

本科生毕业设计（学术论文）



题 目 **基于区块链的分布式AI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AI算力汇聚方法**

软件工程

学生 周林 指导老师 郭兵

**[摘要]** 摘要应以概括地、不加注释地摘录本论文的研究目的、方法、结果和结论；或简洁的介绍本论文阐述的主要内容及取得的进展。客观反映原文内容，不得简单地重复标题名中已有的信息，要着重反映论文的新内容和特别强调的观点，以便读者能够很容易地知道文章的基本内容。摘要应具有独立性和自含性，不应出现图表、冗长的数学公式和非公知公用的符号、缩略语。不必要的词语，诸如副词和形容词、In this paper等，尽可能省去；背景信息、知识介绍、中间实验数据可以不要（结论性的实验数据当然还是要的）；不要说别人的缺点；过去研究的细节不要过长，可一笔带过；今后的计划不要写进去。摘要宜采用第三人称过去式的写法尽量用纯文字叙述。（如“对……进行了研究”，“综述了……”等；不应写成“本文”，“我校……”等）。摘要是提供给文献检索数据库做重复传播的情报资料，应该是一篇简明扼要的短文，一般在500～600字左右，概括介绍（1）主要研究背景，即从事这项工作的目的和重要性，（2）主要的研究内容，（3）采用的主要方法，（4）获得的主要结论，给读者一个对本论文的总体、初步的了解和认识。

**[主题词]** 区块链；合同网算法；智能合约；分布式；算力汇聚

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

Software Engineering

Student： ZHOU Lin Adviser: GUO Bing

**[Abstract]** □□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□

□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□.

**[Key Words]** □□□□□；□□□□□；□□□□□；□□□□；□□□□□□；□□□□□□；□□□□□；□□□□□□。

**目 录**

1 绪论 5

1.1 研究背景 5

1.2 国内外研究现状 5

1.3 论文主要工作 7

1.4 论文组织与结构 7

2 算法基础简介 8

2.1 合同网模型 8

2.2 黑板模型 8

2.3 本章小结 8

3 算法解决方案 8

3.1 实际问题分析（或原算法介绍） 8

3.2 合同网模型的改进 8

3.3 其他算法（算法步骤，伪代码&流程图） 10

3.4 本章小结 10

4 算法的实验验证 11

4.1 实验环境 11

4.2 实验设计 11

4.3 实验结果和分析 11

4.4 本章小结 11

5 小结 12

5.1 工作总结 12

5.2 心得体会 12

5.3 进一步的工作 12

参考文献 12

致 谢 14

附录1 源代码（模块代码主体部分） 15

# 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。

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。

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。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 相关工作

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。

**1.2.1 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。

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模型我们可以实现良好的分布式智能分配算法。

**1.2.2 分布式机器学习技术**

**1.2.2.1 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。对于机器学习算法来说， 我们往往通过建立一个模型，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。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：



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。

**1.2.2.2 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。对于最简单的BP神经网络来说，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。

## 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。
3. 基于数据并行，使用已完成的系统实现支持对糖尿病模型的线性回归分析。

## 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、该论文的主要工作做了阐述，对本文涉及到的一些理论、概念、技术做出了简要介绍，还对论文的组织与结构做出了描述。

第二章：算法基础。对该论文使用的研究工具进行了介绍，对合同网模型、黑板模型的实现进行了简单描述。

第三章：算法解决方案。改进了合同网算法，介绍了该系统实现的主要功能，并给出相应的伪代码以及流程图

第四章：算法的实验验证。测试了本文提供的算法，利用数据并行，通过糖尿病患者数据集，使用该系统对糖尿病患者进行建模。对比不同方案的结果，并且对实验结果进行分析与评价。

第五章：小结。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

## 合同网模型

* + 1. **合同网模型简介**

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。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。

* + 1. **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。

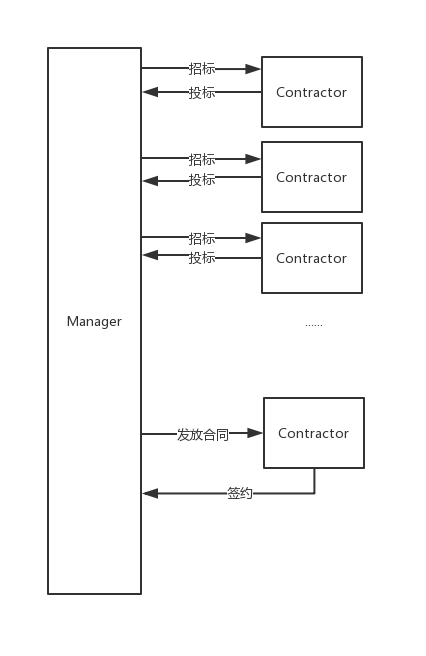
Agent之间对单个任务交互协商图如下：

图2.1

## 黑板模型

## 本章小结

# 算法解决方案

## 实际问题分析（或原算法介绍）

问题分析，分析和评价现有的3个以上相关算法或解决方案。（1500字左右）

目前学术界存在很多分布式分配算法，比如raft算法、paxos算法

## 改进合同网模型

* + 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，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。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，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。

