

本科生毕业设计（学术论文）



题 目 **基于区块链的分布式算力汇聚方法**

学 院  **软件学院**

专 业  **软件工程**

学生姓名  **周林**

学 号  **2015141463261** 年级 **2015**

指导教师  **郭兵**

教务处制表

二Ο一九年五月三日

**基于区块链的分布式算力汇聚方法**

软件工程

学生 周林 指导老师 郭兵

**[摘要]** 随着互联网的高速发展，人们对互联网的日益依赖，全世界所产生的数据量也以惊人的量级增长。数据量的增大对计算机处理能力提出了更高的要求，分布式计算应运而生。区块链本质是一种分布式计算模式，同时伴随着区块链技术的火热，利用区块链技术的数据真实透明与共享、多方共识机制、多方安全验证等分布式处理模式用来解决机器学习的问题已经成为了一个新的研究领域。将区块链技术用于进行算力汇聚便是本文解决机器计算能力不足的思路。分配问题是一种NP问题，有很多研究工作者提出了各种智能分配方法，极具参考意义。本文主要对分布式算力汇聚方法进行了研究，利用虚拟货币和信任度的思想改进了合同网算法，并且使用改进后的算法搭建了区块链分布式实验平台进行实验。最后本文使用西雅图房价数据集实例进行机器学习建模，并且使用本文搭建的平台进行训练，结果达到目的。

**[主题词]** 区块链；合同网算法；机器学习；分布式算法；算力汇聚

**Distributed Power Convergence Method Based on Block Chain**

Software Engineering

Student： ZHOU Lin Adviser: GUO Bing

**[Abstract]** With the rapid development of the Internet, people become increasingly dependent on the Internet, and the amount of data generated around the world also increases by an astonishing order of magnitude. With the increase of data volume, the processing ability of computer is required to be higher. The essence of blockchain is a distributed computing mode. Along with the popularity of blockchain technology, it has become a new research field to use distributed processing modes such as data authenticity, transparency and sharing of blockchain technology, multi-party consensus mechanism and multi-party security verification to solve the problems of machine learning. Applying blockchain technology to computing power convergence is the idea to solve the problem of insufficient computing power of the machine in this paper. Allocation problem is a NP problem, many researchers have proposed various intelligent allocation methods, which is of great reference significance. This paper mainly studied the distributed computing power convergence method, improved the contract network algorithm by using the idea of virtual currency and trust, and built the blockchain distributed experiment platform for experiments with the improved algorithm. Finally, this paper USES the Seattle housing price data set as an example for machine learning modeling, and USES the platform built in this paper for training, and the results achieve the goal.

**[Key Words]** Block chain；contract NET protocol；machine learning；distributed algorithm；calculate force gathered

**目 录**

1 绪论 6

1.1 研究背景 6

1.2 国内外研究现状 6

1.2.1 分布式基本理论 7

1.2.2 算力汇聚方法 7

1.2.3 区块链应用 12

1.3 论文主要工作 12

1.4 论文组织与结构 13

2 算法基础简介 14

2.1 Agent技术 14

2.2 分布式机器学习技术 15

2.2.1 机器学习关键技术 15

2.2.2 数据并行和模型并行 17

2.3 基础知识 18

2.3.1 线性代数 18

2.3.2 概率论 18

2.4 本章小结 19

3 算法解决方案 20

3.1 相关技术 20

3.1.1 合同网模型 20

3.1.2 黑板模型 21

3.1.3 Paxos算法 22

3.1.4 Raft算法 24

3.1.5 总结 27

3.2 合同网改进模型 28

3.2.1 改进思想 28

3.2.2 数学建模 30

3.2.3 详细设计 31

3.3 本章小结 34

4 合同网改进模型实验验证 35

4.1 相关技术 35

4.1.1 区块链 35

4.1.2 以太坊 35

4.1.3 Truffle 37

4.2 实验环境 38

4.3 实验设计 38

4.4 实验结果与分析 39

4.4.1 实验过程数据 39

4.4.2 实验结果分析 42

4.5 本章小结 45

5 基于西雅图房价数据集实验 46

5.1 相关技术 46

5.1.1 梯度提升回归 46

5.1.2 数据可视化 46

5.2 数据预处理 47

5.3 实验设计 49

5.4 实验结果与分析 51

5.5 本章小结 52

6 总结和展望 53

6.1 工作总结 53

6.2 心得体会 53

6.3 进一步的工作 54

参考文献 55

致 谢 57

附录1 实验数据集地址 58

附录2 文献翻译（译文） 59

附录3 文献翻译（原文） 66

# 绪论

## 1.1 研究背景

随着互联网的高速发展，人们对互联网的日益依赖，全世界所产生的数据量也以惊人的量级增长。据统计，仅仅是2013年，百度、阿里巴巴以及腾讯的存储数据量都达到了数百PB（1PB = 1024TB）的级别。同时，随着国防、医疗、教育、工农业等国家计划的信息化，未来的数据只会越来越多。并且，增长的数据量意味着数据处理需要更高的计算能力，越来越多的计算任务也对计算的性能提出了更高的要求，分布式计算应运而生。分布式计算是随着大数据兴起的，在分布式计算中，重点之一便是任务的分配问题。分配问题是一种NP难度问题，有很多研究工作者提出了各种智能分配方法，极具参考意义[1-3]。

2008年11月1日，中本聪（化名）在网络上发表了“白皮书”（全名“比特币——一种点对点的电子现金系统”）。加入到比特币网络中的用户节点被称为“矿工”，“矿工”通过“挖矿”获得比特币，其中完成工作量证明（pow）的节点获得比特币奖励并且可以将新区块接入到区块链上去。伴随着比特币的概念被提出至今，区块链作为比特币的底层技术，受到人们的极度重视。区块链，本质上是一个去中心化分布式的账本。近年来，随着区块链技术的蓬勃发展，人们将区块链技术应用在金融、物联网等领域中。在2013年末，以太坊（Ethereum）作为“去中心化平台”的概念被提出，以太坊提供了一个可编程区块链的平台，开发者通过编写智能合约添加到区块链上，可以利用区块链搭建分布式的应用。官方推荐的编写智能合约的语言solidity是一种图灵完备的语言，因此在可计算理论中，以太坊可以解决所有的可计算问题[4-6]。

利用区块链技术的数据真实透明与共享、多方共识机制、多方安全验证等分布式处理模式用来解决机器学习的问题已经成为了一个新的研究领域[7]。在区块链中，每一个用户可以作为一个Agent，通过将区块链视作一个多Agent组成的可交互的松散的系统，我们可以利用其分布式、并行、可复用易扩展的特点。各个Agent节点从事简单的工作，通过分布式账本对每个Agent进行调用分配，便可以实现大型的计算任务。在前人研究的基础上，本文对在区块链环境中的智能分配算力汇聚方法进行研究，具有一定的创新性和挑战性。

## 1.2 国内外研究现状

区块链本质是一种分布式计算模式，分为私有链、联盟链和公有链三种形式。对于区块链的研究热潮是近年来才兴起的新兴领域，其易于结合各行各业、公开透明、分布式的特性被众多研究者所追捧。分布式计算一直是国内外的研究热点，随着摩尔定律遇到了瓶颈，一味的缩小体积遇到了物理学上的限制。为了摆脱这种约束，采用大规模分布式的计算成为了处理大数据的有效模式。

下面对分布式计算的一些重要理论、算力汇聚方法以及基于区块链的一些应用作出简要介绍。

## 1.2.1 分布式基本理论

**（1）FLP不可能性**

在物理学上存在海森堡提出的“测不准原理”，而在分布式计算领域，同样存在“FLP不可能性原理”。该原理指的是，在一个异步通信场景中，对于任意一个节点的失败，也没有任何算法能保证非失败节点达到一致性。该定理由Fischer, Lynch 和 Patterson三位作者联合发表，并且获得了Dijkstra奖。

该定理告诉了我们一个基本事实：不要浪费时间去设计一个能使得异步分布式系统，在任何场景下都能保证一致性的算法。尽管理论事实很悲观，万幸的是在工程上，我们可以通过其他尝试的方式尽量满足一致性，只不过需要多付出一点代价罢了。至于代价，在下一小节介绍CAP原理时将作出阐述。

**（2）CAP原理**

CAP原理（Consistency，Availablity，Partition）表明了，对于分布式系统中，我们不可能同时满足一致性（Consistency）、可用性（Availablity）和分区容忍性（Partition），这三者不可兼得，如果想要强化某一个属性，则只能弱化其他属性的实现。

一致性：在分布式系统中，各个节点是否同一时刻有同样的值；

可用性：有限时间以内，是否任何非故障节点都能提供应答；

分区容忍性：由于节点之间的通信不可保障，因此系统需要对通信的时限进行容忍。

在设计系统时，我们往往需要牺牲系统对某个属性的支持，而选择牺牲哪一个特点需要根据具体的系统功能而定。

**（3）ACID原则**

指的是对分布式数据库事务的一致性需求，由以下四个特性组成：

原子性（Atomicity）：对于某一事务里的所有操作来说，要么全部操作完成，要么整个事务失败；

一致性（Consistency）：数据库状态是一致的，不存在中间状态；

隔离性（Isolation）：各个操作之间相互隔离，互不影响；

持久性（Durability）：状态的改变不会失效，能够保持持久。

## 1.2.2 算力汇聚方法

目前国内外存在多种任务分配方法，该小节对他们进行一个总体的梳理，在第三章中会对几个重要的分配方法进行详细介绍。首先总的来说，分配方法主要包括以下几个方面：

1. **规划方法**

所谓规划方法，即是在一定环境的限制下（如资源限制），利用数学建模，用运筹学的方法对整个模型进行评价。通过模型建立，便得以得到一个最优化的模型来进行任务分配。目前存在的常用规划方法有：

·匈牙利算法

为了描述匈牙利算法不得不提及指派模型。指派模型指的是存在多个任务，然而各个任务的需求以及耗时都不同，同时存在多个节点，各个节点属性以及能力也不同。由于任务的性质，我们需要指派每个任务给某一节点完成，该模型便是指派模型。匈牙利算法的核心是寻找增广路径，它利用其来求得二分图的最大匹配。利用匈牙利算法，根据效益矩阵可以尽可能的将合适的节点与任务相匹配起来，便得到了最优解[8]。

·整数规划

所谓整数规划，需要从建立目标函数开始，且规划中的变量均为整数。组合最优化的问题通常都可以表述成整数规划的问题。定义代价函数为任务总费用（费用可以包括执行时间、资源消耗等），同时定义诸如任务顺序和任务消耗等约束条件，建立规划模型求解。0-1规划便属于整数规划，由于其变量值只包括0和1，故得其名。0-1规划可以将多种非线性规划问题表示为整数规划的问题，在整数规划中占据重要的地位，常用来解决指派问题。

1. **搜索算法**

搜索算法指利用高效计算，通过输入问题的所有或者一部分解空间，然后得到问题的解决方案的一种算法。搜索算法的目的在于利用初始条件构造搜索树，并且找到与目标最符合的节点。首先第一步是构造搜索树，并且根节点为初始状态，叶节点为最终状态，同层的为兄弟节点，上层是下层的父亲节点。搜索过程便是寻找最佳匹配的叶节点的过程，该过程由于计算量太大，有必要进行剪枝操作，同时也往往会牺牲全局最优解来使得函数快速收敛。

1. **图论**

图论（Graph Theory）以图为研究对象，用点来表示事物或者元素，用线来表示两者之间的关系。图论中的任务分配主要包括以下模型：

·网络流模型

网络流是一种很实用的图论模型，类比水流的解决方法，产生了诸如具有增益的流、多商品流、多终端流等等新领域。同时，网络流应用遍布电力、通讯、任务分配等领域。网络流是图论中的一种理论方法，用以研究网络上的最优化问题。在1956年，福特和富尔克森创建最大流理论，设计了最大流算法。最大流算法解决了在给定的两点之间寻求最大运输量的问题，与运筹学、线性规划问题密切相关。

·偶图匹配模型

偶图匹配是一种二分图模型，对于偶图来说，存在两种节点集分别表示任务集合和处理任务的节点集合，用线段来表示任务和节点之间的从属关系。偶图匹配模型通过根据任务和节点之间的特征建立交互图，然后利用规划方法（如匈牙利算法）建立从任务集合到节点集合之间的映射关系，达到最小化任务总费用的目的。

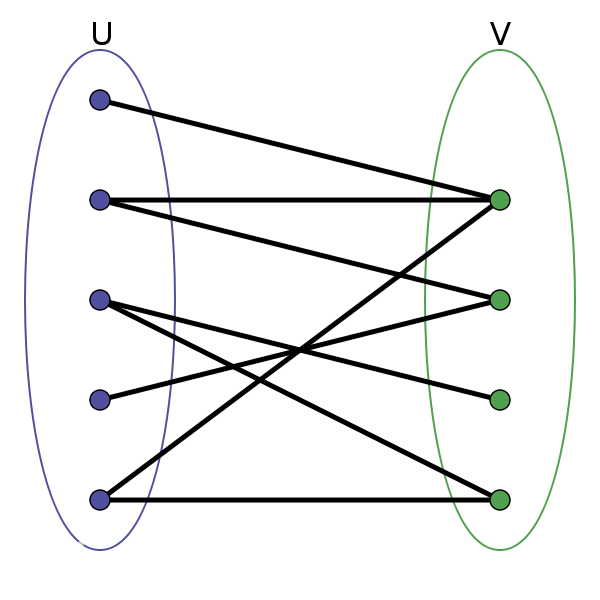


图1 二分图示例

1. **负载均衡**

负载均衡关注点在于使得各个节点处理任务的压力一致。学者郑绪釉总结了负载平衡方法[10]，该方法根据各个节点当前负载主动分配任务，假设各个节点在负载均匀时效果最好，最大限度的提高系统的资源利用率，从而减少单位节点任务的响应时间。为了最大化负载平衡模型的收益，学者设计了很多诸如最短时间的启发式算法、XY算法、异构分布式实时动态的平衡算法等。同时，负载平衡还包括动态负载平衡、静态负载平衡和自适应负载平衡。

1. **经济学方法**

经济学方法利用了市场化模型，基于经济学理论提出，合同网模型也属于经济学方法。其中合同网模型将在第三章详细介绍，这里主要介绍拍卖算法与市场平衡模型。

·拍卖算法

拍卖算法模拟了现实中的市场拍卖规则，买家可以出价竞争竞拍商品，最终根据出价规则系统达成交易。在任务分配系统中，拍卖的物品被具象化成任务，买家变成了各个节点，拍卖算法被广泛应用于电子商务、电力供应、网上零售等应用之中。传统的拍卖方法包含英格兰式拍卖、荷兰式拍卖以及英格兰和荷兰结合式拍卖、美式拍卖等。

英格兰式拍卖也叫增价拍卖，指的是在拍卖过程中，拍卖人规定起始价以及最低涨幅，各个竞拍人可以依次加价。最后在竞拍人确认三次无人加价后，商品由最后出价者得。与之相反，荷兰式拍卖被称为降价拍卖，原因在于在荷兰式拍卖过程中，拍卖人先确定商品的起始价以及降幅，由第一位竞拍人成交，且成交价不得低于保留价。英格兰和荷兰结合式拍卖则是将二者结合，先由拍卖人确定起价以及最低涨幅，竞拍人依次升高叫价，若无人竞拍，则进入二阶段拍卖人确定降幅，第一个竞拍人成交，且成交价不低于保留价。当想要拍卖特定的商品时，可以采用定向拍卖的方法，竞拍者只有满足拍卖人的一定条件后才可以参与竞拍活动。同时拍卖方法中还存在反向拍卖，其指卖家之间通过竞争的方式决定使用谁的产品，让买家获得最优质的体验。

由于其简单有效的特点，拍卖算法已经成为了人工智能与分布式算法的研究热点。拍卖算法往往是和其他方法结合使用，可以取得很好的效果。

·市场平衡模型

市场平衡模型是指通过市场的平衡方法来实现任务分配。在现实的经济活动中，市场往往处于一个供需平衡状态。首先商品的基本属性是价值和使用价值，价值和供需关系决定了商品的价格。当商品供不应求时，该商品价格会上涨，当价格涨到一定程度时商品需求量又会降低使得其不得不降价，最终市场趋于动态平衡。同理，各个Agent节点通过调整自己的出价策略与定价策略，最终系统会达到一个平衡状态，该状态有以下限制条件：市场所有商品均售完；各个消费者购买商品效益达到最大；各个生产者分配的效益也达到最大。

1. **其他方法**

·基于随机博弈的任务分配方法

博弈论是数学的一个分支，在社会生活经济活动中占据重要地位[11]。在博弈论中存在纳什均衡的重要概念，纳什均衡指一个博弈的解，即在其他人保持策略不变的情况下，为了自身利益所有人都不会改变自己的策略。

在每次任务分配时，节点之间对于任务的竞争会构成博弈关系。根据随机博弈纳什均衡存在定理，该随机分配模型存在纳什均衡解。为了解该均衡模型，各个节点相互之间需要通信得到对方的信息，并且维护自身的信息，通过数学建模的方法实现解的过程。

·蚁群算法

蚁群算法是模拟了蚁群寻找食物的过程，得到的最佳路径的算法，可以用来解决云计算中任务分配的问题[8,9]。众所周知，蚂蚁在行动途中会留下信息素，从蚂蚁巢穴到食物的途中，不同的蚂蚁通过不同的路线找到食物会使得信息素分布在不同的路径上。由于在单位时间内更短路径上会存在更多的蚂蚁，使得短路径的信息素浓度比长路径的浓度高，于是蚂蚁们都会大概率的聚集在信息素浓度高的地方，这是一种正反馈的行为，于是便得到了最短路径。

蚁群算法是一种启发式算法，运用蚁群算法可以实现负载均衡，使得系统集群的性能达到最好。首先将任务作为蚂蚁的起点，执行结果作为蚂蚁的终点，而最好的分配方式便是蚂蚁寻找最短路径的过程。

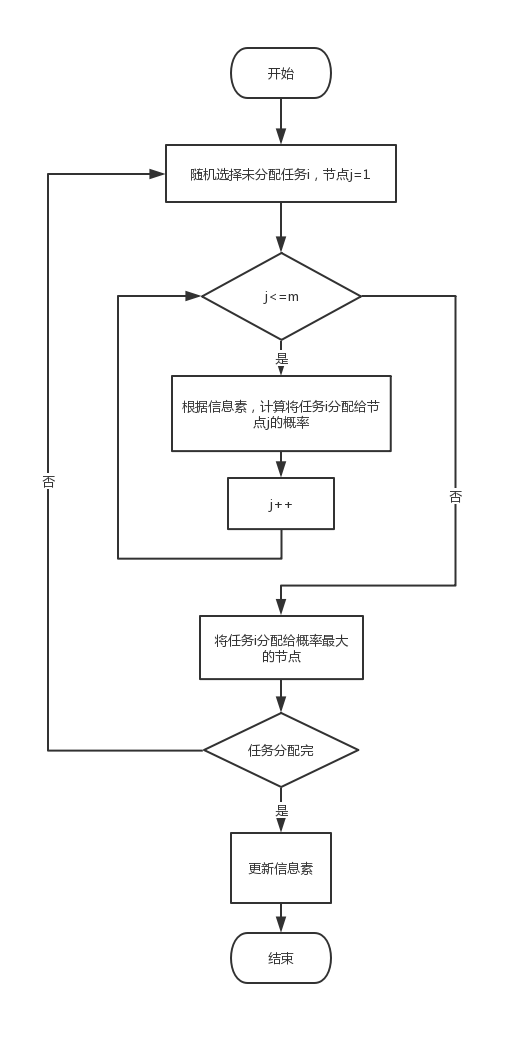


图1.1 蚁群算法流程图

**·**基于马尔科夫链的决策模型

1907年，俄罗斯数学家马尔科夫总结了自然界与现实生活中的普遍现象，提出了基于随机过程的马尔科夫模型。在马尔科夫模型中，状态的变化只依赖于当前环境与行为，与过去的状态无关。现实生活中很多现象都可以用马尔科夫模型来解释，比如食堂排队的人数、小学生跳楼梯的动作等。马尔科夫模型主要分为MDP、POMDP以及DEC\_POMDP模型。通过马尔科夫模型来解决任务分配问题，主要是利用策略π解决问题是一个迭代的过程。首先，节点观察到现在的环境e，然后根据环境按照π(e)来执行动作得到回报，并且系统发生状态转移进入下一个状态，再观察环境并迭代执行前一个操作，直到问题解决为止。

