浙 江 大 學

本 科 生 毕 业 论 文(设计)

题目 空间同位模式通用增量挖掘算法研究

姓名与学号 何 琪

指导教师 何 钦 铭

年级与专业 计算机科学与技术

所在学院 计算机科学与技术学院

A Dissertation Submitted to Zhejiang University for the Degree of Bachelor of Engineering

TITLE: A General Approach To Incremental Maintenance Of Discovered Spatial Co-location

Author: QI HE

Supervisor: QIN MING HE

Subject: Computer Science and Technology

College: College of Computer Science

Submitted Date: 2009. 6. 2

浙江大学本科生毕业论文(设计)诚信承诺书`

1.本人郑重地承诺所呈交的毕业论文(设计),是在指导教师的指导下严格按照学校和学院有关规定完成的。

2.本人在毕业论文(设计)中引用他人的观点和参考资料均加以注释和说明。

3. 本人承诺在毕业论文(设计)选题和研究内容过程中没有抄袭他人研究成果和伪造相关数据等行为。

4. 在毕业论文(设计)中对侵犯任何方面知识产权的行为，由本人承担相应的法律责任。

毕业论文(设计)作者签名：

_____年_____月_____日

摘要

随着时间的推移，空间数据集会定期或者不定期的进行新数据的增加，过期数据的删除以及数据的修改，过去挖掘的空间同位模式也会随之发生变化。本文针对空间数据中存在的空间同位模式，提出了一种当空间数据发生更新时的通用的空间同位模式更新算法 IMCP2。与 IMCP 算法相比，该算法不仅能够处理空间数据集中新数据的增加，而且能处理过期数据的删除以及原有数据的修改，适用面更加广泛。IMCP2 算法利用原有数据集的挖掘结果，使更新挖掘尽可能地只针对挖掘数据的更新部分，达到快速更新的目的。实验结果表明 IMCP2 算法，与重新挖掘整个数据集的传统 Join-less 算法相比，在大部分情况下有着显著的速度优势。

关键词 数据挖掘，空间关联规则，同位模式

Abstract

As time goes on, the spatial data set will increase new data, delete or updata old data at regular intervals or sometimes. And the co-location patterns will lost or generate with the change of the spatial data set. In this paper, we propose an algorithm called a general approach to incremental maintenance of discovered spatial co-location (IMCP2) to deal with this problem. Compare with the IMCP, this algorithm can deal with much more general situations. IMCP2 use the result coming from the old co-location data mining to reduce the worthless time which will improve the effect of incremental co-location data mining. The experiment show the performance of the IMCP2 that it will work much better than the traditional method of Join-less which mining the whole data.

Keywords Data mining, Spatial Association Rule, Co-location Patterns，

目录

第1章 绪论

1.1 课题背景

数据挖掘(Data Mining)，是从大量数据中获取有效的、新颖的、潜在有用的、最终可理解的模式的非平凡过程[1]。其从存放在数据库，数据仓库或其他信息库中的大量的数据中挖掘有用的，可被理解的知识的过程。空间数据挖掘(Spatial Data Mining)是数据挖掘的一个重要分支，用来寻找空间数据之间，空间与非空间数据之间内在关系，帮助人们更好的理解空间数据。空间数据挖掘的对象是空间数据库或空间数据集，其区别于传统的离散型数据库[2]。空间数据之间的联系往往被隐藏在了连续的空间之中，需要使用者用特殊的条件去定义这些关系。地理学第一定律表明在空间中某个位置发生的事件总是与它附近发生的事件有关联，并且相距近的事物之间的联系一般比相距远的事物之间的联系要紧密，因而往往使用空间距离来定义空间数据之间的联系。由于近年来空间数据的迅速增加，以及空间数据库的广泛使用，空间数据挖掘在日常生活中也变得越来越重要，其在是生物界，地质学界，交通，海洋分布方面，都有着广泛的运用。

空间数据挖掘与传统的离散型数据挖掘有着很大的区别，这主要由空间数据的复杂性所决定。空间数据之间存在大量非线性关系，且空间数据点之间的关系被隐藏在连续的空间区域里，需要通过特定的方式去定义空间数据关系[15]。利用空间数据挖掘技术，从大量的空间数据中提取未知的并潜在有用的知识，非显式存在的空间关系以及其它我们感兴趣的模式等，从而帮助人类更好的利用空间数据。空间数据知识主要包含，空间几何知识，空间分布规律，空间关联规则，空间聚类规则，空间演化规则，空间特征规则等。其中空间关联规则挖掘主要用于发现不同事件之间的关联性，即寻找同时发生的多种事务的内在联系。关联规则挖掘的重点在于快速发现那些有实用价值的关联发生的事件。一个关联规则可以特征化为两个参数：支持度(support)和置信度(confidence)[4]。由于关联规则挖掘主要用于分类属性，对于离散数据集有很好的表现，但在空间数据集中使用却有着较大的困难，因为我们将空间数据分类，即离散化有很强的随意性，也有巨大的风险。

空间同位模式挖掘作为空间关联规则挖掘[7][8][9][10][16][17]的重要部分，其与传统的关联规则挖掘有着很大区别。在空间数据集中，没有事务的概念，事件之间的联系被隐藏在了连续的空间之中。且空间数据库中同位模式的规模往往会比较小，规模远小于非空间情况下的项数。在典型的离散型关联规则挖掘过程中，如零售业关联规则挖掘，上万的规模项集的情况非常普遍，而对空间数据集来说，同时挖掘的空间项一般不会多余百个。因而候选集生成的代价不再成为支配因素，取而代之的是实例的寻找的计算代价占主导地位，所以空间同位模式挖掘算法往往注重于减少需要寻找的实例，或者减少寻找实例的时间代价。

最初的空间关联规则研究是基于传统的关联规则，但是进一步研究发现传统的关联规则算法框架并不适用于空间数据。有学者进一步提出了空间同位模的概念，并且做了相应的研究。空间同位模式挖掘是空间数据挖掘中的重要部分，其旨在寻找空间特征的子集，它们的点作为实例频繁地在一起出现，如一些生物的群落信息，土壤的分布特征等等，这样的子集被我们称作同位模式[3]，人们希望可以通过学习这些模式，发现空间数据中隐藏的一些有用的知识，以及关联信息。因为空间数据没有明确的事务概念，传统的关联规则的定义并不适用于空间数据挖掘。空间同位模式充分考虑到了空间数据之间的联系，更适用于空间数据，因而其作为空间关联模式挖掘的重要分支被广泛接受。

由于空间数据库往往是庞大复杂的，空间同位模式挖掘算法经常需要对海量数据集进行挖掘。数据量的庞大性导致挖掘将会是一个非常耗费时间的过程。但空间数据集本身却定期或者不定期的更新，当空间数据集产生变化时，如每次都需要重新对所有数据集进行数据挖掘，这样的时间开销是巨大的[18]。为了解决这一问题，我们希望通过挖掘更新部分以及保留一部分原有挖掘信息，对空间数据实现增量挖掘，这样可以极大的提高时间效率。

1.2 相关工作

关于空间关联规则挖掘算法首先在[4]中被提出来，但是该框架中的关联规则必须实现指定一个参考特征。随后[6]中提出挖掘频繁邻域特征子集的概念，并且采用直接计数的方法定义支持度，但是该算法在挖掘过程中会丢失一些空间同位模式的实例。

空间同位模式通用框架以及Join-Based算法寻找完整的空间同位模式最早在[3]中被提出，该算法建立在 Apriori-Gen[10]的基础上，通过产生候选集的方式大大减少了生成的空间同位模式的规模。而后在[3]的框架上，提出了 Join-less[5]和 Partial-Join[11]的算法，来改进 Join-Based 算法，Join-less 使用查表替代 Join 生成团实例，极大地提高了算法效率。

在[12]中首次提出基于传统的离散数据关联规则挖掘增量更新算法 FUP，该算法解决了离散数据库中关联规则挖掘质量更新问题，随后又在[13]提出了通用的关联规则挖掘增量更新算法 FUP2，该算法改进了 FUP，使其可以对离散数据集的增加，减少以及更新进行处理。

在[14]中首次提出了空间同位模式增量挖掘算法 IMCP，该算法建立在[3]的框架上，以 Join-less[5]为基础，对原始数据集以及空间数据集进行星状域划分，增量挖掘完整的空间同位模式。

1.3 本文主要工作及意义

空间同位模式挖掘不同于离散数据集的关联模式的挖掘，由于空间数据的连续性，每一次增减数据集，都可能会带来整个规则的变化，因此传统的处理离散数据集的增量关联模式挖掘的方法，很难运用到空间同位模式挖掘中来。在传统的离散数据的数据库上，事务是不相交的，所以很自然地可以把数据库划分为原始数据集和增量数据集。通过分治算法，重复使用已经有的原始数据库的信息会变得自然。但是与其不同的，空间特征的事件分布在连续的空间中，且共享大量的空间关系。当新的事件引入，他们可能通过存在的事件产生新的关系，这导致了在划分数据集时一些不明确的问题的产生[19]。当数据集发生改变的时候，我们希望可以通过计算增量部分，更少地计算原有的关系，从而达到减少时间开销的目的。

本文主要针对上述问题，提出一种通用地更新空间同位模式的方法 IMCP2，该算法可以处理空间数据的增加，删除以及更新等多种情况，该算法在保证完整性的同时，在时间上有着显著的优势。本文主要工作就是提出了并且实现了 IMCP2 算法，并与传统算法进行实验结果的比较分析。

1.4 本文组织结构

本文第二章主要描述了空间同位模式挖掘基本框架以及对同位模式增量算法做了一些基本的定义和简单假设。第三章描述狭义的同位模式增量更新算法，即减量数据集为空。第四章描述同位模式减量更新算法，第五章描述空间同位模式通用增量更新算法。第六章描述了总结了本文工作以及提出了一些未来的研究方向。

第2章 空间同位模式挖掘问题描述

2.1 空间同位模式挖掘

在空间数据集中，常常会有一些特征，它们的事件频繁地在一起出现，这些关联性很强的特征形成的规则被我们称为空间同位模式(collocation patterns)，例如生物学家希望了解某一个区域内生物之间的共生关系，这就是一个典型的空间同位模式挖掘问题。

狭义的空间同位模式是那些在几何空间中某些特征的事件(event)频繁的伴随出现。当这种伴随出现的事件累计超过人为设定的阀值的时候，我们认为这些特征之间存在某种必然的联系，把这些特征的集合称为空间同位模式，如图 2-1 中+和 x 以及 O 和Δ经常伴随出现，那么它们可能就是一种空间同位模式。

