浙江工业大学

硕士学位论文

论文题目：面向云制造多目标优化资源调度结果的预测方法研究

作者姓名

指导教师

第二导师

学科专业　　软件工程

学位类型　　工程硕士

培养类别　　全日制专业学位硕士

所在学院　　计算机科学与技术学院

提交日期：2019 年 06 月

Research on Prediction Method for Multi-objective Optimization Resource Scheduling Results for Cloud Manufacturing

Dissertation Submitted to

Zhejiang University of Technology

in partial fulfillment of the requirement

for the degree of

Master of Engineering

by

Dissertation Supervisor: Prof.

Associate Supervisor: Associate Prof.

June., 2019

浙江工业大学学位论文原创性声明

本人郑重声明：所提交的学位论文是本人在导师的指导下，独立进行研究工作所取得的研究成果。除文中已经加以标注引用的内容外，本论文不包含其他个人或集体已经发表或撰写过的研究成果，也不含为获得浙江工业大学或其它教育机构的学位证书而使用过的材料。对本文的研究作出重要贡献的个人和集体，均已在文中以明确方式标明。本人承担本声明的法律责任。

作者签名：　　日期：　　年　　月

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留、使用学位论文的规定，同意学校保留并向国家有关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。本人授权浙江工业大学可以将本学位论文的全部或部分内容编入有关数据库进行检索，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□，在一年解密后适用本授权书。

2、保密□，在二年解密后适用本授权书。

3、保密□，在三年解密后适用本授权书。

4、不保密□。

（请在以上相应方框内打“√”）

作者签名：　　日期：　　年　　月

导师签名：　　日期：　　年　　月

中图分类号　TP391　　学校代码　10337

UDC　004　　密级　公开

研究生类别　全日制专业型硕士研究生

浙江工业大学

工程硕士学位论文

面向云制造多目标优化资源调度结果的预测方法研究

Research on Prediction Method for Multi-objective Optimization Resource Scheduling Results for Cloud Manufacturing

作　　者　　第一导师　　教　授

申请学位　工程硕士　　第二导师　　副教授

学科专业　软件工程　　培养单位　计算机科学与技术学院

研究方向　云制造　　答辩委员会主席　***

答辩日期：　2019　年　　月　　日

面向云制造多目标优化资源调度结果的预测方法研究

摘　要

近些年来随着制造业的转型，人性化、智能化的制造方式越来越受到人们的关注。云制造的概念顺应时代的发展被提出，云制造是在制造即服务理念的基础上融合了云计算、大数据和物联网等技术发展起来的。云制造的任务调度和资源分配一直是当前云制造研究的热点。与侧重计算能力的云计算资源调度不同，云制造资源调度是根据云平台中虚拟制造资源的状态信息和制造任务的实时信息来产生最优的任务执行方案。随着大数据和物联网的发展，如何在最短时间内寻找最好结果是云制造资源调度分配的核心问题，利用深度学习模型训练优化调度后的数据来实现对调度结果的直接预测是一条可行的路径。本文主要从调度算法的优化和深度学习模型的改进两个方面展开，具体如下：

1. 针对云制造资源调度的问题，建立基于 QoS 约束条件下的总任务执行时间最短的调度模型。

2. 在云制造任务调度模型基础上，提出了一种改进的新蝙蝠算法实现对云制造多目标优化资源调度问题的求解，改进的新蝙蝠算法在原有的算法基础上加入二阶振荡环节并融入差分进化算法，改进后算法的速度和精度都有明显提升。

3. 在上述基础上，针对调度效率的优化问题，引入了深度学习模型，并对深度学习模型的学习率进行了改进，通过对学习率的改进达到优化模型训练速度的目的从而加快模型的训练速度。

4. 最后对论文先对改进的新蝙蝠算法进行仿真，使用改进的算法对提出的云制造调度模型进行求解并与其他算法进行比较，然后利用获得的调度数据集训练改进的深度学习模型，之后使用训练好的模型对调度结果进行预测并和传统方法的调度时间进行比较，从而验证本文提出的方法。

关键词：云制造，调度模型，新蝙蝠算法，深度学习，调度结果预测

Research on Prediction Method for Multi-objective Optimization Resource Scheduling Results for Cloud Manufacturing

ABSTRACT

Manufacturing industry is the material basis and main industry of the national economy. It is an important symbol of the country's scientific and technological level and comprehensive strength. In recent years, with the transformation of manufacturing industry, humanized and intelligent manufacturing methods have attracted more and more attention. The concept of cloud manufacturing is proposed in response to the development of the times. Cloud manufacturing is based on the concept of manufacturing-as-a-service, which combines technologies such as cloud computing, big data and Internet of Things. Task scheduling and resource allocation for cloud manufacturing has always been a hot topic in cloud manufacturing research. Different from cloud computing resource scheduling, which focuses on computing power, cloud manufacturing resource scheduling is based on state information of virtual manufacturing resources in the cloud platform and real-time information of manufacturing tasks to generate an optimal task execution plan. With the development of big data and Internet of Things, how to find the optimal result in the shortest time is the core problem of cloud manufacturing resource scheduling and distribution. It is a feasible path to use the deep learning model to train the optimized data to achieve direct prediction of the scheduling results. This paper mainly focuses on the optimization of scheduling algorithm and the improvement of deep learning model, as follows:

1. Aiming at the problem of cloud manufacturing resource scheduling, the scheduling model with the shortest total task execution time based on QoS constraints is established.

2. Based on the cloud manufacturing task scheduling model, an improved new bat algorithm is proposed to solve the multi-objective optimization resource scheduling problem of cloud manufacturing. The improved new bat algorithm adds a second-order oscillating link and integrates the difference based on the original algorithm, the speed and accuracy of the improved algorithm have been significantly improved.

3. On the basis of the above, the deep learning model is introduced for the optimization of scheduling efficiency, and the learning rate of the deep learning model is improved. The training speed of the model is accelerated by improving the learning rate to speed up the training of the model. Then use the scheduling data set to train the improved deep learning model to predict the scheduling results.

4. Finally, the paper firstly simulates the improved new bat algorithm, uses the improved algorithm to solve the proposed cloud manufacturing scheduling model and compares it with other algorithms, then uses the obtained scheduling data set to train the improved deep learning model, and then use the training. A good model predicts the scheduling results and compares them with the scheduling time of the traditional methods, so that the proposed method is practical.

KEY WORDS: Cloud Manufacturing, Scheduling Model, New Bat Algorithms, Deep Learning, Scheduling Results Prediction

目　录

插图清单

附表清单

第一章 绪论

1.1 研究背景与意义

人类社会在 18 世纪蒸汽机的出现迎来了第一次工业革命，人类通过蒸汽机实现了工厂机械化，此次工业革命使得机械生产代替了手工劳动，人类从农业、手工业的经济社会转型为以工业和机械制造业为主的新型经济模式，这次工业革命被称为工业 1.0 时代；工业 2.0 时代是电气化和自动化的时代，批量生产产品的高效新模式在工业 2.0 时代首次被广泛采用；从 20 世纪 70 年代开始，人类社会进入了工业 3.0 时代，工业 3.0 是电子信息化时代，在工业 2.0 的基础上，电子与信息技术被大范围的推广使用，这使得制造业中的自动化控制程度获得了大幅度的提升进步，工厂的生产效率、良品率、机械设备的使用寿命和工人的分工合作都达到了前所未有的高度。在工业 3.0 时代，PLC 和 PC 大量被应用，人类的生产化逐渐变得智能，大量的机器代替原本人类的工作，使得人类得以解放双手；德国政府在 2013 年 4 月的汉诺威工业博览会上正式提出了工业 4.0 的战略[1]，中国政府也在 2014 年提出了《中国制造 2025》[2]，中德两国都在相近的时间提出了工业创新改进的概念，中德两国也因此进入了工业合作新时代，工业 4.0 是智能化的时代，各国都着力于提升制造业的智能化水平，建立有适应性、高效率及基因工程学的智慧工厂，有效利用云计算、大数据、人工智能和物联网等关键技术。

德国和美国等国家作为发达国家，其在制造业和互联网方面的实力都处于领先优势，都力求在第四次的工业革命中获得主动权，占领技术高地。中国作为世界的制造业大国，在实力技术方面还处于劣势，但是本次工业革命是中国制造业弯道超车的机会。本次工业革命世界各国都不约而同的将智能化和互联化作为发展方向，都有意将物理资源与虚拟世界相连接，可见万物互联和虚拟化平台都是各国在本次工业革命中努力发展的技术。在此背景下，云制造的概念被提出，云制造是一种面向需求，由用户主导并将现实世界的物理资源和虚拟的网络联系起来的智能化制造新模式[3-5]。云制造是云计算、大数据、工业互联网和物联网等技术在制造业方面的综合应用[6]，云制造将现实生活中的各种实体制造资源虚拟化和服务化并集中存放在服务云池，利用云平台的集成化和智能化来在虚拟环境中调度资源完成用户提出的需求[7-11]。云制造的目的在于为制造业降低成本，实现资源共享以提高制造业的效率。传统的企业生产工序的调度时在满足一定的约束条件

下完成企业的利润最大化。作为一种新型的产业，云制造的调度是面向服务和以客户为中心的，并且是需求驱动的制造模式。云制造允许客户参与从发起产品需求到产品设计、制造和测试等一系列行为中，与传统制造业相比，在云制造中用户的参与感会大大提升，客户更容易得到符合自己个性化设计的产品。

云制造主要分为三个主体：用户，物理资源提供商和云制造服务平台。用户作为云制造服务平台的使用者，在云制造中扮演了提出需求的角色；物理资源提供商主要是拥有操作设备和加工设备的厂家，一般情况下可以提供例如焊接设备、测试设备、数控机床和热处理设备之类的设备；云制造服务平台主要负责管控整个云制造环境，需要将用户的需求和物理资源提供商提供的服务转换为虚拟资源存放在云制造平台中，然后根据用户的需求和物理资源提供商提供资源的具体情况，通过云平台的集中管理并分配各种异构计算资源来处理匹配这两者的需求。对于云制造平台来说，可以在更短的时间内提供给用户更好的方案可以给用户带来更好的体验，也可以给自身获取最大的利润。因此，对于云制造服务平台来说，有一个可靠合理的调度方案显得尤为重要，这不仅可以获得更大的利润，还可以最大程度的保持平台自身的可靠稳定性，延长物理硬件的使用寿命，减少不必要的开销。

云制造服务平台的资源调度主要分为制造资源调度和计算资源调度两大类。在制造资源调度方面，提供制造资源的企业分布在不同的地理位置，可能同一个需求需要有许多个处于不同地理位置的企业进行合作，这其中就需要考虑到不同企业之间的物流调配时间。在计算资源调度方面，计算资源一般是根据用户的需要动态调配给所需的用户，云制造服务平台的计算资源是制造资源调度的基础，其主要分为基础设施即服务层资源调度、平台即服务层资源调度和软件即服务层资源调度。

在实际应用中，云制造服务平台收到的任务请求大致可分为单服务需求任务和多服务需求任务两大类任务，单服务需求任务只需要一个服务即可完成任务，二多服务需求任务包含一系列子任务，需要平台提供多种不同类型的服务。云制造平台允许多个用户同时发起多个需求，首先，云制造服务平台会根据自身的资源使用情况来对任务进行分配，假如任务过大，那么会现将任务进行分解，然后根据云制造服务平台上拥有的物理资源进行匹配，在这一过程中就主要使用到了调度算法，例如能耗最低、时间最短等目标算法就在这一过程中实现；然后云制造服务平台根据调度结果会把各个任务分配给下面的各个物理资源提供商，因此任务完成的总时间取决于云制造平台调度的效率和底下服务提供商完成工作的效率。在此背景下，如何建立一个任务调度的模型可以有效缩短调度的时间是有十分重要的意义的和研究价值的。当前的各种智能算法、改进算法基本都还是在算

法复杂度和性能之间找到一个平衡，并不能很好的从根本上去解决问题。针对上诉的这些问题，本文提出利用群组优化调度后的数据来训练深度学习模型，然后利用训练好的深度学习模型对调度结果直接实现预测。