为了评价动作好坏需要定义评价函数，各个阶段的动作产生的回报之和构成了评价函数，按照该函数的最大值执行动作便可达到最佳的分配模型。

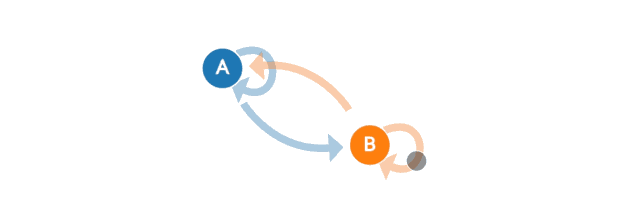


图1.1 马尔科夫模型示意图

## 1.2.3 区块链应用

（1）金融服务

中国央行行长周小川曾表示，央行可能使用区块链技术支持数字货币，彻底改变传统货币的格局，目前已经有专门团队进行评估。

据2016年报道，加拿大央行正在基于区块链开发数字货币加拿大元（CAD币）。

2016年，中国邮储银行宣布推出基于区块链的资产托管系统，新用系统取消了重复的信用校验，大幅度提升了原系统的性能。

（2）众筹交易

作为互联网+金融的代表——众筹交易，由于其对于数据信息透明化的需求，相当适合区块链系统。目前，存在很多公司诸如京东金融、DAO等等众筹平台，这些平台也对区块链技术表现了充分兴趣。

（3）社交

BitMessage：一款基于区块链的p2p安全可靠的发信系统。

Twister：一款p2p的微型“微博”系统，由于其完全去中心化的特点，不需要受到管理员的权限限制。

## 1.3 论文主要工作

本文主要针对基于区块链的分布式分配算法，进行了以下工作研究：

1. 基于以太坊搭建私有链，使用私有链实现了分布式算力汇聚系统。
2. 实现合同网分配算法以及改进，并给出了相关证明过程。用改进后的分配算法搭建分布式系统并进行实验与分析。
3. 基于数据并行，使用已完成的系统实现支持对“西雅图房价数据集”的回归分析。用实验评估改进分配算法的效果。

## 1.4 论文组织与结构

本论文由六个章节组成，文章的主要内容及作用分别如下：

第一章：绪论。对课题的研究背景、国内外研究现状进行文献综述，对该论文的主要工作做了阐述，还对论文的组织与结构做出了描述。

第二章：算法基础。对本文涉及到的一些理论、概念、技术做出了简要介绍。为第三章进行算法改进提供理论基础。

第三章：算法解决方案。比较评价了当前已有的4个解决方案，改进了合同网算法，提供了数学建模并给出相应的伪代码以及流程图。

第四章：改进合同网模型实验验证。测试了本文提供的算法，对系统处于信任和非信任环境进行了实验，分析与评价了实验结果。

第五章：利用梯度提升回归，通过西雅图房价数据集，使用第五章实现的系统对西雅图房价进行建模。对比不同平台训练的结果，并且对实验结果进行分析与评价。

第六章：总结和展望。对该课题的研究工作进行了总结，对自己的心得体会的记录，以及项目的未来的进一步工作。

# 算法基础简介

## 2.1 Agent技术

Agent是面向对象技术的抽象，是一种处于特定环境下的计算机系统。在基于网络的分布式计算中，该技术发挥着越来越重要的作用，因此得到了高度重视，甚至被一些文献视为软件领域的下一个重大突破。与对象相似的是，它封装了属性和方法，同时还具有决策能力，有良好的灵活性和自洽性。Agent技术为解决分布式计算系统提供了一个有效的概念模型，利用该技术我们可以实现多种分配方法。通常认为Agent包含有以下四个特性（目前学术界并没有准确定义）：

1. 自主性。Agent在没有外界环境干涉的情况下，能够根据自身内部状态和外界环境进行合适的行为。
2. 交互性。Agent能够与其他Agent进行交互，类似于现实世界，形成一个个组织以及团体进行活动。
3. 主动性。Agent能够主动给其他对象发送信息，一个Agent可以主动请求对方，实现自己的目的。
4. 反应性。像其他生命体一般，Agent可以对外界环境做出相应的应激行为。对于用户的操作，Agent可以做出相应的反应。

例如，在足球机器人赛中一个机器人就可以看做一个Agent，各个Agent之间相互配合共同完成进攻或者防守任务，共同目的是实现进球。个体Agent的行动示例如图2.1所示。

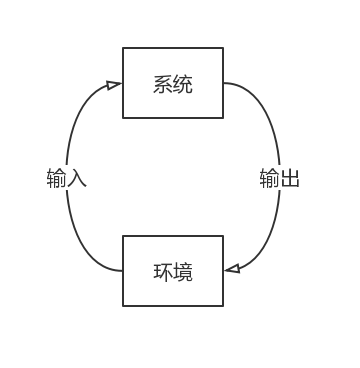


图2.1 个体Agent行动示例图

近年来，人工智能学者通过对单个Agent的技术理论的研究，掌握了很多对于单个个体行为特性的推理和判断的方法。而在真实世界中，往往是各个Agent之间互相影响，并且就如马尔科夫链一般存在状态转移，因此对于多Agent系统的研究才能满足现实的问题的需求。

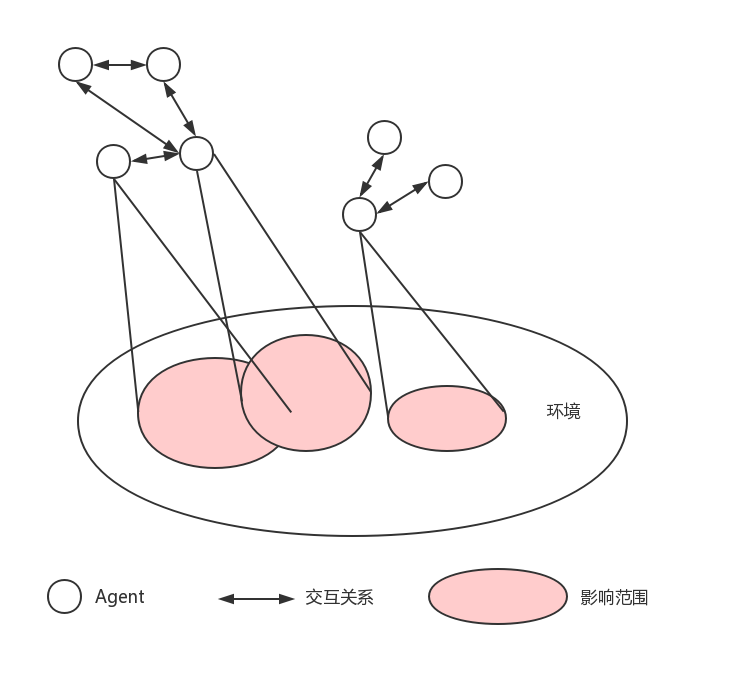


图2.2 多Agent系统示例图

在多Agent的系统中，模仿人类和社会的生活活动具有很大的借鉴作用，但是不能得出完全用同样的方式构建人类社会的结果。自然界和人类社会受到各种各样的制约影响，差之毫厘失之千里，尽管如此，用多Agent的方法来模拟社会现实也是有积极意义的。

Agent技术是如此的模拟现实环境，因此对于区块链来说，每一个节点就相似于一个Agent，基于Agent模型的思想我们可以实现良好的分布式智能分配算法。

## 2.2 分布式机器学习技术

## 2.2.1 机器学习关键技术

机器学习属于人工智能的一个分支，机器学习包含了多个学科的内容，如统计学、概率论、社会学等等，是一门多领域多学科的交叉学科。在历史上，人工智能的发展可谓是一波三折（见图2.3）。1950年，“人工智能之父”图灵（Alan Mathison Turing）通过自己进行的数理逻辑方面的研究提出了著名思想实验—图灵测试的概念，即：在人与机器隔开的情况下，自愿者与机器交流，如果有30%的自愿者不能判断与之交流的是人还是机器，那么这台机器被认为通过了图灵测试，具备人工智能。此后学者们对于人工智能的研究呈现出一种热火朝天的态度。1956年，计算机科学家约翰·麦卡锡在一次会议上提出了“人工智能”的名词，在此之后AI走上了快速发展的道路。在这之后十多年时间里，计算机被广泛应用与诸如自然语言处理、数学、生物学等领域，有学者大胆预言认为机器学习实现人工智能并不遥远。然而到了70年代，人工智能陷入了第一次低谷时期，相关进展缓慢，且计算机计算能力不足与问题太过复杂限制了机器学习的应用，很多人意识到机器学习被媒体夸大，开始冷静下来思考对策。到1980年人工智能又开始复苏，可惜仅仅维持了7年左右人工智能又回到了原点，似乎AI的概念只能存在于科幻作品之中了。AI的再次崛起是在20世纪90年代中期，随着神经网络的发展，AI技术开始进入了稳定发展的时期，机器学习进入了新阶段。1997年IBM的计算机“深蓝”在国际象棋领域战胜了当时的世界冠军卡斯帕罗夫，AI的概念开始为大众所熟知。深度神经网络研究的突破进展也吸引了诸如Google、百度、阿里巴巴等资本巨头的加入，在资本的推动下机器学习呈现一片蓬勃之势，被视为未来的发展方向。2016年，Google推出的“AlphaGo”以压倒性优势击败世界冠军李世石，此后又击败中国国手柯洁，此后在公众中彻底爆发了AI热潮。目前看来AI技术发展蓬勃，这第三落当然也未必会到来。

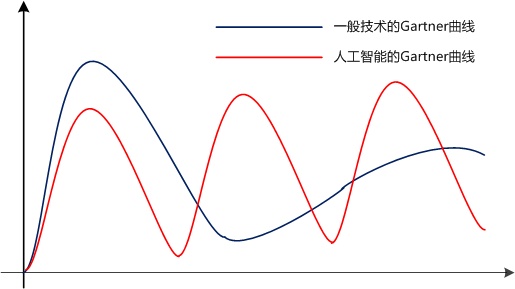


图2.3 人工智能发展曲线

对于机器学习算法来说， 我们一般通过建立一个模型，得到目标函数，再通过对目标函数求最大（小）值来训练参数。

在模型训练过程中，我们往往建立一个代价函数（cost function）来衡量该模型对真实世界的误差大小。因此，原问题则转化为了一个求代价函数最值的问题。在求多元函数最值问题上，存在多种方法，比如随机选取起始点的“梯度下降法”、可以概率性的跳出局部最优解的“模拟退火算法”等等。在机器学习中，我们常常用到的是随机梯度下降法，下面将简要介绍随机梯度下降法。

随机梯度下降是机器学习中一种常用的求函数最值的方法，与批处理梯度下降不同的是，随机梯度下降关注于训练数据集中的每个样本，而不是全体样本。每次迭代，该算法便更新待定参数，直到训练完毕。梯度下降的思想：

1. 对目标函数各个参数求得梯度。
2. 将参数向量根据梯度进行更新，得到新参数。
3. 将新参数带入代价函数进行计算，若满足给定误差，则退出，否则重复步骤1。

梯度下降的迭代过程数学表示如下

 （2.1）

其中，为需要求得的参数向量，为学习率，为参数向量的梯度。梯度下降法的缺点在于容易陷入局部最优解，尽管如此，梯度下降法也极其常用，被广泛运用于机器学习的各个领域。注：反方向即是梯度上升。

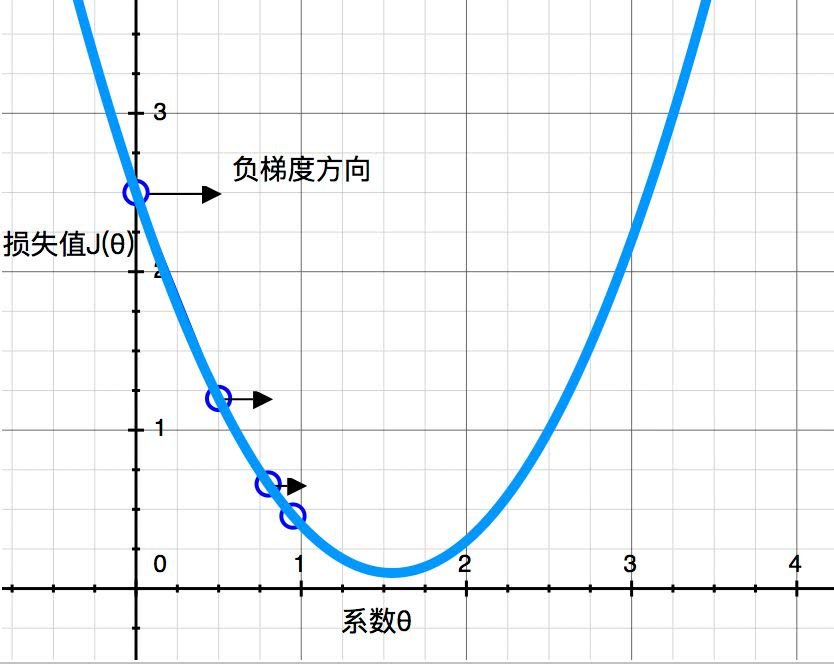


图2.4 梯度下降示例图

## 2.2.2 数据并行和模型并行

在分布式机器学习算法中，存在两种并行方式：数据并行和模型并行。

1. 模型并行

对于神经网络来说，训练过程中往往具有高度的非线性。如最简单的BP神经网络，每次迭代过程可以简单表示为：

1. 在前向传播中将训练参数输入网络并获得激励响应。
2. 在反向传播中将响应和目标输出求差，得到响应误差。

对于此类神经网络，由于存在神经之间的紧密关系与迭代，往往通过模型并行来支撑大规模的训练，模型并行将各个机器作为节点，负责处理不同层次的问题。

（2） 数据并行

本文主要采用数据并行的方式支撑机器学习的计算，同时这也是大多数场景的并行方式选择。数据并行将数据分配到不同的机器，然后将计算结果通过某种方式进行合并。数据并行由于需要组合不同的机器的训练结果，被分为了各种方法：

1. 参数平均法。将各个机器得到的训练结果取平均值，当做全局的参数值。
2. 异步更新法。与参数平均法不同的是，在工作节点与全局服务器之间，只传递更新的信息，如梯度。

基于数据并行，我们可以实现异步随机梯度下降，对于求大规模数据的梯度下降算法有极大的意义。

## 2.3 基础知识

## 2.3.1 线性代数

本文在研究分配关系时主要使用线性代数作为工具，因此在这里对一些简单的概念做出介绍。在下一章本文用到了矩阵的迹的概念。

在线性代数中，一个N阶矩阵A的主对角线的所有元素的和被称为矩阵A的迹，记做。

 （2.2）

迹是所有对角元的和，也是所有特征值的和。

## 2.3.2 概率论

本文实验部分对于概率论主要运用到了二项分布的概念。二项分布即n次重复实验的伯努利分布，当实验次数为1时，二项分布符合0-1分布。我们用X表示随机实验的结果，假设事件发生的概率为p，那么不发生的概率为1-p，对于n次独立重复实验来说，发生k次的概率为：

 (2.3)

其数学期望为：

 （2.4）

其方差为：

 （2.5）

## 2.4 本章小结

本章对该文所涉及到的一些理论、概念、技术知识进行了一个简要介绍。

Agent技术是本文的主要思想，通过将每个节点视为一个Agent，我们能够更好的描述节点的行为。比如，描述下文算法解决方案时，我们便将参与计算的节点视为Agent，用来描述分布式智能分配算法。

目前学术界主要将分布式机器学习技术分为模型并行与数据并行两个模式，我们主要使用数据并行的方式来分析评价我们的模型。

# 算法解决方案

## 相关技术

为了实现算力汇聚，我们需要将一个大任务分解成一个个小任务，然后将每个小任务分配给不同的节点协作完成。任务调度算法主要分为基于效率的调度算法和基于公平的调度算法两类，有启发式遗传算法、蚁群算法等[27]。目前学术界存在很多分布式分配算法与模型，比如合同网模型、黑板模型、paxos算法、raft算法、基于马尔科夫链的决策模型、基于随机博弈的任务分配方法等。这些方法各有各的特点，本小节对几个应用广泛的模型做一个简要的介绍与评价。

## 3.1.1 合同网模型

合同网协议（contract NET protocol，CNP）是分布式人工智能领域中的一种经典协调协议，它为人与资源、Agent之间的协调发展提供了一个良好有效的相处方式[10-11]。合同网模型拥有管理者（Manager）和投标者（Contractor）两种属性的Agent，模拟了经济学中的招投标过程，通过管理者审核投标者来选择任务的发放。

**分配过程**

合同网模型可以分为4个阶段

1. 招标阶段：管理者将需要指派的任务进行发布给投标者。
2. 投标阶段：投标者收到管理者发布的任务后，参与投标，和其他投标者一起竞争该任务的归属。
3. 发放合同阶段：管理者审核投标者的标书，选择它认为最适合的投标者发放任务。
4. 签约阶段：被选中的投标者收到管理者发放的任务，和管理者签约任务的执行。

Agent之间对单个任务交互协商如图3.1所示。

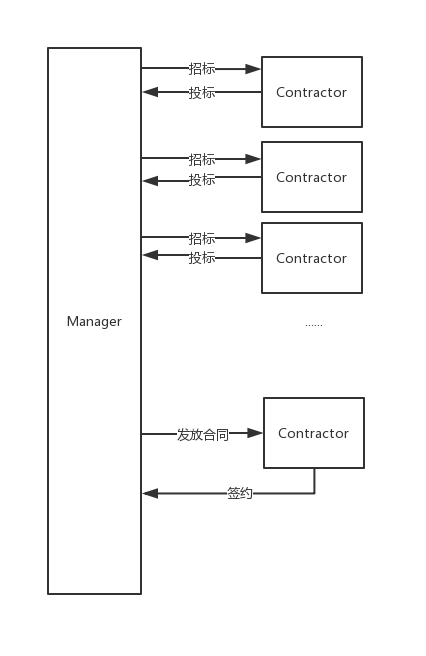


图3.1 合同网模型任务交互协商图

合同网模型应用广泛，其结构流程简单而又有效，受到了广泛的追捧。

## 黑板模型

黑板模型是一种协作模型，一般不会对架构产生大的影响，但对于消息结构有一个清晰的要求[21]。在黑板模型中，黑板是一个共享问题的求解空间，Agent之间共享一块黑板。当某一个Agent解决了自己的问题，便将结果公布在黑板上，公布的信息可能会支持其他Agent求解其他任务。黑板模型包含以下几个模块：

1. 黑板：黑板可以视作一块共有的数据缓存带，所有任务信息以及Agent的处理结果都会按层次被记录在黑板上。黑板是任务的公告板，也是一个生产者，为消费者（Agent）提供待处理的项目；
2. 控制单元：控制单元负责黑板与Agent之间的交互，将生产者（黑板）和消费者（Agent）之间的耦合度降到最低。当Agent在控制单元注册之后，才能够调用控制单元的方法。控制单元直接和黑板交互，类似于中间层的存在；
3. Agent：Agent是计算的主体，控制单元和黑板都是为Agent服务而存在。Agent与控制单元交互来获取任务与提供计算结果。

黑板模型结构如图3.2所示。

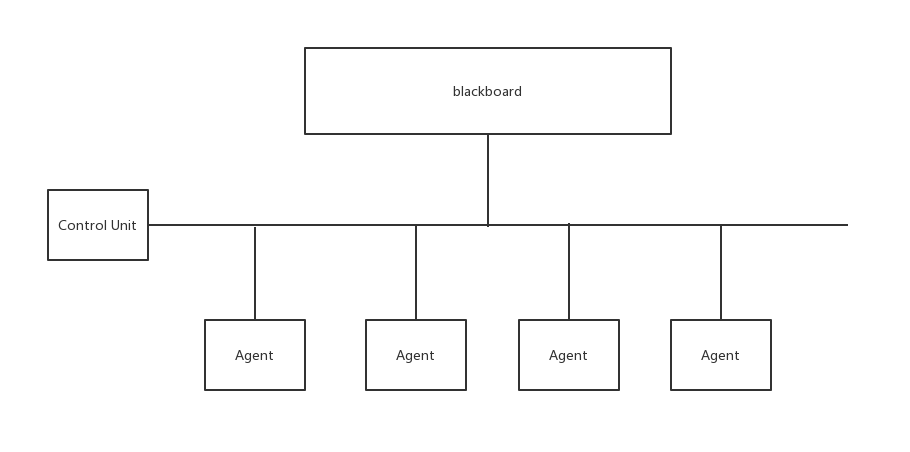


图3.2 黑板模型结构图

黑板模型极其常见，其思想对于多个节点如何协作起了指导作用。

## Paxos算法

Paxos是一种很复杂的算法，它是于1990年提出的一种基于消息传递并且具有高度容错性的一致性算法[22-23]。为了便于理解它的高度容错性的特性，我们使用一个形象的例子来理解：假设有一家人打算假期去游玩，但是目的地并没有确定，于是最简单的解决方案便是在微信群里投票，按照少数服从多数原则决定目的地。这是一种类似于“黑板模型”的共享内存的场景，然而Paxos考虑的复杂得多，为了实现高容错性，该算法假设如果微信服务器发生故障该怎么办，如果一家人中有一半人暂时无法取得联系，又该如何得出一致性的结果。

