分类号_____ 密 级_____

U D C _____

# 学 位 论 文

## 面向搜索经验的查询推荐方法研究


作 者 姓 名 ： 刘大力

指 导 教 师 ： 张斌 教授

东北大学计算机科学与工程学院



申请学位级别： 硕士 学 科 类 别 ： 工学

学科专业名称： 计算机应用技术

论文提交日期： 2017 年 10 月 论文答辩日期： 2017 年 12 月

学位授予日期： 2017 年 12 月 答辩委员会主席：

评 阅 人 ：

东 北 大 学

2017 年 10 月

**A Thesis in Computer Software and Theory**

# Study on Search Experience Oriented Query Recommendation Method

By Liu Dali

Supervisor: Professor Zhang Bin

**Northeastern University**

**December 2017**

# 独创性声明

本人声明，所呈交的学位论文是在导师的指导下完成的。论文中取得的研究成果除加以标注和致谢的地方外，不包含其他人已经发表或撰写过的研究成果，也不包括本人为获得其他学位而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示谢意。

学位论文作者签名：

日　　期：

# 学位论文版权使用授权书

本学位论文作者和指导教师完全了解东北大学有关保留、使用学位论文的规定：即学校有权保留并向国家有关部门或机构送交论文的复印件和磁盘，允许论文被查阅和借阅。本人同意东北大学可以将学位论文的全部或部分内容编入有关数据库进行检索、交流。

作者和导师同意网上交流的时间为作者获得学位后：

半年 □　　一年□　　一年半□　　两年□

学位论文作者签名：　　导师签名：

签字日期：　　签字日期：

# 摘 要

复杂搜索描述了用户在进行搜索过程中的几种典型场景：系统对信息的索引不够充分；搜索任务本身需要浏览及探索；用户对搜索任务领域的知识匮乏以致难以对查询进行组织或定位信息领域。在复杂搜索中，用户的搜索过程常常具有时间跨度长，搜索内容复杂、范围广泛的特点。在复杂搜索中，目前已有的传统查询推荐方法并不能很好地利用复杂搜索的特点，针对用户的复杂搜索行为进行查询推荐，因此本文研究如何提出一种充分利用复杂搜索特性，针对复杂搜索的查询推荐方法。

在此前研究中，作者所在室验室提出了时间树理论帮助用户管理复杂搜索过程，时间树将用户提交的查询词和点击的搜索结果按照时间顺序以树的形式组织起来，形成了对整个复杂搜索过程的可视化描述，为用户记录与管理复杂搜索过程提供了一个有效的手段。时间树作为一种复杂搜索过程管理的理论工具，能够记录用户的复杂搜索过程，本研究利用时间树对用户复杂搜索过程的记录这一事实，以时间树中蕴含搜索经验这一假设为出发点，研究面向搜索经验的查询推荐方法。为了提出面向搜索经验的查询推荐方法，首先需要从时间树中提取出搜索经验，因此本研究需要提出基于时间树的搜索经验提取方法，为了提出基于时间树的搜索经验提取方法，首先需要验证时间树中蕴含用户进行复杂搜索过程中的高质量的搜索经验。本文基于这一思路，进行面向搜索经验的查询推荐方法研究。

首先，本研究提出搜索经验模型以及搜索经验一致性模型，对用户进行复杂搜索过程中的搜索经验以及搜索经验一致性进行了模型化的定义。在此基础上，本研究设计实验，从主观评估、专家评估以及客观评估三个角度对时间树中搜索经验的蕴含性进行了系统性的验证。继而提出基于时间树的搜索经验提取方法，分别针对搜索经验模型中的因果经验以及主题经验，提出基于查询-点击-查询序列识别的因果经验提取算法以及基于子任务划分的主题经验提取算法，并对基于子任务划分的主题经验提取算法有效性进行了实验验证。在提出基于时间树的搜索经验提取方法后，本研究最终提出了面向搜索经验的查询推荐以及查询推荐可视化方法，并设计实验验证了面向搜索经验的查询推荐方法的有效性。最后，本研究设计并实现了面向搜索经验的查询推荐系统，使本研究形成了一套完整的解决方案。

**关键词：**复杂搜索；查询推荐；时间树；搜索经验

# Abstract

Complex search describes several typical scenes during users' search process: the users lack the knowledge or contextual awareness to formulate queries or navigate complex information spaces, the search task requires browsing and exploration, or the system indexing of available information is inadequate. In complex search, the search process of the user always has features of long span of time, complex and broad searching content. In previous studies, the laboratory team in which the author studied proposed the theory of TimeTree to help users manage the complex search process. TimeTree organizes the queries submitted by users and the searching results clicked by users into a form of tree in order of time, forming a visual description of the whole complex search process, providing an effective way for users to record and manage complex search process.

As a theoretical tool for management of complex search process, TimeTree can record users' complex search process. Taking advantage of the fact that TimeTree can be used for managing users' complex search process, this study discusses the search experience oriented query recommendation method. To propose the search experience oriented query recommendation method, the search experience extracting method is needed as a basis. To propose the search experience extracting method, it is necessary to prove that TimeTree contains search experience of users in complex search process. Based on this idea, this study does the research of search experience oriented query recommendation method.

First of all, this study proposes the search experience model and the search experience coherence model, which gives search experience and search experience coherence a modeled definition. Based on the search experience model and the search experience coherence model, this study designs experiment to make a systematic verification of the implication of search experience in TimeTree through 3 ways of subjective evaluation, expert evaluation and objective evaluation. After that, this study proposes the search experience extracting method, which contains the algorithm of causal experience extraction based on query-click-query sequence recognition and the algorithm of thematic experience extraction based on subtask partition aiming at causal experience and thematic experience in the search experience model respectively, and carries out experimental verification of the algorithm of thematic experience extraction based on subtask partition. Then this study proposes the search experience oriented

query recommendation method and query recommendation visualization method, and carries out experimental verification of the search experience oriented query recommendation method. At last, a search experience oriented query recommendation system is designed and implemented to make this study a complete set of solutions.

**Key words:** complex search; query recommendation; TimeTree; search experience

# 目 录

# 第1章 引 言

本章阐述本研究的研究背景以及研究意义，阐明面向搜索经验的查询推荐方法在复杂搜索领域研究中的价值，并对本研究的主要研究内容进行说明，对本研究中作者所进行的主要工作进行概括性的描述。

## 1.1 研究背景与意义

复杂搜索描述了用户在进行搜索过程中的几种典型场景：系统对信息的索引不够充分；搜索任务本身需要浏览及探索；用户对搜索任务领域的知识匮乏以至难以对查询进行组织或定位信息领域[1]。在复杂搜索中，用户通常会根据自己的现有知识或已有的线索提交一些试探性的查询，从返回的查询结果中学习新知识和发现新线索，进而决定下一步的搜索方向，从而逐步逼近搜索目标，最终形成从搜索起点到最终目标的完整而复杂的搜索过程[2]。在这一过程中，如何有效地为用户提供查询推荐，以便缩短用户的探索过程，是复杂搜索研究中的一个重要问题

为帮助进行复杂搜索的用户在三种典型场景下进行复杂搜索过程管理，在此前的研究中，作者所在室验室提出了时间树理论来记录和管理用户的复杂搜索过程。时间树将用户提交的查询词和点击的搜索结果按照时间顺序以树的形式组织起来，形成了对整个复杂搜索过程的可视化描述，为用户记录与管理复杂搜索过程提供了一个有效的手段。与此同时，通过记录查询与点击之间的关联如点击所属的查询及查询的起源，时间树完整地记录了用户解决搜索问题的过程，直接地体现了用户的搜索经验。因此我们认为，我们可以充分利用时间树这样一种可视化的搜索经验的沉淀，进行搜索经验的提取，进而找到可复用的搜索经验，并以可视化的形式呈现给用户，从而使用户避免不必要的搜索尝试，尽快逼近搜索目标，缩短用户的搜索过程。

## 1.2 主要研究内容

本文研究的问题是面向搜索经验的查询推荐方法。研究以时间树及搜索经验的定义模型为基础，首先验证时间树中蕴含用户进行复杂搜索过程中产生的高质量的搜索经验，并在此基础上提出时间树上的搜索经验提取方法，进而提出面向搜索经验的查询推荐及查询推荐可视化方法，并最终设计查询推荐系统，以使本研究形成一套面向搜索经验的完整查询推荐解决方案。

本研究的研究框架如图1.1所示。

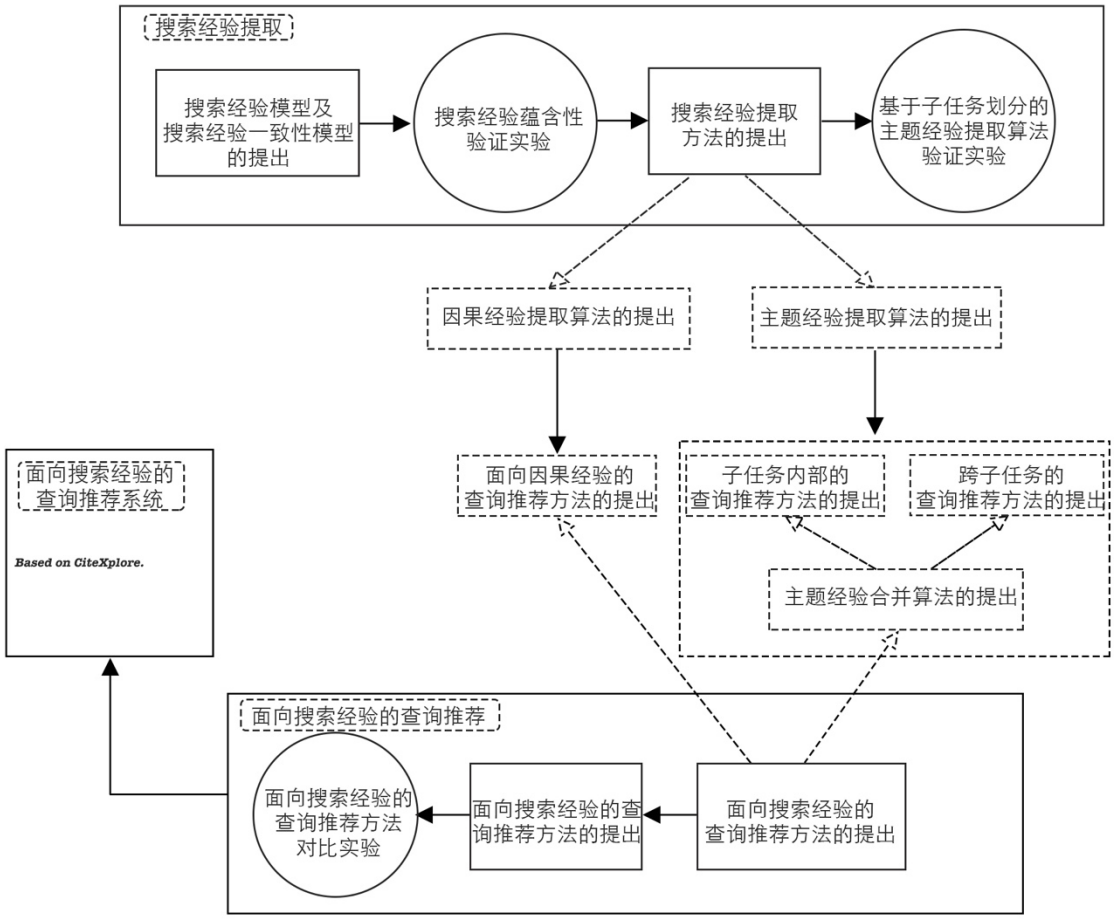


图 1.1 面向搜索经验的查询推荐方法研究研究框架

Fig. 1.1 Research framework of study on search experience oriented query recommendation method

为研究面向搜索经验的查询推荐方法，本研究首先提出搜索经验模型以及搜索经验一致性模型，对搜索经验以及搜索经验一致性给出模型化的定义，并基于此，设计实验验证时间树中搜索经验的蕴含性。在证明了时间树中蕴含高质量的搜索经验后，本研究提出了搜索经验提取方法，该方法分别针对时间树中蕴含的时间经验以及主题经验设计了因果经验提取算法以及主题经验提取算法。之后本研究设计实验验证了基于子任务划分的主题经验提取算法的有效性。在提出搜索经验提取算法后，本研究分别针对因果经验以及主题经验，提出了面向因果经验的查询推荐方法、子任务内部的查询推荐方法以及跨子任务的查询推荐方法和与这些方法相对应的查询推荐可视化方法。并设计实验，验证面向搜索经验的查询推荐方法的有效性。最后，本研究基于 CiteXplore 系统设计面向搜索经验的查询推荐系统，使本研究形成一套完整的面向搜索经验的查询推荐的完整解决方案。

(1) 搜索经验蕴含性验证及搜索经验提取方法提出

为了对时间树中搜索经验的蕴含性进行验证，本部分研究首先提出了搜索经验模型及搜索经验一致性，将用户在进行复杂搜索过程中的搜索经验划分为时间经验、因果经验以及主题经验，与之对应，将搜索经验一致性分为时间一致性、因果一致性与主题一致性。以搜索经验一致性模型为依据设计实验，分别从用户分析中经典的主观评估、专家评估以及客观评估三个方面对实验参与者在进行复杂搜索任务过程中，时间树对参与者的时间一致性、因果一致性以及主题一致性维护质量进行分析，验证时间树中搜索经验的蕴含性。以时间树中的搜索经验蕴含性实验结果为基础，本部分研究进而提出时间树中的搜索经验提取方法，分别针对用户在使用时间树进行复杂搜索过程中产生的因果经验以及主题经验提出搜索经验提取方法，并设计实验验证该方法的有效性。

(2) 面向搜索经验的查询推荐及查询推荐可视化方法提出

搜索经验提取方法作为从时间树中提取出用户复杂搜索的搜索经验的工具，为提出面向搜索经验的查询推荐奠定了基础。因此，本研究在此部分分别面向搜索经验提取方法提取出的因果经验以及主题经验提出查询推荐方法，其中面向因果经验的查询推荐方法基于查询-点击-查询序列识别，面向主题经验的查询推荐方法分为子任务内部的查询推荐方法以及跨子任务的查询推荐方法，基于主题经验合并算法以及节点 Rank 值计算实现。本部分研究设计实验验证面向搜索经验的查询推荐方法的有效性，并且针对该方法给出的多样性推荐结果分别提出面向因果经验的查询推荐可视化方法、子任务内部的查询推荐可视化方法以及跨子任务的查询推荐可视化方法。

(3) 面向搜索经验的查询推荐系统设计

在提出搜索经验提取方法以及面向搜索经验的查询推荐方法的基础上，为使本研究理论形成一套完整的解决方案，本部分设计面向搜索经验的查询推荐系统，以实现本研究理论的工程价值。在此前研究中，作者所在实验室曾开发一款复杂搜索管理系统 CiteXplore，用于时间树理论的研究。本部分研究基于 CiteXplore 系统进行二次设计开发，沿用 CiteXplore 系统的分布式架构，采用高可用的设计思路，将本研究提出的理论与 CiteXplore 系统结合，实现面向搜索经验的查询推荐系统的设计。

## 1.3 论文组织结构

本文的主要工作是对用户的搜索经验以及搜索经验一致性进行模型化的定义，并设计实验验证时间树中蕴含有用户的高质量的搜索经验。在此基础上提出搜索经验的提取方法，并面向提取出的搜索经验提出查询推荐方法以及查询推荐可视化方法。最后设计面向搜索经验的查询推荐系统，使本研究形成一套完整的解决方案。基于本文的主要研究内容，本文分为五章，具体组织结构如下：

第 1 章主要阐述本研究的研究背景以及研究意义，阐明面向搜索经验的查询推荐方法在复杂搜索领域研究中的价值，并对本研究的主要研究内容进行说明，对本研究中作者所进行的主要工作进行概括性的描述。

第 2 章介绍本研究的研究基础。本研究的研究基础主要有：复杂搜索任务与复杂搜索、查询推荐、层次聚类方法以及 PageRank 算法。

第 3 章首先对时间树的 RCST 结构、用户的搜索经验以及搜索经验一致性进行模型化的定义。在此基础上设计实验，对时间树中搜索经验的蕴含性进行验证。在验证了时间树中蕴含高质量的搜索经验后，提出搜索经验提取方法，并设计实验验证该方法的有效性。

第 4 章提出面向搜索经验的查询推荐方法以及查询推荐可视化方法，并设计实验验证查询推荐方法的有效性。

第 5 章以前两章提出的搜索经验提取方法以及查询推荐方法为基础，设计并实现面向搜索经验的查询推荐系统，使本研究形成一套完整的解决方案。

# 第2章 相关研究

本章阐述本研究的研究基础。本研究的研究基础主要包括搜复杂搜索任务与复杂搜素、查询推荐、层次聚类方法以及 PageRank 算法。

## 2.1 复杂搜索

搜索任务表示针对某一种特定信息需求的实现目标或出发点[3]。例如“搜索贾斯汀比伯的生日”就是一个搜索任务，显然，在搜索框中输入“贾斯汀比伯的生日”，任何一种搜索引擎中都能够在搜索结果列表最顶部给用户返回正确的答案。这显然不是一个复杂的任务，用户不需要进行复杂的搜索行为。与前例相反，复杂搜索是一种需要进行多次查询，查看多篇查询结果文档，从多种信息源种提取编辑信息的多步耗时的过程[4]。例如要搜索估算阿富汗境内的安全区域的相关信息或者开放性的任务例如寻找莫扎特与巴赫曲风间的区别。复杂搜索任务被定义为能够导致用户需要进行复杂搜索行为的相应任务[5-6]。

当前被广泛使用的搜索引擎例如 Google、Bing 和 Baidu 所使用的交互模式仍然主要是 1989 年 Bates 所提出的一问一答的模式。然而，随着互联网上信息数量的迅速增长以及人们对信息搜索系统的依赖性的增强，传统的一问一答的信息搜索模式，因其不能全面反应用户与搜索引擎交互的过程，忽略了搜索过程中用户信息需求的变化，而受到了非常大挑战。因此，2006 年，Marchionini G[7]提出了探索式搜索模型这一概念，之后，复杂搜索的概念在 2012 年被进一步提出[8]。文献[9]对于复杂搜索模式与传统的一问一答的模式进行了比较。相比于传统的一问一答模式，复杂搜索作为一种基于发现的模式，更加注重用户在搜索过程中的人机交互。复杂搜索结合了信息检索领域和人机交互领域的研究成果，使得搜索过程中加入了更多的用户自身的影响[10-11]，从而能够启发搜索过程。复杂搜索具有以下的基本特征：不熟悉目标领域，不清楚搜索目标，不确定达到目标的路径[12-14]。因此，复杂搜索主要适用于以下情况：用户对于其所要搜索的目标领域不熟悉，使得用户对于搜索结果的正确与否无法判断；用户无法确定其所要搜索的目标；用户不清楚如何开展搜索[15]。因此，与普通的搜索相比，复杂搜索涵盖了更加广泛的活动，比如，调查、评估、比较、综合等等[16]。

## 2.2 查询推荐

查询推荐（Query Recommendation）是针对用户输入的初始查询，根据不同的策略返回给用户与初始查询或者关键词相关的查询或者关键词，推荐的查询能够更好地表述

用户的真实意图，帮助用户更好地构建查询[17]，从而完成搜索任务。用户输入的查询在大部分时候并不能精确的表达其搜索意图。其原因主要有两个：一是某些情况下用户并不知道自己想要搜索的具体内容是什么而仅仅有一个模糊的目标，此时用户构造的查询反而增大了其意图的不确定性；二是用户知道自己具体要搜索的内容，但是用户构造出准确清晰地表达出其意图的查询仍然是困难的[18]。有研究文献[19]表明，在普通搜索中，只有 25%的查询能清晰的表达用户的意图。因此，在更加复杂的复杂搜索中，构造好的查询对于用户来说是具有很大难度的。

传统的查询推荐方法根据所利用的数据源的不同可以分为基于文档的查询推荐与基于查询日志的查询推荐两类[20-21]。其中基于文档的查询推荐分为全局文档集分析、局部文档集分析以及基于人工语料编辑的方法三种。基于日志的查询推荐分为基于 Session 的方法、基于点击 URL 的方法、基于文本相似度的方法以及基于时间分布的方法四种。

随着复杂搜索研究的展开，已经有越来越多的研究人员加入复杂搜索领域研究行列里，同时有很多可行性较强的研究方法被提出。传统查询推荐的两个主要的信息来源是文本集和搜索日志，但都具有一定的局限性[22]。如何针对复杂搜索这一特定搜索类型进行查询推荐，目前的研究尚有很大空间。

## 2.3 层次聚类方法

层次聚类方法为通过数据集按照某种策略进行层次的分解，直到满足某种条件为止[23]。按照分类原理的不同，可以分为凝聚和分裂两种方式[24]。其中凝聚层次聚类是一种自底向上的策略，首先将每个对象作为一个簇，然后合并这些原子簇为越来越大的簇，直到所有的对象都在一个簇中，或者某个终结条件被满足，绝大多数层次聚类方法属于这一类，它们只是在簇间相似度的定义上有所不同[25-26]。分裂的层次聚类与凝聚的层次聚类相反，采用自顶向下的策略，它首先将所有对象置于同一个簇中，然后逐渐细分为越来越小的簇，直到每个对象自成一簇，或者达到了某个终止条件[27-28]。

给定要聚类的 N 个对象以及 N * N 的距离矩阵(或者是相似性矩阵), 层次式聚类方法的基本步骤如下[29-31]：

(1) 将每个对象归为一类，共得到 N 类，每类仅包含一个对象。类与类之间的距离就是它们所包含的对象之间的距离。

(2) 找到最接近的两个类并合并成一类，于是总的类数少了一个。

(3) 重新计算新的类与所有旧类之间的距离。

(4) 重复第 2 步和第 3 步，直到最后合并成一个类为止（此类包含了 N 个对象）。

根据类间相似度计算方法不同，层次聚类可以被分为 SL 层次聚类、CL 层次聚类以

及 AL 层次聚类[32]。

(1) SL 层次聚类

该方法取两个类中距离最近的两个样本的距离作为这两个集合的距离，即最近的两个样本之间距离越小，这两个类之间的相似度就越大。该方法容易造成链条效应，两个簇明明从大局上离得比较远，但是由于其中个别点距离比较近却被合并了，并且合并后链条效应的影响会进一步被扩大，最终得到一个比较松散的聚类结果。

(2) CL 层次聚类

该方法与 SL 层次聚类相反，取两个集合中距离最远的两个点的距离作为两个集合的距离。其负面效果也与 SL 层次聚类刚好相反，两个簇即使已经很接近了，但由于个别点的距离非常远，而使这两个簇不能被合并。

