# 摘 要

随着信息技术的迅猛发展，“数据爆炸但知识贫乏”的现象越来越明显。在高校发展历程中，累积了大量的学生档案数据；如果应用数据挖掘方法将这些档案数据加以利用，对提升学校的教学质量大有裨益。图书馆图书借阅信息是大学生学习生涯中重要的数据组成部分，利用这些数据对读者进行书籍个性化推荐能够大大提高图书馆资源的利用率，帮助提升图书馆的服务水平和学生的知识素养。学生成绩是高校教学质量的重要评价指标，根据学生历史成绩数据，利用数据挖掘知识对学生成绩进行预测，及时发现成绩有可能不及格的学生并对其进行帮扶和指正，能够有效提高学校学生管理水平，为教学质量的提升保驾护航。

本文对学生档案数据的数据挖掘主要分为两个部分：一是应用学生图书借阅记录信息为学生进行书籍的个性化推荐；二是应用学生历史成绩数据对学生成绩进行预测。对于书籍推荐，本文主要应用ItemCF算法（基于物品的协同过滤算法）、关联规则的Apriori算法和FP-growth算法进行模型的建立，以学校2005年至今信息学院全体学生的借阅记录为数据样本进行模型的求解，最后得出ItemCF算法可行性和效果较好的结论。对于学生成绩预测，本文主要采用贝叶斯网络和基于成绩相似度的成绩预测方法对学生成绩进行预测，首先应用信息论中的条件互信息构建课程之间的相互关系，然后用课程开设顺序为上述关系指定方向以完成贝叶斯网络的构建，最后应用已知学生成绩的先验概率计算未知课程成绩的后验概率，贝叶斯网络能够得出诸如“已知某个学生的《数据结构》不及格，则其《操作系统》不及格的概率为41%”这样的结论。基于成绩相似性的成绩预测方法来自于K-means聚类思想，将拥有相似成绩的学生聚集在一起，应用其中学生的已知成绩去预测其他学生的同科目未知成绩。最后的结论是基于成绩相似性的成绩预测方法可行性和效果较贝叶斯网络好。

关键词：图书推荐； 成绩预测； 协同过滤； 关联规则； 贝叶斯网络； 相似度

# Abstract

With the rapid development of information technology, the phenomenon of “data explosion but poor knowledge” is becoming more and more obvious. A large number of student records data has been accumulated in the development of universities. It will improve the teaching quality of the school if the method of data mining is used to deal with these records. Borrowing information in library is an important part to students' learning life. Using these data for books personalized recommendation can greatly improve the utilization of library resources, the service level of the library and the student knowledge. Student achievement is an important evaluation of teaching quality in high education. Using data mining knowledge to predict students’ achievement can detect students whose achievement might fail. It can improve school management level and teaching quality.

This thesis is divided into two parts. In first part, we use borrowing information in library for books personalized recommendation. Another part we use data mining knowledge and the historical data of students to predict students’ achievement. In recommendation part, we mainly use the ItemCF algorithm (the collaborative filtering algorithm based on Item), Apriori algorithm and FP - growth algorithm to build a recommendation model. Then we use borrow records of all the students institute of information in the school since 2005 as the sample data for solving the model. Finally it is validated the feasibility of the ItemCF algorithm and its performance is better. For students’ performance prediction, this thesis uses Bayesian network and performance prediction method based on similarity scores of students. First of all, we use conditional mutual information in information theory to build the relationship between curriculum and label direction between curriculums in chronological order to complete the construction of a Bayesian network, and then to compute the posterior probability of the unknown course grade by prior probabilities of known course grade. It can be concluded from Bayesian network that the “if a student who fails to pass the data structure, it has about 41% chance that he will fail to pass the operating system”. The performance prediction method based on similarity scores comes from the K-means clustering. Students who have similar grades will be put together, and then the known scores are used to predict others unknown scores in the same subject. It is verified the feasibility of the performance prediction method based on the similarity scores and it is better than Bayesian network.

**Keywords:** Books recommendation; Achievement prediction; Collaborative filtering; Association rules; Bayesian networks; Similarity

# 目 录

[摘 要 III](#_Toc422749425)

[Abstract IV](#_Toc422749426)

[第1章 绪 论 1](#_Toc422749428)

[1.1 本论文的背景和意义 1](#_Toc422749429)

[1.2 本论文的主要方法和研究进展 3](#_Toc422749430)

[1.2.1 数据挖掘技术研究现状 3](#_Toc422749431)

[1.2.2 图书馆图书推荐研究现状 3](#_Toc422749432)

[1.2.3 学生成绩预测研究现状 4](#_Toc422749433)

[1.3 本论文的主要内容 5](#_Toc422749434)

[1.4 本论文的结构安排 5](#_Toc422749435)

[1.5 本章小结 6](#_Toc422749436)

[第2章 图书推荐相关算法 7](#_Toc422749437)

[2.1 协同过滤算法 7](#_Toc422749438)

[2.1.1 协同过滤基本概念 7](#_Toc422749439)

[2.1.2 基于物品的协同过滤 9](#_Toc422749440)

[2.2 关联规则挖掘算法 13](#_Toc422749441)

[2.2.1 关联规则概念 13](#_Toc422749442)

[2.2.2 Apriori算法 13](#_Toc422749443)

[2.2.3 FP-growth算法 15](#_Toc422749444)

[2.3 本章小结 18](#_Toc422749445)

[第3章 基于大数据的图书馆图书推荐 19](#_Toc422749446)

[3.1 研究背景与意义 19](#_Toc422749447)

[3.2 图书馆数据说明 20](#_Toc422749448)

[3.2.1 数据来源 20](#_Toc422749449)

[3.2.2 数据预处理 20](#_Toc422749450)

[3.3 图书推荐算法应用与结论 21](#_Toc422749451)

[3.3.1 基于物品的协同过滤书籍推荐 21](#_Toc422749452)

[3.3.2 基于Apriori关联算法的书籍推荐 25](#_Toc422749453)

[3.3.3 基于FP-growth关联算法的书籍推荐 25](#_Toc422749454)

[3.4 本章小结 27](#_Toc422749455)

[第4章 成绩预测相关算法 28](#_Toc422749456)

[4.1 贝叶斯网络 28](#_Toc422749457)

[4.1.1 贝叶斯分类概述 28](#_Toc422749458)

[4.1.2 贝叶斯分类的基础—贝叶斯定理 29](#_Toc422749459)

[4.1.3 贝叶斯网络 30](#_Toc422749460)

[4.2 K-means聚类算法 33](#_Toc422749461)

[4.2.1 K-means算法概述 33](#_Toc422749462)

[4.2.2 K-means算法距离测度 34](#_Toc422749463)

[4.2.3 点群中心点计算 35](#_Toc422749464)

[4.2.4 K-means算法变形 36](#_Toc422749465)

[4.3 本章小结 36](#_Toc422749466)

[第5章 基于大数据的学生成绩预测 38](#_Toc422749467)

[5.1 研究背景与意义 38](#_Toc422749468)

[5.2 学生成绩数据说明 38](#_Toc422749469)

[5.2.1数据采集 38](#_Toc422749470)

[5.2.2 数据提取与清洗 39](#_Toc422749471)

[5.3 学生成绩预测算法应用与结论 41](#_Toc422749472)

[5.3.1 学生成绩贝叶斯网络分析 41](#_Toc422749473)

[5.3.2 基于相似性的学生成绩预测 47](#_Toc422749474)

[5.4 本章小结 50](#_Toc422749475)