Paxos基于**两阶段提交协议**扩展而来，算法将节点分为了三种类型：

提议者（Proposer）：提出议案；

决策者（Acceptor）：对提议者提出的议案进行投票；

最终决策学习者（Learner）：不参与投票，只为结果做出统一性处理。

对于**两阶段提交协议**而言，整个过程可以简单的分为两步：

1. 准备阶段：提案者将自己的提案发送给决策者，观察决策者的态度，准备进入第二阶段。而决策者保留收到过的最晚的提案和接受的最大提案；
2. 提交阶段：对于某个提案者而言，提案者如果收到大部分的回复，说明它具有一定的话语权，则可以提交它的议案。然而若最终决策者得到的提案不是它之前发过的提案，它则转头支持其他提案。对于决策者而言，如果发现提案大于已经接收过的提案，则接收之并更新，反则弃置之。

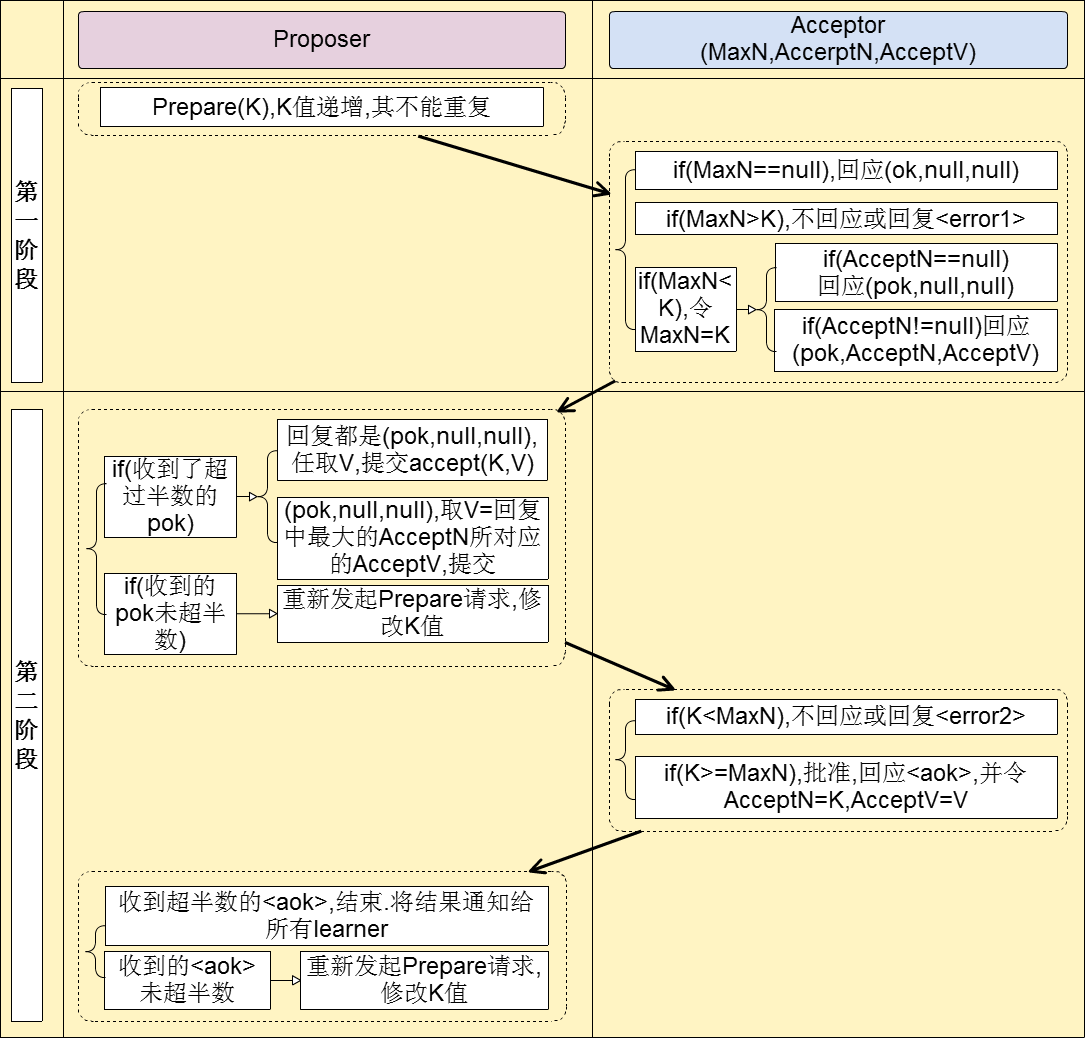


图3.3 paxos流程图

Paxos的交互主要发生在提议者（Proposer）与决策者（Acceptor）之间，在这两者之间交互主要由四类消息组成，如图3.4所示。这四种消息分别对应两个阶段的四个过程：

1. 准备阶段

* 提议者向网络里的一半以上的决策者发送Prepare消息；
* 决策者正常回复promise消息

1. 提交阶段

* 在大部分决策者回复promise消息后发送accept消息
* 决策者正常回复accept消息

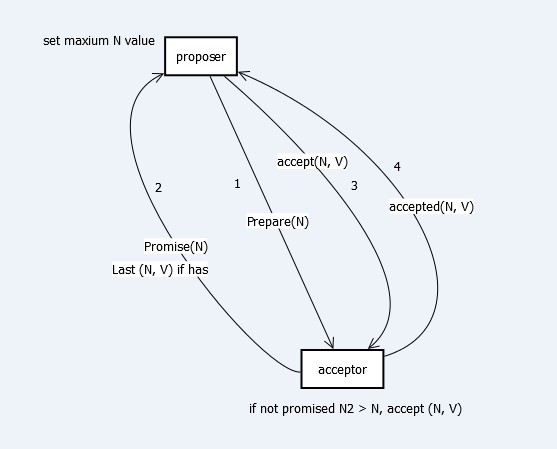


图3.4 Proposer消息通信图

最终，算法一定能得到一个最终提案，该最终提案则成为了所有节点最终的决定。然而该算法最初的论文便以复杂难懂闻名，且难以实现。根据CAP（Consistency, Availability, Partition tolerance）原理，Paxos为了保持高度的一致性牺牲了可用性与性能，这是我们不愿意接受的。

## Raft算法

Raft算法是Paxos的一种简单实现，于2014年被斯坦福大学教授发表。它将节点分为了三种类型[24-25]：

Leader：处理各个节点之间的交互，当Leader不可用时，需要选出新的Leader；

Candidate：候选人，当Leader失效时，拥有被选为新的Leader的机会；

Follower：选民，负责投票选出Leader。

在Raft算法中，任何节点都有机会成为Leader。在一届任期内，整个算法分为两个阶段，一是选举过程，二是选举出来的Leader带领其他节点进行正常的操作。Raft算法有很多实现，与Paxos算法比起来更易理解。

（1）选举阶段：首先一组节点的状态都是Follower，每个节点都有一个倒计时工具，在某一个节点的倒计时结束后，它的状态变为Candidate并且会对其他Follower发送选举请求，让其他节点选举自己成为Leader。若在倒计时结束之前，有一半以上的候选人表示了支持，那么该节点就会被选为Leader。

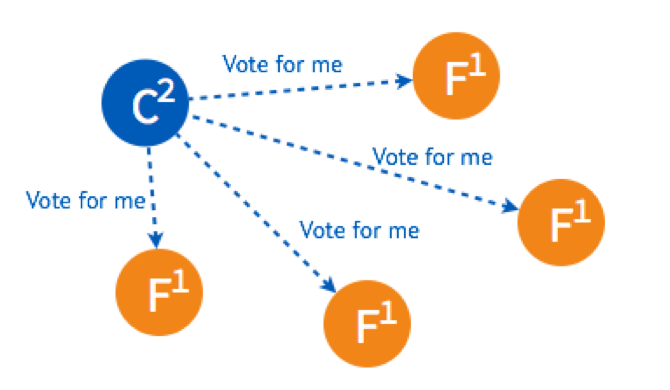


图3.5 正常节点选举图

若存在节点故障，那么其余节点将重新选择Leader。

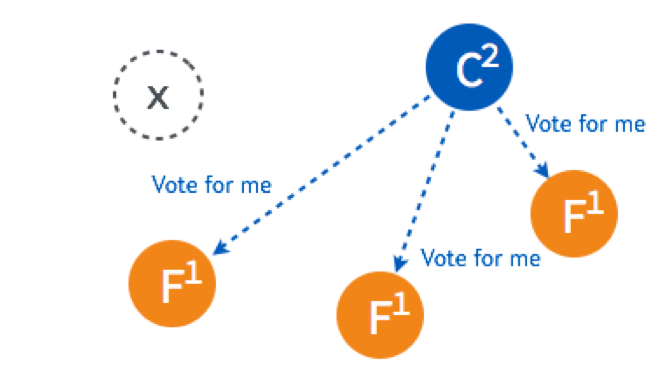


图3.6 节点故障选举图

同时，若是有多个候选者竞争成为Leader，由于无法得到半数的票数，那么将继续运行倒计时，最先Timeout的节点将继续发出邀请投票请求，直到选出Leader为止。

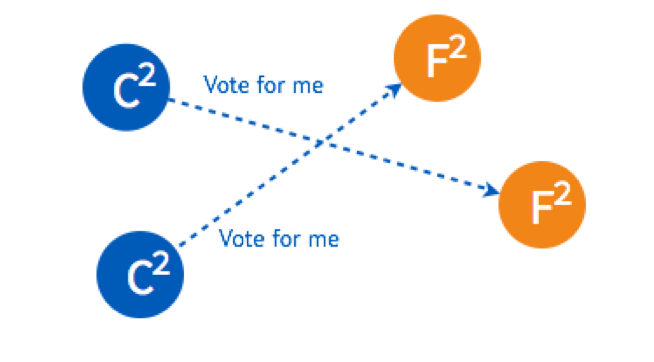


图3.7 多Candidate选举图

（2）复制日志：为了保持一致性，使得任一节点出故障后系统依然能够正常运行，raft在选择Leader后会进行复制日志操作。刚开始所有节点都没有任何数据。

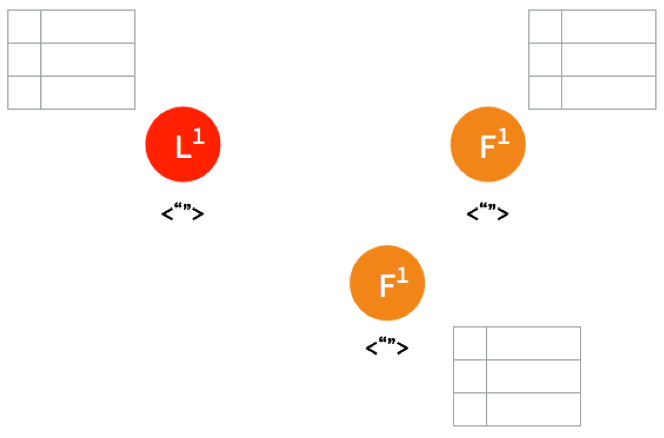


图3.8 日志-空数据

然后客户端请求存储字符串“sally”，此时Leader将数据存入本地，但是数据还没有得到最终的确认，状态为未提交（Uncommitted）。

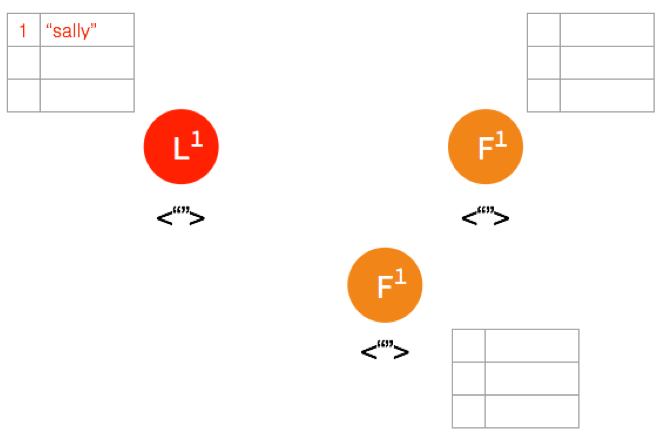


图3.9 日志-Leader保存未确认

Leader此时向选民（Follower）发送请求，若数据未在选民日志数据库产生冲突，便将数据写入日志。当选民将数据写入本地后返回确认信息给Leader，并且Leader收到超过选民半数的确认信息后，Leader便将数据状态改为已提交（Committed）状态。

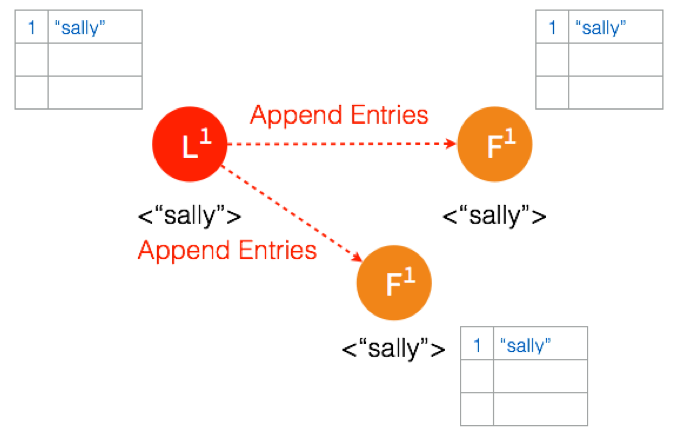


图3.10 日志-状态已提交

## 总结

对于本文讨论的场景而言，其环境远没有paxos设想的那么复杂，考虑到简单易于实现的原则，本文选择合同网作为基础理论，实现适合的分配算法。合同网本身是一个简单的模型，对于现实中存在的很多问题并没有得到很好的解决（如：成本与时间的分析、恶意Agent问题），为了更好的将合同网运用到实际情况，本文对合同网模型进行了改进与优化，下一小节将主要讨论合同网改进模型。

## 合同网改进模型

## 3.2.1 改进思想

传统的合同网模型仅仅给出了分配问题的框架，然而，对于成本与时间的分析、招投标过程中Agent之间通信的损耗、恶意Agent参与竞争的问题并没有得到很好的解决。对于区块链而言，“矿工”参与运算可以得到虚拟货币的奖励，并且对于区块链上的某个节点来说，它可以与其他节点进行交易，同时，可能会有恶意矿工伪造投标消息参与竞争。

因此，为了更好的适应区块链的生态环境，本小节对合同网算法进行了本地化的改造，通过引入奖励机制“虚拟货币”、“信任度”的思想提出了新的模型。

1. 虚拟货币

对agent节点参与分布式计算的奖励，我们按任务复杂度分配给不同任务不同的奖励。事实上，我们会将高奖励的任务分配给计算能力强同时信任度高的节点。首先，为了解决问题我们需要衡量一个节点处理任务的能力。

不妨设评价函数

  （3.1）

其中， 为信任度函数（见公式3.3），表示系统对于节点的信任程度。代表节点执行任务需要的时间。对于，通过节点执行一小段测试任来得到。为比例系数。

对于价值越高的任务，我们优先选择评价好的节点（若该节点处于空闲状态），选择策略的映射关系数学描述为：

 （3.2）

式中，表价值函数，用来衡量任务的价值，函数值由任务发布者自己设定，对于简单任务而言往往各任务价值相等。

1. 信任度

我们无法假定所有参与节点均“诚实可靠”。有可能存在“不诚实”节点，为了投标得到更多的任务，有心伪造自己的算力情况，欺骗管理者。或者因为其他不可抗性原因导致算力提交出现了错误。自私节点以自己利益最大化为目的，如果没有考虑到自私节点的情况，系统会将大量任务分配给自私节点，而由于自私节点的算力不足无法迅速得到结果，导致系统进入漫长的等待，严重拖慢了整个系统的运行效率。考虑到可能存在恶意节点的情况，我们引入拜占庭将军问题的描述进行类比。

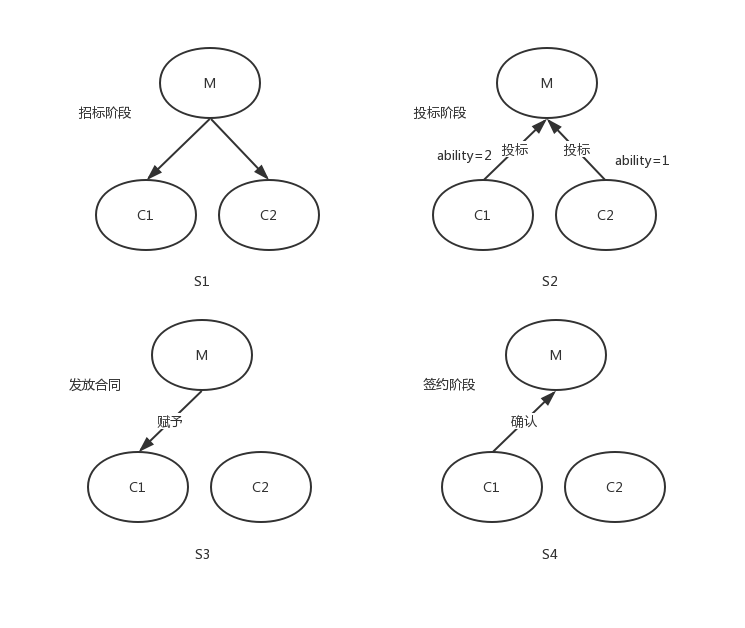


图3.3 欺骗节点行为示意图

图3.3分为4个阶段，其中M为管理者，C1为“欺骗节点”，C2为诚实节点。由于C1伪造了自己的能力，使得管理者相信C1比C2更具备解决任务的能力，于是将合同分配给了C1。为了解决这个情况，我们引入“信任度”的概念，用来衡量一个节点的可信程度。类比于“芝麻信用”的思想，为了描述可信程度，需要将该节点过去的行为进行一个评估，用来确定它的信用值。

设表示节点的信用值，有以下数学描述：

 （3.3）

从公式可以看出，若节点未有过“欺骗”情况，则信用度为1，同时“欺骗”的情况越多，节点信用度越低，将严重影响节点的能力评估。

## 3.2.2 数学建模

首先，对分配关系数学建模如下：

**Step1 参数描述**

定义二元组 ，其中集合 表示所有参与计算的节点Agent的集合，集合 表示任务集合。

不妨设有m个Agent，同时存在n个任务，则分配矩阵定义如下：

 （3.4）

定义代价矩阵 ，表示将任务 分配给 的代价。

**Step2模型建立**

**定义代价函数**

  （3.5）

表示节点执行任务所需要的代价。其中 代表节点执行任务需要的时间。 表示节点执行任务的当前负载。 为信任度函数（见公式3.3），表示系统对于节点的信任程度。 与 与 均为比例系数。

**定义支出函数**

 （3.6）

3.6式中， 表示执行任务需要的支出。 表示系统为任务的估价函数，通过虚拟货币的形式发放。 为比例系数，根据环境变化。

根据方程式（3.5）（3.6），易得

 （3.7）

对于整个模型来说，我们需要一个效能指标函数，来衡量分配方式的好坏。根据分配矩阵与代价矩阵，由（3.4）（3.5）（3.6）（3.7）式可以很容易得到：

 （3.8）

3.8式中，Sum是关于的函数，表示在整个分配过程中，分布式集群执行任务的好坏程度。更一般的，由于同一任务往往只分配一次，观察分配矩阵 的结构，由（3.8）易得

 （3.9）

我们可以得到一个最优解 ，表示全局分配最优的情况，此时

 （3.10）

**Step3约束条件**

约束条件需满足如下情况：

 （3.11）

（3.11）式表示将所有任务分配完毕。

 （3.12）

（3.12）式表示若任务与有顺序关系，则需要让他们顺序执行。

 （3.13）

（3.13）式表示对于任务的定价需要高于任务的花费，让Agent有充分意愿执行。

**Step4小结**

该模型是对合同网算法的本地化处理，为下一节的详细设计中算法改进过程提供了支撑。分配问题是一个NP难题，本文无法做到对所有的分配方式进行讨论，因此该模型有一定的局限性，只是基于前人研究的改进与创新。该模型说明了，只要通过测试任务测试得到每一个节点的算力大小，对于本文选择的简单分配方式就可以得到一个表现良好的分配矩阵。

由于每次的任务的发布可能是动态添加的，因此对于系统而言，数据传输过程便成为了制约整个算法的瓶颈。在实际处理中，我们在一开始对整个系统的节点进行算力测试，构造分配矩阵，每次执行任务完毕后评估Agent执行结果并部分更新分配矩阵，后续则按照合同网招投标流程处理。

