武汉理工大学毕业设计（论文）

基于CNN和CUDA的车牌识别算法优化与实现

|  |  |
| --- | --- |
| 学院（系）： | 计算机科学与技术学院 |
| 专业班级： | 软件工程SY1301班 |
| 学生姓名： | 康康 |
| 指导教师： | 袁景凌 |

学位论文原创性声明

本人郑重声明：所呈交的论文是本人在导师的指导下独立进行研究所取得的研究成果。除了文中特别加以标注引用的内容外，本论文不包括任何其他个人或集体已经发表或撰写的成果作品。本人完全意识到本声明的法律后果由本人承担。

作者签名：

年 月 日

学位论文版权使用授权书

本学位论文作者完全了解学校有关保障、使用学位论文的规定，同意学校保留并向有关学位论文管理部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅。本人授权省级优秀学士论文评选机构将本学位论文的全部或部分内容编入有关数据进行检索，可以采用影印、缩印或扫描等复制手段保存和汇编本学位论文。

本学位论文属于1、保密囗，在 年解密后适用本授权书

2、不保密囗 。

（请在以上相应方框内打“√”）

作者签名： 年 月 日

导师签名： 年 月 日

# 摘 要

云计算是在并行计算、分布式计算及网格计算等技术的基础上发展而来的。它通过将用户任务请求分布在大量设备和资源构成的数据中心上，使用户可以根据需要获取相应服务。在如今是信息化的时代，每时每刻都在产生大量的数据，对大数据的分析处理促使了云计算的普及应用及飞速发展。但随着计算任务规模和数量的不断激增、数据中心的能耗占比不断增加的环境下，如何对任务进行合理有效的调度以实现高效率和低能耗显得尤为必要。

数据中心作为云计算资源和设备的集中之地，其重要程度不言而喻。在云计算数据中心任务调度和节能部署这方面的探索研究中，许多行之有效的调度算法不断的被相关学者所提出，对它们的分析研究发现，大部分算法在调度时针对的是全体资源，而没有考虑到任务和资源自身的特性。实际中，数据中心的任务和资源一般具有规模大、异构多样性的特点，而且任务的到达具有随机性，根据数据中心的任务和资源的上述特点，设计及实现了基于动态任务流的节能部署调度方案和基于数据中心任务与资源的等价类调度算法。主要学习工作如下：

（1）根据任务到达的随机性，将任务流以时间段分割，并以每个时间段里的任务数量确定所需开启虚拟机的数目，从而达到降低数据中心能耗的目的。

（2）根据数据中心任务和资源异构性、多样性的特点，建立数据中心任务及资源模型，并运用等价类算法进行任务及资源的等价类划分，实现资源和任务分类过程，降低任务在资源选择上的开销，缩短任务总执行时间。

（3）针对贪心算法在任务调度时相比其他简单调度算法（如顺序调度）在性能方面优越性，在上述等价类划分的基础上，在每一个任务和资源的等价类内部使用贪心策略，实现性能最优化。

（4）利用CloudSim模拟仿真软件，实现本文算法的仿真。仿真实验结果表明，上述所提到的部署方案和调度算法在一定程度上实现降低数据中心能耗和减少任务总执行时间的目的，同时印证了本文算法的可行性。

关键字：

# **Abstract**

Cloud computing is the development of parallel computing, distributed computing and grid computing. It distributes tasks to a data center constituted of many computers, which enabling users to get services on their demand. In today's information age, a large amount of data generates on every moment, the analysis of large data lead to the popularity and rapid development of cloud computing. But with the surge of computing tasks and the energy consumption of data center, It is necessary to schedule tasks effectively in order to realize the high efficiency and low energy consumption.

As the warehouse of resources and equipments, data center play an importment role in cloud computing. On the research of task scheduling and energy conservation, many effective scheduling algorithm was put forward by relevant scholars, but most of the algorithms is aiming at all the resources in scheduling, without considering the own features of the tasks and resources. Actually, tasks and resources generally has the features of large scale and heterogeneous diversity, and the arrival of the task is random, according to the above characteristics, we designed the energy-saving scheduling algorithm based on dynamic task flow and the equivalence class scheduling algorithm for data center tasks and resources. The main study work is as follows:

(1) According to the random arrive of task, we divide the task flow to period, and the amount of virtual machine to open is depended on the task amount of each time period, so as to achieve the purpose of reduce energy consumption of data center.

(2) According to the heterogeneity and diversity of task and resource, we establish the model of task and resource, and using equivalence class algorithm to finish the division of tasks and resources, so as to increase efficiency and reduce the total execution time.

(3) In view of that greedy algorithm is more effective than other simple scheduling algorithm (such as order scheduling)on task scheduling. we use greedy strategy in each of the tasks and resources which is divided by equivalence class algorithm to achieve performance optimization.

(4) We use cloudsim simulation software to achieve the simulation of algorithm in this paper. the simulation results confirms the feasibility of the above algorithms.

**Keyword:**

# 目 录

[摘 要 I](#_Toc452026090)

[**Abstract** II](#_Toc452026091)

[目 录 III](#_Toc452026092)

[第1章 绪论 1](#_Toc452026093)

[1.1 研究背景及意义 1](#_Toc452026094)

[1.2 国内外研究现状 1](#_Toc452026095)

[1.3 研究路线及工作内容 2](#_Toc452026096)

[1.4 论文主要结构 3](#_Toc452026097)

[第2章 数据中心粗粒度任务调度策略 5](#_Toc452026098)

[2.1 数据中心概述 5](#_Toc452026099)

[2.2 数据中心资源与任务调度特点分析 6](#_Toc452026100)

[2.3 数据中心动态任务流节能部署调度方案 7](#_Toc452026101)

[2.3.1 节能部署方案描述 7](#_Toc452026102)

[2.3.2 节能部署方案模型 7](#_Toc452026103)

[2.4 数据中心等价类调度模型 8](#_Toc452026104)

[2.4.1 任务及资源模型 8](#_Toc452026105)

[2.4.2 等价类算法 9](#_Toc452026106)

[2.4.3 调度策略 11](#_Toc452026107)

[2.5 本章小结 12](#_Toc452026108)

[第3章 基于数据中心贪心模型任务调度 13](#_Toc452026109)

[3.1 贪心算法 13](#_Toc452026110)

[3.2 基于数据中心任务调度贪心算法 13](#_Toc452026111)

[3.2.1 数据中心任务调度的形式化描述 13](#_Toc452026112)

[3.2.2 传统的任务调度算法 14](#_Toc452026113)

[3.2.3 基于数据中心任务调度的贪心算法 14](#_Toc452026114)

[3.3 基于等价类的贪心策略部署 25](#_Toc452026115)

[3.4 本章小结 26](#_Toc452026116)

[第4章 CloudSim仿真实验 17](#_Toc452026117)

[4.1 CloudSim简介 17](#_Toc452026118)

[4.1.1 CloudSim仿真框架特性 28](#_Toc452026119)

[4.1.2 CloudSim体系架构 28](#_Toc452026120)

[4.2 CloudSim技术实现 31](#_Toc452026121)

[4.2.1 CloudSim类图及核心类 31](#_Toc452026122)