[结 论 51](#_Toc422749476)

[致 谢 52](#_Toc422749477)

[参考文献 53](#_Toc422749478)

# 第1章 绪 论

## 1.1 本论文的背景和意义

21世纪的今天，人们生活在一个被信息包围的时代。通信技术、计算机网络技术正在改变着人们的生活[1]，越来越庞大的信息资源累积是对人们数据处理和分析能力的挑战，从这些数据信息中抽取有价值的知识能够使这些数据得到更好的利用，改善人们的生活品质。高校在建设过程中累积起来越来越庞大学生档案信息便是大数据信息中的一部分，如何更好地应用这部分数据为教学、为学生服务是高校发展中应该切实考虑的一个问题。

如何有效地应用这些高速累积的数据，以及发掘数据背后隐藏的重要信息，需要利用更为有效的手段对各种数据源进行数据整理和知识挖掘，并通过分析以发现新的更为有用的信息，从而发挥这些数据的潜在价值。技术总是在永不满足的需求中诞生，数据挖掘技术（Data Mining，DM）就是其中的一位。这种技术能够从无边无际的信息海洋中提取出知识进而转换为社会价值。数据挖掘有时候也被称作数据库中的知识发现，是英文Knowledge Discovery in Database直译过来的词，简称KDD。自从1995年在加拿大召开的第一届知识发现和数据挖掘会议以来，数据挖掘已成为机器学习、数据库系统、人工智能等领域内热门的研究课题[2]。

学生成长档案数据挖掘提高了学生档案利用的水平，我国著名的档案学家吴宝康曾经说过，“档案必须利用，通过利用发挥其作用，为社会服务，并同社会的政治、经济、文化、科学、教育等方面紧密联系起来，实现档案的自身价值”。在校期间，学生成长档案更加突出了学生的个体特点，比传统人事档案更细致，虽然增加了辅导员的工作量，但是大大提高了档案的真实性、全面性和利用价值。将来求职时，成长档案对学生毕业推荐提供真实可信的资料，是记录大学生在校期间学习、生活的真实材料。传统档案存在重视历史、轻视才能的倾向。档案中政审和社会关系方面的材料多，反映培养成长过程的材料少；记录学习成绩格式化的东西多，而记录参加学习实践和考核的材料少；反映学生优点的材料多，反映缺点的材料少。在很大程度上不能反映学生发展全貌，不能为用人单位提供相关信息，因此无论从历史真实性方面，还是从利用价值方面都已经不再适用。学生成长档案能反映学生个性化发展特征，为学生就业、用人单位了解、选拔、任用、考核毕业生提供有效的证明。因此学生成长档案应该成为传统人事档案的新内容，新发展，而对学生成长档案的数据挖掘工作更能使得档案利用率的提高[3]。

随着信息技术的发展,探索新的学习方式、教学方法以及建设数字化的校园是教师义不容辞的责任[4]。合理、恰当、有效地运用信息技术,解决常规教学中的一些问题,是所有老师面对的问题。目前中小学里应用比较多的是对学生成绩的分析，但这个分析仅仅是对录入成绩的查询和一些常规统计，如求平均分、及格率、优秀率、最高分等。这些分析只是某一次考试的当前情况的体现，却无法发现数据中存在的关系和规则，无法根据现有的各项数据预测学生成绩以及学生今后发展趋势。作为学校，学生成绩评估是评价教学质量的重要依据，如果能从现有的成绩和学生的其他相关信息里找出对成绩影响较大的因素，并能对学生的发展做出一定的预测，这对改进学生的培养方式和加强学校的教学质量都会有很大的帮助。

目前国内图书馆只是应用了一些信息化技术代替了原来的人工劳动，例如书籍的信息管理、借书还书业务。应用图书馆数据进行数据分析与挖掘然后用来做决策或预测的情况还没有出现或者很少。在基于校园网络的图书管理系统应用中，有关读者和图书的信息己具备形成一个图书借阅数据仓库的条件。另一方面，随着采购规模的扩大，采购人员很难再像从前那样直接根据读者的人数和专业的分布找出诸如专业课程与图书的关系并据此进行决策。因此应用数据挖掘技术对图书馆数据包括但不仅限于读者借阅信息、书籍资料信息、不同读者的身份信息、专业信息、各个学科之间的关联信息等进行大规模的分析与应用对图书馆书籍资源配置和提高图书馆服务水平大有裨益。

最近几年，人们对“大数据”的关注越来越多。利用以数据为基础的决策来解决诸多领域问题成为大数据时代的显著特征。对于学校管理者，通过数据驱动决策系统可以查看年级、全校乃至整个学区的学生的学习情况。如果发现低效的课堂和学习表现不佳的学生群体（根据性别、收人情况等划分），管理者会依据学习分析结果决定是否给予特定的干预。对于更大范围的异常表现，管理者会依据学习分析数据调整管理策略，以适应教师更好的教和学生更好的学。美国的实践案例表明，数据驱动决策系统在帮助学区和学校领导利用数据制定一个用于持续改进的蓝图方面表现出其积极的作用。面对大数据时代的到来，我们的教育工作者如何利用大数据来服务教育，如何把这些大数据信息转化成知识，改革传统教育教学，这将是一个很大的挑战。我国教育大数据的现状是数据流失比较严重。作为学校的信息管理部门，网站的信息数据，学生学籍信息数据，学生每次考试的数据……可以说是扑面而来，但问题是基本上用过就废。虽然教师和学生的许多教育行为数据都可能构成大数据，但是目前我国学校较少采集这些数据，以及对数据进行建模和挖掘。目前学校比较关注的是考试数据，然而即使是考试数据，也没有得到充分利用。在学校考试长期只用考试总分作为甄别学生标准的情况下，客观上就不存在关注详实考试数据的需求。当前学校对校内考试工作的处理，一般都较少收集详实的考试数据，也较少对考试数据作统计分析。由于放弃了通过考试数据的统计关系去发掘考试和考生水平的相关信息，考试基本上也无法发挥对学生学业的诊断作用，教学就缺乏针对性。这种使考试对教学的反馈作用徘徊在低层次、低水平的状态，构成了制约学校教育质量提高的重要因素。因此，如何从学生和教师的教育行为中随时采集数据，如何借鉴国外经验，最大化地利用数据提升学校教育质量成了当务之急。

## 1.2 本论文的主要方法和研究进展

本文将研究课题细化到学生档案挖掘中的图书推荐和学生成绩预测。主要是对数据挖掘领域内相应算法的应用。

### 1.2.1 数据挖掘技术研究现状

数据挖掘技术起源于二十世纪八十年代末期的美国，1989年8月举行的第11届国际联合人工智能学术会议上，提出了KDD一词。从那以后，数据挖掘技术就在金融、保险、电信行业上得到了率先应用。1995年，数据挖掘已经成为国外的热门研究方向。自1997年开始，数据挖掘拥有了专门杂志《Knowledge Discovery and Data Mining》。目前数据挖掘蓬勃发展，研究重点逐渐发生偏移。主要表现在从理论研究转为系统应用。而且多个学科相互渗透也促进了这一领域的快速提高。

