



华南理工大学

South China University of Technology

本科毕业设计（论文）说明书

基于不确定性数据的 Pt-k 查询算法的研究与实现

学 院 软 件 学 院

专 业 软 件 工 程

学生姓名 何宇翔

指导教师 杜 卿

提交日期 2010 年 6 月 5 日

华南理工大学

毕业设计（论文）任务书

兹发给 06 级双专业 班学生 何宇翔 毕业设计（论文）任务书，内容如下：

1. 毕业设计（论文）题目：基于不确定性数据的 Pt-k 查询算法的研究与实现

2. 应完成的项目：

(1) 2010 年 4 月 1 日前拟定提纲并提交开题报告

(2) 2010 年 5 月 15 日前完成论文初稿

(3) 进行与论文题目相关的调研工作并收集相关的一手和二手资料

(4) 参考外文文献资料并提交外文翻译译文

3. 参考资料以及说明：

(1) 崔斌, 卢阳. 基于不确定性数据的查询处理综述[J]. 计算机应用. 2008, 28(11): 2729-2731

(2) 周傲英, 金澈清, 王国仁, 李建中. 不确定性数据管理技术研究综述[J]. 计算机学报, 2009,

(01): 1-16

(3) 李建中, 于戈, 周傲英. 不确定性数据管理的要求与挑战. 中国计算机学会通讯. 2009,

5(4): 6-15

(4) 申德荣, 于戈, 寇月, 聂铁铮. 可能世界内数值型不确定性数据匹配模型. 计算机应用研究. 2008-7: 2607-2611

(5) M. Hua, J. Pei, W. Zhang, and X. Lin. Efficiently answering probabilistic threshold TOP-K queries on uncertain data (extended abstract). In Proc. International Conference on Data Engineering (ICDE'08), Cancun, Mexico, April 2008.

(6) Pei J, Hua M, Tao Y F, Lin X M. Query answering techniques on uncertain and probabilistic data : Tutorial summary Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data. Vancouver, 2008 :1357-1364

(7) HUA M, PEI J, ZHANG W, et al. Ranking queries on uncertain data: A probabilistic threshold approach[C]. Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data. New York: ACM Press, 2008, :673-686.

4. 本毕业设计（论文）任务书于 2010 年 3 月 1 日发出，应于 2010 年 6 月 11 日前完成，然后提交毕业考试委员会进行答辩。

专业教研组（系）、研究所负责人_____审核_____年____月____日

指导教师_____签发 _____年____月____日

毕业设计（论文）评语：

何宇翔同学的论文《基于不确定性数据的 Pt-k 查询算法的研究与实现》首先介绍了不确定性数据的相关概念，然后列举了不确定性数据的 U-Topk、U-kRanks、Pk-Topk、Skyline 等常用查询方法，然后针对 Pt-k 算法进行实现和测试其效率，为以后研究其他在不确定性数据方面的 TOPK 查询算法奠定基础。

论文选题符合专业培养目标，能够达到综合训练目标，题目有难度，工作量较大。选题具有学术研究价值。该生查阅文献资料能力较强，能较为全面收集关于不确定性数据的资料，写作过程中能综合运用查询算法优化方面的知识，全面分析查询算法效率问题，综合运用知识能力较强。

该生在毕业设计阶段态度认真，积极主动，保质保量地完成了本次毕业设计，表现优秀。

文章篇幅完全符合学院规定，内容较为完整，层次结构安排科学，主要观点突出，逻辑关系清楚，但在算法方面未能做到创新。

答辩过程中，能够准确回答老师的问题，符合本科毕业论文答辩要求。综上所述，本文符合本科毕业论文的学术要求。

毕业设计（论文）总评成绩：

毕业设计（论文）答辩负责人签字：

年 月 日

摘 要

从六十年代开始,传统的确定性数据管理技术得到极大的发展,现在已经是一个数万亿的数据库产业,而且其数据库的作用奠定了现代信息化社会的基础。但是,近年来随着技术的进步和人们对数据采集和处理技术的不断深入,却发现确定数据的瓶颈。确定性数据无法解释数据采样过程中出现的误差或者错误,或者说现有的确定数据不能解释不确定的现象出现。由于不确定性数据的出现,在许多主要领域,如经济、军事、物流、金融、电信等发现确定数据无法满足这种要求。同时新应用中出现不确定性数据也引发了不确定性数据管理这一新的研究课题。这些应用中相关数据的不确定性为传统的数据处理方法提出了新的挑战。因此,不确定性数据的研究成为了现阶段研究的重点。

而在不确定性数据管理研究中,还没有完全解决查询方面的问题。举个简单的例子,不同的查询方法,查询结果不一样。同时,对于 U-Topk 查询、U-kRanks 查询、Pk-Topk 查询、Pt-k 查询,每次查询都会需要建立一个复杂的可能世界模型作为过渡,因此查询的效率十分低下,呈现一个几何级的增长。本文依据 Pt-k 算法优化的基本思路,实现该算法,并进行对优化的效率进行一个评估。

本文首先介绍了不确定性数据管理研究的一些概念和理论,构成本文的理论基础;其次,分析其他查询算法的基本情况,构成与 Pt-k 算法的对比实例;再次,分析 Pt-k 算法的优化算法,为实现该算法提供基础;最后实现该算法,并对优化后的 Pt-k 算法的效率进行一个效率评估。从以上的分析,我们可以提炼出不确定性数据查询的一些优化思路和策略,为以后继续研究不确定性数据管理技术提供借鉴和指导。

关键词: 不确定性数据, Pt-k 查询, TOP-K 查询, 效率评估

Abstract

From the 60's, the traditional deterministic data management technology is a great development, which value a database industry of several billion now, and its role in the database has laid a foundation for the modern information society. However, in recent years as technology advances and people's data acquisition and processing technology continues to in-depth, but found the data to determine the bottleneck. Deterministic data cannot explain the process of data sampling error or an error occurs, or identify existing data cannot explain the phenomenon of uncertainty. Since the phenomenon of data uncertainty in many key areas, such as economic, military, logistics, finance, telecommunications and so on, identifying data cannot be found to satisfy this requirement. New applications also appear in the data also led to uncertainty in the uncertainty of this new research data management issues. These applications, the uncertainty of the data for the traditional data processing method presented new challenges; therefore, the uncertainty of the data has become the focus of current research.

Uncertainty data management in research, have not fully resolve the query problems. Here is a simple example, for different methods query, results are different. Meanwhile, the U-Topk query, U-kRanks query, Pk-Topk query, Pt-k query, each query will need to build a complex model as the Possible World, so the efficiency is very low, showing a geometric growth. This article based on Pt-k algorithm to optimize, implementation of the algorithm, and for the optimization of the efficiency of an assessment.

This paper introduces the uncertainty of some of the data management concepts and theories, the theoretical basis of this thesis; Second, it analyzes the basic situation of the other search algorithms, composition and Pt-k algorithm comparison example; again, analysis of Pt-k algorithm optimization algorithm, provide the basis for the realization of the algorithm; finally realize the algorithm, and optimized the efficiency of Pt-k algorithm for an efficiency assessment. From the above analysis, we can extract some of the uncertainty data query optimization ideas and strategies for the future continue to study the uncertainty in data management technology to provide reference and guidance.

Key words: Uncertainty of the data, Pt-k queries, TOP-K queries, efficiency evaluation

目 录

摘 要.....	I
ABSTRACT	II
第一章 绪论.....	1
1.1 不确定性数据的研究背景及意义.....	1
1.2 研究现状及相关技术.....	2
1.2.1 不确定性数据的含义.....	2
1.2.2 国内不确定性数据研究成果分析.....	3
1.2.3 国外不确定性数据研究成果分析.....	4
1.2.4 不确定性数据查询算法的特点.....	6
1.3 论文研究内容及组织结构.....	7
第二章 不确定性数据集与查询处理综述.....	9
2.1 引言.....	9
2.2 U-TOPK 算法.....	9
2.2.1U-TOPK 算法基本思想.....	9
2.2.2U-TOPK 算法分析.....	9
2.3 U-KRANKS 算法.....	10
2.3.1U-KRANKS 算法基本思想.....	10
2.3.2U-KRANKS 算法分析.....	10
2.4 PK-TOPK 算法.....	10
2.4.1PK-TOPK 算法基本思想.....	10
2.4.2PK-TOPK 算法分析.....	11
2.5 SKYLINE 查询算法.....	11
2.5.1SKYLINE 算法基本思想.....	11
2.5.2SKYLINE 算法分析.....	11
2.6 本章小结.....	12
第三章 PT-K 查询算法理论研究.....	13
3.1 引言.....	13
3.2 术语定义.....	15
3.3 PT-K 查询算法定义.....	16
3.4 PT-K 查询算法统治集.....	17
3.4.1 统治集属性.....	17

3.4.2 统治集的位置概率.....	18
3.5 处理多元规则.....	20
3.5.1 多规则元组压缩.....	20
3.5.2 前缀共享技术.....	23
3.5.3 减枝技术.....	24
3.6 本章小结.....	26
第四章 PT-K 查询算法的设计与实现.....	27
4.1 引言.....	27
4.2 算法流程分析.....	27
4.2.1 运行效率分析.....	27
4.2.2 设计目标.....	32
4.3 不确定性数据在 PT-K 查询算法中的实现.....	32
4.3.1 数据样例分组.....	32
4.3.2 生成样例的实现策略.....	33
4.4 运行平台与开发工具.....	33
4.5 本章小结.....	34
第五章 实验及性能分析.....	35
5.1 实验环境和测试数据集.....	35
5.2 实验设计.....	35
5.2.1 数据源.....	35
5.2.2 评估标准.....	36
5.3 PT-K 查询算法应用.....	36
5.3.1 数据预处理.....	36
5.3.2 阈值不同时的运行结果.....	37
5.3.3 生成规则发生变化时运行结果.....	39
5.3.4 元组数量发生变化时运行结果.....	40
5.4 实验测试与分析.....	42
5.4.1 TOP-K 质量分析.....	42
5.4.2 参数敏感性分析.....	42
5.4.3 数据处理能力分析.....	42
5.4.4 PT-K 算法优化分析.....	43
5.5 实验结论.....	43

5.6 本章小结	44
第六章 总结与展望	45
6.1 结论与总结	45
6.2 展望	45
参考文献	46
致谢	49

第一章 绪论

1.1 不确定性数据的研究背景及意义

从六十年代开始,传统的确定性数据(deterministic data)管理技术得到极大的发展,不仅成就了一个数百亿的数据库产业^[2],而且其数据库的作用奠定了现代信息化社会的基础。传统的确定性数据具有如下几个特点,数据的精确性和数据的存在性,就是说数据一定存在,数据的值稳定不变。这就说明了一点,确定数据是一种固化的数据,数据是某种信息的一种固有属性,而且不具有可变性。在确定数据的数据库,无论是进行 TOP-K 查询或者其他形式的查询,查询结果是精确的、不可变、稳定的。不确定性数据与确定数据相反,无法获知数据的存在性是不确定性数据最大的特点。

但是,在许多现实的应用中,却发现了确定数据无法满足现有信息社会的发展,主要体现在经济、军事、物流、金融、电信等领域。近年来,随着技术的进步和人们对数据采集和处理技术的不断深入,却发现确定数据的瓶颈。确定性数据无法解释数据采样过程中出现的误差或者错误,或者说确定数据有不确定的现象出现。文献[2]认为可能是原始数据本身不准确或是采用了粗粒度的数据集,也可能是为了满足特殊应用目的或是在处理缺失值、数据集成过程中而产生的。就是说数据是本身具有不确定性,无论它是人工造成的还是本身不精确造成的。

正是因为不确定性数据的广泛存在性,现实生活也需要不确定性数据,因此早在 20 世纪 80 年代末期,就有学者关注这方面的内容,当时关注的焦点是如何对关系数据模型进行扩展。只是近几年网络发展和数据信息膨胀特别厉害,不确定性数据才在更广的范围内得到更多的关注。根据 2009 年 1 月中国互联网信息中心(China Internet Network Information Center, CNNIC)的调查报告,截至 2008 年底,中国网站总数为 287.8 万个,全国网页总数约为 160.9 亿,较 2007 年增长 90%,网页字数为 460, 217, 386, 099KB^[31]。但是互联网数据的质量却不尽如人意,作为一个典型的分散管理系统,互联网中并不存在一个统一的信息发布机构,各网站均可自由发布和维护信息。因此,当信息维护机构不同、信息更新不及时、工作人员误操作时,极易导致不同数据源(或者同一数据源内部)对同一对象描述的不确定。正是由于网络的信息的膨胀,如何处理大量的不确定性数据,成了学者们研究的重点。

同时因为在经济、军事、物流、金融、电信等领域,不确定性数据扮演着关键角色,而传统的数据管理技术却无法有效管理不确定性数据,这就引发了学术界和工业界对研发新型的不确定性数据管理技术的兴趣。

现在,基于不确定性数据,大部分学者的观点基本都是在确定数据的基础上,增加一个概率维度。就是说,在原有的确定数据后面,增加一个表述其确定数据存在性的概率,

以说明有多大的概率，这个确定数据的结果确实存在。

以上内容说明了，不确定性数据的来由和不确定性数据的重要性，接着就是要说明不确定性数据和确定数据查询具有极大的差别。很明显，确定数据不具有概率维度，因此其查询结果具有一致性，无论进行多少次查询，结果是不变和固定的。不确定性数据的查询具有数据的不确定性，因此查询的结果和排序的顺序会因各种条件的变化而变化。举个简单的例子，如果不确定性数据的概率变为 0，或者低于某个阈值，会导致这个数据的存在不具有意义，查询的结果就不会出现该值。

但是，大部分面向确定性数据的查询任务在不确定性数据环境中仍然具有现实意义，需要进行处理。在不确定性数据环境下，由于引入了概率维度，查询的种类反而会增加。元组的概率维度值从侧面反映了该元组的重要程度，因而影响着查询的定义。以 TOP-K 查询为例，在确定性数据处理领域，其意义清晰，返回确定数据的秩函数的值最大的 k 个元组。但在不确定性数据管理领域，秩函数值仅是其中一项因素，概率值是表征元组重要性、存在性的另一因素。在此基础上，最近出现了多种面向不确定性数据的 TOP-K 查询，包括 U-Topk、U-kRanks、PT-k 和 Pk-Topk 等^[3]。

总的来说，不确定性数据在一些重要应用领域中是固有存在的，如传感器网络和移动物体追踪。在不确定性数据上使用传统的查询方法会使查询结果出现偏差，不能满足用户的需求。因此，基于不确定性数据的查询处理受到了越来越多的关注^[1]。本文主要针对 Pt-k 查询算法进行研究和实现，对其效率进行试验并对实验数据进行分析。

1.2 研究现状及相关技术

本章主要阐述了不确定性数据的概念、相关理论以及不确定性数据查询的相关理论，这些都是本文结合统计数据以及各家理论，分析 Pt-k 查询算法的理论基础和依据。不确定性数据概念的理解是本文论述的前提，只有深刻地理解不确定性数据的定义和特征，才能就不确定性数据的查询，乃至 Pt-k 算法做更深层次的探讨。各学者所提出的相关理论为本文的撰写提供了基础素材和阐述思路，而仔细归纳前人的研究成果也是本文创新的基础，有关的理论在本章以下内容中均有详细说明。

1.2.1 不确定性数据的含义

申德荣等人认为不确定数据就是在可能世界中的不确定信息，而且不确定性数据普遍存在，如人们只知道某一属性取值的范围或可能的取值，无法确定该属性的确切值^[4]。他还认为，目前支持的数据管理还是主要局限于确定数据的管理，这样约束了数据在实际场景中的描述，最终导致信息的缺失。不确定性数据的研究，有助于实际场景中真实数据的处理，突破确定数据的瓶颈。

裴健认为不确定性数据是由于数据产生的过程不精确产生的，而且不确定性数据存在

于现在已知的大部分领域^[7]。他认为，不确定性数据一般出现在数据无法精确描述的现实场景中，同时解释了现实场景中，出现的双生现象是由不确定性数据产生的。

Motro 和 Trio 将不确定性数据进行了一个详细的分类，并提出了不确定性数据的分类方法，将不确定性数据分为精确和不精确两种。同时，有关不精确数据的描述又有多种，如不确定的数据、概率数据、模糊集数据、近似数据、不完备数据和不精确数据等^[6]。不确定的含义是指属性值的可信性，如根据疑难病人的各项检测信息得出可能的病症，其可信度不是 100%；概率是指属性取某一值的概率，如心脏病人中吸烟者占 75%，非吸烟者占 25%，其概率和为 1；肥胖者得心脏病的可能性为 0.7，但没有模糊度和为 1 的约束；该病人的年龄在 20~25 岁为近似数据；不完备的数据是指有信息丢失，如一部分病人的病例中没有记录病人的血型；不精确数据是指数据的取值可能是集合中的数据之一，等等。

文献[6]中除了将非确定的数据定义为不确定和不精确数据外，还包括不完备、模糊、不一致、不明确。其中除了不明确为语义模糊概念外，其他都涵盖了 Trio 中的定义。

可以明确的一点是，归纳已有文章中讨论的数据不确定性，现有的研究都认为，不确定性数据无法准确表明其存在性，不确定性数据无法用单一因子描述本身。就是说与确定数据的区别在于，无法使用单一变量表达，需要增加一个或多个附属变量共同描述该数据，而数据具有不确定性。归纳已有文章中讨论的数据不确定性，本文把数据分为如下几类：

表 1-1 数据分类

确定数据	不确定性数据
具体值，如元组 A 的值为 100	概率值，如 AMY 的年龄为 34 岁的概率是 0.16，为 45 岁的概率是 0.4
属性值，如 AMY 是年轻人	范围值，如 AMY 的年龄为[32, 56]
	互斥值，如 AMY 的年龄或者为 32 或为 78
	模糊集值，如 AMY 的成绩是 A 的可能性为 0.2，B 的可能性为 0.4

在可能世界中，这些不确定性数据普遍存在，并交叠在一起，单一因子无法描述一个不确定性数据，需要多个不同的值组合描述一个不确定性数据的存在，如 AMY 的年龄为[32, 56]的概率为 0.1，而取[40, 45]的概率为 0.2，这是将概率值与范围值结合表示属性值。很明显的是，不确定性数据的组合方式十分的丰富，无法通过单一因子描述使其与确定数据产生了巨大的不同。本文的 Pt-k 算法研究的数据集是概率值和互斥值的集合，就是说明数据的存在性具有一定的概率，同时有可能和其他数据互斥。这种数据构成我们研究的基础。