## 3.2.3 详细设计

本小节对于算法流程进行详细的描述。

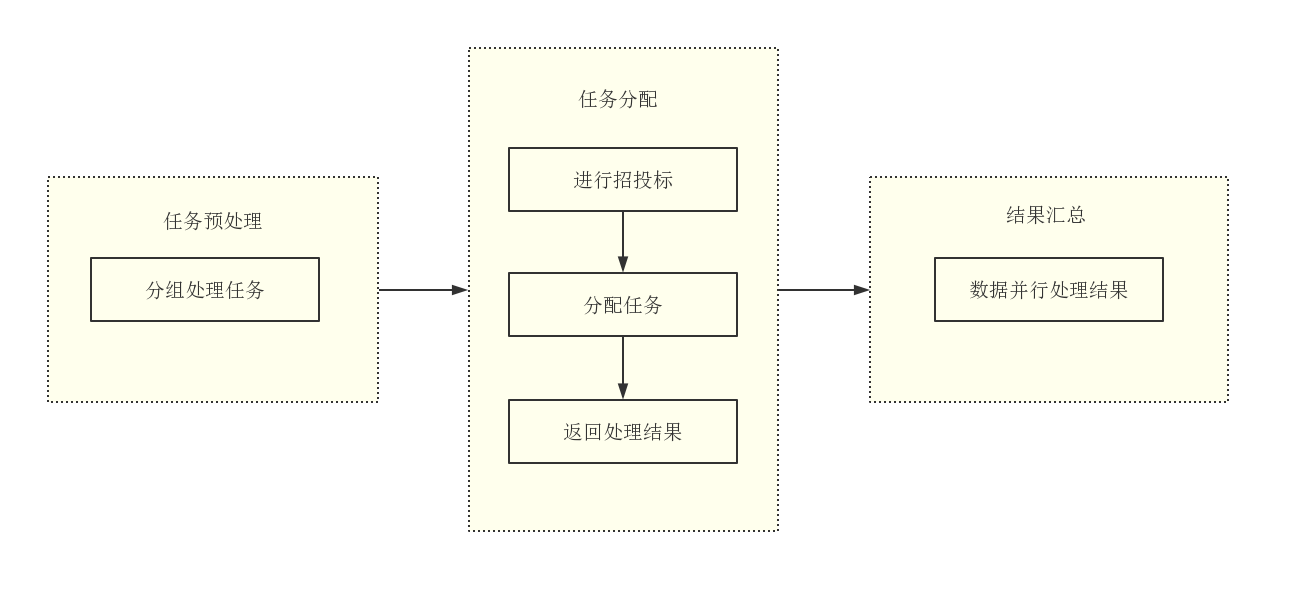
第三章对于整个分配流程有一个整体描述，第四章将对改进的算法进行实验。整个算法分为以下几个步骤

（1）任务预处理：将机器学习任务分批并编号；

（2）任务分配：把任务分配给相应的节点；

（3）结果汇总：将各个节点得到的结果汇总并分析。

其完整流程如图3.4所示

3.4 任务分配完整流程图

当管理者（Manager）接收到一个新任务时，需要对该任务进行定价，然后发起招标请求。管理者（Manager）分析接收到的招标情况，选择将任务分配给最合适的对象，并且设置期望时间。节点接受到标书后确认中标并执行任务，运行完毕后将结果返回给管理者（Manager）。管理者对执行时间与根据节点标书得到的期望时间作对比，若发现该节点存在“欺骗”现象，则降低该节点的可信度。若任务在deadline之前依旧没有执行完毕，则重新对其他节点进行招标。图3.5展示了对于单个任务处理的UML活动图。

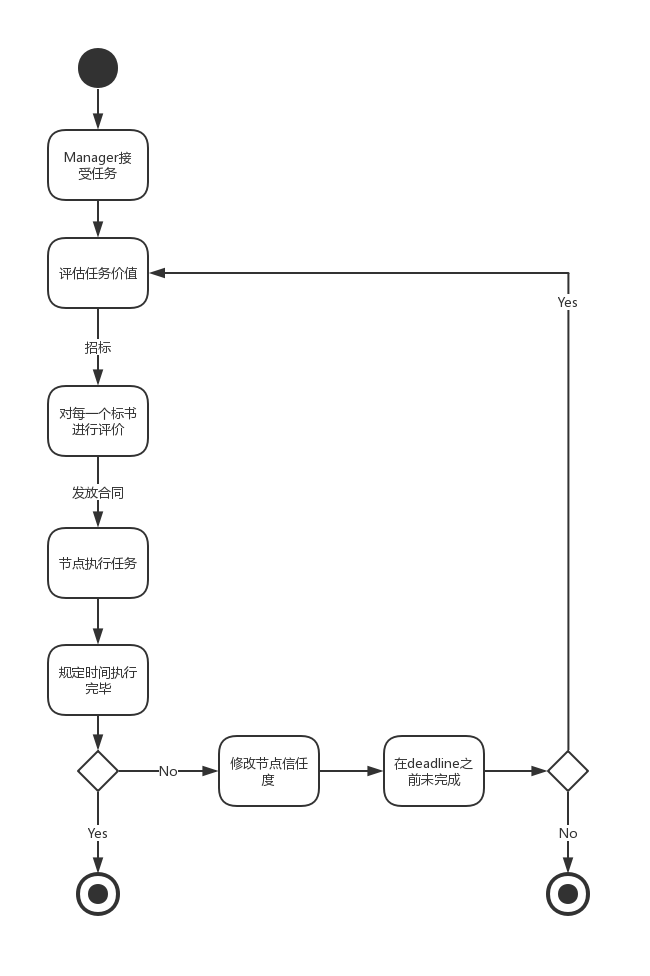


图3.5 单任务任务分配活动图

为了更清晰方便的描述分配过程，算法的伪代码流程如下所示。

|  |
| --- |
| **Algorithm 1** pseudocode for this algorithm |
| 1. **Input**: tasks; 2. **Output**: combined result; 3. **Initialization**: successTaskNum0, totalTaskNumtasks.length, nodeNumber1 4. check connected node, then **do** 5. nodeNumbernodes.length; 6. set tasks value, and evaluate for every nodes. 7. testTaskUint for every nodes, then divide tasks for lots of groups. 8. **while** successTaskNum < totalTaskNum **do** 9. **if** unSuccess.length > 0,  **then** 10. group = unSuccess.pop(); 11. **else** 12. group = groups.pop(); 13. **end if** 14. Select property nodes, and then give out invitation. 15. Listen node's response, then give out one group for selected node. 16. Check node's response. 17. **if** actualTime > estimatedTime, **then** 18. node.successNum --, unSuccess.push(group); 19. **else** 20. successTaskNum += groupCount; 21. **end if** 22. **end while** 23. Combine results for next step. |

## 本章小结

本章对目前被广泛使用的分布式算法进行了一个简要的介绍，对比评价了各个算法的优劣性，并选择合同网算法作为本次实验的基础。而对于合同网模型的缺陷，在3.2节，本章对该模型进行了本地化的改造。在3.2.2节建立了新的数学模型，在3.2.3节详细设计中给出了新模型的活动图以及伪代码，为下一章的实验提供了理论基础。

# 合同网改进模型实验验证

## 相关技术

## 4.1.1 区块链

区块链是一个分布式的账本，随着2008年中本聪（化名）发布比特币白皮书，区块链作为其底层技术受到了极度重视。人如其名，区块链在数据结构上是一个链表，连接着矿工挖出来的区块，区块链结构图见图4.1：

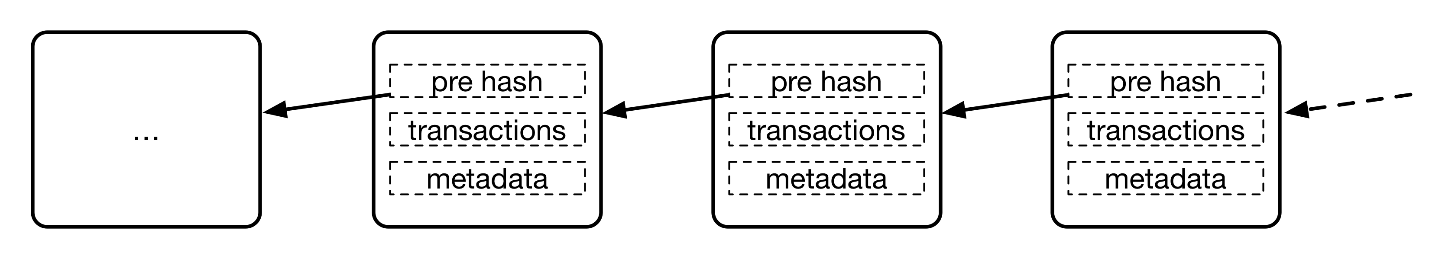


图4.1 区块链结构图

如果用单片机来比喻，区块链是一个单片机，而比特币只是使用这个单片机制造的计算器程序，区块链可以被抽象成更加有效的计算模式。区块链主要由3个关键技术组成：

·分布式账本。整个区块链可以视作一个p2p网络，每台机器上都存在着完整的账本，因此每台机器都可以参与监督作证，共同维护社区的安全。

·非对称加密。非对称加密保护了用户的隐私，虽然交易信息是公开的，但是用户身份是保密的。比起素数分解，目前整个区块链网络更倾向于椭圆曲线方案。

·共识机制。共识机制的数学抽象是拜占庭将军问题，而区块链引入了工作量证明的方法来保证共识，按照少数服从多数的原则，通过使节点需要支付承担不起的代价从而在经济学上避免攻击。

## 4.1.2 以太坊

以太坊（Ethereum）是新一代可编程区块链平台，从更加宏观的角度来说，以太坊是开源的全球化的去中心化架构，它使开发人员能够利用内置的经济学方法构建强大的去中心化程序。区块链融合了多个学科的内容，如：密码学、博弈论、经济学、分布式计算、信息安全、p2p网络等，以太坊集成了这些内容，使得我们不需要从头开始构建区块链，让原本极其陡峭的学习曲线变得平滑。以太坊中也有属于自己的货币，单位为以太（ether），我们需要自己的账户来接入以太坊网络。当然，对于我们搭建的私有网络而言则不需要考虑以太币的限制。

如果把以太坊看做一个状态机，那么交易便是让该状态机改变状态的东西，交易驱动着以太坊的运转。状态转换函数如图4.2，交易包含以下数据：

|  |  |
| --- | --- |
| 名称 | 描述 |
| nonce | 序列号，防止消息重播 |
| gas price | 发送者愿意支付的gas价格 |
| start gas | 发送者可以支付gas的最大值 |
| to | 目标地址 |
| data | 变长的二进制数据 |
| v,r,s | 外部所有账户（EOAs）的ECDSA签名的三个组成部分 |

表4.1 交易的部分数据结构

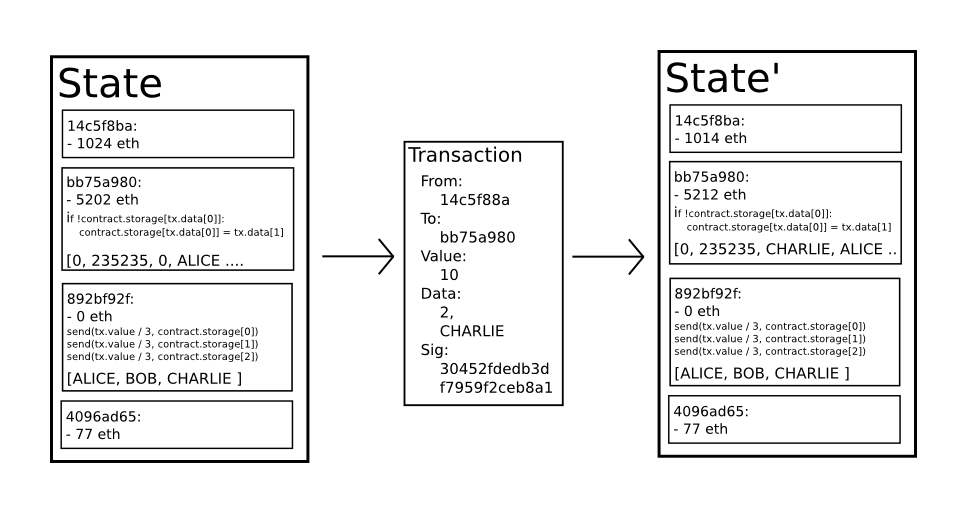


图4.2 以太坊状态转换函数

可以看到表4.1中存在gas字段，gas是以太坊的燃料，每次交易根据任务的难易程度会支付不同的gas，以避免拒绝服务攻击或者逻辑错误导致的无限循环。

同理，要进行交易我们至少需要存在两个账户，在以太坊中账户有两种类型：

|  |  |
| --- | --- |
| 账户类型 | 描述 |
| 外部所有账户（EOAs） | 由以太坊之外的程序控制，如钱包 |
| 合约账户 | 由以太坊内部控制，在虚拟机（EVM）上运行 |

表4.2 账户类型

交易时的对象可以指定为真实存在的外部所有账户（EOAs），也可以为合约账户，合约账户由开发者编写的智能合约控制。智能合约通常由solidity语言编写，编译为字节码后便可以部署在以太坊的链上作为合约账户存在。在交易时指定合约的地址与合约提供的应用程序二进制接口（ABI）即可完成交互。合约的代码不能更改，作为区块链的一部分除非被删除，否则将会一直存在。利用交易的特点，在交易时附加指令到data字段，我们便可以通过编写智能合约控制节点的行为，从而实现相应的算法。

## 4.1.3 Truffle

Truffle是针对基于以太坊的solidity语言的开发框架，solidity本身是一种图灵完备语言，该框架对客户端做了深度的集成。其主要有以下特点：

·开发、测试、部署操作都可以一键命令执行，不需要繁琐的环境配置与命令；

·提供了类似于gradle的项目架构机制，且打包流程可以自动执行，自动生成相关的目录；

·为合约提供了抽象的接口，内部基于web3封装了工具包，可以很方便的调用合约对象进行测试；

·含有控制台，可以在终端中直接使用控制台测试，为开发调试提供了方便；

·自动编译部署合约，不需要每次都重新部署区块链，极大的简化了开发流程。

由于Truffle如此方便，本文选择使用其作为以太坊开发框架实现分配算法，并且开发环境配置如表4.3所示：

|  |  |  |
| --- | --- | --- |
| 名称 | 值 | 描述 |
| host | localhost | 主机地址 |
| port | 7545 | 开放网络端口号 |
| network\_id | \* | 允许任何网络接入 |
| websockets | true | 允许使用websocket连接 |

表4.3 truffle环境配置

这样，私有链上的节点即可通过<http://localhost:7545/>接入区块链网络进行通信。

## 4.2 实验环境

本文基于以太坊平台搭建私有链进行实验，开发框架使用truffle平台，并通过VMWare创建虚拟机加入私有链来模拟各个节点，各个节点之间通过NAT模式与主机连接。我们通过搭建的系统实现区块链分布式系统的交互过程。系统的具体参数如下所示：

**硬件平台**

|  |  |  |
| --- | --- | --- |
|  | 主机 | 虚拟机 |
| CPU | Intel(R) Core(TM) i7-4720HQ CPU @ 2.60GHz | cpu cores：1 |
| GPU | NVIDIA GeForce GTX 950M |  |
| RAM | 8GB | 1GB |

**软件平台**

|  |  |  |
| --- | --- | --- |
|  | 版本 | 描述 |
| Truffle | v5.0.4 | 以太坊开发框架 |
| Ganache | v1.1.0 | 以太坊测试链客户端 |
| Solidity | v0.5.0 | 实现任务调度 |
| Node.js | v10.15.1 | 利用Mocha框架用于测试 |
| Python | v3.7 | 实现机器学习算法以及数据可视化 |
| JavaScript、HTML、CSS |  | 前端可视化平台搭建 |

## 实验设计

在我们实现的系统中，为了起到对照实验的效果，设置两类管理者（Manager）Agent与两类投标者（Contractor）Agent。两类管理者的区别在于对于投标的选择策略不同。其中一类管理者采用简单发放标书的策略，即对于投标者投递的标书按接收顺序给节点分配，先到先得，不考虑其他因素。第二类管理者则结合信任度的思想，通过评估函数（见公式3.1）计算该节点的价值，然后从标书中选择价值高的节点来给予任务。可以预见的是，这两类管理者适用于不同的环境，若所有节点都诚实守信，那么直观上两个算法效率相似；若系统处于非诚实环境，即存在不诚实节点，恶意谎报标书，那么第二类管理者在这种环境下会比第一类管理者工作效果好。管理者类型如表4.4所示。

|  |  |
| --- | --- |
| 管理者类型 | 分配策略 |
| 类型A | 按顺序分配 |
| 类型B | 按评估函数分配 |

表4.4 管理者类型与分配策略

按照同样的思想，接下来我们设计两类投标者（Contractor）。其中一类投标者诚实守信，对于整个系统来说它们是老实节点，从不造假，并且有任务执行时不会参与对其他任务的投标。第二类投标者是所谓的恶意节点，它们以自身利益为重，不考虑整个系统的性能。为了方便描述，对于第二类恶意节点，我们定义一个恶意随机变量。该变量的现实意义是对所有任务参与投标，且每次投标时，该节点有的几率通过设置算力的最大值从而百分百得到该任务。其中，即服从0—1分布。投标者类型如表4.5所示。

|  |  |
| --- | --- |
| 投标者类型 | 投标策略 |
| 类型A | 诚实守信 |
| 类型B | 行为受到恶意随机变量控制 |

表4.5 投标者类型与投标策略

为了方便起见，我们创建4个节点，同时设置单元任务为对10000个随机数的快速排序，通过设置不同的单元任务数量提供给不同类型的Agent执行，我们便可以得到实验数据。我们进行两组实验，分别测试两种类型的管理者在两种环境下的表现。实验一为类型A与类型B的管理者在完全诚实合作下进行协作任务，实验二则为两类管理者在非诚实环境下协作任务。每一组实验类型A与类型B管理者互为对照组。

## 实验结果与分析

## 4.4.1 实验过程数据

在实验过程中会产生很多中间数据，为了更好的观察实验结论，我们利用web前端框架搭建了一个简单的监控平台，用来展示运行算法过程中产生的数据。该平台会间隔一定的时间读取区块链上的数据，以达到监控的效果。取某一次实验中区块上记载的实验数据快照如图4.3所示：