数据挖掘在国际研究中影响广泛，尤其是在数据库方向和信息决策领域，学术界和工业界都十分关注，工业界即将产生重要影响的五项关键技术中，KDD和人工智能排名第一，由此可以看出数据挖掘的重要地位。值得一提的是数据挖掘的研究不仅在学术领域受到重视，更是由于它能够给投资家和企业经营者带来巨大的经济效益，也引起了工业界的广泛关注。利益推动社会发展，国内外众多学者和机构纷纷投入数据挖掘的理论和应用研究的汹涌浪潮中，大批学术论文、著作以及商业应用成功案例不断涌现。某些级别比较高的工业研究实验室例如(ALMADEN和GTE)，以及众多的学术研究单位(例如UC BERKELEG，SAS研究所)某些著名的公司例如(AT&T，IN FORMIX，ORACLE，SYBASE，IBM)等，都在数据挖掘领域开展了各种各样的研究，推动数据挖掘的商业产品的高速发展。当前，世界上比较有影响的典型数据挖掘系统有Knowledge、EXPLORA、Story Cover、Quest、Miner DB、Workbench Discover等。

数据挖掘是在国外兴起的技术所以比较而言，国内的数据挖掘研究应用晚。但是在前人的基础上研究，发展速度相当快。国家的高度重视和许多科研单位以及众多高校的大力支持，说明国内的数据挖掘形势一片大好。

### 1.2.2 图书馆图书推荐研究现状

数据挖掘在图书馆中的应用研究是近七八年才掀起的，尤其是2006年以来，数据挖掘应用热潮的到来，图书馆相关数据挖掘的研究越来越多。而且数据挖掘信息应用技术作为计算机技术里的一种新兴技术，在国内众多学者的鼎力支持下形成一种前所未有的蓬勃局面，象征国内数据挖掘从理论到应用的春天即将来临。

数据挖掘技术在国外的很多图书馆中得到了非常成功地应用。这些图书馆除担任文献的收藏机构外还成了信息机构；身处其中的图书馆馆员摇身一变成了信息专家。荷兰的代夫特大学出版社就是这样的一个活生生的例子，他们早期想和国际水路历史协会创办一个新的期刊，与原来创办期刊的草率决策不同，他们这次充分应用图书馆数据进行了深入的挖掘分析并且形成报告，用来支持这次期刊是否创办的决策。成功的结果可想而知。

由于起步晚，情况不同，数据挖掘在国内发展慢，实现较少。目前国内关于数据挖掘理论研究较多，不过对于数据挖掘应用系统和算法测试的研究相对较少。对于数据挖掘在图书馆中应用和研究，大多停留在理论与设想上，实践很少。

### 1.2.3 学生成绩预测研究现状

在美国，学校收集数据已经几十年，然而许多学区或学校管理者直到最近才发现数据在学校改进方面的力量。近期大家对数据的关注是由“不让一个孩子掉队”(NCLB)法案所引发的。随着技术的最新进步和对学生学习评价的需求不断增加，许多学区或学校管理者发现数据的作用已远远超出了NCLB的报告要求。如今全国各地具有超前意识的学区开始采用数据驱动决策系统，不仅要分析测试分数和学生成就，还要分析如何缩小学生之间的成绩差距，提升教师素质，改进课程，分享学校和学区的最佳实践，与关键利益相关者更有效地沟通教育问题，促进家长在教育过程中的参与以及加强与教育团体的对话。无论人们是否同意立法的范围和意图，NCLB已将全国各地对教育数据的重视提升到了一个新水平[5]。

纽约市教育局是美国最大的教育系统，拥有超过100万的学生和8万名教师。长期以来，纽约市面临着一系列阻碍学生成绩提高的影响因素，例如学生总人数中有423694名是由公共资金援助支持的，153134名学生接受了特殊教育服务。其他影响因素还包括高度集中的贫穷、无家可归、种族隔离和药物成瘾、教师和学校领导的高流动频率以及文化断裂、缺乏技术支持等。2001年，纽约市教育局在卡耐基公司的资助下与“成长网络”(the Grow Network)公司签订了为3-8年级开发数据驱动决策工具的协议u41。该协议涉及到的教师有3万人，学区和学校的教学领导有5千人，以及1200所学校近40万的学生。这展示了纽约市教育局为了提高学校系统多层面的教育决策质量，在将标准化评价数据与支持性的教学资源相链接方面做出了前所未有的努力。

国内学生成绩预测主要针对考试科目变动不大的情况如高考等场景，对高校学生的成绩进行预测大多停留在理论与设想上，实践很少。可能的原因一是学校对学生成绩数据的保密要求较高，一般学者很难拿到相关的数据；二是学校对学生成绩在大数据上的应用不够重视，目前以存档为目的的较多。我相信随着信息的发展，学生档案数据大数据应用将会得到更好的发展。

## 1.3 本论文的主要内容

本文的主要内容分为两个部分：

第一部分为：图书馆图书推荐。本文基于实际图书馆图书内容获取比较困难并且图书馆图书分类比较完善的情况，主要讨论基于物品的协同过滤算法和关联规则挖掘算法在图书馆图书推荐中的应用情况。首先是对理论知识的讲解。协同过滤采用基于物品的协同过滤算法，本文应用简单实例对此算法进行演示使其简单易懂。关联规则挖掘是应用数据项之间的相关关系，探讨两个物品之间相互之间的支持度与置信度以用于物品的推荐；然后是图书馆数据的预处理，因为图书馆数据来源于关系型数据库，需要将学生ID与借阅过的书籍ID一一对应，为了研究与对比，将学生ID、书籍ID存放在txt文件中等待算法处理；最后将理论算法应用于实际数据上产生推荐并对结果进行简要分析。

第二部分为：学生成绩预测。学生成绩的预测有助于促进教师教学的针对性和提高学生预防挂科的警觉性，有助于教学的整体把握和教学质量的提高。本文基于实际学生成绩数据。首先是对理论知识的阐述，贝叶斯网络应用先验概率推导后验概率的特性能够在学生已知的成绩上面估计未知成绩取值的概率。基于相似性的成绩预测来自于K-means聚类的思想，将拥有相似成绩的学生聚集在一起，应用其中学生的已知成绩去预测其他学生的同门未知成绩；然后是数据的清洗与提取。和图书馆数据一样，学生成绩也来自于关系型数据库，首先需要对数据进行去规范化。对于贝叶斯网络而言需要将学生成绩二值化，本文将其划分为两类0表示不及格，1表示及格。基于成绩相似性的预测直接应用每个学生的成绩。最后将理论算法应用于实际数据上产生推荐并对结果进行简要分析。

本文的主要实现语言为Java。其中贝叶斯网络算法应用MATLAB进行算法仿真网络图，还应用Excel对学生预测成绩进行了可视分析。图书馆数据和学生成绩数据都应用SQL SERVER数据库进行存储和预处理。

本文对几种算法在处理实际问题上的可行性做了评估与对比。主要目的在于寻找应对学生档案数据挖掘实际问题时的可行方案。

## 1.4 本论文的结构安排

本文主要讲述了两个现实场景的应用问题；一个是图书推荐，另一个是成绩预测。为了使文章结构清晰增强文章逻辑的紧凑性，本文被划分成了五个章节讲述。

第一章 绪论。介绍本文研究背景和意义、主要方法和研究进展、主要内容和论文的组织安排。

第二章 图书推荐相关理论。这一章主要是对推荐算法的理论进行介绍，主要是基于物品的协同过滤、关联规则挖掘中的Apriori算法和FP-growth算法的理论介绍。

第三章 图书馆图书推荐。将图书推荐相关理论结合西南交通大学图书馆05年至14年信息学院所有借阅记录信息，通过数据收集与预处理、建立算法处理模型，从模型得到数据挖掘的结果，形成为学生推荐书籍的可行方案。

