**目 录**

[第一章 强化学习 1](#_Toc33117984)

[1.1 被动强化学习 1](#_Toc33117985)

[1.1.1 直接效用估计 1](#_Toc33117986)

[1.1.2 自适应动态规划 2](#_Toc33117987)

[1.1.3 时序差分学习 2](#_Toc33117988)

[1.2 主动强化学习 3](#_Toc33117989)

[1.2.1 探索 3](#_Toc33117990)

[1.3 主要功能概述 3](#_Toc33117991)

[1.4 论文主要工作 3](#_Toc33117992)

[1.5 本文组织结构 4](#_Toc33117993)

[第二章 需求分析与可行性分析 5](#_Toc33117994)

[2.1 需求分析 5](#_Toc33117995)

[2.1.1 功能需求分析 5](#_Toc33117996)

[2.1.2 性能需求分析 5](#_Toc33117997)

# 强化学习

强化学习研究Agent在没有“做什么”的标注样例的情形下怎样学习“做什么”。

**强化学习的任务是：利用观察到的回报来学习针对某个环境的最优（或接近最优）策略。**

Agent设计主要有如下三种：

* 基于效用的Agent：学习关于状态的效用函数并使用它选择期望的结果效用最大化。
* Q-学习 Agent学习行动-价值函数，或称为Q函数，该函数提供在给定状态下采取特定行动的期望。
* 反射型Agent：学习一种策略，该策略直接将状态映射到行动。

## 被动强化学习

Agent的策略是固定的，其任务是学习状态的效用，还可能涉及对环境的模型进行学习。

Agent的策略是固定的：在状态，它总是执行行动。其目标只是简单的学习该策略有多好——即学习效用函数。

被动学习的任务类似于**策略评价** ，它是**策略迭代**算法的一部分。主要区别在于被动学习Agent对指定完成行动后从状态到达状态的概率的转移模型一无所知；并且也不知道指定每个状态的回报函数。

在该环境中，Agent应用其策略执行一组试验。每次试验，Agent从状态（1，1）开始，经历一个状态转移序列直至到达状态（4，2）或（4，3）。每个状态感知信息都用下标注明了所获得的回报。目标是利用关于回报的信息学习到与每个非终止状态相关联的期望效用。效用被定义为当遵循策略时所获得的回报的期望总和：

其中是状态的回报，是在时刻当执行策略时达到的状态，，在所有的公式中包含一个折扣因子 .

### 直接效用估计

**直接效用估计：把对一个给定状态全部观察到的未来回报用作学习效用的直接证据。**

直接效用估计成功地将强化学习问题简化为归纳学习问题，但是它忽略了一个重要的信息来源，即“状态的效用并非相互独立的”。每个状态的效用等于它自己的回报加上其后继状态的期望效用。也就是说，效用值服从固定策略的贝尔曼方程：

由于忽略了状态之间的联系，直接效用估计错失了学习的机会。我们可以把直接效用估计视为在比实际需要大得多的假设空间中搜索，其中包含许多违反贝尔曼方程组的函数。因此，该算法的收敛速度经常很慢。

### 自适应动态规划

**自适应动态规划：从观察中学习一个模型和一个回报函数，然后应用价值迭代或策略迭代获得效用或一个最优策略。**

就其价值估计的改进速度而言，ADP Agent受限于它学习转移模型的能力。从这个意义上，它提供了一个用以度量其他强化学习算法的标准。但对于大规模的状态空间来说，它是不可操作的。例如，在西洋双陆棋游戏中，将涉及处理大约个未知量的个方程。

### 时序差分学习

求解前一节内在的MDP并不是让贝尔曼方程承担学习问题的唯一方法，另一种方法是使用观察到的转移来调整观察到的状态的效用。

**基本思想：将效用估计朝着理想均衡方向调整，当效用估计正确时理想均衡是局部成立的。**

它的学习速度不如ADP Agent快，而且表现出更高的易变性，但是它更简单，每次观察所需的计算量也少得多。TD不需要一个转移模型来执行其更新。环境以观察到的转移的形式提供了相邻状态之间的联系。

ADP和TD方法是密切相关的。二者都试图对效用估计进行局部调整，以使每一状态都与后继状态相“一致”。一个差异在于TD调整一个状态使其与已观察到的后继状态相一致，而ADP则调整状态使其与所有可能出现的后继状态相一致，根据概率进行加权。由于转移集合中的每个后继状态的频率与其概率近似成正比，所以当TD调整的影响在大量的转移上计算平均的时候，上述差异便会消失。一个更重要的差异是，TD对每个观察到的转移都只进行单一的调整，而ADP为了重建效用估计和环境模型之间的一致性会进行尽可能多的所需调整。所以，TD可以被视为对ADP的一个粗略而有效的一阶近似。