(3) AL 层次聚类

SL 层次聚类与 CL 层次聚类的共同问题是只考虑了某一组有特点的数据，而没有考虑类内数据的整体特点。AL 层次聚类的基本思路即为考虑类内数据的整体特点，讲两个集合中的点两两距离求平均值，作为两个类的距离。

## 2.4 PageRank 算法

PageRank 算法为 Larry Page 与 Sergey Brin 在 1997 年构建早期的搜索系统原型时提出的链接分析算法，自从 Google 在商业上获得空前的成功后，该算法也成为其他搜索引擎和学术界十分关注的计算模型[33]。目前很多重要的链接分析算法都是在 PageRank 算法基础上衍生出来的。PageRank 是 Google 用于用来标识网页的等级以及重要性的一种方法[34]，是 Google 用来衡量一个网站的好坏的唯一标准。在揉合了诸如 Title 标识和 Keywords 标识等所有其它因素之后，Google 通过 PageRank 来调整结果，使那些更具等级以及重要性的网页在搜索结果中另网站排名获得提升，从而提高搜索结果的相关性和质量[35-36]。其级别从 0 到 10 级，10 级为满分。PR 值越高说明该网页越受欢迎（越重要）。例如：一个 PR 值为 1 的网站表明这个网站不太具有流行度，而 PR 值为 7 到 10 则表明这个网站非常受欢迎或者极其重要[37]。一般 PR 值达到 4，就算是一个不错的网站了。Google 把自己的网站的 PR 值定到 10，这说明 Google 这个网站是非常受欢迎的，也可以说这个网站非常重要。

在 PageRank 提出之前，已经有研究者提出利用网页的入链数量来进行链接分析计算，这种入链方法假设一个网页的入链越多，则该网页越重要[38]。早期的很多搜索引擎也采纳了入链数量作为链接分析方法，对于搜索引擎效果提升也有较明显的效果。PageRank 除了考虑到入链数量的影响，还参考了网页质量因素，两者相结合获得了更好

的网页重要性评价标准[39]。

对于某个互联网网页 A 来说，该网页 PageRank 的计算基于以下两个基本假设[40]：

(1) 数量假设：在 Web 图模型中，如果一个页面节点接收到的其他网页指向的入链数量越多，那么这个页面越重要。

(2) 质量假设：指向页面 A 的入链质量不同，质量高的页面会通过链接向其他页面传递更多的权重。所以越是质量高的页面指向页面 A，则页面 A 越重要。

利用以上两个假设，PageRank 算法刚开始赋予每个网页相同的重要性得分，通过迭代递归计算来更新每个页面节点的 PageRank 得分，直到得分稳定为止[41]。PageRank 计算得出的结果是网页的重要性评价，这和用户输入的查询是没有任何关系的，即算法是主题无关的。假设有一个搜索引擎，其相似度计算函数不考虑内容相似因素，完全采用 PageRank 来进行排序。这个搜索引擎对于任意不同的查询请求，返回的结果都是相同的，即返回 PageRank 值最高的页面。

# 第3章 基于时间树的搜索经验提取方法

本文研究基于搜索经验的查询推荐方法。在此前对复杂搜索的研究中，作者所在实验室提出了时间树理论来帮助用户管理搜索经验。为了有效地基于搜索经验进行查询推荐，首先需要考察用户使用时间树所管理的搜索经验的质量。Habermas 等人提出，人类经验应当满足搜索经验的一致性约束[1]。搜索经验作为人类经验的一种，其质量高低也在搜索经验一致性的高低中体现。基于这一理论，本章研究时间树能否有效帮助用户维护搜索经验的一致性约束。在此基础上，提出基于时间树的搜索经验提取方法，并验证该方法的有效性。

## 3.1 时间树与用户的搜索经验

在时间树理论中，每一个搜索任务都可以被表示成一棵时间树。在时间树中，查询和点击被表示为树的节点，这些节点则被组织成为一种时序溯源的树型结构（Relative Chronological Source-tracking Tree ，*RCST*）。时间树理论中，与 *RCST* 结构相关的定义如下：

**【定义 3.1】***RCST* 节点：*RCST* 节点 *N* 为三元组，$N = \{G, C, T\}$。其中 *G* 为 *RCST* 节点类型。*C* 为 *RCST* 节点内容。*T* 表示 *RCST* 节点产生的时间。

下面给出 *RCST* 节点类型与 *RCST* 节点内容的定义。

**【定义 3.2】***RCST* 节点类型：*RCST* 节点类型 *G* 为 RCST 节点的类型标识值，$G \in \{$查询类型，点击类型$\}$，若 *G* 的取值为查询类型，则称该 *RCST* 节点是一个查询节点，若 *G* 的取值为点击类型，则称该 *RCST* 节点是一个点击节点。一个 *RCST* 节点为查询节点表示该 *RCST* 节点是由用户的查询行为产生的，同理，一个 *RCST* 节点为点击节点表示该 *RCST* 节点是由用户的点击行为产生的。

**【定义 3.3】***RCST* 节点内容：*RCST* 节点内容 *C* 是一段文本。*C* 的值与 *RCST* 节点的节点类型有关，如果该 *RCST* 节点类型 *G* 为查询类型，则 *C* 表示该查询的查询词；如果该 *RCST* 节点类型为点击类型，则 *C* 表示该点击内容的标题。

**【定义 3.4】***RCST* 节点关系：*RCST* 节点间关系 *R(S,T)*为一个标识位，表示 *S* 与 *T* 具有兄弟关系或者父子关系。其中 *S* 与 *T* 是 *RCST* 节点，且 $S \neq T$。

在 *RCST* 节点关系中，*R(S,T)*并不具备广义上的关系的大部分性质，下面给出对 *RCST* 节点关系 *R(S,T)*的具体解释：

(1) *R(S,T)*可以有三种取值：*R(S,T)*是兄弟关系、*R(S,T)*是父子关系及 *R(S,T)*是其他关系。如果 *R(S,T)*为兄弟关系，则表示 *S* 是 *T* 的兄节点；如果 *R(S,T)*为父子关系，则表

示 *S* 是 *T* 的父节点；如果 *R(S,T)*为其他关系，则表示 *S* 既不是 *T* 的兄节点也不是 *T* 的父节点。

时间树作为一种树形结构满足树形结构的一切性质。时间树的 *RCST* 结构从广义的树形结构中继承了父节点的定义，但广义的树形结构中并没有兄节点的概念，是 *RCST* 结构中特有的概念，下面针对时间树的 *RCST* 结构提出兄节点的定义。

**【定义 3.4.1】**兄节点：*RCST* 节点 *A* 是 *RCST* 节点 *B* 的兄节点，当且仅当 *RCST* 节点 *A* 与 *RCST* 节点 *B* 具有共同的父节点并且 *RCST* 节点 *A* 在时间树中的垂直位置高于 *RCST* 节点 *B*。

时间树的 *RCST* 结构按照时间顺序自左向右，自上而下组织而成，根据时间树的 *RCST* 结构组织过程以及父节点及兄节点的定义可以得出如下结论：

**【结论 3.4.1】**一个 *RCST* 节点 *B* 的兄节点 *A* 产生的时间 $T_A$ 一定早于 *RCST* 节点 *B* 产生的时间 $T_B$。

**【结论 3.4.2】**一个 *RCST* 节点 *D* 的父节点 *C* 产生的时间 $T_C$ 一定早于 *RCST* 节点 *D* 产生的时间 $T_D$。

对于 *RCST* 节点 *A*、*B*、*C*，若存在兄弟关系或父子关系 *R(A,B)*及 *R(B,C)*，则根据结论 3.4.1 与结论 3.4.2，*RCST* 节点 A 的产生时间 $T_A$ 早于 *RCST* 节点 *B* 的产生时间 $T_B$，并且 *RCST* 节点 *B* 的产生时间 $T_B$ 早于 *RCST* 节点 *C* 的产生时间 $T_C$。因此 $T_A$ 一定早于 $T_C$。据此可以得出以下推论：

**【推论 3.4.1】***RCST* 兄弟关系与父子关系中，节点产生的时间 *T* 具有传递性。∀ *RCST* 节点 *A*、*B*、*C*，若存在兄弟关系或父子关系 *R(A,B)*及 *R(B,C)*，则 *RCST* 节点 *A* 的产生时间 $T_A$ 一定早于节点 *C* 的产生时间 $T_C$。

(2) *R(S,T)*具有非对称性，即 *R(S,T)*不恒等于 *R(T,S)*。如果 *R(S,T)*为其他关系，则 *R(T,S)*关系不能确定，可能为其他关系，也可能为兄弟关系或父子关系；如果 *R(S,T)*为兄弟关系或父子关系，则 *R(T,S)*一定为其他关系。

(3) 对于父子关系，*R(S,T)*具有非传递性，即∀*RCST* 节点 *A*、*B*、*C*，若存在 *RCST* 关系 *R(A,B)*及 *R(B,C)*，则一定不存在 *RCST* 关系 *R(A,C)*。

(4) 对于兄弟关系，*R(S,T)*具有传递性，即∀*RCST* 节点 *A*、*B*、*C*，若存在 *RCST* 关系 *R(A,B)*及 *R(B,C)*，则一定存在 *RCST* 关系 *R(A,C)*。

(5) 对于兄弟关系与父子关系，*RCST* 节点的产生时间 *T* 的先后关系具有传递性。既若 *R(A,B)*与 *R(B,C)*为兄弟关系或父子关系，则 *RCST* 节点 *A* 的产生时间一定早于 *RCST* 节点 *C* 的产生时间。

**【定义 3.5】**时间树的 *RCST* 结构：时间树的 *RCST* 结构 $T$ 为二元组，$T=\{NC,RC\}$。其中$NC=\{n_1,n_2,\dots,n_p\}$为 *RCST* 节点集，是 *RCST* 节点的集合，$RC=\{r_1,r_2,\dots,r_q\}$为 *RCST* 节点关系集，是 *RCST* 节点关系的集合。

时间树 *RCST* 结构的 *RCST* 节点关系集中包含了所有具有父子关系或者具有兄弟关系的节点对。基于 *RCST* 结构中的 *RCST* 节点关系，时间树能够帮助用户理解查询与点击之间的关系:

(1) 用户可以通过时间树理解查询与查询，查询与点击之间的相对时间关系

时间树上的节点与用户的搜索过程中产生的查询及点击行为一一对应，即时间树上的每个 *RCST* 节点代表用户搜索过程中的一次点击或者查询，并且用户搜索过程中的每一次点击与查询都被表示为时间树上的一个 *RCST* 节点。对于时间树上的任意两个RCST节点 *A*、*B*，一定存在$R(A,B)\in RC$。

若 *R(A,B)*不是其他关系，则依据结论 3.4.1 以及 3.4.2， *RCST* 节点 *A* 的产生时间 $T_A$ 一定早于 RCST 节点 *B* 产生的时间 $T_B$，确定了 *RCST* 节点 *A* 与 *RCST* 节点 *B* 的相对时间关系。

若 *R(A,B)*是其他关系，并且在 *RCST* 关系集 *RC* 中能够找到一系列兄弟关系或父子关系{*R(A,C*$_1$*),R(A,C*$_2$*),…,R(C*$_{n-1}$*,C*$_n$*),R(C*$_n$*,B)*}，则根据推论 3.4.3，*RCST* 兄弟关系及父子关系中的节点产生时间先后具有传递性，可以得到 *RCST* 节点 *A* 的产生时间 $T_A$ 早于 *RCST* 节点 *B* 的产生时间 $T_B$，确定了 *RCST* 节点 *A* 与 *RCST* 节点 *B* 的相对时间关系。

综上所述，时间树的 *RCST* 结构能够确定查询与查询，查询与点击之间的相对时间关系。

(2) 用户可以通过时间树理解查询节点的来源

时间树作为一种树形结构，除根节点外的所有节点都具有父节点。因此如果时间树上的一个查询节点 *A* 不是时间树根节点，一定存在父子关系$R(X,A)\in RC$。用户可以依据该关系理解查询节点的来源。

若 *RCST* 节点 *X* 的节点类型 $G_X$ 为点击类型，则 *RCST* 节点 *A* 的查询词 $C_A$ 来自于以 *RCST* 节点 *X* 的节点内容 $C_X$ 为标题的文档中。

若 *RCST* 节点 *X* 的节点类型 $G_X$ 为查询类型，则 *RCST* 节点 *A* 的查询词 $C_A$ 是对 *RCST* 节点 *X* 的查询词 $C_X$ 的一次查询重构。若不是查询重构，*RCST* 节点 *A* 的查询词 $C_A$ 可能是来自任务要求或者用户自身的知识。

(3) 用户可以根据时间树的 *RCST* 结构对复杂搜索任务进行子任务的划分

对于时间树 *RCST* 节点集 *NC* 中的任一 *RCST* 节点 *A*，至多有一个 *RCST* 节点$F\in NC$

使得父子关系$R(F,A) \in RC$，其中 *RC* 为 *RCST* 节点关系集。若存在 *RCST* 节点关系$R(F_1,A) \in RC$为父子关系，那么任意的 *RCST* 节点关系$R(F_2,A) \in RC$都不是父子关系。

但 *RCST* 节点可以有若干的子节点。即对于 *RCST* 节点集 *NC* 的子集 $NCS\{B,S_1,S_2,...,S_n\}$，关系$R(B,S_1) \in RC$，$R(B,S_2) \in RC$，…，$R(B,S_n) \in RC$可以均为父子关系。

以上述子集 *NCS* 为例，时间树上的 *RCST* 节点 *B*，在与 *RCST* 节点 $S_1,...,S_n$ 产生更多父子关系的同时也会使 *RCST* 节点 $S_1,...,S_n$ 两两之间产生更多的兄弟关系，当这些兄弟关系达到一定数量时，*RCST* 节点 *B* 与它的兄弟节点间的视觉距离就会变大。这种视觉距离直观地将时间树分为几个部分，用户可以根据这种视觉距离进行子任务的划分。

时间树的 *RCST* 结构对用户理解查询与点击之间关系的帮助效果如图 3.1 所示。

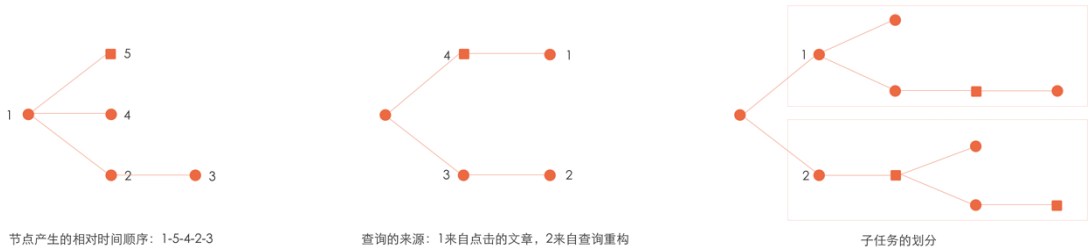


图 3.1 时间树 RCST 结构对用户的帮助效果图

Fig. 3.1 diagram of the RCST structure of TimeTree helping users

(1) *RCST* 节点 1～5 具有父子关系 *R(1,2),R(1,4),R(1,5),R(2,3)*，以及兄弟关系 *R(5,4),R(5,2),R(4,2)*，根据结论 3.4.1、结论 3.4.2 以及推论 3.4.1，*RCST* 节点 1～5 的产生时间先后顺序为 1-5-4-2-3。

(2) *RCST* 节点 1～4 具有父子关系 *R(4,1)*及 *R(3,2)*，查询节点 1 的父节点为点击节点 4，查询节点 2 的父节点为点击节点 3，查询节点 1 来自用户点击的文章，查询节点 2 来自查询重构或题目要求及用户的自身的知识。

(3) 由于以 *RCST* 节点 1 和 2 为根节点的子树中兄弟关系的增多，*RCST* 节点 1 和 2 的视觉距离增大，用户可以直观清晰地进行子任务的划分。

时间树通过组织 *RCST* 结构，在 *RCST* 结构中记录 *RCST* 节点以及 *RCST* 节点关系记录用户的搜索过程。由此引出的一个问题是，时间树是否同时记录了用户的搜索经验以及基于这些搜索经验能否进行查询推荐。本章通过建立用户搜索经验模型，设计用户复杂搜索过程验证实验，并对实验结果进行分析，研究这一问题的第一部分，即时间树是否记录了用户的搜索经验。

为了回答时间树中是否蕴含高质量的用户搜索经验的问题，首先需要对用户的搜

索经验进行模型化的描述。

## 3.2 用户搜索经验表示及一致性模型

Habermas 等人在 2000 年提出，人们在对过去事件的回顾过程中传达出对于过去事件的时间、因果与主题三方面的经验，这三方面的经验在用户对过去事件回顾过程中的时间一致性、因果一致性与主题一致性三种一致性中体现[42]。本文以此为理论依据，定义用户搜索经验模型如下：

**【定义 3.6】**用户搜索经验模型：用户搜索经验模型 $E$ 为一个三元组，$E = \{TE, CE, THE\}$，其中 $TE$ 为用户时间经验，$CE$ 为用户因果经验，$THE$ 为用户主题经验。

(1) 时间经验为用户在搜索过程中进行查询和点击的时间先后关系。例如用户首先查询了某个词，进行了某些点击，又查询了某些词。在进行搜索的过程中，用户的查询和点击会按照时间顺序先后进行，而不可能同时进行两个查询或点击行为，因此用户的查询和点击一定具有时间先后顺序。

(2) 因果经验为用户所进行查询的查询来源。用户进行的某个查询，一定是出于某种原因，例如查询词来自点击结果，或者查询词来自用户当前进行任务的任务需求，或者查询词来自用户的现有经验。

(3) 主题经验为用户对复杂搜索任务所进行的子任务划分。用户在进行复杂搜索任务的过程中，由于任务的复杂性，用户常常需要将任务划分为各个小的子任务，并在逐一完成各个子任务的过程中完成复杂搜索任务。这些子任务提供了一种复杂搜索任务的清晰结构展现，反映了用户对复杂搜索任务的规划。

时间经验，因果经验，主题经验共同组成了用户在复杂搜索过程中的搜索经验模型，但三种搜索经验无法被直观观测或直接衡量。为了解决这一问题，本文采用Habermas等人提出的搜索经验评价方法对搜索经验进行评价，提出搜索经验一致性模型，定义如下：

**【定义3.7】**用户搜索经验一致性模型：用户搜索经验一致性模型$C$为一个三元组，$C = \{TC, CC, THC\}$，其中$TC$为用户在对搜索过程回顾过程中的时间一致性，$CC$为用户在对搜索过程回顾过程中的因果一致性，$TH$为用户用户在对搜索过程回顾过程中的主题一致性。

(1) 时间一致性指用户能够准确回忆起自己在搜索过程中查询与点击的先后顺序。

(2) 因果一致性指用户能够准确回忆起搜索中的某个查询的查询动机。

(3) 主题一致性指用户能够合理地对复杂搜索任务进行子任务划分，以及对搜索过

程中的某一个阶段，能够判断其中的查询与点击分别属于哪些子任务。

如果时间树能够有效记录用户搜素经验 *E<TE, CE, THE>*，那么用户在依据时间树进行搜索过程回顾时能够获得较高的时间一致性，因果一致性以及主题一致性，在不使用时间树进行搜索过程回顾时，在用户搜索经验一致性模型 *C* 上应当有较差表现。

基于上述分析，本章从用户对复杂任务搜索过程的回顾的角度设计实验，依据用户在对复杂搜索过程进行回顾时在时间一致性，因果一致性以及主题一致性上的表现来验证时间树对用户搜索经验的有效记录。

## 3.3 基于时间树的搜索经验蕴含性验证实验设计

本节实验的主要目的是以用户对复杂搜索过程进行回顾为载体，基于用户搜索经验一致性模型，通过分别考察用户使用时间树与不使用时间树，考察时间树能否更有效地维护搜索经验的一致性，从而验证时间树中是否蕴含高质量的用户搜索经验。

### 3.3.1 复杂搜索任务设计

为了针对上述目的进行实验的设计，首先需要设计搜索场景以使用户的搜索过程满足复杂搜索的定义。本文采用 Singer 等[43]提出的复杂搜索任务的定义来定义复杂搜索任务——“复杂搜索任务需要包括以下行为中的一种或几种：汇总，探索和组合。”以此为理论基础，本节进一步参考 White 等[4]提出的用户可能需要在其中进行复杂搜索的三种典型场景：1）系统对信息的索引不够充分；2）搜索任务本身需要浏览及探索；3）用户对搜索任务领域的知识匮乏以至难以对查询进行组织或定位信息领域。以这三种复杂搜索的典型场景为原则，设计复杂搜索任务，使实验参与者进行的搜索过程满足复杂搜索场景。

首先为了满足 White 等提出的复杂搜索的第一种典型场景——系统对信息的索引不够充分，实验采用学术搜索引擎进行。在学术搜索引擎中，受到版权及索引复杂度的限制，文章常常不会被全文索引，而仅仅会对标题及摘要进行索引，因此符合“系统对信息的索引不够充分”的场景需求。

为了满足 White 等提出的复杂搜索的后两种典型场景，实验共设计了两类共 4 个任务。

前两个搜索任务被称为学习型任务，采用实验参与者的母语汉语进行布置。这两个搜索任务的话题都围绕实验参与者不熟悉但在日常生活中能够略微接触到的领域，因此实验参与者需要围绕任务话题进行浏览探索，这符合 White 等提出的第一种复杂搜索典型场景，是那些“需要浏览和探索”的任务。

后两个搜索任务被称为试探型任务，采用实验参与者的非母语英语进行布置。参与者不被直接告知搜索任务的具体描述，而是首先被要求阅读一篇英文摘要，在两天以后，参与者被告知需要围绕之前阅读的材料凭记忆进行搜索。由于人类记忆曲线的作用以及非母语环境下参与者对文章内容理解不够充分，实验参与者难以对查询进行组织，只能凭借记忆进行试探性查询探索，这符合 White 等提出的第二种复杂搜索典型场景，是那些“用户对搜索任务领域的知识匮乏以至难以对查询进行组织或定位信息领域”的任务。

对于学习型任务，参与者被要求根据任务要求完成搜索报告；对于试探型任务，参与者被要求完成一份与阅读材料相关的文章标题清单。两个学习型任务的任务描述及两个试探型任务的英文摘要出处如下：

(1) **任务一**：搜索相关论文，了解化疗中的常用药物，药物机理，使用方法，适用情况及副作用等，尤其针对癌症治疗，了解化疗药物的治疗手段，组合方式等。(不仅仅要了解支持化学疗法的药物，还要求了解对化学疗法进行辅助的药物，如缓解化学疗法副作用的药物等。)

(2) **任务二**：完成一篇报告，内容包括 PM2.5 的概念、来源，PM2.5 的危害以及危害原因。找到中国 PM2.5 污染最为严重的几个城市，并分析其形成原因。研究降低 PM2.5 的方法，并分析其原理以及其实际实施可能性(包括但不局限于利弊)。