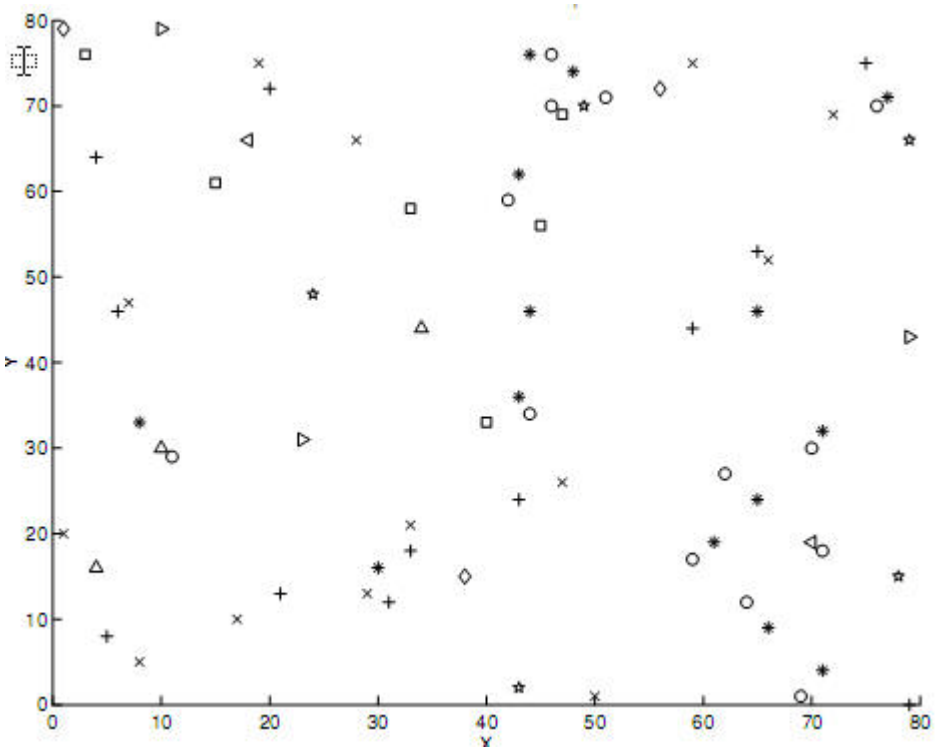

图 2-1 空间点分布示例

给定一组空间特征 F，该空间特征的实例 E，以及 E 上的相邻关系 R，空间同位模式挖掘的目标是去寻找形如 C1 –>C2 (p，cp)的规则，$C_1 \subseteq F$，$C_2 \subseteq F$ 且 $C_1 \cap C_2 = \emptyset$，使得 Pi(C)大于事先取定的阈值。作为重要概念下面简单介绍一下 Pi 的计算方法：

Row instance：I 是同位模式 C 的 Row instance，当且仅当，I 含有 C 中的所有特征，并且 I 的任意子集不满足这样的条件。图 2-2 中，对于同位模 C= (A，B，C)，I = (A1，B1，C1)是 Row instance，而(A1，B1，C1，C3)不是，因为其子集(A1，B1，C1)满足这样的特性。

Table instance：同位模式 C 的 table instance 即其所有 Row instance 的集合。

Participation ratio：$Pr(C, f_i) = \frac{\pi fi\,(|table-instance(C)|)}{|table-instance(fi)|}$

如图 2-2 中 C=(A，B)，I={{A1，B1}，{A2，B4}，{A3，B4}}，因而 Pr({A，B}，A)= 3/4 =0.75。

Participation Index (参与度)及阈值：

同位模式 C 的 Pi 为其 Pr 中最小的值，$Pi = \min(Pr(C, f_i))$，当某个同位模式 C 的流行度大于人为设定的流行度阈值时称其满足要求，并把它作为一个同位关系记录在数据库中。

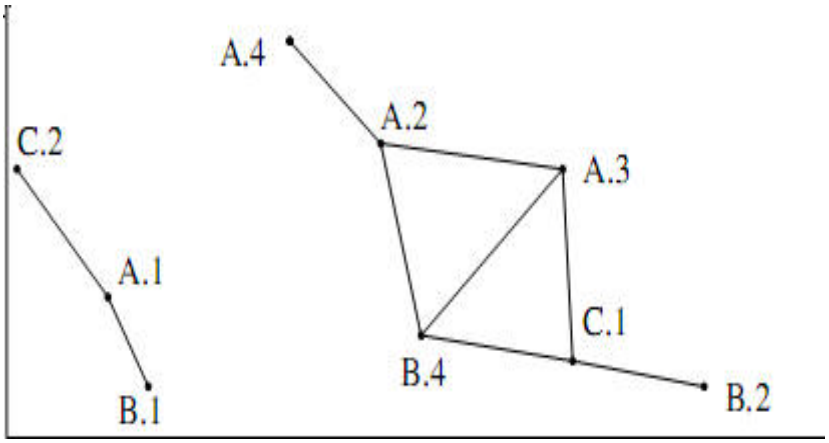

图 2-2 空间点分布示例

空间同位模式挖掘算法 Join-less

本文以 Join-less[3]算法为基础，利用 Join-less[3]的框架挖掘增量更新数据部分。因而本小节简单介绍一下该算法，Join-less 算法与 Join-based 算法相比，旨在减少发现实例的代价，即整个空间同位规则挖掘的时间瓶颈，从而减少算法的时间代价。Join-less 引入了星状域(Star Neighborhood)的概念，主要的思想是根据每个点的小范围邻域，来划定区域，为每个点建立一个这样的星状域估计其实例的上限。一般以人为设定的阈值距离为半径，从而保证了某点星状域内其它点与该点之间的关系，下面简单介绍一下星状域的概念。

给定空间数据集 S 以及特征集合 $F = \{f_1, f_2, f_3\ldots..., f_n\}$

星状域(star neighborhood): 给定空间数据 $O_i \in S$，以及其特征 $f_i \in F$，O_i 的星状域 $SN = \{O_J \in S \mid O_J = O_i \vee (f_j > f_i \wedge R(O_J, O_i))\}$，如图 2-3 所示，以点为单位进行划分，A.1 的星状域为{A.1，B.1，C.1}，其他点的星状域如表 2-1 所示。

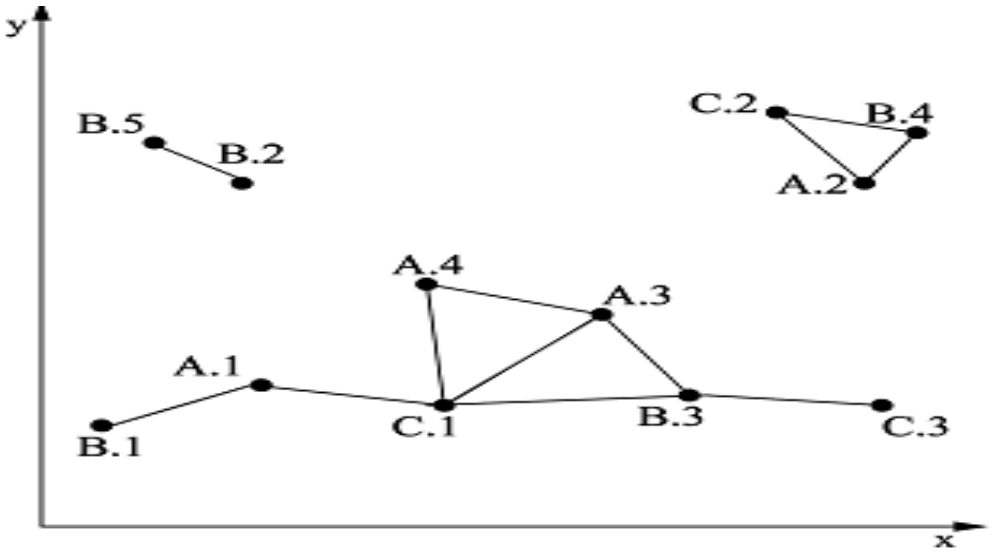

图 2-3 空间点分布示例

表 2-1 图 2-3 对应星状域

feature	Center-object	Neighbor objects
A	A.1 A.2 A.3 A.4	A.1，B.1，C.1 A.2，B.4，C.2 A.3，B.3，C.1 A.4，C.1
B	B.1 B.2 B.3 B.4 B.5	B.1 B.2 B.3，C.1，C.3 B.4，C.2 B.5
C	C.1 C.2 C.3	C.1 C.2 C.3

在算法生成候选集之后，扫描星状域计算 Pi，从而达到估计 Pi 的上限目的，如果该过量估计的 Pi 值小于阈值，直接丢弃该模式，否则对扫描得到的实例一次查表，计算准确 Pi 值。

显而易见，这样的剪枝是有效的，因为该剪枝的代价仅仅是扫描对应的星状域以及初始生成星状域，如果剪枝无效，通过查表的过程来舍弃那些过量计算的实例，保证计算的准确性，从而生成完整的空间同位模式集。例如考察同位模式(A，B，C)，如表 2-1 所示，扫描 A 的星状域，得到实例集 (A1，B1，C1)，(A2，B4，C2) ，(A3，B3，C1)，并且为其计算 Pi 值，如其 Pi 值小于阈值，直接丢弃该同位模式，否则进行查表操作，通过查表发现(A1，B1，C1)，并不是真正的实例，舍弃那些无效的实例后，再一次计算 Pi 值，以此寻找同位模式。由[5]中实验表明该算法主要的时间消耗在于查表过程，比起传统的 Join-Based 算法有着显著的优势。

2.2 同位模式增量更新

在同位模式数据挖掘中，当原始数据集发生改变，新的数据增加或者是原始数据的删除，更新时，重复挖掘更新后的数据集的代价会变得巨大，由于更新的频繁发生，我们需要一种有效的算法去处理这样的问题。对于处理离散数据集增量更新问题的 FUP2 算法在这里并不适用，因为空间数据无法获得直接获得事务，其空间数据关系被隐式地包含在了连续的空间之中。同位模式数据挖掘

中候选集生成的代价不再是支配因素，取而代之的是实例的寻找的计算代价占主导地位。因而要尽可能的减少需要计算实例的同位模式，从而有效的提高算法效率。下面我们系统的描述该问题，并且对于保留的数据进行说明。