1.2.2 国内不确定性数据研究成果分析

在国内知名的期刊、论文数据库中以“不确定性数据”为关键字进行查询，可以发现在中国期刊全文数据库仅有 13 篇期刊、中国优秀硕士学位论文全文数据库仅有 2 篇论文、中国期刊全文数据库_世纪期刊和中国博士学位论文全文数据库均没有相关条目。其中主要以

研究不确定性数据查询相关的文章仅有 2 篇，提及到不确定性数据查询技术的文章仅有 2 篇，而且相关文章的最早发表时间是 2001 年 2 月。但是，在国外的文献数据库，早在 80 年代就有相关的文章，而且当前搜索结果不止 100 篇，国外特别是美国，有多所学校和公司成立相关研究机构，针对不确定性数据进行深入透彻的研究，并且取得不少的研究成果。可见，国内关于不确定性数据处于探索阶段，文章内容多为对国外已有研究成果进行综述性研究，没有提出较为鲜明的观点或者理论。因此，本文研究关于不确定性数据的查询技术是十分具有前瞻性和重要性。

国内关于不确定性数据研究方面的文章有，文献[1]、文献[2]和文献[3]。

其中文献[1]中，崔斌等人是针对不确定性数据查询技术的综述性研究，以国外 20 多篇论文为基础，针对前人研究进行概述，并没有提出鲜明的观点或者论点，相当于国外优秀不确定性数据查询技术研究成果的综述。

文献[2]中，周傲英等人是对整个世界，在不确定性数据技术研究的综述，以国外 100 多篇论文为基础，泛泛的对不确定性数据进行一个概述性描述，也没有提出鲜明的观点或者论点，其中有关于不确定性数据查询技术方面的综述。内容中，也提到了关于国外正在如火如荼的对不确定性数据进行研究，而国内则没有一点动静。同时，他也认为，不确定性数据广泛存在于这个现实世界中，十分具有研究的价值和意义，非常有必要在这个方面进行深入的研究。

文献[3]中，李建中等人提出了不确定性数据研究的一些方向和挑战，并提出一些不确定性数据管理技术的要求。但是文章的主要分析还是建立在国外已有研究的基础上展开的，没有提出一个具体的研究方向，也没有提到国内现在在不确定性数据这个方面的研究情况，或者研究项目。只是一味的对国外研究情况进行一个引用，并在其基础上，提出一些要求和不确定性数据研究方向的挑战。

从他们的文章中也可以发现，他们的研究的结果无不是建立在国外已有的研究基础上，并没有太多自己个人的观点。属于综述性文章，所以我找了一篇和不确定性数据有一定关系的文献[4]进行研究。申德荣等人在文献[4]中提出出匹配度和完备度概念，并给出了匹配模型；提出基于 range 数据类型的匹配度和完备度为最小粒度,定义了其他类型的不确定数据类型的匹配规则，并给出了相应的匹配规则定义。但是作者的观点也是建立在国外已有的研究基础上。

由此可见，国内在不确定性数据研究方向，特别是查询技术研究方向，没有任何的研究成果，所以针对不确定性数据的查询技术进行一个深入的研究是十分有必要的。

1.2.3 国外不确定性数据研究成果分析

国外在关于不确定性数据的研究，有许多研究成果，在此，就不一一列举，仅仅是列举几个比较具有代表性的研究成果，还有国外研究的情况，并对此进行一个简单的分析。

下面介绍一些知名大学以及公司的研究机构正在进行的相关科研项目的基本情况，并

对其进行分析，从而可以了解到国外的研究情况

表 1-2 知名大学以及公司的研究机构的不确定研究项目列举

多伦多大学 ^[32] http://queens.db.toronto.edu/project/conquer/	ConQuer 项目，Ren�� J. Miller 领导的研究小组，主要研究方向是针对不一致数据库的管理技术，利用重写 SQL 语句优化查询和系统执行，建立一个高效和可伸缩的数据库。
普度大学 ^[33] http://orion.cs.purdue.edu/index.html	Orion 项目，研究的是一个处理模糊数据的数据库管理系统。相对于其他的不确定性数据数据库，同时支持任意的属性和相关元组的不确定性，使数据库准确地处理离散和连续的数据。
斯坦福大学 ^[34] http://infolab.stanford.edu/trio/	Trio 项目，基于称为 ULDBs 扩展关系模型，它支持基于 SQL 的查询语言 TriQL，研究不确定性数据的世系分析。Trio 可以在广泛的领域使用，主要包括以下领域：传感器数据管理，数据清理和整合，信息提取系统，以及近似和假设查询处理。
康奈尔大学 ^[35] http://www.cs.cornell.edu/bigreddata/maybms/	MayBMS 项目，由 Lyublena Antova 领导的小组，主要关注于如何利用和扩展成熟的关系数据库技术在不确定性数据查询和管理技术方面进行优化，研究其鲁棒性和可扩展性。
华盛顿大学 ^[36] http://www.cs.washington.edu/homes/suciu/project-mystiq.html	MystiQ 项目，是由 Nilesch Dalvi 领导的研究小组，MystiQ 利用概率模型对来源于大量不同来源的不确定性数据进行查询研究，现在该项目的目标是发展在大型的不确定性数据库进行有效查询的技术。同时，研究小组认为，数据的来源十分重要，如何在多源数据库找到数据的联系是关键。
英特尔/加州大学伯克利分校 ^[37] http://www.eecs.berkeley.edu/Research/Projects/Data/102060.html	BayesStore 项目，由 Eirinaios C. Michelakis 领导的研究小组，该项目的主要研究一种新的基于概率模型和统计的不确定性数据管理架构。主要努力方向是在实现高效率的 SQL 查询算法，增加概率关系运算符的引擎。该项目的主要目的是（1）支持不同模式的高效查询处理；（2）在引入新的模型和算法的时候，能有扩展的 API 提供；（3）支持大型的数据库
马里兰大学 ^[38] http://www.cs.umd.edu/~vs/research.htm#pdb	ProDBs 项目，研究重点是在概率数据库，提出了第一个概率数据模型，ProView，后续扩展这个数据模型，处理实时性的概率数据库和 XML 数据库。该项目现在的重点放在了概率聚集效率方面和概率数据库的时空概念。
IBM Almaden ^[39] http://www.almaden.ibm.com/cs/project/s/avatar/	Avatar 项目，该项目有两大目标：（1）从非结构化数据中抽取结构化的信息；（2）基于这类信息构建下一代搜索和商业智能应用。该项目正在进行对概率数据库的研究，同时该项目还研究信息检索、自动学习机器。

由上表可以看出，不确定性数据管理正成为一个研究热点，国外针对不确定性数据研究已经进行得如火如荼，而国内则鲜有动静，因此，对不确定性数据进行研究是十分有必要和具有前瞻性意义。

可以看到，国外一些研究机构都对概率数据库研究十分感兴趣，概率数据库的研究是不确定性数据的研究重点。本文研究的 Pt-k 算法也是基于概率型的不确定性数据，进行一个有效的查询，并对查询效率进行分析。

另外，关于国外研究成果方面，还有 Ré 与 Suciu^[8]观察到不确定性数据广泛出现在诸多应用之中，并总结了不确定性数据管理所面临的巨大挑战，同时，他们也讲述了不确定性数据的有关概念和现有的研究情况。同时，Dalvi 和 Suciu^[9]进一步从理论角度阐述不确定性数据管理的基础与挑战，并且分析了不确定性数据的一些难点。还有 Aggarwal 与 Yu^[10]从算法与应用角度综述了不确定性数据管理技术。Pei 等人^[11]回顾了近期不确定性查询处理方面的进展，尤其是他们自己的研究成果，如何提高不确定性数据查询的效率是他们研究的重点，包括范围查询、skyline 查询与 Ranking 查询等。

之所以说不确定性数据是近期世界研究的重点，是因为整个世界，除了以上所说的研究机构和研究学者，不确定性数据的概念和研究成果，多次在近几年的国际会议上频频提出。不仅如此，在国际上知名的国际期刊也是有重点的提到不确定性数据研究，例如 SIGMOD (Special Interest Group for the Management of Data)、VLDB (Very Large Data Bases)、ICDE (International Conference on Data Engineering)、TODS (Transactions on Database Systems)、VLDBJ (Very Large Data Base Journal)、TKDE (Transactions on Knowledge and Data Engineering) 等。此外，一些主流的国际会议也积极召开一些小型的研讨会，讨论不确定性数据的热点话题。还有，近几年关于不确定性数据管理方面的研讨会：IEEE ICDM 会议的 DUNE 2007，IEEE ICDE 会议的 MOUND 2009，VLDB 会议的 MUD 2007 和 MUD 2008 等。同时，国际一流刊物 VLDBJ 和 TKDE 也在 2009 年和将在 2010 年分别刊出一个关于该主题的特刊 (Special Issue)。因此，不确定性数据正成为一个研究热点。

1.2.4 不确定性数据查询算法的特点

不确定性数据和确定数据是完全不同的两种事物，确定数据是一种类似于“真理”的存在，无法辩驳，无法修改，永远正确的，因此，关于确定数据的查询结果特点就是，结果同样也是一种“真理”存在，不会变化的。




但是，不确定性数据，完全不同，它类似于“量子”，带有概率性的运动，不确定性，无法确认其存在性，只能用一种概率模型进行一个抽象的描述，是不确定性数据的最大特点。










虽然我们不知道不确定性数据仅仅比确定数据多一个概率维度，但是关于不确定性数据的查询结果却远远不一样，运用不同的查询算法，查询结果往往不同。对于概率维度，常用到可能世界 (Possible World) 模型^[12]，它是描述不确定性数据的通用模型，可见传统




数据模型无法准确描述不确定性数据。该模型包含若干个可能世界实例，在各个实例中，一部分元组存在，剩余元组不存在。可能世界实例的发生概率等于实例内元组的概率乘积和实例外元组的不发生概率的乘积之积。所有可能世界实例的发生概率之和等于 1。但是，往往根据不同的查询算法，所构建的可能世界可能是不同的，所以也导致结果的不确定性。

以图 1 为例，三种夸克（十字形是奇夸克，正方形是上夸克，圆形是下夸克）在某个区域出现的概率分别是 0.7、0.6 和 0.5，颜色表示各夸克存在时的概率值。则共有 $2^3=8$ 个可能世界实例，各实例的发生概率依赖于所包含的元组集合。例如，仅包含上夸克（正方形）、下夸克（圆形）两种夸克的可能世界实例的发生概率等于 $(1-0.7) \times 0.6 \times 0.5 = 0.09$ 。

图 1-1 可能世界模型实例

数据序列			
时间	1	2	3
夸克			
概率	0.7	0.6	0.5

1				0.21
2				0.09
3				0.14
4				0.21

5				0.06
6				0.09
7				0.14
8				0.06

从上面一个例子可以看到，利用可能世界（Possible World）模型^[12]解决一个简单的不确定性数据的问题，3 个元组需要用到 8 个可能世界。如果假设元组仅仅含有是否存在这种单一的不确定性，那么对于 N 条记录，就能演化出 2^N 个可能世界，如果通过简单罗列可能世界，是无法解决一个简单的查询，因为已经无法忍受查询时候，产生的时间消耗。那么，如果元组还有其他的不确定性，如之前提到过的，有可能与其他元组互斥，无法共存，此时的查询处理就更加复杂，如果按照传统的查询思维，以“枚举所有可能世界实例，计算基于该实例的查询结果，然后得出结果集”的处理方式，是无法处理一个最简单的查询。因而，不确定性数据的查询技术成为了研究的重点，迫切需要结合各种减枝、排序等技术以快速计算查询结果。查询技术本身的效率问题就是不确定性数据管理技术的难点和首要解决问题之一。

因此，概率维度不是普通的维度，它的出现改变了传统的数据处理模式。

1.3 论文研究内容及组织结构

本文主要采用定量分析，从理论到实践、宏观与微观相结合的方法。同时，综合应用对比分析和实践研究。通过分析国内外不确定性数据研究的现状，指出不确定性数据查询亟待解决的问题；又通过针对 Pt-k 算法进行一个深入的研究分析，将本文的理论应用到现实数据中去。最终，评估该算法的效率，力求使本文的研究成果既有理论意义，又有实际意义。

本文的结构和各章研究的主要内容如下：

第一章绪论首先说明了本文的研究背景和意义，接着解释了本文主要讲述的内容。并阐述了不确定性数据相关的理论，还有分析不确定性数据查询亟待解决的问题。

第二章主要是针对现在比较热门的查询算法，skyline 查询和 TOP-K 查询进行简单的阐述，并且分析现有的不确定性数据查询算法，为后面和 Pt-k 算法进行比较奠定理论基础。

第三章主要针对 Pt-k 查询算法进行深入的理论研究和分析，为实现该算法提供理论依据。

第四章主要是描述 Pt-k 查询算法的设计与实现，分析和设计元组压缩、统治集概率、前缀共享和减枝技术这四个方面的模块，并且针对每一个方面的模块细节进行一个详细的阐述。

第五章主要是对该算法进行测试，接着进行实验，并分析其数据，评估其效率。

第六章是进行总结和展望。

最后是根据前文的实验结果得出结论。

第二章 不确定性数据集与查询处理综述

2.1 引言

正如上一章对不确定性数据的描述一样，在现今社会，不确定性数据已经是广泛的存在，如传感器网络^[15]和移动物体查询^[16]。正如上一章讨论的内容，传统的查询处理方法已经对不确定性数据不再适用，无法解决不确定性数据的查询问题，而可能世界（Possible World）模型^[12]是通向解决办法的一条途径。但是，也要清晰明白，我们不可能列举所有的可能世界。

在数据存在不确定性的前提下，概率不确定性数据模型也许是我们的解决办法之一。同时，基于该模型的查询能够根据不确定性数据的不确定性区间及其上的概率分布函数得出比传统查询方法含有更多有用信息、具有一定概率保证、更可信的查询结果，但是也付出了更多的计算代价，计算方法更为复杂。这就需要更加高效的查询方法，本章会就现有关于不确定性数据的查询技术进行简单的分析。

本章根据不确定性数据查询方法的分类，介绍各种查询的计算方法，并对某些查询的计算方法提出改进；本章从问题定义和算法等角度对不确定性数据上 TOP-K 查询和 skyline 查询分别进行介绍。

2.2 U-Topk 算法

2.2.1 U-Topk 算法基本思想

U-Topk 查询返回一个长度为 k 的元组矢量，它在所有可能世界中的发生概率最大。当我们要求 TOP-K 返回的所有元组都来自同一个可能世界的时候，使用 U-Topk 查询是适合的。

2.2.2 U-Topk 算法分析

定义 2-1 Uncertain Top-k Query (U-Topk): 令 D 为一个不确定性数据库，它的可能世界空间为 $PW=\{PW_1,...,PW_n\}$ 。令 $T=\{T_1,...,T_m\}$ 为一系列长度为 k 的元组矢量，对于每个 $T_i \in T$: (1) T_i 中的所有元组是按照得分函数 F 排名的, (2) T_i 是非空可能世界 $PW(T_i) \subseteq PW$ 的 TOP-K 查询结果。基于 F 的 U-Topk 查询返回 $T^* \in T$ ，其中

$$T^* = \arg \max_{T^i \in T} \left(\sum_{w \in PW(T^i)} (\Pr(w)) \right) \quad (2-1)$$

U-Topk 查询返回一个长度为 k 的元组矢量，它在所有可能世界中的发生概率最大。当我们要求 TOP-K 返回的所有元组都来自同一个可能世界的时候，使用 U-Topk 查询是适合的。

对于 U-Topk 查询, 我们可以使用它的优化算法 OPTU-Topk 算法^[1], 该算法的基本思路是:

- (1) 设置一个以搜索状态概率优先级排序的队列 Q , 初始化时插入 $s_{0,0}$, 概率 $P(s_{0,0})=1$ 。
- (2) 当 Q 不为空时不断执行下面操作: 从 Q 中弹出 $s_{l,i}$; 如果 $l=k$ 时返回 $s_{l,i}$, 否则根据 i 和 d 的比较情况选择下一个要访问的元组 t ; 分别对 t 可以加入和不加入 $s_{l,i}$ 两种情况, 将 $s_{l+1,i+1}$ 和 $s_{l,i+1}$ 插回 Q 。

2.3 U-kRanks 算法

2.3.1 U-kRanks 算法基本思想

U-kRanks 查询返回在各个级别中出现的总概率最大的元组。这些元组不一定是最可能成为 TOP-K 矢量的, 它们也不一定都出现在同一个可能世界, 并且同一个元组有可能在结果集里面出现多次。这类查询适合那些对元组来自不同世界没有限制的查询。

2.3.2 U-kRanks 算法分析

定义 2-2 Uncertain k Ranks Query (U-kRanks): 令 D 为一个不确定性数据库, 它的可能世界空间为 $PW=\{PW_1,...,PW_n\}$ 。对于 $i=1...K$, 令 $\{x_i^1,...,x_i^m\}$ 为一系列的元组, 每个元组 x_i^j 在一个基于得分函数 F 的可能世界 $PW(x_i^j) \subseteq PW$ 中排名 i 。一个基于 F 的 U-kRanks 查询返回

$$\{x_i^*; i=1...k\} \quad x_i^* = \arg \max_{x_i^j} (\sum_{PW(x_i^j)} (Pr(w))) \quad (2-2)$$

U-kRanks 查询返回在各个级别中出现的总概率最大的元组。这些元组不一定是最可能成为 TOP-K 矢量的, 它们也不一定都出现在同一个可能世界, 并且同一个元组有可能在结果集里面出现多次。这类查询适合那些对元组来自不同世界没有限制的查询。

对于 U-kRanks 查询, 根据文献[1]可以使用 OPTU-kRanks 优化算法, 该算法的基本思路是: 在计算排名 i 时, 对于一个新来的元组 t , 计算其在所有可能世界在排名 i 上出现的概率 P_t ; 如果 P_t 比目前答案的概率大, 并且比将未见元组考虑进来时的概率也大, 那么 t 是结果集中排名为 i 的元组。

2.4 Pk-Topk 算法

2.4.1 Pk-Topk 算法基本思想

Pk-Topk 返回所有在可能世界实例中成为 TOP-K 的总概率最大的 k 个元组。Pk-Topk 与 PT-k 查询算法很相似, 只不过它没有用到阈值 p 将所有可能在 TOP-K 出现的元组输出, 而是返回元组的个数是固定的 k 个。

2.4.2 Pk-Topk 算法分析

和 Pt-k 算法一样, Pk-Topk 算法比较容易理解, 在不确定性数据的可能世界模型中, Pk-Topk 是将处于可能世界的前 K 个, 而且只输出 K 个值, 后面的值就不再输出。可以用以下表达式进行表述,