第四章 成绩预测相关理论。介绍贝叶斯网络和K-means算法的理论知识，对于K-means算法，本文主要借鉴其相似度计算方法。

第五章 学生成绩预测。将成绩预测相关理论结合西南交通大学06年至14年软件工程专业的学生成绩，通过学生成绩清洗与提取处理、分别建立贝叶斯网络模型和学生成绩相似度模型，从模型中得到学生成绩预测的结构，最后对贝叶斯网络和相似性成绩预测进行分析与评估。

最后对论文进行总结。

## 1.5 本章小结

本章首先介绍了本文的研究背景和意义，介绍国内外图书馆图书推荐情况和学生成绩预测发展情况。

然后介绍本文的两部分主要内容，一是图书馆图书推荐；二是学生成绩预测。

最后给出了本文的结构安排。

# 第2章 图书推荐相关算法

技术的出现是为了解决生活中遇到的问题进而提高人类的生活品质，这么说来有应用的地方就会有技术的高速发展。推荐系统领域的研究正是如此。20多年过去诞生了各种各样解决不同场景下不同问题的推荐算法，比如按数据划分有协同过滤、内容过滤、社会化过滤等；按算法划分有基于领域、基于图、基于矩阵分解或者概率模型的算法等。当然这些算法各有应用场景。本文基于实际图书馆图书内容获取困难且图书分类信息比较完善的情况，主要对基于物品的协同过滤算法和关联规则挖掘算法在图书馆图书推荐中的应用情况进行分析。

## 2.1 协同过滤算法

### 2.1.1 协同过滤基本概念

协同过滤[6]是1992年Goldberg提出的概率，是当今推荐系统中应用最为频繁的推荐思想，当前主要分为基于用户的协同过滤（也称UserCF）和基于物品的协同过滤（也称ItemCF[8]）两大类。

UserCF主要有两个步骤；第一是找到和目标用户兴趣相投的用户，主要通过用户的相关记录来计算，在电商行业就是购买记录；第二步从哪些兴趣相投的用户相关的记录信息中为目标用户推荐他原来从未尝试过的记录。

ItemCF的推荐思路和UserCF相仿：首先是计算各个物品之间的相似度，然后将与用户相关记录相似度较高的物品推荐给目标用户。

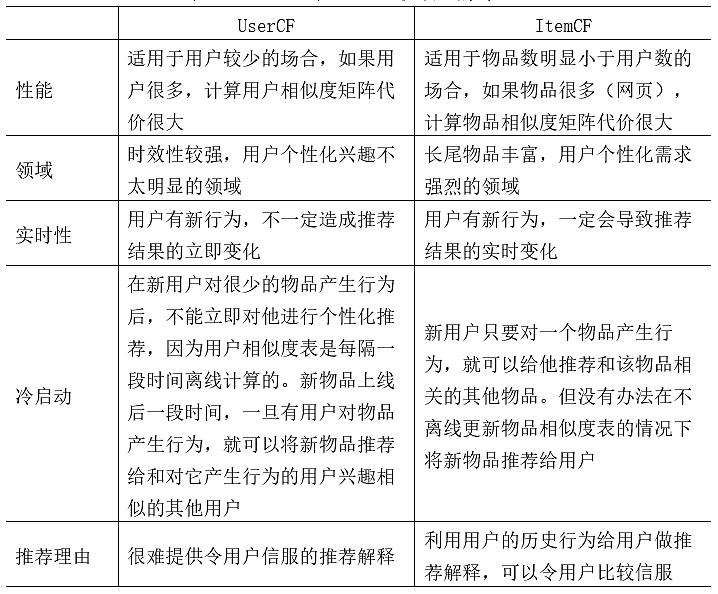
综上所述，UserCF是基于用户的相似性进行推荐的而ItemCF是基于物品相似度来进行推荐的，这两种推荐都用到了用户与相关物品的联系。虽然如此这两个算法的着重点还是不太一样，前者反应的事群体热点效应而后期主要关注的是用户的历史兴趣。这么一说，UserCF的推荐是社会化的推荐而ItemCF的推荐主要是个性化的推荐；UserCF映射的是小群体对物品的热度，而ItemCF映射的是个人兴趣；所以UserCF一般用在新闻类网站中，如Digg；而ItemCF则用在其他非新闻类网站中，如Amazon等网站。

UserCF适合用于新闻推荐的另一个原因是从技术角度考虑的。因为作为一种物品，新闻的更新非常快，每时每刻都有新内容出现，而ItemCF需要维护一张物品相关度的表，如果物品更新很快，那么这张表也需要很快更新，这在技术上很难实现。绝大多数物品相关表都只能做到一天一次更新，这在新闻领域是不可以接受的。UserCF的推荐建立在用户的相似性上面，所以说系统新加了用户就需要更新用户的相似性表，对于新闻网站而言用户的数量相对于每分每秒更新的新闻来说数量基本保持不变。对于新用户来说给他推荐最热门的新闻是最好的，因为看新闻的目的在于获取最新消息而不是最好的知识。因此UserCF显然是利大于弊。

与上述情况有对比意义的是在图书推荐、商品推荐和电影推荐方面。这种情况下ItemCF优势明显；首先这些内容的推荐用户的兴趣点相对固定，比如计算机专业的人士一般而言主要阅读的就是计算机相关的书籍，有的人就是喜欢科幻片而不喜欢偶像剧。此外，这些方面的推荐上用户更关心的是自己的兴趣，在个性化上面的要求更高并且可以更具以往的经验性知识知道什么符合自己的口味。而这是ItemCF擅长的方面。因此，ItemCF算法成为了这些网站的首选算法。更能加强说服力的是这些物品的更新速度都不是很快，例如书籍的增长速度绝对远远小于新闻的更新速度。ItemCF中一天一次物品相似度的更新是可以接受的并且更加合理。

表2-1从不同的角度对比了UserCF和ItemCF算法。

表2-1 UserCF和ItemCF优缺点对比[7]



值得注意的是，UserCF算法或ItemCF算法的选择不能只把关注点放在性能上；首先应该满足产品的需求，比如如果需要提供推荐解释，那么可能得选择ItemCF算法。其次，需要看实现代价，若用户太多，很难计算用户相似度矩阵，这个时候可能不得不抛弃UserCF算法。

综上所述，在图书馆藏书量相对固定的环境下，针对图书馆用户的个性化分析与推荐问题，本文拟采用ItemCF算法作为推荐算法加以应用。下面就是对ItemCF的单独阐述。

### 2.1.2 基于物品的协同过滤

基于物品的系统过滤算法在协同过滤算法中地位比较高，上面已经说过ItemCF关注的是用户的个性化需求，而现实生活中能够产生个性化需求的地方确实要比社会化的地方多很多。ItemCF算法来源于亚马逊，众所周知亚马逊是一家电子商务公司，为用户推荐商品的强烈愿望催使它在2003年提出了个性化推荐度的ItemCF算法，该算法一经提出就得到业界的广泛认可，并且它给父公司亚马逊带来了巨大的经济收益。

ItemCF主要针对用户的喜好来进行推荐，比如你喜欢《天龙八部》那么推荐《射雕英雄传》给你一定是合理的。值得注意的是，和基于内容的推荐不一样，物品的相似度计算不是来自物品的内容，而是来源于用户的行为。例如购买的啤酒的人也购买了尿布，则啤酒喝尿布的相似性就增加1，虽然在实际中他们毫不相关。这就是推荐的神奇之处。