给定：

1) 特征集合 $F = \{f_1, f_2, f_3 \ldots f_n\}$
2) 空间数据集 $S = S_1 \cup \ldots \cup S_n$，这里 S_i 是属于特征 f_i 的点的集合
3) 空间相邻关系 R
4) 最小流行度阈值 min_prev 以及最小条件可能阈值 min_cond_prob
5) 更新后数据集 I，原有数据集 O，增量数据集$\triangle^+$，减量数据集$\triangle^-$，且有$\triangle^- \subseteq O$，$I = (O - \triangle^-) \cup \triangle^+$

寻找:

数据集 I 中所有参与度 Pi > min_prev 的同位模式 C 的集合

目标：

查找正确且完整的空间同位模式的集合

算法假设流行度的阈值 min_prev 以及条件概率阈值 min_cond_prob 不变。令删除后的数据集为 D，则有 $D = O - \triangle^- = I - \triangle^+$，令 L 为原始数据集中同位模式的集合。

该算法采用 Join-less 算法[1]框架，利用 Pi 值去评价同位模式流行度是否达到阈值。由于 Pi 值的特有的非线性性，无法简单使用类似于 FUP 算法的相加或者直接相减的方式来计算，如图 2-1 所示 C=(A，B)，I={ {A1，B1}，{A2，B4}，{A3，B4} } Pr({A，B}，B) = 2/5 = 0.4。这里 B4 参与了 2 次(A，B)中的实例，根据 Pr 的定义，其贡献的计数只有 1。如果简单相加，会导致过量计算的问题。为了避免这个问题，在计算增量时，必须知道某实例的参与点是否在前面被计数过，保留该同位模式下所有已经计算过的实例，而不是仅仅保留同位模式的 Pi 值。对于那些不存在于原始数据集中的模式，该模式的 Pi 一定没有达到阈值，利用该信息，可以有效的进行大量剪枝操作，丢弃增量计算中大部分产生的非有效模式。因而在计算同位模式之前，需要将其实例按是否属于增量部分进行分类。也正因为如此，本算法采用了 Join-less[1]作为框架，其天然的星状域的划分，很好的支持了算法的需要。

数据保留：

If (Pi(C)>阈值) then remain all Pi(c) and clique(c)

当数据变化时，分析上一次的同位模式挖掘中留下的数据，需要保留的部分。假设增量数据集，远少于原数据集，因此新引入的数据，只能较小的改变同位关系，而不会对同位关系产生颠覆性的变化，即原来的同位关系全部被否定，而产生大量新生的同位关系。因此对于我们来说前一个挖掘过程计算过的最后又被证明是同位关系的，它们的 Pr 值有很大的价值，而且那些被生成的实例(Instance)，也很有可能会再一次被用到。因此需要记录下这一部分的数据。

If (Pi(C)<阈值) then remain only Pi(c)

被计算过实例(Instance)最后由于 Pi 小于阈值舍弃的候选同位模式，保留其 Pi 值，放弃那些 Instance，因为有很大的概率不用再次计算，保留其 Instance 价值不大，只记录其 Pi 值，做一个提前的预测。

对于那没有计算过的关系，不管是因为新增加元素带入的，或者是本来就没有进行过计算的，都需要运用 Join-less 重新计算，没有任何与他们相关的信息，来进行剪枝，或者预估计。

第3章 空间同位模式增量更新算法

3.1 背景概述

数据集的频繁更新一直是数据挖掘中一个重要的问题。特别是对于空间数据，数据更新会变得更加频繁，比如生物关系挖掘中，生物的迁徙，生物的出生等带来的数据变更，城市建筑物布局同位模式挖掘中，建筑物的增加带来的新增数据集的问题，为了解决这些问题，提出一种空间同位模式增量更新算法，将变得非常有意义。由于少量数据更新带来的同位规则变更，可以使用下文所述的算法，有效地减少二次重复挖掘带来的时间代价。

3.2 基本定义与问题描述

本章主要介绍了增量更新算法，该算法只针对数据集的增加，即$\triangle^{-}=\varnothing$。

在传统的离散型数据库中，事务是不相交的，因此可以将数据集简单的划分成为增量和原始部分，然后使用分治算法处理。但是在空间数据集中，点之间的关系被隐藏在了连续的空间之中，所以我们首先要将这些关系进行分类，在进行增量挖掘。并且如 2.2 中所述，由于空间数据挖掘的衡量因子 Pi 并非是线性的，不能进行简单的相加计算，否则会导致过量计数的问题。

保留数据：算法保留所有原始数据挖掘中计算过的同位模式的 Pr 值，以及保留那些流行度大于阈值同位模式的团实例。

定义 1：给定空间上的对(pairwise)，如果构成该对(pairwise)的两个点均属于原始数据集，即 $E_i \in O$，那么称其为**原始对**，否则称其为**增量对**。

如图 3-1 所示，图中•表示增加的数据，而★表示原始数据，(B_1, C_1) 为原始对，而(A_2, B_1)为增量对。

定义 2：增量对中的参与点，如果一个属于增量数据集，另外一个属于原始数据集，那么我们称该增量对为**内部对**。

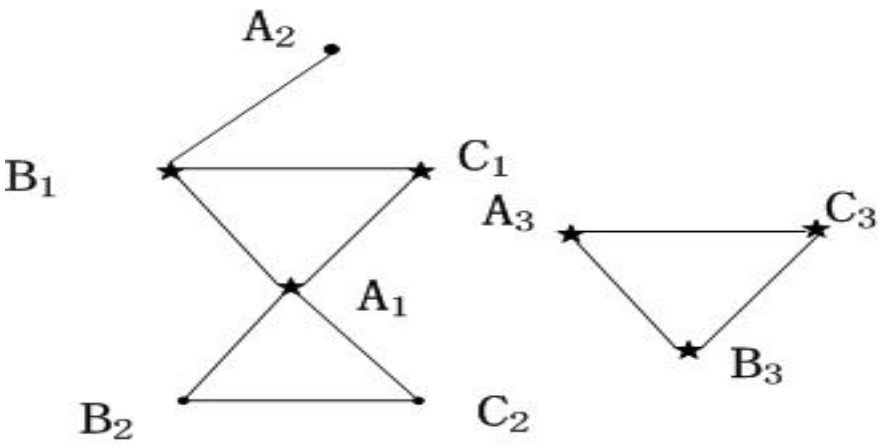

图 3-2 空间点分布及增量数据示例

定义 3：给定更新数据集中的事件 e，如果它至少参与了一个内部对我们就称其为跨越事件，否则称为非跨越事件。

定义 4：对于一个对，如果它由两个跨越事件组成，其为跨越对(Cross pairwise)否则非跨越对

定义 5：原始星状邻域是从原始对中产生的星状邻域，而**增量星状邻域**是从增量对中产生的星状邻域。

定理 1：通过扫描**原始星状邻域**和**增量星状邻域**，可以得到完整的团实例
任意团实例都可以划分成两部分
1. 参与该团的所有点均属于原始数据集 O 称为原始团。
2. 存在某个点属于增量数据集$\triangle^{+}$，称为增量团。

任何原始团均可以通过遍历原始星状邻域得到，而任何增量团都可以通过遍历增量星状邻域得到。所以遍历原始星状邻域和增量星状邻域可以得到完整的团实例。

定理 2：给定一个在原数据集上流行度小于阈值的空间同位模式 C={f_1，…，f_g}，如果对于增加的数据集 C.max_pr <min_prev ，那么 C 在更新后的数据集上流行

度也是小于阈值的。

证明：如果在新的数据集上 C.max_pr< min_prev，那么每个 f_i (1<=i<=g) 都有 Pr(C，f_i) = (ni / mi)<min_prev (ni 是 f_i 在新数据集上参与度，mi 是所有的 f_i 的个数)因为在原始数据集上 f_i 的参与度(no/mo)小于 min_prev，因此最终的 f_i = (ni+no) /(mi+mo)也小于 min_prev。因此，流行度小于阈值。

3.3 算法

图 3-2 给出了本算法的核心流程,其分别处理增量数据集集以及原始数据集，以 C ∈Li 分类计算，如果不满足 C ∈Li，计算增量数据集，如增量数据集小于 min_prev 则丢弃该同位规则，否则扫描原始星状域计算 Pi 值。如果 C ∈Li，则使用保留的团实例进行计算。如 2.2 中所述，因为 Pi 的非线性，无法采用简单计数方法，否则会产生过量或者过少计算的问题，如图 3-1 所示考虑 C=(A，C)，计算实例(A_1，C_1)的时候，显然特征 A 的增量域不需要建立，因为 A_1 早就在(A_1，C_3)中了。但是对于(A_3，C_3)，仍然需要进行计数，因为 A_3 是不属于任何原始数据集中 Instance 的事件，忽略 A_3 会导致少量计数的问题。上述问题会导致过多或者过少的计算参与度，这里使用查表的办法来处理这样的问题。

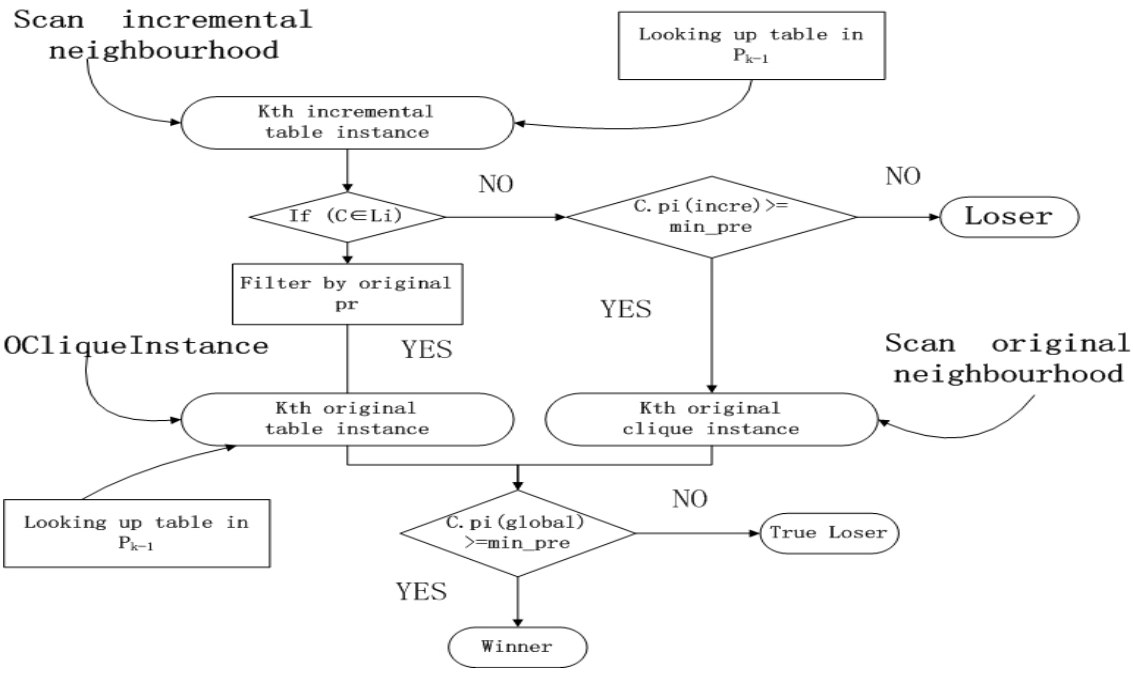

图 3-2 同位模式增量更新算法流程图

输入：ODataset，原始数据集，IDataset:增量数据集，流行度阈值，min_prev

输出：空间同位模式的集合 以及其对应的 Pi

变量：**GP** 全局对(从 ODataset ∪ IDataset 生成)，IP(增量对)，OP(原始对)