1.2 国内外研究现状

自从云制造的理念问世以来，在工业界以及学术界的大力推动之下，云制造在近些年发展趋势十分迅猛，例如美国建立的 MFG.COM 云制造平台，MFG.COM 平台将物理资源供应商的信息虚拟化并封装进云制造平台的资源池，用户只需要将需求发布在平台上，平台就可以很快的根据用户需求匹配到最佳供应商，实现资源的最优化配置；波音公司在基于网络协同、制造服务外包的模式下组织全球 40 多个国家和地区共同研发波音系列飞机，这不仅大大缩短了研发周期，还有效的降低了大量成本。从国内的角度来说，来自南京的 1001 云制造平台将物理资源提供商方面的闲置资源有效利用起来，为用户提供满意的服务，让制造成为像水和电一样随时可以为公众取得的服务；航天二院和北车集团也正在建造云制造服务中心，争取更好的利用各种自造资源为集团内部的制造业服务。

云制造在商业方面的应用正在取得前所未有的成功，在此同时，学术界对云制造的研究也进入了白热化的阶段。在云制造中，所有的制造资源（包括物理资源等）都被封装在云制造的服务平台中[12]，因此根据用户的需求来匹配制造资源很重要，在这一步，调度算法的优劣会直接影响到用户的体验以及平台的成本开销。因此，一个良好的调度算法对云制造平台有着十分重要的意义。

云制造的核心问题就是如何更加合理的对云制造平台拥有的资源进行调度和分配，云制造平台的资源主要包括了计算资源和制造资源两大类，因此目前对于云制造调度问题的研究主要是针对于这两大类资源的调度。在计算资源调度方面，Laili Y 等人为了实现计算资源的最佳分配在云制造系统中提出了一种新的 OARC 模型，在该模型中考虑了特殊情况下的计算、通信和可靠性的约束，并在此基础上提出了一种基于小生境免疫的改进算法[13]，该算法可以更为有效的对云制造平台中的计算资源进行合理分配，但是该模型只是单纯的针对了计算资源的调度，算法的并行化和简化方面做的也不是特别好；因此，在这之后 LaiLi Y 等人将 OARC 模型通过中央控制台的集中管理和服务优化选择(SCOS)结合起来并引入了排序混沌算法，使得云服务和计算基础架构可以通过高效的决策快速组合[14]，该模型将 SCOS 和 OARC 结合起来，使得效率大大的提高，并且配合全新引入的算法，模型的速度和稳定性都有了很明显的进步；Zhang 等人研究了资源服务组合整个生命周期中灵活性的定义和分类，目的在于提高服务质量的优化配置，同时还提出了

一种灵活的资源服务组合管理体系结构[15]；Li 等人提出了一种基于服务输入输出接口的基于聚类网络的服务组合方法，可以有效地处理大规模的服务组合问题[16]。在制造资源调度方面，Zhu C 等人利用不同企业的拥有的制造资源来处理同一批任务，并考虑制造资源的不同分布针对负载平衡、成本最小化和最小处理时间三个目标进行了优化[17]，作者根据不同的需求提出了不同的应对方案，这对于复杂的云制造平台需求是十分具有意义的，可以更好的根据问题来应用解决办法，但是在实际应用中大多是复合型需求，该模型的适用性还待进一步增强；Zhou L 等人研究了分布式 3D 打印机的调度问题，建立了关于云制造调度平台中 3D 打印服务的服务匹配模型，考虑了模型大小、打印材料、打印精度、成本和物流等因素，通过条件的约束和算法的优化大大减小了任务的交付时间[18]，但是该模型只考虑了单纯的任务完成时间，并没有将实际应用中的物流时间以及 3D 打印机的一些突发情况考虑进去。

云制造的核心问题之一就是如何更加合理的分配各类资源，保证云制造平台的高效率运转以及用户获得更好的使用体验，针对这方面的问题有许多研究人员都提出了相关的方案。在云制造平台的调度过程中，服务匹配是需要考虑的比较重要的一个点，合理的服务匹配可以让云制造平台的调度起到事半功倍的效果，Y. Cheng 等人为了更加经济有效的整合各类分布式资源提出了 SOM 系统制造服务模型，该模型中的制造服务网络和制造任务网络可以很好的揭示各个服务和各个任务之间的匹配相关性[19]；H. Li 等人提出了基于工作流中资源服务之间的依赖成都的两阶段组合方法，先利用资源服务之间的相关性和依赖性来解析初始组合，然后通过算法来对服务组合进行更加合理的分类[20]。但是同时我们也不能忽视关于平台调度产生的能耗、成本等问题。因为合理的控制能耗、成本不仅有助于减少污染排放，还可以有效的降低云制造平台的运营成本。Xiang 等人建立了考虑 QoS 和能耗的多目标服务组合，并且引入了 Group Leader 算法来解决资源分配的问题[21]；Cheng 等人提出了考虑能源消耗、成本和资源分配风险的综合效用模型；Beloglazov 等人提出了能量感知分配启发式方法为客户端应用提供数据中心资源，以用来提高数据中心的能源效率，同时保证在云制造服务下的 QoS 值[22]；Uddin 和 Rahman 等人为大型复杂的服务器农场开发了能源效率和低碳促成绿色 IT 框架，以节省电力能源消耗并且同时控制降低温室气体的产生和排放，从而达到云制造平台环保的目的；Tao 等人为考虑能耗的资源分配制定了更为全面的数学模型，并提出了一种基于案例库和 Pareto 解决方案的混合遗传算法，该算法可以有效的提高内存的访问效率[23]。

从以上的研究现状可以看出，归根结底，云制造的调度是对多目标问题的求解，不同的算法对云制造调度结果的约束条件不一样。因此，求解多任务调度选

择合适的约束条件会对平台调度产生多方面的影响。本文的模型不是单纯的考虑算法理论层面上的值，还将物流因素加入进模型，因此在实际应用上也是具有意义的。

1.3 本文的研究内容和主要创新之处

针对上述提到的国内外研究现状存在的问题和解决方案，本文主要做了以下几个方面的研究：

（1）在云制造调度方面，建立基于 QoS 和总任务执行时间最短的调度模型；

（2）在针对调度算法的优化效果方面，本文首先研究了蝙蝠算法和新蝙蝠算法，在新蝙蝠算法的基础上加入了二阶振荡和差分进化算法对其进行改进，提出了改进的新蝙蝠算法，并使用改进的新蝙蝠算法对本文提出的多目标问题进行求解，同时与其他一些算法作对比；

（3）提出了改进的深度学习模型，在传统的 DBN 网络训练中引入自适应学习率，并使用改进的自适应学习率对模型的训练次数加以控制达到快速学习的目的；

（4）最后，使用了改进的新蝙蝠算法对云制造多目标调度问题进行仿真求解，获得调度的数据集，然后使用数据集训练改进的深度学习模型，然后使用深度学习模型对任务的调度结果进行预测并与传统的方法进行对比，以证明该方法的实用性。

1.4 论文组织结构

论文的正文部分主要分为六个部分进行阐述，每一部分的具体内容如下所示：

第一章：绪论。第一章主要介绍了本文的研究背景以及研究的意义，同时讲述了国内外关于云制造方面的研究现状，并简要说明了本文的研究内容和创新之处。

第二章：主要介绍了云制造和深度学习的相关概念和关键技术，并对云制造调度方面常用的一些算法做了简单的介绍。

第三章：提出了本文研究内容的调度算法模型，该模型基于 QoS 的服务匹配和总任务执行时间最短两个方面，并对模型进行了详细的介绍说明。

第四章：根据调度模型提出了改进的新蝙蝠算法，首先介绍了新蝙蝠算法，然后详细说明了对算法的改进过程。

第五章：引入了深度学习模型，并对深度学习模型的学习率和模型的训练次

数算法进行了改进。

第六章：本章主要是实验的仿真结果，主要分为两个部分，第一部分是对改进的新蝙蝠算法的仿真对比试验，第二部分是深度学习的改进效果对比和深度学习预测调度结果和传统调度算法的调度结果的对比，通过实验对比说明本文工作的意义。

第七章：本章主要是对论文进行了总结和展望，总结了自己的论文工作和不足之处，并展望下一步的研究方向。

1.5 本章小结

本章节首先介绍了论文的研究背景，通过背景介绍来说明制造业的重要性并说明了云制造方向的研究意义。然后介绍了目前国内外对于云制造应用方面的一些实例，通过实例的应用可以更深入的了解云制造在现代制造业中的应用场景。同时本文根据云制造平台资源种类的不同介绍了相关的资源调度、云制造平台服务匹配的相关研究工作，通过国内外学者对于云制造工作的研究可以对本文的研究内容有着更明确的认知。最后在此基础上总结了云制造调度方面的问题，并提出了本文的解决方案和使用的技术路线，也对论文的总体架构进行了简要的说明。

第二章 相关研究与关键技术

2.1 引言

本章主要介绍了云制造方面的一些相关技术和相关知识。提到云制造，还会有许多人对云制造的概念不是十分的清楚，而且都会联想到云计算，云制造和云计算的关系是密不可分的，云计算是云制造服务体系的基础，云计算主要在云制造平台计算资源调度中扮演着重要的角色。因此，本章首先对云计算的相关知识进行介绍，然后了解云制造资源优化的问题并对国内外现阶段的研究现状进行了介绍说明。最后对本文的工作的展开进行了说明。

2.2 云计算的相关介绍

2.2.1 云计算的定义

对于云计算的定义有着很多种解释，现阶段被广泛接受的解释是美国国家标准与技术研究院的定义：云计算是一种按使用量付费的模式，这种模式提供可用的、便捷的、按需的网络访问，进入可配置的计算资源共享池（资源包括网络、服务器、存储、应用软件和服务），这些资源能够被快速提供，只需要投入很少的管理工作或与服务供应商进行很少的交互。

云计算是由分布式计算、并行计算、效用计算、网络存储、虚拟化、负载均衡和热备份冗余等多学科领域技术融合进化的产物[24]。根据市场调研，全球目前云计算基础设施的支出占 IT 行业总支出的三分之一，云计算现在已经成为全球各大 IT 厂商争夺的最重要的领域之一。国内外的云计算市场也是异常火爆，例如国外的微软、甲骨文、谷歌和国内的阿里云、华为云、腾讯云等公司都是云计算技术十分出色的，云计算的发展也极大的刺激了市场。

2.2.2 云计算的特点

云计算将计算分布在大量的分布式计算机上，而不是放在本地的计算机或远程服务器中，每个用户都可以根据自己的需要将资源切换到需要的应用服务上，云计算将计算资源同煤气、水电一样作为一种商品流通，云计算具有以下一些特点[25]：

（1）超大规模：云计算的规模普遍都比较大，例如谷歌公司的云计算项目已经拥有了上百万台服务器，谷歌公司庞大的服务器规模赋予了客户强大的计算能力，这是传统的计算机不能比拟的；

（2）虚拟化：云计算可以满足在任意一处位置使用多种终端来获取服务，用户所使用的资源在云计算平台的某一处地方进行运行，用户不用了解与担心平台的地址与后期的维护，只需要按照自己的去要去访问服务器获取相应的服务；

（3）高可靠性：云计算在保障服务的可靠性上使用了计算节点同构可互换、数据多副本容错等措施，相比传统的计算机提高了可靠性；

（4）通用性强：云计算没有特殊的针对群体，在云计算资源的支撑下可以同时应用成千上万个不同的应用；

（5）便捷与廉价：云计算的规模可以动态伸缩，可以满足客户不同的需求，由于使用云计算的客户不需要对数据中心进行管理，云计算使得资源利用率有了大幅度的提高，因此云计算在成本方面有着较大的优势。

2.2.3 云计算的服务模型

如图 2-1 所示，云计算服务模型可以分为软件即服务（Software-as-a-Service）、平台即服务（Platform-as-a-Service）和基础设施即服务（Infrastructure-as-a-Service）三个部分。

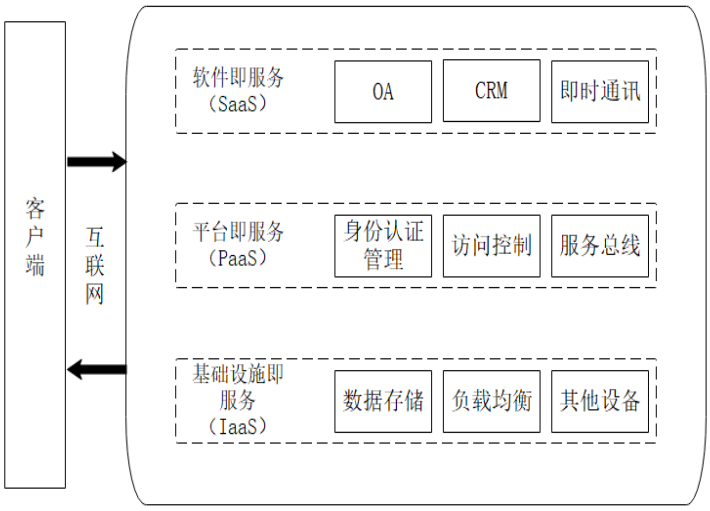

图 2-1 云计算服务模型

Figure 2-1. Cloud computing service model

软件即服务（SaaS）：将某些特定应用软件功能封装成服务，例如 BlueScope 集团从 2008 年就开始部署基于 SaaS 的 CRM 系统，SaaS 只提供某些专门用途的