定义 2-3 Possibility k Top-k Query (Pk-Topk): T 是可能世界模型中元组的集合, t 是元组, $\text{Pr}_Q^k(t)$ 是对 t 元组进行 TOP-K 查询后输出的概率值, p 是阈值。

$$\text{Answer}(Q, p, T) = \{t | t \in T, \text{Pr}_Q^k(t) \geq p\} \quad | \quad \text{Answer}(Q, p, T) | = k \quad (2-3)$$

2.5 Skyline 查询算法

2.5.1 Skyline 算法基本思想

上面几个算法都是属于 TOP-K 查询算法在不确定性数据方面的几种表述。Skyline 查询和上面的 TOP-K 查询不同, 是另外一种解决不确定性数据查询问题的方法。Skyline 查询错误! 未找到引用源。能用于解决多准则决策(Multi-Criteria Decision-Making, MCDM) 问题。

定义 2-4 给定一个确定性的 n -维数据集 D , 任一点 d 可被表示为 $(d.D_1, \dots, d.D_n)$ 。Skyline 查询返回数据集 S , $S \subseteq D$, 则 $\forall u \in S$, 不存在其它点 v , 满足 (1) 对于任一维度 $i(1 \leq i \leq n)$, $u.D_i \leq v.D_i$, (2) 存在一个维度 $j(1 \leq j \leq n)$ 使得 $u.D_j < v.D_j$ 。

Skyline 简单来说是在多属性对象集(U)上的集合, 它由 u 中所有不被其它对象所“支配”的对象组成。所谓支配是指: 如果说对象 p 支配 $q(p, q \in U)$, 则 p 在任意属性上的取值都不“差”于 q , 且 p 至少在某一属性上取值“优”于 q 。

Skyline 查询与 TOP-K 查询十分不同, 他是从另外一个角度去考虑, 不确定性数据。TOP-K 查询主要考虑的是值的排序还有兼顾于概率维度, 计算起来比较简单方便, 而 Skyline 查询则不同, 是完全另外一种思维, 利用多维度的思想, 以几何的方式解决查询问题, 现在 Skyline 查询也是比较热门的研究方向, 在数据挖掘领域以及多规则决策应用领域发挥着重要作用。Skyline 查询也有在概率维度方面的研究成果, Pei[19]^[18]等人根据可能世界模型定义了概率 Skyline 查询, p -Skyline 查询。

2.5.2 Skyline 算法分析

Skyline 查询最早由 Brozoyi 等人在文献[19]提出, 它们还给了一个生动的例子来阐述 skyline 的应用意义: 假设游客 Laneelot 要到美丽的巴哈马首都 Nassau 去旅游。他想选一家既能观赏到美丽海景又不是太贵的酒店。然而, 鱼与熊掌不可兼得, 这两个属性往往是互斥的: 靠近海边的酒店费用往往比较昂贵, 而费用低廉的酒店往往远离海边。要在旅游业数据库中找到哪一家宾馆对 Laneelot 最合适将是个不小的挑战, 但旅行社至少可以找出那些 Laneelot 可能感兴趣的酒店。所谓“可能感兴趣”的宾馆显然就就是那些不被其它宾馆所

“支配”的宾馆，这些不被支配的宾馆就构成了 skyline(由其外形而得名)。

Skvline 只包含那些用户最可能感兴趣的对象。在最优化决策时，用户显然只需要考虑那些属于 Skyline 的对象，而不必再关心那些被过滤掉的对象。这样用户就可以在小规模的 Skyline 对象集上按照自己的偏好进行选择。自从 Skyline 查询在学术界被提出以来，它就一直是一个非常活跃的研究领域，并且被广泛应用于城市导航^[20]、多标准决策^[19]、数据挖掘可视化^[21]及用户偏好查询^[22]等领域。

2.6 本章小结

关于不确定性数据的查询种类丰富，在确定性数据下，单一的 TOP-K 查询也能演化成多种复杂的查询。这样，就侧面反映了不确定性数据的查询内涵十分丰富，虽然只是仅仅在增加一个概率维度上进行讨论，就有四种典型的查询方式，还有其优化的方式，且还没有把我们接下来的介绍的 Pt-k 算法囊括其中。这就能更加明确了一点，不确定性数据的查询研究是一个重点，同时，是一个难点，出现了这么多种不同的查询方式，但是还没有哪一种查询方式的效率比较突出，只能在某种领域或者方面占优，可见，不确定性数据的查询技术还有相当的技术难关。关于 Pt-k 查询算法会在接下来一章进行比较详细的叙述，所以这里不继续展开介绍。本章所在的重点是片面的介绍以概率为基础的，发展比较完善的查询算法。本章的 2.2 到 2.4 节主要讲述的就是多种 TOP-K 算法。2.5 节主要讲述 Skyline 查询，它在数据挖掘领域以及多规则决策应用领域发挥着重要作用。

本章主要是针对现在比较热门的几个查询算法，TOP-K 查询和 skyline 查询进行简单的阐述，并且分析现有的不确定性数据查询算法，为 Pt-k 算法和第五章实验进行比较奠定理论基础。

第三章 Pt-k 查询算法理论研究

3.1 引言

上一章就一些现在比较流行的不确定性数据的算法进行了一个简单的综述和分析，同时可以作为本章内容的一个铺垫。上一章也提到 TOP-K 算法，接下来讨论的内容也是 TOP-K 算法的一种——Pt-k 算法，也就是本文研究的重点。Pt-k 算法最早是在文献[23]提出来的，采用的也是不确定性数据模型也是可能世界模型。

Pt-k 算法作为 TOP-K 查询算法在不确定性数据上的一种类型，反映的是元组在查询中处于较有利的元组，并且提出阈值概念，增加了查询的可控性。Pt-k 算法比 U-kRanks 和 U-Topk 效率更高，因为 U-Topk 和 U-kRanks 查询都是对“排名敏感”，这就要求实例化所有的可能状态。而 PT-k 查询时不用理会各个元组的排列的，只要他们出现在 TOP-K 的概率大于给定的阈值就可以了。因此 PT-k 查询不需要维护像 U-Topk 和 U-kRanks 查询那么多的状态，可以利用更加高效的算法来进行 PT-k 查询。这也是选择 Pt-k 算法作为研究对象最重要的原因。

本章的研究基础是来自于文献[7]不确定性数据集上的 Pt-k 查询技术。

要对 Pt-k 算法进行研究，需要考虑以下几个方面的内容：查询结果的质量、查询的效率、查询的基本方法。首先，我们通过例 3-1，先了解不确定性数据和 Pt-k 查询的基本概念。

例 3-1 世界上最大的鲸鱼，蓝鲸，也是一种濒危动物，科学家为了研究这种濒危动物在地球上的繁衍过程，以保证其能够继续在地球上繁衍生息，还有调查其种群情况，常常会在某些蓝鲸皮肤上安装传感器。然后利用海洋上的监测器，统计蓝鲸种群情况。但是，由于蓝鲸在海上，传感器不敏感或者发生故障的概率较高，侦察不能保证十分精确。但是，侦察的置信度通常可以精确统计出来。表 3-1 显示的是一些传感器侦察蓝鲸的情况。当传感器侦察到蓝鲸在附近经过时，会记录该动物出现的时间和逗留的时间。

在某些地点，需要设置多个侦察器去侦察目标。有两个侦察器在同一个地方（如，B206 和 B231，S063 和 S732）同时侦察到目标，记录包括 Record2 和 Record3，Record5 和 Record6。在这种情况下，最多只有一个传感器是正确的。因为，侦察点都是不会互相覆盖侦察区域的，而且每一个被标记的蓝鲸，其传感器编号都是唯一的，而且被标记过的蓝鲸就不必要被标记，而且每次被标记的蓝鲸都是在种群中随机抽取的样本，因此，对于科学家研究蓝鲸的种群是十分有意义的。但是，由于传感器的误差对科学家的研究有着巨大的影响，所以更加需要处理这种不确定性数据。

表 3-1 蓝鲸种群侦察的记录

RecordID	地点	时间	传感器编号	逗留时间（分）	置信度
Record1	印度洋	9/2/06 2:14	A101	25	0.3
Record2	北冰洋	6/12/09 4:07	B206	21	0.4
Record3	北冰洋	6/12/09 4:09	B231	13	0.5
Record4	大西洋	3/13/06 22:31	E101	12	1.0
Record5	南太平洋	12/12/06 20:32	S063	17	0.8
Record6	南太平洋	12/13/06 22:28	S732	11	0.2

首先，我们考虑用到可能世界语义模型^[24]，利用这个模型我们可以提高查询的效率，就好像文献[27][28]。数据可以经过重新处理得出可能世界的集合。可能世界的集合可以通过一些约束规则进行划分。在表 3-1 中，Record2 和 Record3 不能同时是真，Record5 和 Record6 也是，因此我们可以认为他们在生成规则中， $\text{Record2} \oplus \text{Record3}$ ，还有 $\text{R5} \oplus \text{R6}$ 。以下的表 3-2 就是展示了所有可能世界的集合。

所有的 TOP-K 算法基本思路都是首先列举所有可能世界的集合，然后再继续处理，这也是处理效率低下最重要的原因。

表 3-2 可能世界的集合

可能世界	概率	Top-2 逗留时间
W1= {Record1,Record2,Record4,Record5}	0.096	Record 1,Record2
W2= {Record1,Record2,Record4,Record6}	0.024	Record1,Record2
W3= {Record1,Record3,Record4,Record5}	0.12	Record1,Record5
W4= {Record1,Record3,Record4,Record6}	0.03	Record1,Record3
W5= {Record1,Record4,Record5}	0.024	Record1,Record5
W6= {Record1,Record4,Record6}	0.006	Record1,Record4
W7= {Record2,Record4,Record5}	0.224	Record2,Record5
W8= {Record2,Record4,Record6}	0.056	Record2,Record4
W9= {Record3,Record4,Record5}	0.28	Record5,Record3
W10= {Record3,Record4,Record6}	0.07	Record3,Record4
W11 = {Record4,Record5}	0.056	Record5,Record4
W12 = {Record4,Record6}	0.014	Record4,Record6

表 3-3 关于表 3-1 中数据 TOP-2 的值

RecordID	Record1	Record2	Record3	Record4	Record5	Record6
概率	0.300	0.400	0.380	0.202	0.704	0.014

TOP-K 算法经常被用来处理不确定性数据,例如科学家对 TOP-2 在逗留时间上有兴趣。在不同的可能世界去回答这个问题将会十分的困难。表 3-2 就展示了所有可能世界的 TOP-2 记录。

但是我们通过对所有可能世界的进行计算,求出每个元组的 TOP-2 的概率(表 3-3)十分感兴趣,具体的方法会在 3.3 节进行阐述。这里说明表 3-3,每一个记录的概率是通过该元组在所有可能世界在 TOP-2 出现的概率和。通过表 3-3,将十分容易找到超过阈值的元组,举个例子,我们假如将 $p=0.35$,则{Record2, Record3, Record5}将会是 Pt-k 查询后的结果。如果是 U-Topk 查询的话则返回<Record5, Record3>, U-kRanks 查询则返回<Record5, Record5>。

我们从上面的例子可以发现,当元组个数是一定的时候,如果这种互斥现象越多,产生的可能世界模型越多,它是几何级增长的。可以设想一下,假设其中有 M 条互斥现象,如果某个元组的置信度小于 1 且独立于其他元组,则可以认为它的否定现象为它的互斥元组。在 100 个元组内,如果都是互相独立的置信度小于 1 的元组,则产生的可能世界已经超过现有计算机的计算能力, 2^{100} 个可能世界需要维护,以一个可能世界由一个 byte 存储计算,需要 2^{100} byte,就是说 2^{70} G byte,不用说,即使现在最大的超级电脑,内存也没有这个容量。同时,明显它的计算时间也是一个无法计算的值,从时间效率和空间效率来说,这种以枚举所有可能世界的算法是不可取的。

我之所以选择 Pt-k 算法进行研究的一个最大原因就是, U-Topk 和 U-kRanks 查询算法都需要将所有可能世界实例化,同时不能避免元组重复出现,另一方面, Pt-k 算法可以找出概率较高的元组,而不是排名靠前的元组, Pt-k 算法适合更高效的方法,所以我将研究的重点放在了 Pt-k 算法中。

在接下来的几节中,我们将针对这个算法进行一个详细的分析和叙述,用来奠定我们在实现该算法和实验的理论基础。我们将通过介绍文献[7]不确定性数据集上 Pt-k 查询技术,展示一种比较高效的 Pt-k 查询。

3.2 术语定义

t:	元组, 包含一个数据值和一个概率值
T:	包含一系列的元组, 这些元组的出现概率 $\Pr(t)>0$
R:	生成规则明确了一系列的互斥元组, 这些元组的概率加起来小于等于 1, 并且在一个可能世界实例, 每个生成规则最多只能出现一个元组
\mathcal{R}_T :	有且只有一个元组的生成规则的集合
W:	一个可能世界, 在该世界中, 所有生成规则最多只出现一个元组
ω :	所有可能世界的集合

St_t : 统治集, 排在 t 前面的元组的集合
 $t' \prec_f t$: 元组偏序, 说明经过排序函数 f 后, 元组 t' 在元组 t 前面

3.3 Pt-k 查询算法定义

上面我们已经对 TOP-K 算法现有的弊端进行了讨论, 对于仅仅 100 个元组的不确定性数据的查询, 无论是时间消耗或者空间消耗都是无法计算的, 我们需要对这种算法进行一个优化, 首先考虑的情况就是元组互相独立, 这样的好处就是元组之间不会互斥现象, 方便计算。接着我们需要一种高效的方法去实现 Pt-k 算法。

然后, 我们需要理解的是, 如果要用到所有元组, 则肯定会需要利用上面提到的可能世界语义模型^[24], 这个模型可以方便我们理解如何优化 Pt-k 算法。

对于一个可能世界, 它可以表述为: 如果生成规则 R 和可能世界模型有交集, 则交集的秩为 1 (这就是受互斥值的影响), 且概率存在, 那么, 这个可能世界的概率就是所有交集的元组概率积与所有没有发生交集元组的概率积的积。这个表述和基本概率知识表述一样, 发生此类事件的概率与不发生此类事件的概率积。

一般地, 不确定表 T 包含许多元组, 每个元组 t 属于 T 的概率是 $\Pr(t)$ 。生成规则 R 规定了不能同时存在的元组, $R: t_{r1} \oplus \dots \oplus t_{rm}$, R 的概率 $\Pr(R)$ 为 R 中所有元组的概率和。 T 上所有的产生规则构成了产生规则集合 R_T 。一个可能世界 W 是 T 的子集。 W 的存在概率计算公式如式 (3-1) 所示。 $\Pr(W)$ 是所有可能世界的集合。

$$\Pr(W) = \prod_{R \in R_T, |R \cap W|=1} \Pr(R) \prod_{R \in R_T, |R \cap W|=\emptyset} (1 - \Pr(R)) \quad (3-1)$$

显然, 对于一个可能世界 W , $\Pr(W) > 0$, 进一步, $\sum_{W \in \omega} \Pr(W) = 1$ 所有可能世界概率和为 1。这个很明显, 就是将所有可能事件列举出来, 那么, 概率和必然为 1。

一个生成规则的长度(length)就是这个规则的元组的数目, 表示为 $|R|$, 就是它的秩。

一个可能世界 W 就是 T 的一个子集, 使得对每一个生成规则: 如果 $\Pr(R)=1$, 则 R 是独立元组, 则 $|R \cap W|=1$; 如果 $\Pr(R) < 1$ 则 $|R \cap W| \leq 1$ 。如果 $|R| > 1$, 则 R 是一个多元规则, 里面的元组是依赖关系, 否则是互相独立的关系。

对于一个不确定表 T , 它的生成规则集合为 R_T , 他的所有可能世界实例的数目就是那些概率为 1 的生成规则的长度与那些概率不为 1 的生成规则的长度加 1 的乘积。表示为:

$$|W| = \prod_{R \in R_T, \Pr(R)=1} |R| \prod_{R \in R_T, \Pr(R)<1} (|R| + 1) \quad (3-2)$$

一个 TOP-K 查询表示为 $Q^k(P, f)$, 包含一个谓词 P , 一个排序函数 f , 和一个整型 $k > 0$ 。 $Q^k(W)$ 表示 TOP-K 在可能世界 W 上查询 Q 返回的元组集合, 它包含 k 个元组。一个传统的 TOP-K 查询 Q 可以直接应用到可能世界 W 上, 用 $Q^k(W)$ 表示结果集中的 k 个元组。

一个元组 t 的 TOP-K 概率计算公式如式 (3-3) 所示, 在没有歧义的情况下我们简写为 $\Pr^k(t)$ 。则 PT-k 查询结果集中所包含元组的 TOP-K 概率至少是 p , p 是给定的极限值。

$$\Pr_{Q,T}^k(t) = \sum_{W \in \omega, t \in Q^k(W)} \Pr(W) \quad (3-3)$$

PT-k 返回所有在可能世界实例中成为 TOP-K 的总概率超过阈值的元组, 他的结果集合是:

$$Answer(Q, p, T) = \{t | t \in T, \Pr_Q^k(t) \geq p\} \quad (3-4)$$

Pt-k 查询没有经过优化的算法需要实例化所有可能世界实例, 这在时间和空间上的开销都是非常大的, 3.1 节也对这个问题进行讨论, 所以需要探索一种更加高效的算法来避免搜索所有可能世界实例。

3.4 Pt-k 查询算法统治集

3.4.1 统治集属性

Pt-k 算法可以找出概率较高的元组, 而不是排名靠前的元组, 因此, 可以考虑利用统治集属性计算某个元组在 TOP-K 里面出现的概率。

统治集可以这样理解, 那就是对于任意一个元组 t , 它在 TOP-K 查询结果受这个元组 t 前面的元组影响, 可以理解为元组 t 统治他前面的元组, 其排名与后面元组情况无关。

$P(T)$ 包含了所有满足查询的元组、元组的概率以及生成规则。去除不在 $P(T)$ 的元组不会影响 TOP-K 查询结果。因此, $Answer(Q, p, T) = Answer(Q, p, P(T))$ 。我们在 TOP-K 查询的时候只需要考虑 $P(T)$ 。

在 $P(T)$ 中的一个元组 t 是否成为 TOP-K 查询结果只受那些排列在 t 前面的元组的影响。

统治集(dominant set)就是对于 $P(T)$ 中的一个元组 t , 它的统治集就是 $P(T)$ 中所有排在 t 前面的元组的集合。表示为 $S_t = \{t' | t' \in P(T) \wedge t' \prec_f t\}$