IN，ON，分别对应增量，原始星状域

ICI_k，OCI_k 分别从 IN，ON 中获得的团实例

IPN_k，GPN_k，SPN_k 增量，更新后的数据集，数据库中存储的 Pr 值

TIS_k 原始数据中保留的团实例

TCE 所有同位模式的实例集合

GP = Line_sweep(ODataset ∪ IDataset);

OP，IP =calssify(GP);

ON = gen_star_neighborhoods(OP);

IN = gen_star_neighborhoods(IP);

K = 2

1 While not empty P_{k-1}

2 {

3 C_k =gen_candidate_colocations(P_{k-1});

4 ICI_k = apply_join_less(IN，C_k);

5 IPN_k = Calculate_pariticpate_num(ICI_k，TCE);

6 C_k =Filter_candidate_by_maxpr(ICI_k, TCE);

7 C_k =Filter_by_original_pr(SPN_k，IPN_k);

8 OCI_k = apply_join_less(ON，C_k);

9 GPN_k = Calculate_pariticpate_num(ICI_k，OCI_k ，TCE);

10 P_k = select_prevalent_ collocations(C_k ，GPN_k，min_prev);

11 R_k =gen_colocation_rule(P_k);

12 K++;

}

算法流程 Kth：

1. 使用 Apriori 算法产生 candidate(行 3)
2. 对于增量域使用 Join_less(注意：这里仅使用该算法过滤出需要的团实例)。算法(行 4)，如图 3-2 所示从 IN 中过滤出 table_instance，而后去查 K-1 的团实例的表判断其是否是有效的团实例。
3. 如图 3-2 所示，对于那些属于 L 的(L 为原始数据库中流行度大于阈值的同位模式)同位模式，跳过 6，7 行，否则进行 6，7 行。
4. 应用定理 2，对于那些在原数据集上流行度小于阈值的空间同位模式 C={f_1，…，f_g}，如果在新增数据集上也有 C.max_pr <min_prev，那么 C 在更新后的数据集上流行度也是小于阈值的，我们将其标记为 loser (6 行)
5. 对于那些被记录 SPN_k =no/mo 的同位模式，以及新计算增量部分 $IPN_{k=}$ ni/mi，用 pr = (ni+no) /(mi+mo)分别计算，如果 C.max_pr <min_prev，那么 C 在更新后的数据集上流行度也是小于阈值的，我们将其标记为 loser (7 行)
6. 对原始邻域使用 Join_less，并且为全局计算 GPN_k (8，9 行)
7. 用流行度阈值进行过滤，并产生 R_k (10，11 行)

3.4 实验设计与分析

本小节通过实验考察该算法的效率，所有的算法均用 C++实现，在 WINXP 平台下，Core 2 DUO 2.0 Ghz，2G 内存的机器上运行。

数据生成:

根据 S，P，D，F 选择初始同位模式，初始同位模式满足平均维数等于 S，并且分布在 D×D 大小的空间中，个数等于 P，且总特征数等于 F。而后将 D×D 空间划分成 d 边长的正方形，在随机挑选的正方形中为这些同位模式分别产生实例，直到使用的点数达到要求的 95%，多余的点作为噪音加入，这样便完成了原始数据集的生成。而后根据 incre_ratio 以及 cross_ratio 生成增量数据集，取 S * incre_ratio 数目的点，并且其特征属于 F。后以 cross_ratio 为约束，将这些点随机散布到各个正方形中，完成增量数据的生成。

实验结果:

本实验通过限定 S，P，F，D，d，min_prev，观察算法随着 incre_ratio 和 cross_ratio 变化的情况，用 Join-less 进行对比。

表 3-2 实验数据说明

变量	描述	数据取值范围
S	初始同位模式维数	10
P	初始同位模式个数	20
F	总特征数	50
N	原始数据集点数	1000-100000
D	空间大小 D×D	10000
d	相邻距离阈值	10
incre_ratio	增量数据集占原始数据集比例	0-1
min_prev	最小流行度阈值	0.2
cross_ratio	跨越度	0-1

实验 1：限定 incre_ratio 不变为 0.1，N 为 100000，cross_ratio 以每次 0.1 增加，其它参数如表 3-1 所示

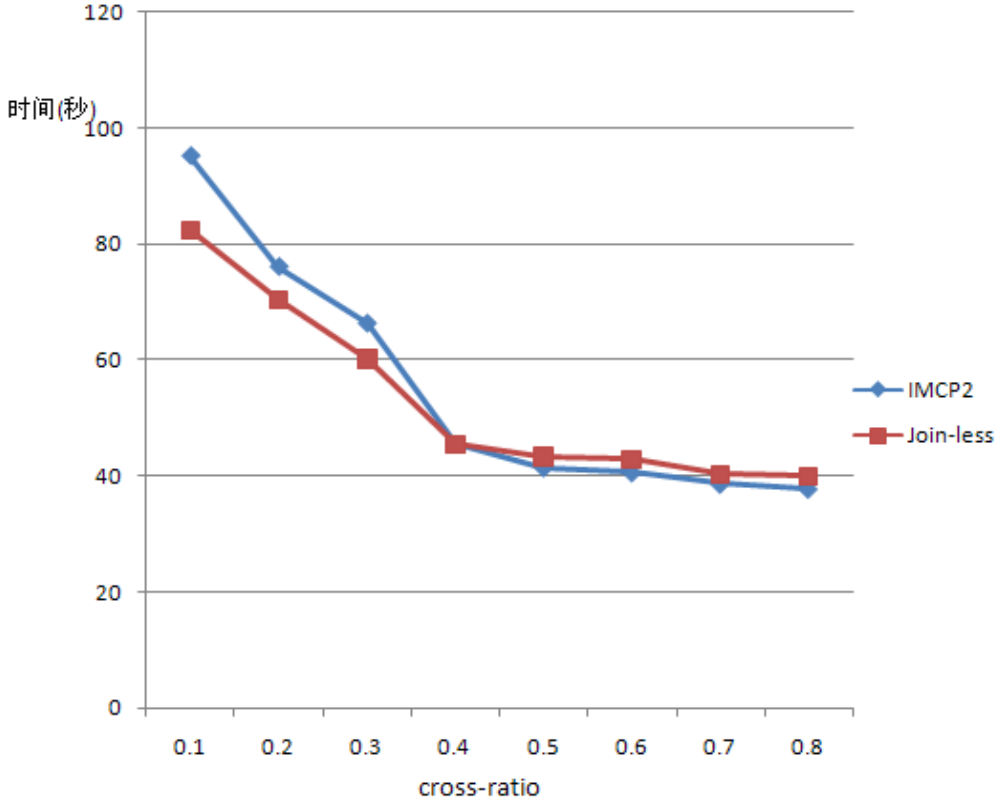

图 3-3 3-1 实验数据

由图 3-3 实验结果可得，本算法与 Join-less 算法相比，随着 cross_ratio 增加，

效果逐渐变差，当 cross_ratio 过大时，同位模式增量挖掘算法将变得比 join-less 算法更差。由定义可知，cross_ratio 越大，那么增量数据集与原始数据集之间形成的团实例就更多，因此需要扫描的增量星状域会变大，而导致剪枝效果变差。但是随着 cross_ratio 增加，Join-less 算法与 IMCP2 算法消耗的时间都减少，因为当 cross_ratio 增加时，空间点将变得单位密集，因而产生的候选集将会变少，且星状域的过滤效果将变得更好。

实验 2：限定 cross_ratio 为 0.2，incre_ratio 为 0.1，N 非线性增长，其余参数如表 3-1 所示。

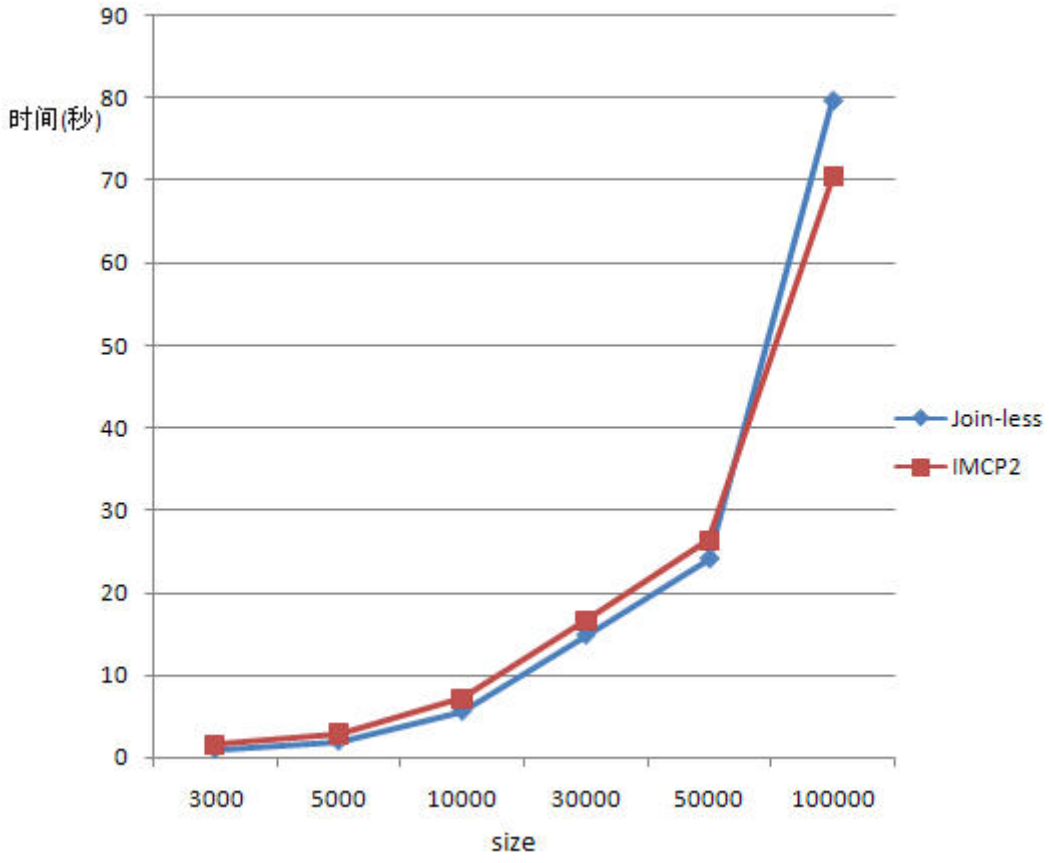

图 3-4　3-2 实验数据

由实验 2 结果图 3-4 可得，随着 N 的增加，即整体数据集规模增加，算法的

效果会变得更好，因为同位模式增量挖掘算法比 Join-less 需要额外的划分星状域的代价，并且要计算 2 次 Pi，因而当点数过少，导致实例过于稀疏时，该额外代价会拖累整个算法的效率。