服务供应用调用；

平台即服务（PaaS）：云计算用户应用程序的运行环境由平台云负责提供，例如 Google App Engine、MicroSoft Windows Azure 等等。PaaS 自身负责资源的动态扩展和容错管理，用户应用程序不必过多考虑节点配合问题，但是用户必须在特定的编程环境下使用特定的模型；

基础设施即服务（IaaS）：将硬件设备等资源封装成服务供用户使用，以往用户想运行一些企业应用需要自己购买服务器和硬件设备来保证应用的正常使用，IaaS 为用户提供了场外服务器和硬件设备，大幅度减少了用户的成本。在 IaaS 中用户相当于在使用裸机和磁盘，用户可以使用不同的系统，但是必须协调好多个机器的协同工作。

2.2.4 云计算的部署形式

云制造按照部署形式的不同可以分为私有云、公共云、社区云和混合云四种部署模型。

私有云：云端的资源只给一个单位组织内的用户使用，这是私有云的核心特征，而云端的所有权、日常管理和操作的主体到底属于谁并没有严格的规定[26]。私有云模式可以为企业带来显著的帮助，也可以更大成度的保护企业的私密性，但是在私有云模式下云计算平台的检查维护工作也需要自己来负责。

公有云：在公有云模式下，所有的应用程序、资源和其他服务都是由云服务提供商来供应给客户[27]，这些服务基本都是免费的，但是也有一些服务需要根据用户的使用量来进行收费，这种模式只能通过互联网来访问和使用，公有云的私密性与私有云也有一定的差距。

社区云：社区云模式是建立在一个特定的小组里多个目标相似的公司之间的，这些公司共享一套基础设施，各个公司共同承担该模式所产生的成本，该模式下可以使社区内的成员具有较高的参与性[28]。

混合云：混合云融合了公有云和私有云两种模式，是近些年来云计算的主要模式和发展方向，私有云的面向客户主要是企业用户，企业用户处于安全考虑更愿意将数据存放在私有云中，但是又希望获得公有云的计算资源，在这种情况下越来越多的混合云模式被采用，这种个性化的解决方案达到了既节约成本同时又可以拥有较高的安全性。

2.3 云制造资源优化问题简介

2.3.1 云制造的定义

云制造是借助云计算的力量成长起来的新兴概念，其理念是制造即服务。云制造是先进的信息技术、大数据技术、制造和物联网技术等多种技术交叉融合的产品。采取包括云计算在内的多种当代信息技术前沿理念，支持制造业在当今高度发达的网络资源环境下为产品提供高附加值、低成本和全球化制造的服务。

云制造作为一种新型的制造模式[29-30]，其理念是资源的互联和共享，因此，如何更好的实现资源的分配和共享以用来提供更好的服务时云制造的关键问题。云制造的环境较为复杂，拥有的资源数量、资源类型和一些用户的个性化需求都显著高于传统制造行业，所以云制造的资源优化问题、资源匹配问题的难度和复杂度都远高于传统的制造行业。传统的制造业中调度决策往往是在车间层决定的，制造资源和调度系统的决策距离更近，这就使得调度系统解决问题更加及时，也可以更好的应对一些突发情况。而在云制造中，调度系统位于云端，各类制造资源位于不同的地点，调度系统依靠互联网获取各类制造资源的信息，这就导致调度系统对于获取各类制造资源的信息有一个延迟性，因此云制造平台的抗干扰能力十分重要。云制造调度的问题主要有以下几个特点：

（1）规模大：与传统制造业相比，云制造不仅局限某些设备设施的生产，云制造平台拥有更多的制造资源，可以实现更多种类的产品。

（2）资源统一管理：云制造平台虚拟化了所有可利用的制造资源，并存放在云制造平台中，可以更快更好的知道各类资源的使用情况，相对于传统制造业可以更好的实现不同领域制造业的合作。

（3）智能高效：云制造平台管理更多的制造资源，根据用户需求统一对资源进行合理安排，并且云制造平台的资源和用户都有着更大更广的覆盖面积，在一定程度上打破了原有的传统制造业的地域局限，同时可以根据用户自己的需求提供更符合用户自身需求的服务，可以让用户获得更高效、便捷且个性化的服务。

（4）成本更低：云制造平台统筹了各项资源之后可以为同一类服务选择更为合理、成本更低的服务，有效的降低了平台和用户的开销。

2.3.2 云制造平台的资源分类

云制造平台中的资源按照资源类型可以分为制造资源和计算资源两大类。云制造平台中的制造资源是云制造平台的一大特点之一，制造资源主要是将现实中的一些物料配套资源、人力资源和设备资源等资源进行虚拟化并封装形成制造服务然后存放在云制造平台上以供之后方便调用。但是在实际使用的环境中，制造服务所代表的制造资源会处于不同的地理位置，因此在调用制造服务的时候，不

仅需要考虑各个制造服务之间的关联性，还需要考虑不同地理位置的制造服务所产生的时间和物流开销。计算资源位于云制造平台的计算中心，云计算的调度技术是云制造计算资源的基础，因此云制造的计算资源和云计算一样也包括基础设施即服务（IaaS），平台即服务（PaaS）和软件即服务（SaaS）。与云计算有着本质区别的是，云制造平台中的计算资源不仅提供计算调度，还在制造资源虚拟化和后期的管理方面起了至关重要的作用。

云制造平台根据自身制造资源虚拟化得来的制造服务，通过计算资源的调度来得到一个满足用户需求的调度方案，这些调度方案最终会下达到制造服务所对应的各个制造资源的企业和车间。

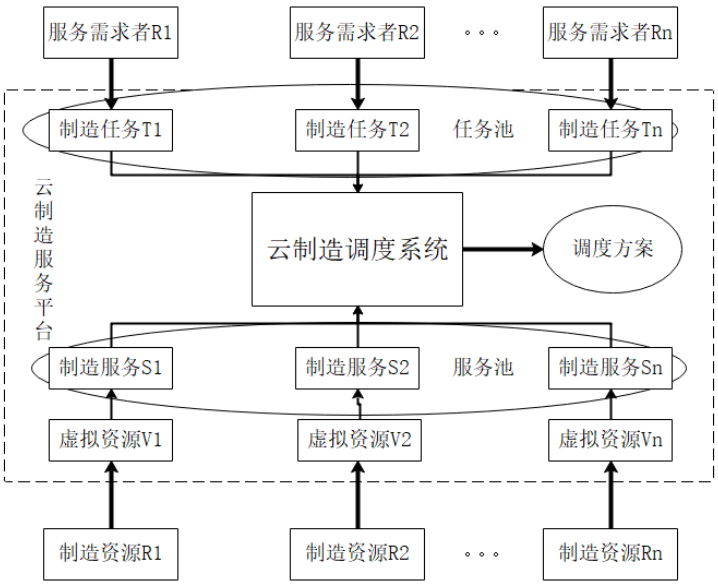

图 2-2　云制造的运作模式

Figure 2-2. Operation Mode of Cloud Manufacturing

从图 2-2 中可以简单了解云制造平台的运作模式，可以明白云制造平台是如何将制造资源和用户的需求联系起来。

2.4 常用的资源调度算法研究

在云制造环境下，调度问题一直是众多公司和学者一直在研究探讨的问题，在调度中被研究的最多的就是任务切割和服务组合匹配的算法，其实质是将任务池中的任务先与云制造平台中的制造服务进行匹配，若任务过大不能匹配到相应的制造服务，则将任务分解为一系列的子任务，再将子任务与制造服务进行匹配。

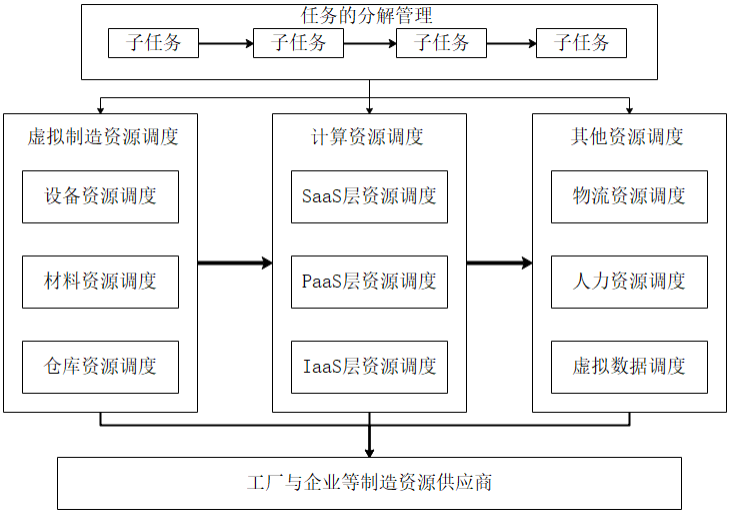

图 2-3 云制造平台的调度过程

Figure 2-3. Scheduling Process of Cloud Manufacturing Platform

如图 2-3 所示，任务在上传至云制造平台之后首先会根据需求被分解为一个个子任务，然后匹配对应的虚拟制造资源进行匹配，虚拟制造资源是由各类工厂和企业提供的，经过云制造平台的虚拟化并封装成一个个方便存储的云制造资源。云制造平台在接收到服务请求之后会调用计算资源对全部子任务进行分析排序，在各类服务中选择最合适的服务来进行服务组合并执行最优的选择。但是每个云制造平台对服务的选取有着不同的调度算法，每个调度算法的约束条件也不一样，因此每个云制造平台所产生的调度结果也有较大的区别。

针对以上的问题，有许多专家也在致力于提出更合理、快速和可靠的算法来解决。不同的算法所针对的约束条件也不一样，针对约束条件的不同所得到的调度策略也大不一样。因为调度问题被证明是典型的 NP 难问题，近些年来，启发式智能算法在解决 NP 难问题上有着非常不错的效果。目前，启发式智能算法经常用于在云制造平台中搜索服务组合的最优解。一些比较经典的启发式智能算法有：粒子群算法、遗传算法、鸟群算法、蚁群算法和蝙蝠群算法等。

2.4.1 粒子群算法

粒子群算法是由 J.Kennedy 和 R.C.Eberhart 在 1995 年开发的一种进化算法，粒子群算法是从随机解出发，通过一系列的迭代来寻找最优解，然后通过适应度函数来评价解的品质[31-38]。在粒子群算法中，每一个粒子都代表着当前问题的一个解，每个粒子都拥有自身的初始速度并且随机分布在规定的区域内，此外，每一个粒子都会记录自己搜索过的区域的最好位置和最佳的适应度函数值，所有粒子都可以知道当前群体中的最好位置，然后粒子会根据自身的历史最佳位置和当

前群体中的最好位置来不断的调整自身的速度和位置。

Tao 等人在考虑了模型中资源服务之间的相关性之后，利用粒子群算法来解决服务组合问题，其在粒子群算法中结合了非支配排序技术帮助粒子可以更好的选择自身最佳位置和全局最佳位置。在该改进的粒子群算法中，粒子群的位置和速度更新公式中的参数都是动态生成的，可以让粒子群算法不容易陷入局部最优解。

