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](#_Toc483053976)

[Abstract II](#_Toc483053977)

[目 录 III](#_Toc483053978)

[第1章 绪论 1](#_Toc483053979)

[1.1 研究背景及意义 1](#_Toc483053980)

[1.2 国内外研究现状 1](#_Toc483053981)

[1.3 研究路线及工作内容 2](#_Toc483053982)

[1.4 论文主要结构 3](#_Toc483053983)

[第2章 卷积神经网络基础知识 4](#_Toc483053984)

[2.1 深度学习背景 4](#_Toc483053985)

[2.2 人工神经网络 5](#_Toc483053986)

[2.3 梯度下降和反向传播 6](#_Toc483053987)

[2.3.1 梯度下降算法简介 6](#_Toc483053988)

[2.3.2 反向传播算法 7](#_Toc483053989)

[2.4 卷积神经网络 8](#_Toc483053990)

[2.4.1 卷积神经网络结构 8](#_Toc483053991)

[2.4.2 常用CNN网络介绍 10](#_Toc483053992)

[2.5 本章小结 10](#_Toc483053993)

[第3章 车牌定位算法 11](#_Toc483053994)

[3.1 常见的车牌定位算法 11](#_Toc483053995)

[3.1.1 基于边缘检测的车牌定位算法 11](#_Toc483053996)

[3.1.2 基于彩色分割的车牌定位算法 11](#_Toc483053997)

[3.1.3基于小波变换的车牌定位方法 12](#_Toc483053998)

[3.1.4基于遗传算法的车牌定位 12](#_Toc483053999)

[3.1.5基于数学形态学的车牌定位方法 13](#_Toc483054000)

[3.1.6基于灰度图像纹理特征分析的车牌定位方法 13](#_Toc483054001)

[3.2 MSER算法 14](#_Toc483054002)

[3.2.1 数据中心任务调度的形式化描述 14](#_Toc483054003)

[3.2.2 传统的任务调度算法 14](#_Toc483054004)

[3.2.3 基于数据中心任务调度的贪心算法 15](#_Toc483054005)

[3.3 基于等价类的贪心策略部署 16](#_Toc483054006)

[3.4 本章小结 16](#_Toc483054007)

[第4章 CloudSim仿真实验 17](#_Toc483054008)

[4.1 CloudSim简介 17](#_Toc483054009)

[4.1.1 CloudSim仿真框架特性 17](#_Toc483054010)

[4.1.2 CloudSim体系架构 17](#_Toc483054011)

[4.2 CloudSim技术实现 19](#_Toc483054012)

[4.2.1 CloudSim类图及核心类 19](#_Toc483054013)

[4.2.2 任务处理过程 20](#_Toc483054014)

[4.3 CloudSim模拟仿真步骤 21](#_Toc483054015)

[4.4 基于动态任务流的节能部署方案 21](#_Toc483054016)

[4.4.1 仿真方案场景 21](#_Toc483054017)

[4.4.2 仿真方案流程 22](#_Toc483054018)

[4.4.3 实验结果分析 23](#_Toc483054019)

[4.5 基于任务与资源的等价类调度算法 23](#_Toc483054020)

[4.5.1 任务描述 23](#_Toc483054021)

[4.5.2 集群模型模拟 24](#_Toc483054022)

[4.5.3 仿真实验流程 25](#_Toc483054023)

[4.5.4 实验结果分析 25](#_Toc483054024)

[4.6 本章小结 27](#_Toc483054025)

[第5章 总结与展望 28](#_Toc483054026)

[5.1 论文总结 28](#_Toc483054027)

[5.2 工作展望 28](#_Toc483054028)

[致谢 29](#_Toc483054029)

[参考文献 30](#_Toc483054030)