在上节中已经提到ItemCF主要分为两步。

首先是计算物品之间的相似度，经典公式就是(2-1)。

|  |  |  |
| --- | --- | --- |
|  |  | (2-1) |

式中 —表示的事喜欢物品的用户数；

—表示的是同时喜欢物品和物品的用户数。

这个公式看起来非常符合思维逻辑，但是却没有考虑到热门商品的情况。这里假设很热门，就会导致就会非常大，接近1。这会导致热度高的商品越来越高即长尾效应。为了避免这个情况的发生可以应用(2-2)公式，通过惩罚物品的权重来削弱热门物品的影响力。

|  |  |  |
| --- | --- | --- |
|  |  | (2-2) |

ItemCF计算物品的相似性可以借助一种特殊的结构就是共现矩阵，这是ItemCF算法推荐中一个重要的特殊技巧。下面是详细代码。

|  |
| --- |
| def ItemSimilarity(train):  #calculate co-rated users between items  C = dict()  N = dict()  for u, items in train.items():  for i in users:  N[i] += 1  for j in users:  if i == j:  continue  C[i][j] += 1  #calculate finial similarity matrix W  W = dict()  for i,related\_items in C.items():  for j, cij in related\_items.items():  W[u][v] = cij / math.sqrt(N[i] \* N[j])  return W |

图2-1是一个根据上面的程序计算物品相似度的简单例子。

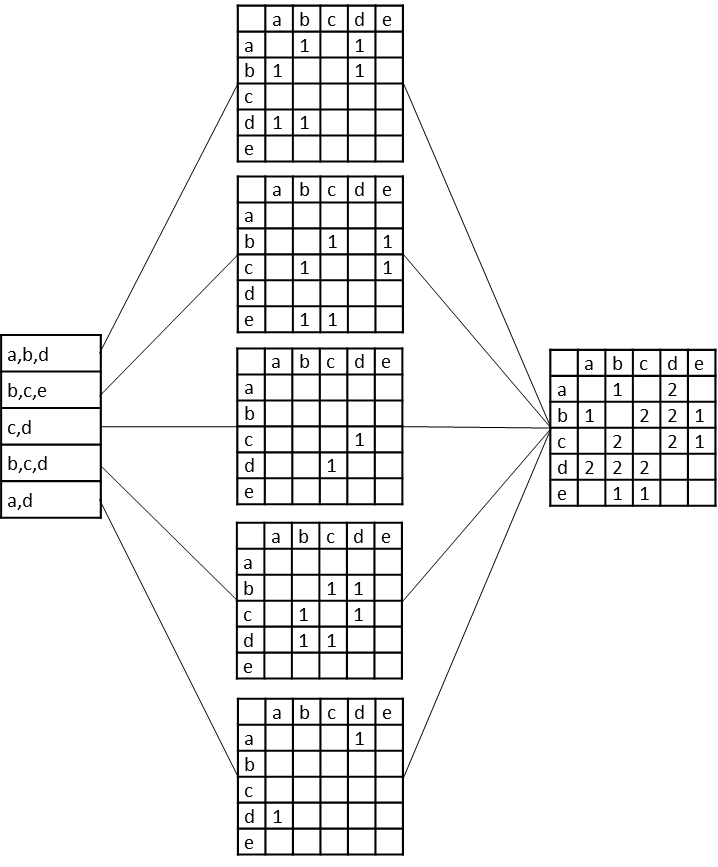


图2-1计算物品相似度的简单示例

图中最左边表示的是用户对商品产生的行为记录，比如说第一条表示用户1购买了*a*、*b*和*d*三个物品。中间的5个矩阵就是针对每个用户建立的相似度矩阵，将同时出现在单个用户记录中的物品增量置为1记录在共现矩阵中。最后一列是将每个用户的共现矩阵进行汇总得到的最终每个物品的共现矩阵。

接下来就是根据物品的相似度和用户的历史行为给用户推荐物品；得到物品的相似度后可以根据(2-3)计算用户对物品的兴趣。

|  |  |  |
| --- | --- | --- |
|  |  | (2-3) |

式中 *N*(*u*)—用户喜欢的物品集合；

—物品*j*最相似的*K*个物品集合；

—物品*j*和*i*的相似度;

—用户*u*对物品*i*的兴趣。

这个公式的实现代码如下。

|  |
| --- |
| def ItemRecommendation(itemTrain, user\_id, W, K):  itemRank = dict()  ru = itemTrain [user\_id]  for i,pi in ru.items():  for j, wj in sorted(W[i].items(), /  key=itemgetter(1), reverse=True)[0:K]:  if j in ru:  continue  itemRank [j] += pi \* wj  return itemRank |

图2-2表示的就是上面综述的一个过程。



图2-2 一个简单的基于物品推荐的例子

一般来说，每个网站都会有一个UI系统，UI系统负责给用户展示网页并和用户交互。网站会通过日志系统将用户在UI上的各种各样的行为记录到用户行为日志中。日志可能存储在内存缓存里，也可能存储在数据库中，也可能存储在文件系统中。将这些数据应用推荐算法加以处理生成推荐列表，最后通过界面展示给用户共用户参考，整个过程如图2-3所示。



图2-3 推荐系统与其它系统之间的关系

可以从相似度计算和预测值计算方向改进整个推荐过程。

Item-based算法计算相似度的方法有以下几种：

1、余弦相似度计算，应用两个向量之间夹角的余弦值表示物品之间的方向相似度，公式如下：

|  |  |  |
| --- | --- | --- |
|  |  | (2-4) |

2、关联相似度计算，应用两个向量之间的Pearson-r关联度表示物品之间的方向相似度，公式如下：

|  |  |  |
| --- | --- | --- |
|  |  | (2-5) |

是用户*u*对物品*i*的打分，是第*i*个物品打分的平均值。

3、调整的余弦相似度计算，在余弦的相似度的基础上减去用户打分的平均值,这样就可以消除不同用户打分习惯造成的不良影响，公式如下：

|  |  |  |
| --- | --- | --- |
|  |  | (2-6) |

是用户*u*打分的平均值。

根据之前算好的物品之间的相似度，下面就可以对用户为打分的物品预测，可以应用及加权求和，另一个可以用回归。

首先是加权求和见(2-7)式。

|  |  |  |
| --- | --- | --- |
|  |  | (2-7) |

表示的是*i*和*N*之间的相似度；是用户*u*对物品*N*的打分。

接下来是回归：回归和加权求和的方法十分相似，但回归要比加权求和复杂很多。回归的关键点在于不直接使用相似物品的打分值，它认为不同用户可能因为打分习惯的不同导致本该相似的物品相差较远。它的值是应用(2-8)重新估算得出的。

|  |  |  |
| --- | --- | --- |
|  |  | (2-8) |

这里的*N*是*i*的相似物品，通过对*N*和*i*的打分向量线性回归得到和，是回归模型的误差。

## 2.2 关联规则挖掘算法

### 2.2.1 关联规则概念

在数据挖掘中存在一个十分重要的算法那就是关联规则[9]。从字面意义上来说关联规则就是提取数据项之间的关联关系。可以用一个通俗的问题来说明关联规则要解决的问题，就是如果一个人购买了萝卜，那么这个人还要购买大白菜的几率有多大？或者这个人购买了萝卜和大白菜那么他还要购买西红柿的几率？这是个现实问题，应用关联规则算法就能很好地回答这个问题。从上述例子还可以知道关联规则其实对顺序没有要求而只考虑组合形式。要理解关联规则，下面的相关概念也十分重要。