(3) **任务三**：S. Shunmuga Krishnan, Ramesh K. Sitaraman: Video Stream Quality Impacts Viewer Behavior: Inferring Causality Using Quasi-Experimental Designs. IEEE/ACM Trans. Netw. 21(6): 2001-2014 (2013)

(4) **任务四**：Hanqiang Cheng, Yu-Li Liang, Xinyu Xing, Xue Liu, Richard Han, Qin Lv, Shivakant Mishra: Efficient misbehaving user detection in online video chat services. WSDM 2012: 23-32

### 3.3.2 实验过程设计

本实验通过分别考察用户在对复杂搜索过程回顾中的时间一致性，因果一致性和主题一致性来综合考察时间树在用户对复杂搜素过程回顾中搜索经验一致性模型 $E$ 上的表现，从而考察时间树能否更有效地维护搜索经验的一致性，以此验证时间树上搜索经验的蕴含性。

为了衡量用户使用时间树对复杂搜索过程回顾过程中的三种一致性，本实验采用按照时间顺序线性地记录查询和点击的方法（Timeline SearchLog，以下简称 SearchLog）与时间树进行对比。SearchLog 形式如图 3.2 所示。

若你喜欢怪人其实我很美
----若你喜欢怪人其实我很美_打回原形_百度音乐
----“若你喜欢怪人,其实我很美”你怎么理解陈奕迅的这句歌词? - 知乎
----若你喜欢怪人,其实我很美_百度知道
陈奕迅
十年
----十年_陈奕迅在线试听_高音质歌曲_百度音乐
----十年歌词是什么意思?_百度知道
十年之前

图 3.2 SearchLog 示意图
Fig. 3.2 diagram of SearchLog

与普通浏览器日志类似，SearchLog 按照用户进行查询点击的顺序记录查询与点击行为。其中查询以查询关键词的形式进行记录与展示，点击以点击链接标题加“----”前缀的形式进行记录与展示。

参与者被要求完成 3.3.1 节中所述的 4 个复杂搜索任务。每一位参与者分别从两类任务中被随机分配一个任务使用时间树进行搜索过程的记录与回顾，另一个任务则使用 SearchLog 进行搜索过程的记录与回顾。并且最终保证每个任务分别有 4 位参与者使用时间树和 SearchLog 进行搜索过程的记录与回顾。

由于实验所设计任务的复杂性，搜索难以一次性完成，并且为使实验场景满足 Singer 等提出的复杂搜索任务的定义，使实验参与者能够对实验任务进行深入的探索，本实验分为 4 个阶段进行，实验过程的设计如下：

(1) 第一阶段：准备及搜索阶段

由于学习型任务与试探型任务在任务形式上的不同，第一阶段的设计有所不同。

学习型任务的第一阶段仅有搜索一部分，分 3 次进行，每次实验参与者搜索全部 2 个学习型任务，每个任务每次搜索时间限制为 30 分钟。

在复杂搜索过程中，由于搜索任务的复杂性，搜索者常常需要将复杂搜索任务分为多次完成，因此在复杂搜索过程中常常会发生搜索任务的中断与恢复。在搜索任务从中断状态恢复时，搜索者需要回忆自己已经从之前的搜索中获取到的信息，在这一过程中，蕴含了搜索者的回顾过程。因此，本实验利用这一特性，设计每次搜索的次间有 1 天的时间间隔，以探究在实验参与者将搜索任务从中断状态恢复时，时间树对参与者的帮助情况。

试探型任务的的第一阶段分为准备部分与搜索部分。

准备部分实验参与者被要求阅读 2 个试探型任务的英文阅读材料。非母语的阅读材料将大大增加用户组织查询的困难度，因此准备阶段不对实验参与者的阅读进行限

时，以防止实验参与者在搜索阶段无词可搜的情况。

搜索部分在准备部分完成两天后开始，分 2 次进行，每次实验参与者搜索全部 2 个试探型任务，每个任务每次搜索时间限制为 30 分钟。出于与学习型任务一样的理由，次间有 1 天的时间间隔。

(2) 第二阶段：报告阶段

该阶段在实验开始的一周后进行，时间为 2 天。参与者被要求完成报告，时间不限。

对于学习型任务，实验参与者需要在报告中汇报自己的搜索成果，搜索成果应当覆盖搜索任务中涉及的所有内容。

对于试探型任务，实验参与者需要在报告中完成一份文章清单，清单中包括参与者认为与所给阅读材料内容相关的文章标题。

(3) 第三阶段：搜索过程调查及小组讨论阶段

该阶段在第二阶段结束后进行，持续时间为 1 小时。在第三阶段中，组织小组讨论，要求参与者探讨时间树使用中的主观感受并且填写主观调查问卷，问卷包括 11 个问题，具体的问卷内容及结果分析将在 3.4 节中介绍。

(4) 第四阶段：搜索过程回顾及回顾过程调查阶段

在第三阶段结束一周后进行。在这一阶段实验参与者将使用时间树或 SearchLog 对第一阶段中进行的搜索任务进行回顾，并进行讲解。

为了保证实验参与者在对搜索任务进行回顾的过程中尽量依靠时间树及 SearchLog 进行回忆，避免参与者在有所准备的情况下本能地对搜索过程进行记忆，增加实验的干扰因素，在整个实验进行到第四阶段以前，实验参与者都不被告知需要进行搜索过程的回顾与讲解。

参与者首先将被允许根据时间树或 SearchLog 对自己的搜索过程进行回忆，回忆完成后将依据时间树或 SearchLog 进行搜索过程的讲解，讲解过程中将有两名专家对其讲解过程进行评分。在参与者的讲解过程中，专家可以随时针对参与者的讲解提出问题。在讲解结束后，参与者被要求对搜索任务进行子任务划分，并需要根据时间树或 SearchLog 对随机生成的 10 个查询关键词给出查询原因。

在上述回顾与讲解过程完成后，针对回顾的过程填写调查问卷，问卷包括 3 个问题，具体问卷内容及结果分析将在 3.4 节中介绍。

为了最大限度减少实验的干扰因素，实验需在可控环境下进行。实验环境控制原则如下：

(1) 参与者只可以使用百度学术进行搜索。

(2) 参与者只可以进行搜索任务范围内的搜索。

(3) 参与者之间不允许进行任务相关话题的讨论。

(4) 参与者在实验时间以外，尽最大可能不对任务相关话题进行思考。

## 3.4 实验结果分析

本实验的第三阶段及第四阶段中，对实验参与者的搜索及回顾讲解过程进行了主观评估、专家评估及客观评估三种评估，本节将针对三种评估的结果分别进行分析。

### 3.4.1 主观评估分析

实验参与者在实验的第三阶段以及第四阶段分别完成了一份调查问卷，用于实验参与者在搜索过程中以及对搜索过程进行回顾的过程中主观感受的研究，本节分别对两次调查的结果进行分析，从用户感受的角度对时间树维护一致性的效果进行评估，称为主观评估。

#### 3.4.1.1 复杂搜索过程的主观评估

第三阶段的调查是关于用户在使用时间树进行复杂搜索过程中的主观感受。搜索过程主观调查通过给出一系列断言并要求实验参与者对这些断言进行评分，以此来评估参与者在使用时间树进行搜索的过程中的主观感受。

复杂搜索过程主观调查分为 3 个部分共 11 个断言组成。第一部分为断言 1～断言 4，用于通过实验参与者的主观感受验证时间树的易用性；第二部分为断言 5～断言 7，用于考察实验参与者是否感受到在搜索过程中利用了时间树的 *RCST* 结构进行了搜索经验的提取；第三部分为断言 8～断言 11，用于验证实验参与者将搜索任务从中断状态恢复时，参与者能否感受到时间树对参与者提供了帮助。

搜索过程主观调查的 3 个部分共 11 个断言具体如下：

第一部分：

(1) 断言 1：很容易学习如何使用时间树，在学习过程中不需要花费很多的时间和精力。

(2) 断言 2：时间树是非常易于操作的。

(3) 断言 3：使用时间树来管理搜索任务没有带来额外的负担。

(4) 断言 4：相比使用时间树所带来的额外负担，使用时间树所带来的好处更多。

第二部分：

(1) 断言 5：我曾经利用过时间树节点的相对位置来判断查询和点击的先后顺序。

(2) 断言 6：我曾经利用过时间树节点的父子关系来回忆我为什么要搜索某个查询。

(3) 断言 7：我曾经利用过时间树的树形结构来划分搜索子任务。

第三部分：

(1) 断言 8：在每次间隔两天再继续学习型搜索任务时，与不使用时间树相比，使用时间树能帮助我更好地回忆起上次的搜索任务进展到哪里，以及下一步应该做什么。

(2) 断言 9：时间树很直观地展示了我的学习型任务搜索过程。

(3) 断言 10：在每次间隔两天再继续试探型搜索任务时，与不使用时间树相比，使用时间树能帮助我更好地回忆起上次的搜索任务进展到哪里，以及下一步应该做什么。

(4) 断言 11：时间树很直观地展示了我的试探型任务的搜索过程。

下面针对搜过过程主观调查的每一部分分别进行结果分析：

搜索过程主观调查结果第一部分如表 3.1 所示。

表 3.1 搜索过程主观调查结果第一部分

Table 3.1 results of subjective survey on search process part 1

| 断言编号 | 断言 | 评分范围 | 最高评分 | 最低评分 | 平均评分 | 标准差 |
|---|---|---|---|---|---|---|
| 1 | 断言 1 | 0-10 | 10 | 8 | 9.25 | 0.886 |
| 2 | 断言 2 | 0-10 | 10 | 7 | 8.25 | 0.886 |
| 3 | 断言 3 | 0-10 | 9 | 4 | 7.38 | 1.51 |
| 4 | 断言 4 | 0-10 | 10 | 5 | 8.88 | 1.73 |

搜索过程主观调查第一部分的主要目的是考察参与者主观感受中时间树是否易用。

观察断言 1 和断言 2 的最高评分、最低评分、平均评分及评分标准差可以得出，参与者认为时间树是易于学习并且易于操作的。

断言 3 的最高评分分别为 9 分，平均评分为 7.38，但是最低评分 4 分，并且标准差偏大，说明参与者在时间树在使用中带来的负担问题上，参与者的主观感受上有所不同。在小组讨论部分，参与者提出的使用时间树时遇到的问题解释了这一评分结果。参与者认为，使用时间树时带来的负担主要来自于搜索过程中对时间树节点的调整。时间树的 *RCST* 结构中很重要的一个性质是具有来源追踪能力，但在实验场景下，由于捕获参与者当前查询来源所使用的技术尚不成熟，造成偶尔需要参与者手动调整时间树节点的情况发生，参与者认为，这是时间树给他们带来负担的主要原因。因此可以认为，参与者的负担主要是由技术原因导致，而并非时间树的 *RCST* 结构本身，并且本断言获得的平均评分高于 7 分，可以认为，时间树给实验参与者带来的使用负担在可接受的范围内，断言 4 获得的评分也证明了这一点。

断言 4 是断言 3 的后续断言，为了考察在参与者认为时间树为其带来负担的前提下，是否认为使用时间树的优点大于其缺点。虽然断言 4 获得的最低评分偏低，标准差相较于其他问题偏大，但从断言 4 的最高评分及平均评分结果中仍可以看出，参与者倾向于认为使用时间树进行复杂搜索带来的好处要多于坏处。

综合断言 1～断言 4 的参与者评分结果可以得出，在参与者的主观感受中，时间树具有较高的易用性。搜索过程主观调查结果第二部分如表 3.2 所示。

表 3.2 搜索过程主观调查结果第二部分

Table 3.2 results of subjective survey on search process part 2

| 断言编号 | 断言 | 评分范围 | 最高评分 | 最低评分 | 平均评分 | 标准差 |
|---|---|---|---|---|---|---|
| 5 | 断言 5 | 0/1 | n/a | n/a | 0.875 | n/a |
| 6 | 断言 6 | 0/1 | n/a | n/a | 0.750 | n/a |
| 7 | 断言 7 | 0/1 | n/a | n/a | 0.750 | n/a |

搜索过程主观调查第二部分的主要目的是考察实验参与者是否感受到在搜索过程中利用了时间树的 *RCST* 结构进行了搜索经验的提取。参与者为此部分断言的评分只有 0 和 1 两种，0 代表不同意断言，1 表示同意断言。因此，平均评分代表了同意该断言的参与者的比例。

断言 5 的平均评分为 0.875，因此有百分之 87.5%的参与者认为在使用时间树进行搜索的过程中利用过时间树节点的相对位置来判断查询和点击的先后顺序，即 87.5%的参与者主观上认为自己利用时间树的 *RCST* 结构从时间树上获取到了时间经验。

断言 6 及断言 7 的平均评分虽然略低于断言 5 的平均评分，但它们仍然处于较高的范围内。说明多数参与者认为在使用时间树进行搜索的过程中利用过时间树的节点的父子关系来回忆为什么要搜索某个查询以及树形结构来划分搜索子任务，即多数参与者主观上认为自己利用时间树的 *RCST* 结构从时间树上获取到了因果经验以及主题经验。

断言 5～断言 7 的评分结果可以说明，多数参与者主观上认为自己在使用时间树进行搜索的过程中从时间树上获取到了搜索经验模型中的时间经验，因果经验以及主题经验。搜索过程主观调查第三部分的结果进一步支持了这样的结论。

搜索过程主观调查结果第三部分如表 3.3 所示。

表 3.3 搜索过程主观调查结果第三部分

Table 3.3 results of subjective survey on search process part 3

| 断言编号 | 断言 | 评分范围 | 最高评分 | 最低评分 | 平均评分 | 标准差 |
|---|---|---|---|---|---|---|
| 8 | 断言 8 | 0-10 | 10 | 8 | 9.38 | 0.518 |
| 9 | 断言 9 | 0-10 | 10 | 7 | 9.25 | 0.886 |
| 10 | 断言 10 | 0-10 | 10 | 8 | 9.25 | 0.886 |
| 11 | 断言 11 | 0-10 | 10 | 8 | 9.00 | 0.756 |

搜索过程主观调查第三部分是对调查第二部分的场景细化，主要目的是考察实验参与者将搜索任务从中断状态恢复时，能否感受到时间树的 *RCST* 结构对参与者提供了帮助。

断言 8 与断言 10 分别针对学习型任务与试探型任务提出，综合观察它们所获得的最高评分、最低评分以及平均评分，可以得出参与者认为在将搜索任务从中断中恢复时，时间树能够帮助参与者进行任务进展情况的判断。但断言 8 所获得的评分标准差更小，说明对于学习型任务，参与者们达成这种认识的统一性更高。搜索经验模型所定义的三种经验中，时间经验与主题经验是判断任务进展情况的重要参考因素。由此可以得出，在参与者的主观感受中，参与者在将搜索任务从中断中恢复的过程中，参与者从时间树中提取了时间经验以及主题经验。

断言 9 与断言 11 分别针对学习型任务及试探型任务考察了时间树在展示搜索过程上带给实验参与者的主观感受。从获得的最高评分、最低评分、平均评分以及评分的标准差可以看出，参与者普遍认为时间树很直观地展示了学习型任务以及试探型任务的搜索过程。

#### 3.4.1.2 搜索过程回顾的主观评估

第四阶段的调查是关于参与者在使用时间树以及 SearchLog 进行复杂搜素过程的回顾中的主观感受，以搜索经验模型为理论依据分别针对时间树与 SearchLog 提出了 3 个问题，要求实验参与者依据自身感受对其进行范围在 0-10 范围内的整数评分，从实验参与者主观感受的角度，通过对比使用时间树与不使用时间树时参与者的主观评分，评估时间树在搜索过程回顾中对时间一致性，因果一致性以及主题一致性的维护效果。

搜索过程回顾主观调查中的 3 个问题如下：

(1) 问题 1：使用该方法对我回忆起查询点击的先后顺序的帮助程度。

(2) 问题 2：使用该方法对我回忆起为什么要搜索某个查询，或点击某个搜索结果的帮助程度。

(3) 问题 3：使用该方法对我回忆起搜索过程中的子任务的帮助程度。

搜过过程回顾主观调查问题在时间树与 SearchLog 上获得的最高评分、最低评分、平均评分以及评分的标准差如表 3.4 所示。

对于问题 1，时间树与 SearchLog 获得的最低评分分别为 7 与 4，时间树明显高于 SearchLog，并且时间树的评分标准差也小于 SearchLog 的评分标准差。但时间树与 SearchLog 获得的最高评分都是 10 分，并且时间树获得的平均评分仅略高于 SearchLog

获得的平均评分。由此可以得出，时间树与 SearchLog 在帮助参与者回忆查询点击的先后顺序上都带给参与者较优的主观感受。

表 3.4 搜索过程回顾主观调查结果

Table 3.4 results of subjective survey on search process recalling

| 问题 | 数据类别 | 时间树 | SearchLog |
|---|---|---|---|
| 问题 1 | 最高评分 | 10 | 10 |
| | 最低评分 | 7 | 4 |
| | 平均评分 | 8.50 | 7.63 |
| | 标准差 | 1.20 | 1.85 |
| 问题 2 | 最高评分 | 10 | 8 |
| | 最低评分 | 7 | 0 |
| | 平均评分 | 8.75 | 6.63 |
| | 标准差 | 0.886 | 1.19 |
| 问题 3 | 最高评分 | 10 | 8 |
| | 最低评分 | 7 | 5 |
| | 平均评分 | 8.88 | 5.88 |
| | 标准差 | 0.991 | 2.85 |

对于问题 2，时间树获得的最高评分为 10，高于 SearchLog 获得的最高评分 8。时间树获得的最低评分为 7，而 SearchLog 获得了极低的最低评分 0。并且时间树获得的平均评分高于 SearchLog 的平均评分，评分标准差低于 SearchLog 的评分标准差。由此可以得出，参与者认为时间树对回忆查询动机的帮助程度高于 SearchLog 对回忆查询动机的帮助程度。

对于问题 3，时间树获得的最高评分与最低评分分别高于 SearchLog 获得的最高评分与最低评分。并且时间树获得的平均评分远高于 SearchLog 获得的平均评分，评分标准差远低于 SearchLog 的评分标准差。说明参与者普遍认为时间树对回忆起搜索过程中的子任务的帮助程度高于 SearchLog 对回忆起搜索过程中的子任务的帮助程度。

综合问题 1～问题 3 的结果分析，在帮助参与者进行时间经验提取方面，参与者认为时间树与 SearchLog 都有较优的表现。在帮助参与者进行因果经验以及主题经验提取方面，参与者认为时间树的表现要优于时间树。

为了验证以上结论，本研究使用显著性水平$\alpha$=0.05 的配对 T 检验（Paired Student's T-test）对搜索过程回顾主观调查结果进行了分析。

配对 T 检验用于在样本数量较小时，检验来自两个总体的配对样本的样本均值是否有显著性差异。当配对 T 检验对应的 p-值（p-value）小于等于显著性水平$\alpha$时，两样本来自总体的差异是显著的，当配对 T 检验对应的 p-值大于显著性水平$\alpha$时，两样本来自总体的差异是不显著的。

搜索过程回顾主观调查的检验结果如表 3.5 所示。

表 3.5 搜索过程回顾主观调查 T 检验结果

Table 3.5 T-test results of subjective survey on search process recalling

| 问题 | 配对 T 检验 p-值 |
|---|---|
| 问题 1 | **0.176** |
| 问题 2 | 0.0103 |
| 问题 3 | 0.0263 |

在帮助参与者回忆起查询点击的相互顺序方面，时间树与 SearchLog 获得的参与者评分的 T 检验 p-值大于显著性水平 0.05，即时间树与 SearchLog 获得的评分差异不是统计显著的。

在帮助参与者回忆查询动机及搜索过程中的子任务两方面，时间树与 SearchLog 获得的参与者评分的 T 检验 p-值均小于显著性水平 0.05，说明时间树与 SearchLog 在这两方面获得的评分差异是统计显著的。

综合上述分析，得出结论：在帮助参与者进行时间经验提取方面，参与者认为时间树与 SearchLog 都有较优的表现。在帮助参与者进行因果经验以及主题经验提取方面，参与者在使用时间树时的主观感受统计显著地优于使用 SearchLog 时的主观感受。

### 3.4.2 专家评估分析

在实验的第四阶段，实验参与者使用时间树或 SearchLog 对第一阶段中进行的搜索任务进行回顾，并进行讲解。两名博士研究生将作为专家，对实验参与者的回顾性讲解进行 0-10 范围内的整数评分。两名专家评分的平均值，将作为实验参与者回顾过程的专家评分。

专家评估部分将针对 4 个评价因子对参与者使用时间树以及 SearchLog 时所作的回顾性讲解作出质量评价。根据用户搜索经验一致性模型，专家评估的前三个评价因子分别为参与者在对搜索任务进行回顾过程中的时间一致性、因果一致性以及主题一致性。第四个评价因子为参与者的回顾过程带给专家的整体感受。

由于学习型任务与试探型任务在复杂搜索场景上差异，参与者在对两种类型任务进行回顾的过程中表现不同，带给专家的感受也有所不同，因此，下面分别针对学习型任务及试探型任务对专家评估结果进行分析。

(1) 学习型任务的专家评估结果如表 3.6 所示。

从结果中可以看出，时间一致性方面，参与者在对学习型任务的搜索过程进行回顾过程中，使用时间树获得的专家评分仅略低于使用 SearchLog 的专家评分。这与

3.4.1 中进行的主观评估结果一致，在学习型任务中，时间树与 SearchLog 在帮助参与者维护时间一致性方面都有较优的表现。

表 3.6 学习型任务专家评估结果表

Table 3.6 expert evaluation results of learning task

| 编号 | 评价因子 | 时间树 | SearchLog |
|---|---|---|---|
| 1 | 时间一致性 | 7.88 | 8.13 |
| 2 | 因果一致性 | 8.19 | 6.94 |
| 3 | 主题一致性 | 8.25 | 6.88 |
| 4 | 整体感受 | 7.81 | 6.81 |

参与者使用时间树时在时间一致性方面的专家评分低于在因果一致性与主题一致性方面获得的专家评分；使用 SearchLog 时在时间一致性方面的专家评分高于在因果一致性与主题一致性方面获得的专家评分。说明在使用时间树对学习型任务的搜索过程进行回顾时，相对于时间一致性，参与者维护了更高的因果一致性与主题一致性，而在使用 SearchLog 对学习型任务的搜索过程进行回顾时，相对于因果一致性与主题一致性，参与者维护了更高的时间一致性。

在因果一致性与主题一致性方面，参与者在使用时间树时与使用 SearchLog 时相比均获得了更高的专家评分。说明专家认为，参与者在使用时间树时维护了更高的因果一致性与主题一致性。