实验 3：限定 cross_ratio 为 0.2，N 为 100000，根据 incre_ratio 观察结果

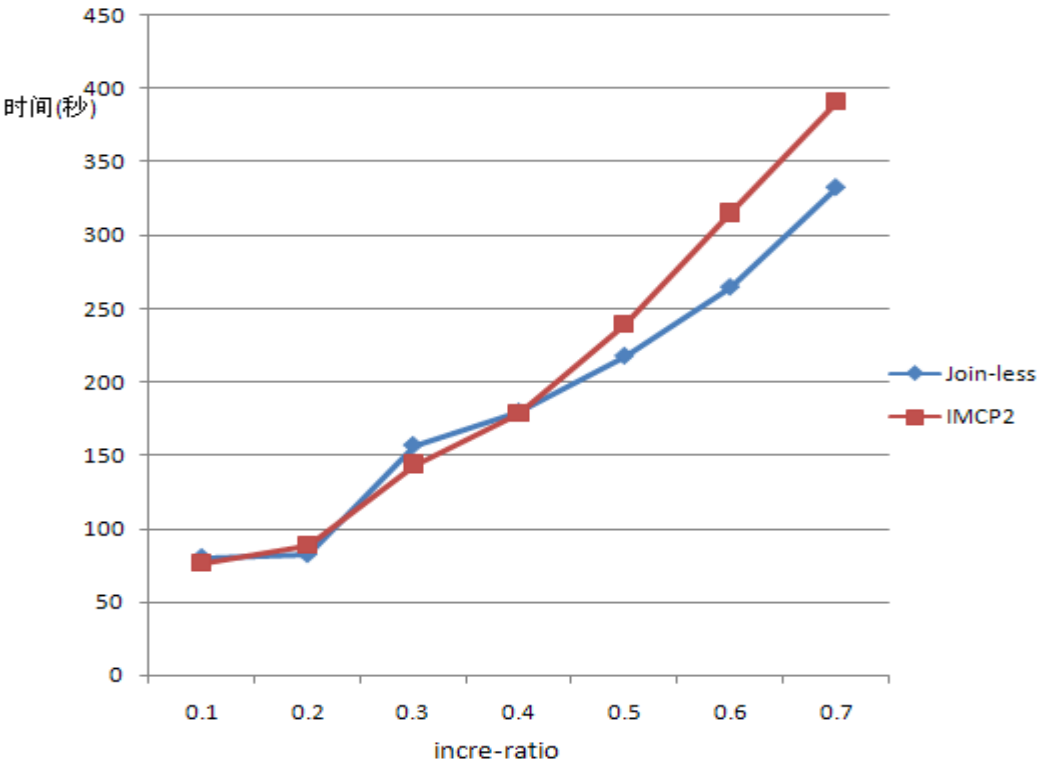

图 3-5 3-3 实验数据

由实验 3 图 3-5 可得，当 incre_ratio 增加时，算法效率会下降，又由实验 2 知，随着 incre-ratio 上升，数据规模会变大，因而会抵消一部分效率下降，甚至导致上升，但是当 incre_ratio 过大时，算法效果下降明显，因为过大的增量数据集会极大的影响同位模式的分布，导致原始数据集同位模式在更新后的数据集中失效或者产生一些本来不存在的同位模式。

3.5 小结

如 3.4 所示，同位模式增量更新算法效率，受增量规模以及 cross_ratio 的限制，因此本算法要求增量规模远小于原始数据规模，实际应用中，增量数据集与原始数据集相比，一般属于较小数据集，因而能满足这样的条件，对于 cross_ratio，由实验可知，cross_ratio 越小算法效率越高，因而本算法更适用于新增数据集处于原有数据的边缘，如地理空间同位模式挖掘中，扩展挖掘范围等情况，这样可以有效的减小 cross_ratio，提高算法效率，并且对于大规模数据集本算法的优势将更为明显。

第4章 空间同位模式减量更新算法

4.1 背景概述

作为空间同位模式通用更新算法的一部分，同位模式减量更新算法也有着重要的意义。虽然空间数据的删除不如增加平凡，但是在实际运用中空间数据的删除也是空间数据集更新的重要组成部分，例如城市建筑物布局同位模式挖掘中，建筑物的拆迁带来的数据集的减少，或者在生物关系同位模式挖掘中，某种生物的大量死亡，以及人们不再关心某种特性，想要去除该特性的影响。当然减量更新算法作为增量更新的一种弥补，它们共同解决了通俗意义下的空间数据更新问题。本章详述了空间同位模式减量更新算法，该算法有效地减少二次重复挖掘带来的时间代价。

4.2 基本定义与问题描述

本章主要介绍了减量更新算法，该算法只针对数据集的减少，即$\triangle^+ = \varnothing$，$\triangle^- \subseteq O$

保留数据：算法保留原始同位模式挖掘中流行度大于阈值的同位模式的团实例和其 PR 值。

定义 1：给定空间上的对(pairwise)，如果构成该对(pairwise)的两个点均属于减量数据集，即 $E_i \in \triangle^-$，那么称其为**减量对**。

如图 4-1 所示，图中•表示减量数据，而★表示剩余数据，(B_2，C_2) 为减量对

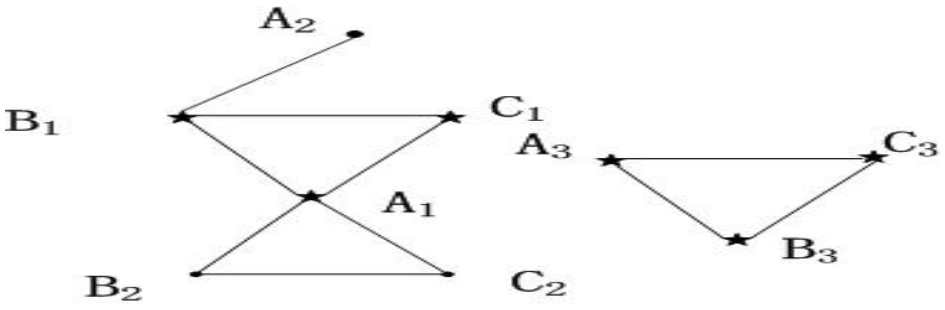

图 4-3 空间点分布及减量数据示例

定义 2：原始星状邻域是从剩余数据集(D = O - △⁻)产生的星状邻域，而**减量星状邻域**是**减量数据**对中产生的星状邻域。

定理 3：给定一个在原数据集上流行度小于阈值的空间同位模式 C={f_1，…，f_g}，如果对于减量数据集其 C.max_pr > min_prev，那么 C 在更新后的数据集上流行度是小于阈值的。

证明：如果在新的数据集上 C.max_pr< min_prev，那么每个 f_i (1<=i<=g) 都有 Pr(C，f_i) =(ni / mi)<min_prev (ni 是 f_i 在新数据集上参与度，mi 是所有的 f_i 的个数)因为在原始数据集上 f_i 的参与度(no/mo)大于 min_prev，因此最终的 f_i = (ni-NO) / (mi-mo)，(NO >= no 因为由于某个点的删除导致有剩余数据集参与的团缺失)，f_i 小于 min_prev. 流行度小于阈值。

由定理 3 可知，与增量算法不同是，减量不需要为了删除某些点而消失的团实例去做额外的划分邻域，减量算法并没有过量计算 Pr 值或者少量计算 Pr 的问题，如 3.3 中增量部分所述。因此不需要额外的维护增量与减量形成的实例，尽管它们是存在的，相反该算法能够直接使用减量部分，这种天然的划分，能够很好的减少算法的复杂度，以及时间消耗。

4.3 算法

算法核心流程如图 4-2 所示，该算法分别处理减量数据集集以及原始数据集，以 C ∈Li 分类计算，如果不满足 C ∈Li，计算减量数据集，如减量数据集 C.Pi 大于 min_prev 则丢弃该同位规则，否则扫描原始星状域计算 Pi 值。如果 C ∈ Li，那么从原始保留团实例中删除不再有效的点后，计算 Pi。减量数据集没有过量计算的问题，如定理 3 所示，因而我们只需要简单的计算减量部分数据即可，与增量更新空间同位模式算法相比，减少了划分数据集的代价，但是本算法需要对减量数据集进行额外数据挖掘，因而可以预计的是当减量数据集规模过大时，该算法效率将显著下降。