第一个就是项集，它是是*m*个有特征属性的集合。事务*T*是项集中某些元素组成的集合。事务数据库*D*是所有事务*T*的集合。关联规则中有两个十分重要的指标就是支持度和置信度。诸如的蕴含式，其中*X，YI*并且*X*∩*Y*=就所*X*到*Y*的一条关联规则。其中支持度*s*定义为support(*XY*)=P(*X*∪*Y*)；置信度*c*定义为confidence(*XY*)=P(*Y*|*X*)。可以看出支持度就是两个物品一起出现的概率。置信度就是在购买了一个物品的条件下购买另一件物品的概率。关联规则的目标就是在设定的最小支持度和置信度的情况下挖掘出物品之间的关联关系，然后用这些关联关系来做推荐或者决策[10]。

关联规则算法中最有代表性的事Apriori算法和FP-growth算法。其中后一个是对第一个的改进。

### 2.2.2 Apriori算法

上面提到，Apriori算法是一个关联规则挖掘算法[12]。它的核心思想是通过候选集寻找频繁项集。它使用逐层搜索迭代的方法先找出所有频繁1-项集*L*1，然后用*L*1寻找*L*2，用*L*2寻找*L*3直到找不到频繁项集为止。它是有由Rakesh Agrawal于1994年首次提出的。Apriori算法有两个非常重要的性质是Apriori算法成立的基础[13]。第一个性质是如果*k*维数据项目集*X*是频繁项目集的必要条件是它的所有*k-*1维子集均是频繁项目集。第二个性质是*k*维项目集*X*中有*k-*1维子集不是频繁项集则*X*不是频繁项集。利用这两条性质，我们在使用频繁(*k-*1)*-*项集*L*(*k-*1)寻找频繁*k-*项集*L*(*k*)时分两个过程：连接步和剪枝步。

连接步：*L*(*k-*1)与其自身进行连接，产生候选项集*C*(*k*)。*L*(*k-*1)中某个元素与其中另一个元素可以执行连接操作的前提是它们中有(*k-*2)个项是相同的，也就是只有一个项是不同的。如：项集{*I*1,*I*2}与{*I*1,*I*5}连接之后产生的项集是{*I*1,*I*2,*I*5}，而项集{*I*1,*I*2}与{*I*3,*I*4}不能进行连接操作。

剪枝步：候选集*C*(*k*)中的元素可以是频繁项集，也可以不是。但所有的频繁k-项集一定包含在*C*(*k*)中，所以，*C*(*k*)是*L*(*k*)的超集。扫描事物集*D*，计算*C*(*k*)中每个候选项出现的次数，出现次数大于等于最小支持度与事务集*D*中事务总数乘积的项集便是频繁项集（这里的最小支持度指的是概率，实际中我们经常直接将最小支持度看成乘积后的结果），如此便可确定频繁*k-*项集*L*(*k*)了。

但是，由于*C*(*k*)很大，所以计算量也会很大。为此，需要进行剪枝。即压缩*C*(*k*)，删除其中肯定不是频繁项集的元素，依据就是前面提到的Apriori性质：如果一个子集是非频繁的，那么它的超集也一定是非频繁的。这在Apriori算法里面很重要。如此，如果一个候选(*k-*1)*-*子集不在*L*(*k-*1)中，那么该候选*k-*项集也不可能是频繁的，可以直接从*C*(*k*)中删除。

图2-4是Apriori算法的演示示例。

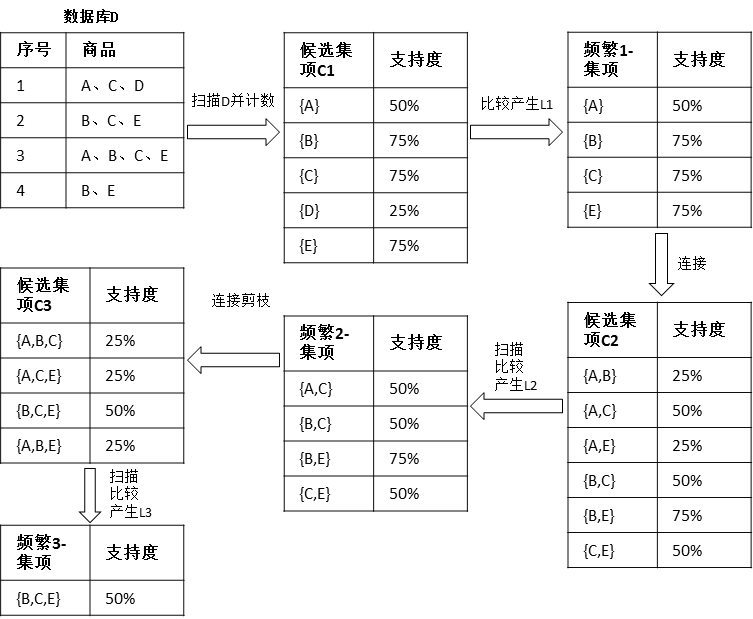


图2-4 Apriori算法演示示例

所以，满足条件的所有频繁项集有{*A*}、{*B*}、{*C*}、{*E*}、{*A*、*C*}、{*B*、*C*}、{*B*、*E*}、{*C*、*E*}、{*B*、*C*、*E*}。

算法伪代码如下所示。

|  |
| --- |
| C[k]: 长度为 k的候选项集  L[k]: 长度为k的频繁项集  L[1] = {频繁项};  for (k = 1; L[k] !=∅; k++) do begin  C[k+1] = 由 L[k]产生的候选;  for each 数据库中的事务t do  增加包含在t 中的所有候选C[k+1]的计数  L[k+1] = C[k+1]中满足 min\_support的候选  end  return L[1..k]; |

Apriori算法的优点：简单、易理解、数据要求低。

Apriori算法的缺点主要还是在运行时间上面，原因在于两个，第一个是每一步产生候选集时循环产生的组合太多，没有对不应该参与组合的元素进行排除；第二个是每次计算项集支持度的时候都需要对数据库中的全部记录进行扫描比较，这样带来的计算机系统的I/O开销随着数据记录的增加成几何增长导致算法性能较低。

### 2.2.3 FP-growth算法

FP-growth算法是韩家炜等在2000年发表的《Mining Frequent Patterns without Candidate Generation: A Frequent-Pattern Tree Approach》里提出的关联分析算法。它的核心是FP-Tree的构建，这也是该算法比Apriori算法性能上能够显著提高的一个根本上的原因。FP-Tree是一个树状结构，它通过合并一些重复路径实现了数据的极大压缩，通过对FP-Tree的遍历替代Apriori对数据库所有事物的遍历来提高效率。算法运行示例如下[14]。

第一步：读取事务数据库，然后构建频繁1-项集和FP-树，这是FP-growth算法的基础。

这里假设事务数据库中有5条数据如表2-2第一列所示并且假定最小支持度为阈值是3。

构建FP-Tree需要对数据库扫描两遍。第一遍的目的在于频繁1-项集并且排序。在扫描第一遍的时候就要提取频繁项集。{(*f*:4),(*c*:4),(*a*:3),(*b*:3),(*m*:3),(*p*:3)}就是第一遍扫描应该产生的结果。数据中各项冒号之后的数字是对应项出现的频次，一般说来程序给定最小支持度是个小数，但是因为对于同一个实例来说分母相同所以可以直接用一个数值来比较大小避免不必要的麻烦。上述结果是排好序的值得注意的是这个排序非常重要因为之后的计算中每个数据库事务都会按照这个排序，这也是有效合并重复路径的前提。处理之后的数据库记录为表2-3第二列。第二次扫描数据库是建立在第一遍扫描产生的结果之上的，用于产生FP-Tree。