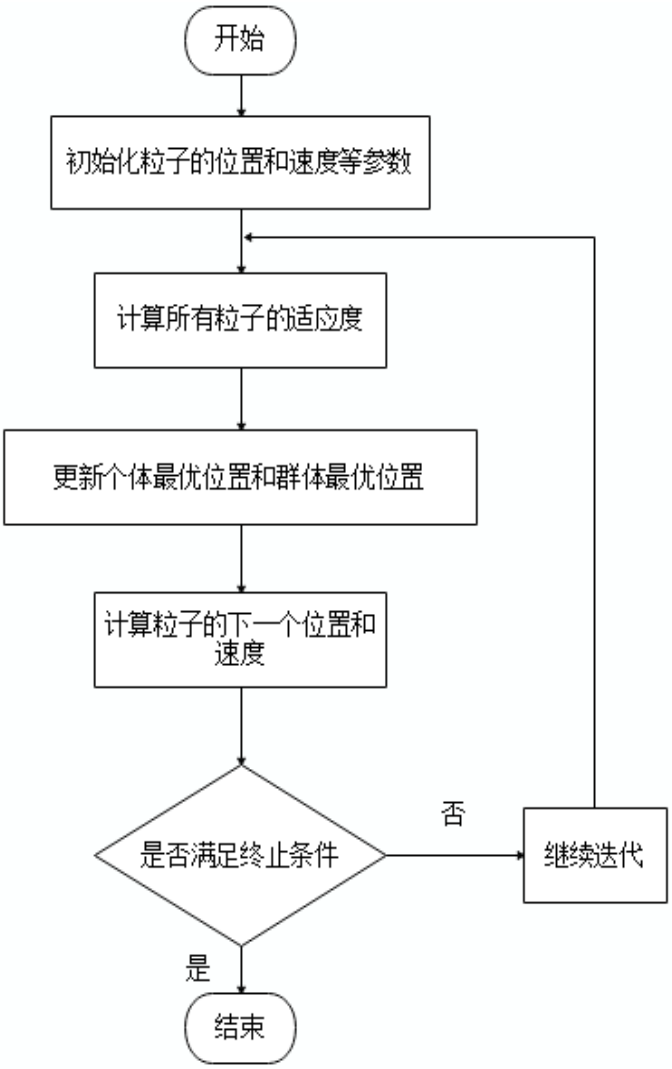

图 2-4 粒子群算法流程图

Figure 2-4. Particle swarm algorithm flow chart

2.4.2 遗传算法

遗传算法是 J.Holland 教授在 1975 年提出的，其起源于人们利用计算机对生物系统所进行的研究，借鉴了达尔文的生物进化论和孟德尔的遗传学说，遗传算法模拟了自然界生物在进化过程中优胜劣汰的天性，是一种高效、并行的全局搜索方法[39-45]。遗传算法的主要特点是对于需要求解的优化问题没有太多的数学要求，

因为遗传算法的进化特性，对目标函数的要求比较低，目标函数是否线性、是否连续都不影响处理结果；具有内在的隐并行性和更好的全局寻优能力；进化算子的各态历经性让遗传算法可以很有效的进行概率意义的全局搜索。遗传算法是利用种群搜索技术将种群作为一组问题的解，通过对当前种群施加类似生物遗传环境因素的选择、交叉和变异等一系列操作来产生新一代的种群，并通过适应度函数来逐渐将种群逼近至最优解的状态。参数的编码、种群的初始化、适应度函数的设计、遗传算子的设计和参数控制是遗传算法的核心内容。

现在遗传算法的应用十分广泛，例如 Jin 等人提出了一种关联感知云制造服务模型，用于表征单个服务对其他相关赋予的 QoS 的相关依赖性，并且在这个模型上提出了一种服务相关的映射模型用于自动获取各服务之间的 QoS 值，然后通过遗传算法在相关感知的服务中选择最优的组合方案。Ding 等人提出了一种性能评估方法来计算事务性复合 Web 服务的执行时间，然后把这个性能评估方案加入进了遗传算法，实现该约束条件下全局最优的服务组合选择。

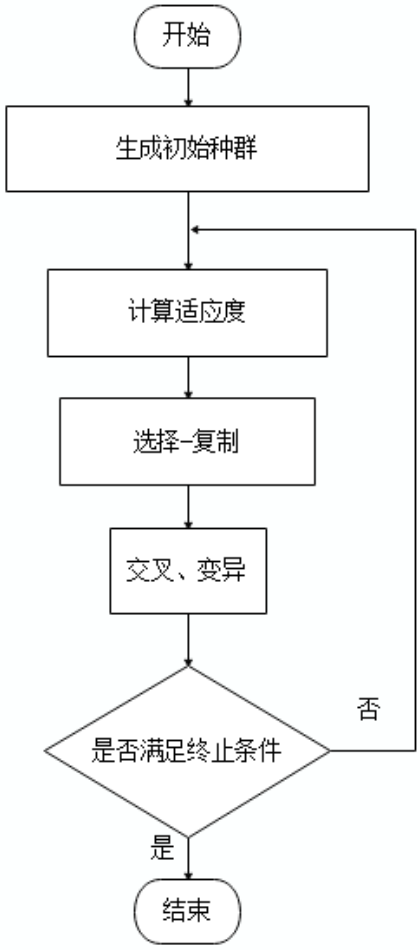

图 2-5　遗传算法流程图

Figure 2-5. Genetic algorithm flow chart

2.5 深度学习

深度学习一词由 Rina Dechter 在 1986 年引入机器学习社区，然后基于人工神经网络的基础上各领域的专家对深度学习开始进行研究。深度学习使用多层非线性处理单元的级联进行特征的提取和转换，每个连续层使用前一层的输出作为输入。Hinton 基于深度置信网络（DBN）提出非监督贪心逐层训练算法[46]，为解决深层结构相关的优化难题带来希望，随后提出多层自动编码器深层结构。大多数现在深度学习模型都基于人工神经网络，是基于对数据进行表征学习的方法，其本质是模仿了大脑视觉的分层神经网络，从接收到视觉信号开始，根据算法对图像进行初步处理并且提取出物体的边缘信息，接着抽象的判断物体的大致形状，最后再进一步的识别物体。

同机器学习一样，深度学习方法也分为监督学习和无监督学习，不同学习框架下建立的学习模型有很大的不同，经典的卷积神经网络和深度置信网就是有监督学习和无监督学习的典型代表。深度学习的发展主要分为以下两个阶段：

第一个阶段：在最早期的模式识别任务中，研究者的目标一直是使用可以训练的多层网络来替代经过人工选择的特征，虽然使用多层神经网络很简单，但是训练的结果都很糟糕。直到上个世纪的八十年代，使用简单的随机梯度下降来训练多层神经网络，这种糟糕的情况才有所被改变，只要网络的输入和内部权值之间的函数相对平滑，梯度下降的效果就十分不错，在上世纪八十年代前后，有许多不同的研究团队发明了不同的梯度下降方法，其中反向传播（BP）神经网络算法是应用最广泛的神经网络模型之一，BP 神经网络算法是根据误差反向传播算法训练的多层前馈网络。BP 网络可以用来学习和存储大量的输入输出模型的映射关系，而不需要事先公开描述这些映射关系的数学方程。BP 算法的学习过程由信号的正向传播（求损失）和误差的反向传播（误差回传）两个过程组成。正向传播时，根据输入的样本，给定的初始化权重值和偏置项值，计算最终输出值以及与实际值之间的损失值，假如损失值不再给定范围内则进行反向传播的过程，否则停止更新权重值和偏置项值；反向传播时，将输出以某种形式通过隐层向输入层逐层反传，将误差分摊给各层的所有单元，从而获得各层单元的误差信号，这个误差信号作为修正各个单元权值的依据，这一计算过程使用梯度下降法完成，这个过程是周而复始的进行的，直到网络的输出误差降低到可以接受的范围或者达到了预先设定的学习次数限制。这个时期的神经网络基本只有输入层、隐层和输出层，因此也被称为浅层学习模型。在这之后因为多年没有特别突出的方法被提出，神经网络的热潮逐渐减弱。

第二个阶段：2006 年被称为深度学习的元年，在 2006 年 Hinton 提出了深层网络训练中梯度难题的解决方案：无监督预训练对权值进行初始化并增加有监督训练微调。其主要思想是先通过自学习的方法学习到训练数据的结构（自动编码器），然后在该结构上进行有监督训练微调。从 2012 年开始至今，深度学习的发展进入井喷式爆发期，2012 年 Hinton 课题组为了证明深度学习的潜力参加了 ImageNet 图像识别的比赛，通过构建的 CNN 网络获得冠军，该网络有如下创新点：

（1）首次采用 ReLU 激活函数，极大的增加了收敛速度并且从根本上解决了梯度消失问题；

（2）因为 ReLU 方法可以很好的抑制梯度消失问题，因此该网络完全抛弃了之前的训练方法，采用有监督训练。

从这次比赛之后，深度学习的主流变成了纯粹的有监督学习并开始使用 GPU 对计算进行加速。

从 2006 年至今短短的十多年，我们在日常生活中也见证了深度学习的快速发展，从现在我们使用的语音识别、智能客服和人脸支付都已经走入了我们的生活，极大的方便了我们的日常。通过众多公司、研究所和研究人员的努力，目前各种深度学习模型的错误率都十分低。

2.6 本章小结

云制造是基于云计算的基础上发展起来的一个新兴概念，因此本章首先介绍了云计算的相关知识，然后根据云制造的概念展开，介绍了云制造的资源分类，并说明了云制造服务平台是如何利用这些资源进行运作的，同时介绍了云制造服务平台进行资源调度时常用的算法。本文最后引入深度学习模型对调度结果进行预测，因此本章最后对深度学习的发展和深度学习的训练方法进行了分析。

第三章 模型的分析与建立

3.1 引言

云制造服务平台在接收到用户的任务请求之后会根据平台的资源和调度算法对任务进行分解和资源匹配，在该过程中不同的云制造服务平台所得到的调度分配结果是不一样的，这是因为每个平台所构建的模型特点不一样。本文构建的模型对云制造服务请求主要由以下两个阶段构成：（1）基于 QoS 的服务资源调度；（2）任务执行时间最短的服务资源调度。

基于 QoS 的调度目标主要是为了考虑云制造服务平台的经济性并且也考虑到了用户的满意度。而针对任务执行时间最短的调度策略则是考虑到了云制造平台的性能，并且可以在最短时间内给用户调度结果，获得用户的认可的同时可以实现利益最大化。本章主要介绍了本文的模型的分析与建立。

3.2 云制造的资源调度

云制造调度是建立在云计算调度的基础上的，与传统的调度方法不同，传统的调度方法主要的决策是位于车间层，而云制造将所有的资源虚拟化到云端的云制造平台，对于资源的调度通过互联网来实现。其本质最终的目标是将计算任务更加合理的分配到分布式的计算资源上以降低计算成本，提高资源利用率并提升用户体验。云制造平台中的服务不仅包括软件服务，还包括不同类型的制造服务，在大多数情况下，一个制造服务无法完成复杂的任务，因此，必须结合不同的服务来完成复合任务的要求。

在云制造中一般有三个用户角色，分别是服务提供者、服务平台管理者和服务需求者。服务需求者向服务平台管理者发起需求，服务平台管理者对需求进行响应，针对服务需求者提出的需求调动单一云服务或者是一系列的云服务组合，此时服务平台管理者对服务提供者发出请求，并根据需求落实相应的服务资源。因此，一个好的调度策略需要考虑到计算资源调度和制造资源调度两个方面。在云制造服务平台中，计算资源是云制造服务平台架构的基础，计算资源提供了云制造平台的底层计算能力和存储能力，同时计算资源的调度也是制造资源调度的基础。制造资源指的是一系列服务提供者拥有的硬件资源，例如焊接设备、热处

理设备、冲锻压设备和铸造设备等。在云制造服务平台中，这些制造资源会被虚拟化和服务化。虽然这些资源会被统一存放在云服务平台，但是在实际生活里他们是分布在不同的地理位置的，因此我们在考虑云制造服务调度的时候不仅要考虑服务的关联性，还要考虑不同的服务资源之间的物流成本。

在云制造平台中，一方面用户的任务量会逐渐增加变多，一方面云制造平台的候选服务也会因为更多的厂家企业加入进来变得更丰富，因此想找到最佳的服务组合解决方案会变得十分麻烦。本文首先根据研究的实际情况建立了基于 QoS 和总任务执行时间最短的调度模型，然后为模型选择了相应的算法，并讨论了在不同条件下的算法比较证明模型的可靠性。

3.3 模型的分析与建立

云制造的调度问题其本质是以一定的性能指标为前提，把任务按照设定的调度策略进行排序，将任务按顺序的分配到资源点上。云制造平台上的各个企业或者车间作为云制造平台的制造资源。由于云制造任务大多都为复合型的任务，所以我们需要先将任务分解为一个个可以由企业或者车间单独加工的子任务，然后根据调度策略进行排序使其达到最优。完成一个大规模复杂的云制造生产任务,一般经历三个过程：

(1) 首先将一个大型的任务经过多次分解后划分成独立的子任务,并且保证每个子任务由某一单一工序即可完成。

(2) 为每一道生产工序选择相应的生产资源和生产厂商。

(3) 确定每道工序之间的相互关系和每道工序之间加工物流的时间。而对于云制造业的批量生产任务，云制造生产管理系统需要根据批量任务数，计划完成时间，生产成本等要求，选择相应的生产资源供应商并确定生产工序。