# 

# 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。

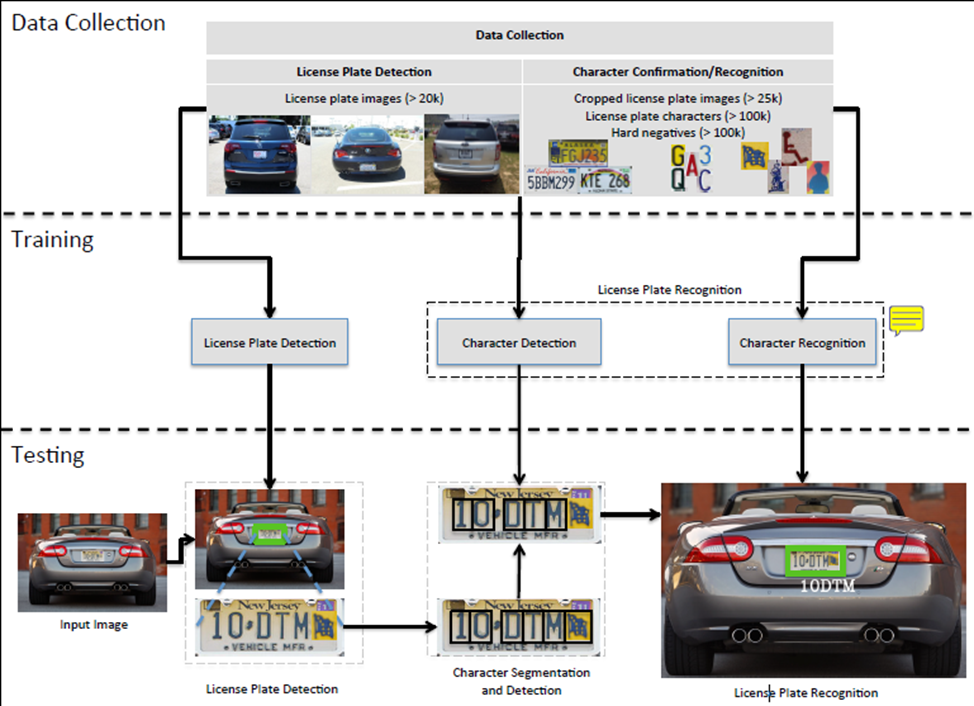
车牌识别技术要求能够将运动中的汽车牌照从复杂背景中提取并识别出来，通过车牌提取、图像预处理、特征提取、车牌字符识别等技术，识别车辆牌号、颜色等信息，目前最新的技术水平为字母和数字的识别率可达到99.7%，汉字的识别率可达到99%。车牌识别在[高速公路](http://baike.baidu.com/item/%E9%AB%98%E9%80%9F%E5%85%AC%E8%B7%AF" \t "_blank)车辆管理中得到广泛应用，电子收费（ETC）系统中，也是结合DSRC技术识别车辆身份的主要手段。

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" \t "_blank)、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" \t "_blank)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）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，车牌定位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，车牌定位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。本章从常用的车牌定位算法出发，进一步介绍了本文中要使用的MSER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。



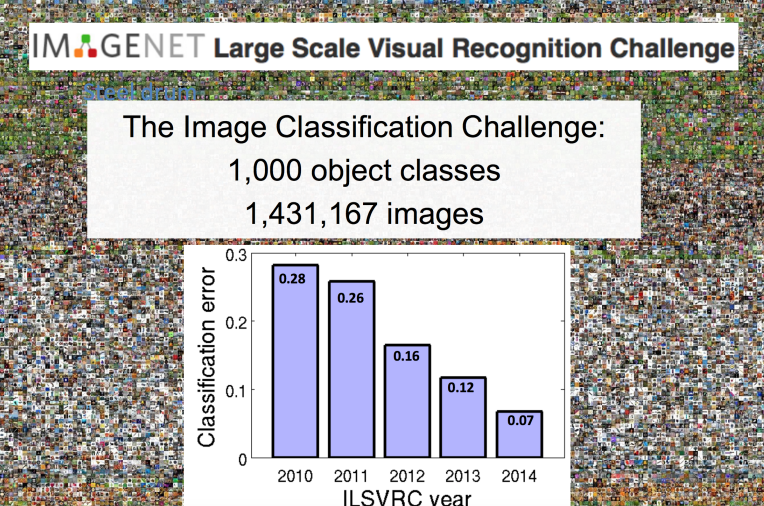
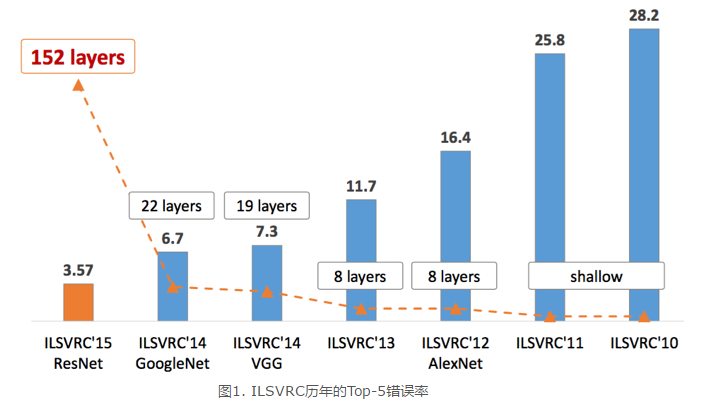
图1.2 论文主要组织结构