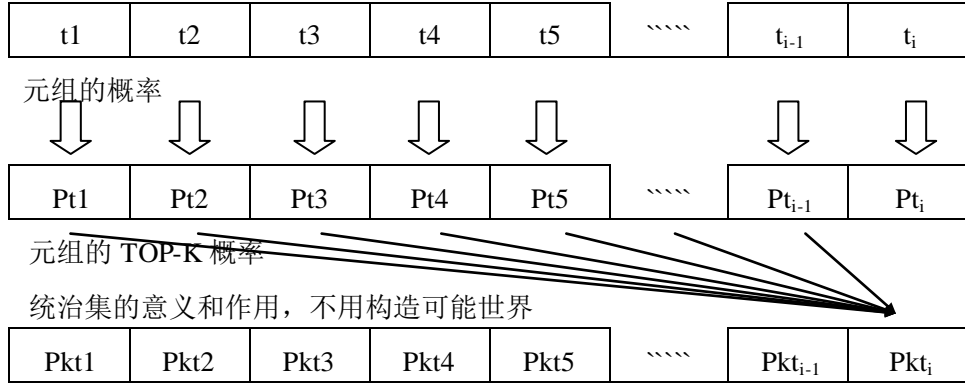
对于一个元组的统治集概率就是它的 TOP-K 概率, 可以用以下定理表示, 以后将会继续用到。要使用统治集, 那么有一个前提条件, 元组之间必须互相独立, 不在同一个生成规则中。因为, 如果前面有元组在同一个生成规则中, 那么元组不能同时计算它的概率, 不会在同一个可能世界中, 因此, 无法运用统治集。

统治集是与本元组无关的, 只与本身的前面元组有关的属性, 元组互相独立, 才能让所有互斥的元组不会出现在同一个可能世界中, 统治集的概念就是设法满足这个前提条件而提出来的。

定理 1 统治集属性(The dominant set property): 对于一个元组 $t \in T$, $\Pr_{Q,T}^k(t) = \Pr_{Q,S_t}^k(t)$ 。

证明: 假设每个元组都是互相独立的, 则可以通过式 (3-2) 和 (3-3) 可以推出上面的定理。

图 3-1 统治集属性概念描述



由上图可以看出，统治集的意义就是用来构造元组的 Top-k 概率，因为所有元组都是可能会在同一个可能世界出现，前面元组的概率极大影响在 i 位置的元组概率。位于 i 位置的元组能否出现在 Top-k 列表中的概率，取决于它前面元组的存在概率，如果前面某个元组存在，则会导致 i 位置的元组往后移，不存在则会导致 i 位置的元组位置往前移，将这个元组这两种情况的概率叠加，就是位置 i 元组在 Top-k 出现的概率。上图已经很好的表明这个意思。

3.4.2 统治集的位置概率

在 3.4.1 节我们已经对元组的统治集属性进行了一个简单的介绍，现在就对如何应用统治集进行一个相信的论述，统治集的位置概率可以说是本文中 Pt-k 算法的核心部分。

为了使用统治集，我们首先需要假设一个前提，那就是元组都是互相独立的。同时，他们是按照某种顺序排列的，那么，一个元组 $t_i \in W (1 \leq i \leq n)$ 出现在一个可能世界的第 j 个位置当且仅当它的统治集有 $j-1$ 个元组出现， $S_{t_i} = \{t_1, t_2, \dots, t_{i-1}\}$ 。

在一个可能世界中一个元组出现在第 j 个位置的概率，表示为 $\Pr(t_i, j)$ 。在一个可能世界中 S_{t_i} 出现 j 个元组的概率表示为 $\Pr(S_{t_i}, j)$ 。

一般地，我们可以知道 $\Pr(\phi, 0) = 1$ ，且 $\Pr(\phi, j) = 0 (0 < j \leq n)$ 。

一个元组出现在一个可能世界的第 j 个位置的概率等于该元组出现的概率乘以该元组的统治集有 $j-1$ 个元组出现在该可能世界的概率，公式为：

$$\Pr(t_i, j) = \Pr(t_i) \Pr(S_{t_i}, j-1) \quad (3-5)$$

显然，一个元组 t_i 的 TOP-K 概率为这个元组出现在第 1, 2, 3...k 个位置的概率和，表示公式为：

$$\Pr^k(t_i) = \sum_{j=1}^k \Pr(t_i, j) = \Pr(t_i) \sum_{j=1}^k \Pr(S_{t_i}, j-1) \quad (3-6)$$

显然，当 $i \leq k$ 的时候，一个元组 t_i 成为 TOP-K 的概率等于它本身的概率，也就是：

$$\Pr^k(t_i) = \Pr(t_i) \quad (3-7)$$

定理 2： 对于 $1 \leq i, j \leq |T|$ ，有：

1) 元组 t_i 的统治集 S_{t_i} 出现 0 个元组的概率为 t_{i-1} 不出现的概率与 t_{i-1} 的统治集出现 0 个元组的概率的乘积，公式为：

$$\Pr(S_{t_i}, 0) = \Pr(S_{t_{i-1}}, 0)(1 - \Pr(t_{i-1})) = \prod_{j=1}^{i-1} (1 - \Pr(t_{i-1})) \quad (3-8)$$

2) 元组 t_i 的统治集 S_{t_i} 出现 j 个元组的概率为 t_{i-1} 出现的概率与 t_{i-1} 的统治集出现 $j-1$ 个元组的概率的乘积与 t_{i-1} 不出现的概率与 t_{i-1} 的统治集出现 j 个元组的概率的乘积的和，公式为：

$$\Pr(S_{t_i}, j) = \Pr(S_{t_{i-1}}, j-1)\Pr(t_{i-1}) + \Pr(S_{t_{i-1}}, j)(1 - \Pr(t_{i-1})) \quad (3-9)$$

证明：由于是基于相互独立的元组，所以定理 2 可以很容易从概率的基本定理推导得出。

定理 2 可以很高效的计算 TOP-K 概率，我们可以给一个简单的例子。

例 3-2 我们可以给出一个按照得分函数进行排序的不确定性数据表 3-1 求出该表每个元组的 TOP-3 概率

表 3-4 排序后的不确定性数据表

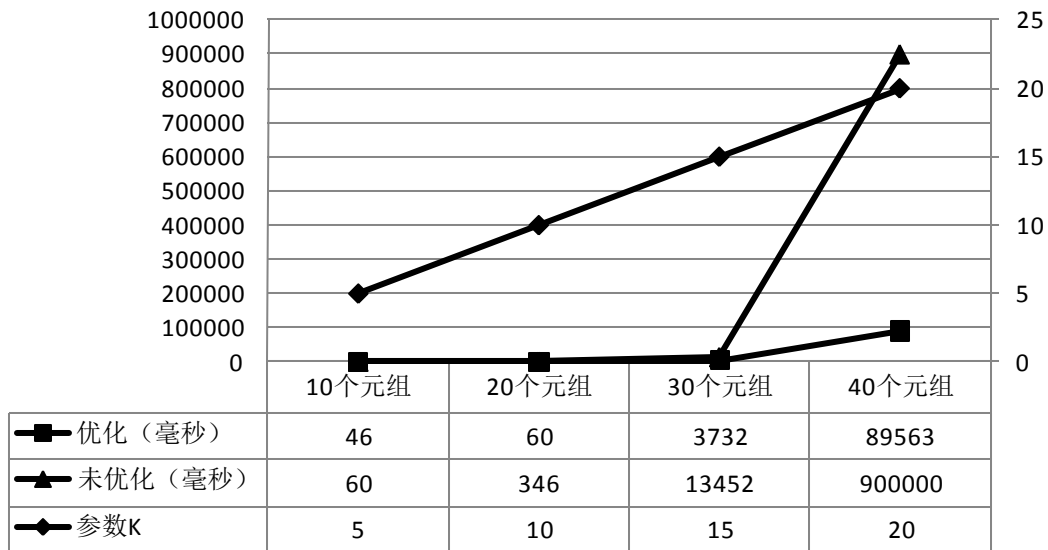
元组 ID	t1	t2	t3	t4	t5	t6	t7	t8	t9
元组存在概率	0.7	0.2	1	0.3	0.5	0.8	0.1	0.8	0.1

我们首先进行初始化，显然 $\Pr(\phi, 0) = 1$ ，且 $\Pr(\phi, 1) = 0$ ， $\Pr(\phi, 2) = 0$ ， $\Pr^3(t_1) = 0.7$ ， $\Pr^3(t_2) = 0.2$ ， $\Pr^3(t_3) = 1$ （因为 $0 < j \leq 3$ ）为了计算 $\Pr^3(t_4)$ ，我们首先利用定理 2 计算 $\Pr(S_{t_3}, 0) = 0$ ， $\Pr(S_{t_3}, 1) = 0.24$ ， $\Pr(S_{t_3}, 2) = 0.62$ ，则由式子 (3-9)，我们可以计算 $\Pr^3(t_4) = \Pr(t_4)(\Pr(S_{t_3}, 0) + \Pr(S_{t_3}, 1) + \Pr(S_{t_3}, 2)) = 0.258$ 。

由于统治集的位置概率是 Pt-k 算法中比较重要的部分，我们可以用一些测试数据测试其效率是否有明显的改善。

首先，我们输入几组已经排好序的 P(T)，且元组之间互相独立，设立阈值 $P = 0.2$ ，和没有使用统治集概率的 Pt-k 算法进行一个简单比较。比较结果如图 3-2：

图 3-2 优化后的算法效率



从图上的结果可以看出来, 未经优化的算法在元组数量 30 以上时已经远远超过优化后的算法 (900000 毫秒是程序运行时间上限值), 但是优化后的算法也不是说特别好, 在 40 个元组数量级就已经不是十分理想了, 可想而知, 统治集概率的优化还需要进一步, 优化的效率有提高, 但是还是不能满足现实生活中的庞大数据集还有多元生成规则。

这个小型的程序建立在随机生成元组和随机生成的生成规则的前提下进行的, 所以所有数据具有一定的离散型和随机性。未经过优化的程序, 意味着算法的效率比较低, 进行一个普通的查询就是一个几何级的增长, 可见如果仅仅是通过建立可能世界模型这个简单而且容易理解的模型, 是不能解决查询效率的问题。

同时, 可以看得出, 如果利用建立统治集元组的方法针对其进行改进的话, 算法效率有着明显的提高, 显然比不用更加快, 效率更加高。最重要的原因就是没有为每个元组建立他的可能世界模型, 改为计算它前面元组的存在概率。

3.5 处理多元规则

为了使定理 2 具有一般化, 我们首先需要处理元组的生成规则, 接下来就生成规则不同的情况进行处理, 务必使其符合定理 2 所要求的, 使元组间是互相独立的关系。

根据上述一节, 我们知道我们如果要用定理 2, 就必须是所有元组都是互相独立的情况, 由于元组之间还存在多元规则的关系, 我们不能马上就是用定理 2, 我们必须处理多元规则, 让所有元素都符合定理 2 的前提条件, 这样就需要多规则元组压缩策略、前缀共享技术和减枝技术。我们需要将非独立的元组组成的统治集, 转化成为独立元组组成的统治集。

3.5.1 多规则元组压缩

考虑一下 $P(T)=t_1, t_2 \dots t_n$, $P(T)$ 是已经排序好的序列。如果 $i < j$, 则 $t_i \prec_f t_j$, 我们如果计算 $\Pr^k(t_i)$, 就是每一个元组在 TOP-K 列表出现的概率, 同时已经知道多元规则 $R: t_{r_1} \oplus \dots \oplus t_{r_m}$ $1 \leq r_1 < \dots < r_m \leq n$, 则我们可以知道会出现以下几种情况:

情况 1: $t_i \prec_f t_{r_1}$ t_i 的位置比生成规则 R 中任何一个元组的位置要高, 根据定理 1, R 可以被忽略

情况 2: $t_{r_m} \prec_f t_i$ t_i 的位置比生成规则 R 中任何一个元组的位置都要低, 则 R 中最多有一个元组出现在可能世界。根据定理 1, 我们可以将规则 R 记为 t_R , 且他的联合概率为 $\Pr(R)$ 。

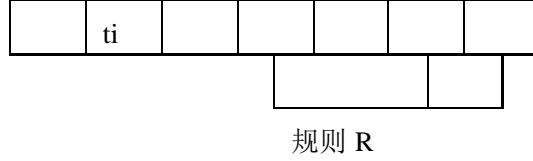
情况 3: $t_{r_1} \prec_f t_i \prec_f t_{r_m}$ t_i 的位置是在生成规则 R 中某些元组之间, 令 $tr_{m0} \in R$ 是排名比较高的元组, 多元规则 R , 则被分为 2 个部分, 左边是 $R_{left} = \{tr_1 \dots tr_{m0}\}$, 和右边部分 $R_{right} = \{tr_{m0+1} \dots tr_m\}$, 元组 t_i 的概率 $\Pr^k(t_i)$, 只受左边部分的影响, 右边部分可以忽略。

上面三种情况都是为了让之前的统治集概率一般化而作的条件假设, 对每一种条件, 我们需要让生成规则 R 中的元组对整个统治集概率没有影响, 从而可以使用定理 1。下面

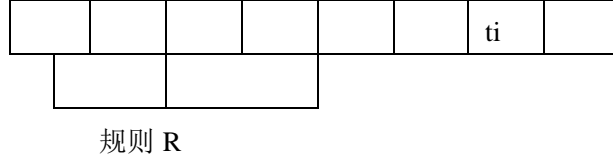
图 3-3 清楚表示每一种情况，并且就所遇到的每一种情况，提出一个可以解决的办法。

图 3-3 计算每一个元组的 $\Pr^k(t_i)$

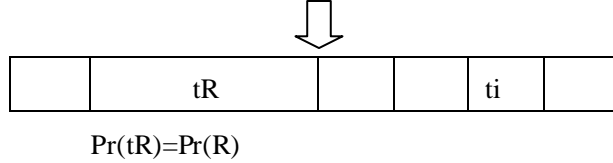
情况 1: t_i 比规则 R 的高



情况 2: t_i 比规则 R 的所有多元规则要高



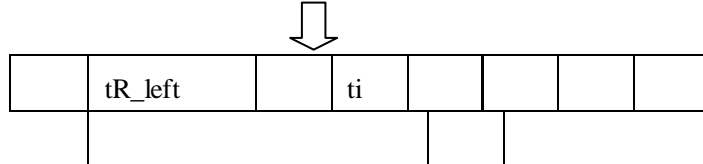
元组压缩之后



情况 3: t_i 被规则 R 的元组夹着规则 R



元组压缩之后



两种子情况也可能会出现，第一种如果 $t_i \notin R$ ，就好像情况 2 一样，我们可以将 t_i 左边的元组进行压缩，使其联合概率为 $\Pr(tr_1 \dots tr_{m_0}) = \sum_{j=1}^{m_0} \Pr(tr_j)$ ；

第二种情况，如果 $t_i \in R$ ， $t_i = tr_{m_0+1}$ ，在可能世界之中， t_i 出现了，所以 R 中的其他元组都不会出现。根据定理 1，这时在计算 $\Pr^k(t)$ 的时候只需要考虑排在 t 前面的并且不在 R 中的元组，也就是 $S_t - \{t' | t' \in R\}$

由上面的情况列举，我们可以发现如下两个推论，可以有效处理生成规则。

推论 1: 多元规则元组压缩：

对于一个元组 $t \in P(T)$ ，如果 $\forall t' \in R$ ， $t' \prec_f t$ 那么 $\Pr_{Q,T}^k(t) = \Pr_{Q,T(R)}^k(t)$ ， $T(R) = (T - \{t | t \in R\}) \cup \{tr\}$ ， tr 就是 R 中统治 t 的元组， $\Pr(tr) = \Pr(R)$ ，其他 T 中的生成规则依然留在 $T(R)$ 。

推论 2: 规则中的元组:

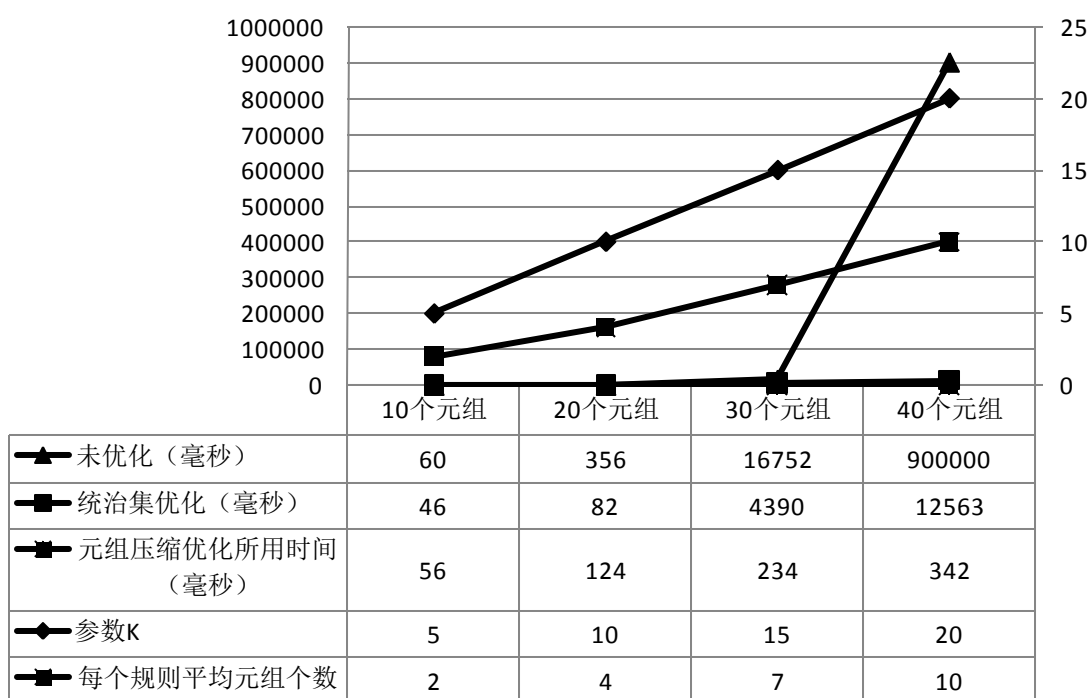
对于一个元组 $t \in R$, $P(t)=\text{true}$ 并且 $|R|>1$, $\Pr_{Q,T}^k(t)=\Pr_{Q,T'}^k(t)$, T' 为不确定表, $T'=(S_{t_i}-\{t'|t' \in R\}) \cup \{t\}$ 。

因此, 对于一个元组 $t \in P(t)$, 所有在 $T(t) \cup \{t\}$ 的元组都是独立的, 于是 $\Pr_{Q,T}^k(t)=\Pr_{Q,T(t) \cup \{t\}}^k(t)$, 这样就可以根据定理 2 来计算 $\Pr^k(t)$, 并且只需要扫描一次 $T(t)$ 。