[4.2.2 任务处理过程 20](#_Toc452026123)

[4.3 CloudSim模拟仿真步骤 34](#_Toc452026124)

[4.4 基于动态任务流的节能部署方案 21](#_Toc452026125)

[4.4.1 仿真方案场景 21](#_Toc452026126)

[4.4.2 仿真方案流程 36](#_Toc452026127)

[4.4.3 实验结果分析 22](#_Toc452026128)

[4.5 基于任务与资源的等价类调度算法 23](#_Toc452026129)

[4.5.1 任务描述 23](#_Toc452026130)

[4.5.2 集群模型模拟 39](#_Toc452026131)

[4.5.3 仿真实验流程 24](#_Toc452026132)

[4.5.4 实验结果分析 41](#_Toc452026133)

[4.6 本章小结 44](#_Toc452026134)

[第5章 总结与展望 27](#_Toc452026135)

[5.1 论文总结 27](#_Toc452026136)

[5.2 工作展望 46](#_Toc452026137)

[致谢 28](#_Toc452026138)

[参考文献 29](#_Toc452026139)

# 

# 第1章 绪论

## 1.1 研究背景及意义

[车牌识别系统](http://baike.baidu.com/item/%E8%BD%A6%E7%89%8C%E8%AF%86%E5%88%AB%E7%B3%BB%E7%BB%9F)(Vehicle License Plate Recognition，VLPR) 是计算机视频图像识别技术在车辆牌照识别中的一种应用。

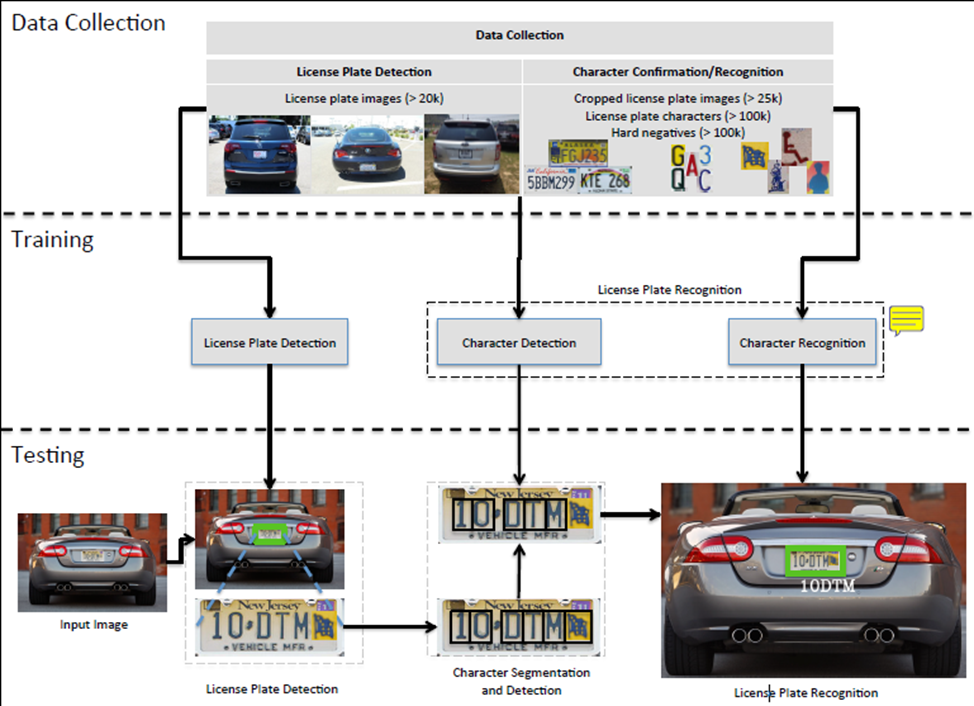
车牌识别技术要求能够将运动中的汽车牌照从复杂背景中提取并识别出来，通过车牌提取、图像预处理、特征提取、车牌字符识别等技术，识别车辆牌号、颜色等信息，目前最新的技术水平为字母和数字的识别率可达到99.7%，汉字的识别率可达到99%。车牌识别在[高速公路](http://baike.baidu.com/item/%E9%AB%98%E9%80%9F%E5%85%AC%E8%B7%AF)车辆管理中得到广泛应用，电子收费（ETC）系统中，也是结合DSRC技术识别车辆身份的主要手段。

车牌识别技术结合电子不停车收费系统（ETC）识别车辆，过往车辆通过道口时无须停车，即能够实现车辆身份[自动识别](http://baike.baidu.com/item/%E8%87%AA%E5%8A%A8%E8%AF%86%E5%88%AB)、自动收费。在车场管理中，为提高出入口车辆通行效率，车牌识别针对无需收[停车费](http://baike.baidu.com/item/%E5%81%9C%E8%BD%A6%E8%B4%B9)的车辆（如月卡车、内部免费通行车辆），建设[无人值守](http://baike.baidu.com/item/%E6%97%A0%E4%BA%BA%E5%80%BC%E5%AE%88)的快速通道，免取卡、不停车的出入体验，正改变出入停车场的管理模式。

## 1.2 国内外研究现状

车牌检测与识别技术作为一个集商业与公共于一身的项目，无论是学术界还是商业界对它的重视程度相当高。从运输和通行违规到事故监测，自动检测和车牌识别的能力是世界各地执法机构的关键工具之一。由于条件和车牌类型的差异，车牌检测和识别仍然是一个具有挑战性的问题。大多数现有解决方案本质上是限制性的，即对于特定类型的车牌模板，以特定分辨率为特定视角进行固定照相机的工作。这意味着，例如，为静态摄像机设计的车牌系统在移动摄像机表现不佳。另外，由于处理速度的瓶颈，大多数技术严重依赖启发式方法。这在一个不同类型的特征的组合已经表现出比为给定任务选择单个特征表现更好的世界不太理想。

随着深度CNN的出现[1]和增强[2]，以及更快的处理硬件的出现，我们需要重新审视这个问题。 CNN的设计进展导致许多任务的性能准确度显着提高。

图XXX 研究路线

## 1.3 研究路线及工作内容

（1）图XXX展示了本文的研究路线，研究路线主要分为三个阶段：收集数据、训练网络及测试。首先，将收集的数据进行清理、组织和适当地注释（以半自动的方式），然后进行CNN的训练。每个阶段分为三个模块：车牌检测、字符切割及字符识别。1）车牌检测器用于发现和裁剪车牌; 2）字符分割用于分离每个字符; 3）使用字符检测模块去掉非字符元素， 4）通过识别系统识别每个剩余字符，以获得正牌号码。

（2）本文主要工作内容包含以下几点：

1. 车牌检测部分采用Maximally Stable Extremal Regions（MSER）和BP神经网络算法[3]。车牌检测是车牌识别的最重要部分，其质量直接影响车牌识别的结果。 在自然光线下，车牌位置需要克服不均匀的照明、车牌倾斜角度、复杂的背景条件等。本文中，车牌位置使用的方法主要是基于MSER算法和多重分类。
2. 根据拥有的中国车牌分割识别的知识，本文提出了一套关于车牌分割的算法，即采用垂直投影方法进行字符分割。
3. 在字符识别中，卷积神经网络（CNN）的性能优于其他大多数模型。 然而，为了保证令人满意的表现，CNN通常需要大量的样本。 由于中文和字母数字的差异，识别两类的最常用方法是分别使用两个独立的CNN。 在本文中，为了解决汉字短缺的问题，我们实现了一个CNN模型，分别为汉语和字母数字字符共享隐藏层和两个不同的softmax层。