# 第2章 卷积神经网络基础知识

## 2.1 深度学习背景

其实CNN并不是近几年才出现的。早在1985年，Rumelhart和Hinton等人就提出了反向传播（Back Propagation，BP）算法[4]，使得神经网络的训练变得简单可行。

几年后，LeCun利用BP算法来训练多层神经网络用于识别手写邮政编码[5]，这个工作就是CNN的开山之作，如图2所示，多处用到了5\*5的卷积核，但在这篇文章中LeCun只是说把5\*5的相邻区域作为感受区域，并未提及卷积或卷积神经网络。

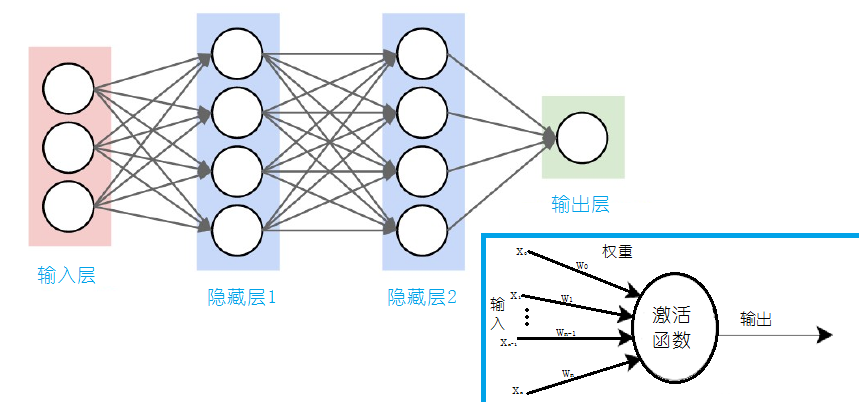
 1998年的LeNet5[6]标注着CNN的真正面世，但是这个模型在后来的一段时间并未能得到学术界的重视，主要原因是训练的计算量巨大，而且其他算法（如SVM）也能达到类似的效果甚至超过。随着计算机计算能力的提升尤其是GPU并行计算的出现和流行，使得更深的网络成为可能，使深层的神经网络的学习能力得以彰显出来。

ImageNet大规模视觉识别大赛(ImageNet Large Scale Visual Recognition Competition, ILSVRC)是全球最为权威的计算机视觉比赛，该赛事的目标是对1000类图像进行分类，比赛的训练数据集数据达126万张图像，验证集5万张，测试集有10万张。2012年，Hinton的学生Alex Krizhevsky一举摘下了ILSVRC 2012冠军，效果大幅度超过传统的SVM方法，使TOP5的错误率首次降到20%以下。这个深度学习模型就是后来大名鼎鼎的AlexNet模型。从此，深度学习一发不可收拾，ILSVRC每年都不断被深度学习也就是CNN刷榜，如图XXX所示，随着模型变得越来越深，Top-5的错误率也越来越低，2015年降到了3.5%附近，而在同样的ImageNet数据集合上，人眼对其分类的辨识错误率大概在5.1%，也就是目前的CNN对图像分类的能力已经超过了人眼。