整个云制造平台自顶向下可分为三个层次。第一层任务层，即云制造所要完成的批量生产任务所在的层次。任务层的工作内容主要是进行任务的切割与细化，将粗粒度任务逻辑切割为细粒度任务。任务切割后形成的子任务工作流决定了该任务的工序的数量和各个子任务之间的相互依赖关系。对于多个相同性质的云制造生产任务，它们的生产流程即生产工序是大体一致的。同时子任务的切割还要考虑到整个生产任务的数量。生产工序集合由 $P=\{P_1,P_2,.....P_n\}$ 表示。第二层是加工商调度，假设有 j 个加工商可以同时为工序 P_i 提供加工程序，则 P_i 的候选加工商集合可以表示为 $C_{Pi}=\{C_{P_i}^1,C_{P_i}^2,...C_{P_i}^j\}$。$r_{f(i)}^i$ 用来表示其中的一个生产工序 P_i 由加工商 $f(i)$ 来完成，其中 $f(i)$ 是工序 P_i 在其候选的加工商集中选择的合适的加工商，即

$f(i) \subset C_{Ti}$。所以工序 P 所使用加工商的集合可以表示为 $C=\{r^1_{f(1)}, r^2_{f(2)}, ...r^m_{f(3)}\}$。第三层是数据信息传递层，在云制造业中由于生产资源和生产厂商的分布式的特点，在各个工序之间需要由各个生产资源供应商进行数据信息甚至管理信息的双向传递。传递的信息包括上一道工序的生产完成信息，两两工序之间的实时物流信息等。$D=\{D_1, D_2, ...D_n\}$ 是转发信息的集合。各个生产资源的供应商之间均可进行信息的双向传输。

如上所诉，随着子任务分解之后需要对子任务和候选服务进行匹配，由于子任务和候选服务的数量我们要找到最佳组合的解决方案在不同情况下有很多选择的优化算法。在本文中，我们首先研究基于 QoS 和总任务执行时间最短的调度模型，然后讨论不同规模的任务和不同其他条件下的算法比较。

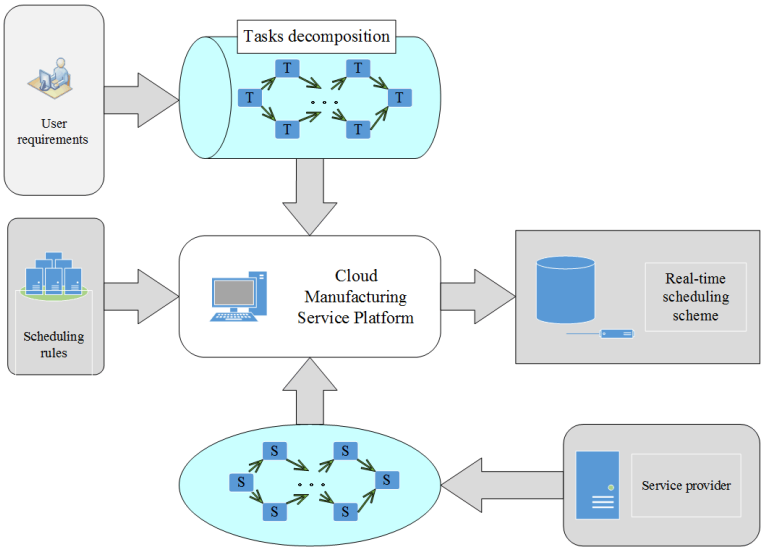

图 3-1　云制造平台工作示意图

Figure 3-1 Cloud manufacturing platform work diagram

QoS 作为面向服务体系中重要度量标准，云制造中 QoS 属性主要由服务，可靠性，可利用性和生产加工费用等方面的约束，其中服务的属性按照服务的等级分为一级，二级，三级等；可靠性主要由服务资源上运行任务的历史成功率来体现；可用性是加工商向用户提供的服务和用户需求的匹配程度。云制造平台的任务和资源都是实时变化的，用户对资源的访问成功率会随着时间的变化而产生不同的结果，在本文中，我们用数组 *AccessibilityTable*[] 来记录一天中每个小时的访问成功率。加工费用：表明了加工商加工某工序所需要的成本。

不同的用户有着不同的需求，因此其所看重的 QoS 属性也会有一些区别。根据用户的不同需求来改变目标函数的各个权重值，使用户的不同需求得到满足。集合 Q 代表了全局 QoS 属性，其中包含的多种不同的局部 QoS 属性，即 Q={QoS_1,

$QoS_2 \ldots QoS_i \ldots QoS_k\}$。而 α_w 是各种 QoS 属性在全局 QoS 中各自所占的权重比例，且 $\sum_{w=1}^{k} \alpha_w = 1$。下面的表达式描述了加工工序 P_i 使用加工商 $f(i)$ 的各个 QoS 属性度量值的总和：

$$QoS(P_i) = \sum_{W=1}^{k} (\alpha_w \times QoS_w(r^i_{f(i)})) \tag{3-1}$$

其中 $QoS_w(r^i_{fL(j)})$ 代表了加工工序 P_i 使用加工商 $f(i)$ 中属性 QoS_w 的度量值。加工工序集合 $P = \{P_1, P_2, \ldots P_n\}$ 当前选择的一套完整的加工的路径可以表示为 $C = \{r^1_{f(1)}, r^2_{f(2)}, \ldots r^m_{f(3)}\}$ 的加工工序的 QoS 表达式可以描述为：

$$QoS(P) = \sum_{j=1}^{m} \sum_{w=1}^{k} (\alpha_w \times QoS_w(r^j_{f(j)})) \tag{3-2}$$

从上式可以看出，不同的 QoS 属性权重值设置会在很大程度上影响最优服务路径的选择。因此，可以考虑根据用户不同的需求和局部属性要求来切换不同的服务提供商，从而可以给用户再云制造平台中挑选最适合用户期望的制造服务。

对于大规模的云制造生产任务主要的优化是尽可能的减小整个任务的执行时间而加工工序分配的前提条件是每个加工服务请求必须使用其对应的最优服务组合。所以对于大规模的云制造来说主要关键点如下：

（1）物流时间：云制造管理系统集中协调控制生产工序节点之间的物流管理信息，从而形成云制造物流链。而异地之间的生产工序节点则通过第三方的物流配送完成细粒度任务之间的连接。所以物流的时间一定程度影响了大规模批量云制造任务完成的时间和成本。

物流时间矩阵 $\log_{n \times n}$ 记录了生产节点之间的物流时间开销，其中 $\log_{ij}$ 代表了生产节点 i 到生产节点 j 的物流时间开销。对于同一类型的每个云制造生产任务，生产过程中的物流时间开销是一致的，但是两两生产工序之间物流时间一定程度上影响了生产任务之间的顺序。若对于某一任务而言，其中的云制造车间中的某道工序是不需要的，那么对应的物流时间即为上一道生产时间不为 0 的工序到生产下一道生产时间不为 0 的工序之间的物流时间。

（2）总生产时间最小化：云生产平台中通常存许多的加工供应商，这些供应商通常在加工成本，可靠性，可用性等方面各有特长如：有的加工商在加工某一道工序时加工成本较高但是可用性较低，但是加工另一道工序时可能成本较低，可靠性较高，不同的加工商加工不同的工序时 QoS 属性各不相同。基于 QoS 的约束模型为各个复杂的加工任务选择最优的的服务组合。假设有加工请求 $J = \{J_1, J_2, J_3 \ldots\ldots J_m\}$ 等待云制造平台加工平台处理，经过上面的最优加工服务组合可以确定出多维数组 $Time_{m \times n}$，其记录了各个生产任务在每道生产工序上的时间花

费，$Time_{ij}$表示第i号任务在第 j 道工序上P_j执行所花费的时间。$STime_{rs_k^i}$与$ETime_{rs_k^i}$分别表示J_i在第k道工序P_k上开始生产的时间和结束时间。由于每道工序每个时刻只进行一个任务的生产工作，并且每个生产任务必须按照生产顺序执行，所以存在以下的约束关系：

任务J_i在工序P_k上执行完后，需要在P_{k+1}上执行。但是每道工序同一时间内只能执行一个任务，所以此时任务J_i是否能在P_{k+1}上执行取决于P_{k+1}上现在是否有任务在执行。而任务是按照序列数组J来执行的，所以如果P_{k+1}上仍有任务在执行，那么必然是J_{i-1}.所以任务J_i在执行P_{k+1}上执行的起始时间是任务J_i在P_k上执行的终止时间与J_{i-1}在P_{k+1}上执行的终止时间的较大值。

假如将物流时间考虑在内，我们需要考虑以下两种情况：

对于任务J_i,若它在云制造车间中的第k道工序的生产时间为 0，意味着该任务没有这一道工序。那么$STime_{rs_k^i}=ETime_{rs_k^i}=ETime_{rs_{k-1}^i}$。这是因为$J_i$不进行第$k$道工序，那么在第$k$道工序开始的时间就是上一道工序结束的时间，又因为这道工序的生产时间为0，所以任务J_i在第k道工序上开始和结束的时间是一样的。

对于任务J_i，若它在云制造车间中的第k道工序的生产时间不为0，即其需要进行该道工序。那么任务J_i在第k道工序的生产开始时间就应该是第k道工序前面一道生产时间不为0的工序生产结束时间加上该道工序到第k道工序的物流时间。

假设$d_k^i=lastpnz(k,i)$是第k道工序之前的一道生产时间不为 0 的工序。那么结合上述的情况，有如下关系：

$$STime_{rs_k^i}=\begin{cases}ETime_{rs_{k-1}^i}, Time_{ij}=0\\ \max(ETime_{rs_{k-1}^i}+\log_{d_k^i} i, ETime_{rs_{k+1}^{i+1}}), Time_{ij}!=0\end{cases} \tag{3-3}$$

$$ETime_{rs_k^i}=\begin{cases}STime_{rs_k^i}, Time_{ij}=0\\ STime_{rs_k^i}+Time_{ij}, Time_{ij}!=0\end{cases} \tag{3-4}$$

综上所述问题可以描述为：云制造生产车间共有m道生产工序和n个云制造生产任务。每个生产任务必须按照各自的生产工序进行生产，并且每道生产工序在任意时刻最多只进行一个任务的生产工作。已知每个任务所需的生产工序和在每道生产工序的生产时间(若当前任务没有此道工序,则在该工序的生产时间为0)，以及各道工序之间的物流时间的开销。所以在考虑物流时间影响的情况下，需要找到最优的生产任务顺序J，并按照此生产任务序列进行调度，使得批量云制造任务完成生产的总的时间开销达到最小。

3.4 多目标优化

多目标优化是在现实各个领域中都普遍存在的问题，每个目标都不可能同时达到最优。在单目标优化问题中，通常最优解只有一个，而且能用比较简单和常用的方法求出其最优解，然而在多目标优化问题中，各个目标之间都互相有约束，可能使得一个目标性能的改善会导致另外一个目标性能的损失，不可能同时使两个目标都达到最优。

处理云服务请求的传统研究大多都是只为服务选择一个单目标优化，但是在云制造服务平台中，每时每刻都会有用户上传新的需求，这就要求云制造平台在处理请求的时候考虑多个用户的需求，要根据实际情况来选择最优的服务组合。根据前两节的介绍，本文的调度模型时基于 QoS 和云制造任务完成生产的总时间开销最小的，所以本文调度模型的目标优化主要分为两个阶段，首先根据云制造平台接收到的用户需求，将用户上传的任务划分成可以由单独工序完成的子任务，然后根据云制造平台中的服务资源对子任务的服务请求进行匹配，首先满足的是用户的 QoS 期望和整个平台的全局 QoS 度量值；然后根据云制造平台选出的服务进行调度，在考虑各个资源供应商的实际情况选择最合理的资源以达到任务总的完成时间最小的目的。云制造平台调度的整个过程的目标优化的依赖关系如下图所示：