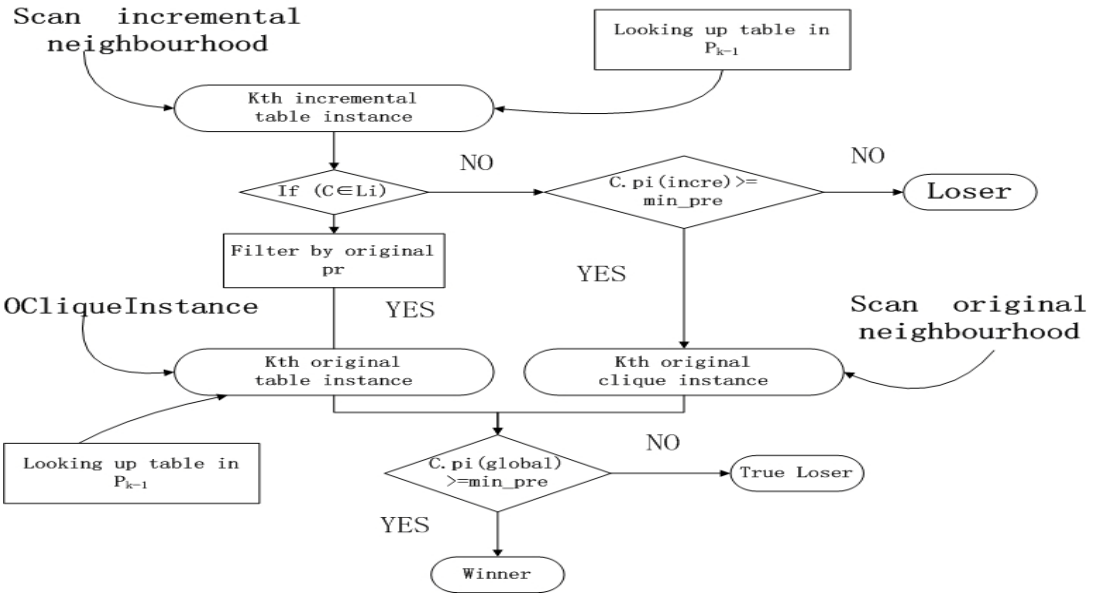

图 4-2 同位模式减量更新算法流程图

输入： ODataset，原始数据集，DDataset，减量数据集，流行度阈值，min_prev

输出：空间同位模式的集合 以及其对应的 Pi

变量：GP 全局对(从 ODataset - DDataset 生成)，DP(减量对)，OP(原始对)

DN，ON，分别对应增量，原始星状域

DCI_k ，OCI_k 分别从 IN，ON 中获得的团实例

DPN_k，GPN_k 减量，更新后的数据集 Pr 值

TIS_k 原始数据中保留的团实例

TCE 所有同位模式的实例集合

DCE 所有减量数据集中同位模式的实例集合

GP = Line_sweep(ODataset-DDataset);

DP = Line_sweep(DDataset);

ON = gen_star_neighborhoods(GP);

DN = gen_star_neighborhoods(DP);

K = 2

1 While not empty P_{k-1}

2 {

3 C_k =gen_candidate_colocations(P_{k-1});

4 DCI_k，DCE = apply_join_less(DN，C_k);
5 DPN_k = Calculate_pariticpate_num(ICI_k，DCE);
6 C_k =Filter_candidate_by_maxpr(ICI_k，TCE);
7 OCI_k = apply_join_less(ON，C_k);
8 GPN_k = Calculate_pariticpate_num(OCI_k，TCE);
9 P_k = select_prevalent_ collocations(C_k，GPN_k，min_prev);
10 R_k =gen_colocation_rule(P_k);
11 K++;
}

算法流程 Kth：

1. 使用 Apriori 算法产生 candidate(行 3)
2. 如图 4-2 所示，对于那些属于 L 的(L 为原始数据库中流行度大于阈值的同位模式)同位模式，跳过 4，5，6 行，否则进入。
3. 对于减量邻域使用 Join_less，如图 4-2 所示从 IN 中过滤出 table_instance，而后去查 K-1 的团实例的表判断其是否是有效，并且计算其 Pr 值。(行 4，5)
4. 应用定理 3，对于那些在原数据集上流行度小于阈值的空间同位模式 C={f1，…fg}，如果在新增数据集上也有 C.max_pr > min_prev ，那么 C 在更新后的数据集上流行度将会小于阈值的，我们将其标记为 loser (行 6)
5. 对原始邻域使用 Join_less，并且为全局计算 GPN_k (8，9 行)，对于那些属于 L 的同位模式，如图 4-2 示，我们删除其邻域中对应的 instance。
6. 用流行度阈值进行过滤，并产生 R_k (10，11 行)

4.4 实验设计与分析

本节通过实验考察该算法的效率，所有的算法均用 C++实现，在 WINXP 平台下，Core 2 DUO 2.0 Ghz，2G 内存的机器上运行。

数据生成:

与增量更新空间同位模式算法类似，生成原始数据集以及增量数据集，而后将生成的原始数据集和增量数据集当作新的原始数据集，并且把增量部分作为减量数据集，进行实验。

本实验通过限定 S，P，F，N，D，d，min_prev，incre_ratio，如表 4-1 所示，

decre_ratio 以每次 0.1 增加。

表 4-3 实验数据说明

变量	描述	数据取值范围
S	初始同位模式维数	10
P	初始同位模式个数	20
F	总特征数	50
N	原始数据集点数	100000
D	空间大小 D×D	10000
d	相邻距离阈值	10
decre_ratio	增量数据集占原始数据集比例	0-1
min_prev	最小流行度阈值	0.2
cross_ratio	跨越度	0.2

实验结果：

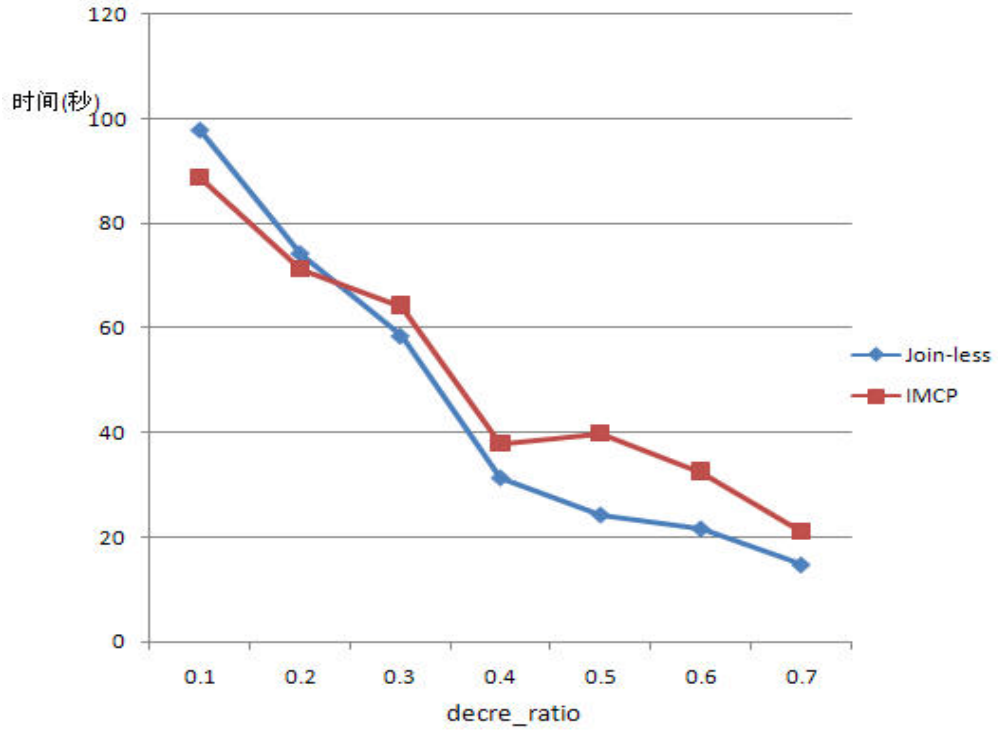

图 4-3 Join-less 与 IMCP 随 decre_ratio 时间对比图

由图 4-3 实验结果可得，本算法随着 decre_ratio 增加，效率逐渐下降，当

decre_ratio 过大时，同位模式增量挖掘算法将变得比 join-less 算法时间消耗更多。造成这样的原因是，如 4.2 所述，本算法需要额外地对减量数据集进行同位模式挖掘，而当减量数据集过大时，则原有数据集规模远小于减量数据集，因而额外剪枝的代价超过了计算该问题本身，导致算法效果下降。

4.5 小结

由 4. 4 可知，本算法适用于减量数据集规模远小于原有数据集的情况，当减量数据规模过大时，该算法的效果将不如重新计算更新后的整体数据集，这种情况在减量数据挖掘中普遍存在。不过对于现实应用，空间数据集的减量数据集比起原书数据集往往属于小规模部分，例如城市建筑物地理位置同位模式挖掘中，拆迁的建筑物均为少量部分，因而本算法适用于实际应用中绝大部分情况，并且其作为空间同位模式通用增量更新算法的一部分，有着重大意义。

第5章 空间同位模式通用增量更新算法

5.1 背景

作为空间同位模式通用增量更新算法，本算法能够处理所有的空间数据集更新问题，而 3，4 章所述狭义增量以及减量空间同位模式更新算法都属于该算法的特例。需要强调的是，由于空间数据集的减量与增量都有着各自的特殊性，因而采用该算法会比分别使用前述的算法处理空间数据集仅存在减量或者增量的算法更为复杂。但是其作为能够同时处理增量与减量的算法，重要性不言而喻，这里将空间数据的更新，看作是减量与增量共同作用的结果，因此该算法可以处理所有的空间数据集更新问题。在实际运用中，空间数据集的减和增量同时存在是最为普遍的一种情况，该算法解决了通俗意义下空间数据更新问题。本章详述了空间同位模式通用更新算法，该算法有效地减少二次重复挖掘带来的时间代价，并且是第一个空间同位模式通用增量更新的算法。

5.2 基本定义与问题描述

本章介绍了通用增量更新算法，该算法能同时处理空间数据集的增量，减量情况，算法要求$\triangle^{-} \subseteq O$，即删除点在原数据集中存在。

本算法作为 IMCP2 算法的核心，是对前面两章介绍的算法的综合应用，因此使用前面所述的所有定义与定理，算法针对数据集更新同时存在增加以及删除的情况。因为增量会造成过量或者少量计数的问题，因而其会影响到减量的剪枝，同时减量也同样会影响增量的剪枝。所以我们要使得它们同时满足，其才为真正的失败者**(loser)**。

由于增量是必须要计算的，而不论减量部分效果如果，因此增量将优先于减量部分计算，并且当增量不满足剪枝条件时，我们不再计算减量部分，并且因为增量数据集的存在，本算法需要额外划分增量域，又因为减量数据集的存在，需要额外计算减量域，因此可以预见本算法的效率比 3，4 章所述单独运用的特殊情况算法处理，有一定差距。

5.3 算法

本算法核心流程如图 5-1 所示，因为减量与增量数据集同时存在，因而本算法需要考虑过量计算问题，如 3.3 中所述，又因为增量部分必须进行挖掘，因而本算法采用先对增量数据集进行挖掘，而后用减量数据集过滤增量挖掘所得到的失败者(Loser)，只有该同位规则在减量数据集中仍然是失败者(Loser)，那么其才为真正失败者(True Loser)，否则该同位模式仍需重新计算。