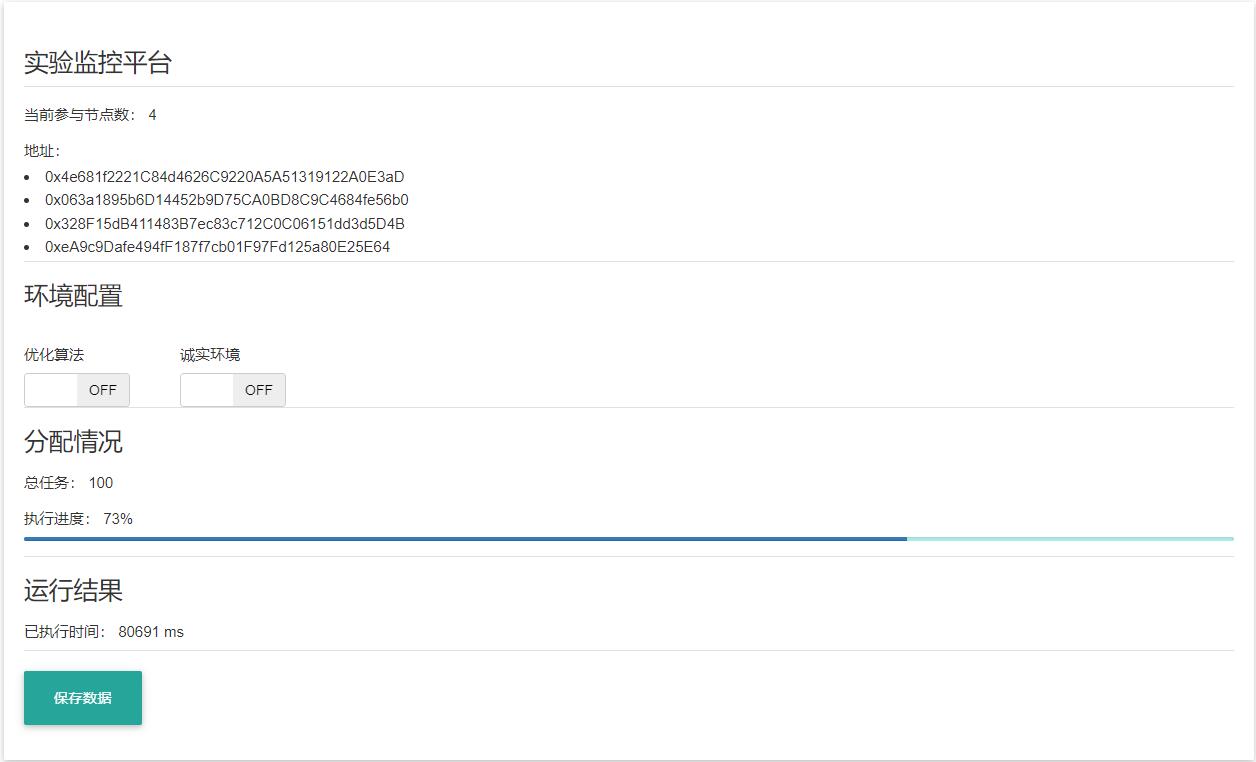


图4.3 实验快照

从图中可以看出，搭建平台的结构分为5个部分：

第一部分是参与节点信息，表示了参与节点的数量及地址；

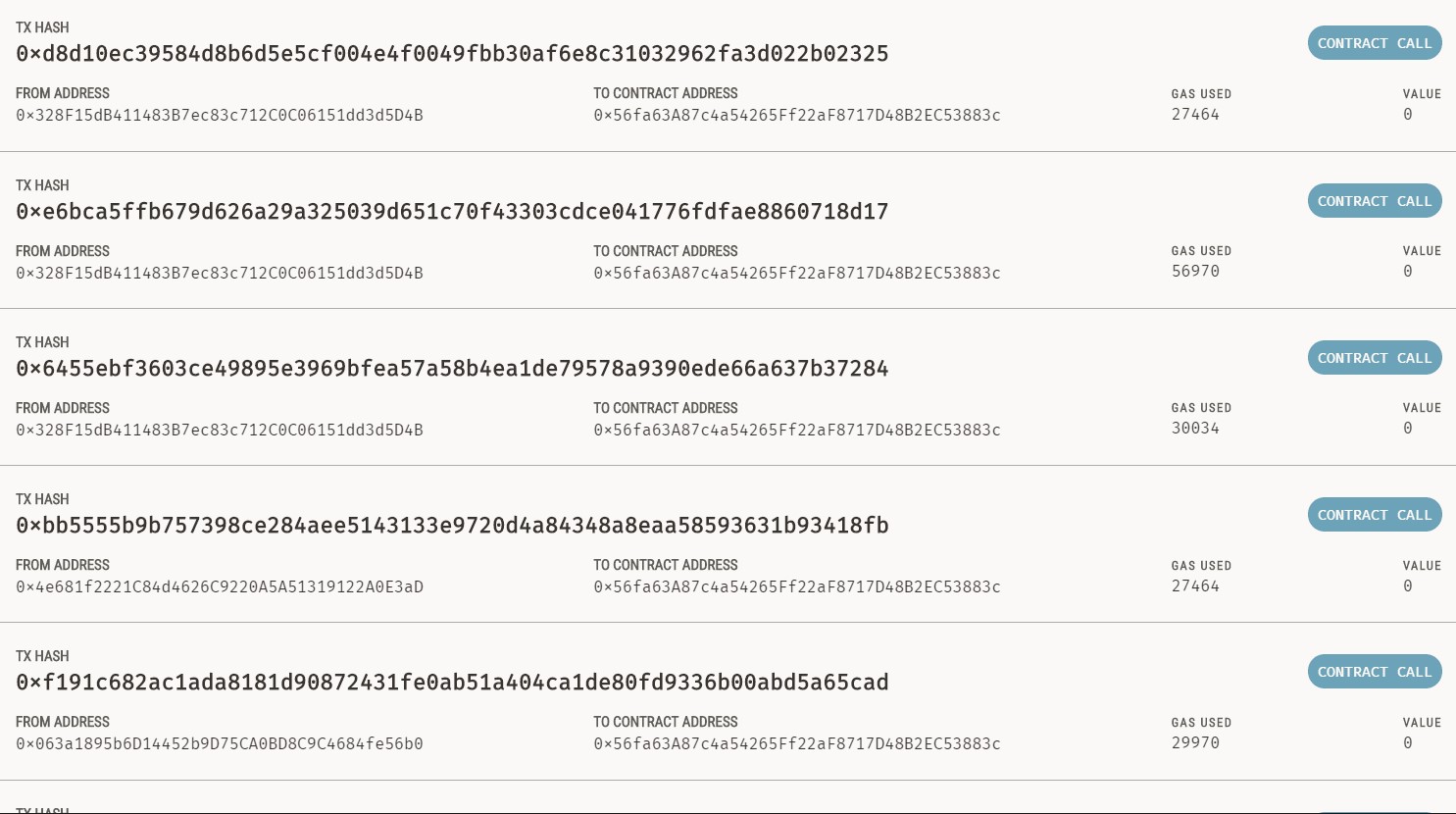
第二部分是关于实验环境配置信息，可以选择是否启用优化算法，以及当前是否为诚实环境；

第三部分展示了任务完成情况，进度条显示了当前执行进度；

第四部分为已执行时间；

第五部分可以保存当前执行状态数据。

同时，节点每次招投标都通过一次交易来完成，账户产生的每一笔交易都记录在本地客户端中，如图4.4所示：

图4.4 交易记录

这些交易记录保存了每次招投标的状态与结果。

在算法运行过程中私有链会生成新的区块，生成的区块记录如图4.5所示：



图4.5 区块记录

## 4.4.2 实验结果分析

**实验一（完全诚实环境）**

首先，设定环境为完全诚实合作的，即所有的投标者都是类型A，我们来比较不同类型的管理者对于结果的影响（图4.6）。该组实验共有4个诚实节点加入运算，实验数据为4次重复实验取的平均值（表4.6）。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 任务数 | 50 | 100 | 150 | 200 |
| 管理者类型A/ms | 55094 | 104914 | 158476 | 203867 |
| 管理者类型B/ms | 54932 | 99941 | 156932 | 201743 |

表4.6 实验一

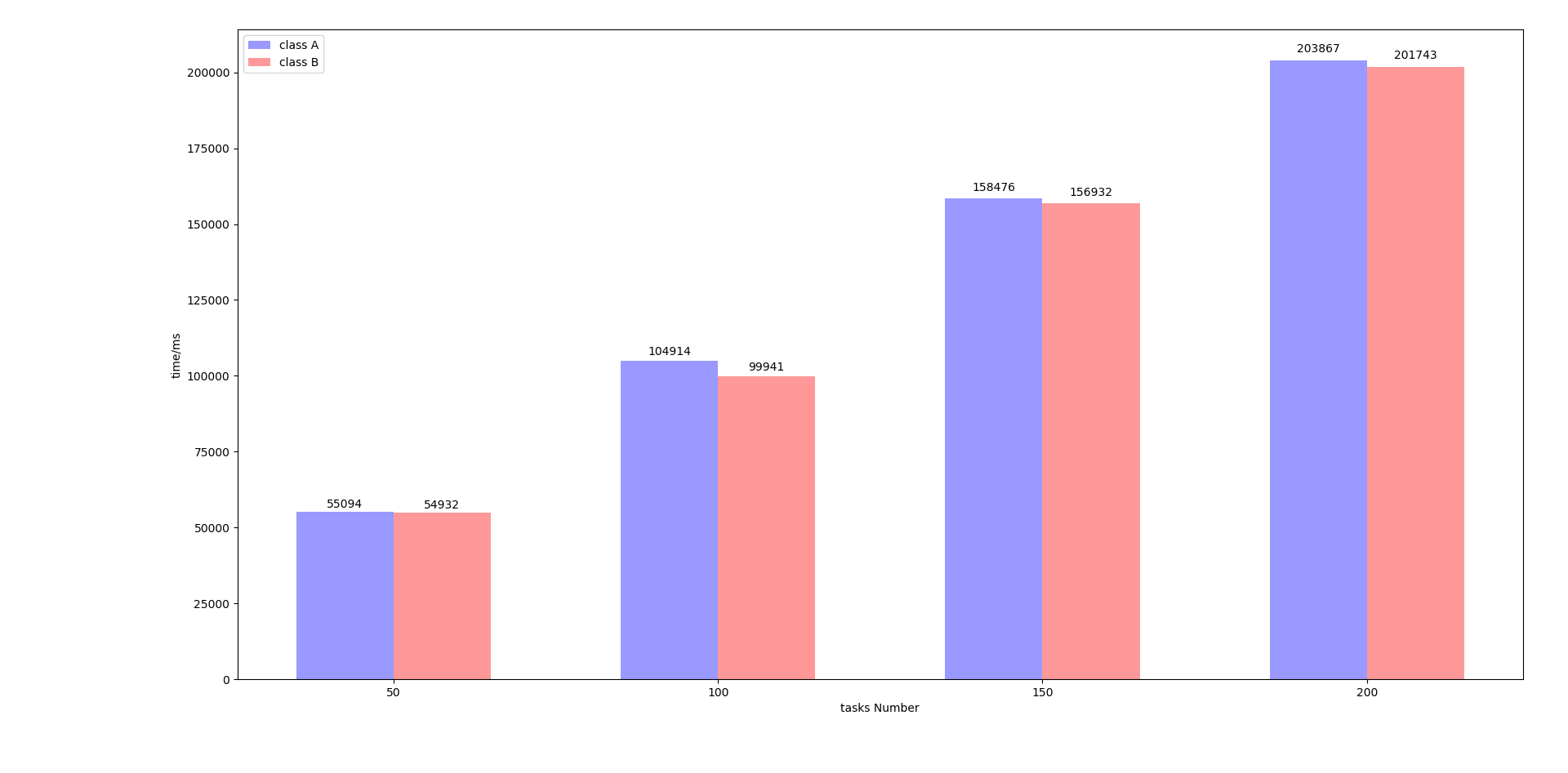


图4.6 实验一

在实验一中，我们分别取任务数量为50、100、150、200，类型A与类型B的数据如图4.4所示。从图表可以看出，在环境为完全诚实合作时，类型B管理者与类型A管理者表现相差无几，但类型B表现稍微好一些。直觉上分析，对于类型A来说，由于按顺序分配标书，会导致有充分算力的节点不能得到充分利用。对于类型B来说，由于利用了评估函数（式3.1），测试得到每个节点的能力，可以充分发挥节点的算力水平。

**实验二（非诚实环境）**

下面进行实验二，测试两种类型管理者在非诚实环境下的表现。我们来比较不同类型的管理者对于结果的影响（图4.7）。该实验共有4个节点参与计算，其中有两个诚实节点，两个恶意节点，实验数据为4次重复实验取的平均值（表4.7）。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 任务数 | 50 | 100 | 150 | 200 |
| 管理者类型A/ms | 56726 | 117421 | 173741 | 236907 |
| 管理者类型B/ms | 55412 | 114597 | 163948 | 220110 |

表4.7 实验二

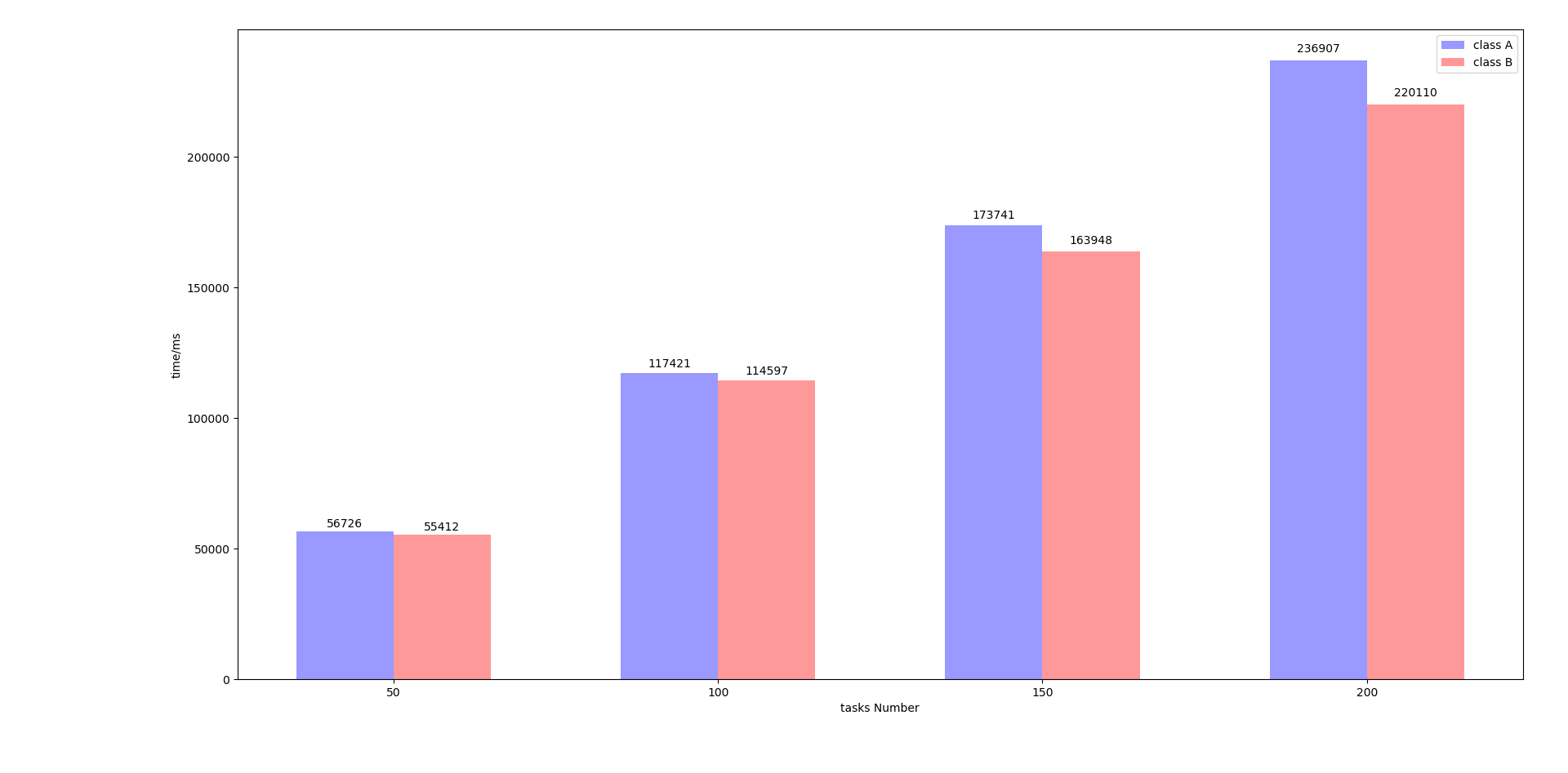


图4.7 实验二

从图4.7可以看出，在非诚实环境下，优化算法的表现明显比原始算法效果好，并且随着任务数的增多，效果会越来越好。对比实验一与实验二的图表，我们可以发现在不完全信任的环境中系统处理任务时间成本明显上升。我们对实验结果进行分析，当存在有欺骗节点时，由于欺骗节点伪造了自己的算力，使得管理者会将任务分配给欺骗者，其他正常节点得不到任务，导致浪费与管理者交流的机会，时间成本直线上升。在优化算法中，我们由评估函数（式3.1）对节点的算力进行了调整，并且根据大数定律，随着任务数的增加，评估结果会越来越准确，最终欺骗节点与正常节点之间的算力之比会调整到一个正常值。

实验一与实验二折线图如图4.8所示。

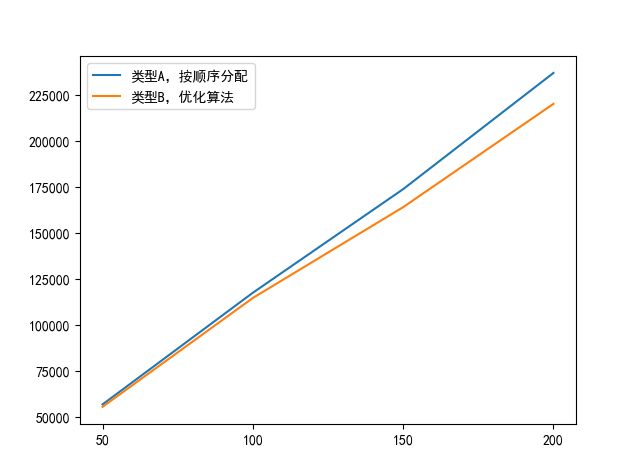


图4.8 实验二趋势图

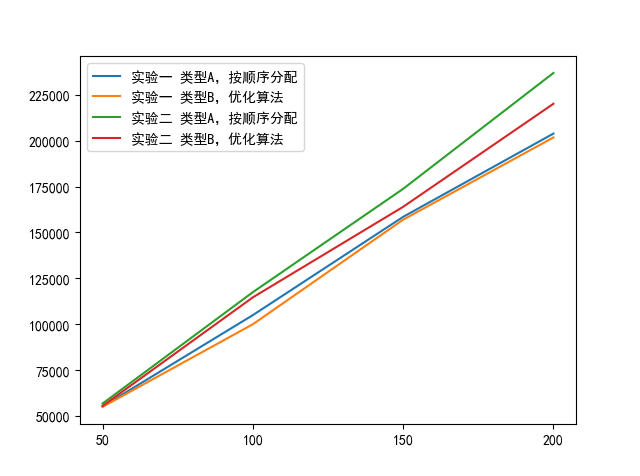


图4.9 实验一和实验二趋势图

## 本章小结

从第三章算法设计到第四章实验验证，我们得到了一个适用于非诚实环境的分配算法模型。该模型在完全信任的环境中表现一般，在非完全信任的环境下表现良好。在实验过程中，发现系统损耗的时间往往来自于管理者（Manager）与投标者（Contractor）进行通信时的损耗，这也是一个可以优化的点。

# 基于西雅图房价数据集实验

## 相关技术

## 5.1.1 梯度提升回归

梯度提升回归（Gradient boosting regression，GBR）是一种从错误中学习的技术。它本质上是集思广益，使用一堆较差的学习算法进行学习。每个学习算法精确率都不高，但是它们集成起来可以得到很好的准确率。每个学习算法都是依次使用的，换言之每个算法都在之前的算法的错误中进行学习。其基本公式如下：

 （5.1）

式5.1中，为基学习器的个数，为系数，是基学习器，为总的模型。我们的目的是得到一个总模型，使得损失函数尽可能的小，即：

 （5.2）

由于最终是多个基学习器加权组成，为了求得，我们使用每次求解一个基学习器以及系数，让损失函数在其梯度方向下降，从而一步步提升的性能。Python中的sklearn包提供了函数的GradientBoostingRegressor接口，使得我们可以很方便的调用函数就可以完成模型的训练。

## 数据可视化

数据科学的日益发展，带动了数据可视化领域的蓬勃生机，目前学术界有很多可视化工具，其中Python的第三方库Matplotlib是最为流行的图形库之一。它可以与NumPy一起使用，以开源的方式提供了一种有效的MatLab替代方案。在实验中我们用Python实现机器学习算法，同时将结果用Matplotlib呈现。

使用Matplotlib我们可以实现很多直方图、条形图、误差图、散点图等，该工具绘制的函数

图像与散点图分别如图5.1和图5.2所示：

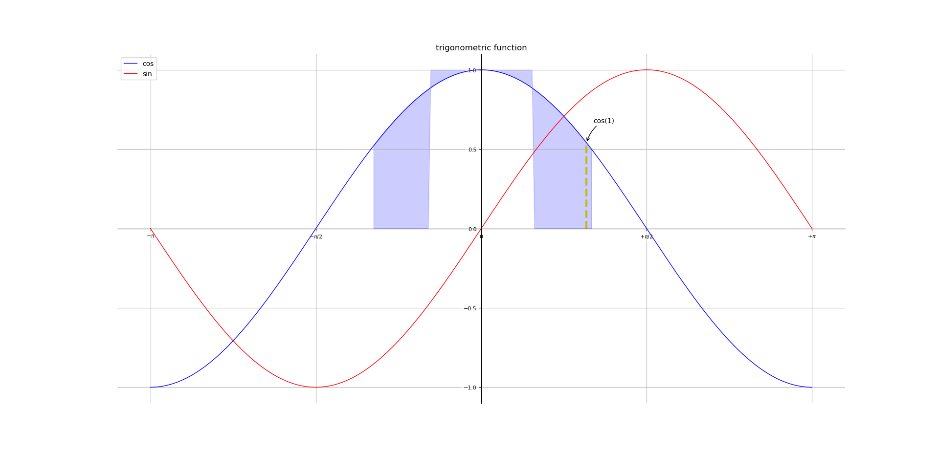


图5.1 三角函数图示例

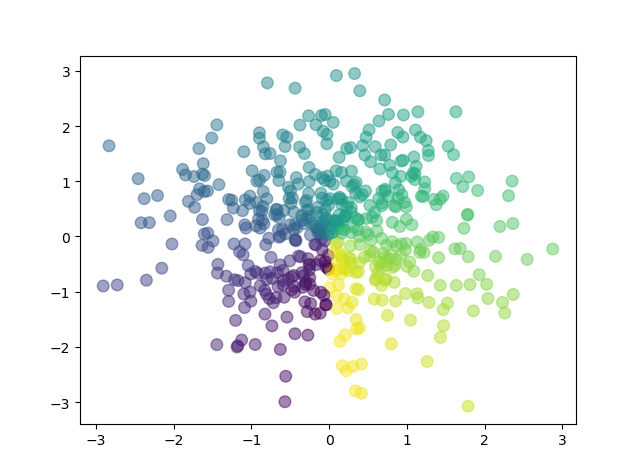


图5.2 散点图示例

## 5.2 数据预处理

本章使用前面章节搭建的分布式平台，对西雅图房价模型进行训练并预测。本实验数据来自华盛顿大学（University of Washington）提供的2014-2015年西雅图地区房价售卖情况（house prices sold in Seattle area in 2014-15），数据集地址链接参见附录1。实验数据共有21613条，每行数据有21列，内容如表5.1所示。