## 1.4 论文主要结构

第一章 绪论。介绍了论文研究背景、意义及国内外研究现状，综合阐述了论文的主要工作和组织结构。

第二章 卷积神经网络基础知识。本章从基本的BP神经网络开始，进而介绍了卷积神经网络以及梯度下降算法。

第三章 车牌定位算法。

第四章 字符切割算法。

第五章 字符识别算法。

第六章 CUDA环境介绍。

第七章 总结与展望。



图1.2 论文主要组织结构

# 第2章 数据中心粗粒度任务调度策略

## 2.1 数据中心概述

数据中心是众多计算机设备与资源的集中之地，同时对这些设备和资源提供能源供给和维护。数据中心既可以单独建设，也可以置于其他建筑之内，可能是分布在不同地理位置的多个网络系统[16]。

为数据中心动态分配和管理物理和虚拟共享资源面临着许多问题与挑战：物理资源如何与虚拟资源高效结合；云计算应用平台资源具有广泛性、多样性以及实时动态变化性等特点；系统性能和成本因素的考虑等。以上的问题与挑战都是在实际生活中经常遇到的，所以需要设计高效率的云计算数据中心任务资源调度算法，以适应和满足不同业务场合的不同需求。

图2.1简单阐明了一个多数据中心任务资源调度算法的体系结构，大致流程描述如下：

① 用户请求：用户通过任务请求的方式向数据中心发出请求。

② 调度管理：根据用户任务请求数量和特性，找到数据中心合适的资源，并使用合适的调度算法进行任务调度。

③ 使用资源：数据中心找到可用的资源，并把资源信息展现给用户。



图2.1 数据中心任务调度体系结构

## 2.2 数据中心资源与任务调度特点分析

数据中心聚集了大量的计算机设备与资源，同时它处理来自不同用户的海量不同任务请求，完成用户的各种计算需求。显然，数据中心在给我们带来方便的同时，我们也对其提出了更高的要求。在建立云计算数据中心和应用平台后，一项非常重要的技术就是如何把云计算数据中心的虚拟共享资源合理有效的分配到不同用户的不同请求，从而达到提高资源使用效率的目的。图2.2为数据中心的简单调度管理流程图。



图2.2 数据中心任务调度管理流程

要实现数据中心任务调度的低成本、高效和易用性，仍然面临着许多的挑战，其中一个亟待解决的现状就是数据中心的高能耗问题。造成数据中心高能耗的原因还是和数据中心资源和任务的特点相关：

（1）数据中心任务到达的随机性。由于用户提交任务的不确定性，导致了数据中心任务的随机性，如在一个数据中心，早上五六点的任务请求数肯定比晚上七八点时的任务请求数少，就算在一天中的高峰期，任务的请求数也是动态变化的。而数据中心为满足能处理峰值时的任务请求数，通常将所有服务节点都开启，在任务空闲时间段这些服务节点就会因没任务而空转，这其中就导致了很多不必要的资源浪费；

（2）由于数据中心不同的计算设备通常有不同的配置，同时处理任务的效率也不尽相同。若任务分配不当，使高性能的服务节点去处理简单的任务，就好比让超算中心去计算简单的加减乘除，虽然确定能得到正确的结果，但也未免太过“奢侈”，同时也造成能耗的浪费和效率低下。

数据中心节点的“空闲”和任务分配不当造成的“奢侈”都是造成数据中心高能耗的重要原因。

## 2.3 数据中心动态任务流节能部署调度方案

### 2.3.1 节能部署方案描述

由于现实世界中数据中心所处理的任务数是动态变化的，而运营商在部署集群时通常都是按照当前集群所能处理任务的峰值设定的。但实际在运行的过程中，大部分的时间都不会达到集群的峰值，这带来的问题就是能耗的浪费。在全球气候变暖、能源资源紧张的环境下，节能减排或许是我们必须重视的问题。根据数据中心任务到达的随机性和任务数量的不确定性，以时间轴为间隔进行任务时间段的划分，统计每个时间段的任务特征，确定每个时间段所需开启的虚拟机数目，以此来降低数据中心的能耗。

### 2.3.2 节能部署方案模型

本调度方案主要考虑的是动态任务流的节能部署调度，为简化工作的复杂性以得到更明显的效果，本方案假定所有任务及资源配置相同，即在同构的环境下研究该调度方案对数据中心能耗的影响。

如图2.3，该图模拟了数据中心动态任务流的到达模型，任务的产生是按时间节点进行的，不同的时间点根据不同用户的不同任务请求，所以产生的任务数不同。时间假定为单位时间，该图表明每单位时间产生的任务数，正如图所表明的，任务的数量随着时间在不断地波动。



图2.3 模拟动态任务流

图2.4实现了将任务流以时间段进行分割，我们可以根据自己的需要将时间轴划分为不同的时间段，以下的划分是每5个单位时间为一个时间段，任务流共被划分为3个时间段。



图2.4 分段后的动态任务流

各阶段对应的任务数为

时间段1：20、10、30、20、40

时间段2：40、50、30、50、40

时间段3：30、20、30、20、10

因为在单位时间内所有的虚拟机处理能力是一定的，所以在每个时间段内只需开启特定数目的虚拟机，该虚拟机数目必须能满足处理该时间内最大任务数的能力。如在时间段1内，最大任务数是40，假定只需开启40台虚拟机就能满足在单位时间内处理完最大任务数，那么在该时间段内所有任务都能按时完成。

## 2.4 数据中心等价类调度模型

### 2.4.1 任务及资源模型

在一般的任务调度算法中，任务在选取资源时大多针对的是全体资源，没有考虑到资源自身的特性，所以会造成任务在选取资源时的时间开销过大。因此，用合适的方法对任务和资源进行划分分类，缩小任务在选择资源时的选择范围，提高任务的执行效率[17]。

任务模型： 设T={t1，t2，…，tn}表示任务的集合，第i个任务我们可以对其进一步细化特征E=（e1，e2，…，en），n表示特征属性个数，这里我们选取了任务中最常用的四个特征属性，各个特征属性描述如下：

①e1表示任务的长度，是任务在执行过程中完成时间的重要参考指标。

②e2表示了任务执行时最优匹配的系统类型，当任务类型和资源类型匹配时，执行效果最佳，否则会增加额外的时间开销。

③e3表示任务的输入文件大小，和资源的带宽相对应。

④e4表示任务执行完成后输出文件大小。

资源模型： 设P={p1，p2，…，pn}表示资源的集合，第i个资源可以进一步为其细化特征Q=（q1，q2，…，qn），n表示特征属性个数，这里我们选取了资源中最常用的五个特征属性，各个特征属性描述如下（注：因任务的实际处理是由虚拟机完成的，故这里的资源指虚拟机）：