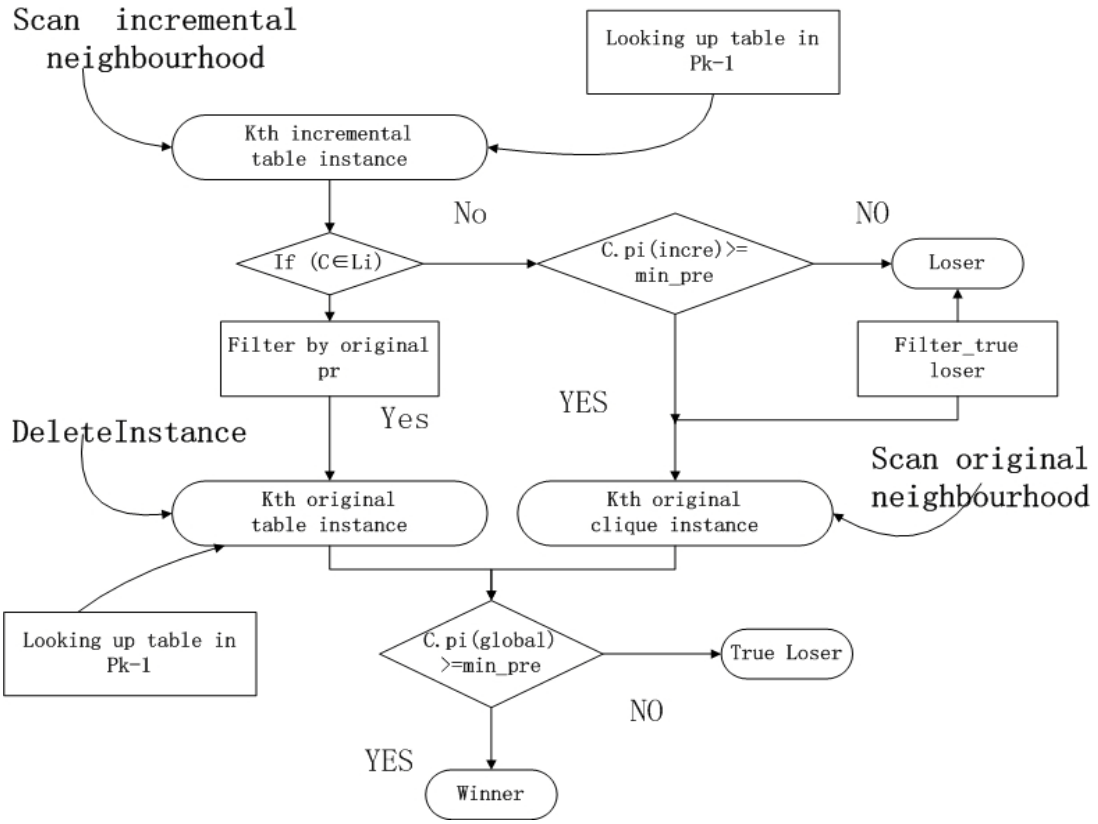

图 5-1 同位模式通用增量更新算法流程图

对比图 3-2，可以发现，当增量数据挖掘产生(**Loser**)时，本算法需要额外通过挖掘减量数据集过滤失败者，即图 5-1 中的 Fiter_true loser 模块，该模块具体流程如图 5-2 所示，对于那些在增量中被计算为(**Loser**)的同位模式，计算其在减量数据集中 Pi，如果 C.pi (incre) >= min_pre 成立，则其为 true_loser，否则其进入计算 Kth_original_clique instance 部分。其他步骤与增量计算相同，这里不再重复叙述。

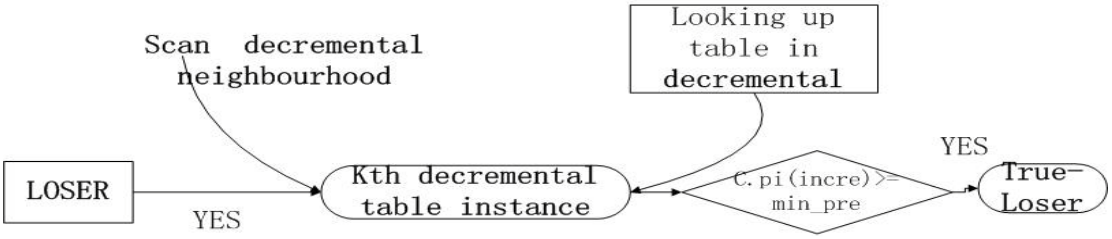

图 5-2 Fiter_true loser 模块流程图

5.4 实验设计与分析

本小节通过实验考察算法的效率，所有的算法均用 C++实现，在 WINXP 平台下，Core 2 DUO 2.0 Ghz，2G 内存的机器上运行。

数据生成:

本实验原始数据集生成及增量数据集生成与 3.4 中所述相同，并且由实验 4.4 可知，减量算法效果仅受 **decre_ratio** 影响，因而从原始数据集中随机挑选一定比例的点，作为减量数据集，进行挖掘，由实验 3.4 可知增量数据集规模 **incre_ratio** 将很大程度的影响算法效率，因而本实验限定增量数据规模为给定值。

表 5-4 实验数据说明

变量	描述	数据取值范围
S	初始同位模式维数	10
P	初始同位模式个数	20
F	总特征数	50
N	原始数据集点数	1000-100000
D	空间大小 D×D	10000
d	相邻距离阈值	10
incre_ratio	增量数据集占原始数据集比例	0.1
decre_ratio	增量数据集占原始数据集比例	0.1
min_prev	最小流行度阈值	0.2
incre-cross_ratio	跨越度	0-1

实验限定 S，P，F，N，D，d，decre_ratio，min_prev 如表 5-1 所示，观察算法随着 **incre-cross_ratio** 性能的变化，incr-cross_ratio 每次增加 0.1。

实验结果：

由图 5-3 可见，该算法受 incre-cross_ratio 影响较大，其随 **incre-cross_ratio** 上升快速下降。因为需要额外的挖掘减量数据集进行过滤，因而本实验结果与仅存在增加数据集的情况相比，算法效率下降更快，其没有想象中理想，因为该算法需要某同位规则同时满足减量与增量数据集中均为失败者**(Loser)**，这样的制约存在，很大程度上降低了剪枝效率。

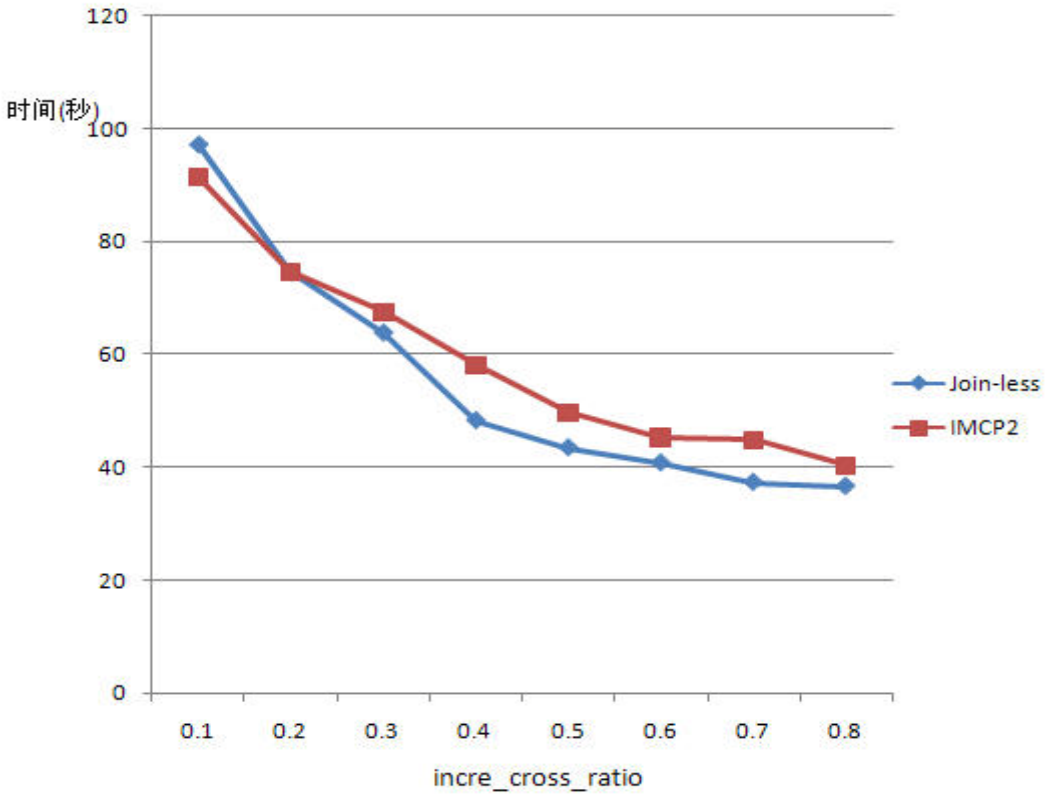

图 5-3 Join-less 与 IMCP 随 incre_cros_ratio 时间对比图

真实数据实验：

本数据取自美国国家交通地理数据库，数据规模 N = 5000，从其中随机挑选

增量数据 500，减量数据 500，保持 incre-cross_ratio 小于 0.2，多次测试实验效果对比。

表 5-2 实验数据说明

变量	描述	数据取值范围
N	原始数据集点数	5000
d	相邻距离阈值	30000
incre_ratio	增量数据集占原始数据集比例	0.1
decre_ratio	增量数据集占原始数据集比例	0.1
min_prev	最小流行度阈值	0.3
incre-cross_ratio	跨越度	0-0.2

实验结果：

本实验结果如图 5-4 所示，其并不理想，主要原因是数据规模过小，额外的划分增量域与计算减量域的时间消耗太大，并且真实数据随机挑选增，减数据集比较难控制 incr-cross_ratio，因此算法效率与 Join-less 差距较大。