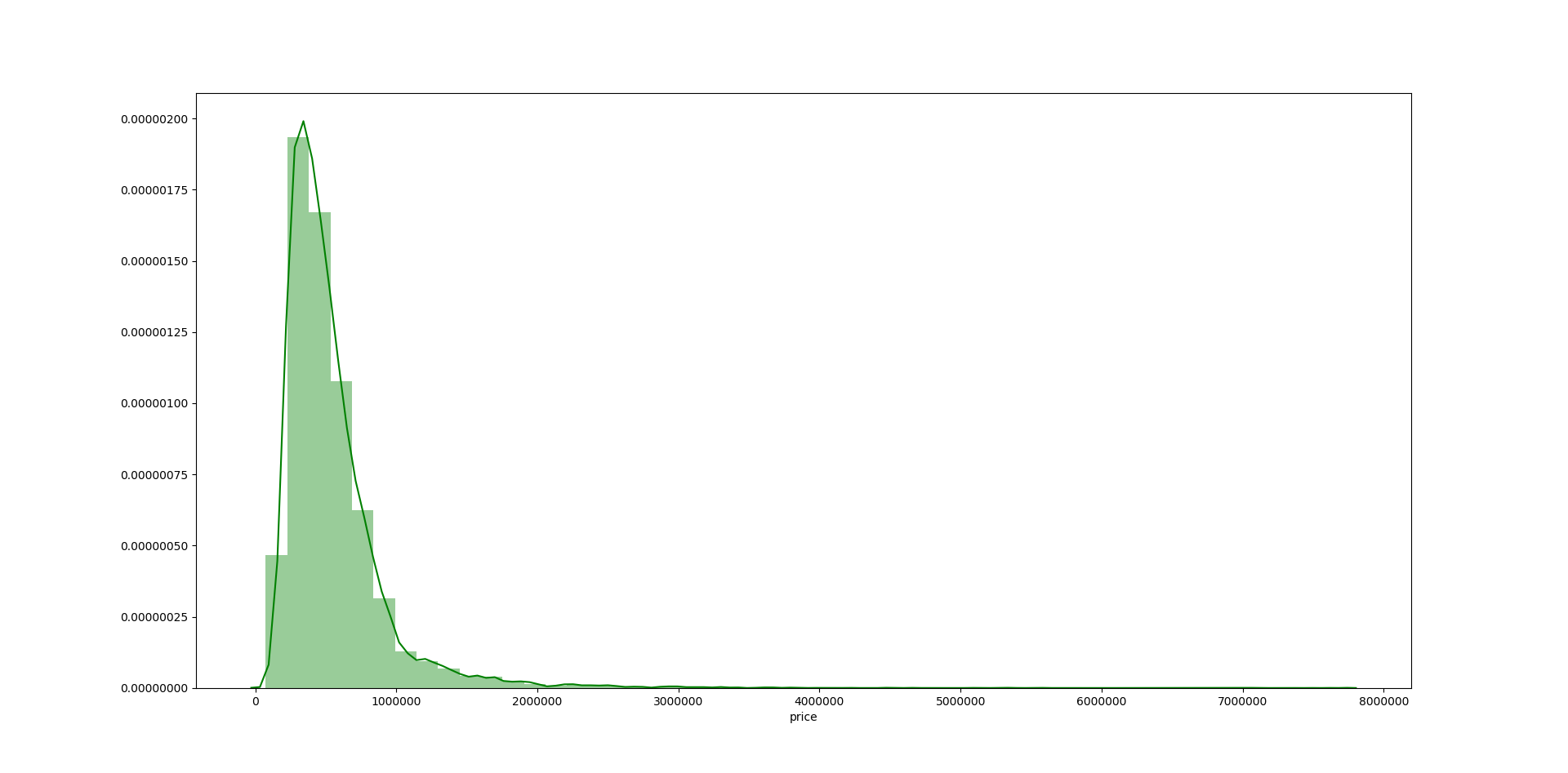
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| id | date | price | bedrooms | bathrooms | sqft\_living | sqft\_lot |
| 7129300520 | 20141013T000000 | 221900 | 3 | 1 | 1180 | 5650 |
| floors | waterfront | view | condition | grade | sqft\_above | sqft\_basement |
| 1 | 0 | 0 | 3 | 7 | 1180 | 0 |
| yr\_built | yr\_renovated | zipcode | lat | long | sqft\_living15 | sqft\_living15 |
| 1955 | 0 | 98178 | 47.5112 | -122.257 | 1340 | 5650 |

表5.1 单条房价数据示例

其中，id属于无关变量，price属于需要预测的变量，我们通过19个影响因子来判断房价的情况。

下面我们对数据特征进行研究。

首先观察该数据集所要预测的房价（price）数据分布，质量分布图见图5.3。

图5.3 房价数据分布

从图5.3可以看出，大部分房价集中在1,000,000以内，其中中位数的值为45,000。

由于在数据集中存在着大量需要二次处理的数据（如日期），我们需要对其进行数据清洗。对于日期在2014年的数据我们使用0替代日期，对于2015年的数据我们使用1来替代。同时，为了更加直观看出数据与数据之间的关系，我们需要了解各个数据之间的相关程度。计算相关性最常用的是皮尔森相关系数，用来反映两个变量的线性相关程度，公式如5.3所示：

 （5.3）

我们对于各个属性计算相关系数，得到的相关性热力图如图5.4所示。

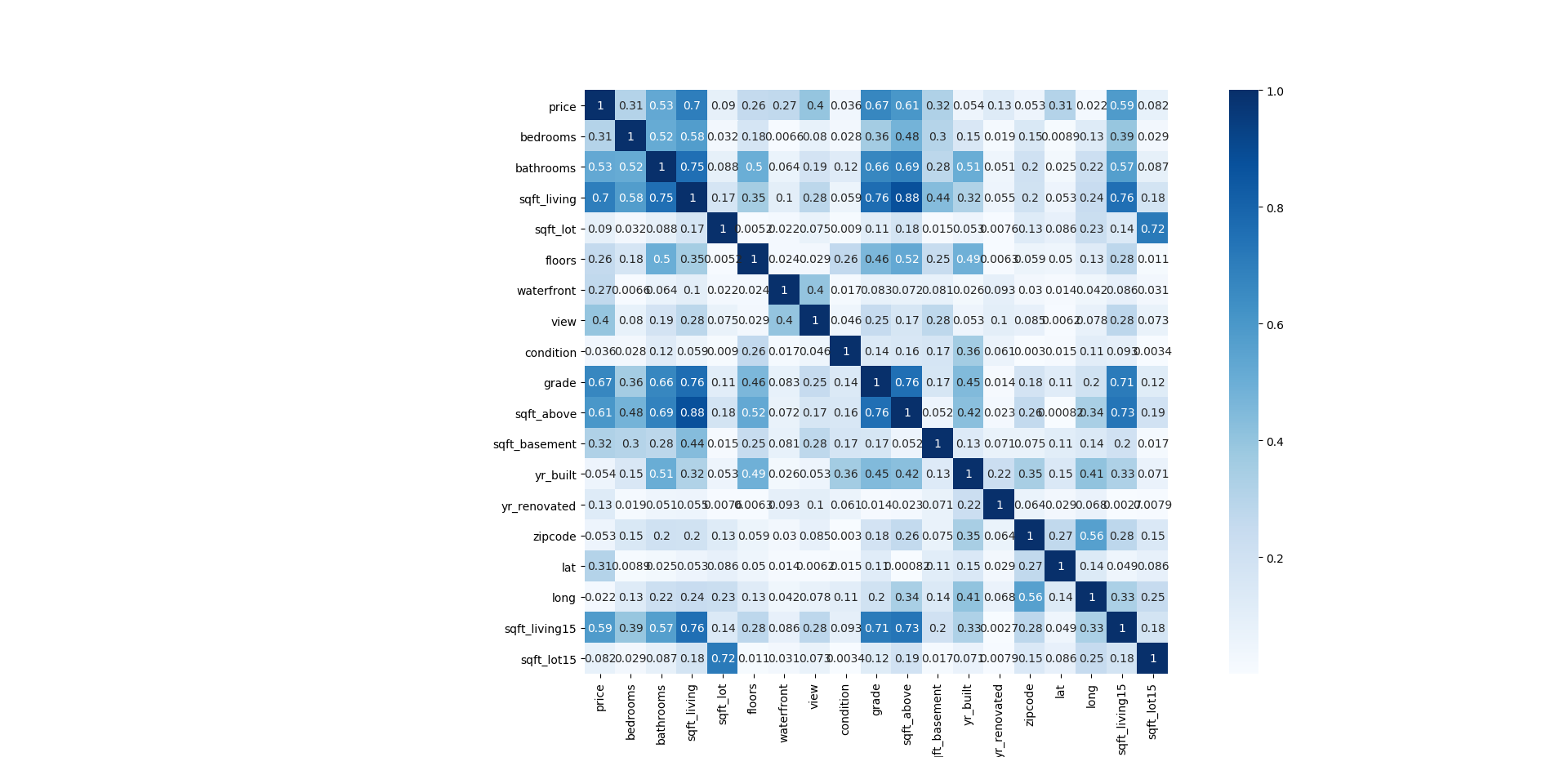


图5.4 相关性热力图

## 实验设计

我们选择使用交叉验证法来进行实验评估。

首先将数据分成两份，实验数据和测试数据共21600条，剩余13条数据用于交叉检查模型。对于21600条数据集，我们将其分为10个大小均为2160的互斥子集，即，每个子集尽可能保持数据一致性，即从数据集中分层采样而来。然后，我们每次使用相邻9个子集的并集作为训练集，剩下的那个子集作为测试集，这样我们就得到了10组训练/测试集，从而可以进行10次训练和测试。最后，我们将10次的训练结果取平均值就得到了最终的训练模型数据。

在实验中，我们将这10次训练/测试任务作为10个任务单元，分配给节点去执行，这样我们就能得到训练结果以及执行效率情况。

对于训练模型，考虑到该问题为预测房价，我们使用回归分析对数据进行拟合。在具体的方法上，下表显示了在一个任务中多元线性回归与梯度提升回归（GBR）的准确率对比：

|  |  |
| --- | --- |
| 回归方法 | 准确率 |
| 多元线性回归（MLR） | 73.2 % |
| 梯度提升回归（GBR） | 92.0 % |

表5.2 回归方法比较

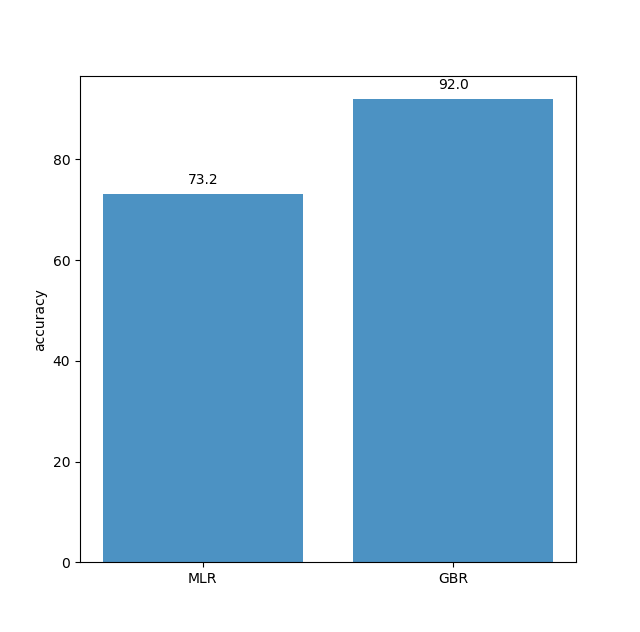


图5.5 回归方法比较

在GBR参数选择上，基学习器的个数（n\_estimators）设置为400，决策树最大深度（max\_depth）为5，随机梯度提升步长（learning\_rate）为0.1，损失函数（loss）为最小二乘法。从图5.5可以看出梯度提升回归算法明显效果较好，准确率达到0.92，因此对于任务单元我们都采用梯度提升回归来处理之。

## 实验结果与分析

在实验中，我们对这10个任务单元分配到4个节点上去执行，管理者（Manager）类型为类型B（按评估函数分配），投标者（Contractor）类型为A（诚实守信）。运行我们前面搭建的分布式平台，得到了计算时间如图5.6所示：

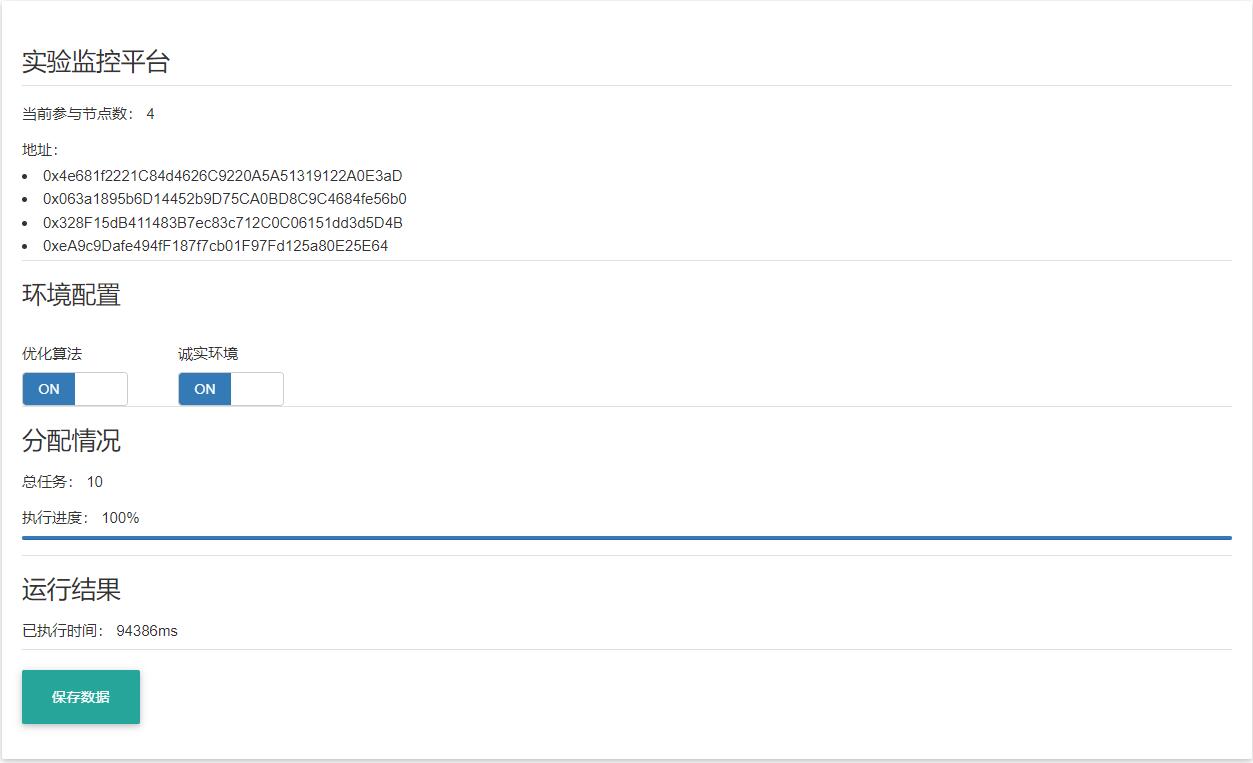


图5.6 任务结果快照

我们对该实验设置对照组为在单一节点运行10个任务单元，两种环境见表5.3：

|  |  |  |
| --- | --- | --- |
| 运行环境 | 描述 | 组别 |
| 单一节点 | 单一机器 | 对照组 |
| 分布式平台 | 4节点，合同网优化算法，诚实环境 | 实验组 |

表5.3 运行环境对比

实验组和对照组执行效率如表5.4所示：

|  |  |  |
| --- | --- | --- |
| 运行环境 | 执行时间/ms | 组别 |
| 单一节点 | 185693 | 对照组 |
| 分布式平台 | 94386 | 实验组 |

表5.4 执行时间对比

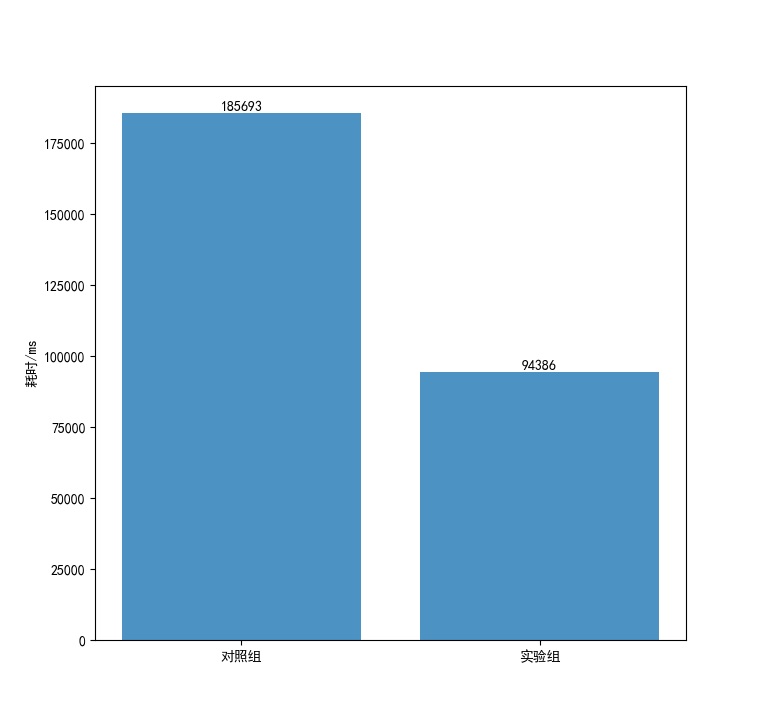


图5.7 执行时间对比

可以看出，相比对照组，本项目搭建的区块链分布式平台当前实验环境下，大约节省了一半的执行时间。

## 本章小结

本章用一个具体的实例，对本项目搭建的区块链分布式算力汇聚系统进行了验证。总的来说在节点数越高的情况下系统执行效果越好，在4个节点的情况下对本章训练的算法的效率提高了约49%，考虑到数据传输的损耗与单元任务的耗时，该结果基本合理。

# 总结和展望

## 工作总结

互联网的高速发展使得全世界的数据量也以几何倍数增长。为了满足机器学习对计算能力日益增长的需求，伴随着区块链技术的火热，利用区块链技术的数据真实透明与共享、多方共识机制、多方安全验证等分布式处理模式用来解决机器学习的问题已经成为了一个新的研究领域。本文对基于区块链平台的算力汇聚方法进行了研究，主要进行了以下工作：

1. 对现有的智能分配算法进行了调研，主要关注了当前流行的合同网算法、paxos算法、raft算法，比较各个算法的优缺点。Paxos算法目前应用广泛，但该算法实现复杂，而且为了保证强一致性牺牲了可用性与性能。最终由于合同网简单可用易于实现而选择其作为本文的基础算法。
2. 在设计本实验的算法时，考虑到合同网算法本身存在诸多可待优化的地方，于是对合同网进行了本地化的改进。通过引入信任度的思想构造了评估函数，对整个分配模型进行了数学建模，并且在详细设计模块设计了算法的细节并提供伪代码。
3. 接下来对设计的算法进行实验，本文选择使用以太坊作为实验平台，为此搭建了私有链并使用合约控制各个节点的执行。为了建立对比实验，需要设计两种管理者，分别为按顺序分配（A）和根据评估函数分配（B）。同时设计两种投标者模拟信任与非信任环境，分别为完全诚实节点（A）和受恶意随机变量控制的节点（B），其中恶意随机变量符合0-1分布，现实意义为该节点有的概率作弊获得标书。通过进行实验一与实验二，得到了基本满意的数据。
4. 最后本文使用前面搭建的平台对西雅图房价预测模型进行训练，在训练方法选择上，用多元线性回归与梯度提升回归进行了对比，得到了梯度提升回归较为适合的结论。在评估方法上选择交叉验证法进行评估，将训练/测试集分层采样分为10份，然后，每次使用相邻9个子集的并集作为训练集，剩下的那个子集作为测试集，这样就得到了10组训练/测试集，从而可以进行10次训练和测试。将10次单元任务分配给节点执行，得到数据后，本文比较了使用单个机器与使用该分布式平台训练的效果，结果基本符合预期。

## 心得体会

这次是我在大学生活中第一次完成实验室的任务，心中有激动也有忐忑。在当初接受题目的时候，面对完全陌生的领域，也曾怀疑自己是否能够实现。在学习的过程中，我翻阅了很多资料与论文，给了我很大的启发。当我研究工作陷入瓶颈的时候，经常晚上睡不着觉，感觉思路停滞不前，心中充满焦虑。后来通过看别人的论文和书籍，站在巨人的肩膀上才对自己遭遇的问题有了一个清晰的认识。这种感觉从开始到现在一直伴随着我，往往是发现问题然后解决问题，一直在发现同时也一直在解决。在解决问题的过程中，我感觉自己的能力与眼界都得到了显著的提高，也增强了自己的自信心，我想这便是毕业设计给我带来的成长吧。

我仍感觉到，对于知识的冰山我仅仅只是窥视到它的一角，还存在着很多内容等待着我去发掘。正所谓“看山是山，看山不是山，看山还是山”，学无止境，对于知识的敬畏将伴随着我的终身。

## 进一步的工作

任务的调度与分配问题一直是学术界的一个难点，而任务与任务之间也具有顺序、循环、条件选择等关系。下一步可以研究在多种复杂任务的环境下，本算法是否还能保持比较好的效果以及该如何优化。对于机器学习而言，训练往往有一个迭代的过程，而不同的学习算法如何拆分任务实现并行化也是需要深入研究的领域。

千里之行始于足下，区块链技术本身具有很大的价值，本文的目的在于以区块链为媒介对智能分配算法做了测试与验证，并且验证可行性。下一步可以通读以太坊源码，通过修改底层共识机制来实现算力的汇聚。

# 参考文献

[1] 董祥千，郭兵，沈艳，段旭良，申云成，张洪. 一种高效安全的去中心化数据共享模型. 计算机学报，2018，41（5）：1021~1035.