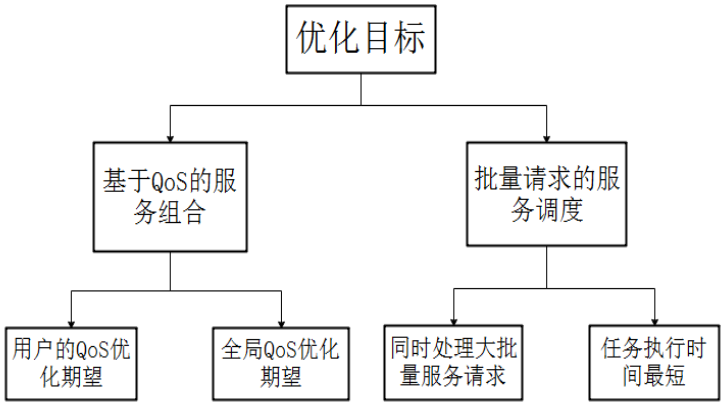

图 3-2 优化目标关系图

Figure 3-2. Optimization goal diagram

3.5 本章小结

本章主要介绍了本文的云制造调度模型，针对云制造服务平台的大批量服务请求同时到达，云制造服务平台主要针对 QoS 值和任务执行最短时间两个目标进行调度。本章的内容充实了论文在模型分析和建立方面的内容，说明了云制造平台的调度问题本质上是多目标调度的问题，并解释了本文在调度过程中对多目标选择和优化的策略。

第四章 基于改进蝙蝠群算法的云服务组合与调度

4.1 引言

本文提出的云制造平台的调度模型本质上是属于多目标调度问题，在第三章详细说明了本文的调度模型。针对多目标的优化问题，现在更多的是使用智能启发式算法来求解，在本章中我们首先简单的介绍了新蝙蝠算法，并根据本文研究内容的实际情况对新蝙蝠算法进行了优化改进。

4.2 新蝙蝠群算法简介

蝙蝠群算法是 Yang 等人在 2010 年提出的智能启发式全局搜索算法[47]，根据观察得知蝙蝠利用回声定位自己和食物的位置，蝙蝠在运动时会发出不同的脉冲发射率、波长以及响度来得到信息。蝙蝠算法为了模拟蝙蝠捕食的情况，做了如下假设：

设搜索空间为 D 维，蝙蝠在位置 x_i 时的飞行速度为 V_i，且蝙蝠在运动时发出声波频率固定为 f，但波长 λ 和响度 A_0 会根据食物的距离不同会自动调节，并且同时调整脉冲的发射率 $r \in [0,1]$。

蝙蝠的位置 x_i 和速度 v_i 的更新公式为：

$$f_i = f_{\min} + (f_{\max} - f_{\min})\beta \tag{4-1}$$

$$v_i^{t+1} = v_i^t + (x_i^t - x_*) f_i \tag{4-2}$$

$$x_i^{t+1} = x_i^t + v_i^{t+1} \tag{4-3}$$

上述式(4-1)中，β 为 0 到 1 之间的随机数，x_* 为当前最优解，一般情况下频率 $f \in [f_{\min}, f_{\max}]$ 其对应的波长范围为 $\lambda \in [\lambda_{\min}, \lambda_{\max}]$，但是 λf 在实际应用中一般是一个固定的值，因此可以根据需求的不同来调整波长。通过调整波长或者频率来调整所需的搜索范围，通常 $f \in [0, f_{\max}]$。频率越高波长就越短，蝙蝠的飞行距离随之也就变短。脉冲发射率在[0,1]之间。蝙蝠的脉冲响度 A_i 和发射率 r_i 按照下式

进行更新：

$$A_i^{t+1} = \alpha A_i^t \tag{4-4}$$

$$r_i^{t+1} = r_i^0[1-\exp(\gamma t)] \tag{4-5}$$

式中，α、γ 都为常数值，其中 $0<\alpha<1$，$\gamma>0$；A_i^t 和 A_i^{t+1} 分别是在 t 次迭代和 t+1 次迭代的时候蝙蝠发射脉冲的响度；r_i^{t+1} 是蝙蝠 i 在第 t 次迭代的时候的脉冲发射率；A_i^0 通常情况下在[1,2]之间，初始的 r_i^0 一般情况下都趋近于 0。

2015 年 Xian-Bing Meng 等人在上述的蝙蝠群算法的基础上提出了新型蝙蝠群算法（NBA）[48]。在自然界中，蝙蝠发射出的超声波脉冲参数会随着自己的生存环境的变化不断调整，这点特征有利于蝙蝠在不同的环境中更高效的活动。新型蝙蝠群算法相比于传统的蝙蝠群算法，其引入了栖息地选择策略以及自适应补偿回声多普勒效应机制。新型蝙蝠群算法规定所有的蝙蝠都可以随机的选择在不同地方的栖息地进行觅食；并且同时所有蝙蝠都可以从回声多普勒效应中获得补偿，并根据和目标的接近程度自适应的调整补偿率。

新型蝙蝠群算法中蝙蝠的栖息地的选择模型时基于量子行为和机械行为之间的选择，并且该选择随机决定，R 是一个随机数，如若随机数 $R<P$，则蝙蝠会在更大、更广阔的的空间中进行量子行为，其中 $0\le P\le 1$。蝙蝠的量子行为的位置变换公式如下所示：

$$x_{ij}^{t+1} = \begin{cases} g_j^t + \rho\left|m_j^t - x_{ij}^t\right|\ln\left(\dfrac{1}{u_{ij}}\right), R < p \\ g_j^t - \rho\left|m_j^t - x_{ij}^t\right|\ln\left(\dfrac{1}{u_{ij}}\right), R \ge p \end{cases} \tag{4-6}$$

上式中 g_j^t 是 t 时刻蝙蝠的最佳位置，m_j^t 是 t 时刻群体中的平均适应度函数值，ρ 为收缩膨胀系数，u_{ij} 是在[0,1]之间的一个随机数，P 在一般情况下取值为 0.5。

基于多普勒效应，蝙蝠的机械行为的位置和速度变换公式如下所示：

$$f_{ij} = f_{\min} + (f_{\max} - f_{\min})r \tag{4-7}$$

$$f_{ij}' = \frac{c+v_{ij}^t}{c+v_{gj}^t} f_{ij}\left(1+C_i\frac{g_j^t - x_{ij}^t}{\left|g_j^t - x_{ij}^t\right|+\vartheta}\right) \tag{4-8}$$

$$v_{ij}^{t+1} = wv_{ij}^t + (g_j^t - x_{ij}^t)f_{ij}' \tag{4-9}$$

$$X_{ij}^{t+1} = X_{ij}^t + V_{ij}^{t+1} \tag{4-10}$$

上式中，$f_{\min}$ 和 $f_{\max}$ 分别是频率的最大值和最小值；c 为声速，通常情况下取

值为 340；C_i 为范围在[0,1]之间的多普勒补偿率；w 是取值在（0,1）的惯性权重系数；ϑ 为一个较小的随机常数，其作用是避免分母为零；V_{ij} 为蝙蝠在该时刻全局最优位置上的速度，但次速度不会超过 c。

蝙蝠在接近食物之后，会更增加自身脉冲的发射频率、减少响度从而获得更加精准的局部搜索能力，应注意的是响度不仅会影响自身的行为，还会对其他蝙蝠产生影响，位置调整后的蝙蝠的速度更新公式如下：

$$x_{ij}^{t+1} = g_j^t + (1 + rand(0, \sigma^2)) \tag{4-11}$$

$$\sigma^2 = \left| A_i^t - A_{mean}^t \right| + \varepsilon \tag{4-12}$$

A_i 是该时刻所有种群的平均响度；ε 为一个随机值。

在新蝙蝠算法中，系数 G 是更新频率的判定系数，脉冲发射率和脉冲响度的更新都依据该系数来决定。若系数 G 设置的过小，算法将会不能快速收敛从而不能取到全局最优值；若系数 G 设置的过大，在进行局部搜索时蝙蝠群的多样性会减弱，可能导致算法陷入局部最优解。因此，在新蝙蝠算法中对于系数 G 的取值很重要。

4.3 新蝙蝠群算法的优化及改进

新蝙蝠群算法是在蝙蝠群算法上增加了蝙蝠的栖息地选择策略和自适应补偿回声多普勒效应机制，这虽然在一定程度上弥补了先前蝙蝠群算法在收敛精度和收敛速度上面的缺陷，但是算法在搜索后期容易因为搜索范围减小，蝙蝠过于聚集，从而导致陷入局部最优解。通过分析蝙蝠的机械性行为可以知道，若蝙蝠进行机械性的行为，那么蝙蝠将在较为有限的空间中进行食物的搜寻，这时蝙蝠的位置变换情况就与粒子群算法十分相似，也会遇到跟粒子群算法十分相似的早熟现象。针对粒子群早熟的现象，文献[49]将二阶振荡环节加入到粒子群算法中并更新粒子群的速度公式，经过改进的算法加入到求解第 $k+1$ 次的迭代速度公式中，并且更新了第 $k-1$ 次粒子迭代的速度信息。这样不仅可以有效利用第 $k-1$ 代的速度信息加强粒子的学习能力，同时可以更好的丰富种群的多样性。

根据以上所述，本文在新蝙蝠群算法中同样引入了二阶振荡环节，将公式(4-9)加入二阶振荡环节之后的速度位置更新公式如下：

$$\begin{aligned} v_{ij}^{t+1} = {} & \omega v_{ij}^t + (\beta_1(p_{ij} - (1+\varepsilon_1)x_{ij}^t - \varepsilon_1 x_{ij}^{t-1}) \\ & + \beta_2(g - (1+\varepsilon_2)x_{ij}^t + \varepsilon_2 x_{ij}^{t-1})) * f_{ij}' \end{aligned} \tag{4-13}$$

上式中，$\beta_1 = c_1 r_1$，$\beta_2 = c_2 r_2$。ω 与 C_i 分别为惯性权重和学习因子。p_{ij} 表示

蝙蝠 i 的第 j 维上的历史最好位置， g 表示群体的最好位置。当最大迭代次数 $G_{\max} > 2t$ 时，且满足 $\varepsilon 1 < \frac{2\sqrt{\beta_1}-1}{\beta_1}$、$\varepsilon 2 < \frac{2\sqrt{\beta_2}-1}{\beta_2}$ 时，此时算法搜索最优解的能力较强。当迭代次数 $G_{\max} \le 2t$ 时，且满足 $\varepsilon 1 \ge \frac{2\sqrt{\beta_1}-1}{\beta_1}$、$\varepsilon 2 \ge \frac{2\sqrt{\beta_2}-1}{\beta_2}$，此时算法的收敛速度快。为了算法在拥有较好的全局搜索能力的同时可以快速收敛，对 $\varepsilon 1, \varepsilon 2$ 的取值如下式所示：

$$\varepsilon 1 = \begin{cases} \frac{2\sqrt{\beta_1}-1}{\beta_1} = \frac{2\sqrt{c_1 r_1}-1}{c_1 r_1}, t < G_{\max}/2 \\ \frac{(2\sqrt{\beta_1}-1)(1+r_3)}{\beta_1} = \frac{(2\sqrt{c_1 r_1}-1)(1+r_3)}{c_1 r_1}, t \ge G_{\max}/2 \end{cases} \tag{4-14}$$

$$\varepsilon 2 = \begin{cases} \frac{2\sqrt{\beta_2}-1}{\beta_2} = \frac{2\sqrt{c_2 r_2}-1}{c_2 r_2}, t < G_{\max}/2 \\ \frac{(2\sqrt{\beta_2}-1)(1+r_4)}{\beta_2} = \frac{(2\sqrt{c_2 r_2}-1)(1+r_4)}{c_2 r_2}, t \ge G_{\max}/2 \end{cases} \tag{4-15}$$

上述两式中，r_3, r_4 为取值在(0,1)的随机数。同时，为了避免种群多样性过低和陷入局部最优解，对 c_1, c_2 引入了动态参数机制：

$$c_1 = \beta_1' + \beta_1 \cos(\rho_2 \times t / G_{\max}) \tag{4-16}$$

$$c_2 = \beta_2' + \beta_2 \cos(\rho_2 \times t / G_{\max}) \tag{4-17}$$

其中 $\beta_1' = c_1' + \frac{\alpha_1}{2}$，$\beta_1 = \frac{\alpha_1}{2}$，$\rho_2 = 2\rho_1$，$\beta_2' = c_2' + \frac{\alpha_2}{2}$，$\beta_2 = -\frac{\alpha_2}{2}$，$\rho_2 = 2\rho_1$，$c_1', c_2'$ 是为定值的学习因子，$\alpha_1, \alpha_2, \rho_1$ 是为一个常量的权重因子。此外本文使用的缩放系数 ρ，多普勒补偿率 C_i，栖息地的选在概率 P，惯性权重 ω 也不是定值，而是根据蝙蝠位置的不同而进行动态变化的，其公式如下所示：

$$\rho = \frac{(\rho_{\max} - \rho_{\min})(G_{\max} - t)}{M} + \rho_{\min} \tag{4-18}$$

$$C_i = rand(0,1)(C_{\max} - C_{\min}) + C_{\min} \tag{4-19}$$

$$P = rand(0,1)(P_{\max} - P_{\min}) + P_{\min} \tag{4-20}$$

$$\omega = \frac{(\omega_{\max} - \omega_{\min})(G_{\max} - t)}{M} + \omega_{\min} \tag{4-21}$$