## 主动强化学习

被动学习Agent有固定的策略决定其行为，主动学习Agent必须自己决定采取什么行动。

Agent同时必须还学习要做什么。主要的问题是探索：为了学会如何在环境中行动，Agent必须尽可能多地经历所处环境。

### 探索

**贪婪Agent：Agent在每一步都遵循其所学模型的最优策略的建议，在经历多次试验后，Agent一直坚持某个策略，再没有学习其他状态的效用，也从没有发现最优路径，称此Agent为贪婪Agent。**

greedy agent选择最优行动导致非最优结果，是因为学习到的模型与真实环境并不相同。ga忽视的是：行动不仅仅根据当前学习到的模型提供回报，也通过影响所接收的感知信息对真实模型的学习做出贡献。通过改进模型，Agent将在未来得到更高的回报。因此，一个Agent必须要在充分利用信息以最大化回报——反映在其当前效用估计上——和探索以最大化长期利益之间进行折中。

## 主要功能概述

本工程是根据爬虫爬取的数据来搭建学者信息大数据平台，主要功能如下：

1. 以表格的形式展示学者基本信息，这些信息包括姓名、h指数、g指数、论文数、被引用次数、群集度、多样性、活跃度等。
2. 以柱状图的形式展示各研究领域的人数。
3. 以饼状图的形式展示各研究领域所占比重。
4. 提供国际化功能，支持web页面中英文切换。
5. 支持学者信息模糊检索，信息字段列隐藏。
6. 支持学者信息的excel导出。
7. 提供学者详情页面，并为用户提供相似学者推荐。

## 论文主要工作

基于上述功能，以下是我进行该平台设计的主要工作：

（1）大数据平台搭建：无论是大数据的存储计算，还是分布式的爬虫，都对机器数量有一定的要求。这里使用四台Ubuntu服务器在同一个内网段下搭建Hadoop集群。

（2）数据获取：设计和实现分布式爬虫，利用该爬虫爬取学者信息，除了爬取姓名、论文数、h指数等基本信息，还要爬取该学者各研究方向的权重。

（3）数据处理：利用Hadoop的编程模型MapReduce对数据进行计算。包括根据h指数对学者排名、统计各领域研究人数以及各领域所占比重等。

（4）数据展示：通过web技术对处理后的数据进行可视化展示。

（5）web界面交互实现：提供图表自适应、excel导出、模糊查询、详情介绍、相似学者推荐功能。

## 本文组织结构

本篇论文总共有六个章节，按照整个项目的开发流程循序渐进。每一章的侧重点不同，具体内容分别如下：

（1）第一章是绪论部分，依次介绍高校学者大数据平台的研究背景、研究意义、功能概述和实现这些功能的主要工作。

（2）第二章是对平台的需求分析，并从不同的点出发，阐述开发该平台的可行性。

（3）第三章是开发此平台所用到的一些技术，做了一个系统的整理。

（4）第四章是针对平台的各项需求进行总体设计，主要包括分布式爬虫架构的设计、数据库的表格设计、MapReduce计算的设计以及web系统的设计。

（5）第五章是在进行整体设计后，进入各模块的编码实现阶段，并在遇到的实际问题中给出解决方案。

（6）第六章为总结和展望，总结开发这个平台所做的主要工作，并对该平台的发展进行展望。

# 需求分析与可行性分析

本章主要对高校学者信息大数据平台进行需求分析，然后在明确需求的基础上，从各方面分析开发此平台的可行性。

## 需求分析

### 功能需求分析

一般的学术网站大而全，专业性强，操作界面较为复杂，很难上手。对于普通用户而言，他们只想了解学者的基本信息以及当下有哪些热门的研究领域。基于平台简洁、交互友好的设计目

1. 的功能。
2. 当用户设备未联网，或者用户想要保存这些学者的信息时，需要提供excel导出的功能，这样方便用户离线查看。
3. 点击某行学者信息，提供学者详情页面，并为用户推荐相似的学者。

### 性能需求分析

开发这样一个平台，不管是数据的获取、数据的计算、网页的加载，还是数据的检索，性能都是要着重考虑的因素。

在数据获取方面，尽管爬虫是异步加多线程的，但是只能在一台服务器上运行，所以爬取效率是有限的。如

pReduce工程队的工作过程与分布式计算过程是基本对应的。

在网页加载方面，如果用至以后不

大概有长达四个月的时间供我去学习