[2] 何蒲，于戈，张岩峰，鲍玉斌. 区块链技术与应用前瞻综述[J]. 计算机科学，2017，04，1.

[3] Tencent FiT, Tencent Research Institute. White paper for tencenttrustSQL. White Paper, 2017(in Chinese)(腾讯FiT, 腾讯研究院. 腾讯区块链方案白皮书. White Paper, 2017).

[4] 郭兵，李强，段旭良，申云成，董祥千，张洪，沈艳，张泽良，罗键. 个人数据银行—一种基于银行架构的个人大数据资产管理与增值服务的新模式.计算机学报，2017，40（1）：126~143.

[5] Nakamoto S. Bitcoin: A peer-to-peer electronic cash system. White Paper, 2008.

[6] Beijing PeerSafe Technology Co., Ltd. White paper for blockchain database application platform. White Paper, 2017(in Chinese)(北京众享比特科技有限公司. 基于区块链的数据库应用平台技术白皮书. White Paper, 2017).

[7] 郭兵，沈艳，董祥千.基于区块链的数据共享方法.国家发明专利（申请号：201810101734.7）.

[8]王云松,孙佳林,龚跃.基于蚁群算法的云任务调度研究[J].长春理工大学学报(自然科学版),2017,40(01):133-136.

[9]施龙. 基于蚁群算法的分布式多智能体任务分配研究[D].浙江工商大学,2011.

[10]郑绪釉.负载平衡-提高分布式计算机系统效率的一种方法[J].计算机工程及应用,1984,(2):54-61.

[11]刘小梅，田彦涛，杨茂.基于博弈论的多机器人任务分配算法[J].2010,28(3).

[12]张海俊, 史忠植.动态合同网协议.计算机工程,2004,30(21)：44~46.

[13]Smith R G. The Contract Net Protocol: High Level Communication and Control in a Distributed Problem Solver. IEEE Transactions on Computers, 1980, 29: 1104~1113.

[14]李明,刘玮,张彦铎.基于改进合同网协议的多Agent动态任务分配[J].山东大学学报(工学版),2016,46(02):51-56+63.

[15]基于改进合同网的多Agent系统协作机理研究[J]. 秦久峰,曾凡明,陈于涛,李杰.武汉理工大学学报(交通科学与工程版). 2014(05).

[16]多Agent系统任务分配方法综述[J]. 唐苏妍,朱一凡,李群,雷永林.系统工程与电子技术. 2010(10).

[17]多Agent系统中信任和信誉模型的研究[D]. 贺利坚.北京交通大学 2011.

[18]Research on cooperation of multiple-agent based on contract-net protocol.Zhang Jin,Cao Yaoqin. International Conference on Industrial Control and Electronics Engineering. 2012.

[19]童向荣. 多Agent合作求解中的信任与协商研究[D].北京交通大学,2010.

[20]张广胜，蒋昌俊，沙静，等. 基于时间Petri网的合同网模型研究[j]．系统仿真学报， 2008， 20( 20) : 5438 － 5441.

[21]刘海龙，吴铁军．基于合同网的多Agent 任务分配分布式优化算法［J］．浙江大学学报，2001， 35( 2) : 550 － 554.

[22]李纪超. 大规模分布式机器学习平台[D].北京邮电大学,2016.

[23]亢良伊,王建飞,刘杰,叶丹.可扩展机器学习的并行与分布式优化算法综述[J].软件学报,2018,29(01):109-130.

[24]万小平. 分布式任务自主分配和协同控制[D].陕西师范大学,2014.

[25]孙隅隅，黄光球． 基于黑板模型的多Agent智能决策支持系统的研究［J］．嵌入式与单片机，2007， 20( 25):85 －88．

[26]许子灿,吴荣泉.基于消息传递的Paxos算法研究[J].计算机工程,2011,37(21):287-290.

[27]杨革,徐虹.Paxos算法的研究与改进[J].科技创新与应用,2017(07):25-26.

[28]朱汉成.基于改进Raft算法的私有链模型[J].现代计算机(专业版),2019(01):40-42.

[29]鲁子元.浅析RAFT分布式算法[J].信息技术,2017(09):168-170.

[30]周侨,方明.基于多Agent的众包任务分配算法的研究[J].智能计算机与应用,2019,9(01):104-107.

# 致 谢

子在川上曰：逝者如斯夫！不舍昼夜。

一晃四年时间过去，时光荏苒，而刚刚入校时的情景我仍记忆犹新。犹记得“海纳百川，有容乃大”的校训，在这四年里，我奔波在图书馆与自习室的次数数不胜数，站在毕业前夕回头看，我无时无刻不在感激那时候奋斗的自己。当初的汗水终会浇灌出美丽的花朵，所有的辛苦终将得到回报。

我尤其感谢我的导师郭兵老师。在我确定了自己保研的消息后，我第一时间便联系了郭老师并表达了想要加入实验室的愿望。郭老师为人师表，在了解我的兴趣点后亲切的解答了我对未来的困惑，并且为我的毕业设计选定了方向。未来的三年我将成为郭老师众多研究生中的一份子，学生不才，请多多指教。

感谢实验室的张珍博士学姐。张学姐为我敲定了选题，并且在我遇到难题时耐心的为我解答困惑，为我提供了数不尽的帮助，让我守得云开见月明。张学姐为人亲和，即便面对天资愚钝的我也充满了耐心，师兄师姐们也给了我很多建设性的意见，我的感激之情无以言表。

我还要感谢我的家人，一直在背后默默的支持着我，我知道他们一直竭尽全力为我提供最好的学习环境，他们就是我永远的后盾。我要感谢此刻身在千里之外的某个人，遇见过你是我一生的幸运。美丽有两种，一是深刻又动人的方程，一是你泛着倦意淡淡的笑容。我时常在想若是我们能够守得一辈子陪伴，那又将是多么令人惊喜的事情。希望你身体健康，每天开心。

四年很短，人生很长，在川大学习生活的记忆将是我一辈子的宝藏。我奔跑过炎炎夏日下的长桥，也感受过秋日明远湖上的微风；我怀念总是手抖的食堂阿姨，也怀念六毛三两的白米饭。长桥上的情书是专属于江安的浪漫，黑板上的涂鸦是专属于少男少女的恶作剧。感谢我的朋友，特别是我的室友们，正是你们的陪伴让我知晓了大学的意义，这段美好的时光将使我终生受益。在这里生活的点点滴滴，我必将永远铭记！

# 附录1 实验数据集地址

2014-2015年西雅图地区房价售卖情况数据集地址：

https://github.com/Jonasyao/Machine-Learning-Specialization-University-of-Washington-/blob/master/Regression/Assignment\_four/kc\_house\_data.csv

# 附录2 文献翻译（译文）

来源：2012 International Conference on Industrial Control and Electronics Engineering

基于合同网协议的多Agent协作研究

**摘要**：经典的合同网协议可以实现开放异构环境下的协作。但多Agent系统是一个典型的复杂系统。MAS的开放性、分散性和动态性使得MAS环境中充满了许多不确定因素，导致网络流量激增，谈判陷入僵局。为了解决这些问题，本文引入了信任层次域和声誉机制，建立了合同网模型的信任和声誉系统。基于多Agent协作经验的模型可以有效地减少网络流量，提高MAS的效率和协作实时性。

**关键词：**多Agent系统，信任域，声誉，合作

1. 引言

合同网协议(Contract-Net Protocol，简称CNP) [1]是Randall Devis和Reid G Smith在分布式计算中针对任务和资源分配问题提出的一种经典的协作策略。它广泛应用于多智能体系统(multi-Agent System，简称MAS)的协作策略设计中。基本的合同网协议模拟了人的商业活动，在多代理竞标过程中进行拉合作，解决任务分配中的资源冲突。

经典的合同网协议在合作过程中存在许多缺陷，具体表现为:

(1)在招标过程中，为了向所有可能的代理商发送任务合同信息，系统在广播中通知所有代理商，并等待代理商的所有反馈或投标截止日期的到来。这将导致网络流量激增、协商时间开销或死锁，并浪费系统大量的计算资源。

(2)由于MAS环境的开放性、分布式和动态性，传统的合同网协议不能适应多agent能力和工作量的动态变化。这将导致合作任务过于集中，使得任务等待时间增加，任务代理的工作量严重失衡。

(3)任务代理被动接受任务，不根据其能力和工作量情况。这将降低合作的效率。

目前，在开放的网络环境下，分布式计算问题越来越受到人们的关注。在许多工程应用中，传统的契约网协议已经失效。大量研究者致力于多智能体动态协作策略的研究，并提出了若干扩展和改进。文献[7-8]介绍了信托代理的概念。文献[8,10]将Agent的心理状态和社会属性加入到合作策略中。文献[7-10]研究了延伸性投标状态、评标、拒绝投标规则、策略投标价格等标准。在提高多智能体协作成功率、降低网络流量、平衡工作负载、缩短协作时间等方面进行了大量的研究工作，取得了良好的效果。本文在经典合同网协议[1,2]的基础上，引入了信任域和声誉的概念，建立了信任域和声誉的合同网模型。

1. 信任域
2. 函数域的定义和分类

1)定义 1

功能域: 根据多智能体的容量，将其划分为多个子域。系统域可以划分为不同的功能域。根据多智能体的功能域划分，给出了多智能体的作用。同一区域内的多智能体具有很强的相关性。因此，多智能体之间的合作可能性相对提高。提高了任务代理的响应速度和协作效率。函数域模型如图1所示。

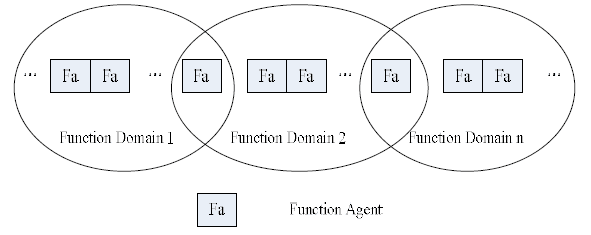


图1 函数域模型

多智能体属于系统域。根据多智能体提供的能力，将多智能体系统域划分为不同的子功能域，如域1、域2和域3等。每个功能域包含许多提供相同功能服务的代理。不同的功能域可能重叠，一些代理可能属于多个不同的功能域。

1. 信任域的定义

多Agent系统信任机制的目的是实现多Agent之间的合作与交互。由于不同的应用研究背景，对MAS信任的定义不能统一[3-5]。为了便于描述，本文将trust机制定义如下:

2)定义2

信任:在多Agent系统中，Agent i委托Agent j完成任务t等一种类型的任务，Agent j有能力执行任务t的信任程度记为Trust (i, j, t)，其范围为[0,1]。

3)定义3

信任阈值:系统将任务委托给任何代理，不需要最小信任阈值。信任阈值表示为Vtr。信任(i, j, t)≥Vtr是系统委托Agent j完成任务的必要条件。

4)定义4

识别阈值:。最小信任值称为系统识别多agent具有与任务t相关能力的识别阈值，表示为Vre。信任(i, j, t)≥Vre是系统数据库保留多agent容量信息的必要条件。

显然，0 < Vre < Vtr <1。

5)定义5

信任域:根据多agent的能力和信任水平，将多agent在一定的功能域中分为高级、中级和低级信任域，如图2所示。容量和tust值大于或等于信任阈值Vtr的代理程序集是高级信任域。容量和tust值大于识别阈值Vre但小于信任阈值Vtr的agent assembly是中间信任域。没有容量的代理程序集是低级信任域。

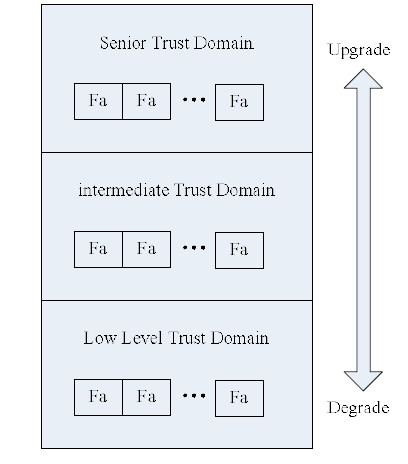


图2 信任域模型

6)定义6

优先级和关键Agent:在多Agent系统的功能域中，只有一个Agent具有与子任务k相关的特征能力项，任务k是多Agent系统的优先任务。能够完成之前任务的Agent k是关键Agent。显然，先验任务越少，系统的稳定性越好。

C.信任域的转移

多Agent的提升和下降与两个因素有关。首先，他们的能力变化剧烈。其次，在完成多个多agent的合作后，多agent的信任水平将会随着过渡规则进行调整。

多Agent的能力变化受自身软硬件升级、协作成功率的变化、并行任务间的干扰等因素的影响。

1)规则1

学习:如果与任务t相关的Agent i的信任值小于信任阈值Vtr，则Agent i可以通过了解任务的知识来提高其信任水平:



ε∈(0,1)是学习因素,信任增加了学习的程度。Learning(i, t)作为一个学习函数。T(i, t)表示函数Trust(i, t)系统委托Agent i执行任务T的信任程度。箭头表示Agent i执行学习动作的结果。

2)规则2

提升:在功能域中，如果所有功能Agent i相关任务t的信任值小于信任阈值Vtr，则系统将提升信任值大于或等于识别阈值Vre的信任值:



δ∈(0,1)是促进因子。

3)规则3

前序任务:当系统实施之前的任务k时，如果Agent i的信任值小于阈值的信任Vtr(假设关键能力的信任Agent i总是大于或等于阈值Vre)，那么Agent i的信任度将调整到Vtr的值:



4)规则4

过渡:在多Agent系统中，每个Agent的信任值在[0,1]中动态变化。当Agent i成功完成合作任务时，系统增加了信任值，反之，则降低了信任值:





τ∈(0,l)属于转变的因子。Act (i, t)是任务执行函数。系统使用该规则在信任域中提升或降低Agent i的信任值。

1. 声誉

在多agent系统中，以往的历史经验是agent之间合作任务的反映。一旦某个代理完成了委托的任务，系统就更有可能将任务委托给具有成功合作经验的代理。代理商也对有合作经验的代理商更感兴趣。本文将以往多agent之间的合作经验作为评价声誉的基础。根据相互作用产生的评价记录的存储和聚合方式，将声誉评级结构分为集中式、分布式和混合式[6]。在本文中，集中式声誉评级更适合。

集中式结构具有声誉评价中心，收集和汇总所有合作评价信息[4]。每次合作结束后，合作双方将各自的评估报告发送至信誉评估中心。信誉评价中心负责评价的存储和汇总。在下次合作之前，系统可以从信誉评估中心查询一些代理商的情况，作为投标决策的依据。

为便于对声誉进行定量分析，现达成如下约定:

变量m表示多代理系统的管理代理，变量i表示任务代理。变量t是一种特定类型的任务。变量tlate表示最近的时间段。变量tpast表示从开始到现在的总时间。下面针对同一任务对多agent进行定量分析。

(1)合作总数:Agent m与Agent i目前合作总数，记为sum\_tpast (m, i, t)。

(2)合作频率:Agent m与Agent i在单位时间内合作的次数，记为freq (m, i, t) = sum\_tpast (m, i, t) / tpast。

(3)近期合作频率:Agent m与Agent i在最近某段时间内的合作频率，记为freq\_tlate (m, i, t) = sum\_tlate (m, i, t) / tlate。

(4)单个合作的质量评价:Agent i完成任务时，Agent m对i进行评价，记为eval (m, i, t)。函数qual (m, i, t)表示协作解决方案的质量。函数time\_restricted (m, i, t)是合作时间约束。参数λqλt,分别解决方案的权值和时间约束的权值。



(5)近期合作总体评价:近期某段时间内合作质量总体评价，记为eval\_tlate (m, i, t):



参数集合{λ1,λ2,…λn}是一个随时间递减级数。参数λx∈{λ1,λ2,…λn}是每个合作的权值。参数n的值为sum\_tpast (m, i, t)。

(6)声誉:多agent系统将质量评价与合作频率相结合，综合反映了以往合作的情况。当前时间的信誉值tnow在下一次评估前不会发生变化。管理代理向任务代理发送投标是任务代理的重要依据。它表示为repute\_tnow (m, i, t)。



5)规则5

信誉更新:每次合作由Agent i完成后，系统将对Agent i进行信誉评估。变量tnow为当前时间。变量tnow-1是之前的时间。参数δ是声誉的损耗系数与时间。



对代理商i的信誉评估也会影响到其他代理商:



因此，基于信任和声誉的契约网络模型如图3所示。

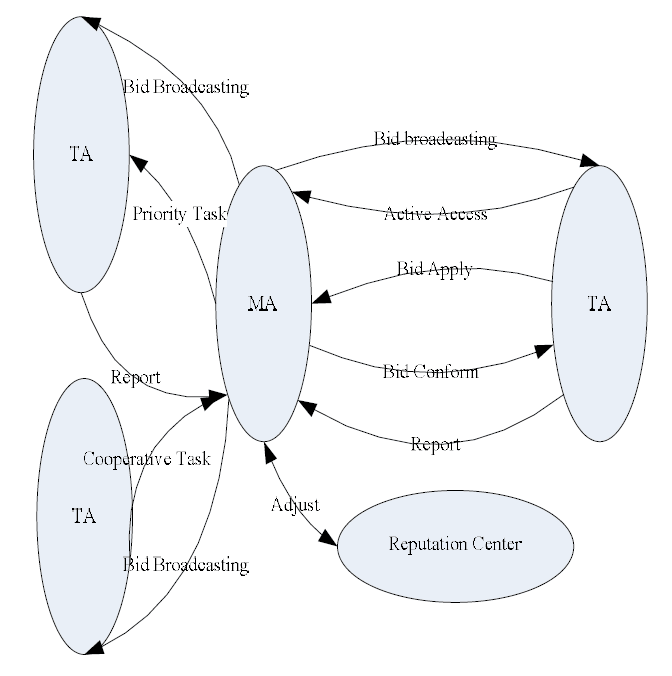


图3 基于信任和信誉系统的合同网模型

4. 基于信任域和信誉契约网的合作策略

合作战略的整个过程如下:

任务代理执行任务时，可以向代理m报告有合作请求。这一合作请求包括两种情况。一是任务代理不能独立完成任务，需要与他人合作。另一个是任务代理根本无法承担这个任务。

(一)符合招标范围

代理m确保任务相关的功能域，然后，代理m符合该功能域中来自高级信任域的代理程序集。如果高级信任域为空，则代理m符合来自此函数域中的中级信任域的代理程序集。通过调用规则2，系统将增加这些代理的信任值。如果中间信任域为空，则投标的范围为此函数域。

(二)播放招标信息

Agent m在黑板上按照步骤(1)所规定的范围内广播投标信息。报价信息

定义格式如下:

Task\_Announcement= {TaskID, Initiator, address, TaskDescription, TaskConstraint, Deadline}

(3)接收投标信息

为了提高合作效率，节省谈判时间，本文将该机制引入到黑板访问机制中。将黑板访问阈值设置为Rep，如果任务代理i的信誉值大于或等于访问阈值Rep，任务代理i可以访问黑板查询投标信息。



无论任务代理人主动还是被动地接收合同投标信息，都应检查其投标编号。为了提高协作效率，每个可能的任务代理都设置了一个契约参数Tsend和一个上限契约参数Tmax。参数Tsend根据投标条件动态增加，直到任务代理拒绝接受任何新的合同投标时，Tsend等于Tmax为止。一旦任务代理i与代理m建立正式的契约关系，参数Tmax的值变为0。Task Agent i将投标申请发送给Agent m。

(4)符合合作条件的投标人

如果Agent m收到声誉值大于或等于access threshold Rep的task Agent i的投标申请，Agent m将合作构词直接发送给task Agent i。否则，Agent m选择最合适的投标人作为合作任务Agent i，确定最合适投标人的方法如下:



然后代理m检查投标人的信任值是否大于或等于信任阈值Vtr。如果投标人的信任值达到信任阈值Vtr，则代理m发送合作确认书;如果投标人的信任值不能达到信任阈值Vtr, Agent m将通过调用规则1使投标人的信任值增加到等于信任阈值Vtr。然后代理m发送合作确认书。参数α,β分别的重量信任和声誉,而α+β= 1。

(5)报告

执行合作任务的任务代理将任务执行结果发送给代理m。根据任务执行结果，系统通过调用规则4调整任务代理i的信任值。同时，系统通过调用规则5更新任务代理i的信誉值。

(6)之前的任务

无论多Agent系统在协商或合作的过程中，如果之前的任务到达，系统会忽略key Agent i的状态，不管它是否繁忙，都会将之前的任务直接交付给key Agent。如果密钥代理是空闲的，它将立即执行优先任务。如果密钥代理正忙，它将中断正在进行的实现，然后将合作请求报告给代理m。中断的任务将被交付给其他代理执行。对于前一项任务，如果密钥代理能够独立执行前一项任务，密钥代理将向代理m报告执行情况。如果密钥代理不能独立执行前一项任务，密钥代理将根据合作请求向代理m报告。与其他任务代理一样，无论任务是否优先，任务是否中断，关键代理的信任值和声誉都将根据任务实现结果进行调整。

5. 总结

在开放多用户系统中，传统的合同网在工程应用中存在不足。针对这些弊端，本文提出了一种基于信任和声誉的契约网络模型。该模型的优点如下:

(1)由于引入了函数域、信任域等概念，该模型可以大大减小任务代理的搜索范围和网络流量，避免网络拥塞。

(2)由于引入了信任和声誉机制，该模型可以对合作历史进行评价和记录，从而提高系统的性能，避免盲目谈判，有效提高合作质量和实时合作。

(3)在谈判过程中，由于引入了主动感知机制、拒标标准和优先任务分配策略，该模型能够平衡多agent系统的工作量，减少谈判时间。此外，这种合作模式摒弃了代理人的自私、竞争等不利因素。它适用于目标相同的多智能体系统，尤其适用于基于多智能体的频繁协作的仿真系统。

# 附录3 文献翻译（原文）

2012 International Conference on Industrial Control and Electronics Engineering

**Research On Cooperation Of Multiple Agent Based On Contract-Net Protocol**

Zhang Jin High-Tech Institute Xi’an [jimyeah4016@yahoo.com.cn](mailto:jimyeah4016@yahoo.com.cn)

Cao Yao-Qin High-Tech Institute Xi’an [jimyeah4016@yahoo.com.cn](mailto:jimyeah4016@yahoo.com.cn)

***Abstract*—Classical Contract-Net protocol among the multi-Agent bidding can realize the cooperation of open heterogeneous environment. But MAS (Multiple Agent System) is a typical complex system. MAS’s open, distributed and dynamic characteristics make the MAS environment filled with a lot of indeterminate factors, lead to surge in network traffic and negotiation deadlock. To solve these problems, this paper introduces the trust level domain and reputation mechanism to establish trust and reputation of the Contract-Net model. The model based on the history of cooperation experience in multi-Agent can effectively reduce network traffic, improve the efficiency and cooperation real-time of MAS.**

***Keywords*—Multiple Agent System, Trust domain, Reputation, Cooperation**

**1. INTRODUCTION**

Contract-Net Protocol (Contract-Net Protocol, referred to as CNP) [1] is a classic cooperation strategy, put forward by Randall Devis and Reid G Smith in the distributed computing for the task and resource allocation issues. It is widely used in multi-Agent System (Multi-Agent System, referred to as MAS) cooperation strategy design. The basic Contract-Net Protocol simulates human commercial activities, pull cooperation in the multi-Agent bidding process to resolve resource conflict in the task allocation.

There are many defects in cooperation process through classical Contract-Net Protocol as follows:

(1) During the bidding process, in order to send task contract information to all of the possible Agents, the system informs all of the Agents in broadcasting, and waits for all the feedback of Agents or the bidding deadline to come. This will lead to network traffic surge, negotiation time overhead or deadlock, and waste the system a lot of computing resources.

(2) Since the open, distributed and dynamic characteristics of MAS environment, the classical Contract-Net Protocol can not adapt to dynamic change of ability and workload of multi-Agent.This will result in cooperation tasks too concentrated, make the task waiting time increasing and the task Agent workload a serious imbalance.

(3) The task Agent accepts mission passively, not according to its capacity and workload conditions. This will make cooperation less efficient.

At present, in the open network environment, more and more studies focus on the distributed computing issue. In many engineering applications, the classic Contract-Net Protocol have been powerless. A large number of researchers dedicated to multi-Agent Dynamic Cooperation Strategy, and proposed several extensions and improvements. Literature [7-8] introduced the concept of trust Agent. In Literature [8,10],the Agent's mental state and social attributes was added to the cooperation strategy. In literature [7-10], such standard as extended bidding state, evaluation, rejecting bidding rule and bidding price for strategies were studied. More research work in multi-Agent cooperation success rate, reducing network traffic, balancing workload and shortening the time for cooperative problem, have been effective. This paper, based on the classic Contract-Net Protocol [1,2], introduced the concept of trust domains and reputation, established the trust domain and reputation Contract-Net model.

**2. TRUST DOMAIN**

*A. Definition and classification of function domain*

*1) Definition 1*

Function domain: Multi-Agent will be divided into various sub-domains according to their capacity. The system domain can be divided into different function domains. The role of multi-Agent is provided in accordance with its division of function domain. The multi-Agent among the same region have a strong correlation. So the possibility of cooperation among multi-Agent is relatively enhanced. And then, the response speed and cooperation efficiency of the task Agents is improved. The function domain model is shown as in Fig.1.

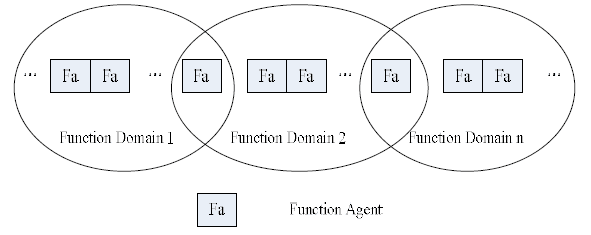


Fig.1 Function domain model

Multi-Agent belong to system domain. According to the capacity provided by multi-Agent, Multi-Agent System domain is divided into different sub-function domains, such as domain 1, domain 2 and domain 3 and so on. Each function domain includes a number of Agents offering the same functional service. Different function domains Possibly overlap and some Agents can belong to several different function domains.

*B. Definition of trust domain*

The purpose of trust mechanism in Multi-Agent System is to realize cooperation and interaction among multi-Agent. Because of different application research backgrounds, the definition of trust for the MAS can not be unified [3-5]. To facilitate the description, the tust mechanism in this paper is defined as follows:

*2） Definition 2*

Trust: In the Multi-Agent System, Agent i entrust Agent j to accomplish one type of task like task t. The level of trust that Agent j has the ability to carry out the task t is denoted as Trust (i, j, t ), with its range [0,1].

*3） Definition 3*

Trust Threshold: The system entrusting any Agent the task t need the minimum trust threshold. The trust threshold value is denoted as Vtr. Trust (i, j, t) ≥ Vtr is the necessary condition that system entrust Agent j the task.

*4） Definition 4*

Recognition Threshold:. The minimum trust value is called the recognition threshold that the system recognize multi-Agent has a capacity related to task t, denoted as Vre. Trust (i, j, t) ≥ Vre is the necessary condition that the system database retain the capacity information of multi-Agent.

Obviously, 0 < Vre < Vtr <1.

*5） Definition 5*

Trust Domain: According to the capacity and trust level of multi-Agent, multi-Agent is divided into the senior, intermediate and low-level trust domains in the certain function domain, shown as in Fig.2. Agents assembly with a capacity and tust value greater than or equal to the trust threshold Vtr is the senior trust domain. Agents assembly With a capacity and tust value greater than the recognition threshold Vre but less than the trust threshold Vtr is the intermediate trust domain. Agents assembly that do not have a capacity is the low-level trust domain.

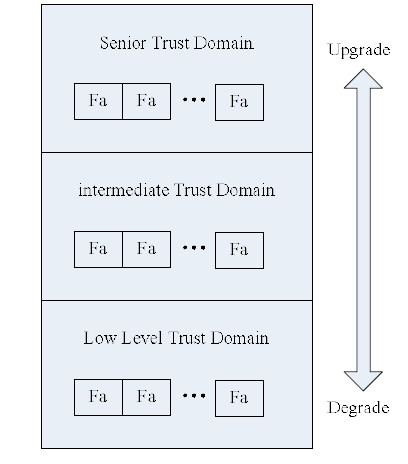


Fig.2 Trust domain model

*6） Definition 6*

Priority and Key Agent: In the function domains of Multi-Agent System, there is only one Agent with feature capability item related to sub-task k. Task k is a prior mission of Multi-Agent System. Agent that can accomplish the prior task k is the key Agent. Obviously, the less prior tasks, the better stability of the system.

*C. Transition of trust domain*

The promotion and demotion of multi-Agent relates to two factors. First, their ability changes dynamicly. Second, after accomplishment of several multi-Agent cooperation, trust level of multi-Agent will be adjusted with the transition rule.

Ability change of multi-Agent as a result of many factors, such as their own software and hardware upgrade, the change of cooperation success rate, interference between parallel tasks and so on.

*1） Rule 1*

Learning: If the trust value of Agent i related to task t is less than the turst threshold Vtr, the Agent i can learn about the knowledge of the task to increase its trust level:



ε ∈ (0,1) is the learning factor, the degree of trust increased by learning . Learning (i, t) as a learning function. T(i, t) means the function Trust (i, t), which is the degree of trust that system entrust Agent i to perform task t. Arrows indicate the result that Agent i execute learning action.

*2） Rule 2*

Promotion: In the function domain, if the trust value of the all function Agent i related task t is less than the trust threshold Vtr, the system will promote the one whose trust value is greater than or equal to recognition threshold Vre:



δ ∈ (0,1) is the the promoting factor.

*3）Rule 3*

Prior task: When the system has to implement prior task k, if the trust value of the key capability Agent i is less than the trust threshold Vtr (assuming the trust of key capability Agent i is always greater than or equal recognition threshold Vre), the trust value of Agent i will be adjusted to Vtr:



*4） Rule 4*

Transition: In Multi-Agent System, the trust value of each Agent changes dynamicly in [0,1]. When Agent i accomplish a cooperation task successfully, the system increase its trust value, otherwise, the system reduces its trust value:





τ ∈ (0, l) is called the transition factor. Act (i, t) is the task executing function. The system uses the rule to promote or demote the trust value of Agent i in the trust domain.

**3.REPUTATION**

In Multi-Agent System, previous historical experience is the reflection of cooperation task among Agents. Once some Agent have carried out the task entrusted, the system are more likely to entrsut the task to the Agent with successful cooperation experience. Agents also are more interested in the one with cooperation experience. This paper regards previous cooperation experience among multi-Agent as the basis of reputation evaluation. According to the store and aggregate way of assessment record resulting from interactions, reputation rating structure is divided into centralized, distributed and mixed-type[6]. In this paper, the centralized reputation rating is more suitable.

Centralized structure has a reputation evaluation center to collect and summarize all of the cooperation assessment information [4]. After each cooperation, both cooperation sides send their own assessment report on each other to the reputation evaluation center. Reputation evaluation center is responsible for assessment storage and aggregate. Prior to the next cooperation, the system can query from the reputation evaluation center about some Agents as a bid basis for decision making.

To facilitate the quantitative analysis on reputation, there is an agreement as follows:

Variable m represents management Agent of the Multi-Agent System and Variable i is on behalf of the task Agent. Variable t is a certain type of tasks. Variable tlate means a recent time period. Variable tpast represents the total time from the beginning to the present. The following quantitative analysis among multi-Agent is in terms of the same task.

(1) The total number of cooperation: the total number of cooperation between Agent m and Agent i so far, denoted as sum\_tpast (m, i, t).

(2) Cooperation frequency: the number of cooperation in the unit time between Agent m and Agent i, denoted as freq (m, i, t) = sum\_tpast (m, i, t) / tpast.

(3) The frequency of the recent cooperation: the cooperation frequency between Agent m and Agent i in the recent certain period of time, denoted as freq\_tlate (m, i, t) = sum\_ tlate (m, i, t) / tlate.

(4) Quality evaluation of a single cooperation: When Agent i accomplish a task, Agent m will evaluate i, denoted as eval (m, i, t) . Function qual (m, i, t) is cooperation solution quality. Function time\_restrict (m, i, t) is the cooperation time constraint. Parameter λq and λt, is respectively the solution quality weight and time constraint weight.



(5) Overall evaluation of the recent cooperation: overall evaluation of the cooperation quality in recent certain period of time, denoted as eval\_tlate (m, i, t):



Parameter assembly {λ1, λ2, ... λn} is a decreasing series with time. Parameter λx ∈{λ1, λ2, ... λn} is the weight of each cooperation quality. Parameter n is sum\_tpast (m, i, t).

(6)Reputation: Multi-Agent System combines the quality evaluation with frequency of cooperation, which reflects synthetically the situation in previous cooperation. Reputation value of the current time tnow dose not change before the next assessment. It is an important basis that the management Agent sends bid to task Agent. And it is denoted as repute\_ tnow (m, i, t).



*5） Rule 5*

Reputation update: When each cooperation is done by Agent i , the system will make the reputation assessment of Agent i. Variable tnow is the current time. Variable tnow-1 is the previous time. Parameter δ is the loss factor of reputation with the time.



This reputation assessment on Agent i will also affect on other Agents:



So, the Contract-Net model based on trust and reputation is shown as Fig.3.

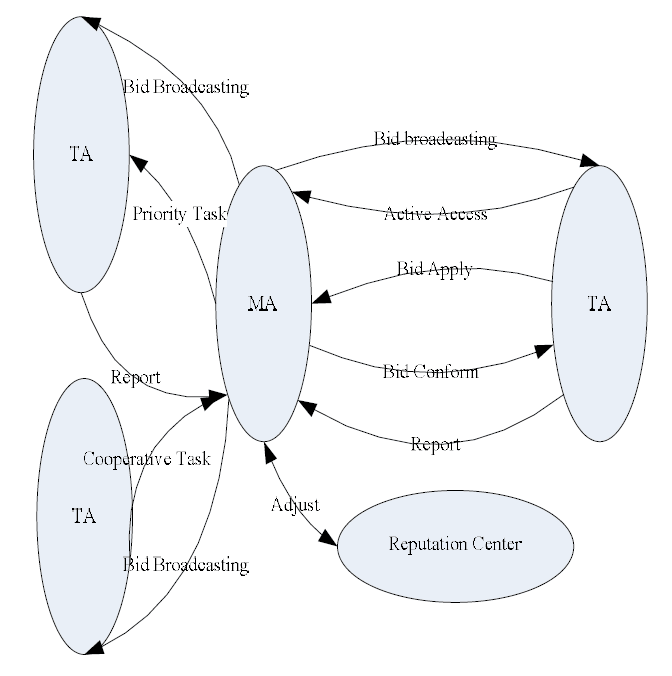


Fig.3 The Contract-Net model based on trust and reputation

**4. COOPERATION STRATEGY BASED ON TRUST DOMAIN AND REPUTATION CONTRACT NET**

The whole process of cooperation strategy is as follows:

When the task Agent is carrying out a mission, it may report to the Agent m that there is an cooperation request. This cooperation request includes two situations. The one is that task Agent can not accomplish this mission independently and task Agent need to cooperate with someone else. The other is that task Agent can not afford this mission at all.

(1) Conforming the scope for bid

Agent m make sure the task-related function domain, and then, Agent m conform Agent assembly from senior trust domain in this function domain. If the senior trust domain is empty, Agent m conform Agent assembly from the intermediate trust domain in this function domain. And system will increase trust value of these Agents through calling Rule 2. If the intermediate trust domain is empty, the scope for bid is this function domain.

(2) Broadcasting bid information

Agent m broadcast bid information within the scope conformed by step (1) on the blackboard. Bid information is defined in the following format:

Task\_Announcement= {TaskID, Initiator, Addresses, TaskDescription, TaskConstraint, Deadline}

(3) Receiving bid information

To improve cooperation efficiency and save negotiation time, this paper introduce the initiative to the blackboard access mechanism. The blackboard access threshold is set as Rep. If the reputation value of task Agent i is greater than or equal to access threshold Rep, task Agent i can access blackboard to inquire the bid information.



Whether task Agent i receive the contract bid information actively or passively, Agent i should check its bid number. To improve the efficiency of cooperation, every possible task Agent is set a contract parameter Tsend and a upper limit contract parameter Tmax. Parameter Tsend increases dynamically according to the bidding condition, until Tsend equal to Tmax, when the task Agent refused to accept any new contract bids. Once task Agent i establish formal contractual relationship with Agent m, the value of parameter Tmax turns into 0. Task Agent i sends the bid application to Agent m.

(4) Conforming cooperation Bidder

If Agent m receives the bid application of task Agent i with reputation value greater than or equal to access threshold Rep, Agent m send the cooperation conformation to task Agent i directly. Otherwise, Agent m select the most suitable bidder as the cooperative task Agent i. The method of conforming the most suitable bidder is as follow:



And then Agent m check the trust value of bidder whether greater than or equal to trust threshold Vtr. If the trust value of bidder reach trust threshold Vtr, Agent m send the cooperation confirmation; If the trust value of bidder can not reach trust threshold Vtr, Agent m will increase the trust value of bidder until equal to the trust threshold Vtr through calling Rule 1. And then Agent m send the cooperation confirmation. Parameter α, β are respectively the weight of trust and reputation, and α + β = 1.

(5) Reporting

Task Agent that executed cooperation mission sends task execution result to the Agent m. According to the task execution result, system adjusts the trust value of task Agent i through calling Rule 4. Meanwhile, the system updates the reputation value of task Agent i through calling Rule 5.

(6) Prior task

Whatever Multi-Agent System is in the process of consultation or cooperation, if the prior task arrives, the system will ignore the state of the key Agent i whether it is busy or not, delivery prior task directly to the key Agent. If the key Agent is free, it will implement prior task immediately. If the key Agent is busy, it will interrupt ongoing implementation, then report to Agent m with the cooperation request. And the interrupted task will be delivered to other Agent to perform. For the prior task, if the key Agent is able to implement the prior task independently, the key Agent will report back the implementation to Agent m. If the key Agent can not undertake the prior task independently, the key Agent will report to Agent m with the cooperation request. Like other task Agents, regardless of whether the task is prior or not, and whether the task is interrupted or not, the trust value and reputation of the key Agent will be adjusted based on the task implementation result.

**5. CONCLUSIONS**

In the open MAS, there are disadvantages in engineering applications in the classical Contract-Net. For these series disadvantages, this paper puts forward a Contract-Net model based on trust and reputation. The model's advantages are as follows:

(1) Due to the introduction of concepts such as function domain and trust domain, this model can greatly reduce the search scope of the task Agent and the network traffic to avoid network congestion.

(2) Because of the introduction of trust and reputation mechanism, this model can evaluate and record the history cooperation to promote the performance of the system, avoid blind negotiation, effectively improve the cooperation quality and the cooperation real-time.

(3) In the process of negotiation, this model can balance the workload and reduce the negotiation time of Multi-Agent System, as a result of the introduction of active perception mechanism, standard of rejecting bid and allocation strategy of prior task. In addition, this cooperation model abandons the selfishness, competition and other unfavorable factors of Agent. It is suitable for Multi-Agent System which has the same target, especially for the simulation system with frequent cooperation based on multi-Agent.