表2-2 示例事务数据库

|  |  |  |
| --- | --- | --- |
| TID | Items bought | (Ordered)frequent items |
| 1 | *f a c d g i m p* | *f c a m p* |
| 2 | *a b c f l m o* | *f c a b m* |
| 3 | *b f h j o* | *f b* |
| 4 | *b c k s p* | *c b p* |
| 5 | *a f c e l p m n* | *f c a m p* |

1. 取出第一条事务数据，构建FP-Tree的第一条路径，{*f c a m p*}。注意其中项的排序与第一步中得到的频繁项集合的排序是一致的。
2. 取出第二条事务数据，{*f c a b m*}，不难发现，它与第一条路径共享了部分数据{*f c a*}。因此，可以重复利用已有的路径，只需要将其计数加1，即{(*f*:2),(*c*:2),(*a*:2)}。而对于后面不同的部分，我们创建新的路径，{(*b*:1),(*m*:1)}，其中，*b*为*a*的子节点，*m*为*b*的子节点。
3. 取出第三条事务数据，{*f b*}，发现没有重复路径存在。但*f*点是存在的，因此，可以重复利用*f*点，新建一个*b*节点，作为*f*的子节点，得到路径{{*f*:3},(*b*:1)}。注意，之前已经存在的*b*节点无法重复使用，因为其父节点为*a*。
4. 取出第四条事务数据，{*c b p*}，发现没有重复路径存在。因此，从现有*c*点出发，构建一条新路径{(*c*:1),(*b*:1),(*p*:1)}。
5. 取出第五条事务数据，{*f c a m p*}，同上原理构建路径，{(*f*:4),(*c*:4),(*a*:3),(*m*:2),(*p*:2)}。

经过两遍数据库扫描，完成了FP-Tree的构建。在此例中，可以看到这里的根节点并不是唯一的，即有多棵子树。为了方便树结构的遍历，可以人为添加一个超级根节点，通常标记为 root<null>。参照图2-5，可以更清楚的理解整个过程。