式中的 $\rho_{\max}, \rho_{\min}$ 分别代表缩放系数的最大值和最小值，缩放系数的值一般是

一个给定的数值；$C_{\max}, C_{\min}$ 分别代表多普勒补偿率的最大值和最小值，通常也为一个常数值；$P_{\max}, P_{\min}$ 分别代表蝙蝠选择栖息地概率的最大值和最小值；$\omega_{\max}, \omega_{\min}$ 分别代表惯性权重的最大值和最小值。

根据上文所述，在云制造平台环境中，会有许多不同的任务，调度的环境较为复杂，根据不同的需求所占用的资源会产生一定的差异。因此，学习因子、脉冲发射频率、响度和不同的权重因子都会对蝙蝠群产生不同的影响，使得算法在搜索的后期种群的多样性受到影响。在结合二阶振荡环节的基础上再将 DE 算法结合进来用以增强算法中蝙蝠群的多样性，扩大搜索范围。DE 算法的公式如下：

$$v_{ij}^{t+1} = x_{p1j}^{t} + \bar{\omega}(x_{p2j}^{t} - x_{p3j}^{t}) \tag{4-22}$$

$$u_{ij}^{t+1} = \begin{cases} v_{ij}^{t+1}, rand(0,1) \le cr \quad or \quad j = jr \\ x_{ij}^{t}, rand(0,1) > cr \quad or \quad j \ne jr \end{cases} \tag{4-23}$$

$$x_{i}^{t+1} = \begin{cases} u_{i}^{t+1}, f(u_{i}^{t+1}) < f(x_{i}^{t}) \\ x_{i}^{t}, otherwise \end{cases} \tag{4-24}$$

其中 $i \ne p1 \ne p2 \ne p3, p1$，$p2$，$p3$ 分别表示种群中的个体；$\bar{\omega}$ 为缩放因子，取值在[0,2]之间；jr 为粒子维度内的一个随机整数；函数 f 为目标函数；cr 为交叉概率。

新型蝙蝠群算法不仅引入了二阶振荡环节增强了种群的学习能力，丰富了种群在搜索后期多样性，还加入了 DE 算法不断进化择优，从而使得算法增强了全局搜索能力和有效的避免了粒子的早熟现象。具体的算法如下所示：

Input: N: the number of individuals

M: maximum number of iteration

ρ, ω, C， $P, \alpha, \gamma, f_{\min}, f_{\max}$， cr， ϖ

```
While(t<M)
  if(rand(0,1)<P)
    update x with(4-6)
  else
     update parameters with(4-7)-(4-10)(4-13)-(4-17)
  end if
  if(rand(0,1)<r)
     update parameters with(4-11)(4-12)
  end if
  compute objective value f(x)
  select p1,p2,p3
  update parameters with(4-22)
  update parameters with(4-23)
```

compute objective value f(x)
update parameters with(4-24)
if(rand(0,1)<A_i&&f(x_i)<f(x)
update parameters with(4-4)(4-5)
end if
t=t+1
end while

4.4 本章小结

本章节首先针对云制造平台的多目标调度问题进行了说明，并且根据第三章中提出的本文的调度模型，引入了新蝙蝠算法并且对新蝙蝠算法进行了介绍。经过对新蝙蝠算法的介绍分析，根据自己的认知对新蝙蝠算法进行了改进，改进的算法融入了传统的二阶震荡粒子群算法(MPSO)，差分进化算法(DE)，新蝙蝠群算法(NBA)中的一些元素。本章提出的改进算法不仅可以在求解多目标问题时可以扩大种群的搜索范围，增加种群的多样性，还可以避免过早的陷入局部最优解。

第五章 深度学习模型的引入及优化改进

5.1 引言

在上一章中主要对任务调度的模型进行了建立和分析，在一些不足之处进行了改进，从而使调度模型进行多目标调度时效果更优，实现了分配的任务在最短时间内实现较优分配，并同时保证 QoS 的属性值。但是在现实场景中，云制造平台可能出现任务量过大，供不应求的状态，在调度过程中不仅需要考虑平台在虚拟资源调度过程中的耗费问题，还需要考虑调度算法本身的时间消耗问题。从第 3 章最后的实验结果可以看出，虽然经过改进的新型蝙蝠群算法可以比其他算法更快更好的完成调度，但是其本质还是一个智能算法，在求解过程中仍然需要不断地进行迭代找出最优解，这个求解过程的时间复杂度仍比较高，其花销也会很大。在此基础上，本章节引入深度学习模型对任务的调度方案进行预测以辅助智能算法，从而减小调度算法运行的时间开销。由于深度学习模型的本身也很复杂，所以在学习过程中也会造成较大的花销，因此对 DBN 的深度学习模型进行改进优化对于整体的模型具有十分重要的意义。

5.2 DBN 模型及改进

深度信念网络（Deep Belief Network）是 Hinton 在 2006 年提出的一种建立在概率模型上的深度学习框架[50-52]，是神经网络的一种，既可以用于非监督学习，也可以用于监督学习。从非监督学习方面来看，其目的是尽可能的保留原始特征的特点，同时降低特征的维度；从监督学习方面来看，其目的在于尽可能降低分类错误率。DBN 模型深度学习的本质是特征学习，即如何获得更好的特征表达。DBN 是由若干个神经元构成，组成的元件是受限玻尔兹曼机（RBM），先通过逐层贪婪无监督训练，然后经过有监督的微调，最终实现模型的训练过程。

RBM 模型[53]是由 Hinton 和 Sejnowski 在 1986 年提出的一种生成式随机神经网络，该网络由一些可见单元和一些隐藏单元构成，可见变量和隐藏变量都是二元变量，即取值都为 0 或 1。其典型结构图如下所示：

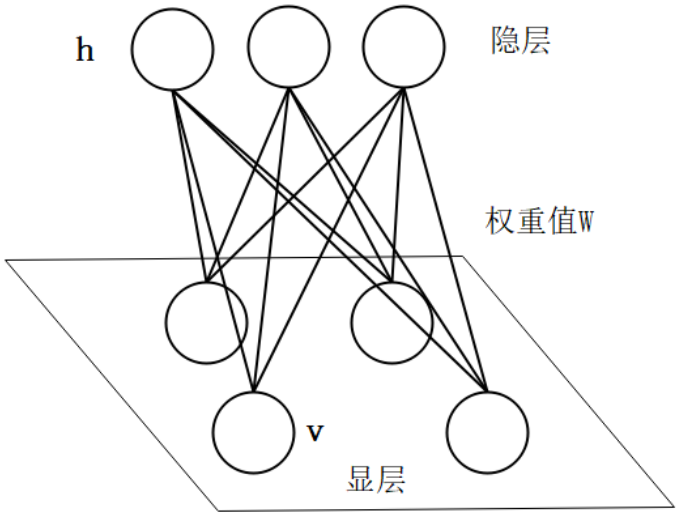

图 5-1 RBM 结构图

Figure 5-1. RBM structure diagram

如上图 5-1 所示，由可见单元组成的称为显示层，由隐藏单元组成的称为隐藏层。隐藏层和显示层之间的神经元进行全连接，而显示层和隐藏层本身之间无连接。由于这一性质，RBM 有如下特征：当显示层中的神经元的状态已经确定时，隐藏层中神经元的激活条件互相独立。同理，在隐藏层中的神经元状态已经确定时，显示层中的神经元的激活条件也相互独立。

图 5-1 中的神经单元可以服从如高斯分布、正态分布等任意的分布，因为本文中使用的神经元只有两种状态，所以本文中的神经元是二值分布。$v_i \in \{0,1\}, h_j \in \{0,1\}$，$v_i$ 和 h_j 分别表示显层神经元 i 和隐层神经元 j 的取值状态。

RBM 是一种基于能量的模型，其可见变量 v 和隐藏变量 h 的能量函数定义如下：

$$E_\theta(v,h) = -\sum_{i=1}^{m} b_i v_i - \sum_{j=1}^{n} c_j h_j - \sum_{i=1}^{m}\sum_{j=1}^{n} v_i W_{ij} h_j \tag{5-1}$$

上式中，θ 为 RBM 的参数{W,c,b}，W 为显层和隐层之间的边的权重，b 和 c 分别为显层和隐层的单元的偏置；m 表示显层的神经元个数，n 表示隐层的神经元个数。有了 v 和 h 的联合配置的能量之后，可以推出其他状态下的(v,h)的联合概率分布，如下式所示：

$$P_\theta(v,h) = \frac{e^{-E_\theta(v,h)}}{\sum_{v,h} e^{-E_\theta(v,h)}} \tag{5-2}$$

最大化数据 v 的边缘概率分布式 RBM 训练的一个重要的目标，其公式如下所

示：

$$P_\theta(v) = \frac{\sum_h e^{-E_\theta(v,h)}}{\sum_{v,h} e^{-E_\theta(v,h)}} \tag{5-3}$$

确定了训练的数据集之后，训练 RBM 意味着先确定θ，确保训练完成的模型可以很好的拟合给定的训练集。若训练集 Q 为给定训练集，那么含有多个训练数据集的 RBM 训练目标如下面的最大似然函数所示：

$$L_\theta = \prod_{t=1}^{Q} P_\theta(v^t) \tag{5-4}$$

根据 RBM 结构可以知道，当显层 v 中的所有神经元状态被给定，隐层中的第 j 个神经元被激活的概率如下式所示：

$$P_\theta = (h_j = 1 | v) = \frac{1}{1 + \exp(-c_j - \sum_{i=1}^{m} v_i W_{ij})} \tag{5-5}$$

同理可知，在隐层 h 中的所有神经元状态被给定时，显层中的第 i 个神经元被激活的概率如下式所示：

$$P_\theta(v_i = 1 | h) = \frac{1}{1 + \exp(-b_i - \sum_{j=1}^{n} W_{ij} h_j)} \tag{5-6}$$

因为在 RBM 的结构中，每一层之间其本身是没有连接的，因此我们可以得到下式：

$$P(h | v) = \prod_{j=1}^{n} P(h_j | v) \tag{5-7}$$

$$P(v \mid h) = \prod_{i=1}^{m} P(v_i \mid h) \tag{5-8}$$

上式为每一层的概率密度函数。

从本节开始的介绍中可知，RBM 的训练目的就是最大化似然函数（5-4），对公式（5-4）两边同时取对数得到下式：

$$\ln L_\theta = \sum_{t=1}^{T} \ln P_\theta(v^t) \tag{5-9}$$

根据对数的性质可以知道，假如要最大化式(5-4)，等同于最大化式(5-9)，此时需要找到一个θ用来拟合输入集，对上述公式使用梯度上升法，利用迭代来进行逼近，可得如下式子：

$$\theta = \theta + \eta \frac{\partial \ln L_\theta}{\partial \theta} \tag{5-10}$$

其中，η 是一个给定的整数值。

根据上述可以知道，学习 RBM 的任务时求出参数的值来拟合给定的训练数据，经典的近似算法由随机近似法，比例匹配法和对比散度（CD）等方法。CD 算法是由 Hinton 提出的 RBM 的一个快速学习算法，因为在 CD-K 算法中（K 表示采样的次数），当 K=1 是，只需要进行一步吉布斯采样就可以达到很好的拟合效果，所以 CD 算法中一般采用 CD-1 算法的形式来拟合各个参数的值。

CD 算法的基本思想为：首先使训练集 $v(0)=v$，然后进行 K 步的吉布斯采样，最后利用公式 $P(h(k\text{-}1)\,|\,v(k\text{-}1))$ 获取隐藏层在 $k-1$ 的状态 $h(k-1)$，同时利用公式 $P(v(k)\,|\,h(k\text{-}1))$ 获取重构的显层在 K 步的状态 v(K)。具体的 CD 算法如下所示：