我们可以根据得分函数将 $P(T)$ 里面的所有元组进行排序, 得到一个有序链表 L 。对于每个元组 t_i 扫描一次它前面的元组, 得到一个压缩的统治集 $T(t_i)$, 在这个统治集里面所有的元组都是独立的, 这样我们就可以使用定理 2 来计算 $\Pr^k(t_i)$ 了, 此时只需考虑 $T(t_i) \cup \{t_i\}$ 集合里面的元组。这样, 计算所有元组的 TOP-K 概率的实际复杂度为 $O(n^2)$, n 是不确定表的元组的个数。

在这里我们可以进行一次小测试, 看多元生成规则对整个列表的影响, 我们可以用回上一节的测试数据, 得出如下的结果, 如图 3-4。

图 3-4 元组压缩优化所用时间



从上面的图可以分析出, 元组压缩是让定理 2 一般化的手段, 因此他不会耗费大量的时间, 时间增长也是一种线性, 比较缓慢的增长, 对数据整体的改变不大, 几乎可以忽略不计算的。同时, 也发现, 时间消耗最为庞大的是统治集概率计算上。还有需要考虑一点, 如果用堆栈实现该算法, 是否会出现堆栈溢出的情况。在真正实验过程中, 我们会针对这种情况进行一个详细的讨论。

3.5.2 前缀共享技术

根据上面几节的阐述，我们可以知道，在可能世界模型中，采取回避生成可能世界的方法是最有效的。但是，我们也可以发现，对于统治集概率的计算，计算量还是十分巨大，时间消耗还是十分巨大。如何进一步提高查询的效率，将是我们这一节的主要任务。

通过数据的观察，我们注意到，对每个元组的统治集概率计算，每次都要计算它前面元组的概率，这样算法消耗的时间肯定会很大。所以，十分有必要避免计算前导元组的统治集概率，类似于避免生成可能世界。

但是，生成可能世界和统治集概率不同的两个概念。统治集概率始终需要计算，所以，我们只要尽可能利用已经计算过的统治集概率。

例 3-3 我们使用在例子 3-2 中的表 3-1, t_6 和 t_7 是相邻的两个元组,但是,因为 t_5 在 $T(t_6)$, 不在 $T(t_7)$, 所以计算它们的统治集的概率时候, t_5 是不被用到的。

同时的, $T(t_6)$ 和 $T(t_7)$, 共同有相同的子集 $\{t_1, t_2, t_4, t_3\}$, 这个子集的统治集概率将会重复计算, 那就是说, 如果他们的前缀共享的话, 我们将节省很多的时间成本, 因为计算统治集的概率是 Pt-k 查询算法中消耗最多的部分。

公式 3-5 表明, 使用子集概率 $Pr^k(S_{ti,j})$ 来计算 $Pr^k(t_i)$ 的时候, 在 S_{ti-1} 中的元组的顺序如何并没有关系。这就使得我们可以通过排序不同元组的统治集从而使得对应的子集的统治集概率值能够尽可能多的被共享。

首先, 如果元组 t 的情况正如情况 2 的话, t 是在 T 里面, 因此, t 应该在出现情况 3 前进行了排序。

其次, 如果有多个多元规则包含元组 t , 每一个规则 R_j 可以记为 t_{Rjleft} , 我们可以通过更新规则中的包含元组, 尽量少的改变共享前缀, 使其共享前缀尽量多,。

这时候有两种办法, 一种是积极的方法, 把所有独立的和完整的规则元组放在多元规则元组之前, 把规则中多元规则按它们出现的先后顺序来排序。

一种是消极的方法, 尽可能多的利用 $T(t_{i-1})$ 中的共有前缀, 尽量不重新排序, 只有当新来的元组会影响之前的元组的时候才要重排。

例 3-4 图 3-2 展示的是一个不确定性数据表 T 在排序后的情况, 有 2 个多元规则 $R1 : t1 \oplus t2 \oplus t8 \oplus t11$ 和 $R2 : t4 \oplus t5 \oplus t10$, 通过下面的图 3-2, 我们看到这两种前缀共享的方法不同的地方, 特别要关注他们所消耗的资源。

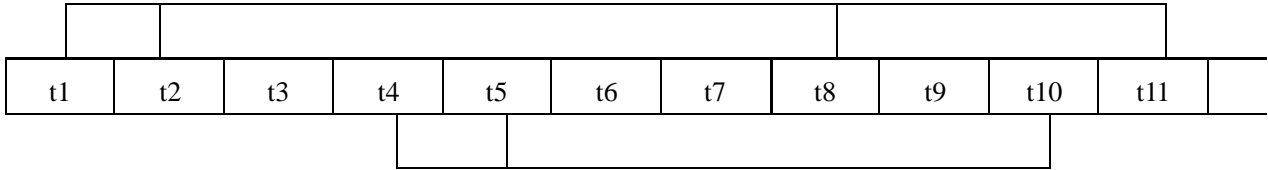
可以看到, 在开始的时候, 他们的前 3 个是一致的;

到了 $t4$ 的时候, 积极的方法要重新排 $t3$, $T(t3)$ 在前缀共享中无法使用, 但是在消极的方法那里没有重排, 所以 $T(t3)$ 共享前缀, 因此减少了消耗。

假设 L 为 $P(T)$ 中所有元组的有序链表, $L(t_i)$ 表示 $T(t_i)$ 中所有元组的有序链表, $Prefix(L(t_i), L(t_{i+1}))$ 为 $L(t_i)$ 和 $L(t_{i+1})$ 的最长的共同前缀, 那么, 总共需要计算的子集概率的数目为:

$$Cost = \sum_{i=1}^{n-1} (|L(t_{i+1})| - |Prefix(L(t_i), L(t_{i+1}))|) \quad (3-10)$$

图 3-2 两种重新排序的方法



P(T)排好序的列表

元组	积极方法		消极方法	
	前缀	Cost	前缀	Cost
t1	∅	0	∅	0
t2	∅	0	∅	0
t3	t1,2	1	t1,2	1
t4	t3t1,2	2	t1,2t3	1
t5	t3t1,2	0	t1,2t3	0
t6	t3t4,5t1,2	2	t1,2t3t4,5	1
t7	t3t6t4,5t1,2	3	t1,2t3t4,5t6	1
t8	t3t6t7t4,5	2	t3t6t7t4,5	4
t9	t3t6t7t1,2,8t4,5	2	t3t6t7t4,5t1,2,8	1
t10	t3t6t7t9t1,2,8	2	t3t6t7t9t1,2,8	2
t11	t3t6t7t9t4,5,10	1	t3t6t7t9t4,5,10	1
	总消耗: 15		总消耗: 12	

由以上的图，我们可以知道，相对于积极办法，效率比较高的是消极的办法，节省更加多的时间。但是，我们要考虑到一点，前缀共享技术主要是为了减少统治集位置概率计算时候的消耗。而统治集概率的计算时间，明显从图 3-4，可以看出效率比较低的，亟待需要优化。因此，需要对统治集概率计算可以重复利用的元组就重复利用。使用前缀共享技术，从理论上说可以实现减少统治集概率计算的时间，从上面的图 3-2 也可以得知，在计算时候，真正有用的共享前缀只有 t4, t5，所以对于其效率来说，改进幅度不是十分明显，同时由于需要维护一个共享前缀的列表，需要消耗更多的时间和空间，是否有必要进行前缀共享，我们在算法实现部分和实验部分继续进行讨论。

3.5.3 减枝技术

一个 PT-k 查询只对那些 TOP-K 概率超过阈值的元组感兴趣，因此可以通过剪枝的方法来避免检索所有的元组是否符合查询谓词。

一些已有的方法如 TA^[29]算法批处理排序。如果这种办法，我们可以对 P(T)进行有效的

检索。现在，我们关注如何进行减枝，以达到减少排序排名比较后的元组。

因此，给出 3 种减枝方法，可以不用计算它的 TOP-K 概率值就能将其去掉。一次检索就能将 $P(T)$ 中不能通过阈值 P 的元组去掉。

但是要注意的是我们仍然要检索排名较低元组 t ，因为它还是有可能超过阈值 P 的。

定理 3: $\Pr^k(t) \leq \Pr(t)$ ，对于一个独立元组 t ，如果 $\Pr^k(t) < p$ ，那么对于任意

- 1) t' 有 $t \prec_f t'$ ，且 $\Pr(t') \leq \Pr(t)$ 则 $\Pr^k(t') < p$;
- 2) 对于 $\forall t'' \in R$ ，有 $t \prec_f t''$ ，且 $\Pr(R) \leq \Pr(t)$ 则 $\Pr^k(t'') < p$

要用定理 3，则需要将比阈值小的最大的独立元组概率 p 保存，每次进行减枝时候，去检查元组是否小于这个值。

定理 4: 对于一个多元规则 R ，如果有 $t \prec_f t'$ ，且 $\Pr(t') \leq \Pr(t)$ ， $\Pr^k(t) < p$ ，则 $\Pr^k(t') < p$

在上面减枝技术的基础上，每一个生成规则 R ，我们将他其中不能通过阈值 P 的元组的概率值记录，以后如果检索到同一个 R 中的元组，如果他的概率没有大过保存的概率，则直接跳过该元组，因为根据定理 4，该元组 TOP-K 概率肯定就是小于阈值 P 。

定理 5: 令 A 为一系列成为 TOP-K 概率超过阈值 p 的元组的集合。

如果 $\sum_{t \in T} \Pr^k(t) > k - p$ ，那么对于每个没有在 A 中的元组 有： $\forall t \notin A$ ，有 $\Pr^k(t) < p$

我们可以通过观察所有 TOP-K 概率的超过阈值的值和为 k ， $\sum_{t \in T} \Pr^k(t) = k$

上面几个定理，是继续统治集概率之后最重要的技术，减枝技术，在实验过程中，我们会发现，减枝技术，给我们减少很多不必要的检索，不必要的计算，大大的减少计算时间。

首先我们要观察，在减枝技术中，对于定理 3 和定理 4 都是比较好理解的，定理 5，我们可以进行以下分析：

- 1) 假设集合 A ，是一切满足搜索结果元组的集合，且 k 是一个固定值，那么，相对于集合 A 会有任意元组 t ， $\Pr^k(t) > p$ ，集合 A 的秩为 n ，那么 A 集合所有元组的和如果为 $\sum_{t \in T} \Pr^k(t) > n \times p$
- 2) 对于秩 n ，如果 $k < np$ ，那么 $n > k/p$ ，又 $p < 1$ ，那么， p 足够小， n 必会大于检索元组总个数，可见， $k > np$
- 3) 如果 $\sum_{t \in T} \Pr^k(t) > k$ ，那么， $\sum_{t \in T} \Pr^k(t)$ 中，平均概率为 p' ， $np' > k$ ， $p' < 1$ ，如果 p' 很小，那么 $np' < k$ ，所以 $\sum_{t \in T} \Pr^k(t) \leq k$
- 4) 如果 $\sum_{t \in T} \Pr^k(t) < k$ ，那么， $np' < k$ ，又 $np < k$ ，而一般情况， $np < k < np'$ ，与条件相悖，所以 $\sum_{t \in T} \Pr^k(t) = k$ ，也就推出来定理 5

3.6 本章小结

通过上面针对 Pt-k 算法的改进和效率的优化，我们可以将整个算法的流程用伪代码写出来（图 3-3）。对于多元规则 $R: R: t_{r_1} \oplus \dots \oplus t_{r_m} \ 1 \leq r_1 < \dots < r_m \leq n$ ，令 $\text{span}(R) = r_m - r_1$ 。当元组 $t_i (1 \leq i \leq m)$ 进入队列时候，我们需要移除 $t_{r_1} \dots t_{r_{i-1}}$ ，还有计算他的统治集的概率。因此，最坏的情况就是对于每一个通过阈值 p 的元组，多要对每一个属于 $P(T)$ 的元组进行一次计算，时间复杂度是 $O(kn + k \sum_{R \in \mathcal{RT}} \text{span}(R))$

图 3-3 伪代码

Input: 不确定性数据表 T ，元组生成规则的集合 \mathcal{RT} ，一个查询 $Q_k(P, f)$ 阈值 p ;

Output: Answer(Q, p, T);

Method:

- 1: 对 $P(T)$ 进行检索排序
- for each $t_i \in P(T)$ do
- 2: 利用元组压缩计算集合 $T(t_i)$;
- 3: 计算元组 t 的统治集概率和 TOP-K 概率 $\text{Pr}^k(t_i)$;
- 4: if $\text{Pr}^k(t_i) > p$ then 输出 t_i ;
- 5: 检查 t_i 能否用于减枝技术;
- 6: if 所有在 $P(T)$ 中的元组都没有通过阈值 p then exit;
- end for

我将会在第四章阐述如何实现该算法，并在第五章阐述进行了相应的实验，来证明这些定理的有效性。通常的结果是，我们只需要对整个 $P(T)$ 的很小一部分元组进行检索就能得到 Pt-k 查询的结果。

第四章 Pt-k 查询算法的设计与实现

4.1 引言

根据前几章的理论基础和 Pt-k 查询算法的研究，我们接下来的工作就是对该算法进行设计与实现，以提高 TOP-K 算法的效率。首先我们先对上一章的结论进行分析，图 4-1 是由上一章得到的伪代码，接着会对该算法进行一个分析，看有哪些地方实现可以更加高效，还有如何避免数据的冗余，提高查询的效率这些将是本章的主要内容。

Pt-k 算法的改进已经在上一节有所体现，并将主要的理论进行一个简单的阐述。通过上一章的算法理论分析，主要思想是避免检索所有元组，避免重复计算统治集概率是上一章的研究重点。本章的研究重点将是在如何实现，算法流程方面上进行一个详细的考虑和阐述。并且提出一些在实现过程中的问题，和自己的改进意见。

4.2 算法流程分析

根据上一章的结论，我们可以从该算法的伪代码进行分析，设计高效的 Pt-k 查询。首先我们先对上一章的结论进行分析，图 4-1 是由上一章得到的伪代码，本节将会对以下这个算法进行一个分析。

图 4-1 伪代码

Input: 不确定性数据表 T ，元组生成规则的集合 \mathcal{RT} ，一个查询 $Q_k(P, f)$ 阈值 p ;

Output: Answer(Q, p, T);

Method:

```

1:对  $P(T)$  进行检索排序
for each  $t_i \in P(T)$  do
2: 利用元组压缩计算集合  $T(t_i)$  ;
3:计算元组  $t$  的统治集概率和 TOP-K 概率  $Pr^k(t_i)$ ;
4: if  $Pr^k(t_i) > p$  then 输出  $t_i$ ;
5: 检查  $t_i$  能否用于减枝技术;
6: if 所有在  $P(T)$  中的元组都没有通过阈值  $p$  then exit;
end for

```

4.2.1 运行效率分析

我们首先分析在该算法中，每一个环节所需要的开销，然后计算哪一个开销可以减少或者可以避免。表 4-1 罗列算法中将会有开销的部分，根据之前的讨论，我们总是想避免

检索所有元组，避免重复计算统治集概率，如何更合理利用已有或者已经计算的部分相当重要。

表 4-1 Pt-k 算法中有开销的部分

元组压缩
统治集概率计算
$\Pr^k(t_i)$ 计算
减枝技术

接下来我们就分析图 4-1 伪代码，从而分析该算法的效率。我们首先要理解一点，伪代码的作用是我们更容易理解算法的核心思想。

在伪代码中，我们没有展开的部分由元组压缩、统治集概率、减枝技术，所以接下来就是阐述如何解决这些问题，实现中如何解决各种情况，达到可以求出结果集的程度。

在元组压缩开销方面，应该首先有一个前提，那就是一个元组最多存在于一个规则 R 之中，就是说生成规则 R 之间的元组不重合、不重复。

在元组压缩阶段，根据前面 3.5.1 节的阐述，我们将有两个元组压缩的推论，在该节中，我们用一个比较简单的例子去定性研究了他们。

但是，在实现过程中，我们无法预知后面的原则规则，所以在实现的时候需要有对象去记录元组所在的规则。通过上一章的讨论，大致将元组情况分为三种，但是在实现过程中，还是会遇到各种各样的情况，而且各种情况可以分类讨论，如情况 1 可以进行一个延伸，分出三种子情况，这是之前讨论不够详细的部分，这里注重将上面没有讨论的部分进行一个相信的分析。通过分析，可以发现情况 1 是比较复杂的情况，处理的时候需要务必注意他们的前提条件和生成规则，不能在计算统治集概率的时候，出现无法计算的互斥现象。

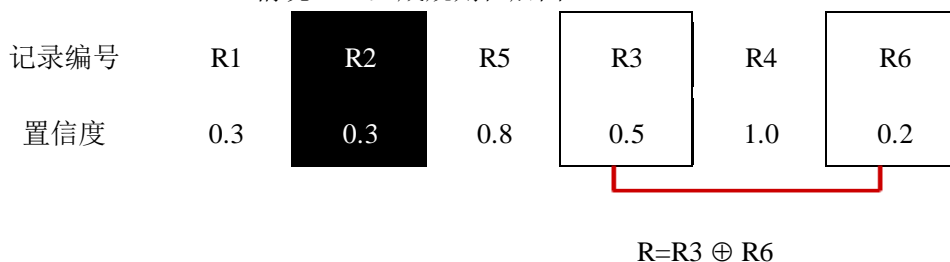
现在我将各种情况如何处理罗列出来：

图 4-2 如何处理多元组生成规则

情况 1 元组 t 不在生成规则之中

情况 1 的子情况

情况 1.1 生成规则在后面



情况 1.2 生成规则全部在前面

记录编号	R1	R2	R5	R3	R4	R6
置信度	0.3	0.3	0.8	0.5	1.0	0.2

$$R = R2 \oplus R3$$



记录编号	R1	R2,3	R5	R4	R6
置信度	0.3	0.3+0.5=0.8	0.8	1.0	0.2

情况 1.3 元组 t 在生成规则之间

记录编号	R1	R2	R5	R3	R4	R6
置信度	0.3	0.3	0.8	0.5	1.0	0.2

$$R2 \oplus R3 \oplus R6$$



记录编号	R1	R2,3	R5	R4	R6
置信度	0.3	0.3+0.5=0.8	0.8	1.0	0.2

$$R2,3 \oplus R6$$

情况 2 元组 t 是规则 R 中的元组

记录编号	R1	R2	R5	R3	R4	R6
置信度	0.3	0.3	0.8	0.5	0.5	0.2

$$R = R2 \oplus R4$$



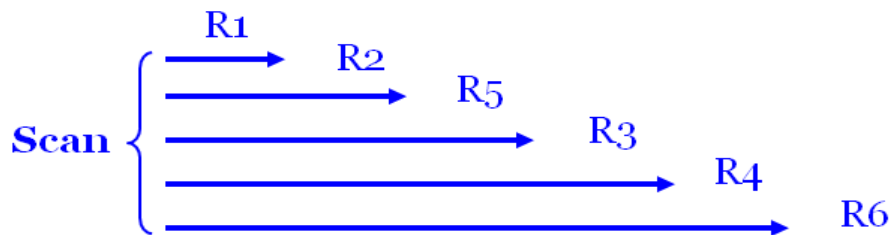
记录编号	R1	R5	R3	R4	R6
置信度	0.3	0.8	0.5	0.5	0.2