## 2.2 人工神经网络

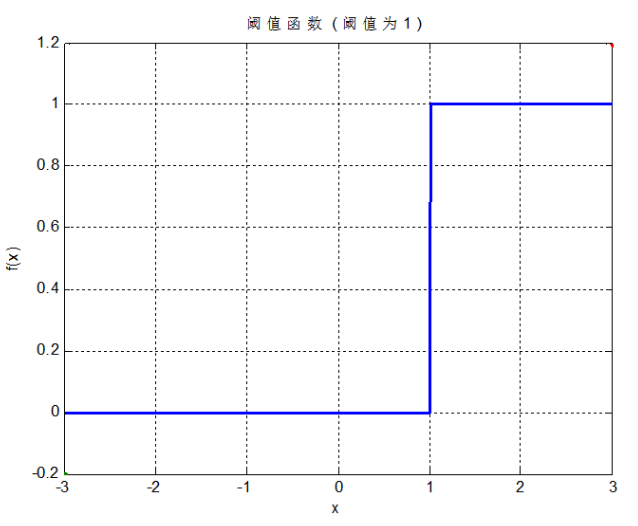
人工神经网络由多个神经元模型构成（如图XXX），这种由许多神经元组成的网络具有可并行计算的分布结构。每个神经元具有单一输出，并且能够与其它神经元连接。每条连线上都有一个权重，代表上一个神经元对下一个神经元的影响程度。令来自其它处理单元(神经元)的信息为，它们与本处理单元的互相作用强度为，。那么本神经元的输入为,而神经元的输出为：。

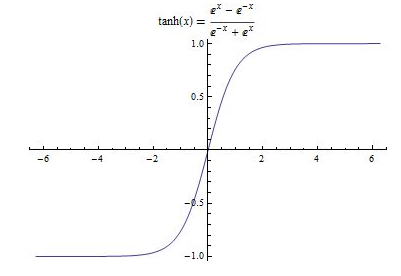
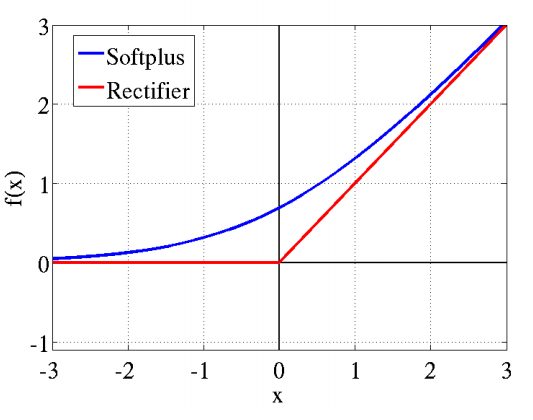
式中，为第个元素的输入，为第个元素与本处理单元的互联权重。称为激发函数(activation function)。它决定节点(神经元)的输出，提供了神经网络的非线性特性。



常用的激活函数有阈值函数、sigmoid函数、ReLU（改进版Leaky ReLU）、双曲正切函数，这些函数都有两个共同的特点：（1）其函数值都在某个处发生急剧的变化；（2）其导数都能用函数值容易的表示出来。

1. sigmoid函数
2. 阈值函数
3. ReLU
4. Leaky ReLU
5. 双曲正切函数



(a)sigmoid函数 （b）阈值函数

(c)双曲正切函数 (d)ReLu函数

## 2.3 梯度下降和反向传播

### 2.3.1 梯度下降算法简介

顾名思义，梯度下降法的计算过程就是沿梯度下降的方向求解极小值（也可以沿梯度上升方向求解极大值）。

其迭代公式为http://h.hiphotos.baidu.com/baike/s%3D118/sign=65c70ae8778da9774a2f822a8850f872/63d0f703918fa0ec27adadd6229759ee3c6ddb87.jpg,其中http://g.hiphotos.baidu.com/baike/s%3D22/sign=d2fd6faed288d43ff4a996f07f1e7b9a/63d0f703918fa0ec2604acd6229759ee3c6ddb2e.jpg代表梯度负方向，http://g.hiphotos.baidu.com/baike/s%3D15/sign=617e99e2cd3d70cf48faae08f9dc6b5d/a8014c086e061d95143bfce37ff40ad163d9cae8.jpg表示梯度方向上的搜索步长。梯度方向我们可以通过对函数求导得到，步长的确定比较麻烦，太大了的话可能会发散，太小收敛速度又太慢。一般确定步长的方法是由线性搜索算法来确定，即把下一个点的坐标看做是ak+1的函数，然后求满足f(ak+1)的最小值的 即可。

因为一般情况下，梯度向量为0的话说明是到了一个极值点，此时梯度的幅值也为0.而采用梯度下降算法进行最优化求解时，算法迭代的终止条件是梯度向量的幅值接近0即可，可以设置个非常小的常数阈值。

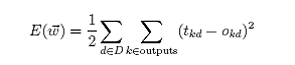
### 2.3.2 反向传播算法

人工神经网络为学习实数值和向量值函数提供了一种实际可行的方案，连续的和离散的属性都可以使用，并且对训练数据中的噪声具有很好的健壮性。反向传播算法是网络学习算法中最常见的算法。反向传播算法是一种具有很强学习能力的系统，结构比较简单，且易于编程。