①q1表示虚拟机的执行能力MIPS（Million Instructions Per Second），对应任务中的长度属性。

②q2表示了该虚拟机对应的系统类型。

③q3表示虚拟机的容量。

④q4表示虚拟机的内存，即随机存取存储器。

⑤q5表示虚拟机的带宽，带宽越大，虚拟机单位时间传输量越大。

### 2.4.2 等价类算法

（1）定义：设五元组表示一个数据中心决策表，其中代表的是对象的非空有限集合，称作论域；代表的是条件属性的非空有限集；代表的是决策属性的非空有限集且；，是属性的值域，是一个信息函数，它对每个对象的每个属性赋予一个信息值，即；每一个属性子集决定了一个二元不可区分关系:



关系构成了的一个划分，用表示，简记，中的任何元素称为等价类[18]。

（2）实例分析过程：对表2.1的决策表进行等价类划分

表2.1 决策表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **q1** | **q2** | **q3** | **q4** | **Q** |
| **X1** | 2 | 1 | 1 | 3 | 1 |
| **X2** | 2 | 1 | 1 | 1 | 2 |
| **X3** | 1 | 1 | 2 | 3 | 0 |
| **X4** | 3 | 2 | 1 | 2 | 2 |
| **X5** | 2 | 1 | 1 | 3 | 1 |
| **X6** | 3 | 2 | 1 | 2 | 3 |
| **X7** | 1 | 1 | 2 | 3 | 0 |
| **X8** | 3 | 2 | 1 | 2 | 3 |
| **X9** | 1 | 2 | 1 | 3 | 2 |
| **X10** | 4 | 2 | 3 | 1 | 3 |
| **X11** | 2 | 1 | 1 | 1 | 2 |
| **X12** | 4 | 2 | 3 | 1 | 0 |
| **X13** | 1 | 2 | 1 | 3 | 1 |
| **X14** | 3 | 3 | 2 | 2 | 0 |
| **X15** | 1 | 2 | 1 | 3 | 1 |

将决策表转换成以上定义描述的知识系统，我们可以令，，则。

根据以上知识系统，我们可以定义四个等价类关系：q1，q2，q3，q4，通过这些等价关系，可以得到下面四个等价类

















这些等价类是由知识库中的初等范畴构成的，而基本范畴是由初等范畴构成的，如下面表示的一些集合：

=





它们都是的基本范畴，则要求上的等价类，只需求U在上的基本范畴。根据初等范畴的交集，求得U在上的等价类为



### 2.4.3 调度策略

按照以上等价类划分算法和任务资源模型，我们设计了任务与资源的等价类调度算法，该算法主要分为如下步骤

（1）对任务和资源的等价类划分。在上述的任务及资源模型中，我们设定了它们的基本特征属性。首先我们对一些特征属性进行内部分区处理(如对任务的长度分为20000-40000、40000-60000等长度段，每一段代表新的一类特征值)，属性分类后按等价类划分算法实现对任务和资源划分。

（2）每一个任务和资源划分后，其对应的等价类都指定了一个标识，该标识表明了该等价类资源处理任务能力的强弱，若为任务，则表明任务长度的大小。然后将任务长度长的等价类组分配给资源处理能力强的等价类组。



图2.5 数据中心任务与资源等价类调度

算法流程：

输入：任务请求列表

输出：所有任务完成的总时间之和

1. 初始化虚拟机及任务列表

2. for i=from 1 to m do

对于所有的虚拟机，确定其所属的等价类组，存入Map中，其中Key 表示该等价类虚拟机执行能力强弱。

endfor

3. for j=from 1 to n do

对于所有的任务，确定其所属的等价类组，存入Map中，其中Key 代表了该等价类任务的长度大小。

endfor

4. 迭代任务等价类的Map

if 任务等价类的Key和虚拟机等价类的Key相匹配

将该等价类任务分配给该等价类虚拟机

else

找到和该等价类相匹配的虚拟机组，分配任务

endif

## 2.5 本章小结

本章首先介绍了数据中心的基本概念，同时在对数据中心理解的基础上，研究和分析了数据中心资源和任务的调度特点，介绍了数据中心任务调度遇到的一些问题以及产生这些问题的原因。最后，根据数据中心任务到达随机性的特点，提出了动态任务流的节能部署调度方案；根据任务和资源的异构性，提出了任务和资源的等价类调度算法。

# 第3章 基于数据中心贪心模型任务调度

## 3.1 贪心算法

（1）贪心算法思想

从字面意思上看，贪心算法考虑的范畴是每一个局部，而不是从问题的整体考虑，它总是在每一步作出认为是当前最好的选择。虽然这样每个局部最优得到的结果未必是整体最优的结果，但对于许多问题它都能产生最优解，即使不是最优解，其最终结果也是最优解的很好近似。

（2）贪心算法的基本要素

|  |  |
| --- | --- |
| **①贪心选择性质** | **所谓贪心选择性质指的是所求解问题的最优解可以通过一系列局部最优解，即贪心选择得到。这是贪心算法可行的第一个基本要素，也是区别动态规划算法的主要特征。** |
| **②最优子结构** | 当一个问题的最优解包含其子问题的最优解时，称此问题具有最优子结构性质。问题的最优子结构性质是该问题可用动态规划算法或贪心算法的关键特征。 |

（3）贪心算法的基本思路

从问题给定的某一初始解开始，逐渐逼近最终的目标，从而更快的得到更好的解，当进行到算法某一步不能再进行时，算法结束。

算法实现过程：

从某一问题的初始解出发

while 能朝给定的总目标前进一步 do

求出一个可行的解

由所有解元素组成问题的一个可行解

## 3.2 基于数据中心任务调度贪心算法

### 3.2.1 数据中心任务调度的形式化描述

定义1：假设在数据中心任务调度环境下，把用户提交的任务请求分解成m个任务的集合，同时认为各任务之间相互独立，在任务调度时不需要考虑任务之间的约束及关联关系。定义为任务的集合，其中表示分解后任务的数量，ti表示第个任务，且任务ti的总指令长度为。

定义2：假设在数据中心任务调度环境下，参与任务调度的虚拟资源有n个，而且该虚拟资源对应于数据中心集群平台中的虚拟机。定义为虚拟机资源集合，为资源数量，vi表示第个虚拟机资源。第i个虚拟机的指令执行速度为。

定义3：假设数据中心中的任务数量大于或等于虚拟机的数量，一个任务只能分配给一个虚拟机执行，而且一个虚拟机在执行过程中，同时间段不能去执行其他任务。定义表示第个任务ti在第j个虚拟机vj上的执行时间，时间计算公式为，则m个任务在n个虚拟机上的执行时间对应为一个的矩阵

### 3.2.2 传统的任务调度算法

（1）顺序调度算法：顺序调度是将一组任务顺序的分配给虚拟机，是最基本的调度策略。顺序调度能保证每个虚拟机分配到相同数量的任务，实现简单的均衡负载，但该算法没有考虑到任务和虚拟机之间的差异性，故在执行效率上不是很理想。