由上面的图可以很简单分析出元组压缩的伪代码（图 4-3），并且可以通过观察发现他的复杂度为 $O(n^2)$ ，如图 4-4，元组压缩也是需要消耗一定的时间，需要对整个列表进行一个重新排布，我们应该考虑如何才能减少对整个列表的重新排序。通过观察可以发现，如果我们对每个元组进行一个简单的属性检查，检查它是否符合某个生成规则即可。

图 4-3 元组压缩的伪代码

```

For 在序列化后的每一个元组 t
    For 在统治集  $S_t$  中每一个元组  $t'$ 
        计算  $t'$  的统治集
    
```



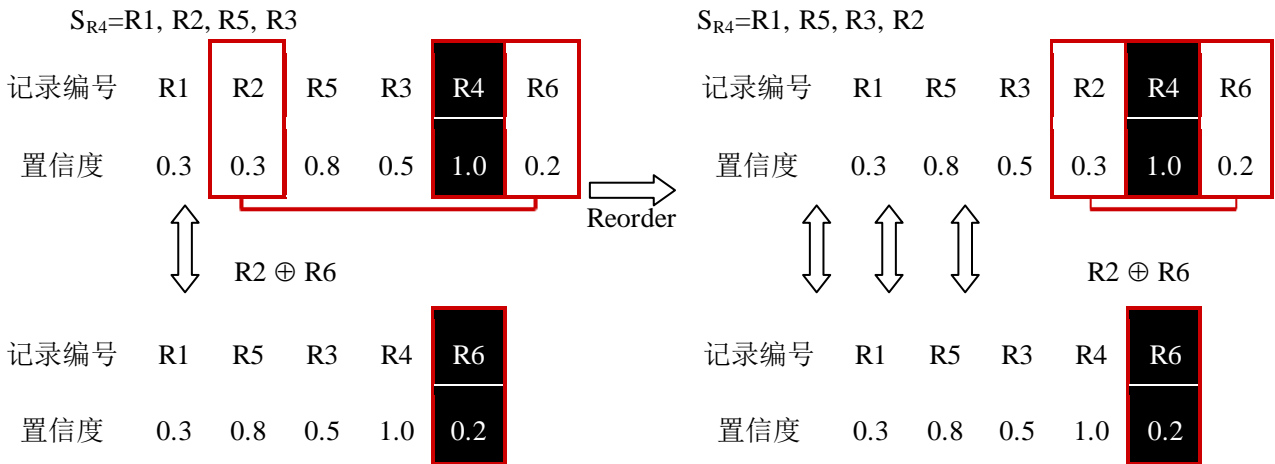
记录编号	R1	R2	R5	R3	R4	R6
置信度	0.3	0.4	0.8	0.5	1.0	0.2

图 4-4 元组压缩检索顺序

从上面的分析来看，元组压缩需要一定的空间和需要检索时间，每一次元组压缩都会让序列发生变化，我所采取的策略就是将整个序列保存在字符串中。这样检索的时候有几个好处：以流的方式保存数据，无论是存储还是查找，效率和速度都会有多提高，每次更新序列的时候，只需要将需要修改部分替换就行，每一个元组 t 关于更新元组压缩不确定性数据表 T 时候的复杂度为 $O(n)$ 。

关于前缀共享的方法，在第三章阐述了理论基础，现在将其模型化，用图 4-5 表示。可以看到前缀共享在计算统治集概率时候的作用是十分有效，避免了重复计算相同元组列表的统治集概率。

图 4-5 前缀共享模型



$S_{R6} = R1, R5, R3, R4$

共享前缀: R1

$S_{R6} = R1, R5, R3, R4$

共享前缀: R1, R5, R3

假设 L 为 $P(T)$ 中所有元组的有序链表, $L(t_i)$ 表示 $T(t_i)$ 中所有元组的有序链表, $\text{Prefix}(L(t_i), L(t_{i+1}))$ 为 $L(t_i)$ 和 $L(t_{i+1})$ 的最长的共同前缀, 那么, 总共需要计算的子集概率的数目为:

$$\text{Cost} = \sum_{i=1}^{n-1} (|L(t_{i+1})| - |\text{Prefix}(L(t_i), L(t_{i+1}))|) \quad (4-1)$$

在这里可以看到, 关于前缀共享的开销是一个代数级增长的关系, 所以设计的时候针对其如何更快更新重排后的序列和更新共享前缀, 使共享前缀更加多。

关于减枝技术的模型, 我在这里简单阐述一下设计模型, 见图 4-6

图 4-6 减枝技术模型

情况 1 R_i 和 R_j 都是互相独立, 不在同一个 R

记录编号					R_i				R_j			
置信度					0.8				0.6			

情况 2 R_i 和 R_j 不是互相独立, 都在同一个 R

记录编号					R_i				R_j			
置信度					0.8				0.6			

... $R_i \oplus R_j$...

如果 $\text{Pr}(R_i) \geq \text{Pr}(R_j)$, 那么, $\text{Pr}^k(R_i) \geq \text{Pr}^k(R_j)$, 如果 R_i 不在结果集中, 则 R_j 也不在结果集中

通过上述模型阐述可以知道，减枝技术也是代数级的增长，我们每次都只需要将不符合结果集的最大的概率记录，每次和新加入的元组进行一个比较，就可以知道是否应该计算它的 TOP-K 概率或者统治集的概率。在这一步中，可以有效减少无关，或者减少计算概率的元组的数量。

以上的阐述就是关于压缩规则、前缀共享和减枝技术的开销计算，在实验中发现，这几个技术模型，都不是开销主要的方面，开销最大的是进行统治集的计算，特别是当统治集十分巨大的时候，如果前面元组数量超过 100 的话，时间复杂度是几何级的增长，所以如果针对统治集概率计算模型才是关键因素。

对于整个算法来说，它的复杂度有如下的分析。对于多元规则 $R: t_{r_1} \oplus \dots \oplus t_{r_m} \mid 1 \leq r_1 < \dots < r_m \leq n$ ，令 $\text{span}(R) = r_m - r_1$ 。当元组 $t_i (1 \leq i \leq m)$ 进入队列时候，我们需要移除 $t_{r_{i-1}}$ ，和 t_{r_i} 进入队列。因此，最坏的情况就是对于每一个生成规则都要计算它的统治集概率值，复杂度为 $O(2k \cdot \text{span}(R))$ 。而且最坏的情况是，对于计算每一个 $P(T)$ 是否通过阈值，所有在 $P(T)$ 的元组都要至少读取一次，时间复杂度是 $O(kn + k \sum_{R \in \mathcal{RT}} \text{span}(R))$

4.2.2 设计目标

实现 Pt-k 算法，并且根据该算法分析其可以改进的地方，结合上面的理论分析，我认为实现后应该达到的目标为：

- a) 运行可靠
- b) 操作简单
- c) 容易获取精确的计算开销
- d) 不实现界面，因为点击按钮等响应事件会影响时间开销计算

4.3 不确定性数据在 Pt-k 查询算法中的实现

为了精确计算该算法的效率，这一节中，将阐述一种简单的有效方法生成不确定性数据样例的方法。

4.3.1 数据样例分组

对于一个元组 t ，令 X_t 是一个指向在随机数构成的经过排序的不确定性数据表的指针。如果 t 在 TOP-K 列表中，则 $X_t = 1$ ，否则 $X_t = 0$ 。显然， t 的 TOP-K 的概率是 X_t 的期望， $\Pr^k(t) = E[X_t]$ 。我们的对象就是生成一些不可能世界样例的集合 S ，计算在 S 中的期望 X_t ， $E[X_t]$ 。

我们用一些格式化的数据去替代。对于一个不确定性数据表 T 和生成规则的集合，一个样例单元 s 就是一个可能世界。基于不确定性数据表 T 的分布生成样例单元：每次生成一个样例单元 s ，就扫描 T 一次。一个独立元组 t_i 在 s 中就有概率 $\Pr(t_i)$ ，一个生成规则在 s 中的概率就是 $\Pr(R)$ 。如果 s 拿到生成规则 R 中的一个元组，那么元组 $t_i (1 \leq i \leq m)$ 就会有

概率值 $\frac{\Pr(tr_l)}{\Pr(R)}$ 。一个样例单元 s 最多就只能拿到生成规则 R 的一个元组。

只要生成样例单元 s ，我们就可以计算它的 TOP-K 的元组。对于每一个在 TOP-K 序列的元组，它的 $X_t=1$ ，其他元组则为 0。

上面的生成样例的生成过程在一个样例集合 S 中可以重复获得，那么样例集合 S 的期望 $E_S[X_t]$ 就可以近似 $E[X_t]$ 。当样例样本的容量足够大的时候，近似的质量就会越好。

关于样本容量，可以通过以下的定理获取。

定理 6: 如果对于任意 $\delta(0 < \delta < 1), \varepsilon(\varepsilon > 0)$ 和一个可能世界的样本 S ，如果 $|S| > \frac{3 \ln \frac{2}{\delta}}{\varepsilon^2}$ 对于任意元组 t 有 $\Pr\{|E_S[X_t] - E[X_t]| > \varepsilon E[X_t]\} \leq \delta$

证明：这个定理可以通过切尔诺夫界^[30]推导得出。

4.3.2 生成样例的实现策略

上面所说的样本方法有以下两种实现策略：

第一种方法是：我们可以对 $P(T)$ 进行排序然后放在列表 L 中，在样例单元 s 中的前 k 个元组就是该单元的 TOP-K 元组。因此，当生成一个样例单元，而不是扫描整个不确定性数据表，只需要扫描列表 L 。只要样例单元中有 k 个元组的话，那么扫描可以结束。这种方法可以减少生成样例单元时候的开销，但是不影响样例的质量。为了看到效果，考虑一下不确定性数据表中所有数据都是互相独立的。如果平均概率是 μ ，那么需要生成样例单元而扫描的元组有 $\left\lceil \frac{k}{\mu} \right\rceil$ 。这样的话，在 k 很小的情况下，扫描数量远远少于 $|P(T)|$ 。最坏的情况是，对整个列表 L 进行一次扫描。

第二种办法是：一般的，在还没有达到基于定理 6 所得到的样本容量的时候，近似值已经非常精确。因此，可以对生成的样例抽取的办法进行改进：我们可以在得出每一个样例单元之后，才计算它的 TOP-K 概率。对于给定参数 $d>0$ 和 $e>0$ ，如果在最后的 d 个样例单元中，任意元组 t 的 X_t 期望小于 e 时候可以停止扫描。

在第五章的实验中，我们将证明 Pt-k 算法和上面数据的处理办法的效率。

我们需要注意的一点就是，我们到此的所有讨论都是围绕如何减少扫描的次数展开讲述的，但是惟独这一节对数据的实现进行一个简单的描述，从而我们可以知道，如果在实际数据中使用这个算法，还是需要一定的处理，原始数据无法进行一个查询，这也说明不确定性数据的查询还有待完善，关于数据处理方面的研究不是本篇文章研究重点，因此，在这里仅仅以这一节进行一个简单的描述，达到说明数据处理的重要性。

4.4 运行平台与开发工具

操作系统: Windows7 64 位旗舰版
CPU: 英特尔 i5-430
内存: 2G

开发工具: Microsoft visual studio team suite 2008
开发语言: C++

4.5 本章小结

本章首先介绍了在 **Pt-k** 查询算法的基本流程, 以及根据该算法流程, 分析和设计元组压缩、统治集概率、前缀共享和减枝技术这四个方面的模块, 并且针对每一个方面的模型细节进行一个详细的阐述。然后分析不确定性数据中的数据处理方法, 提出了两种样本数据生成的方法模型。在样本数据处理阶段, 尽量将 $P(T)$ 的所有元组进行扫描, 避免计算统治集概率, 然后采取一种近似值的策略, 尽量让数据精确的情况下进行计算, 最后得出最终结果。本章是第五章实验的设计基础, 第五章通过一些数据会对本章的设计模型进行一个效率分析和验证。

本章可能对某些模型相对于第三章理论进行了一个优化, 目的就是为了更加贴合优化统治集概率, 实现更加高效的 **Pt-k** 算法。

第五章 实验及性能分析

5.1 实验环境和测试数据集

实验采用带有 Inter 处理器、2GB 内存的 PC 机，内装 WINDOWS 7 Ultimate 64 位操作系统，整个算法用 Microsoft Visual C++2008 实现。为了体现 Pt-k 算法的效率和优越性，收集了其他 TOP-K 算法的实验结果进行比较。在实验中，使用的测试数据集是模拟数据集。

和 Pt-k 算法比较的 TOP-K 算法主要有 U-kRanks 和 U-Topk 这两种不确定性数据的 TOP-K 算法。使用模拟数据集的最大原因有三个，一个是我们主要关注点是 Pt-k 算法的效率和优势，不是处理数据的能力。第二点就是根据现在不确定性数据的研究，概率维度大多都是人工生成，不是固有属性，也不是十分精确，同时概率维度也是一个容易变化的变量。第三，那就是跟 Pt-k 算法共同比较的那几种的算法都是属于 TOP-K 算法，和 Pt-k 算法具有相同的数据源，主要使用的都是单因子数据，因此，使用各种 TOP-K 算法进行效率测试评估的时候，主要关注点是数据的结果集和数据运行效率。

模拟数据集具有可以控制数据点的生成规则的组合、概率值、数据的元组个数和分布演化的特征，所以它对于测试 Pt-k 算法的效率来说，是理想的测试工具。对于 TOP-K 算法，模拟数据集优于实际数据集，实际数据集需要进行得分函数处理或者不确定性数据处理，才能得到与模拟数据集相近的不确定性数据。且在实际数据中，往往没有生成规则的组合和概率值，也需要进行计算，不是本论文主要研究的方向，所以，没有必要计算其效率。模拟数据集是较为理想的输入数据。所以论文主要是利用模拟数据集来测试和比较 TOP-K 算法的效率。

5.2 实验设计

5.2.1 数据源

为了验证 Pt-k 算法的效率，我们采用两种数据源。一组是离散的概率值和生成规则的组合，另一组是具有一定分布的概率值和生成规则的组合。数据集和生成规则的组合元组个数有上限，在上限值内进行随机分布和取值。

数据源一：

数据源一包括了得分值、概率值和生成规则的组合，而且是完全离散分布的。为了达到这个目的，使用随机函数，随机函数的结果是用获取计算机瞬时时间作为算子得出的，且每组生成规则和生成概率值的时候，都会重新获取一次系统瞬时时间，为了保证每次获取的时间值不同，采取精确到微秒的 WIN32 时间函数 QueryPerformanceCounter。这样得出的数据集是一个高度离散的数据集。

数据源二：

数据源二同样包括了得分值、概率值和生成规则的组合，首先生成一个正态分布，然后进行随机分配。使用具有一定分布的数据源主要是考虑到，当数据值或者概率具有一定分布的时候，算法的运行效率如何的问题，是否与离散的数据源相同。这样可以比较算法的性质差异。因为在实际数据集中，有部分数据集是以分布的形式出现的。

5.2.2 评估标准

实验主要的目的是为评估 Pt-k 算法和其他 TOP-K 算法的性能，所以我们选用以下两个参数作为评判算法性能的标准。

- 计算速率：用以计算一个查询算法一组数据源所需要的时间。算法的计算速率可以确定一个算法是否可以实现高效的查询。
- 准确率：用以评估算法在结果集中是否会出现重复的数据，出现重复的数据说明查询算法在某些方面有些缺陷。
- 堆栈峰值：由于在计算过程中，统治集概率计算需要利用堆栈，对于较大的数据量，无论是机器的内存还是 IDE 预设的堆栈，很有可能发生堆栈的溢出。打个比方，Delphi7 预设的堆栈为 1000 个，超过这个数目，就会自动跳出函数，继续执行，这样肯定就会导致程序执行错误。不过考虑到测试数据，在本次实验中没有超过 IDE 预设的堆栈峰值。

5.3 Pt-k 查询算法应用

5.3.1 数据预处理

由于数据源二中共包含三种不同的分布方式，可以是只有概率值具有一定的分布，可以是只有得分函数的数值具有一定的分布，也可以是概率和得分函数的数值都具有一定的分布，所以我们生成数据的时候需要考虑到这三种情况，同时也需要评估这三种情况的准确率。

对于随机函数生成的概率和生成规则的随机，我们采用的是二次随机的策略，即首先生成所有元组的概率，然后随机生成的生成规则，由于生成规则中，所有元组的概率和小于 1，因此，对于在生成规则中的元组，需要再次计算它的概率值，务必使所有不确定性数据满足条件，同时，不会产生错误的数据，保证生成的数据的正确性。

对于随机函数，我们通过改变算子的方法得到不同的数据，由于根据现有的技术无法保证所有数据做到随机，所以本次实验中的所有生成数据，或者生成规则，或者概率都是通过伪随机的方法生成的。

对于数据、生产规则和概率，为了达到一个随机分布的效果，还有保证数据的正确性和可控性，采取以下伪代码处理方式：

图 5-1 伪代码

Input: 元组个数 MAXN, 生成规则的个数 MULTIRULE, 每个生成规则所包含的元组范围 RULEMIN 和 RULEMAX;

Output: List;

Method:

For each ti, $i < \text{MAXN}$

- 1: 生成随机分布数据;
- 2: 加入数据列表 datalist;
- 3: 加入概率列表 prolist;

End for

- 4: 随机计算一个规则包含元组的大小 rulecount, 范围在 RULEMIN 和 RULEMAX 之间;

For each i, $i < \text{rulecount}$

- 5: 对 1 进行随机分割成 rulecount 块, 每块就能保证随机;
- 6: 加入生成规则列表 rulelist;
- 7: 随机生成 0 至 MAXN 的一个值

If 生成规则标志位为没有标志

- 8: 修改 prolist 的概率值, 随机决定其是否减去一个随机值
- 9: 修改标志位, 标志其已经加入某个生成规则;