第二步：遍历FP-tree的头表，对于每个频繁项*x*，累积项*x*的所有前缀路径形成*x*的条件模式库CPB。

第三步：对CPB上每一条路径的节点更新计数为*x*的计数，根据CPB构造条件FP-树。

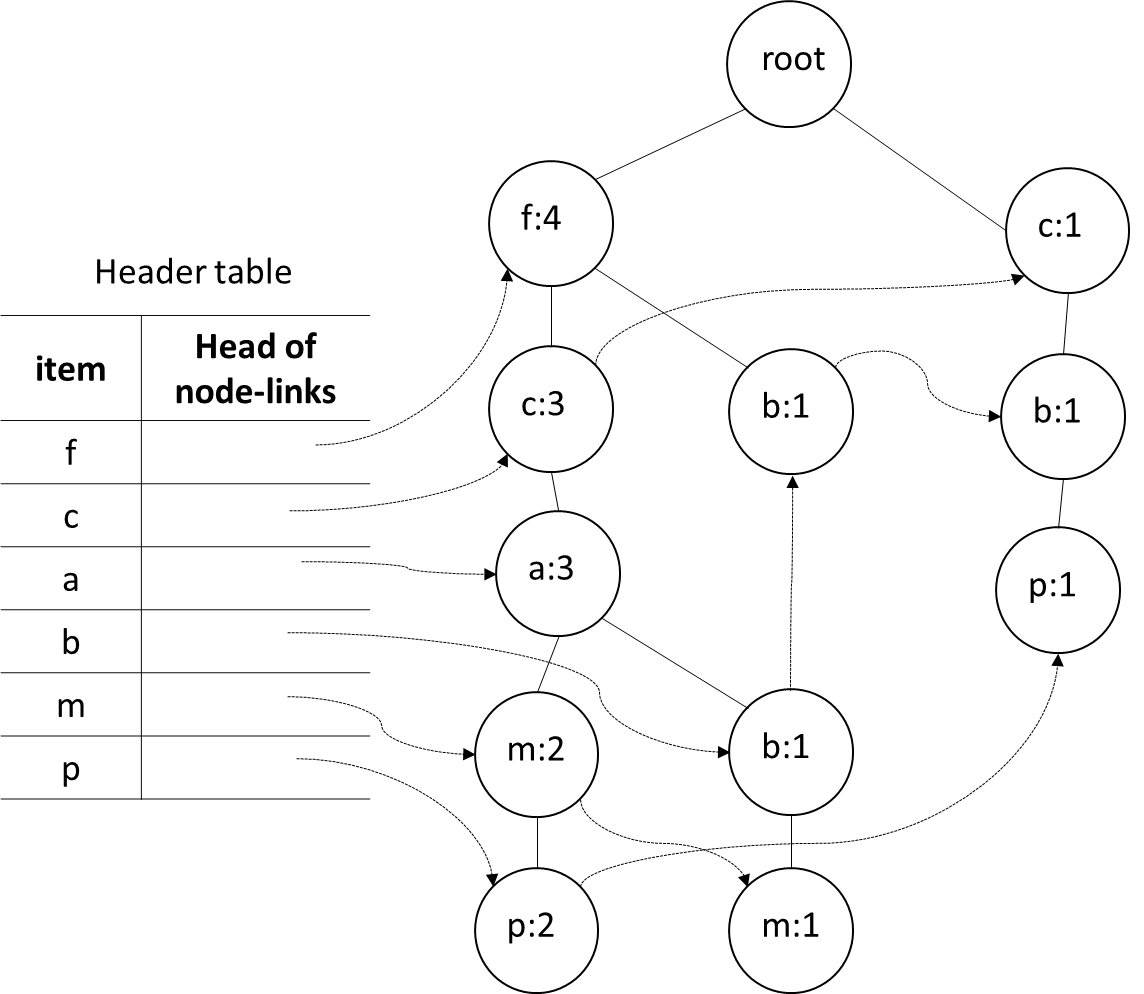


图2-5 FP-树

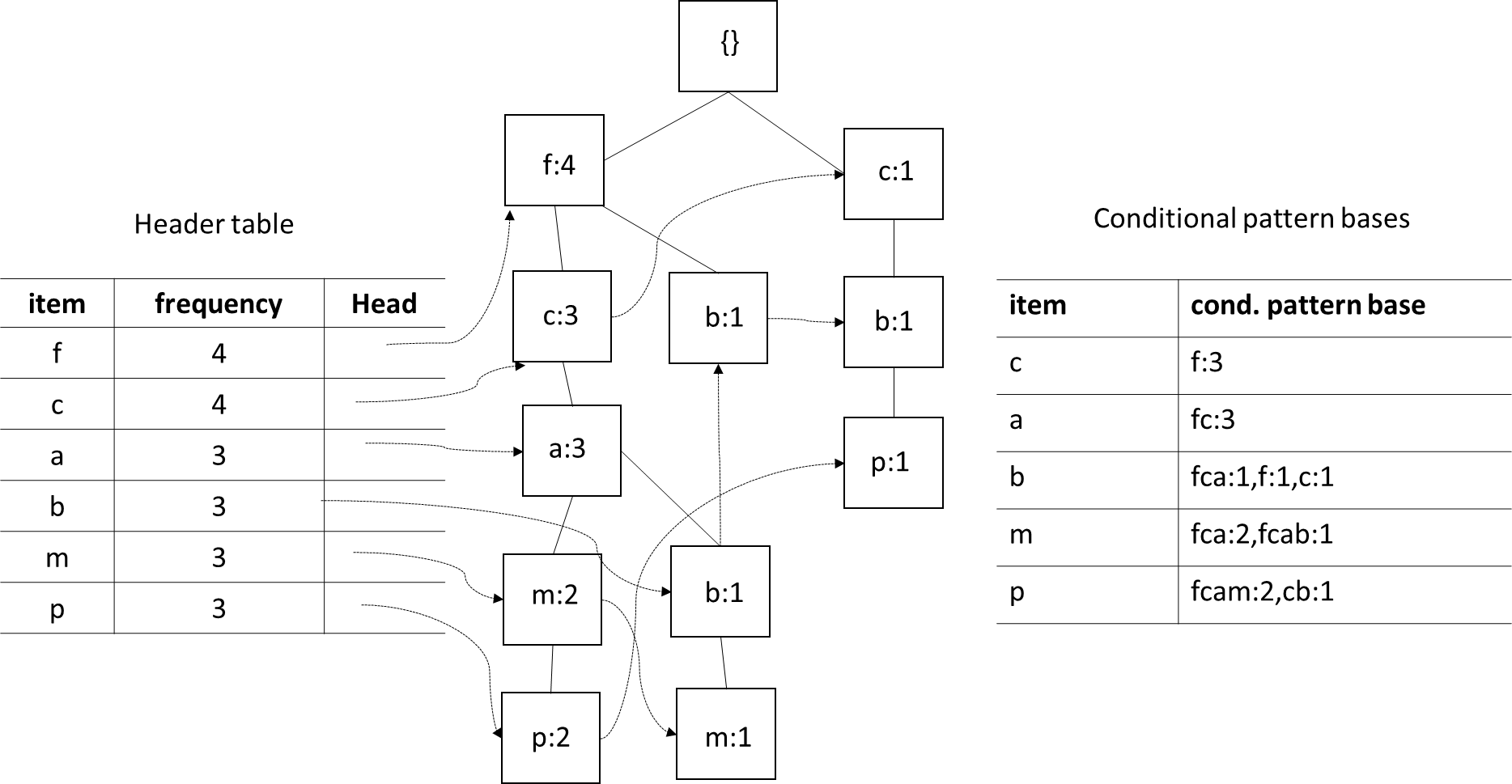


图2-6 生成条件模式库CPB

表2-3 通过条件子模式库挖掘频繁集

|  |  |  |
| --- | --- | --- |
| Item | Conditional pattern-base | Conditional FP-three |
| *p* | {(*fcam*:2),(*cb*:1)} | {(*c*:3)}|*p* |
| *m* | {(*fca*:2),(*fcab*:1)} | {(*f*:3,*c*:3,*a*:3)}|*m* |
| *b* | {(*fca*:1),(*f*:1),(*c*:1)} | ∅ |
| *a* | {(*fc*:3)} | {(*f*:3,*c*:3)}|*a* |
| *f* | ∅ | ∅ |

第四步：从条件FP-树中找到所有长路径，对该路径上的节点找出所有组合方式，然后合并计数。

第五步：将Step4中的频繁项集与*x*合并，得到包含*x*的频繁项集。

重复第二步到第五步，直到遍历头表中的所有项。

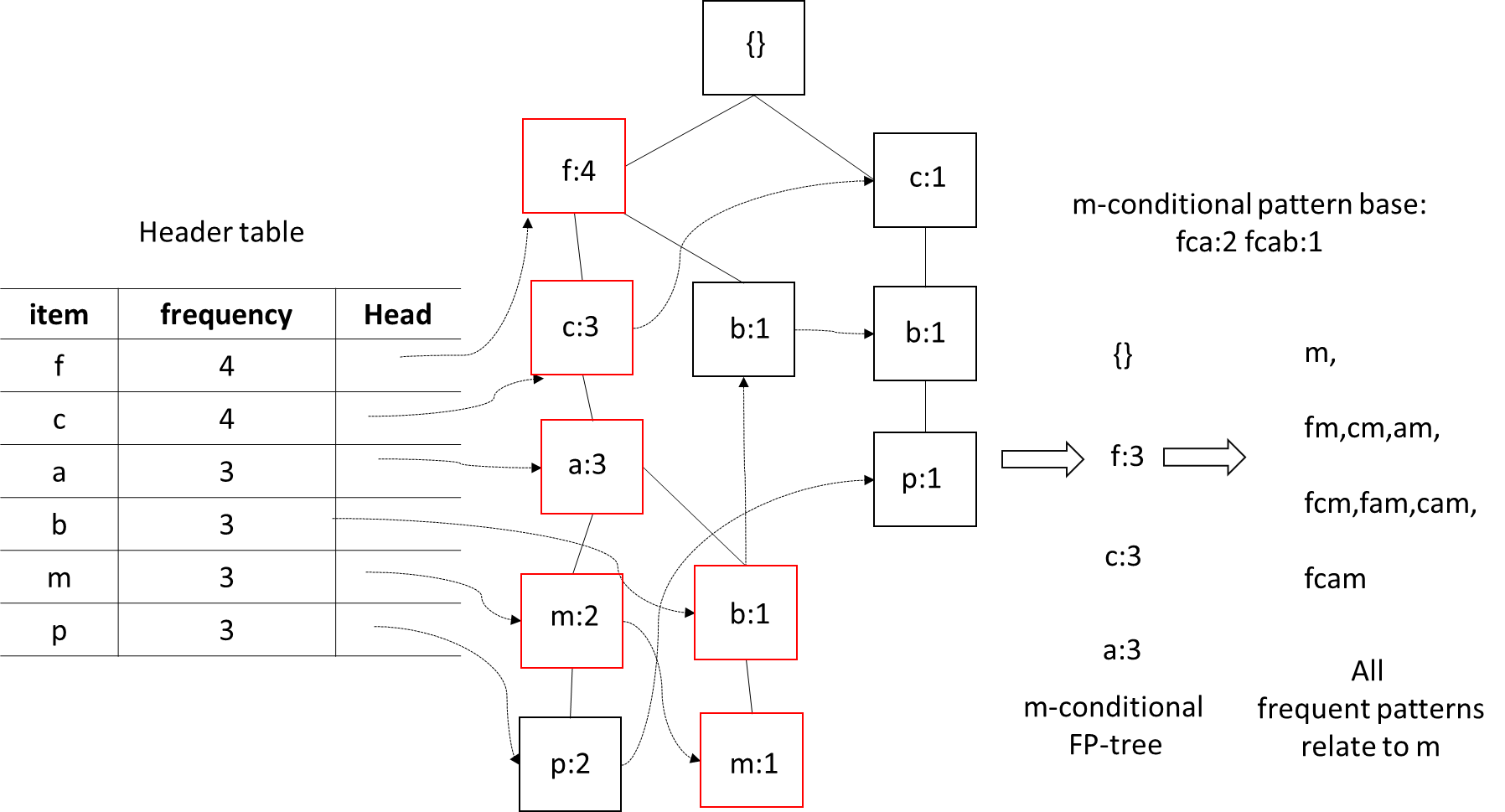


图2-7 生成*m*的频繁集的过程

## 2.3 本章小结

本章对推荐系统中的两大类算法即协同过滤算法和关联规则挖掘算法进行理论学习和研究；具体包括协同过滤算法中的基于物品的协同过滤算法以及关联规则的Apriori算法和FP-growth算法；其中关联规则的算法复杂程度要比协同过滤算法的复杂程度高一些。下一章将会论证它们各自在图书馆图书推荐中的实际应用。