. Yoshikawa ,Query Processing for Distributed Database Using Generalized Semi－joins, Proc. ACM SIGMOD Inter. Conf. on Management of Data,1982.

[5]Ozkarahan,E. A. Schuster,S. A. Sevcik, K. C. ,Performance Evaluation of a Relational Association Processer, ACM Trans. Database Syst. 2,2 (June 1977).

[6]Y. Kambayashi and M. Yoshikawa ,Query Processing Utilizing Dependencies and Horizontal Decomposition, Proc. ACM SIGMOD Inter. Conf. on Management of Data,1983.

[7] 冯志林，分布式数据库系统有回路查询的处理，《计算机学报》，No. 4 ，1989.

中国计算机学会一九九三年学术活动计划表

序号	会议名称	时间	地点	人数	主办单位	承办单位及联系人
1	第一届全国多值逻辑基础理论研讨会	8月	乌鲁木齐	60	多值逻辑专委	南京航空航天大学计算机系 朱梧槚
2	第六届全国固件工程与计算机学术方向研讨会	10月	福州	60	分布计算系统与固件工程专委	上海科技大学计算机系 唐毅
3	第十三届全国计算机维护与管理技术交流会	10月	待定	100	维护与管理专委*	中软公司,北京学院南路55号 谢健
4	全国第二届计算语言学联合学术会议	11月	厦门	150	中国计算机学会*	厦门大学计算机系 李棠秋
5	全国程序设计语言发展与教学学术会议	10月	待定	30	软件专委	南京航空航天大学计算机系 徐宝文
6	第八次全国计算机安全技术交流会	6月	杭州	120	安全专委*	北京95信箱 崔书昆 肖京华
7	"分组交换技术"学术交流会	三季度	待定	100	网络专委	中国邮电器材总公司 付华京
8	数据压缩技术在信息存储技术上的应用	下半年	海南	60	信息存储技术专委	北京自动化工程学院 林定基
9	抗恶劣环境计算机专委第四届学术年会	10月	大庸	150	抗恶劣环境专委	长沙湖南计算机厂 毛京声
10	抗恶劣环境计算机专委工艺与结构学组成立暨学术会	4月	山西	100	抗恶劣环境专委	太原三益公司(785)科协 贾方峥
11	抗恶劣环境计算机专委外部设备学组学术交流会	5月	西安	60	抗恶劣环境专委	西安电子科技大学407室 杨梭
12	'93全国计算机体系结构研讨会	11月	广州	50	体系结构专委	航空航天部706所 程经霖
13	'93全国单片机学术交流会暨多国技术展示会	4月	深圳	200	微机专委单片机学组	广东省单片机学会 (微机专委 王潼)
14	全国第八届微机电路学术会议	5月	大庸	50	微机医药应用学组	青海制药厂 (微机专委 王潼)
15	微机新技术研讨会	8月	乌鲁木齐	80	微机专委	新疆计算机学会 金纪难
16	全国服务信息处理学术会议	10月	上海	100	服务信息专委	清华大学科技产业楼 金国芬
17	国际工控机发展研讨会暨工控机创刊五周年庆祝	10月	南京	100	工控专委*	江苏计算机研究所 王璐璐
18	计算机工程工作站应用技术交流会	三季度	待定	90	应用专委	机电部六所 兰云吉
19	第五届全国容错计算学术会议	8月	北京	100	容错计算专委	航空航天部502所 王东磊
20	中国首届计算机艺术作品展和研讨会	11月	北京杭州	80	学术工委*	学会办,宁伟成 杭州科迪所,娄旻璐
21	计算机名词审定会	上半年	北京	50	名词审定工委	全国名词审定委员会办公室 陈宇飞
22	团体会员信息交流会	5月	郑州	150	组织工委	北京619信箱 王彪
23	专委会工作会议	10月	北京	80	学会办公室、学术工委	清华大学计算中心(学术工委) 金千方
24	地方学会工作会议	11月	广州	80	组织工委、学会办	广东省计算机学会 黄汉良
25	第十届全国青少年信息学(计算机)竞赛	8月	太原	150	普及工委	北京师范大学计算中心 丁丹 山西省计算机学会 杨蔡鑫
26	首届全国计算机安全知识(大奖赛)	3月	北京		安全专委*	北京东长安街14号 刘凤昌
27	知识工程自动推理会议	9月	南京	80	人工智能与模式识别专委	南京大学计算机系 陈世福
28	机器学习会议	9月	四川松潘	70	人工智能与模式识别专委	四川科技大学计算机系 刘国衡

注:*是联合召开的会议或活动