End if

End for

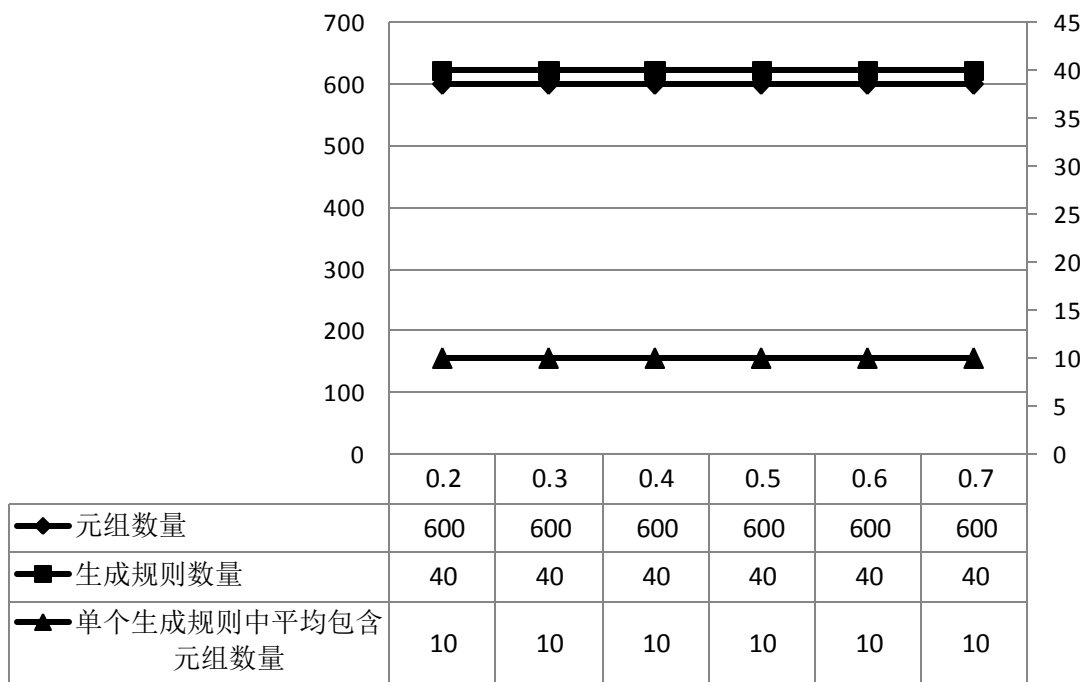
从上面的伪代码可以知道, 实验室尽量符合, 还有优化需要用到的元素, 让整个 Pt-k 算法更加完整, 效率更加高而进行的计算。而且现实数据库也可以知道, 大量的不确定性数据都会有如上的属性——数据、概率值、生成规则。我只是通过一些随机分布计算方式, 将演算过程进行压缩, 以达到我们研究 Pt-k 算法的目的。

根据图 3-2 的运行结果, 明显可以看出, 没有优化过的其他 TOP-K 算法根本就无法跟优化后的 Pt-k 算法进行同一种数量级的比较, 因此, 我们主要根据现有的一些优化技术进行一个比较, 测试 Pt-k 算法在上述不同数据模式下的一些结果。由于在优化 Pt-k 算法中, 我们需要优化的主要部分是统治集概率, 因此, 我们需要针对统治集概率优化进行一个验证。同时, 由于统治集概率优化中, 存在两种不同的算法, 一个是积极的前缀共享, 一个是消极的前缀共享, 需要对此进行一个进一步的分析 and 研究。

5.3.2 阈值不同时的运行结果

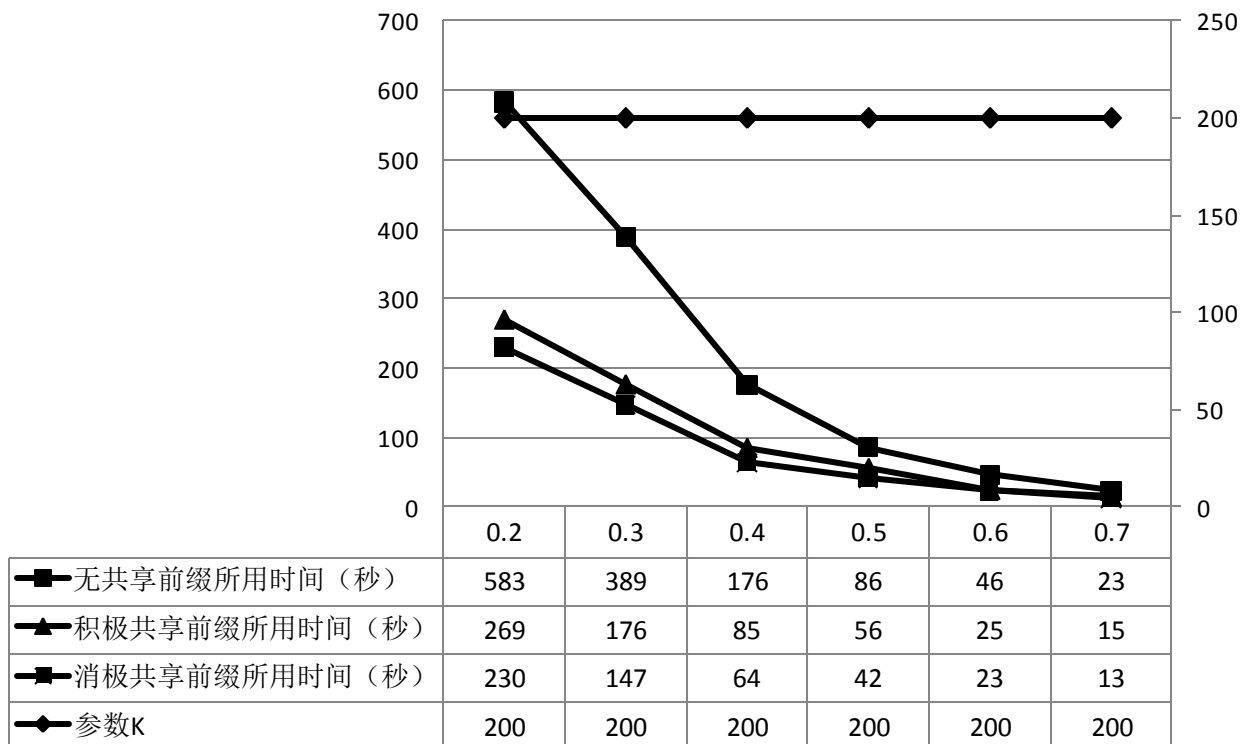
对于不同类型的数据, 我们首先列举我们即将进行实验的数据范围, 还有需要进行测试的数据集合。

图 5-1 测试数据参数分布



从上面的测试数据可以发现，我们关注的重点是阈值改变对整个查询的时间有何影响，变量约束为阈值。为了让结果更加明显，我们采用了多次生成数据，取它平均值方法进行测量，运行结果为图 5-2：

图 5-2 不同阈值情况下的运行情况

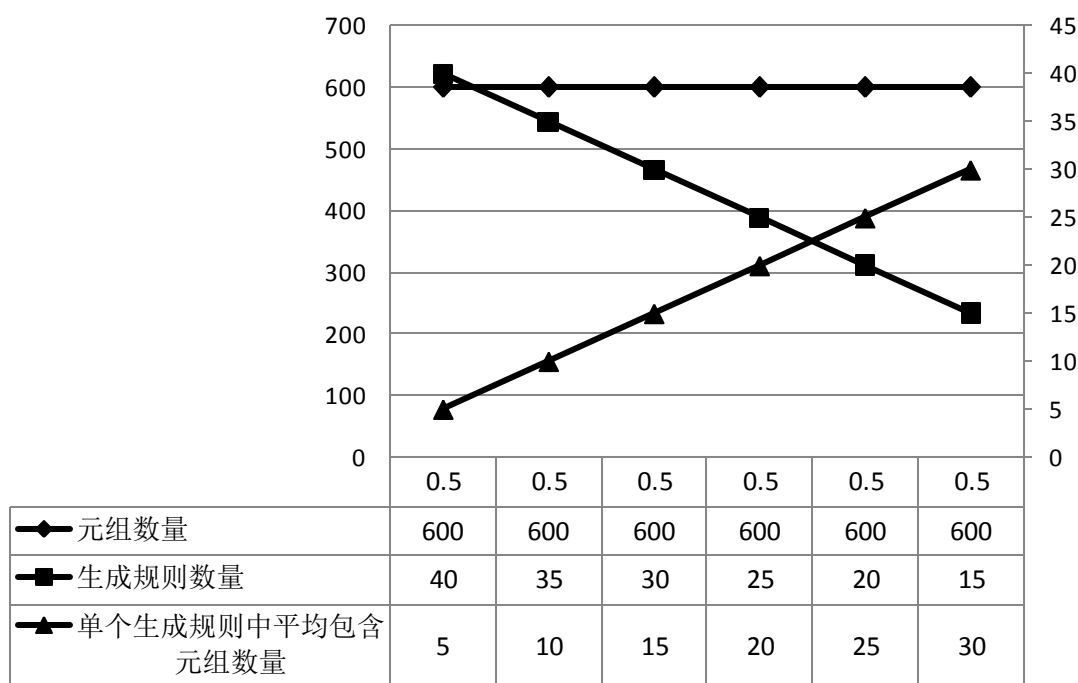


从上面的实验结果可以发现，查询的效率发生着巨大的改变，对于较大的阈值，可以知道的是，阈值 p 发挥减枝作用，根据上两章的讨论，阈值最大的作用就是减少元组扫描的数量，提高程序检索的效率。在这个实验中，可以明显看出，改变阈值让减枝技术发挥了作用，先前的理论是正确的。同时，也可以看出，用户对于选择不同的阈值对程序的运行效率有着重大的影响。

5.3.3 生成规则发生变化时运行结果

生成规则作为不确定性数据中的一个重要属性，对它的影响和效率十分有必要进行一个度量。我们首先固定其他变量，只修改生成规则的数量和生成规则中平均包含的元组数量。如图 5-3 所示，我们关注的重点在于如何改变生成规则数量。

图 5-3 测试数据参数分布

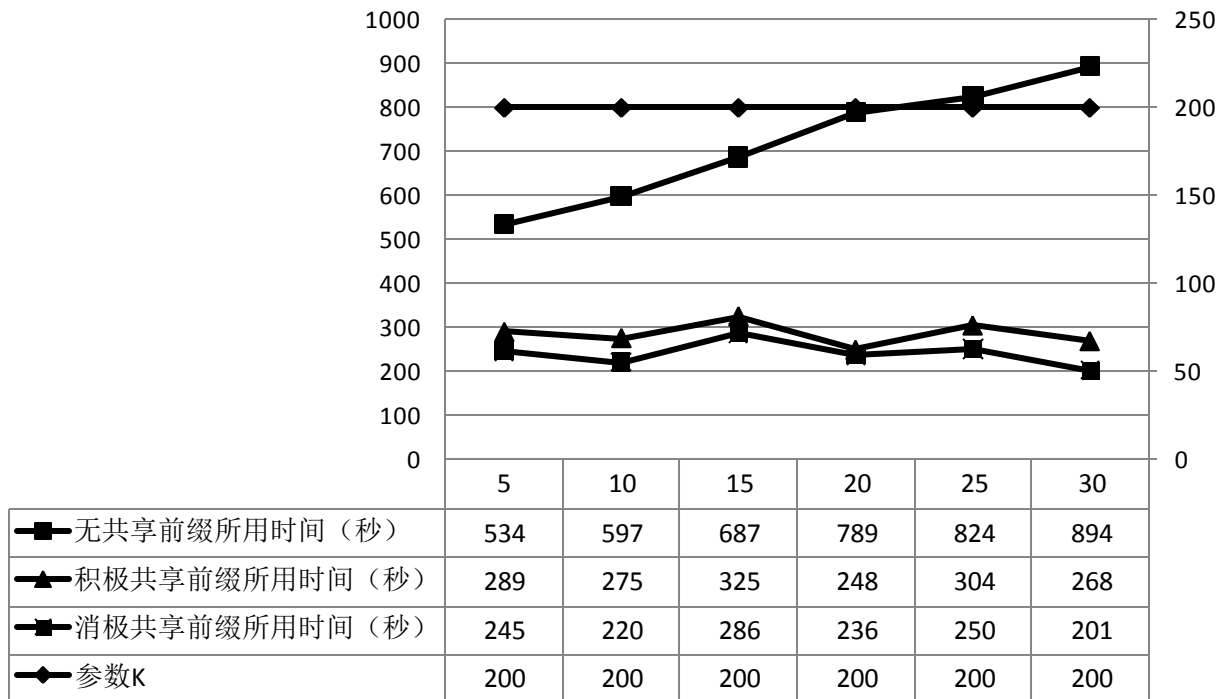


从上面的数据进行演算，计算不同生成规则数量对整个程序的运行效率有什么影响，我们将结果展示，如图 5-4，我们可以发现，生成规则的数量对整个算法的效率不是关键因素，在实验中，我们通过对运行时间进行一个检查，对于较大的数量的元组，我们在不断改变元组的生成规则之间的关系，可以明显观察到，利用元组压缩技术，维护重排列表，对于整个程序的运行时间来说，是一个较小的时间。

我们选取的数据都是模拟数据，而且是经过一定的处理，具有一定的离散性，可能和实际的数据表现不同，这个是有可能的。因为在实际数据中，生成规则不会如此稳定的出现，出现的方式和现象可能有所不同。举个简单的例子，如果一个元组在多个生成规则中出现，那么整个算法的效率如何表现，这个我们并没有分析，也没有具体的讨论。不过，也可以利用相同的思路，将整个数据集变为单一生成规则的原则进行一个处理。

算法运行结果是：

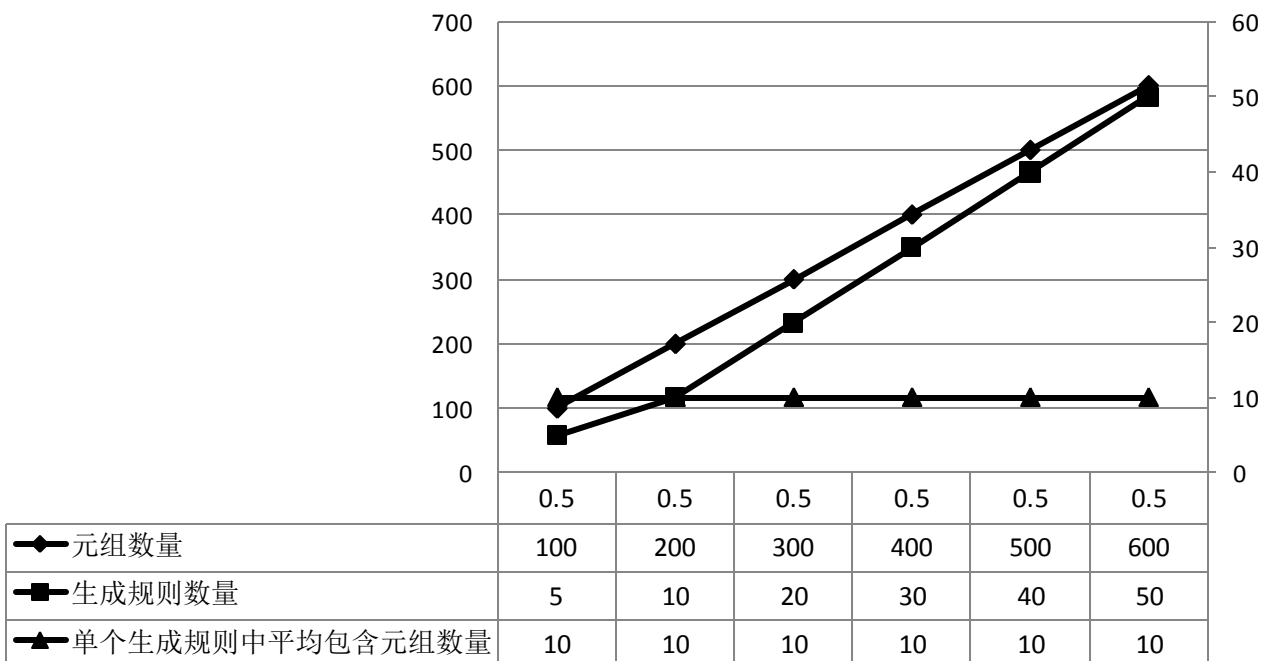
图 5-4 不同元组规则的测试结果



从上图结果显示，我们可以明显观察到元组的生成规则在整个算法流程中起到数据的预处理的作用，并不会对整个程序的运行效率有所影响。

5.3.4 元组数量发生变化时运行结果

图 5-5 测试数据参数分布

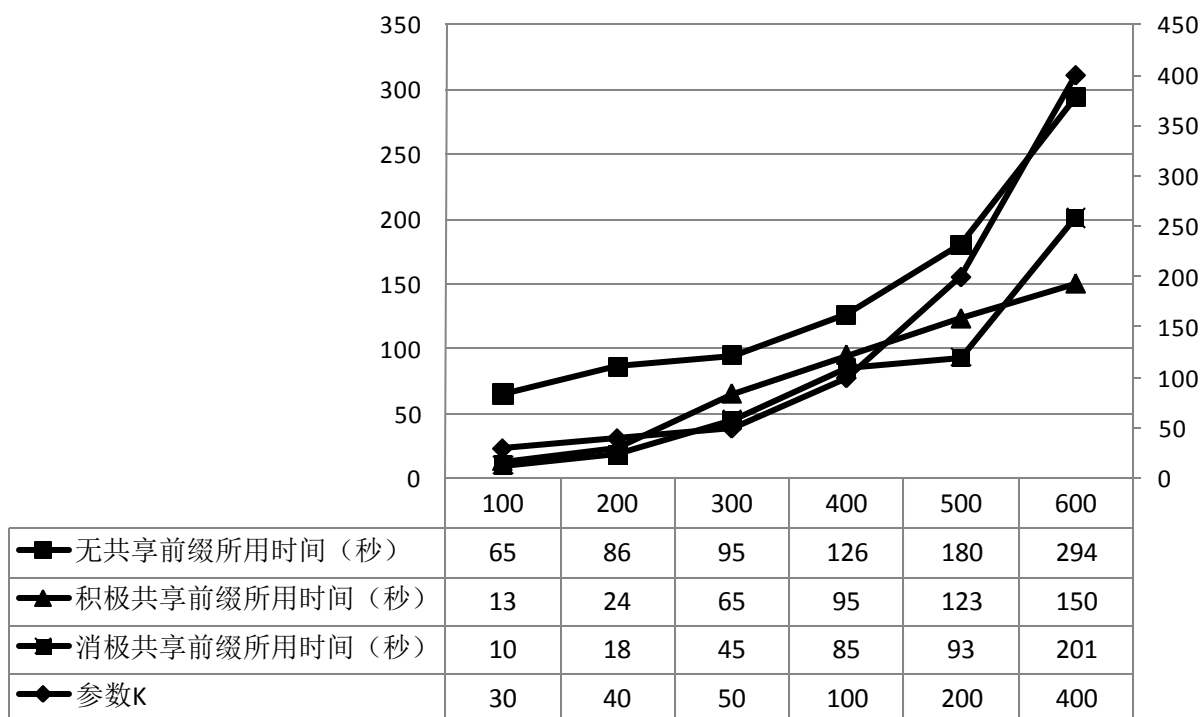


上面两个小节已经对元组的阈值和生成规则数量进行一个简单的验证测试，结果就是生成规则的元组压缩策略是一种数据的预处理作用，让数据达到符合数据要求的作用，因此，它的运行效率对整个算法的效率影响不大。