（2）Min-Min[19]调度算法：Min-Min算法思想是，尽可能的将任务长度小的任务分配给执行能力强的虚拟机，以实现所有任务的总完成时间最小。Min-Min算法首先统计每个任务在各个虚拟机上的最小执行时间，然后选择这些最小完成时间中的最小值，实现任务的分配调度。

Min-Min算法能实现任务的总体完成时间最小，但是它将小任务优先分配给处理能力强的虚拟机也导致一个问题，就是执行能力强的虚拟机过度负载而使一些执行能力弱的虚拟机空闲，导致系统负载不均衡。

（3）Max-Min[20]调度算法：Max-Min算法思想和Min-Min算法基本类似，只不过Max-Min算法是优先调度大任务。它从上述最小完成时间中选择最大值，完成任务的分配。

Max-Min算法在许多情况下比Min-Min算法更好，但这种优先调度大任务的策略也会导致系统的负载不均衡。

### 3.2.3 基于数据中心任务调度的贪心算法

实际生活中，任务之间和虚拟机之间的配置参数不可能完全一样。顺序调度算法实现简单，但是忽略了它们之间的差异性，如任务的指令长度和虚拟机的执行速度等；Min-Min算法和Max-Min算法虽然考虑到了这些，但是都没能实现系统的均衡负载。如是，提出了贪心算法的思想，它在考虑任务和虚拟机配置不一样的情况下，既能实现所有任务的总完成时间接近最短，也能兼顾系统的负载均衡。

基于数据中心任务调度的贪心算法考虑到了任务和虚拟机之间配置不一样的情况，这样任务在不同虚拟机上执行的时间就不同。此时，根据以上贪心算法的思想，任务在选择虚拟机的时候使用贪心策略，确保每次选择的虚拟机都是最佳的，从而减少任务的执行时间。算法假定与描述如下：

（1）假定：任务的完成时间只取决于任务的指令长度(MI)和虚拟机的执行能力(MIPS)。根据此，有如下结论成立：

①如果许多任务运行在同一个虚拟机上，不论使用哪种共享策略，这些任务的总完成时间是不变的，因为任务的总指令长度和虚拟机的执行速度不变。

②如果某个任务相比如其他任务来说，在一个虚拟机上能最快完成，则它在其他虚拟机上也是最快完成的。

③如果某个虚拟机的执行能力最强，则它执行所有任务都比其他虚拟机快。

（2）算法描述：假设m个任务分配到n个虚拟机上，因任务完成时间只取决于任务指令长度和虚拟机执行能力，所以我们用一个矩阵表示第个任务在第个虚拟机上的完成时间，显然有。在对矩阵初始化前，我们先把任务以指令长度的大小降序排列，将虚拟机以执行能力升序排列。

经以上排序后，矩阵的行号和之前的任务不再一致，同样列号和虚拟机的号也被打乱。如是，矩阵的每行、每列都降序排列，然后以为对象使用贪心策略。选用的贪心策略是：从矩阵第一行的任务开始，每次把任务分配给该行最后一列的虚拟机，如果该分配是最佳的，则将该任务分配到此虚拟机，否则寻找除此虚拟机外的最佳虚拟机。如果不止一种分配方案使分配结果最佳，则选择其中运行任务数最小的虚拟机完成分配，保证系统简单的均衡负载[21]。这种算法的思想反映了指令长度越长的任务越需要执行能力越强的虚拟机来执行，以解决负杂任务造成的瓶颈，降低所有任务的总执行时间。

## 3.3 基于等价类的贪心策略部署

在上一章中我们介绍了数据中心任务和资源的等价类调度算法，该算法在整体方面对任务和资源进行了粗粒度的划分，然后再对划分后的任务和资源进行调度。结合本章我们的对贪心算法的分析了解，以及贪心算法在实际数据中心任务调度中运用，提出了一种数据中心等价类-贪心算法的任务调度策略，即在等价类调度的基础上接着使用贪心算法。

贪心算法在原则上应满足数据中心任务数不小于资源数，而在实际中这一条件很容易满足，即使经过等价类划分后，划分后每个等价类任务中的任务数也基本大于每个等价类资源中的资源数。假设等价类任务组T(A)={t1，t2，…，tm}匹配等价类资源组V(A)={v1，v2，…，vn}，则T(A)和V(A)满足贪心算法调度条件，我们对它使用贪心算法。对等价类任务和资源进行贪心算法的基本流程如图3-1所示。



图3-1 等价类贪心调度

## 3.4 本章小结

本章介绍了基于数据中心贪心模型的任务调度策略。为了在数据中心应用贪心算法，我们首先对贪心算法的基本概念作了介绍，然后结合贪心算法的思想，提出了在数据中心任务调度过程中的贪心策略模型。在介绍任务调度贪心模型时，着重介绍了数据中心任务调度的形式化描述以及传统的一些任务调度所采用的一些方法及面临的问题，然后顺势引出贪心算法。最后，在对贪心算法的理解基础上，结合上一章提出的等价类调度算法，我们整合了这两种算法，同时运用到任务调度的过程中，为后期仿真实验做准备。

# 第4章 CloudSim仿真实验

## 4.1 CloudSim简介

CloudSim是澳大利亚墨尔本大学Rajkumar Buyya教授领导团队开发的云计算仿真器，它是一个基于事件的仿真器，实体和实体间基于消息进行通讯。它的主要目的是在基于软件、硬件、服务的云基础设施上，对不同的应用程序和调度策略进行性能的比较。它的开发使用户能够更加方便的测试自己的服务，使自己的服务在部署实施之前调节性能瓶颈，不仅给用户的开发带来了方便，同时也节约了大量的资金成本。

### 4.1.1 CloudSim仿真框架特性

（1）大规模的云计算仿真与实例化可以在一个节点上完成。

（2）具有独立仿真平台，可以对数据中心、服务代理和分配调度策略进行管理。

（3）可以在一个数据中心模拟和创建多个虚拟化服务。

（4）提供多种共享策略供用户选择，用户可根据需求灵活切换服务。

### 4.1.2 CloudSim体系架构

CloudSim采用了分层的体系结构，其软件框架的多层设计特征和体系结构如图4-1所示：



图4-1 CloudSim体系架构图

（1）CloudSim核心模拟引擎层

该层主要涉及到CloudSim核心模拟引擎内部的实现细节，其中最主要的类是CloudSim，它是模拟引擎的主类，负责管理事件队列和控制仿真事件有条不紊的顺序进行。其中每个事件都会根据其时间参数构成有序队列存放在未来事件队列（Feture Event Queue）中，执行时从未来事件队列中删除并放到延时事件队列（Deffered Event Queue）中，最后每个事件触发内部事件代码，执行相应的操作。这种事件的管理方式具有较强的灵活性和其独特的优势。



图4-2 CloudSim核心模拟引擎类图

（2）CloudSim层

CloudSim层为云计算模拟和仿真提供底层支持，其中包括主机、虚拟机、内存、磁盘和带宽的专用管理接口。该层处理云计算模拟时的一些基本问题，如如何为不同的虚拟机分配不同的主机以及管理应用程序的执行、监控系统状态的动态变化等。其中虚拟机分配涉及到CloudSim中的虚拟机分配策略类VmScheduler，CloudSim自带了基本的虚拟机分配策略，同时CloudSim为云应用开发者提供了接口，使我们可以通过这一层实现自己的调度策略，只需扩展相应的功能就可以实现复杂的工作负载分析和应用性能研究。CloudSim层又可以细分为5层：网络层、云资源层、云服务层、虚拟机服务层、用户接口结构层。