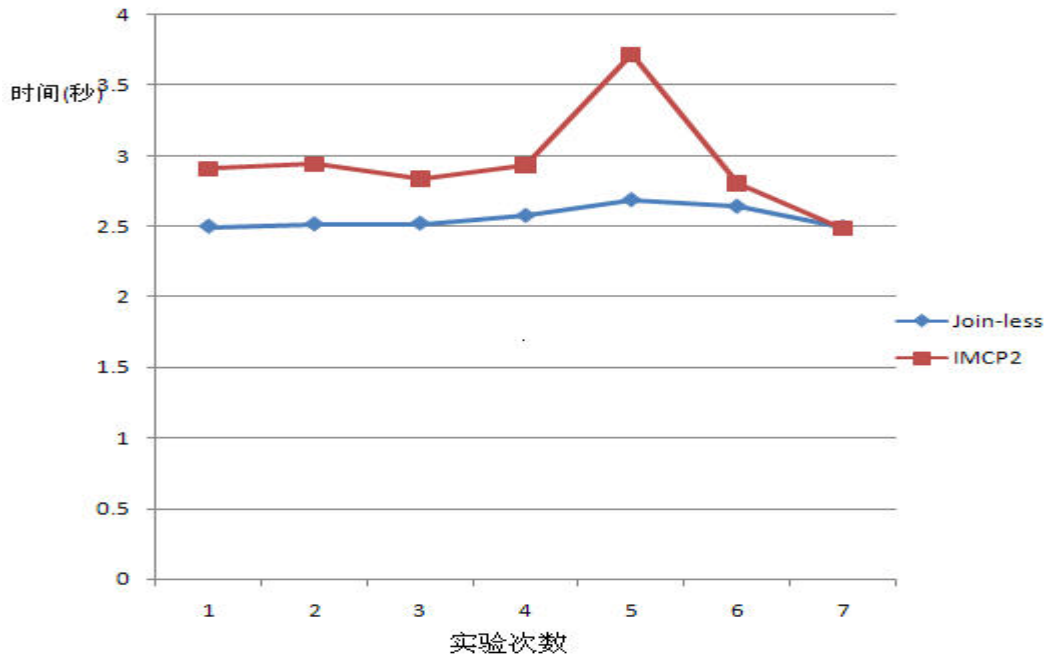

图 5-4 Join-less 与 IMCP 真实数据对比图

5.5 小结

由实验 5.4 可以看出本算法效率随着 **incre-cross_ratio** 增加而变差，当 **incre-cross_ratio** 过大时，同位模式增量挖掘算法将变得比 join-less 算法更差。因而本算法更适用于新增数据集处于原有数据集的边缘，如地理空间同位模式挖掘中，扩展挖掘范围等情况，并且删除数据集较小的情况。并且由于需要同时满足同位模式在增加数据集与删除数据集中均为失败者的条件，因而实际应用中如图 5-4 所示，算法效率并不高。不过其作为第一个能够处理空间同位模式通用增量更新算法，有着重要的研究价值，与探索价值。

第6章 总结与展望

本文研究了处理空间数据集增加，删除以及更新问题的同位模式通用增量更新算法，该算法在更新数据集数据规模较小的情况下，有效地解决了重复挖掘代价过大的问题，与传统的重新挖掘整个数据集相比，有着显著的时间优势。

不过本算法同样存在较大的制约，难以处理更新数据集数据规模较大的情况。且由于需要保留原始数据集同位关系下的实例，该算法对于内存有较大的额外开销。并且其在同时存在增加，删除数据集的情况下，实验优势并不明显，究其原因主要是刻画同位模式流行度的 Pi 值定义的非线性性，如 2.2 所述，使得必须记录原始数据实例进行查表操作，极大的影响了算法效率。

作为未来的工作，可以考虑使用线性定义值替代 Pi 作为阈值，如最新提出的密度等刻画方式，或者放弃寻找完整同位模式的约束，而改为寻找绝大多数的同位模式进行算法研究，使问题本身更接近于天然的增量更新的要求。

参考文献

[1] W. Frawley and G. Piatetsky-Shapiro and C. Matheus. "Knowledge Discovery in Databases: An Overview". AI Magazine: pp. 213-228. ISSN 0738-4602ISSN 0738-4602.

[2] 江宝得，江雯倩，李 洋. 空间数据挖掘与发展趋势研究. 国土资源部信息中心 2006

[3] Yan Huang, Member, Shashi Shekhar, Hui Xiong. Discovering Co-location Patterns from SDatasets: A General Approach. IEEE Trans Knowledge and Data Eng vol.17, no.12, pp. 1472-1485, 2004

[4] K. Koperski and J.Han. Discovery of spatial association rules in geographic information databases. LECTURE NOTES IN COMPUTER SCIENCE, pages 47–66, 1995.

[5] JIN SOUNG YOO and Shashi SHEKHAR. A joinless approach for mining spatial colocation patterns. IEEE transactions on knowledge and data engineering 18:1010, 2004

[6] Y. Morimoto. Mining frequent neighboring class sets in spatial databases. Proceedings of the seventh. ACM SIGKDD international conference on Knowledge discovery and data mining. pages 353–358, 2001..

[7] J.Hipp, U.Guntzer and G.Nakaeizadeh. Algorithms for Association Rule Mining–A General Survey and Comparison. ACM SIGKDD Explorations Newsletter, 2000

[8] Agrawal R, Imielinski T, Swami AN. Mining Association Rules between Sets of Items in Large Databases. Proceedings of the 1993 ACM SIGMOD international conference, 1993.

[9] G. Piatetsky-Shapiro and W. J. Frawley. Discovery, analysis, and presentation of strong rules, In Knowledge Discovery in Databases pp. 229-248, 1991.

[10] Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules in large databases. International Conference on Very Large Data Bases, pages 487–499, 1994.

[11] J. Yoo, S. Shekhar, J. Smith, and J. Kumquat. A partial join approach for mining collocation patterns. Proceedings of the 12th annual ACM international workshop on Geographic information systems, page 241-249, 2004.

[12] D. Cheung, J. Han, V. Ng, and C. Wong. Maintenance of discovered association rules in large databases: An incremental updating technique. PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON DATA ENGINEERING, pages 106–114, 1996.

[13] David W. Chung, Jiawei Han, V. T. Ng, C. Y. Wong. Maintenance of discovered association rules in large databases an incremental updating technique. Proceedings of the Twelfth International Conference on Data Engineering

[14] Jiangfeng He, Qinming He, Feng Qian and Qi Chen. Incremental Maintenance of Discovered Spatial Colocation Patterns**.** Data Mining Workshops, 2008. ICDMW '08. IEEE International Conference. 2008

[15] JunmeiWang, Wynne Hsu, Mong li LEE. A framework for mining topological patterns in spatio-temporal databases Proceedings of the 14th ACM international conference on Information and knowledge management pages:429-436, 2005

[16] Jiawei Han, Jian Pei, Yiwen Yin, and Runying Mao. Mining frequent patterns without candidate generation. Data Mining and Knowledge Discovery pages:53-87, 2004

[17] L.Arge, O. Procopiuc, S. Ramaswamy, T. Suel, and J. Vitter. Scalable Sweeping-Based Spatial Join. Proceedings of the 27th International Conference on Very Large Data Bases Pages: 39 - 48 , 2001.

[18] R. Feldman, Y. Aumann, A. Amir, H. Mannila. Efficient algorithms for discovering frequent sets in incremental databases. Proc. ACM SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery (DMKD), pages 59–66, 1997.

[19] Y. Huang, J. Pei, H. Xiong. Mining co-location patterns with rare events from spatial data sets. Geo-Informat-ica, pages 239–260, 2006.

[20] Y. Morimoto. Mining frequent neighboring class sets in spatial databases.

Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining, pages 353–358, 2001.

[21] Z. Duan, Z. Cai, J. Yu. Incrementally updating association rules based on multiple previously mined results.Natural Language Processing and Knowledge Engineering, 2005. IEEE NLP-KE’05. Proceedings of 2005 IEEE Inter- national Conference on, pages 741–745, 2005.

致谢

本文最终能够成文，首先要感谢我的导师何钦铭教授以及博士生何江峰学长，钱峰学长，从研究方向确立，文献查找，算法设计，以及算法实验实现，这一路上，无一不见你们的帮助。是你们在最困惑的时候给予了我指导，点拨以及启示。

其次要感谢浙江大学，以及本科以来教育我的师长，是你们教会了我基本的科学方法，基本的基础编程，以及探索精神，给了我人生最大的学习的收获，教会了我学习的方法。

再次要感谢我的父母，多年来的养育之恩以及辛勤的教育，教育了我人生最基本的道理，没有你们就没有现在的我。

最后要感谢多年来一直与我一起学习的同学们，在学习道路上有你们的帮助，以及探讨，才使我可以走到今天。

最后再一次，对那些被我感谢的，或者因为篇幅关系，没有提及的人们，说一次谢谢。

本科生毕业论文(设计)任务书

一、题目： 空间同位模式通用增量挖掘算法研究

二、指导教师对毕业论文(设计)的进度安排及任务要求：

该生的毕业设计的要求是，针对空间数据集发生更新时的空间同位模式维护问题进行研究，设计并实现一个空间同位模式的通用增量更新算法。设计目标是与原有传统的重新计算空间同位模式的算法相比，在大部分情况下能提高时间效率，并且分析指出该算法不适用的情况。整个毕业设计的进度安排大致如下：

第一阶段：阅读资料文献，了解空间同位模式的基本概念以及已有算法。

第二阶段：提出空间同位模式的通用增量更新算法，并且实现该算法（同时实现传统算法，以便进行比较)。

第三阶段：设计实验，针对提出的算法进行效率分析，并且与传统算法进行比较分析。

第四阶段：完成毕业论文的撰写。

起讫日期 200 年 月 日 至 200 年 月 日

指导教师(签名) **职称**

三、系或研究所审核意见:

负责人(签名)

年 月 日

毕 业 论 文(设计) 考 核

一、指导教师对毕业论文(设计)的评语:

该生从本学期毕业设计的准备开始至今，按照要求完成了文献翻译。并且阅读了很多有关空间数据挖掘以及空间同位模式挖掘方面的科技文献，在查阅过程中，做了较为完善详尽的读书笔记，比较好地完成了文献综述的工作。在开题报告中，系统分析了增量式挖掘空间同位模式的可行性以及拟采用的方案，并明确了设计实现一个空间同位模式的通用增量更新算法的任务。在毕业设计过程中，认真分析了增量式挖掘在空间数据上特点，提出了一些具体可行的剪枝策略减少算法在更新空间数据时搜索同位模式实例的时间，随后实现了该算法且做了许多实验与原有传统方法进行比较，实验结果表明该算法在大部分情况下有效可行。该生在毕业设计全过程中，表现了良好的独立科研的精神。

指导教师(签名)_____

年 月 日

二、答辩小组对毕业论文(设计)的答辩评语及总评成绩:

成绩比例	文献综述占(10%)	开题报告占(20%)	外文翻译占(10%)	毕业论文(设计)质量及答辩占(60%)	总 评成绩
分值					

答辩小组负责人(签名)_____

年 月 日