使用时间树时参与者的回顾过程带给专家的整体感受高于使用 SearchLog 时参与者的回顾过程带给专家的整体感受，说明专家认为在使用时间树时，相比于使用 SearchLog 时，参与者能够更准确地回顾学习型任务的搜索过程并且进行更清晰的讲解。

为了更准确地验证上述结论，在对专家评估结果的分析中，同样采用了显著性水平$\alpha$=0.05 的配对 T 检验对专家评估结果进行了分析。配对 T 检验结果如表 3.7 所示。

表 3.7 学习型任务专家评估配对 T 检验结果

Table 3.7 expert evaluation paired T-test results of learning task

| 编号 | 评价因子 | 配对 T 检验 p-值 |
|---|---|---|
| 1 | 时间一致性 | 0.792 |
| 2 | 因果一致性 | 0.00107 |
| 3 | 主题一致性 | 0.000142 |
| 4 | 整体感受 | 0.00347 |

在时间一致性方面，参与者在使用时间树与 SearchLog 进行学习型任务的搜索过程回顾时获得的专家评分配对 T 检验 p-值大于显著性水平 0.05，说明从专家评估的角度，在帮助用户维护时间一致性方面，时间树与 SearchLog 的差异并不是统计显著的。

因果一致性、主题一致性以及整体感受三个方面，参与者在使用时间树与 SearchLog 进行学习型任务搜索过程回顾时获得的专家评分配对 T 检验 p-值均小于显著性水平 0.05，说明从专家评估的角度，在帮助参与者维护因果一致性、主题一致性以及更准确地回顾学习型任务的搜索过程并进行清晰的讲解方面，时间树与 SearchLog 的差异是统计显著的。

综合上述分析，在参与者对学习型任务进行回顾及讲解过程中，时间树与 SearchLog 都能够有效帮助用户维护时间一致性。与使用 SearchLog 时相比，在使用时间树时，参与者能够统计显著地获得更高的因果一致性与主题一致性。并且使用时间树，参与者能够更准确地回顾学习型任务的搜索过程并进行更清晰地讲解。

(2) 试探型任务的专家评估结果如表 3.8 所示。

表 3.8 试探型任务专家评估结果表

Table 3.8 expert evaluation results of exploratory task

| 编号 | 评价因子 | 时间树 | SearchLog |
|---|---|---|---|
| 1 | 时间一致性 | 7.94 | 7.94 |
| 2 | 因果一致性 | 7.75 | 7.13 |
| 3 | 主题一致性 | 8.06 | 6.94 |
| 4 | 整体感受 | 7.69 | 6.88 |

时间一致性方面，参与者在对试探型任务的搜索过程进行回顾中，使用时间树与使用 SearchLog 获得了相同的专家评分。与学习型任务中的结果相似，说明在试探型任务中，时间树与 SearchLog 在帮助参与者维护时间一致性方面也都有较优的表现。

使用时间树时，时间一致性获得的专家评分略高于因果一致性获得的专家评分，略低于主题一致性的专家评分；使用 SearchLog 时，时间一致性获得的专家评分高于因果一致性与主题一致性获得的专家评分。说明在试探型任务中，专家认为 SearchLog 对时间一致性的维护依然同样高于对因果一致性与主题一致性的维护。但专家认为，参与者在使用时间树对试探型任务进行搜索过程回顾时，因果一致性低于时间一致性。

单独考察因果一致性，时间树在因果一致性方面获得的专家评分略高于 SearchLog，但分差似乎并不明显。这一点将在下文假设检验中验证。

主题一致性方面，参与者使用时间树获得的专家评分高于使用 SearchLog 获得的专家评分。说明与学习型任务相同，从专家评估的角度，参与者在使用时间树进行试探型任务的搜索过程回顾时主题一致性高于参与者在使用 SearchLog 时的主题一致性。

从整体感受的专家评分可以看出，使用时间树时参与者的回顾过程带给专家的整体感受高于使用 SearchLog 时参与者的回顾过程带给专家的整体感受，说明专家认为在使用时间树时，相比于使用 SearchLog 时，参与者能够更准确地回顾试探型任务的

搜索过程并且进行更清晰的讲解。

与学习型任务相同，为了更准确地验证上述结论，在对专家评估结果的分析中，采用显著性水平α=0.05 的配对 T 检验对专家评估结果进行分析。配对 T 检验结果如表 3.9 所示。

表 3.9 试探型任务专家评估配对 T 检验结果

Table 3.9 expert evaluation paired T-test results of exploratory task

| 编号 | 评价因子 | 配对 T 检验 p-值 |
|---|---|---|
| 1 | 时间一致性 | **0.383** |
| 2 | 因果一致性 | **0.0555** |
| 3 | 主题一致性 | 0.00369 |
| 4 | 整体感受 | 0.00280 |

配对 T 检验结果验证了上文对试探型任务专家评估结果的分析。

在时间一致性与因果一致性两个方面，参与者在使用时间树与 SearchLog 进行试探型任务搜索过程回顾过程中获得的专家评分配对 T 检验 p-值大于显著性水平 0.05，说明在试探型任务中，从专家评估的角度，在帮助用户维护时间一致性与因果一致性方面，时间树与 SearchLog 的差异不是统计显著的。

而在主题一致性与整体感受两个方面的专家评分配对 T 检验 p-值小于显著性水平 0.05，说明试探型任务中，从专家评估的角度，在帮助参与者维护主题一致性以及更准确地回顾学习型任务的搜索过程并进行清晰的讲解方面，时间树与 SearchLog 的差异是统计显著的。

综合上述分析，在参与者对试探型任务进行回顾及讲解过程中，时间树与 SearchLog 都能够有效帮助用户维护时间一致性以及因果一致性。与使用 SearchLog 时相比，在使用时间树时，参与者能够统计显著地获得更高的主题一致性。并且使用时间树，参与者能够更准确地回顾试探型任务的搜索过程并进行更清晰地讲解。

### 3.4.3 客观评估分析

#### 3.4.3.1 因果一致性及主题一致性的行为分析

在 3.4.1 及 3.4.2 节对主观评估及专家评估结果的分析中，参与者及专家都认为时间树在帮助参与者维护因果一致性与主题一致性方面相对 SearchLog 的优势是统计显著的。这引出了一个问题，是什么导致了两种系统在帮助参与者维护因果一致性与主题一致性方面具有这种统计显著的差异。为了解释这种显著性差异产生的原因，本研究对参与者在使用时间树及 SearchLog 时体现因果一致性及主题一致性的行为进行了分析。

在本实验的第四阶段参与者对搜索过程进行回顾及讲解阶段，参与者被要求对搜

索任务进行了子任务划分，并且对随机抽取的 10 个查询关键词进行查询原因的解释。对于学习型任务及试探型任务，参与者使用时间树及 SearchLog 时，划分的子任务及给出明确查询原因的查询关键词的平均个数如表 3.10 所示。

表 3.10 因果一致性及主题一致性维护行为表

Table 3.10 maintenance behaviors for causal coherence and thematic coherence

| 一致性 | 学习型任务 | | 试探型任务 | |
|---|---|---|---|---|
| | 时间树 | SearchLog | 时间树 | SearchLog |
| 因果一致性：查询原因明确的查询关键词平均个数 | 8.06 | 8.13 | 8.44 | 8.00 |
| 主题一致性：子任务划分平均个数 | 4.00 | 3.88 | 2.50 | 2.25 |

结果显示，参与者对自己所进行的每一种任务，随机抽取的 10 个查询关键词中，能够给出明确查询原因的查询关键字个数以及划分的子任务平均个数几乎相同。

为了更为严谨地分析参与者使用使用时间树及 SearchLog 在各种任务上能够给出明确查询原因的查询关键字个数以及划分子任务平均个数的相似程度，这一部分分析采用了显著性水平$\alpha$=0.05 的 F 检验（F-test）与配对 T 检验两种假设检验方式对上述问题进行分析。F 检验与配对 T 检验的结果如表 3.11 所示。

表 3.11 因果一致性及主题一致性维护行为假设检验结果表

Table 3.11 hypothesis test result of maintenance behaviors for causal coherence and thematic coherence

| 一致性 | F 检验 p-值 | | 配对 T 检验 p-值 | |
|---|---|---|---|---|
| | 学习型 | 试探型 | 学习型 | 试探型 |
| 因果一致性：查询原因明确的查询关键词平均个数 | 0.978 | **0.0171** | 0.935 | 0.573 |
| 主题一致性：子任务划分平均个数 | 0.791 | 0.948 | 0.781 | 0.678 |

观察查询原因明确的查询关键词平均个数及子任务划分平均个数在时间树与 SearchLog 上的 F 检验 p-值以及配对 T 检验 p-值可以得出，参与者在各种任务上使用时间树以及 SearchLog 能够给出明确查询原因的查询关键字个数以及划分子任务平均个数不具有显著性差异。

#### 3.4.3.2 因果一致性及主题一致性的表现因子分析

上文中对参与者能够明确查询原因的查询关键字个数以及划分的子任务平均个数的分析结果似乎与 3.4.1 节主观评估分析以及 3.4.2 节专家评估分析中的结果相矛盾。由此引出一个问题，是什么导致了参与者在使用时间树对搜索任务进行回顾时在因果一致性与主题一致性方面能够获得比使用 SearchLog 更优的主观评估及专家评估结果，为了回答这一问题，本研究从参与者对搜索任务的回顾过程中提取了反映参与者因果一致性以及主题一致性的正面表现因子以及负面表现因子。

提取出的因果一致性正面表现因子及负面表现因子如下：

- 因果一致性正面表现因子：

- 清晰的来源描述，例如参与者这样的描述：“我读到了一篇……所以我搜索了……”

- 因果一致性负面表现因子：
  - 在来源描述中缺少确定性，例如参与者这样的描述：“应该是有一篇……里面说了……吧。”
  - 缺少查询来源的描述。
  - 在讲解中被问到关于查询来源的问题时，不能较好地回答。

提取出的主题一致性正面表现因子以及负面表现因子如下：

- 主题一致性正面表现因子：
  - 对查询和点击清晰的主题描述。

- 主题一致性负面表现因子：
  - 不清晰的主题描述，例如参与者的描述中带有“可能……”,“差不多……”,“发现了几篇相关文章……”（没有说明这些文章是关于什么的。）
  - 在讲解过程中没能较好地回答主题相关的问题。

在提取出因果一致性以及主题一致性的正面表现因子及负面表现因子后，针对每个任务，本研究对每位实验参与者对搜索过程的讲解中各表现因子的出现次数进行了统计。正面因子出现次数与负面因子出现次数的关系如图 3.3 所示。

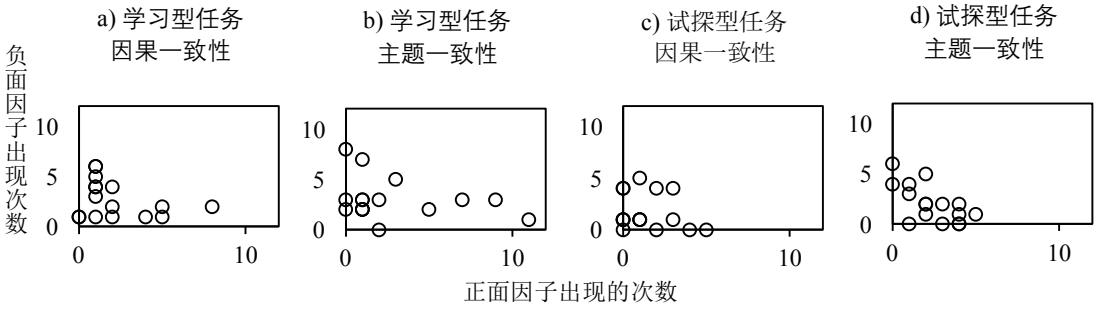


图 3.3 正/负面表现因子关系图

Fig. 3.3 relation diagram of positive and negative factors

图 3.3a～图 3.3d 分别表示学习型任务和试探型任务中因果一致性正负面表现因子关系以及主题一致性正负面表现因子关系。其中横坐标为实验参与者在对搜索任务进行回顾后的回想过程中正面因子出现的次数，纵坐标为这一过程中负面因子出现的次数。观察图中节点的分布可以看出，正面因子出现次数增多时，负面因子出现次数会减少。基于这一观察，本研究计算了正负面表现因子出现次数的差值，观察是否是正负面表现因子导致了专家认为参与者在对搜索任务进行的回顾后讲解过程中对因果一致性以及主题一致性的维护表现的不同。

下面将分别针对学习型任务以及试探型任务对因果一致性及主题一致性的正负表现因子出现次数差值与专家评分的关系进行分析。

(1) 学习型任务

学习型任务的正负表现因子出现次数差与专家评分的关系如图 3.3 所示。

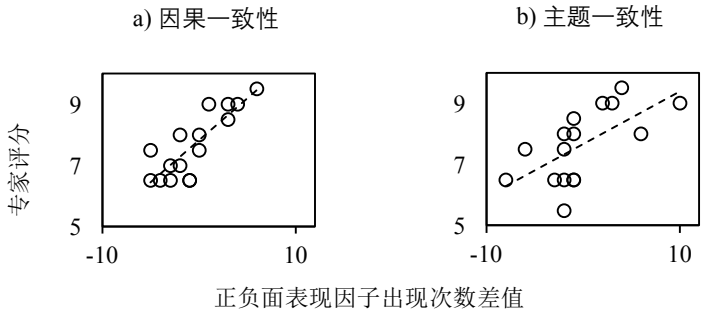


图 3.4 学习型任务正负面表现因子出现次数差值与专家评分关系图

Fig. 3.4 relation diagram of difference between positive and negative factors and expert evaluation result on learning tasks

图中横坐标为参与者在对学习型任务进行回顾后讲解过程中正负面表现因子出现次数的差值，纵坐标为专家评分。从图中观察到，在学习型任务中，正负面表现因子的出现次数差值与专家评分呈线形相关。为了验证这一观察，本研究采用显著性水平为 0.05 的皮尔逊相关检测（Pearson’s correlation test）对实验参与者在对学习型任务的回顾后讲解过程中，正负面表现因子出现次数的差值与专家评分之间的相关性进行检测。检测结果如表 3.12 所示。

表 3.12 学习型任务正负表现因子出现次数差值与专家评分皮尔逊相关性检测结果

Table 3.12 pearson test result of difference between positive and negative factors and expert evaluation result on learning tasks

| 一致性 | 相关系数 | p-值 |
|---|---|---|
| 因果一致性 | 0.752 | 7.85e-4 |
| 主题一致性 | 0.525 | 0.0367 |

结果表明，在在学习型任务中，因果一致性与主题一致性的正负表现因子出现次数差值与专家评分的相关系数较高，并且 p-值均小于 0.05。

根据上述分析可以得出结论，学习型任务中，因果一致性与主题一致性的正负面表现因子出现次数差值都与专家评分具有统计显著的线性相关关系。

(2) 试探型任务

试探型任务的正负表现因子出现次数差与专家评分的关系如图 3.5 所示。

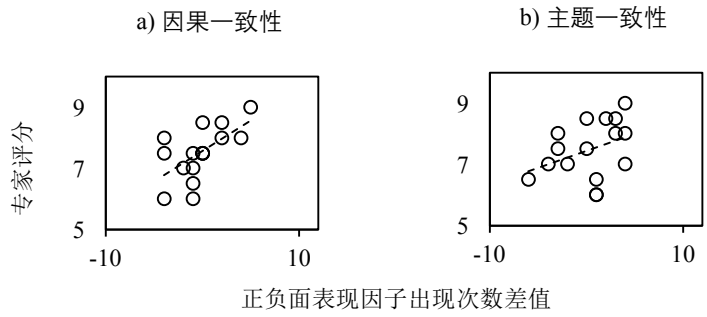


图 3.5 试探型任务正负面表现因子出现次数差值与专家评分关系图

Fig. 3.5 relation diagram of difference between positive and negative factors and expert evaluation result on exploratory tasks

图中横坐标为参与者在对试探型任务进行回顾后讲解过程中正负面表现因子出现次数的差值，纵坐标为专家评分。与学习型任务中一样，观察图中节点分布，发现在试探型任务中，正负面表现因子的出现次数差值似乎也与专家评分呈线形相关。为了验证这一观察，同样采用显著性水平为 0.05 的皮尔逊相关检测对实验参与者在对试探型任务的回顾后讲解过程中，正负面表现因子出现次数的差值与专家评分之间的相关性进行检测。检测结果如表 3.13 所示。

表 3.13 试探型任务正负表现因子出现次数差值与专家评分皮尔逊相关性检测结果

Table 3.13 pearson test result of difference between positive and negative factors and expert evaluation result on exploratory tasks

| 一致性 | 相关系数 | p-值 |
|---|---|---|
| 因果一致性 | 0.481 | **0.0593** |
| 主题一致性 | 0.414 | **0.111** |

从表中可以看出，检测结果与学习型任务并不相同。

试探型任务中，因果一致性与主题一致性的正负表现因子出现次数差值与专家评分的相关系数分别为 0.481 与 0.414，说明它们之间具有一定的线性相关性。但 p-值分别为 0.0593 与 0.111，均大于显著性水平 0.05，说明这种相关性并不是显著的。

据此可以得出，试探型任务中，因果一致性与主题一致性的正负面表现因子出现次数差值与专家评分并不具有统计显著的线性相关关系。

综合上述分析，学习型任务中因果一致性与主题一致性的正负面表现因子出现次数差值与专家评分之间的线性相关关系是统计显著的，而试探型任务中它们之间的线性相关关系并不是统计显著的。

本章研究的目的是考察参与者在使用时间树进行复杂搜索过程的回顾过程中的一致性表现。在主观评估及专家评估中，时间树在维护因果一致性及主题一致性方面的表现均优于 SearchLog，因此本研究在分析了正负表现因子出现次数差值与专家评分之间的关系的基础上，进一步考察了正负表现因子出现次数差在时间树与 SearchLog 之

间的不同统计规律，分别针对学习型任务与试探型任务的因果一致性与主题一致性计算了参与者使用时间树与 SearchLog 时的正负表现因子出现次数的平均差值，并对参与者使用时间树与 SearchLog 时的差值进行了显著性水平α=0.05 的配对 T 检验，结果如表 3.14 所示。

表 3.14 时间树与 SearchLog 正负表现因子出现次数平均差值及配对 T 检验结果表

Table 3.14 average difference between positive and negative factors and paired T-test result of TimeTree and SearchLog

| 一致性及任务类型 | 正负表现因子出现次数平均差值 | | 配对 T 检验 p-值 |
|---|---|---|---|
| | 时间树 | SearchLog | |
| 因果一致性，学习型任务 | 1.13 | -2.25 | 0.0230 |
| 因果一致性，试探型任务 | 0.125 | -0.750 | **0.277** |
| 主题一致性，学习型任务 | 1.13 | -1.63 | 0.0106 |
| 主题一致性，试探型任务 | 0.500 | 0.125 | **0.528** |

结果表明，在学习型任务中，参与者使用时间树与 SearchLog 时的因果一致性与主题一致性正负表现因子出现次数差值具有显著性差异。同时从结果中也可以看出，使用时间树时因果一致性与主题一致性正负表现因子出现次数平均差值高于使用 SearchLog 时的平均差值。根据上述结果，本研究认为，参与者使用时间树进行搜索任务回顾后的讲解时，正面表现因子出现因子出现次数更多，负面表现因子出现次数更少，从而导致了参与者在主观评估及专家评估中得到更优的结果。

### 3.4.4 实验分析结论

从对实验结果的主观评估分析、专家评估分析以及客观评估分析可以得出结论：

(1) 学习型任务

在学习型任务中，时间树在主观评估以及专家评估比 SearchLog 得到了更高的评分，并且在客观评估分析中有效地分析了评分更高的原因，证明了时间树在参与者进行学习型任务时有效地帮助参与者维护了时间一致性、因果一致性以及主题一致性，其中因果一致性以及主题一致性相对于 SearchLog 的维护效果更为明显。

(2) 试探型任务

在试探型任务中，时间树在主观评估以及专家评估中也获得了比 SearchLog 更高的评分，但对客观评估的分析中，没有有效地分析出评分更高的原因，没能完整地证明主观评估以及专家评估的有效性。因此得出结论，时间树在参与者进行试探性任务时，有可能能够帮助参与者维护搜索经验一致性。

综上所述，参与者使用时间树进行复杂搜索任务过程中，能够有效帮助参与者维护搜索经验一致性，说明时间树中蕴含着参与者进行复杂搜索任务产生的高质量的搜

索经验。

## 3.5 基于时间树的搜索经验提取方法

### 3.5.1 搜索经验提取过程

时间树中蕴含了高质量的因果经验及主题经验，其中因果经验体现在用户的查询-点击-查询序列中，主题经验体现在用户对搜索任务进行的子任务划分中。因此，本研究提出基于时间树的搜索经验提取方法。该方法分别针对用户在复杂搜索过程中的因果经验以及主题经验进行提取。搜索经验提取过程如图 3.6 所示。

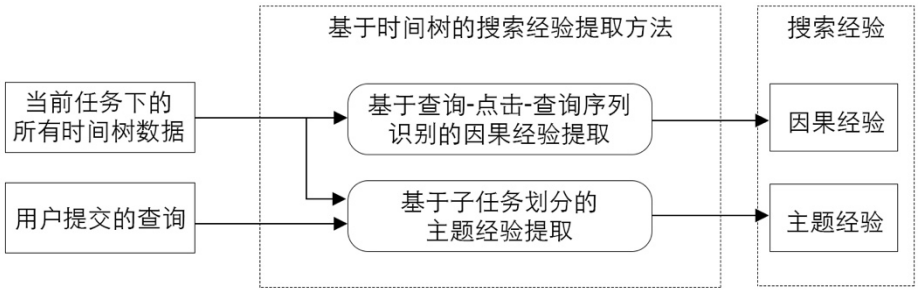


图 3.6 搜索经验提取过程图

Fig. 3.6 diagram of search experience extraction process

基于时间树的搜索经验提取方法包括基于查询-点击-查询序列识别的因果经验提取算法以及基于子任务划分的主题经验提取算法，两算法分别针对用户在复杂搜索过程中产生的因果经验以及主题经验进行提取。

其中基于查询-点击-查询序列识别的因果经验提取算法以当前任务所有用户的时间树数据作为输入，提取出用户在复杂搜索过程中产生的因果经验。

基于子任务划分的主题经验提取算法以用户提交的查询和当前任务所有用户的时间树数据作为输入，算法利用时间树的 RCST 结构特性，根据时间树上节点的相对位置关系，将时间树上的节点进行聚类，从而将节点划分为不同子任务，提取出用户主题经验。