不同的元组数量是否会对整个算法效率有影响，不同的参数 K 对整个算法运行效率是否有影响，整个算法的运行结果如图 5-6

程序运行结果：

图 5-6 不同元组数量运行时间



从上面图 5-6 运行结果所示，整个算法的运行效率是关于元组数量敏感，这个结论也可以通过之前的分析可以知道，如果我们将数据固定，只改变数据的扫描数量得到的结果也是一样的。

从上面的数据也可以分析得出，元组的数量对整个算法的运行效率起着重要的参考作用，为了验证数据的敏感度，算法的运行效率，利用监控程序运行时间，可以发现，当元组数据量比较大的时候，越是计算靠后的元组的统治集概率，所耗费的时间越多。

正如之前讨论的一样，整个算法的运行时间取决于算法扫描元组时候，计算元组统治集概率的时间，如果需要统治集概率计算的元组数量越多，则耗费的时间久越多，对整个算法效率影响也就越大。

对于参数 K 的改变也是相同的，如果让排名较后的元组计算统治集概率会严重影响算法的效率，因此，如果参数 K 较大，则运行效率也会较低，时间更长，如果参数 K 较小，则通过减枝技术就能避免大量计算统治集。总得来说，整个算法需要避免计算统治集概率。

5.4 实验测试与分析

通过上面的实验过程，我们可以注意到，整个算法的效率有着巨大的差距，对参数十分敏感，特别是阈值 p 的改变对整个程序有着重要的影响。但是，到底都是因为计算统治集概率，从而影响到算法的效率。

这样不仅说明，在整个算法的运行过程中，起着决定性作用的是计算统治集概率，还有计算统治集概率所在元组的深度，这就说明如果计算统治集概率，和整个算法的复杂度有着重要的关系。

5.4.1 TOP-K 质量分析

关于 Pt-k 算法的质量，可以这样理解，对于固定阈值，由于存在于 TOP-K 列表中的元组是唯一的，而且检索过程只需要扫描一次所有数据，对于正确率和准确率还是很高的，误差可以接受的范围。

TOP-K 质量和阈值 p 的选择十分有关系，当阈值选择比较大的时候，即使是靠前的元组也未必能出现在 TOP-K 列表中。

相反，如果阈值 p 比较小，那么，对于所有元组来说，都很有可能出现在 TOP-K 列表中，如果阈值 p 足够小，结果集是有可能包含所有元组的，因此，TOP-K 的质量会有所下降。

结论就是，在 Pt-k 算法中 TOP-K 的质量取决于阈值 p 的选取。

5.4.2 参数敏感性分析

根据上面几个表，可以看出，在同种情况下，Pt-k 查询算法对阈值 p 敏感度最高，从图 5-2 中数据量较大，阈值较大的情况，运行时间反而比数据量较小的元组运行时间要短很多，可以看得出是减枝技术发挥了重要作用。对于数据量的大小，反而敏感度没有那么明显，呈现的效果，没有阈值影响重要。

对于元组生成规则的变化，是最不敏感的，对于元组生成规则的个数和每个生成规则所包含的元组数目，可以看到是一种线性的增长。

Pt-k 算法对于阈值 p 的敏感度最高，从 5.4.1 节的分析也可以看出，TOP-K 查询的质量也是关于阈值 p 敏感，因此，阈值 p 的选取，对于查询的效率和查询的质量有着决定性作用。

5.4.3 数据处理能力分析

关于 Pt-k 算法对不确定性数据处理，可以发现有着明显的提高，当然这是和确定性数据是无法比拟的，只能通过本身的比较得出较为合理的分析。

对于实验的数据，无论不确定性数据的概率值是否具有一定的概率分布，或者足够离散，对于数据处理效率来说，没有太大的区别，因此，并没有将数据结果分类展示出来。

5.4.4 Pt-k 算法优化分析

我们从上面的数据分析，可以对三种不同的优化进行一个简单的比较。

- 1) 对于没有前缀共享优化来说，效率是比较低，虽然比原始的数据有了很大的提高，但是仍然具有缺陷，所以需要前缀共享；
- 2) 对于积极的前缀共享优化来说，效率比没有前缀共享有着巨大的提高，有着一种质的飞跃，但是，相对于校级共享前缀，还是有着一点差别，效率上还是具有一点劣势，就是对于数据重新排序十分积极，因此弱化了前缀共享的效率。通过上一章的讨论可以知道，统治集概率计算是最为消耗时间的计算，效率最低的部分，所以，只能强化前缀共享的作用；
- 3) 对于消极的前缀共享优化来说，效率有着明显的提高，主要是利用了前缀共享的优化作用，但是无可避免的是还是要对整个数据集进行至少一次的检索，没有充分利用定理 5 的作用；
- 4) 在实验过程中，通过观察结果集，可以发现，有些排名较后的元组是没有必要进行一个繁琐的统治集计算，需要完成的只是大部分靠前的检索，很大部分时间都浪费在靠后元组的检索过程中，而且输出的结果集往往大于参数 K ；
- 5) 如果是采用 $Pk\text{-}Topk$ 的算法，效率会有着更大的提高，因为它比 $Pt\text{-}k$ 算法更进一步，只需要输出 K 个元组的结果集就行了。不过，极端情况，如果结果集没有 K 个的话，数据的质量也会出现问题。因此，我们可以采取折中处理的办法，输出个元组完成整个查询，否则继续查找到最后一个元组。
- 6) 如果还是采用 $Pt\text{-}k$ 算法，可以进一步优化定理 5，优化减枝技术，减少靠后排名的元组检索时间。

5.5 实验结论

本实验得出的结论就是， $Pt\text{-}k$ 算法优化后，处理不确定性数据性能有着 70% 的提高，但是相对于较大数据集来说，提高的效率还不够。查询过程对于阈值 p 十分敏感，原因是程序大部分时间耗费在计算统治集概率上，只能通过减枝技术和共享前缀方法优化效率。

此算法，还可以进一步优化，提高效率，可以减少不确定性数据查询的时间消耗，减少不必要的统治集计算时间。

统治集概率的计算是比较消耗时间的，尽可能避免计算统治集概率，还有深度较高的元组的统治集概率，这样才能提高整个算法的效率。

总的来说，统治集概率避免了建立全部的可能世界，是一个质的飞跃，极大提高了算法的效率。但是，统治集概率的计算仍然十分消耗时间，如果能避免计算统治集概率，将会是另一次的质的飞跃。

5.6 本章小结

本章首先经过前面几章的分析，得出算法的伪代码，然后通过建立模拟数据和评估标准，验证算法的正确性和效率。

为了知道整个算法的参数敏感度，分别对阈值 p 、元组生成规则、参数 K 和元组数量进行一个简单的数据模拟实验，得出了结果是统治集概率的计算时间是影响整个算法运行时间的根本因素，其他因素都是在这个基础上发挥自己的作用。

同时，本章通过对一些参数约束后的数据进行一个简单的运行分析，实现了第四章的优化算法和进一步优化 $Pt-k$ 算法的输入数据，使其更加符合数据检索要求。通过对数据的列举和分析，得出了实验结论和后续优化的建议。

后面一章将是本文的结论和展望，本章也有研究不够深入的地方，例如扫描的元组数量，还有元组数据的特性分析，都没有深究。因此，继续延伸研究和发展 $Pt-k$ 算法是十分有必要的。

第六章 总结与展望

6.1 结论与总结

本文基于 Pt-k 算法的优化进行了一个实现，并对测试数据进行一个效率分析，得到的结果是，这种算法的优化能够有效提高 Pt-k 算法的效率，但是在更加大型的数据库，元组数量更加庞大的时候，显然这种算法也有其缺陷。Pt-k 算法即使优化后，还是要对排名较后的元组进行检索，不确定该元组的 TOP-K 概率是否超过阈值 p ，而计算过程消耗最大的就是计算其统治集概率，因此，对统治集概率计算优化是一个研究重点。

我们还可以延伸对 Pt-k 算法的研究，处理更多种不同类型的不确定性数据检索。举个例子，可以尝试解决不确定性数据查询中的 TOP-K Skyline 查询。

但是，根据上面实验的结果，我们可以发现 Pt-k 算法的效率有了明显的提高，但是仍然存在以下几个问题：

- 1) 对于数据的要求很高，处理所需要的时间仍然是需要计算的；
- 2) 如果原始数据发生变化，查询的效率仍然是十分巨大的；
- 3) 元组数量和效率是成反比，阈值和效率成正比，参数 K 和效率成正比，这个取决于计算统治集概率的时间，元组数量愈多，阈值发挥作用越少，参数 K 越小，则需要计算更多的统治集，直接降低了效率；

6.2 展望

通告前面几章的讨论，可以发现，确定性数据和不确定性数据有着巨大的区别，我们无法使用确定性数据的属性和方法解决不确定性数据问题，是直接导致新热点的关键。由于真实世界应用中数据固有的不确定性和数据不确定性的广泛存在，以及相关应用系统需求的增加，针对不确定数据管理的研究已经成为数据库技术研究中一个新的研究方向。

本文基于不确定性数据的查询处理研究基础上，对确定性数据中 TOP-K 算法在不确定性数据领域的一种算法，Pt-k 算法进行研究、算法改进、算法实现与效率测试，进一步延伸 TOP-K 算法在不确定性数据领域的应用。

与传统的查询处理相比，不确定数据上的数据模型引入了概率，结果集多以概率为衡量指标。为了减少查询代价，多种优化技术已经应用到不确定数据查询当中，比如剪枝策略和索引等。目前，不确定数据的管理、模型以及查询处理的研究受到了广泛关注，但是还有很多研究问题亟待解决，比如不确定数据建模、不确定数据整合、不确定数据管理的评价指标和不确定数据挖掘等。诸如上述问题是不确定性数据的研究重点，本文由于研究重点为查询，没有具体涉及上述方面，但是查询也是为解决上述问题的铺垫。

参考文献

中国期刊全文数据库

- [1] 崔斌, 卢阳. 基于不确定性数据的查询处理综述[J]. 计算机应用. 2008, 28(11): 2729-2731
- [2] 周傲英, 金澈清, 王国仁, 李建中. 不确定性数据管理技术研究综述[J]. 计算机学报, 2009, (01): 1-16
- [3] 李建中, 于戈, 周傲英. 不确定性数据管理的要求与挑战. 中国计算机学会通讯. 2009, 5(4): 6-15
- [4] 申德荣, 于戈, 寇月, 聂铁铮. 可能世界内数值型不确定性数据匹配模型. 计算机应用研究. 2008-7: 2607-2611

西文参考文献

- [5] WIDOMJ. Trio: a system for integrated management of data, accuracy, and lineage[C]. Proc of the 2nd Biennial Conference on Innovative Data Systems Research. 2005.
- [6] MOTRO A. Management of uncertainty in database systems [EB /OL]. (20052032 15). <http://ise.gmu.edu/~ami/research/publications/pdf/modern94.pdf>.
- [7] M. Hua, J. Pei, W. Zhang, and X. Lin. Efficiently answering probabilistic threshold TOP-K queries on uncertain data (extended abstract). In Proc. International Conference on Data Engineering (ICDE'08), Cancun, Mexico, April 2008.
- [8] R&C, Suciu D. Management of data with uncertainties Proceedings of the 16th ACM Conference on Information and Knowledge Management. Lisbon, 2007: 3-8
- [9] Dalvi N, Suciu D. Management of probabilistic data foundations and challenges Proceedings of the 26th ACM SIGMOD SIGACT SIGART Symposium on Principles of Database Systems. Beijing, 2007: 1-12
- [10] Aggarwal C C, Yu P S. A survey of uncertain data algorithms and applications. IBM Research Report, October 31, 2007
- [11] Pei J, Hua M, Tao Y F, Lin X M. Query answering techniques on uncertain and probabilistic data: Tutorial summary Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data. Vancouver, 2008: 1357-1364
- [12] Todd J. Green, Vat Tannen. Models for incomplete and probabilistic information. IEEE Data Engineering Bulletin, 2006, 29(1): 17-24
- [13] HUA M, PEI J, ZHANG W, et al. Ranking queries on uncertain data: A probabilistic threshold approach[C]. Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data. New York: ACM Press, 2008, : 673-686.
- [14] B R ZS NYI S, KOSSMANN D, STOCKER K. The skyline operator[C]. Proceedings of the 17th International Conference on Data Engineering. Washington, DC: IEEE Computer Society,

- 2001, :421-230 .
- [15] SILBERSTEIN A, BRAYNARD R, ELLIS C, et al. A sampling-based approach to optimizing TOP-K queries in sensor networks[C] .Proceedings of the 22nd International Conference on Data Engineering. Washington, DC: IEEE Computer Society, 2006, :68
- [16] CHENG R, PRABHAKAR S, KALASHNIKOV D V. Querying imprecise data in moving object environments[C] .Proceedings of the 19th International Conference on Data Engineering. Washington, DC: IEEE Computer Society, 2003, :723-725 .
- [17] CHENG R, XIA Y, PRABHAKAR S, et al. Efficient indexing methods for probabilistic threshold queries over uncertain data[C] .Proceedings of the 30th International Conference on Very Large Data Bases. San Francisco: Morgan Kaufmann, 2004, :876-887 .
- [18] PEI J, JIANG B, LIN X, et al. Probabilistic skylines on uncertain data[C] .Proceedings of the 33rd International Conference on Very Large Data Bases. New York: ACM Press, 2007, :15-26 .
- [19] Borzsonyi S, Kossmann D. and Stoeker K. The Skyline Operator[A]. In: Proc. of ICDE[C]. 2001, Pages 421-430.
- [20] Chan C. Y. , Eng P. K. , Tan K. L. Stratified Computation of Skylines with Partially-ordered Domains [A]. In: Proc. of the ACM SIGMOD[C]. 2005, :203-214.
- [21] Jin W. , Han J, Ester M. Mining Thick Skylines over Large Database[A]. In: Proc. of KDD[C]. 2004, : Pages 255-266.
- [22] Hristidis V, Koudas N, Papakonstantinou Y. PREFER: A System for the Efficient Execution of Multi-Parametric Ranked Queries [A]. In: Proc. of the ACM SIGMOD [CI], 2001, : Pages 259-270.
- [23] ABITEBOUL S, KANELAKIS P, GRAHNE G. On the representation and querying of sets of possible worlds[C] .Proceedings of the 1987 ACM SIGMOD International Conference on Management of Data. New York: ACM Press, 1987:34-48.
- [24] S. Abiteboul, P. Kanellakis, and G. Grahne. On the representation and querying of sets of possible worlds. In Proceedings of the 1987 ACM SIGMOD international conference on Management of data (SIGMOD'87), pages 34–48, New York, NY, USA, 1987. ACM Press.
- [25] Tomasz Imielinski and Jr. Witold Lipski. Incomplete information in relational databases. Journal of ACM, 31(4):761{791, 1984
- [26] D. Sarma, O. Benjelloun, A. Halevy, and J. Widom. Working models for uncertain data. In ICDE '06: Proceedings of the 22nd International Conference on Data Engineering (ICDE'06), page 7, Washington, DC, USA, 2006. IEEE Computer Society
- [27] M. A. Soliman, I. F. Ilyas, and K. C.-C. Chang. TOP-K query processing in uncertain databases. In Proceedings of the 23rd International Conference on Data Engineering (ICDE'07), Istanbul, Turkey, April 2007. IEEE.
- [28] L. Antova, C. Koch, and D. Olteanu. 10^{106} worlds and beyond: Efficient representation and processing of incomplete information. In Proceedings of the 2007 IEEE 23rd International Conference on Data

Engineering (ICDE'07), April 2007

- [29] Ronald Fagin, Amnon Lotem, and Moni Naor. Optimal aggregation algorithms for middleware. In PODS '01: Proceedings of the twentieth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems, pages 102-113. ACM Press, 2001.
- [30] D. Angluin and L. G. Valiant. Fast probabilistic algorithms for Hamiltonian circuits and matching's. In Proceedings of the ninth annual ACM symposium on Theory of computing (STOC'77), pages 30-41, New York, NY, USA, 1977. ACM Press.

统计资料

- [31] 2009 年中国互联网信息中心 (China Internet Network Information Center, CNNIC) 的调查报告: 中国互联网信息中心, 2009-1

参考网址

- [32] <http://queens.db.toronto.edu/project/conquer/>
- [33] <http://orion.cs.purdue.edu/>
- [34] <http://infolab.stanford.edu/trio/>
- [35] <http://www.cs.cornell.edu/database/maybms/>
- [36] <http://www.cs.washington.edu/homes/suciu/project-mystiq.html>
- [37] <http://www.eecs.berkeley.edu/Research/Projects/Data/102060.htm>
- [38] <http://www.cs.umd.edu/~vs/research.htm#pdb>
- [39] <http://www.almaden.ibm.com/cs/projects/avatar/>

致谢

通过三个多月的努力，我的专业毕业论文终于要如期脱稿付梓。本论文得以顺利完成，首先要感谢我的指导老师杜卿老师。杜卿老师在本文撰写过程中，至始至终都给予了悉心的指导，尤其是在论文的选题和拟定大纲时，更是得到了精心的点拨，通过阅览往届毕业生的论文，使我少走许多弯路，也使得资料搜寻和论文撰写工作能够迅速展开。对于论文的每一点细微错漏，她都进行了耐心的指点和纠正。杜卿老师严谨的治学精神和渊博的专业知识使我受益匪浅，在此向我的导师表示最真挚的感谢。

感谢华南理工大学软件学院为我提供了攻读软件工程专业的机会和良好的学习环境，感谢学院各位老师的辛勤培养和教育。老师们渊博的学识和无私的奉献精神，不仅让我从中学到了许多专业知识，还让我学到了许多为人处世的道理。

感谢共同学习和生活的同学和室友，跟他们一起度过了美好的学生生涯，他们也为本文的写作提供了不少好的建议。

还要感谢我的父母，是他们在物质和精神上给予我的支持和关怀，伴随着我走过了十六年的求学岁月，使我得以潜心学习和研究而顺利完成学业。

最后，感谢在百忙之中评阅本文和参加答辩的各位专家和老师！谢谢大家！