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。

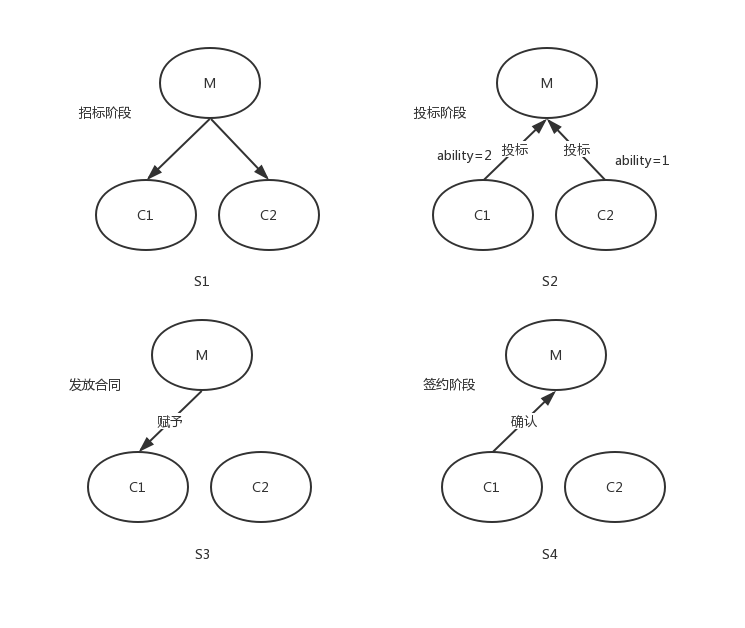


图3.1

图中分为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。

* + 1. **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 的收益。

**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，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）

其中， 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）

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。

* + 1. **详细设计**

本小节对于算法流程进行详细的描述。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3.2展示了对于单个任务处理的UML活动图。

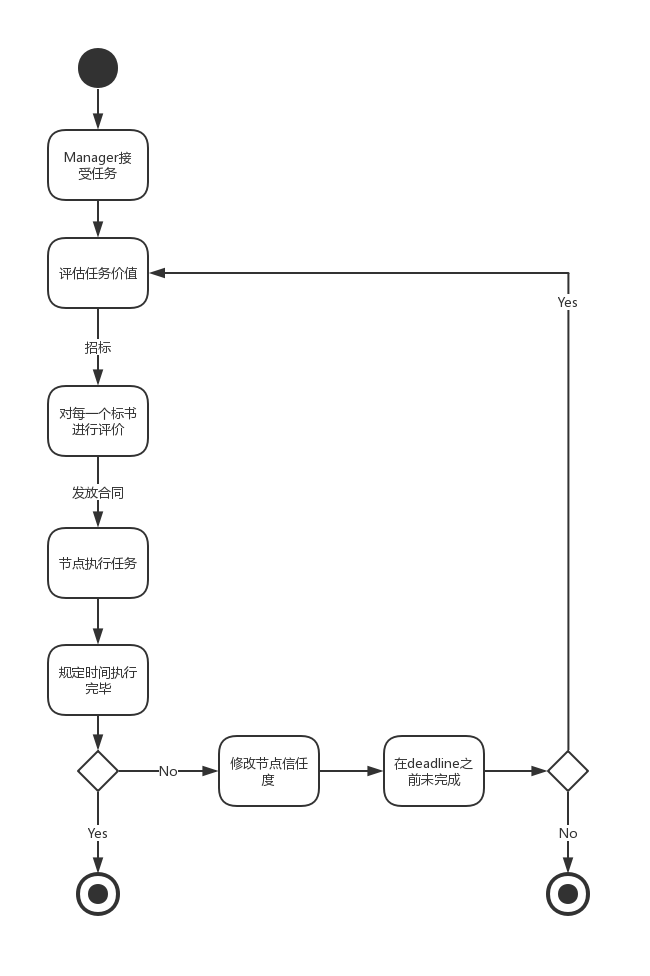


图3.2

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 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. |

## 其他算法（算法步骤，伪代码&流程图）

设计你的解决方案

## 本章小结

简要总结本章内容，突出自己的工作部分。

# 算法的实验验证

## 实验环境

说明测试硬件、软件等必要环境。

## 实验设计

说明测试方法，步骤等

## 实验结果和分析

给出测试结果，推荐用图、表方式。

## 本章小结

# 小结

## 工作总结

全面介绍毕业设计期间的工作情况，总结全过程，进度安排，出现问题及其解决，取得成果。内容分多段写。

## 心得体会

毕业设计期间取得的认识和收获，也可以对大学学习生涯有简短的总结。请结合论文工作，写出自己的真实想法。

## 进一步的工作

# 参考文献

# 致 谢

# 附录1 源代码（模块代码主体部分）