鲁梅尔哈特(Rumelhart)和麦克莱兰(Meclelland)于 1985 年发明并发展了 BP 网络学习算法，实现了明斯基的多层网络设想。BP网络不仅含有输入节点和输出节点，而且含有一层或多层隐(层)节点。输入信号先向前传递到隐藏节点，经过作用后，再把隐藏节点的输出信息传递到输出节点，最后给出输出结果。节点的激发函数选用Sigmoid函数。

反向传播(back－propagation，BP)算法是一种估计单个权重变化引起整个网络性能变化值的较为简单的方法。由于BP算法过程包含从输出节点开始，反向地向第一隐藏层(即最接近输入层的隐藏层)传播由总误差引起的权重修正，所以称为"反向传播"。反向传播特性与所求解问题的性质和所作细节选择有极为密切的关系。

对于由一系列确定的单元互连形成的多层网络，反向传播算法可用来学习这个多层网络的权值。它采用梯度下降方法试图最小化网络输出值和目标值之间的误差平方，因为我们要考虑多个输出单元的网络，而不是像以前只考虑单个单元，所以我们要重新计算误差E，以便对所有网络输出的误差求和:



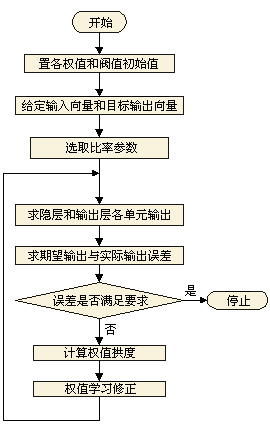
其中，Outpus 是网络输出单元的集合， 和 是与训练样例和第个输出单元的相关输出值。

反向传播算法的一个迷人特性是：它能够在网络内部的隐藏层发现有用的中间表示:

1.训练样例仅包含网络输入和输出，权值调节的过程可以自由地设置权值，来定义任何隐藏单元表示，这些隐藏单元表示在使误差E达到最小时最有效。

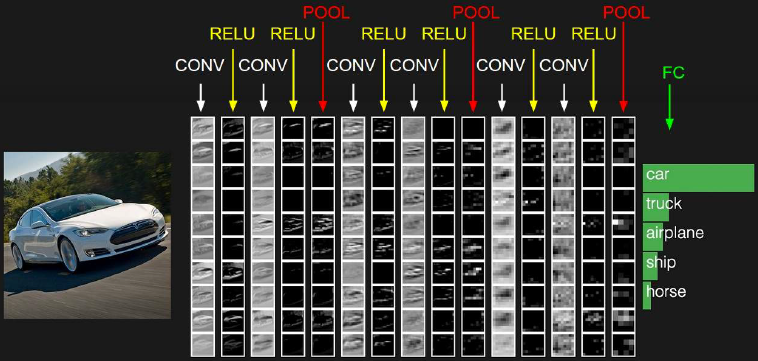
2.引导反向传播算法定义新的隐藏层特征，这些特征在输入中没有明确表示出来，但能捕捉输入实例中与学习目标函数最相关的特征

反向传播训练神经元的算法如下：



## 2.4 卷积神经网络

### 2.4.1 卷积神经网络结构

 （1）数据输入层（Input Layer）

数据输入层的工作是对图片进行预处理，一般处于网络的最前端。常见的预处理方法有去均值、归一化、PCA白化。

* 取均值：把输入数据各个维度都中性化到0。
* 归一化：把输入数据的幅度规约到同一范围。
* PCA：用主成分分析降维。
* 白化：对数据每个特征轴上进行归一化。

（2）卷积计算层（CONV Layer）

对于一副图像来说，其实后一层的每个节点没必要和上一层的所有节点相连，如果后一层一个单元和前一层所有连接，权重参数会异常的多。而我们的图像在视觉机制上表现为：图像上的某一部分的像素所表示的意义其实只和它周围的像素领域有关，这也是图像的空间特性。那么我们何不将上面每个节点连接范围缩小呢？这也叫做是局部感知。