下文将分别针对基于查询-点击-查询序列识别的因果经验提取算法以及基于子任务划分的主题经验提取算法进行介绍。

### 3.5.2 基于查询-点击-查询序列识别的因果经验提取算法

时间树作为一种树形结构，除根节点外，每一个节点都拥有且唯一拥有一个父节点。用户搜索经验中的因果经验就蕴含在每一个节点的唯一父节点中。

本研究在第三章中提出了时间树的 *RCST* 结构，在时间树的 *RCST* 结构中，每一个 *RCST* 节点都拥有一个 *RCST* 节点类型，*RCST* 节点类型的取值包括查询类型与点击类型。对于时间树上除根节点外的每一个查询节点，其父节点的 *RCST* 节点类型，都蕴含着用户进行该查询的原因。

对用户的复杂搜索过程进行观察，当用户在进行某一领域范围内的复杂搜索任务时，查询词的来源可以分为三种：来自任务描述、来自用户自身知识以及来自点击的内容，除以上三种来源以外，用户进行一个查询的原因还可能是查询重构。当时间树上一个查询节点的父节点是查询节点，则用户进行的该查询可以是从任务描述中得到启发、以自身的知识及对任务的理解进行组织或正在进行查询重构。但若一个查询节点的父节点是点击节点，则用户进行该查询一定是从点击的内容中获得了启发。

时间树中的查询-查询序列中所蕴含的因果经验受到过多用户自身因素的影响而不易于提取，并且带有过多用户个性化因素的搜索经验对后期查询推荐的意义并不大，而查询-点击-查询序列所蕴含的时间经验清晰明确，即用户阅读了点击的内容，受到了启发，进行了查询。因此本节中对时间树中蕴含的用户复杂搜索因果经验的提取，主要通过提取时间树中的查询-点击-查询序列来实现。

提取时间树中所有的查询-点击-查询序列，首先需要对时间树中所有有边连接的节点序列进行遍历。对节点序列的遍历可以通过对节点的遍历来实现。树形结构中节点的遍历主要有两种方式，一种是深度优先遍历，另一种是广度优先遍历。本节提取算法采用广度优先遍历的方式对时间树中的所有 *RCST* 节点进行遍历，以提取所有查询-点击-查询序列的方式提取时间树中蕴含的用户进行复杂搜索过程中的因果经验。

基于查询-点击-查询序列识别的因果经验提取算法如表 3.15 所示。

表3.15 基于查询-点击-查询序列识别的因果经验提取算法

Table 3.15 The algorithm of causal experience extraction based on query-click-query sequence recognition

**算法3.1** 基于查询-点击-查询序列识别的因果经验提取算法

算法输入：时间树根节点 *Root*

算法输出：查询-点击-查询节点序列集合

```
1.  Begin
2.    Queue.add (Root)
3.    While (Queue is not empty)
4.      CurrentNode ← Queue.poll()
5.      If (CurrentNode.type  is Click Type)
6.        For each  Child  ∈ CurrentNode.children
7.          addToSequenceSet (CurrentNode.father, CurrentNode,  Child)
```

续表3.15基于查询-点击-查询序列识别的因果经验提取算法

Table 3.15 The algorithm of causal experience extraction based on query-click-query sequence recognition

**算法3.1** 基于查询-点击-查询序列识别的因果经验提取算法

```
8.            End For
9.          End If
10.         For each Child ∈ CurrentNode.children
11.           Queue.add (Child)
12.         End For
13.       End While
14.   End
```

遍历过程使用一个缓存队列存储当前层次及下一层次的 *RCST* 节点。算法的开始，将时间树的根节点加入缓存队列，然后算法进入以缓存队列不是空队列为条件的循环，每次循环首先从缓存队列中有一个 *RCST* 节点出队并将该出队的 *RCST* 节点的所有子节点加入缓存队列。如果该出队的 *RCST* 节点的节点类型是点击节点，还要将该点击节点的父节点和其本身以及其所有子节点组成的查询-点击-查询节点序列加入查询-点击-查询节点序列集合。

在得到时间树上所有查询-点击-查询节点序列的集合后，对于集合中所有查询-点击-查询节点序列中的后一查询节点所代表的查询，查询原因都是受到了该序列中的点击节点所代表的点击的内容的启发。至此，时间树中蕴含的用户在复杂搜索过程中的因果经验被成功提取。

### 3.5.3 基于子任务划分的主题经验提取算法

用户复杂搜索过程中的主题经验为用户对复杂搜索任务所进行的子任务划分。第三章中讨论了时间树对用户的主题经验的蕴含主要体现在时间树的 *RCST* 结构造成了某些节点在时间树上的视觉距离非常大，从而方便用户使用时间树进行子任务划分。本节将要解决的问题是，怎样利用时间树的 *RCST* 结构的这一特性将时间树上的 *RCST* 节点划分为不同子任务，为后文提取主题经验提供基础。

为解决上述问题，本研究提出基于子任务划分的主题经验提取算法，通过对时间树上的 *RCST* 节点进行基于节点位置及关系的聚类，进行子任务的划分，提取用户主题经验。

主题经验提取算法如表 3.16 所示。

表3.16 基于子任务划分的主题经验提取算法

Table 3.16 The algorithm of thematic experience extraction based on subtask partition

**算法3.2** 基于子任务划分的主题经验提取算法

算法输入：时间树 *RCST* 结构数据

算法输出：主题经验（时间树*RCST*节点聚类结果）

```
1.  Begin
2.    For each RCSTNode
3.      setCoordinate (RCSTNode, a)
4.      setRectangle (RCSTNode, 2a)
5.    End For
6.    For each (RelatedNodeA, RelatedNodeB) ∈ RCSTNodeRelationSet
7.      checkRange (RelatedNodeA, RelatedNodeB)
8.    End For
9.  End
```

首先，根据 RCST 节点关系集中的父子关系及 *RCST* 节点产生的时间顺序，将时间树绘制为网格上的点，其中垂直方向与水平方向上每一个节点层次之间的距离都为单位网格的边长。这样，以根节点坐标为(0,0)，时间树中每一个 *RCST* 节点都将拥有一个坐标。在得到所有 *RCST* 节点的坐标后，算法以每一个节点为中心，以 2 倍单位网格边长为边长绘制正方形，称为范围正方形。这样，每一个 *RCST* 节点将获得边长的 4 个顶点坐标，4 个顶点坐标可以表示为一个2×2的坐标矩阵，称为范围矩阵。在得到每个 *RCST* 节点的范围矩阵以后，便可以对 *RCST* 节点关系集中具有父子关系或兄弟关系的节点的范围矩阵进行两两比较，从而确定具有父子关系或兄弟关系的节点的范围正方形是否重合，并将范围正方形有重合的节点聚为一类。聚类将代表整个复杂搜索任务的时间树划分为若干棵子树，每一棵子树代表着该复杂搜索任务的一个子任务。子任务划分算法的示意图如图 3.7 所示。

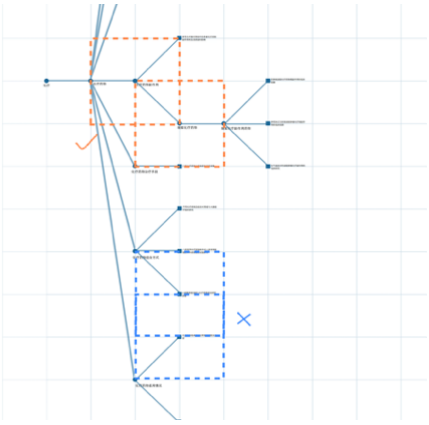

图 3.7 基于子任务划分的主题经验提取算法示意图

Fig. 3.7 diagram of thematic experience extraction algorithm based on subtask partition

如图中所示，当两个 *RCST* 节点的节点关系为父子关系或兄弟关系，并且它们的范围正方形有重合时，这两个 *RCST* 节点将被聚为一类，即被划分到同一个子任务中；但当两个 *RCST* 节点的节点关系为其他关系时，即使它们的范围正方形有重合，这两个 *RCST* 节点也不会被划分为同一个子任务。

## 3.6 基于子任务划分的主题经验提取算法对比实验

本章中提出了用户因果经验提取算法以及用户主题经验提取算法。其中用户因果经验是以往基于时间线性日志的查询推荐所做不到的，因此其有效性显而易见。而用户主题经验的提取在非时间树上有实现的可能。因此，本节将设计对比实验，用以对本文提出的用户主题经验提取算法作出评价。

### 3.6.1 实验设计

本实验针对基于子任务划分的主题经验提取算法设计对比算法对第三章实验中用户的搜索过程数据进行子任务划分。并设计评价标准，以考察本研究所提出的主题经验提取算法的质量。

### 3.6.2 对比算法

用户主题经验提取算法基于时间树的 *RCST* 结构特性，通过对时间树上的节点进行聚类来提取用户的主题经验。在时间线形日志中并不具有时间树所具有的结构特性，因此，对比算法采用基于文本聚类的子任务划分算法，提取用户的主题经验。

基于文本聚类的子任务划分算法首先将用户的所有查询做分词处理，并做 *TF-IDF* 处理形成词向量，然后对词向量使用 *K-Means* 算法进行聚类，以此形成子任务的划分。在使用 *K-Means* 算法进行聚类时，对于每一份时间树数据，均采用从 $K = 2$ 至 $K = 10$ 的聚类方法，并从中选取聚类效果最佳的 K 值聚类结果作为该时间树数据的最终对比实验结果，以保证实验的客观性及准确性。

### 3.6.3 评价标准

在传统的聚类评价方法中，主要分为两种：基于聚类结果中点与点间距离的判定以及基于聚类结果与人工判定结果吻合程度的判定。本研究中使用的聚类算法并非传统的文本聚类方法，因此基于聚类结果中点与点间距离的判定方法并不适用，因此，本实验的聚类评价标准采用基于聚类结果与人工判定结果吻合程度的判定标准，验证基于时间树的主题经验提取算法以及基于传统文本聚类算法的主题经验提取算法的质量高低。

本实验采用子任务划分的平均准确率作为算法质量的评价标准。基于人工判定的方式要求在聚类语料库中存在用于人工判定的类结构 *A*，依据此类结构 *A*，通过判定聚类结果中任意两个文档在该类结构 *A* 中以及聚类结果 *B* 中是否属于同一个簇，得到 4 种可能结果，如图 3.8 所示。

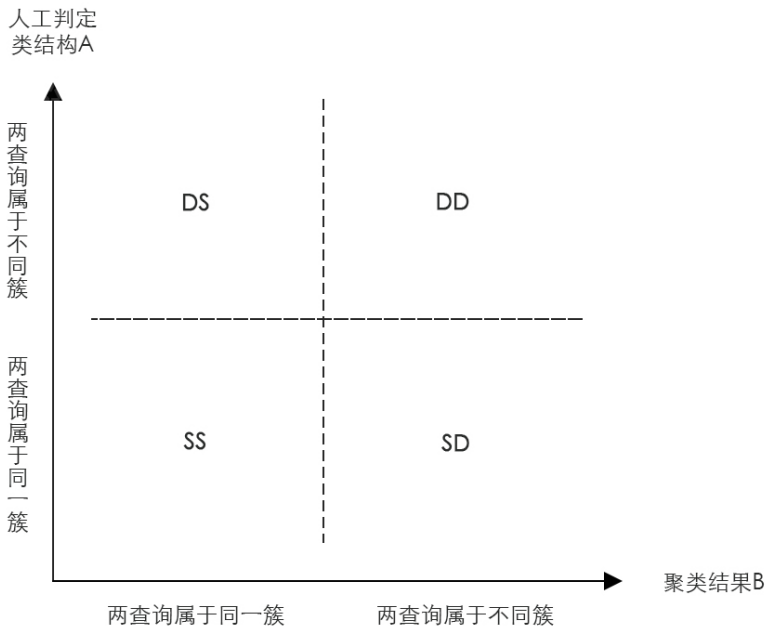


图 3.8 准确率计算方法示意图

Fig. 3.8 diagram of accuracy calculation method

在两种算法划分了子任务以后，通过两两检查所有查询的划分结果，计算属于同一子任务的两查询被成功划分入同一子任务与没有被成功划分入同一子任务的查询对个数，以及不属于同一子任务的两查询没有被划分入同一子任务与被误划分入同一子任务的查询对个数，考察算法对子任务划分的准确性。

在分别得到属于同一子任务的两查询被成功划分入同一子任务的情况个数 *a*、属于同一子任务的两查询被划分入不同子任务的情况个数 *b*、不属于同一子任务的两查询被划分入同一子任务的情况个数 *c* 以及不属于同一子任务的两查询被划分入不同子任务的情况个数 *d* 以后，可以分别计算得到积极准确率以及消极准确率并在此基础上得到平均准确率，其中积极准确率 *AP*、消极准确率 *AN* 以及平均准确率 *AA* 的计算公式分别为公式 3.1、公式 3.2 以及公式 3.3。

$$AP = \frac{a}{(a+b)} \tag{3.1}$$

$$AN = \frac{d}{(c+d)} \tag{3.2}$$

$$AA = \frac{AP + AN}{2} \tag{3.3}$$

在第三章实验中，参与者的搜索过程都根据任务描述进行，并且在实验过程中，要求参与者对其所进行的复杂搜索任务进行了子任务划分。因此，人工判定类结构 $A$ 根据任务描述以及参与者对子任务进行的划分结果定义如下：

(1) 任务一人工判定类结构包括：总述、常用药物、药物机理、使用方法、适用情况、副作用、治疗手段、组合方式、辅助药物及其他内容。

(2) 任务二人工判定类结构包括：总述、概念、来源、危害及危害原因、中国 pm2.5、严重城市及形成原因、降低 pm2.5 的方法以及治理可行性。

(3) 任务三人工判定类结构包括：web video service，web viedo charge，video category，user behavior，video quality，web video safety 以及 web video revisit。

(4) 任务四人工判定类结构包括：video chat content，online chat misbehavoir 以及 online chat algorithms。

为了避免人工判定类结构分类粒度过粗导致实验结果不准确，本实验人工判定类结构划分尽量以细粒度进行。并且本实验评价标准采取面向同类划分的乐观评价，即在获取人工判定结果时，文档可被划分入多个类时，在每个类中都添加该文档，即认为该文档属于可被划分的每一个类。

### 3.6.4 结果分析

在定义类评价标准后，将第三章实验中用户的搜索过程数据分别使用本研究提出的基于子任务划分的主题经验提取算法以及对比算法进行计算。接下来将分别针对学习型任务和试探性任务进行实验结果的分析。

(1) 学习型任务

学习型任务实验结果如表 3.17 所示。

表 3.17 学习型任务主题经验提取实验结果表

Table 3.17 results of thematic experience extraction experiment on learning task

| 参与者编号 | 基于子任务划分的主题经验提取算法 | | | | 基于 K-Means 文本聚类的主题经验提取算法（对比算法） | | | |
|---|---|---|---|---|---|---|---|---|
| | 聚类个数 | 积极准确率 | 消极准确率 | 平均准确率 | 聚类个数 | 积极准确率 | 消极准确率 | 平均准确率 |
| 1 | 6 | 0.732 | 0.743 | 0.737 | 7 | 0.675 | 0.677 | 0.676 |
| | | | | | 6 | 0.444 | 0.647 | 0.545 |
| 2 | 3 | 0.469 | 0.964 | 0.717 | 3 | 0.515 | 0.889 | 0.702 |
| | | | | | 3 | 0.515 | 0.889 | 0.702 |
| 3 | 7 | 0.250 | 0.927 | **0.588** | 10 | 0.333 | 0.929 | **0.631** |
| | | | | | 7 | 0.143 | 0.921 | 0.532 |
| 4 | 6 | 0.543 | 0.872 | 0.708 | 10 | 0.529 | 0.757 | 0.643 |
| | | | | | 6 | 0.323 | 0.745 | 0.534 |

续表 3.17 学习型任务主题经验提取实验结果表

Table 3.17 results of thematic experience extraction experiment on learning task

| 参与者编号 | 基于子任务划分的主题经验提取算法 | | | | 基于 K-Means 文本聚类的主题经验提取算法（对比算法） | | | |
|---|---|---|---|---|---|---|---|---|
| | 聚类个数 | 积极准确率 | 消极准确率 | 平均准确率 | 聚类个数 | 积极准确率 | 消极准确率 | 平均准确率 |
| 5 | 3 | 0.551 | 0.936 | 0.744 | 10 | 0.523 | 0.762 | 0.643 |
| | | | | | 3 | 0.252 | 0.722 | 0.487 |
| 6 | 6 | 0.513 | 0.969 | **0.741** | 10 | 0.645 | 0.963 | **0.804** |
| | | | | | 6 | 0.450 | 0.953 | 0.702 |
| 7 | 10 | 0.522 | 0.762 | 0.642 | 9 | 0.344 | 0.725 | 0.535 |
| | | | | | 10 | 0.324 | 0.723 | 0.523 |
| 8 | 5 | 0.373 | 0.825 | 0.599 | 9 | 0.343 | 0.773 | 0.558 |
| | | | | | 5 | 0.295 | 0.775 | 0.535 |

对于学习型任务，在使用本文提出的基于子任务划分的主题经验提取算法提取主题经验即对用户的复杂搜索过程进行子任务划分的 8 组实验结果中，有 6 组实验的平均准确率高于使用 *K-Means* 聚类进行子任务划分的最高平均准确率，有 2 组实验的平均准确率低于使用 *K-Means* 聚类进行子任务划分的最高平均准确率。

从结果中可以看出，在多数情况下，本研究提出的基于子任务划分的主题经验提取算法能够有效对用户所进行的复杂搜素任务进行子任务划分，从而提取用户主题经验。

在使用 *K-Means* 聚类进行子任务划分时，对于每组对比实验，采用了从 $K=2$ 到 $K=10$ 共 9 种参数的 *K-Means* 算法，对每个参与者进行的复杂搜索任务划分出 2 至 10 个子任务，并从中取出平均准确率最高的结果作为对比算法的最终结果。表 3.18 中，每一位参与者使用 *K-Means* 文本聚类算法进行的子任务划分结果分为 2 行，第一行为平均准确率最高的划分结果，第二行为划分子任务数等于使用基于时间树结构的主题经验提取算法所划分的子任务数的结果。从结果中可以看出，在确定 *K-Means* 聚类个数的情况下，使用本研究提出的基于子任务划分的主题经验提取算法得到结果的平均准确率均高于 *K-Means* 算法。

对少数情况下使用本研究提出的基于子任务划分的主题经验提取算法进行主题经验提取的效果不如 *K-Means* 算法的主要原因进行分析，发现 2 次实验所使用的时间树中参与者进行复杂搜索的主题分布离散，致使主题经验提取算法实效。这主要是因为参与者没有合理使用维护时间树，致使时间树中不同主题在局部上出现了线性衔接的情况，这样的局部线性衔接一旦超过一定数量，就会导致时间树上主题分布离散凌乱。

(2) 试探型任务

试探型任务实验结果如表 3.18 所示。

表 3.18 试探型任务主题经验提取实验结果表

Table 3.18 results of thematic experience extraction experiment on exploratory task

| 参与者编号 | 基于子任务划分的主题经验提取算法 | | | | 基于 K-Means 文本聚类的主题经验提取算法（对比算法） | | | |
|---|---|---|---|---|---|---|---|---|
| | 聚类个数 | 积极准确率 | 消极准确率 | 平均准确率 | 聚类个数 | 积极准确率 | 消极准确率 | 平均准确率 |
| 1 | 4 | 0.543 | 0.380 | **0.461** | 5 | 0.729 | 0.355 | **0.542** |
| | | | | | 4 | 0.660 | 0.337 | 0.499 |
| 2 | 3 | 0.404 | 0.821 | **0.613** | 7 | 0.614 | 0.778 | **0.696** |
| | | | | | 3 | 0.338 | 0.758 | 0.548 |
| 3 | 3 | 0.656 | 0.848 | 0.752 | 5 | 0.667 | 0.789 | 0.728 |
| | | | | | 3 | 0.385 | 0.721 | 0.553 |
| 4 | 1 | 0.356 | NaN | **0.356** | 5 | 0.800 | 0.613 | **0.706** |
| | | | | | 1 | - | - | - |
| 5 | 2 | 0.327 | 1.000 | **0.663** | 4 | 0.458 | 0.938 | **0.698** |
| | | | | | 2 | 0.204 | 0.893 | 0.548 |
| 6 | 7 | 0.294 | 0.568 | **0.431** | 10 | 0.778 | 0.625 | **0.701** |
| | | | | | 7 | 0.700 | 0.621 | 0.661 |
| 7 | 1 | 0.426 | NaN | **0.426** | 10 | 0.765 | 0.534 | **0.649** |
| | | | | | 1 | - | - | - |
| 8 | 3 | 0.263 | 0.719 | **0.491** | 7 | 0.761 | 0.366 | **0.563** |
| | | | | | 3 | 0.640 | 0.342 | 0.491 |

对于试探型任务，在 8 组实验结果中，有 2 组失效结果，即主题经验提取失败。是因为基于位置进行聚类的方式将参与者进行的所有查询都归为一类，致使子任务划分失败。在 6 组有效结果中，只有一组使用基于子任务划分的主题经验提取算法进行搜索经验提取的平均准确率高于对比算法 *K-Means* 的最高平均准确率。虽然 6 组有效结果仍然高于同聚类个数下的 *K-Means* 算法，但该结果仍然说明在试探型任务中，本研究提出的基于子任务划分的主题经验提取算法提取主题经验的效果并不理想。

基于子任务划分的主题经验提取算法对试探型任务进行主题经验提取效果不理想的原因是，试探型任务中参与者对其搜索的领域知识匮乏，难以对查询进行有效组织，因此出现主题经验蕴含性不显著的情况。

### 3.6.5 实验结论

基于上述对实验结果的分析可以得出结论，对于学习型任务，基于子任务划分的主题经验提取算法能够有效地提取出主题经验，但对于试探型任务，基于子任务划分

的主题经验提取算法对主题经验的提取效果不理想，在时间树维护状况极短不理想情况下有失效的可能。

## 3.7 结论

本章介绍了时间树的 *RCST* 结构，并提出了用户在进行复杂搜索过程中的搜索经验模型及搜索经验一致性模型，进而设计搜索经验一致性验证实验验证了时间树的 *RCST* 结构中高质量地蕴含了搜索经验模型中的因果经验以及主题经验，并在此基础上提出了用户搜索经验的提取算法并对该算法的有效性进行了实验验证。

# 第4章 面向搜索经验的查询推荐及查询推荐可视化方法

本研究在第三章中提出的搜索经验模型指出搜索经验包括时间经验，因果经验及主题经验；验证了时间树帮助用户管理的搜索经验能够维持较高的因果一致性及主题一致性；证明了时间树帮助用户高质量地管理了因果经验以及主题经验。并在此基础上提出了基于时间树的因果经验以及主题经验提取方法。本章将研究如何利用提取出的因果经验及主题经验进行查询推荐。