（3）用户代码层

这是CloudSim中用户接触最多的层，该层提供了一些实体如主机、虚拟机、任务以及一些调度策略供用户调用。我们可以对这些提供的实体进行扩展以完成自己的云任务应用。

## 4.2 CloudSim技术实现

### 4.2.1 CloudSim类图及核心类

图4-3是CloudSim云仿真器的类设计图。该图中只列出了CloudSim的最基本的类，正是这些类完成了一个云任务环境的模拟。



图4-3 CloudSim类设计图

主要核心类功能描述如下

（1）Cloudlet：该类代表了云计算环境下的任务，主要指标参数是任务长度、输入输出文件大小。

（2）Datacenter：该类代表了一个数据中心，也即虚拟化资源中心。它里面可设置不同数目的主机，通过不同分配策略为主机分配不同的内存、带宽和容量等参数。

（3）Vm：虚拟机类，分配在主机上，实际对任务进行处理的类，主要性能指标参数MIPS，和任务中长度对应。

（4）Host：主机，数据中心资源的表现形式，包括内存、带宽、容量等参数。

（5）DatacenterBroker：数据中心代理对象，它隐藏了虚拟机的管理，如Vm的创建，Vm上任务的提交以及销毁Vm等一些操作，只保留用户关心东西。

（6）VmAllocationPolicy：虚拟机分配策略，指定将虚拟机分配到哪台主机上。

（7）VmScheduler：虚拟机资源分配策略，指定如何为虚拟机分配内存、MIPS、带宽、容量等资源。

（8）CloudletScheduler：任务的分配策略类，为每个虚拟机中的任务分配处理能力。自带了两种调度策略：空间共享（CloudletScheduler SpaceShared）和时间共享（CloudletSchedulerTimeShared）。

### 4.2.2 任务处理过程

任务单元的处理是由虚拟机完成的，因此任务的处理过程必须在仿真的每一个时刻被更新。为实现这个目的，在仿真过程中通过事件的方式来告知数据中心下一时刻需要处理的任务[20]。任务的处理及更新流程如图4-4所示：



图4-4 任务处理过程图

## 4.3 CloudSim模拟仿真步骤

整个仿真流程大致可以分为以下几步：

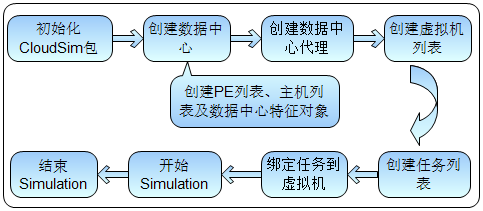


图4-5 仿真流程图

## 4.4 基于动态任务流的节能部署方案

### 4.4.1 仿真方案场景

表4-1展示了各型号节点的CPU及功耗数据。本次仿真实验中，为简化实验过程，确保实验结果的可靠性，假定数据中心服务节点配置相同。选取的功耗数据模型为Intel Pentium M745，空载功耗71.6W，满载功耗103.2W。

表4-1 各型号节点的CPU及功耗数据

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **CPU** | **频率**  **(GHz)** | **空载功**  **耗(W)** | **满载功**  **耗(W)** | **负载能力(reg/s)** | **CPU利**  **用率(%)** |
| **AMD Athlon** | 1.0 | 63.9 | 71.6 | 79 | 83.1 |
| **64 3800+** | 1.8 | 67.2 | 85.5 | 159 | 91.1 |
|  | 2.2 | 69.9 | 96.5 | 200 | 93.1 |
| **Intel Pentium** | 2.4 | 71.6 | 103.2 | 221 | 93.9 |
| **M 745** | 1.2 | 44.0 | 49.0 | 99 | 86.2 |
|  | 1.8 | 49.0 | 60.0 | 159 | 91.1 |
| **AMD Athlon** | 1.0 | 66.3 | 81.5 | 180 | 92.3 |
| **64 X2 3800+** | 1.8 | 70.5 | 101.8 | 351 | 97.3 |
| **Dual Core** | 2.0 | 72.7 | 109.8 | 395 | 97.9 |

仿真实验中资源及任务参数设置：

主机：MIPS为1500、RAM为2048、storage为20000、BW为2000

虚拟机：MIPS为1000、RAM为512、storage为10000、BW为1000

任务：长度为1000、输入文件大小300、输出文件大小300

其中主机数和虚拟机数对应的都是60，也即一台主机上只分配一台虚拟机，虚拟机分配依照的是VmAllocationSimple算法。任务数量随时间动态产生，实验中假定每五个单位时间为一组任务的时间间隔。

### 4.4.2 仿真方案流程

（1）首先通过随机任务流文件构建随机任务参数列表，然后读取任务参数列表文件构建任务列表。任务通过时间段的划分存放在不同的List中，然后所有的List存放在以Integer为Key、以List为Value的Map<Integer,List<Integer>>中来构成任务流。

（2）然后通过对CloudSim中如何动态添加实体方法的研究，得知在CloudSim运行过程中添加实体的方法。通过在程序中使用GlobalBroker类，为Map中每一组任务创建一个GlobalBroker，实现任务流动态到达的模拟。

（3）接着通过对DatacenterBroker的简单改进，为其增加一个构造参数。计算每组任务中任务数的最大值，把它当作参数传递给DatacenterBroker，以实现动态调节虚拟机数的目的。

（4）最后接收任务输出列表，打印输出信息及整个过程中的能耗。其中能耗计算是自己定义的计算公式，每台主机能耗计算如下:

HPower = Pmax x texc + Pidle x tidle

其中Pmax代表该主机所能提供的满载功率（实验中的满载功率为上述假定的103.2W）、texc表示在当前时间段内虚拟机执行时间、Pidle代表主机空载功率（实验中空载功率为上述假定的71.6W）、tidle表示在该段时间内主机的空载时间。

### 4.4.3 实验结果分析

对Map中每一组任务，我们采用了两种不同的任务调度方案：一种是按照CloudSim自带的顺序调度模式，即把任务遍历分配到所有的虚拟机；另一种是上面所说的根据任务特征，将任务只分配到指定数目的虚拟机上。图4-6是根据不同的任务流所进行的几次实验结果对比图。横轴表示实验的次数，纵轴表示数据中心的能耗。



图4-6 节能部署调度实验结果对比

如图4-6所示，通过两种不同的任务调度算法的能耗对比图可以明显的看出，为每个时间段的任务选择合适的虚拟机数对节省能耗起到很大的作用，证明该算法在特定的应用场景下具有较实际的意义（因图中每一次产生的任务流都是通过随机函数随机生成的，所以实验结果出现以上波动，但大致结果符合实验预测）。