当然CNN减少参数的方式可不止这些，CNN减少参数的另一个核心方法是参数共享。所谓卷积，其实就是一种滤波方式。这在信号处理、图像处理上经常用到。最简单的在图像里面，一副图像和一个给定图像或者模版的进行卷积，卷积就是对应位置数据相乘然后再相加，如公式XXX。

其中，为原图像，为卷积核。

这种卷积运算实质上是将全连接的参数进行了共享。每个神经元连接数据窗的权重是固定的；每个神经元只关注某一种特性。这样便大大减少了网络参数，例如经典网络AlexNet如果采用全连接参数将会有一亿多，而实际数字是3.5万。

（3）ReLU激励层（ReLU Layer）

和人工神经网络的激励函数一样，是将卷积层的输出结果利用激励函数做非线性的映射。

（4）池化层（pooling Layer）

池化层一般夹在连续的卷积层中间，对数据进行下采样，用于压缩数据和参数的量，并减小过拟合。常用的池化方法有Max Pooling和Average pooling。

（5）全连接层（FC Layer）

全连接层就是普通人工神经网络的层级结构，即两层之间所有神经元都有权重连接。通常处于神经网络的尾部，由于其参数过于巨大，近几年的CNN多采用的小卷积核所构成的卷积层作为最后一层。

1. BN层（Batch Normalization Layer）

BN层是对进行过多次处理后的数据进行处理使得数据重新符合正态分布且将数据归一化。由于本篇论文没有涉及此层，将不多赘述。

一般CNN的层次结构依次为：

* 输入层
* [[卷积层->激励层]\*N->池化层?]\*M
* [全连接层->激励层]\*K
* 全连接层或小卷积层

### 2.4.2 常用CNN网络介绍

* LeNet，最早用于数字识别的CNN。
* AlexNet，ILSVRC2012远远超过第二名的CNN，比LeNet更深，用多层小卷积层叠加替换掉大卷积层。
* ZF Net，ILSVRC2013比赛冠军。
* GoogleNet，ILSVRC2014比赛冠军。
* VGGNet，ILSVRC2014比赛中用到的模型，图像识别率略差于GoogleNet，但是在很多图像迁移学习问题（比如目标检测）上效果很好。
* ResNet，ILSVRC2014比赛冠军，采用了残差学习以适应更深层次的CNN训练。从此CNN的层数与日俱增。

## 2.5 本章小结

本章首先介绍了深度学习的背景知识，然后介绍了人工神经网络的基本概念，同时在对人工神经网络的理解的基础上，介绍了求解神经网络的一般性方法。介绍了反向传播算法和梯度下降算法。最后进一步介绍了卷积神经网络和常见的CNN结构。

# 第3章 车牌定位算法

## 3.1 常见的车牌定位算法

根据车牌的不同特征，可以采用不同的定位方法。目前车牌定位的方法很多，最常见的定位技术主要有基于边缘检测的方法、基于彩色分割的方法、基于小波变换的方法、基于遗传算法的方法、基于数学形态学的车牌定位和基于灰度图像纹理特征分析的方法等，在此对几种常用的定位算法进行简单的介绍。

### 3.1.1 基于边缘检测的车牌定位算法

边缘是指其周围像素灰度有阶跃变化的那些像素的集合。边缘的两侧分属于两个区域，每个区域的灰度均匀一致，而这两个区域的灰度在特征上存在一定的差异。边缘检测的任务是精确地定位边缘和抑制噪声。检测的方法有多种, 例如Roberts 边缘算子、Prewitt 算子、Sobel 算子以及拉普拉斯边缘检测。这些方法正是利用边缘处灰度变化剧烈这一特点来检测图像的边缘。各算子对不同边缘类型的敏感程度不同, 产生的效果也不同, 经过大量实验分析可知, Roberts边缘算子是一种利用局部方差算子寻找边缘的算子, 定位比较精确; Prewitt算子和Sobel算子对噪声有一定的抑制能力, 但不能完全排除伪边缘; 拉普拉斯算子是二阶微分算子, 对图像中的阶跃型边缘点定位准确且具有旋转不变性, 但容易丢失一部分边缘的方向信息, 同时抗噪能力较差。针对不同的环境和要求, 选择合适的算子来对图像进行边缘检测才能达到好的效果。具体定位流程如图XXX所示。