## 4.1 面向搜索经验的查询推荐方法

本研究在第三章中分别针对用户使用时间树进行复杂搜索过程中的因果经验以及主题经验进行了提取。本节将提出面向搜索经验的查询推荐方法，利用提取出的搜索经验，结合用户提交的查询关键词，进行查询推荐。

### 4.1.1 面向搜索经验的查询推荐过程

面向搜索经验的查询推荐方法分别利用提取出的用户在复杂搜索过程中产生的因果经验以及主题经验进行查询推荐，分为面向因果经验的查询推荐方法、子任务内部的查询推荐方法以及跨子任务的查询推荐方法。本节首先介绍面向搜索经验的查询推荐过程，对面向搜索经验的查询推荐方法进行过程性的概述。

为了利用提取出的因果经验进行查询推荐，首先需要将用户提交的查询与一提取出的因果经验即查询-点击-查询序列进行匹配，再提取出匹配度高的查询-点击-查询序列作为面向因果经验的查询推荐结果。

第三章中基于子任务划分的主题经验提取算法所提取的主题经验以子任务子树的形式存在，为了利用提取出的主题经验进行查询推荐，首先需要将提取出的不同用户的主题经验集合合并为总体的主题经验集合。然后分别进行子任务内部以及跨子任务的查询推荐。面向搜索经验的查询推荐过程如图 4.1 所示。

面向搜索经验的查询推荐方法分为面向因果经验的查询推荐方法、子任务内部的查询推荐方法与跨子任务的查询推荐方法。

对于因果经验，面向因果经验的查询推荐方法以提取出的因果经验集合以及用户提交的查询为输入，通过因果经验匹配算法以及因果经验匹配度排名及筛选方法计算得到面向因果经验的查询推荐结果。

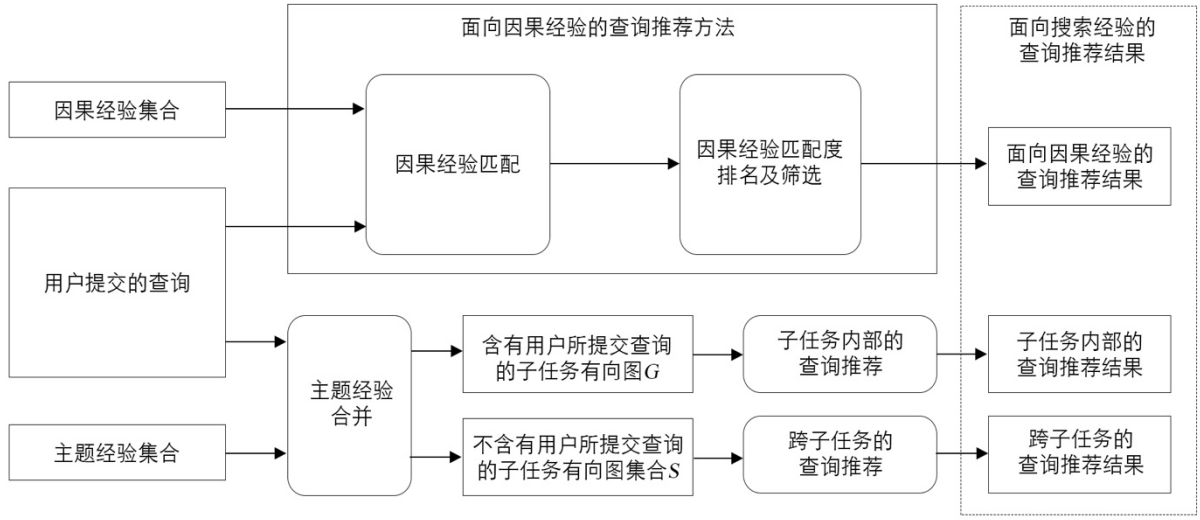


图 4.1 面向搜索经验的查询推荐过程图

Fig. 4.1 diagram of search experience oriented query recommendation process

对于主题经验，在使用子任务内部的查询推荐方法以及跨子任务的查询推荐方法针对提取出的主题经验进行查询推荐之前，首先需要将当前任务下所有用户的主题经验集合进行合并。因此，以用户提交的查询以及主题经验集合作为输入，主题经验合并算法计算出含有用户所提交查询的子任务有向图 $G$ 以及不含邮用户所提交查询的子任务有向图集合 $S$，再分别以 $G$ 和 $S$ 作为输入，通过子任务内部的查询推荐方法以及跨子任务的查询推荐方法计算得到子任务内部的查询推荐结果以及跨子任务的查询推荐结果。

### 4.1.2 面向因果经验的查询推荐方法

第三章中提出的基于查询-点击-查询序列识别的因果经验提取算法提取出的因果经验以查询-点击-查询序列集合的形式呈现。本节利用提取出的因果经验进行查询推荐，面向因果经验的查询推荐方法分为因果经验匹配算法以及因果经验匹配度排名及筛选方法两部分。

(1) 因果经验匹配算法

首先使用因果经验匹配算法计算查询关键词的匹配度。当用户提交了一个查询 $Q$ 以后，使用 $Q$ 与查询-点击-查询序列集中所有查询-点击-查询序列的起始查询进行匹配度计算，这样，查询-点击-查询序列集中的所有查询-点击-查询序列都将得到一个与用户提交的查询 $Q$ 的匹配度。因果经验匹配算法如表 4.1 所示。

表4.1 因果经验匹配算法

Table 4.1 The algorithm of causal experience matching

**算法4.1** 因果经验匹配算法

算法输入：用户提交的查询 $Q$，查询-点击-查询序列集合 $QCQSet$

算法输出：主题经验（时间树$RCST$节点聚类结果）

```
1.  Begin
2.    subTextSetQ ← segmentation(Q)
3.    For each QCQ ∈ QCQSet
4.      firstQuery ← QCQ.firstQuery()
5.      subTextSetX ← segmentation(firstQuery)
6.      QCQ.matchDegree ← computeMatchDegree(subTextSetQ, subTextSetX)
7.    End For
8.  End
```

其中，匹配度的计算以用户提交的查询 $Q$ 为基准。首先对 $Q$ 及查询-点击-查询序列集合中每个查询-点击-查询序列的前置查询进行分词以及去停止词处理，得到 $Q$ 及该前置查询的分词集。然后将这两个分词集进行比较，每匹配到一个分词，匹配度加 $1/n$。以此得到一个查询与用户提交的查询 $Q$ 的匹配度。

(2) 因果经验匹配度排名及筛选方法

在每一个查询-点击-查询序列得到与用户提交的查询 $Q$ 的匹配度以后，将这些查询-点击-查询序列按照匹配度排名。排名靠前的查询-点击-查询序列将按照设置的匹配度阈值参数α以及推荐个数阈值参数 c 进行筛选，本研究中设置参数$\alpha = 0.5$，$c = 3$。当匹配度最高的 c 个查询-点击-查询序列的匹配度均高于参数α时，则推荐该 c 个查询-点击-查询的后置查询给用户，当所有匹配度中有高于参数α但个数不超过 c 时，则向用户推荐所有匹配度高于参数α的查询-点击-查询序列的后置查询，当所有匹配度都低于参数α时，则推荐匹配度最高的一个查询-点击-查询序列的后置查询给用户。

### 4.1.3 主题经验合并算法

在第三章中用户的搜索经验模型的定义中，主题经验被定义为用户对复杂搜索任务进行的子任务划分，第三章中提出的基于子任务划分的主题经验提取算法可以提取用户复杂搜索过程中的主题经验，主题经验表现为复杂搜索任务的时间树子任务子树集合的形式。在针对每一个用户提取出主题经验以后，需要解决如何将多个用户的子任务合并为针对一个复杂搜索任务的整体的主题经验的问题。本节提出主题经验合并算法解决这一问题。

主题经验合并算法如表 4.2 所示。

表4.2 主题经验合并算法

Table 4.2 The algorithm of thematic experience combining

**算法4.2** 主题经验合并算法

算法输入：种子关键词 $Q$，已提取的主题经验 *TimeTreeSet*

算法输出：领域整体主题经验

```
1.  Begin
2.    SubTaskCount ← 0
3.    For each TimeTree ∈ TimeTreeSet
4.      SubTaskCount ← SubTaskCount + TimeTree.subTaskNumber
5.      For each SubTaskTree ∈ TimeTree
6.        If Q is involved in SubTaskTree
7.          IncludedSet.add(Q)
8.        Else
9.          ExcludedSet.add(Q)
10.       End If
11.     End For
12.   End For
13.   n ← SubTaskCount / TimeTreeSet.size
14.   combineTreesToGraph (IncludedSet,1)
15.   combineTreesToGraph (ExcludedSet, n-1)
16. End
```

给定种子关键词 $Q$，首先遍历时间树集合 *TimeTreeSet* 中的每一棵时间树，计算出每一棵时间树上划分子任务的平均数量 $n$，并且将所有时间树上包含关键词 $Q$ 的子任务子树放入集合 *IncludedSet* 中；将所有不包含关键词 $Q$ 的子任务子树放入集合 *ExcludedSet* 中。然后将 *IncludedSet* 中的所有子任务子树合并为 1 张有向图 *IncludedGraph*，将 *ExcludedSet* 中的所有子任务子树合并为 $n$-1 张有向图 *ExcludedGraph*[1]～*ExcludedGraph* [$n$-1]。

其中，将含有种子关键词 Q 的子任务子树合并为一张有向图的过程使用基于文本匹配的时间树合并算法完成，算法如表 4.3 所示。

算法需要维护一个已合并节点集合，遍历所有包含种子关键词 $Q$ 的子任务子树上的每一个查询节点，对于每一个查询节点 $Q_i$，如果已合并节点集合中有与该查询关键词相同的查询节点 $Q_j$，则将该查询节点所具有的父子关系集$\{R(Q_i, M),R(N, Q_i)\}$加入 $Q_j$ 中，在将 R 加入 $Q_j$ 中时，采用相同方式将关系集$\{R(Q_i, M),R(N, Q_i)\}$中与 $Q_i$ 相关的查询或点击节点 $M...N$ 加入已合并节点集合中。然后将查询节点 $Q_i$ 及其父子关系集$\{R(Q_i, M),R(N, Q_i)\}$从其所在的子任务子树中删除。

表4.3 基于文本匹配的时间树合并算法

Table 4.3 The algorithm of TimeTree merging based on text matching

**算法4.3** 基于文本匹配的时间树合并算法

算法输入：包含种子关键词 $Q$ 的子任务子树集合 SubTaskTreeSet

算法输出：合成后的子任务有向图

```
1.  Begin
2.    For each SubTaskTree ∈ SubTaskTreeSet
3.      checkAndAdd(Qi):
4.        For each Qi ∈ SubTaskTree
5.          If MergedNodeSet contains Qj whose content is equals to Qi's content
6.            For each R(Qi, M) ∈ RCSTRelationSet and R(N, Qi) ∈ RCSTRelationSet
7.              checkAndAdd(M)
8.              checkAndAdd(N)
9.              MergedRelationSet.add (R(Qj, M), R(N, Qj))
10.           End For
11.         Else
12.           MergedNodeSet.add(Qi)
13.         End If
14.         SubTaskTree.remove(Qi)
15.       End For
16.     End checkAndAdd
17. End
```

将不包含种子关键词 Q 的子任务合并为 n-1 张有向图的过程使用基于凝聚层次聚类的时间树合成算法完成，算法如表 4.4 所示。

表4.4 基于凝聚层次聚类的时间树合成算法

Table 4.4 The algorithm of TimeTree merging based on agglomerative hierarchical clustering

**算法4.4** 基于凝聚层次聚类的时间树合成算法

算法输入：不包含种子关键词 $Q$ 的子任务子树集合 SubTaskTreeSet

算法输出：合成后的子任务有向图集合

```
1.  Begin
2.    MaxSimilarity ← 0
3.    While SubTaskTreeSet.size ≥ n
4.      For each SubTaskTree1 ∈ SubTaskTreeSet
5.        For each SubTaskTree2 ∈ SubTaskTreeSet, SubTaskTree2 ≠ SubTaskTree1
6.          CurrentSimilarity ← calculateSimilarity(SubTaskTree, SubTaskTree2)
7.          If CurrentSimilarity > MaxSimilarity
8.            MaxSimilarity ← CurrentSimilarity
9.            SubTreeToBeMerged1 ← SubTaskTree1
```

续表4.4 基于凝聚层次聚类的时间树合成算法

Table 4.4 The algorithm of TimeTree merging based on agglomerative hierarchical clustering

**算法4.4** 基于凝聚层次聚类的时间树合成算法

```
10.                SubTreeToBeMerged2 ← SubTaskTree2
11.              End If
12.            End For
13.          End For
14.          MergedGraph ← mergeSubTree(SubTreeToBeMerged1,SubTreeToBeMerged2)
15.          SubTaskTreeSet.add(MergedGraph)
16.          SubTaskTreeSet.remove(SubTreeToBeMerged1)
17.          SubTaskTreeSet.remove(SubTreeToBeMerged2)
18.      End While
19.  End
```

算法循环执行，每次循环中首先对不包含种子关键词 $Q$ 的子任务有向图集合中的子任务有向图进行两两遍历，取出当前相似度最高的两个子任务有向图进行合并，并用合并后的子任务有向图替换合并前的两张子任务有向图。这样，每次子任务有向图集合的规模减 1，直到有向图集合中的子任务有向图数量为 *n*-1 为止，*n* 为每一棵时间树上划分子任务的平均数量。

考虑到不同用户的搜索习惯不同，造成不同用户对应同一子任务的子树在拓扑结构上未必相似，因此计算子任务有向图的相似度使用基于查询节点向量的子任务有向图相似度计算算法，算法如表 4.5 所示。

表4.5 基于查询节点向量的子任务有向图相似度计算算法

Table 4.5 The algorithm of similarity calculation of subtask digraph based on query node vector

**算法4.5** 基于查询节点向量的子任务有向图相似度计算算法

算法输入：子任务有向图 *SubTaskGraphA*，子任务有向图 *SubTaskGraphB*

算法输出：有向图相似度*GraphSimilarity*

```
1.   Begin
2.      For each  QueryA ∈ SubTaskGraphA
3.         SubTextA[] ← textSegmentation(QueryA)
4.         SubTextArraySetA.add (SubTextA [])
5.      End For
6.      For each  QueryB ∈ SubTaskGraphB
7.         SubTextB[] ← textSegmentation(QueryB)
8.         SubTextArraySetB.add (SubTextB [])
9.      End For
10.     SimSumA ← 0
```

续表4.5 基于查询节点向量的子任务有向图相似度计算算法

Table 4.5 The algorithm of similarity calculation of subtask digraph based on query node vector

**算法4.5** 基于查询节点向量的子任务有向图相似度计算算法

```
11.  For each SubTextArrayA ∈ SubTextArraySetA
12.      MaxSim ← 0
13.      For each SubTextArrayB ∈ SubTextArraySetB
14.        CurrentSim ← calculateSim(SubTextArrayA, SubTextArrayB)
15.        If CurrenstSim > MaxSim
16.          MaxSim ← CurrentSim
17.        End If
18.      End For
19.      SimSumA ← SimSumA + MaxSim
20.    End For
21.    SimSumA ← SimSumA / SubTextArraySetA.size
22.    SimSumB ← 0
23.    For each SubTextArrayB ∈ SubTextArraySetB
24.      MaxSim ← 0
25.      For each SubTextArrayA ∈ SubTextArraySetA
26.        CurrentSim ← calculateSim(SubTextArrayB, SubTextArrayA)
27.        If CurrenstSim > MaxSim
28.          MaxSim ← CurrentSim
29.        End If
30.      End For
31.      SimSumB ← SimSumB + MaxSim
32.    End For
33.    SimSumB ← SimSumB / SubTextArraySetB.size
34.    GraphSimilarity ← (SimSumA+SimSubB) / 2
35.  End
```

算法的主要思想是使用有向图 A 中所有节点词向量与有向图 B 的相似度平均值作为有向图 A 与有向图 B 的相似度。其中节点词向量与有向图的相似度取节点词向量与有向图中所有节点词向量的相似度的最大值。并且分别计算有向图 A 与有向图 B 的相似度以及有向图 B 与有向图 A 的相似度并求得平均值，以保证相似度的自反性。

算法首先对有向图 A 及有向图 B 中的所有查询节点的查询关键词进行分词以及去停止词处理。然后分别针对有向图 A 中每个查询节点，计算该查询节点的分词向量与有向图 B 中所有查询节点的分词向量的相似度并取最大值作为该查询节点与有向图 B 的相似度，将有向图 A 中所有查询节点与有向图 B 的相似度求平均值。用同样的方法

求得有向图 B 中所有查询节点与有向图 A 的相似度的平均值。将两平均值取平均，则求得有向图 A 与有向图 B 的相似度。

### 4.1.4 子任务内部的查询推荐方法

子任务内部的查询推荐分为两个步骤进行：节点 Rank 值计算与最优路径求解。

(1) 计算节点 Rank 值

在 4.1.3 节对主题经验进行合并后，包含用户提交的查询 Q 的子任务子树被合并为一张子任务有向图，子任务内部的查询推荐首先需要计算这张子任务有向图上所有节点的节点 Rank 值。

本研究采用带有总体衰减系数的 PageRank 算法计算节点的 Rank 值。带有总体衰减系数的 PageRank 算法公式为公式 4.1。

$$PR(A) = \mu \times \left\{ (1-d) + d \times \sum_{i=1}^{n} \left[ \frac{PR(T_i)}{L(T_i)} \right] \right\} \tag{4.1}$$

公式中除 μ 以外的部分为 *Naive PageRank*，$PR(A)$ 是指节点 A 的 *Rank* 值，$T_1$，$T_2$，…，$T_n$ 是指节点 A 的链入节点，$PR(T_i)$ 是指节点 $T_i$ 的 *Rank* 值($i = 1,2,...,n$)，$L(T_i)$ 是指节点 $T_i$ 的链出数量($i = 1,2,...,n$)，$d$ 是一个衰减因子，0<$d$ <1，通常取值为 0.85。

$\mu$ 为总体衰减系数。节点 *Rank* 值从用户提交的查询 $Q$ 开始，加入总体衰减系数是为了使与 $Q$ 距离越远的节点 *Rank* 值越低。因此 $\mu$ 的计算公式为公式 4.2。

$$\mu = 1 - \frac{1}{l} \tag{4.2}$$

其中 $l$ 为节点 $A$ 到用户提交的查询 $Q$ 的最短步数，即节点 $A$ 与节点 $Q$ 的不带权最短路径上边的条数。

(2) 求最优路径

在得到子任务有向图中所有节点的 *Rank* 值后，为了给出查询推荐，还需要求出从用户提交的查询节点到推荐的查询节点间的最优路径。本研究认为最优路径的选择应当遵循的原则是最优路径上的节点 *Rank* 值的平均值最高。

因此求最优路径的问题转化为从用户提交的查询节点到推荐的查询节点的路径中，节点 *Rank* 平均值最高的路径。本研究采用边所指向的点的 *Rank* 值的倒数作为边的权值，将该问题进一步转化为求有向图中两固定点的带权最短路径问题。并使用 *Dijkstra* 算法求解。

### 4.1.5 跨子任务的查询推荐方法

跨子任务的查询推荐分为两个步骤进行，首先针对每个不包含用户提交的查询 Q 的子任务有向图求最高 Rank 值节点，然后提取最优路径，以求得用户从该节点出发所能满足的信息需求。

(1) 从每个不包含用户提交的查询 Q 的子任务有向图中计算出 *Rank* 值最高的查询

与 4.1.4 节中子任务内部的查询推荐方法一样，针对每个不包含用户提交的查询 $Q$ 的子任务有向图计算节点 *Rank* 值同样采用 *Naive PageRank* 算法，但由于子任务有向图中不包含查询 $Q$，因此不添加总体衰减系数，即使用 *Naive PageRank* 算法，公式为公式 4.3。

$$PR(A) = (1-d) + d \times \sum_{i=1}^{n} \left[ \frac{PR(T_i)}{L(T_i)} \right] \tag{4.3}$$

同样，$PR(A)$ 是指节点 A 的 *Rank* 值，$T_1$，$T_2$，…，$T_n$ 是指节点 A 的链入节点，$PR(T_i)$ 是指节点 $T_i$ 的 *Rank* 值 $(i = 1,2,...,n)$，$L(T_i)$ 是指节点 $T_i$ 的链出数量 $(i = 1,2,...,n)$，$d$ 是一个衰减因子，$0 < d < 1$，通常取值为 0.85。

(2) 从每一个子任务有向图中的推荐查询出发，提取 *Rank* 值最高的路径集合

在获取到一个子任务有向图中所有查询词的 *Rank* 值后，从选取的 *Rank* 值最高的推荐查询词出发，对子任务有向图进行深度优先遍历，并在遍历过程中记录每条路径上节点的 *Rank* 值之和，选取出 *Rank* 值之和最高的路径加入推荐路径集合。在遍历过程中，如果路径成环，则不选择该路径作为推荐路径，因为该路径上的查询点击所能够满足的信息需求是循环的。

## 4.2 面向搜索经验的查询推荐可视化方法

在提出子任务内部的查询推荐方法及跨子任务的查询推荐方法基础上，为了保证解决方案的完整性，本研究提出面向查询起源的查询推荐结果可视化方法。

### 4.2.1 面向因果经验的查询推荐可视化

在第三章中，本研究提出了基于点击-查询序列识别的因果经验提取算法，利用该算法，本研究可以提取出所有进行过某一复杂搜索任务的用户的搜索过程中所有的点击-查询序列。在此基础上，当有进行该复杂搜索任务的用户提交了某一个查询词，该查询词将与提取过的查询-点击-查询序列集合进行匹配，如果用户提交的查询词能与序列中前一查询匹配成功，则给出该查询-点击-查询序列作为查询推荐。对于此种情况的可视化设计如图 4.2 所示。

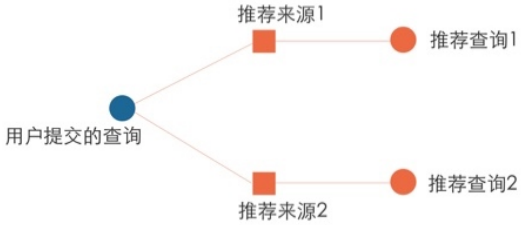


图 4.2 基于因果经验的查询推荐可视化

Fig. 4.2 query recommendation visualization based on causal experience

当用户提交一条查询，在用户所进行的当前复杂搜索任务的时间树上将显示所有与该查询匹配的查询-点击-查询序列。用户将清晰地看出，给出这些查询推荐的原因——该查询来自某个用户曾经点击过的内容。