## 4.5 基于任务与资源的等价类调度算法

### 4.5.1 任务描述

上述节能部署调度算法为了简化问题的复杂性，没有考虑到数据中心的异构性，即主机、虚拟机的配置是完全相同的，而且任务的大小、长度等也都一样。而在现实生活场景中，主机、虚拟机的配置会随着数据中心本身的特点，会存在不同的配置，用样的，任务也会随着不同的用户和不同的提交时间而有不同的变化。本算法在考虑到数据中心异构环境特点的情况下，提出了对任务和虚拟机进行等价类处理，然后再根据虚拟机的执行能力和任务的长度来实现任务和虚拟机的绑定。同时在此调度算法中还综合运用了贪心算法，使执行能力和任务长度相匹配的虚拟机列表和任务列表在调度时达到结果最优化。

### 4.5.2 集群模型模拟

本算法是基于异构环境下的算法，故集群模型与假设与上述不同。模拟实验中我们假设数据中心主机的数量为100，同样虚拟机的数量为100，任务的总数随实验次数动态设置。还假设每台主机上只有唯一的一台虚拟机，因为将一台主机部署成多台虚拟机，这些虚拟机的性能总和比原主机的性能要差，而且多台虚拟机之间的需要额外开销。

为了模拟不同的任务需要调度到特定的虚拟机上执行，我们对Cloudlet类和Vm类进行了简单的改写，为它们添加一个新的字段：系统类型（OS），表明任务需提交到和其OS类型相同的虚拟机上执行，否则会增加额外的时间开销。

以下为主机、虚拟机、任务参数设置：

表4-2 主机参数列表

|  |  |  |  |
| --- | --- | --- | --- |
| **MIPS** | **Ram** | **Storage** | **Bw** |
| **1500** | 2048 | 20000 | 2000 |

主机为容纳虚拟机的地方，同时它也为虚拟机分配相应的资源，如MIPS、容量、带宽等。主机的参数配置应至少满足分配一台虚拟机的要求

表4-3 虚拟机参数列表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **MIPS** | **OS** | **Ram** | **Storage** | **Bw** |
| **530** | Linux | 512 | 10000 | 490 |
| **650** | Windows | 512 | 10000 | 570 |
| **920** | Mac | 512 | 10000 | 860 |

虚拟机的MIPS（执行能力）和Bw（带宽）的范围是400-1000，OS类型考虑到三种情况：Linux、Windows和Mac，Ram和Storage目前设置的为定值。

表4-4 任务参数列表

|  |  |  |  |
| --- | --- | --- | --- |
| **length** | **OS** | **Filesize** | **Outputsize** |
| **33000** | Linux | 320 | 320 |
| **45000** | Windows | 470 | 470 |
| **76000** | Mac | 790 | 790 |

任务长度的范围为20000-80000，OS类型同虚拟机有三种取值，Filesize（输入文件大小）和Outputsize（输出文件大小）的范围为200-800，基本和任务的长度相匹配。

### 4.5.3 仿真实验流程

（1）首先使用随机函数构建虚拟机和任务的参数列表文件，构建时指定虚拟机和任务的数目及一些参数的取值范围。

（2）然后在CloudSim主流程中读取上述虚拟机和任务参数文件，创建虚拟机列表和任务列表并提交到数据中心。DatacenterBroker接收任务和虚拟机列表，准备对任务进行调度。此时我们修改了DatacenterBroker类，为其增加调度方法bindCloudletToVmByOsCapacity()，该方法对接收到的虚拟机列表和任务列表进行等价类的划分，划分后不同类型、不同能力的虚拟机和任务分别存放在以String为Key，List为Value的Map<String, List<Vm或Cloudlet>>中，其中Key代表了不同虚拟机和任务列表的类型，供后续任务的分配作参考。

（3）接着把任务长度较长的任务列表分配给执行能力较强的虚拟机列表，同时还匹配任务和虚拟机的OS特性，此时的分配采用了贪心算法。

（4）最后打印输出所有任务的执行时间总和以及任务的完成时间，和普通的调度算法进行结果的比较，得出结论。

### 4.5.4 实验结果分析

（1）基于第三章介绍的贪心算法，在仿真实验阶段首先对贪心算法在任务调度方面的优势作了对比实验。图4-7展示了贪心算法和CloudSim自带的顺序调度算法在任务总执行时间和任务完成时间上的对比图，实验过程是将40个参数配置不同的任务分配到10个不同的虚拟机上。从五次的实验结果可以看出，贪心算法无论是在任务的总执行时间上，还是任务的完成时间上，都比普通的顺序调度算法有明显的效率提升，证明了贪心算法的可行性，为后面的任务及资源的等价类-贪心调度作铺垫。



图4-7 贪心算法实验对比结果

（2）基于任务与资源的等价类-贪心调度实验。实验中根据任务总数的不同，分别做了五次实验，任务总数分别为2000、2500、3000、3500、4000。采用了三种任务调度模型对任务进行调度，并统计了所有任务完成的总时间之和作为参考指标，以下图标是根据实验中的实际数据绘制的。



图4-8 等价类-贪心调度实验结果对比

图4-8中橙色柱形代表的调度模型是，将任务依次遍历给所有的虚拟机，实现简单的均衡负载，当任务的OS和虚拟机的OS不匹配时，相应的根据比例增加任务的执行时间；绿色柱形表示的是，任务和资源的等价类调度；蓝色的柱形表示的是，将任务和虚拟机进行等价类的划分，然后再综合使用贪心算法。

从图中可以明显的看出，将任务和虚拟机进行等价类划分后，再使用贪心算法进行任务的调度，能明显减少任务执行的总时间。同时通过对不同的调度算法的分析对比，也表明该算法具有实际可行的意义。图中等价类-贪心调度算法相比等价类调度算法在任务总执行时间上并没有很大的差距，是因为在等价类调度过程中已实现了将任务和资源的等价类划分，而实验中每个参数划分的区间不是很大，故每一个等价类内部的资源和任务就没有很大的不同，所以此时使用贪心算法，结果不是特别明显。

## 4.6 本章小结

CloudSim设计的初衷就是用于云计算应用的仿真与模拟，作为一个通用的仿真工具，它允许用户扩展和自定义自己的调度策略。在学习使用CloudSim的过程中，也曾遇到过很多问题，比如仿真过程中动态实体的添加、如何添加自己的调度算法，最终通过自己学习了解的深入，慢慢懂得一些原理。基于此，我们使用它对自己提出的任务调度算法进行了模拟仿真，仿真的结果也大致符合预期的要求，同时也印证了我们算法设计的准确性与合理性。

# 第5章 总结与展望

## 5.1 论文总结

数据中心任务调度是目前研究的热点与重点，在数据成指数型增长的今天，如何对数据中心任务进行合理有效的调度，达到既提高任务的执行效率又降低数据中心能耗的目的，是研究的重点与难点。国内外的一些学者在这方面都做了很多的分析研究，也提出了很多的高效的的调度算法。本文通过在阅读大量文献的基础上，提出了如下的两种调度策略，并使用CloudSim进行模拟仿真实验。做的主要工作如下：

（1）设计了基于动态任务流的节能部署调度方案。该方案通过对实际生活中数据中心任务数量动态到达的模拟，以及根据不同时间段内任务数量不同的特点，将任务划分成不同的时间段，对每个时间段统计该时间段任务数量的最大值，使用该最大值确定需要开启虚拟机的数目。因数据中心一般为满足最大峰值的要求，都会使所有的虚拟机都运行，根据动态任务流确定所需要开启的虚拟机数目，从而关闭不必要的虚拟机达到节能的目的。