原始图像

图像预处理

边缘算子及其二值

车牌定位

图XXX 基于边缘检测的车牌定位流程

    该方法的定位准确率较高、反应时间快、能有效去掉噪声，适合于包含多个车牌的图像, 在多车牌图像的情况下定位速度也很快。但是对车牌严重褪色的情况，由于检测不到字符笔画的边缘会导致定位失败，在有外界干扰以及车牌倾斜时，定位后的区域比车牌稍大。

### 3.1.2 基于彩色分割的车牌定位算法

基于彩色分割的车牌定位算法由彩色分割和目标定位等模块组成，采用多层感知器网络对彩色图像进行分割，然后通过投影法分割出潜在的车牌区域。在进行彩色分割时采用神经网络模型，一般图像采用RGB三原色，但由于RGB三原色中两点的欧氏距离不成线性比例。为了更好地进行彩色分割，一般将RGB模式的彩色图像转化为HSI模式，即色调、饱和度和亮度，然后对输出图像的饱和度作调整。为了减少计算量，将彩色图像下采样后再进行模式转化。同时，为了减少光照条件对图像分割产生的影响，采用对数方法进行彩色饱和度调整。然后对模式转化后的彩色图像进行彩色神经网络分割，最后根据车牌底色及长宽比等先验知识，采用投影法分割出合理的车牌区域。具体定位流程如图XXX所示。

原始图像

抽稀

将RGB图像转成HIS模式

饱和度调整

颜色分割

饱和度分割

车牌定位

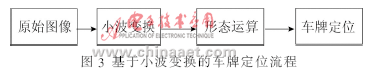
图XXX 基于彩色分割的车牌定位算法流程

该定位算法正确率较高，但由于采用了神经网络计算法，当区域颜色与附近颜色相似时，计算速度较慢。当获取的彩色图像质量较高时，尤其是车牌区域颜色与附近颜色差别较大时，准确率将有所下降。

### **3.1.3 基于小波变换的车牌定位方法**

 小波分析是一种应用于图像处理的重要分析工具, 具有“显微镜”的特性。小波分析的多分辨率特性使得小波分解系数在不同方向的高频子波系数具有不同特性, 因此利用方向小波能够反映出图像在不同分辨率上沿任一方向变化的情形。小波分析的多尺度分解特性更加符合人类的视觉机制。

小波变换的基本思想是将原始信号经过伸缩、平移等运算分解为一系列具有不同空间分辨率、不同频率特性和方向特性的子带信号, 这些子带信号具有良好的时频特性, 通过利用这些特性可以实现对信号的时域、频域的局部分析。具体定位流程如图XXX所示。



目前利用小波分析的车牌定位算法大多是利用小波变换与其他多种方法相结合来实现更准确、快速的定位。例如基于小波分析和数学形态学的车牌定位方法，该方法通过小波多尺度分解提取出纹理清晰且具有不同空间分辨率、不同方向的边缘子图，然后利用车牌目标区域具有水平方向低频、垂直方向高频的的特点实现子图提取，最后用数学形态学方法对小波分解后的细节图像进行一系列的形态运算，进一步消除无用信息和噪声，以确定车牌位置。该方法在噪声较小的情况下定位效果好，分割精度高；其缺点是速度较慢，且在噪声较大时误定位机率也随之增大。

### 3.1.4 基于遗传算法的车牌定位

    基于遣传算法的车牌定位方法利用遗传算法对图像进行优化搜索，结合区域特征矢量构造适应度函数，最终寻找车牌区域的最佳定位参量。车牌定位是寻找一个符合“车牌区域特征”最佳区域的过程，本质上就是从参量空间寻找最优定位参量的问题，而寻找参量空间的最优解正是遗传算法所擅长的。但是在实时系统中，车牌定位速度受遗传算法中迭代次数的影响很大。 具体定位流程如图XXX所示。



### 3.1.5 基于数学形态学的车牌定位方法

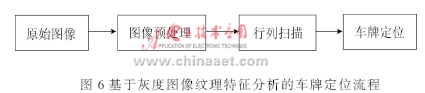
数学形态学图像处理的基本思想是利用一个结构元素来探测一个图像, 看是否能将这个结构元素很好地放在图像内部, 同时验证填放元素的方法是否有效。腐蚀、膨胀、开启和关闭是数学形态学的基本运算。具体定位流程如图XXX所示。