### 4.2.2 子任务内部的查询推荐可视化

在子任务内部的查询推荐中，计算出了包含用户所提交查询的子任务有向图中 Rank 值最高的查询，以及从用户提交的查询到该推荐查询的路径中 Rank 值最高的路径。在子任务内部的查询推荐结果可视化中，该路径默认被隐藏，用户只看到从自己提交的查询出发经过某些过程，可能会想要查询的内容。如果用户想要详细了解该推荐查询的起源，则完整的路径将被展现。子任务内部的查询推荐结果可视化设计如图 4.3 所示。

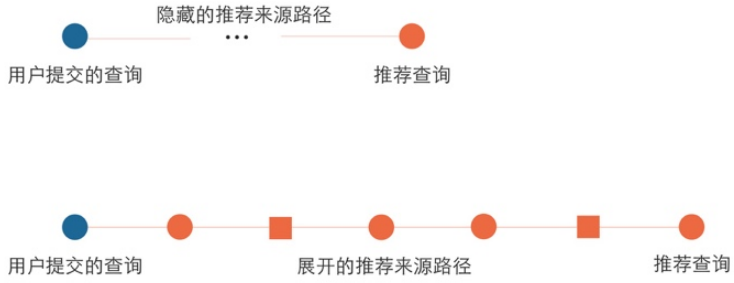


图 4.3 子任务内部的查询推荐可视化

Fig. 4.3 query recommendation visualization for results inside the subtask

当用户提交一个查询，展现给用户的内容如图 4.3 的上半部分所示，所推荐查询的来源被隐藏。而当用户想要详细了解该查询推荐的起源，则如图 4.3 的下半部分所示，用户可以看到从提交的查询到推荐的查询的完整路径。

### 4.2.3 跨子任务的查询推荐可视化

在跨子任务的查询推荐中，通过计算每一张子任务有向图上节点的 Rank 值，每一个被提取的子任务有向图都可以得到一个 Rank 值最高的查询。在跨子任务的查询推荐结果可视化中，各子任务的查询推荐结果都将被展示，并且，用户还可以选择查看从

特定的查询推荐出发，用户可能能够获得的信息需求。跨子任务的查询推荐结果可视化如图 4.4 所示。

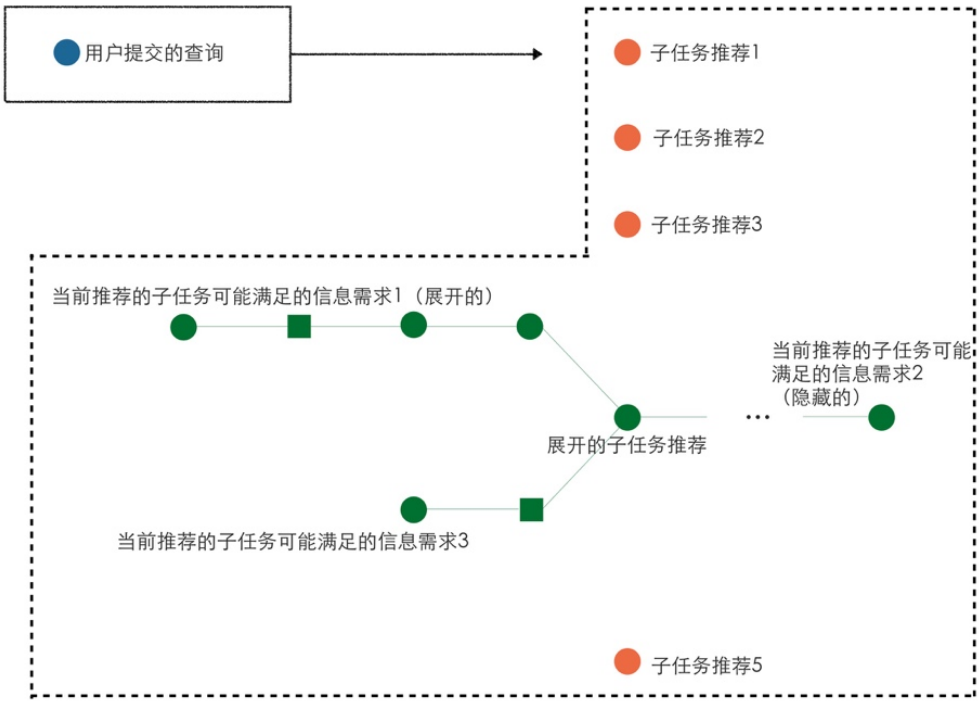


图 4.4 跨子任务的查询推荐可视化

Fig. 4.4 query recommendation visualization for results outside the subtask

用户提交一个查询后，将得到 5 个属于不同子任务的查询推荐。对于每一个子任务推荐，用户都可以将其展开，查看从该查询出发，未来可能会得到什么样的查询结果。

## 4.3 面向搜索经验的查询推荐方法对比实验

本章分别提出了面向用户因果经验、子任务内部以及跨子任务的查询推荐方法。本节将设计对比实验，用以对本文提出的面向用户搜索经验的查询推荐方法有效性进行验证。

### 4.3.1 实验设计

本实验针对面向搜索经验的查询推荐方法，将第三章时间树中搜索经验验证性实验中用户的搜索过程数据随机平均分为训练集与测试集，使用训练集进行用户搜索经验提取，使用测试集进行查询推荐结果验证。并设计对比方法以及评价标准，以考察本研究所提出的查询推荐方法的质量。

### 4.3.2 对比方法

本实验采用传统的基于用户搜索历史匹配的查询推荐方法，当用户提交查询词时，将用户提交的查询与已有的用户查询历史进行匹配，将得到的匹配结果的后续查询作为查询推荐结果。

### 4.3.3 评价标准

本研究所提出的面向搜索经验的查询推荐方法的优势在于查询多样性，当用户提交一个查询，传统的查询推荐结果只能基于统计给出大多数用户在该查询的后续查询，但使用本研究提出的查询推荐方法，可以给出基于因果经验的查询推荐，子任务内部的查询推荐以及跨子任务的查询推荐结果集，因此在多样性上具有显著的优势。但在大多数研究中，均使用准确率与召回率作为推荐系统推荐质量的评价标准，因此本实验仍验证查询推荐方法的准确率与召回率。准确率与召回率的计算方法如图 4.5 所示。

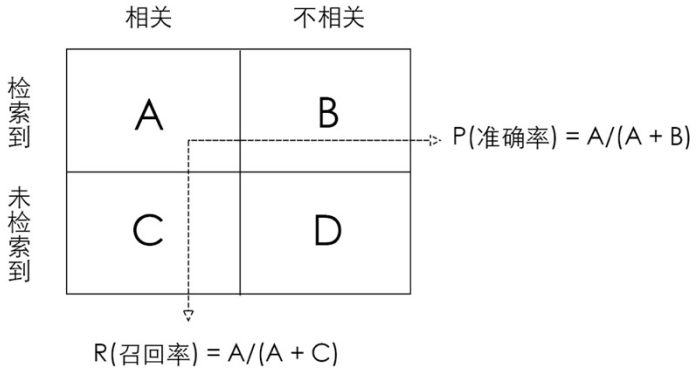


图 4.5 准确率与召回率的计算方法

Fig. 4.5 calculation method of accuracy and recall rate

给出查询推荐与用户进行的查询的相关和不相关，用户进行的所有查询被检索到以及未被检索到，将查询推荐以及用户进行的查询分为 4 个部分。准确率即为被检索到与用户查询相关的查询个数除以被检索到的查询总数。召回率即为被检索到与用户查询相关的查询个数除以与用户查询相关的查询总数。

### 4.3.4 结果分析

在第三章搜索经验提取验证实验中得到结论：对于学习型任务，基于子任务划分的主题经验提取算法能够有效地提取出主题经验，但对于试探型任务，基于子任务划分的主题经验提取算法对主题经验的提取效果不理想，在时间树维护状况极短不理想

情况下有失效的可能。本节实验仍分别针对学习型任务及试探型任务对实验结果进行分析。

(1) 学习型任务

学习型任务的查询推荐准确率及召回率结果如表 4.6 所示。

表4.6 面向搜索经验的查询推荐方法对比实验结果表（学习型任务）

Table 4.6 results of search experience oriented query recommendation method contrast experiment (for learning task)

| 评价指标 | 面向因果经验的查询推荐 | 子任务内部的查询推荐 | 跨子任务的查询推荐 | 基于搜索历史匹配的查询推荐（对比方法） |
|---|---|---|---|---|
| 准确率 | 0.911 | 0.768 | 0.474 | 0.664 |
| 召回率 | 0.520 | 0.634 | 0.680 | 0.748 |

从表中实验结果可以看出，子任务内部的查询推荐及跨子任务的查询推荐准确率及召回率均与用于对比的基于搜索历史匹配的查询推荐方法相差不大，其中子任务内部的查询推荐准确率高于对比方法，子任务内部的查询推荐召回率以及跨子任务的查询推荐准确率及召回率低于对比方法，但均在可接受范围内。面向因果经验的查询推荐方法准确率高，但召回率相对较低，主要原因来自于数据量不足，造成因果经验提取量不足，导致查询推荐结果有一定局限性，覆盖面不够广，并最终体现在召回率偏低上。

从整体来看，在学习型任务中，虽然本研究提出的面向搜索经验的查询推荐方法准确率及召回率并不能完全高于对比算法，但其给出了多样性的查询推荐结果，该方法的有效性得到了证明。

(2) 试探型任务

试探型任务的查询推荐准确率及召回率结果如表 4.7 所示。

表4.7 面向搜索经验的查询推荐方法对比实验结果表（学习型任务）

Table 4.7 results of search experience oriented query recommendation method contrast experiment (for exploratory task)

| 评价指标 | 面向因果经验的查询推荐 | 子任务内部的查询推荐 | 跨子任务的查询推荐 | 基于搜索历史匹配的查询推荐（对比方法） |
|---|---|---|---|---|
| 准确率 | 0.545 | 0.510 | 0.125 | 0.895 |
| 召回率 | 0.807 | 0.665 | 0.831 | 0.631 |

从表中结果可以看出，试探型任务中，所有推荐方法的召回率相差不大，但面向搜索经验的查询推荐方法查询推荐准确率均低于对比算法，其中，跨子任务的查询推荐方法准确率最低，与对比算法相差最大。造成这样结果的原因是，试探型任务中用

户进行搜索的子任务不明显，主题经验提取效果不理想，导致基于主题经验的查询推荐效果不理想。这与第三章中主题经验提取实验中得到的结论相吻合。

因此，试探型任务中，面向搜索经验的查询推荐结果在跨子任务的查询推荐中，效果不理想。但在面向因果经验的查询推荐及子任务内部的查询推荐中，方法的有效性得到了证明。

### 4.3.5 实验结论

基于上述对实验结果的分析可以得出结论，对于学习型任务及试探型任务，本章提出的面向搜索经验的查询推荐方法在面向因果经验查询推荐及子任务内部的查询推荐中均能够有效地进行查询推荐，在学习型任务中，面向搜索经验的查询推荐方法能够有效进行跨子任务的查询推荐，但在试探型任务中，该方法并不能有效进行跨子任务的查询推荐。

## 4.4 结论

本章提出了面向搜索经验的查询推荐方法及查询推荐可视化方法，并设计实验对该查询推荐方法进行了验证，证明了该方法在学习型任务中的有效性及在试探型任务中的部分有效性。

# 第5章 面向搜索经验的查询推荐系统

本章根据上文提出的用户搜索经验提取及面相搜索经验的查询推荐方法，基于已有的复杂搜索管理系统 CiteXplore 实现面向搜索经验的查询推荐系统，以提供一套完整的面相搜索经验查询推荐解决方案。本章首先对系统用例进行分析，对系统功能、架构及数据库进行设计，然后分别针对各个功能模块进行实现。

## 5.1 系统分析与设计

本节对系统进行分析，首先对系统用例进行分析，在用力分析的基础上对系统功能进行设计，并在此基础上做出系统架构以及数据库设计。

### 5.1.1 用例分析

本系统基于 CiteXplore 系统进行二次开发，CiteXplore 系统的系统参与者为使用系统进行复杂搜索过程管理的用户，并没有设计管理员，面向搜索经验的查询推荐系统沿用这样的设计。因此，本系统中用例的主要驱动者也为使用本系统的用户。

系统用户的用例图如图 5.1 所示。

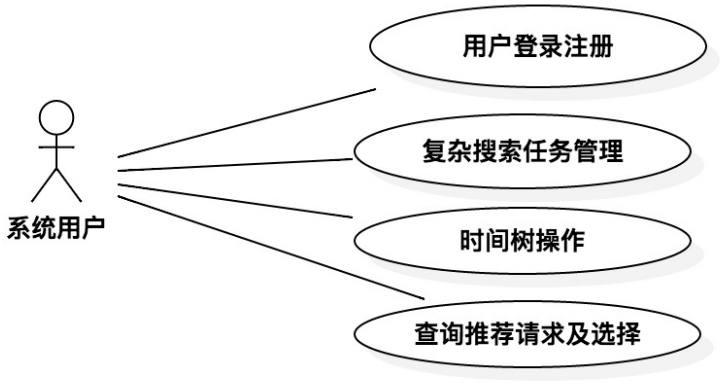


图 5.1 系统用户用例图

Fig. 5.1 Use Case Diagram of The System User

系统用户用例主要包括：用户登录/注册，复杂搜索任务管理，时间树操作及查询推荐请求及选择。其中用户登录/注册、新建复杂搜索任务以及时间树操作的用例直接沿用 CiteXplore 系统的用例设计，本文中不予以重点介绍，接下来将重点介绍查询推荐请求及选择用例。

系统用户在使用本系统进行复杂搜索过程管理时，可以对时间树进行操作，包括时间树节点拖拽、展开及收起、评分、评论、请求查询推荐以及对查询推荐结果进行选择。其中查询推荐请求以及对查询推荐结果进行选择为本系统相对于 CiteXplore 系

统的新增用例，用例说明如表 5.1 所示。

表 5.1 查询推荐请求及选择用例说明

Table 5.1 TimeTree operation use case

| 用例名称 | 查询推荐请求及选择 |
|---|---|
| 简要描述 | 系统用户通过对时间树的操作请求查询推荐，并对查询推荐结果进行选择 |
| 主执行者 | 系统用户 |
| 触发条件 | 用户提出查询推荐请求/用户选择查询推荐结果或重新组织查询 |
| 后置条件 | 查询推荐结果展示/恢复时间树 |
| 基本事件流 | 获取查询推荐结果，展示查询推荐结果，提交用户所选择的推荐查询 |
| 异常事件流 | 查询推荐结果获取失败 |

### 5.1.2 系统功能设计

根据 5.1.1 节用例分析中给出的系统用户用例图，本节将对系统功能进行设计，给出系统的功能模块图，对本系统进行进一步的分析。系统功能模块图如图 5.2 所示。

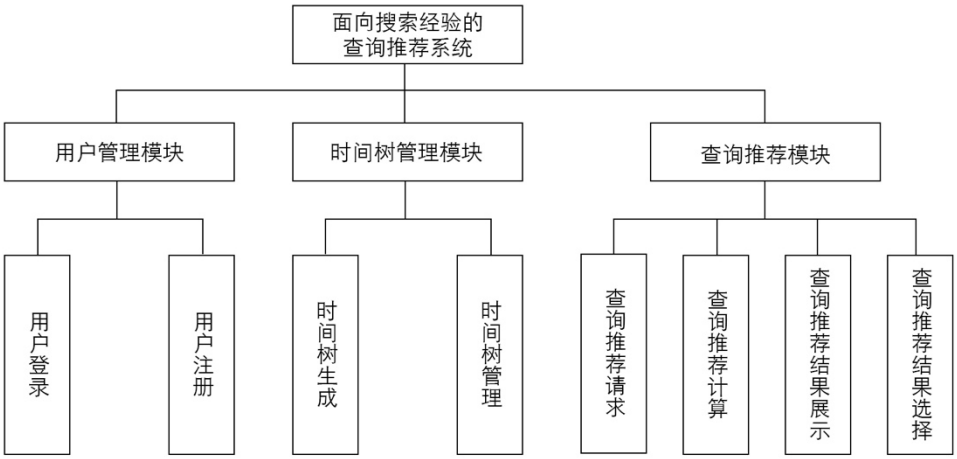


图 5.2 系统功能模块图

Fig. 5.2 System function module diagram

系统功能模块主要包括用户管理模块，时间树管理模块，以及查询推荐模块。其中用户管理模块与时间树管理模块分别实现用户的登录注册以及时间树的生成与管理，这两部分由 CiteXplore 系统实现，在本文中不再详作介绍。查询推荐模块为本系统的主要设计研发部分，该模块实现用户对查询推荐结果的请求与选择，计算查询推荐结果，并负责展现查询推荐结果。因此该功能模块包含四部分：查询推荐请求，查询推荐计算，查询推荐结果展示以及查询推荐结果选择。

(1) 查询推荐请求功能

在用户进行复杂搜索的过程中，系统将自动生成时间树以维护用户的搜索经验，当用户提交了一个查询，用户即可以选择请求查询推荐，请求方式为键盘 Ctrl 键加鼠标悬停动作，悬停位置为时间树上当前查询节点。

(2) 查询推荐计算功能

查询推荐的计算在用户输入查询时即进行预计算，当用户点击请求查询推荐时直接将查询推荐结果发送给前端查询推荐结果展示功能，以保证系统的流畅性，避免用户感受过长的等待时间。

(3) 查询推荐结果展示功能

当用户请求了查询推荐结果，查询推荐需要根据本研究提出的面向搜索经验的查询推荐可视化方法向用户展示查询推荐结果。查询推荐结果同样以时间树的形式展现。

(4) 查询推荐结果选择功能

在向用户展示了查询推荐结果以后，用户需要从查询推荐结果中选择自己下一步想要进行的查询，用户可以在查询推荐结果时间树上对查询节点进行点击，将查询推荐结果中的某个查询作为自己下一步的查询。如果用户自己重新组织查询，则认为用户放弃对查询推荐结果的选择，则系统根据用户新提交的查询进行下一步的查询推荐计算。

### 5.1.3 系统架构设计

为实现 5.1.2 节中对系统功能模块的设计，需要对系统架构进行设计，系统架构设计如图 5.3 所示。

系统架构分为 4 个基本部分，用户管理模块、时间树管理模块、查询推荐计算模块以及前端时间树及查询推荐结果可视化模块。其中用户管理模块由 CiteXplore 系统直接提供支持，下面将分别针对时间树管理模块、查询推荐计算模块和前端时间树及查询推荐结果可视化模块的架构进行介绍。

(1) 时间树管理模块架构设计

时间树管理模块为系统的基础，用户在使用系统进行复杂搜索的过程中，时间树管理模块需要实现时间树的生成、存储以及管理，查询推荐计算模块计算出的查询推荐结果，也以时间树的形式存在，因此也需要由时间树管理模块来进行管理。基于上述需求，时间树管理模块分为 2 部分：时间树存储以及时间树生成与维护。

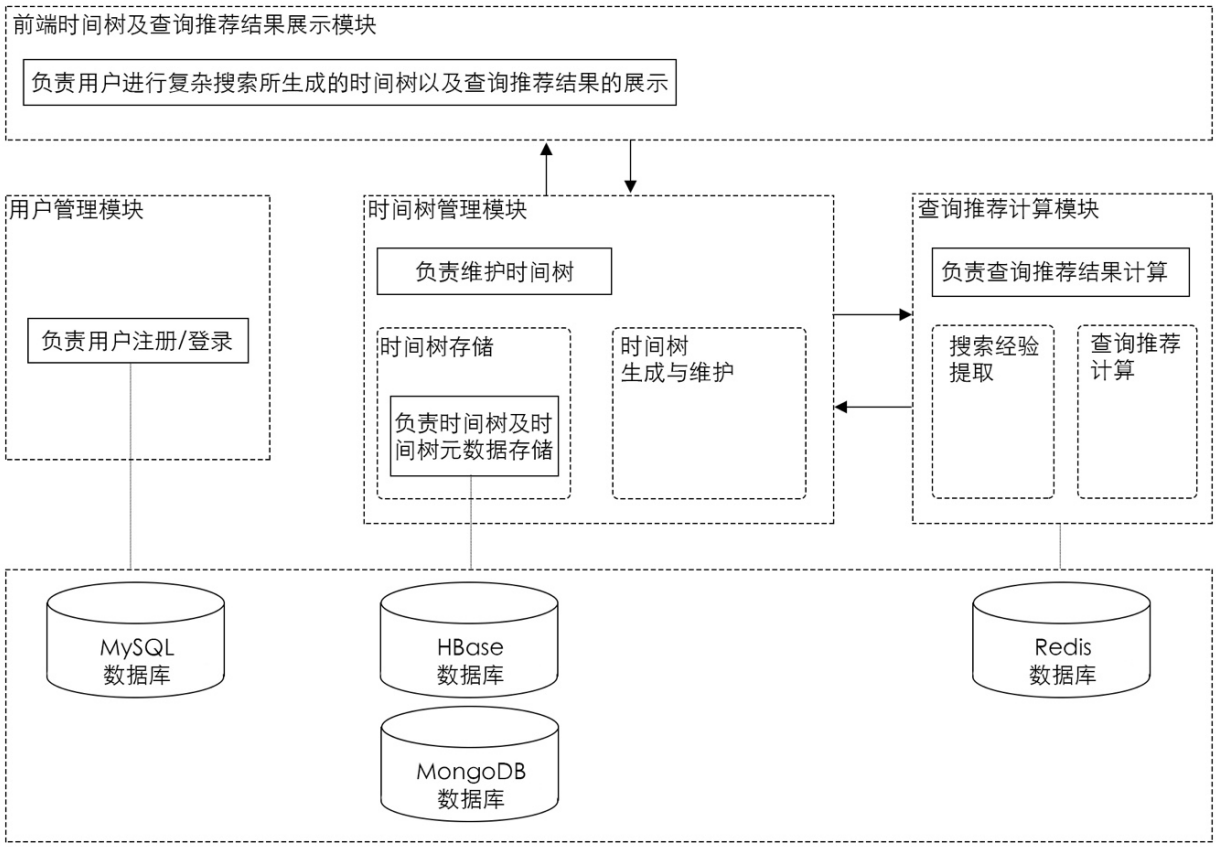


图 5.3 系统架构设计图

Fig. 5.3 System architectural design diagram

当用户在前端提出查询推荐结果查看请求时，时间树生成与维护模块将调用查询推荐计算模块的计算服务，并将返回结果生成时间树返回给前端模块，并且由时间树存储模块将生成的查询推荐时间树以及查询推荐元数据信息存储到数据库中。其中查询推荐时间树使用 HBase 以键-值形式存储，查询推荐元数据使用 MongoDB 以文档形式存储。