（2）设计了基于任务与资源的等价类调度算法。考虑到数据中心资源与任务的异构性，不同的任务在分配给不同的资源时，其执行效率可能不同。鉴于此，我们设计了任务与资源的等价类调度算法，该算法把任务和资源进行等价类划分归类，然后根据归类后的每一组任务特征为其分配匹配的资源组去执行。同时在每一个匹配的任务调度中，使用贪心算法，尽量使结果最优化。

## 5.2 工作展望

在本文的研究中，所提出的算法虽然在实验结果上基本符合预期的设想，基于动态任务流的调度算法使能耗很大程度的降低；基于任务和资源的等价类调度算法使任务总执行时间大幅减少。但因为自己知识水平和能力的有限，一些算法设计在后续还可以作如下方面的改进：

（1）任务和资源等价类调度中，在最后使用贪心算法是针对每个等价类内部的贪心算法，这样在上一步等价类归类后，再使用贪心效果不是很明显。后续可把每个等价类当作一个整体，基于所有等价类使用贪心算法。

（2）目前任务执行时间只与任务长度和虚拟机执行能力有关，后续可考虑虚拟机带宽和任务文件大小等对执行时间的影响。

# 致谢

四年时间一晃而过，毕业论文完成之时也就意味着我的大学四年生活即将踏上句点。

首先，我由衷的感谢我的指导老师袁景凌教授，从毕业设计最初的定题到资料的收集与具体设计，她都给了我耐心的指导和无私的帮助。从袁老师给我们上课时的那一种专注和认真，我就知道她是一个知识渊博的学者，同时对待学生的提问，她总能和蔼耐心的解答。在论文的创作及完成阶段，她真给了我莫大的鼓励与帮助，在这里再次的感谢袁老师，祝她工作顺利、生活幸福美满。

其次，我要感谢在我本科阶段所有教授我知识的老师们，是他们让我走进知识的殿堂，教会我学习及生活的技能。感谢实验室师兄陈旻骋在我论文写作阶段对我的悉心指导以及室友和同学在生活中给予我的帮助与照顾。

最后，我要感谢生我养育我的父母，是他们培育我长大，教我做人的道理。在这一路的学习生涯中始终伴我左右，给我关怀与鼓励。你们永远的健康快乐就是我最大的心愿。

刘东领

2016年5月于武汉

# 参考文献

[1] Krizhevsky, A., Sutskever, I., Hinton, G.: Imagenet classi\_cation with deep convolutional neural networks. In: Advances in Neural Information Processing Systems 25. (2012) 1097{1105

[2] Deghan, A., Masood, S.Z., Shu, G., Ortiz., E.G.: View independent vehicle make, model and color recognition using convolutional neural network. In: arXiv:1702.01721. (2017)

[3] Hong T, Gopalakrishnam A K. License plate extraction and recognition of a Thai vehicle based on MSER and BPNN[C]//Knowledge and Smart Technology (KST), 2015 7th International Conference on. IEEE, 2015: 48-53.

[3] Illikkal R,Tickoo O，Zhao L，et a1．VM3：Measudng，Modeling and Managing VM Shared Resources[J]. Computer Networks:The International Journal of Computer and Telecommunications Networking，2009，53(17)．

[4] Freeman T，Keahey K．Flying low: Simple leases with workspace pilot[C]. Euro-Par 2008-Parallel Processing．Springer Berlin Heidelberg, 2008：499-509．

[5]Sotomayor B，Keahey K，Foster I，et a1．Enabling cost-effective resource leases with virtual machines[C]．Hot Topics session in ACM/IEEE International Symposium on High Performance Distributed Computing，2007.

[6]李建锋，彭舰．云计算环境下基于改进遗传算法的任务调度算法[J]．计算机应用，2011，31(1)：184-186.

[7]Kong X，Lin C，Jiang Y et a1．Efficient dynamic task scheduling in virtualized data centers with fuzzy prediction[J]．Journal of network and Computer Applications，2011，34(4)：1068-1077．

[8] Song Y, Wang H, Li Y, et al. Multi-Tiered On-Demand Resource Scheduling for VM-Based Data Center[C]// 9th IEEE/ACM International Symposium on Cluster Computing and the Grid, CCGrid 2009, Shanghai, China, 18-21 May 2009. 2009:148-155.

[9]Chauhan SS，Joshi R C．A heuristic for QoS based independent task scheduling in grid environment[C]．Industrial and Information Systems (ICIIS) , 2010 International Conference on．IEEE，2010：102-106．

[10]Xu M，Cui L，Wang H，et a1．A multiple QoS constrained scheduling strategy of multiple workflows for cloud computing[C]. Parallel and Distributed Processing with Applications，2009 IEEE International Symposium on．IEEE，2009：629-634．

[11]孙大为，常桂然，李风云等．一种基于免疫克隆的偏好多维QoS云资源调度优化算法[J]．电子学报，2011，39(8)：1824-1831．

[12]Buyya R Yeo CS, Venugopal S,et a1．Cloud computing and emerging IT platforms：Vision, hype, and reality for delivering computing as the 5th utility[J].Future Generation computer systems，2009，25(6)：599-616．

[13] Buyya R, Yeo C S, Venugopal S. Market-oriented cloud computing: Vision, hype, and reality for delivering it services as computing utilities [C]//High Performance Computing and Communications, 2008. HPCC'08. 10th IEEE International Conference on. Ieee, 2008: 5-13.

[14]葛新，陈华平，杜冰等．基于云计算集群扩展中的调度策略研究[J]．计算机应用研究，2011，28(3)：995-997．

[15]尹红军，李京，宋浒等．云计算中运营商效益最优的资源分配机制[J]．华中科技大学学报(自然科学版)，2011，1.

[16] 田文洪 赵勇. 数据中心资源优化调度--理论与实践[M]. 电子工业出版社, 2014.

[17] 郭凤羽, 禹龙, 田生伟,等. 云计算环境下对资源聚类的工作流任务调度算法[J]. 计算机应用, 2013, 33(8): 2154-2157.

[18] 徐章艳, 刘作鹏, 杨炳儒,等. 一个复杂度为max(O(|C||U|),O(|C^2|U/C|))的快速属性约简算法[J]. 计算机学报, 2006, 29(3): 391-399.

[19] Etminani K, Naghibzadeh M. A min-min max-min selective algorihtm for grid task scheduling[C]//Internet, 2007. ICI 2007. 3rd IEEE/IFIP International Conference in Central Asia on. IEEE, 2007: 1-7.

[20] Moreno R, Alonso-Conde A B. Job scheduling and resource management techniques in economic grid environments[C]//Grid Computing. Springer Berlin Heidelberg, 2004: 25-32.

[21] 刘鹏. 云计算丨 M][J]. 电子工业出版社 2011.

[22] Calheiros R N, Ranjan R, Beloglazov A, et al. CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms[J]. Software: Practice and Experience, 2011, 41(1): 23-50.