基于数学形态学的车牌区域定位方法不能精确确定车牌左右边界的位置, 所以必须结合其他定位方法进行精确定位。例如基于数学形态学和边缘特征的车牌定位方法, 这种方法先对车牌图像进行预处理, 然后基于垂直方向结构元素的腐蚀运算进行滤波, 再用闭合运算来填补车牌区域内细小孔洞, 进而增强车牌区, 使车牌区成为一个连通区域, 最后利用字符边缘的特征对车牌进行准确的定位。该方法将数学形态学运算与数字图像的特征相结合, 有效改进了传统的车牌定位方法, 提高了车牌定位的速度和准确度。

### 3.1.6 基于灰度图像纹理特征分析的车牌定位方法

传统的纹理特征分析定位算法大多基于灰度图像来分析的, 因此该算法需要对图像进行预处理, 将彩色图像转换为灰度图像，然后进行行扫描, 找出图像中每一行所含有的车牌线段, 记录下它们的起始坐标和长度，如果有连续若干行均存在不少于一个的车牌线段, 且行数大于某一确定的阈值, 则认为在行的方向上找到了车牌一个候选区域, 并确定了该候选区域的起始行和高度；在已找到的可能存在车牌的区域做列扫描, 以确定该车牌候选区域的起始行和高度以及起始列坐标和长度，由此确定一个车牌区域；继续在其他可能存在车牌的区域寻找, 直至找到所有的车牌候选区域。



该算法对于牌照倾斜或变形以及光照不均、偏弱或偏强有很好的效果, 但对噪声敏感, 对于背景复杂的图像可以结合垂直投影的方法来得到真正的车牌区域,可以有效地解决背景复杂的车牌定位。

车牌定位技术是车牌识别系统中的一个重要环节,在定位的精度、计算速度和适用的可靠性方面还需要进一步改进和提高。目前，还没有一种算法能够实现对于任意背景、位置和光照条件下的汽车图像进行车牌定位。

## 3.2 MSER算法

最稳定极值区域（maximally stable extremal region，MSER）[7]。下面是MSER提取过程：

1.使用一系列灰度阈值对图像进行二值化处理，得到一系列二值图像；

2.对于每个阈值得到的二值图像，得到相应的黑色区域与白色区域；

3.在比较宽的灰度阈值范围内保持形状稳定的区域就是MSER，判断标准为；

David Nister等人于2008年在对J.Matas提出的MSER[算法](http://lib.csdn.net/base/datastructure)进行了改进[8]，使其在时间复杂度方面比原算法要小得多，算法思路如下：

1. 像素点排序→②极值区域生成→③稳定区域判定→④区域拟合→⑤区域归一化



图XXX 原始图像



图XXX MSER分步操作结果

如图XXX，先使用MSER算法提取出图像的区域，然后对这些区域进行形态学闭操作，使其成较为完整的区域，再对这些区域求外接矩形，根据外接矩形的参数去判断是否是车牌区域。

## 3.3 本章小结

本章分别介绍了基于边缘检测//彩色分割//小波变换//遗传算法//数学形态学//图像纹理特征分析地车牌定位方法，进而介绍了本文采用的MSER算法。在介绍这些算法时，着重介绍了这些算法的适应情况及面临的问题。

# 第4章 字符切割算法

## 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：

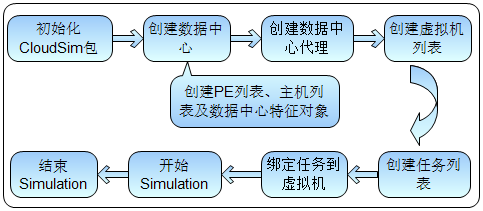


图4-5 仿真流程图

## 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。

康康

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.

[4] DE Rumelhart, GE Hinton, RJ Williams, Learning internal representations by error propagation. 1985 – DTIC Document.  
[5] Y. LeCun , B. Boser , J. S. Denker , D. Henderson , R. E. Howard , W. Hubbard and L. D. Jackel, “Backpropagation applied to handwritten zip code recognition”, Neural Computation, vol. 1, no. 4, pp. 541-551, 1989.

[6] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11):2278–2324, 1998.

[7] Matas J, Chum O, Urban M, et al. Robust wide-baseline stereo from maximally stable extremal regions[J]. Image and vision computing, 2004, 22(10): 761-767.

[8] Nistér D, Stewénius H. Linear time maximally stable extremal regions[J]. Computer Vision–ECCV 2008, 2008: 183-196.

[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.