(2) 查询推荐计算模块架构设计

查询推荐计算模块以独立服务的形式部署在系统中，前端时间树及查询推荐结果可视化模块中获取到的用户查询行为触发计算，时间树管理模块为查询推荐计算提供数据基础。当用户提交一个查询，查询推荐计算模块首先向时间树管理模块请求时间树信息，首先通过搜索经验提取模块，使用用户提交的查询以及已有的时间树信息提取出用户的搜索经验，再经过查询推荐计算模块计算出要向用户进行的查询推荐结果。并将查询推荐结果存储在 Redis 缓存中，当调用模块请求查询推荐结果时，查询推荐计算模块从 Redis 中读取查询推荐结果并返回给调用模块。

(3) 前端时间树及查询推荐结果可视化模块

在 CiteXplore 系统中，时间树的展示以 Firefox Add-on SDK 开发的浏览器插件形式进行，本系统沿用这一设计，仍旧使用浏览器插件脚本注入的方式获取用户的查询点击数据，并且当用户进行查询时，调用系统查询推荐计算模块提供的查询推荐计算服务，并将返回的数据，展示给用户。

## 5.2 系统实现

本系统主要采用 JDK8 进行开发，Web 服务器采用 Tomcat 9.0，使用数据库 MySQL 5.7、HBase 1.1.3、MongoDB 3.2.1 以及 Redis 4.0.2。前端采用 D3.js 以及 Firefox Add-on SDK 进行开发。本节将分别着重交代时间树管理模块、查询推荐计算模块以及时间树及查询推荐可视化模块的系统实现。

### 5.2.1 时间树管理模块

时间树管理模块主要实现的功能包括时间树生成与维护以及时间树存储。

(1) 时间树生成与维护

当用户请求了查询推荐结果，时间树管理模块向查询推荐计算模块转发了请求并得到返回结果后，需要将返回的基于因果经验的查询推荐结果、子任务内部的查询推荐结果以及跨子任务的查询推荐结果生成时间树并与用户当前时间树进行结合。时间树生成与维护模块主要涉及的文件如表 5.2 所示。

表 5.2 时间树生成与维护模块涉及的文件列表

Table 5.2 File list of TimeTree generating and maintaining module

| 文件名 | 文件说明 |
|---|---|
| Arc.java | 时间树边结构类 |
| Click.java | 用户点击动作封装类 |
| ClickVertex.java | 点击节点类 |
| ExplorationGraph.java | 时间树类 |
| ExplorationGraphException.java | 时间树异常类 |
| GraphJsonBuilder.java | 时间树 Json 构建工具 |
| GraphManager.java | 时间树管理工具 |
| GraphManagerListener.java | 时间树管理监听，用于更新前端数据 |
| Query.java | 用户查询动作封装类 |
| QueryVertex.java | 查询节点类 |
| Vertex.java | 节点抽象类 |
| VertexOperation.java | 节点操作抽象类 |

(2) 时间树存储

时间树存储模块主要负责对时间树生成与维护模块生成好的时间树及其元数据进行持久化存储，时间树存储模块主要涉及的文件如表 5.3 所示。

表 5.3 时间树存储模块涉及的文件列表

Table 5.3 File list of TimeTree storage module

| 文件名 | 文件说明 |
|---|---|
| GraphMetadata.java | 时间树元数据类 |
| GraphMetadataStorage.java | 时间树元数据存储类 |
| GraphStorage.java | 时间树存储类 |

### 5.2.2 查询推荐计算模块

查询推荐计算模块负责查询推荐结果的计算，当用户提交查询后，查询推荐计算模块首先提取搜索经验，然后根据提取的搜索经验计算查询推荐结果。

(1) 搜索经验提取

搜索经验提取分为因果经验提取以及主题经验提取。其中因果经验提取主要过程为对查询-点击-查询序列的识别与提取，主题经验提取的主要过程为对已有时间树数据进行聚类及合并。搜索经验提取模块主要涉及的文件如表 5.4 所示。

表 5.4 搜索经验提取模块涉及的文件列表

Table 5.4 File list of search experience extraction module

| 文件名 | 文件说明 |
|---|---|
| CausalExtractor.java | 因果经验提取类 |
| RCSTNode.java | RCST 节点结构类 |
| Tool.java | 基本工具类 |
| TreeNode.java | 树结构类 |
| TimeTreeNodeClusteror.java | 时间树聚类，即主题经验提取类 |

(2) 查询推荐计算

查询推荐计算模块使用提取的搜索经验对查询推荐结果进行计算，其主要涉及的文件如表 5.5 所示。

表 5.5 查询推荐计算模块涉及的文件列表

Table 5.5 File list of query recommendation computing module

| 文件名 | 文件说明 |
|---|---|
| CausalRecommender.java | 基于因果经验的查询推荐类 |
| PageRankBasic.java | 节点 Rank 计算类 |
| Synthesizer.java | 时间树合成器 |
| ThematicRecommender.java | 基于主题经验的查询推荐类 |

### 5.2.3 时间树及查询推荐可视化模块

时间树及查询推荐可视化模块使用 d3.js 与 Firefox Add-on SDK 进行开发，以火狐浏览器插件作为容器，使用 d3.js 绘制时间树，其主要涉及的文件如表 5.6 所示。

表 5.6 时间树及查询推荐可视化模块涉及的文件列表

Table 5.6 File list of visualization module of TimeTree and Recommendation Result

| 文件名 | 文件说明 |
|---|---|
| index.jsp | 系统主界面 |
| common.js | Websocket 连接及全局功能脚本文件 |
| graph.js | 探索图绘制主脚本文件 |
| injector.js | 浏览器插件注入脚本 |
| sender.js | 数据传送脚本 |
| sidebar.js | 浏览器插件侧边栏容器脚本 |
| baiduXueshu.js | 针对百度学术的前端解析脚本 |
| bingXueshu.js | 针对必应学术的前端解析脚本 |
| googleXueshu.js | 针对谷歌学术的前端解析脚本 |

## 5.3 实例分析

本节以面向搜索经验的查询推荐业务流程作为实例对系统进行分析。

当用户提交一个查询，首先需要在前端获取用户的查询以及对查询结果页面中的点击链接进行 js 注入。js 注入结果如图 5.4 所示。


图 5.4 查询结果页面 js 注入图

Fig. 5.4 Query result page Javascript injecting

在获取用户查询或点击结果后，系统将用户查询或点击加入时间树，并且在前端使用 d3.js 进行可视化。时间树可视化结果如图 5.5 所示。


图 5.5 时间树可视化结果图

Fig. 5.5 TimeTree visualization result

用户可以随时对查询推荐结果进行请求，用户在时间树查询节点上使用“Ctrl + 点击”操作请求查询推荐结果，查询推荐结果可视化模块将查询推荐结果展示给用户。查询推荐结果可视化如图 5.6 所示。


图 5.6 查询推荐结果可视化

Fig. 5.6 Query recommendation result visualization

图中查询推荐结果可分为 4 部分，其中第一部分为面向因果经验的查询推荐；第二部分为子任务内部的查询推荐，灰色节点为未展开的从用户提交查询到推荐查询的最优路径；第三部分为跨子任务的查询推荐，灰色节点为未展开的从推荐节点出发用户可以获得的信息需求；第四部分为展开的从推荐节点出发用户可以获得的信息需求。

用户可以同样使用“Ctrl + 点击”操作对查询推荐结果进行选择，以查询推荐词为关键词提交查询，或重新提交查询。查询推荐后续操作结果如图 5.7 所示。


图 5.7 查询推荐后续操作结果

Fig. 5.7 Query recommendation follow up operation result

## 5.4 小结

本章首先给出了面向搜索经验的查询推荐系统的设计，包括用力分析、系统功能设计以及架构设计，并根据设计实现了面向搜索经验的查询推荐原型系统，最后以查询推荐实例验证了该系统的实用性。

# 第6章 总结与展望

在用户使用时间树进行复杂搜索的过程中，时间树记录了用户的搜索过程的同时，是否维护了用户的搜索经验，能否利用这些搜索经验进行查询推荐，以方便用户满足信息需求，基于以上思路，本研究提出了面向搜索经验的查询推荐方法，并基于该方法，实现了面向搜索经验的查询推荐系统，为复杂搜索过程中面向搜索经验的查询推荐提供了一套完整的解决方案。

## 6.1 本文主要工作

针对复杂搜索过程中面向搜索经验的查询推荐，本研究主要工作如下：

(1) 提出了搜索经验模型以及搜索经验一致性模型，将搜索经验分为时间经验、因果经验以及主题经验，并对应地将搜索经验一致性分为时间一致性、因果一致性以及主题一致性，为搜索经验以及搜索经验一致性提供了模型化定义的基础。

(2) 分别针对复杂搜索中的学习型任务以及试探型任务对时间树中搜索经验的蕴含性进行了实验验证，验证了时间树中搜索经验的蕴含性，证明了时间树在用户进行复杂搜索的过程中能够有效帮助用户维护搜索经验一致性。

(3) 针对时间树提出了搜索经验提取方法，分别提出了基于查询-点击-查询序列识别的因果经验提取算法以及基于子任务划分的主题经验提取算法，为时间树中搜索经验的提取提供了理论方法，为面向搜索经验的查询推荐提供了基础，并且设计实验对基于子任务划分的主题经验提取算法有效性进行了验证。

(4) 以搜索经验模型以及搜索经验提取方法为基础，提出了面向搜索经验的查询推荐方法以及查询推荐可视化方法，针对搜索经验模型中的因果经验以及主题经验提出了面向因果经验的查询推荐方法、子任务内部的查询推荐方法、跨子任务的查询推荐方法以及对应的查询推荐可视化方法，并设计实验验证了面向搜索经验的查询推荐方法的有效性。

(5) 基于复杂搜索管理系统 CiteXplore 以及面向搜索经验的查询推荐方法理论，设计并实现了面向搜索经验的查询推荐系统，使本研究形成了一套完整的解决方案。

## 6.2 未来工作展望

本研究所提出的面向搜索经验的查询推荐方法为复杂搜索中基于时间树的查询推荐提供了有效的解决方案。然而本研究尚有一些不足，未来该方向可在以下几个方面继续进行研究：

(1) 本研究所提出的基于时间树的搜索经验提取方法以及面向搜索经验的查询推荐方法对于复杂搜索任务中的学习型任务有很好的支持，但对于试探型任务的支持不足。其原因在于用户在进行试探型任务时，时间树难以进行有效的自组织，因此，在未来的研究中，需要针对试探型任务，进行试探型任务的特性分析，寻找到支持用户进行试探型任务、有效对试探型任务进行时间树自组织的方法，进而寻找到有效提取试探型任务中搜索经验、进一步利用搜索经验进行试探型任务的查询推荐的方法。

(2) 目前的复杂搜索任务管理系统 CiteXplore，在学术搜索中对用户的复杂搜索过程有良好的支持作用，因为学术搜索天然具有复杂搜索的特性，但在通用搜索中，由于通用搜索情况复杂，需要对用户在通用搜索中的复杂搜索行为进行识别，并且针对通用搜索，需要更加完善的时间树源节点解析以支持时间树的建立与自组织。因此，在未来的研究中，需要研究一种新的溯源策略，基于该策略，支持完美建立时间树，为通用搜索中面向搜索经验的查询推荐提供基础。

(3) 本研究所提出的针对主题经验的查询推荐方法中所涉及到的主题经验合并算法，在当前数据量下有很高的效率表现，但未来如果数据量增大，需要满足大量有向图合并时，需要考虑算法效率问题。针对这一问题，可以考虑研究大数据量下的有向图合并方法，加快有向图合并及层次聚类效率，以使面向搜索经验的查询推荐方法在大数据背景下同样具有良好的应用效果。

# 参考文献


1. Rutter S, Blinzler V, Ye C, et al. Complex search task: how to make a phone safe for a child[C]//CEUR Workshop Proceedings. Rheinisch-Westfaelische Technische Hochschule Aachen, 2017, 1798: 34-36.
2. Singer G, Pruulmann-Vengerfeldt P, Norbisrath U, et al. The relationship between Internet user type and user performance when carrying out simple vs. complex search tasks[J]. arXiv preprint arXiv:1511.05819, 2015.
3. Dori-Hacohen S, Yom-Tov E, Allan J. Navigating Controversy as a Complex Search Task[C]//SCST@ ECIR. 2015.
4. Hassan Awadallah A, White R W, Pantel P, et al. Supporting complex search tasks[C]//Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management. ACM, 2014: 829-838.
5. Gäde M, Hall M, Huurdeman H, et al. Supporting complex search tasks[C]//European Conference on Information Retrieval. Springer, Cham, 2015: 841-844.
6. Franken S, Norbisrath U. Trail Building During Complex Search Tasks[C]//Mensch & Computer. 2014: 135-144.
7. Marchionini G. Exploratory search: from finding to understanding[J]. Communications of the ACM, 2006, 49(4): 41-46.
8. Singer G, Danilov D. Complex search: aggregation, discovery, and synthesis[J]. Proceedings of the Estonian Academy of Sciences, 2012, 61(2).
9. Kules B, Capra R. Designing exploratory search tasks for user studies of information seeking support systems[C]//Proceedings of the 9th ACM/IEEE-CS joint conference on Digital libraries. ACM, 2009: 419-420.
10. Franken S, Norbisrath U. Supporting the evaluation of complex search tasks with the SearchTrails tool[C]//Proceedings of 24th Annual International Conference on Computer Science and Software Engineering. IBM Corp., 2014: 262-274.
11. Zhang Y, Gao K, Zhang B, et al. TimeTree: A Novel Way to Visualize and Manage Exploratory Search Process[C]//International Conference on Human-Computer Interaction. Springer International Publishing, 2016: 313-319.
12. Jiang T. Exploratory search: a critical analysis of the theoretical foundations, system features, and research trends[M]//Library and Information Sciences. Springer Berlin

Heidelberg, 2014: 79-103.

13. Sun H C, Jiang C J, Ding Z J, et al. Topic-oriented exploratory search based on an indexing network[J]. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 2016, 46(2): 234-247.
14. Palagi E, Gandon F, Giboin A, et al. A Survey of Definitions and Models of Exploratory Search[C]//Proceedings of the 2017 ACM Workshop on Exploratory Search and Interactive Data Analytics. ACM, 2017: 3-8.
15. Athukorala K M, Glowacka D, Jacucci G, et al. Is Exploratory Search Different?[J]. American Society for Information Science and Technology. Journal, 2016.
16. Medlar A, Glowacka D. Using Topic Models to Assess Document Relevance in Exploratory Search User Studies[C]//Proceedings of the 2017 Conference on Conference Human Information Interaction and Retrieval. ACM, 2017: 313-316.
17. Baeza-Yates R A, Hurtado C A, Mendoza M. Query Recommendation Using Query Logs in Search Engines[C]//EDBT workshops. 2004, 3268(2005): 588-596.
18. Gardner T, Griffin C, Koehler J, et al. A review of OMG MOF 2.0 Query/Views/Transformations Submissions and Recommendations towards the final Standard[C]//MetaModelling for MDA Workshop. 2003, 13: 41.
19. Marcel P, Negre E. A survey of query recommendation techniques for data warehouse exploration[C]//EDA. 2011: 119-134.
20. Wang J G, Huang J Z, Guo J, et al. Query ranking model for search engine query recommendation[J]. International Journal of Machine Learning and Cybernetics, 2017, 8(3): 1019-1038.
21. He Q, Jiang D, Liao Z, et al. Web query recommendation via sequential query prediction[C]//Data Engineering, 2009. ICDE'09. IEEE 25th International Conference on. IEEE, 2009: 1443-1454.
22. Chao M A, Yin Z, Zhang B. Query Recommendation for Exploratory Search Process[J]. Journal of Northeastern University, 2015, 36(6):777-779 and 785.
23. Charikar M, Chatziafratis V. Approximate hierarchical clustering via sparsest cut and spreading metrics[C]//Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms. Society for Industrial and Applied Mathematics, 2017: 841-854.
24. Liu A A, Su Y T, Nie W Z, et al. Hierarchical clustering multi-task learning for joint human

action grouping and recognition[J]. IEEE transactions on pattern analysis and machine intelligence, 2017, 39(1): 102-114.

25. Bouguettaya A, Yu Q, Liu X, et al. Efficient agglomerative hierarchical clustering[J]. Expert Systems with Applications, 2015, 42(5): 2785-2797.
26. Farinelli A, Bicego M, Bistaffa F, et al. A hierarchical clustering approach to large-scale near-optimal coalition formation with quality guarantees[J]. Engineering Applications of Artificial Intelligence, 2017, 59: 170-185.
27. Jin R, Hong L, Wang C, et al. A Hierarchical clustering community algorithm which missed the signal in the process of transmission[J]. Review of Computer Engineering Studies, 2015, 2(3): 27-34.
28. Prasad J, Prasad R S, Kulkarni U V. Impact of Feture Selection Methods in Hierarchical Clustering Technique: A Review[J]. Lecture Notes in IMECS, 2008.
29. Tashobya C K, Dubourg D, Ssengooba F, et al. A comparison of hierarchical cluster analysis and league table rankings as methods for analysis and presentation of district health system performance data in Uganda[J]. Health policy and planning, 2015, 31(2): 217-228.
30. Almeida J A S, Barbosa L M S, Pais A, et al. Improving hierarchical cluster analysis: A new method with outlier detection and automatic clustering[J]. Chemometrics and Intelligent Laboratory Systems, 2007, 87(2): 208-217.
31. Jainontee K, Lee V S, Prasitwattanaseree S, et al. K-means clustering and hierarchical cluster analysis coupled with linear discriminant analysis to classify signals in osmotic fragility test for thalassemia screening[J]. 2013.
32. Schonlau M. Visualizing non-hierarchical and hierarchical cluster analyses with clustergrams[J]. Computational Statistics, 2004, 19(1): 95-111.
33. Sergey Brin, Larry Page. The Anatomy of a Large-Scale Hypertextual Web Search Engine[C], Seventh International World-Wide Web Conference (WWW), 1998, 14-18.
34. L Page, S Brin, R Motwani, T Winograd. The PageRank citation ranking: Bringing order to the Web Tech[J], Stanford University, 1998, 66.
35. T H Haveliwala. Topic-sensitive PageRank: A context-sensitive ranking algorithm for web search[J], IEEE Transactions on Components and Packaging Technology, 2003, 4: 15.
36. T H Haveliwala, S Kamvar, G Jeh. An Analytical Comparison of Approaches to

Personalizing PageRank[J], Stanford University, 2003, 48.

37. M Eirinaki, M Vazirgiannis. Usage-based Page Ranking for Web Personalization[C], Proceedings of 5th IEEE International Conference on Data Mining (ICDM), 2005, 12-20.
38. H Tong, C Faloutsos, J Y Pan. Fast Random Walk with Restart and Its Applications[C], ICDM Proceedings of the Sixth International Conference on Data Mining IEEE Computer Society, 2006, 613-622.
39. G Jeh, J Widow. Scaling personalized web search[C], In Proceedings of the 12th International World-Wide Web Conference, 2002, 36-48.
40. A Vattani, D Chakrabarti, M Gurevich. Preserving Personalized PageRank in Subgraphs[C], In Proceedings of the 28th International Conference on Machine Learning, 2011, 52-60.
41. K Avrachenkov, N Litvak, D A Nemirovsky, E Smirnova, M Sokol. Monte Carlo Methods for Top-k Personalized PageRank Lists and Name Disambiguation[J], INRIA, 2010, 3-36.
42. Habermas T, Bluck S. Getting a life: the emergence of the life story in adolescence[J]. Psychological bulletin, 2000, 126(5): 748.
43. Singer G, Norbisrath U, Lewandowski D. Ordinary search engine users assessing difficulty, effort, and outcome for simple and complex search tasks[C]//Proceedings of the 4th Information Interaction in Context Symposium. ACM, 2012: 110-119.

# 致 谢

2015年初春，当刚刚步入25岁的我走出研究生入学考试复试面试考场的时候，并没有也不可能想到未来的研究生生活将会是怎样一番体验。转眼两年半的时间过去，在硕士学位论文即将成稿的此刻，眼望着前方即将28岁的我，回想研究生期间的日子，想来应该有很多感慨，真要下笔书写，却也不过寥寥。这是一段不那么好走的路，幸而得到很多帮助，走下来并能自觉有所收获，我心怀感激，深觉自己的幸运。苦乐我大概说不出，得失我也许记不得，面对4万余字的论文，就借这最后一方空间，对那些在我学位论文写作过程中以及硕士研究生期间从生活上到学业上对我有所帮助的人表达我的感激之情。

硕士两年多的时间，我首先要感谢我的导师张斌老师，能够成为他的学生是我的荣幸。张斌老师思维敏捷，头脑灵活，不仅在学术上严格要求我们，还教给我们把事情做好并且讲清楚的思路与方法。在学位论文的写作中，张老师严格把关，多次与我讨论，对论文中不恰当的逻辑与结构进行重新梳理，从总体上把控了论文的大方向与进度。其中帮助，学生将一直铭记在心。

我还要感谢张引老师。张引老师于我亦师亦友，硕士就读期间，教给了我很多技术知识，我受益匪浅。张引老师在学术上与工程上都非常出色，但对于技术粗糙的我却非常的包容，在我出错的时候总是有极大的耐心一步一步教我解决问题。在学位论文写作过程中，张引老师也提供了极大的帮助，从选题到具体细节的写作方法，学生受益良多。

求学二十载，我最应该感谢我的父母，他们无条件的付出，使我有机会专心求学，不必为生活上的琐事所扰，每每想起，总觉于心有愧。此前父母出游，我独自在家生病时，方才想起父母在身边的好，身处家中，却倍有想家之感。来日可期，但希望今时今日起，我便能珍惜与他们在一起的每一分钟，多陪他们散步也好，聊天也罢，希望能够略尽我的孝心吧。

感谢我的好友：感谢魏珩珺，每当遇到困难烦恼，他总能对我进行劝慰，希望我们都能够跑得过时间，成为自己想成为的人。感谢袁野，杨成，贾洪倓，他们都给予了我精神上支持，希望过年回家的时候，都能不感冒。

感谢审阅我论文的专家、学者和老师们，感谢你们拨冗审评学生的论文。学生不才，有不足之处，还盼老师们批评指正。

最后的致谢，留给我的姥姥姥爷。我从小被姥姥姥爷带大，他们一向非常尊重知识，尊重读书人，自己对知识充满渴求，也非常重视培养子孙儿女，因此，姥姥姥爷家一直

以来书香气很浓重，对我的影响深远。姥姥虽没有看到我考上硕士研究生，但她在天之灵一定为我高兴。姥爷在我研二那年冬天辞世，希望他能够与姥姥重聚。我常常会思念他们，我会将他们生前所教所授、他们对待生活的积极态度、他们的精神力量传承下去。这种传承将使他们以无可替代的形式，永生于世。

最后，祝师长安康，同窗顺意。