Input: K, Sample s, n, m, RBM(W, b, c)
Output: ΔW, Δb, Δc
Initialize ΔW=0, Δb=0, Δc=0
for v
  v(0)=v;
  for t=0,1,…K-1
    for i=1,2,…n
      if rand(0,1)<P(h_i=1|v)
        h_i(t)=1;
      else
        h_i(t)=0;
      end if
    end for
    for j=1,2,…m
      if rand(0,1)<P(v_j=1|h)
        v_j(t+1)=1;
      else
        v_j(t+1)=0;
      end if
    end for
  for i=1,2,…n
    for j=1,2,…m
      $\Delta W_{ij}=\Delta W_{ij}+[P(h_i=1|v^{(0)})v_j^{(0)}-P(h_i=1|v^{(K)})v_i^{(K)}]$;
      $\Delta b_j=\Delta b_j+[v_j^{(0)}-vj^{(K)}]$;
      $\Delta c_i=\Delta c_i+[P(h_i=1|v^{(0)})-P(h_i=1|v^{(K)})]$;
    end for
  end for

end for
end

在 CD 算法中得到的 ΔW,Δb,Δc 用来对 RBM 进行训练，RBM 的训练就是用 CD 算法中得到的结果不断地来调整显层和隐层之间的参数。RBM 算法如下所示：

Initialize W, b,c, learning rate η, iterations m
  for t=1,2,…m
    using the CD algorithm to obtain ΔW, Δb, Δc;
    update: W=W+ηΔW, b=b+ηΔb, c=c+ηΔc;
  end for
end

上述算法中 η 是 RBM 训练过程中的学习率。当学习率 η 较大时，收敛速度更快，但可能会引起算法的不稳定，而当 η 较小时，虽然可以避免不稳定的情况的出现，但是收敛速度较慢。因此，对 RBM 学习率的改进时尤为重要的。RBM 学习率的改进算法中参数的更新公式如下：

$$\Delta\omega_{i,j} = \Delta\omega_{i,j} + \eta[c_1 - c_2] \tag{5-11}$$

$$\Delta a_j = \Delta a_j + \eta[v_j^{(0)} - v_j^k] \tag{5-12}$$

$$\Delta b_i = \Delta b_i + \eta[c_1 / v_j^{(o)} - c_2 / v_j^{(k)}] \tag{5-13}$$

$$c_1 = P(h_i = 1 | v^{(0)}) v_j^{(0)} \tag{5-14}$$

$$c_2 = P(h_i = 1 | v^{(k)}) v_j^{(k)} \tag{5-15}$$

其中学习率 η 为定值，在训练过程中无法调整学习率导致参数的更新能力较弱。结合神经网络学习率改进算法，有如下公式：

$$\eta(t+1) = \begin{cases} \alpha\eta(t) & ,e(t) < e(t-1) \\ \beta\eta(t) & ,e(t) > e(t-1) \end{cases} \tag{5-16}$$

α 为(1,2)之间的定值，β 为(0,1)之间的定值,t 为当前迭代次数，e 为训练重构误差。当 $e(t)<e(t-1)$时以固定的速率增加学习率以增加模型的训练速度，当 $e(t)>e(t-1)$时以一定的速率减小学习率以确保模型的训练精度。实验结果表明虽然学习率能够根据实际情况进行动态调整从而实现加快误差的收敛程度，但是仍然很难满足寻优要求。因此在此基础上，根据 RBM 的重建误差进行动态的调整学习率，公式如下：

$$\eta(t+1) = \begin{cases} \alpha\sqrt{e(t-1)+e(t)}\eta(t)/t & ,e(t) < e(t-1) \\ \beta\eta(t)/t & ,e(t) > e(t-1) \end{cases} \tag{5-17}$$

其中 $e(t)$为当前误差，$e(t-1)$为上一次训练误差，学习率的调整过如下：

（1）当 $e(t)<e(t\text{-}1)$，则说明随着 t 的增加误差呈下降趋势，此时适当的增加 η 可以提高误差的下降速度，由于 $e(t)$和 $e(t\text{-}1)$随着训练次数呈现下降的趋势而 t 不断的上升，因此 η 的增加率呈下降的趋势，这样就避免因 η 增加过快而越过误差的极小值。

（2）当 $e(t)>e(t\text{-}1)$则说明误差很有可能越过了极小值此时减小 η 以使得误差继续减小直至收敛。

在训练过程中为了防止过拟合使用权重衰减：

$$\Delta\omega_{i,j}=\Delta\omega_{i,j}+\eta[c_1\text{-}c_2\text{-}\delta\Delta\omega_{i,j}] \tag{5-18}$$

其中 δ 为权重衰减系数位(0,1)之间的值。传统方法中 δ 为一个定值，导致模型的泛化能力不强。采用自适应的方法对 δ 进行改进，如以下公式所示：

$$\delta=\delta/\sqrt{e(t)+t} \tag{5-19}$$

改模型中的学习率的改变不再是根据固定的速率来进行增加的，而是根据模型训练后的误差进行动态调整，同时因为在模型训练时的重构误差直接参与其中，使学习率的改变具有一定的自适应性。

当 DBN 模型中的 RBM 训练完成之后使用标签数据集对模型进行整体的微调以达到模型的整体最优，在这一阶段主要使用梯度下降算法对模型的权重进行更新：

$$\omega_{i,j}=\omega_{i,j}\text{-}\eta\nabla E(\omega) \tag{5-20}$$

其中 η 为反向调整的学习率，$\nabla E(\omega)$ 为误差梯度。

为克服传统算法中学习率手动调整的缺陷，Adagrad 和 AdaDec 算法被大多数采用。这两种方法中学习率 η 都呈非线性下降的趋势，虽然其涉及到梯度下降的速率和学习率之间的关系，但是没有考虑误差减小时的因素。因此如果为了达到收敛的目的而减小 η 可能会导致训练误差靠近收敛误差的速度减慢，从而需要增加训练次数才能收敛。从模型训练过程分析可知当模型误差不断减小时 η 可以适当的增加以加快模型的收敛速度，但是 η 的增加速率应该不断的减小。因为随着迭代次数的增加误差越来越小，距离极小值越来越近所以 η 也要减小以免越过极小值。结合上面的学习率改进公式和分析，提出的改进算法 IRBM 公式如下：

$$\eta(t+1)=\begin{cases}\eta(t)\left(1+\dfrac{\sqrt{e(t)}}{t}\right) & e(t)<e(t\text{-}1)\\ \dfrac{\eta(t)}{\sqrt{1+G(t)}} & e(t)>e(t\text{-}1)\end{cases} \tag{5-21}$$

$$G(t)=\zeta E(g(t\text{-}1)^2)+g(t)^2 \tag{5-22}$$

其中 e(t)为误差值，ζ 为一个（0,1）之间的定值，g(t-1)为上一次迭代的梯度，$E(g(t-1)^2)$为上一次迭代的梯度平方和的均值。g(t)为当前的梯度。学习率调整过程如下：

（1）当 e(t)<e(t-1)说明当前误差值小于上一次误差，误差随着迭代的增加在不断的减小。由于随着 t 的增加，e(t)呈下降趋势所以 ε 增长速率需要减小。

（2）当 e(t)>e(t-1)时说明当前的学习率太大导致梯度下降太快，因此应当减小学习率。

在误差下降的阶段适当的增加学习率能够增加梯度下降的速度从而快速减小误差到一个合理的范围内，当误差越过最小值时适当的减小学习率能够保证在合理的范围内对误差进行精调。学习率下降阶段中在上一回合的学习率的基础上利用当前的梯度去自适应调节学习率的大小，这样设计的学习率更能准确描述模型的运行状态，调节得到的学习率也更合理，因此能加快模型的收敛速度，同时也能减小历史梯度参与计算从而减小计算量。

5.3 训练次数的控制

传统的训练次数通常根据经验进行设置，为了保证模型在训练时能够收敛，需要反复的调整。根据提出的学习率调整方法，在实际训练中结合误差和学习率来控制模型的训练次数。当η减小到一定的范围后，模型的误差下降幅度极小此时误差已经很接近极小值，整个模型的误差范围在可接受范围内所以此时再增加 t 已经没有意义则停止训练。另外考虑到 t 可能还没有下降到设置的范围时模型已经收敛，所以训练误差也是控制模型训练次数的因素之一。当本文提出的训练模型（IGD）满足下面公式的两个条件时就不断的进行迭代：

$$\begin{cases} \eta(t) > \partial \\ \left|\beta(t-n) - \beta(t-n-1)\right| \le \alpha(n = 1, 2..) \end{cases} \tag{5-23}$$

上式中∂为一个设置好的的阈值；β为训练误差；n 为连续统计误差的次数；α为一个误差的波动范围。由上面的公式可知当$\eta > \partial$，连续$e > \alpha$时，模型进行迭代训练,当$\eta < \partial$时误差下降幅度小此时停止训练，当连续几次的$e < \alpha$时，基本认为模型收敛也停止迭代。

DBN 的训练采用的是逐层无监督贪婪的预训练方法，由低层到高层及使用前向传播来训练每一层的参数，前向训练就是分层对 RBM 进行训练以后去对应的权重等参数，预训练完成后再使用标签数据集由高层到低层的反向传播算法来微调各层的参数。本文提出的改进算法 IRBM 具体过程为：

Input：trainset, hidden unit numbers m

Learning rate η, reconstruction error err
Statistical continuous error N, μ
Output: ω, a, b

---

initialize, ω,a, b
while $\eta(t) > \partial$ &&n<N do
  for all hidden units i do
   compute $P(h_{1i}=1|v_1)$
   sample $h_{1j} \in \{0,1\}$ from $P(h_{1i}=1|v_1)$
  end for
  for all visible units j do
   compute $P(v_{2j}=1|h_1)$
   sample $v_{2j} \in \{0,1\}$ from $P(v_{2j}=1|h_1)$
  end for
  for all hidden units i do
   compute $P(h_{2i}=1|v_2)$
  end for
  compute reconstruction error e(t)
  update $\eta(t)$ with (4-17)
   update ω,a,b use (4-11)-(4-14)and(4-18)-(4-19)
  if |e(t)-err|≤μ
    n++
   else
    n=0
   end if
  err ← e(t)
end while

当 RBM 训练完成后在使用带标签的数据集对模型进行反向训练，反向训练的目的就是对损失函数求偏导以调整权重和偏置使得误差最小。IGD 训练具体过程如下：

（1）初始化参数；

（2）while ε>θ&&n<M，及计算当前 e(t)和梯度；

（3）根据式 $\omega_{i,j} = \omega_{i,j} - \eta \nabla E(\omega)$ 更新参数 ω，a 和 b；

（4）根据公式（5-21），（5-22）更新参数 ε；

（5）若|e(t)-err|≤μ，n 加一，否则 n=0。

5.4 本章小结

本章节主要介绍了 DBN 模型的原理并对 DBN 模型训练过程进行了部分改进。

本文的模型无论是在前向训练 RBM，还是在反向调整学习率和误差的时候都是紧密相关的，针对这一个特点，本章节提出了用自适应学习率来调整算法用来加快模型训练时的误差减小速度，并提出方法有效减少模型训练过程中训练次数的问题，从而达到加快训练速度的目的。

第六章 实验的仿真及结果分析

6.1 引言

本章节是为了验证本文提出的模型和算法在云制造服务平台的调度情况下的有效性，根据本文的内容将验证情况分为了两部分，第一部分是针对改进的新蝙蝠算法来进行的；第二部分是针对深度学习模型的改进情况以及利用改进的新蝙蝠算法来获得深度学习模型的训练数据后对深度学习模型进行训练，在模型训练完毕后用来预测任务的分配，然后跟传统的调度算法进行对比。

6.2 改进的新蝙蝠算法仿真结果对比

实验在 Matlab 上进行以下两方面的测试：

(1) 新改进的蝙蝠算法性能对比。

(2) 调度模型与算法的实例应用。为了测试改进的新蝙蝠算法（ONBA）的性能,我们将其与二阶震荡粒子群算法(MPSO)，差分进化算法(DE)，新蝙蝠群算法(NBA)相对比。实验通过模拟多种情况下的云制造批量生产的情况，展示了改进的新蝙蝠算法的较好优化效果以及快速收敛性，实例应用的测试中我们以转接线的生产为例。

实验中 QoS 属性的值为随机生成度量范围为[1,50]。实验过程中权重因子分别取值为 0.3，0.3，0.2，0.2。每个任务在各个工序上加工时间是随机生成的数值，取值区间为[0,60]，单位分钟，由矩阵 $Time_{m*n}$ 进行存储。实验重复进行多次并取平均值作为实验数据，以避免偶然因素对实验的影响。实验中三种算法的迭代数统一设置为 100，种群规模设置为 50。算法的惯性因子的值设置为 0.8。

为了测试算法在求解 QoS 约束的表现情况，本文随机的选取了任务数为 50，和 100 两种情时的 QoS 值结果如下：

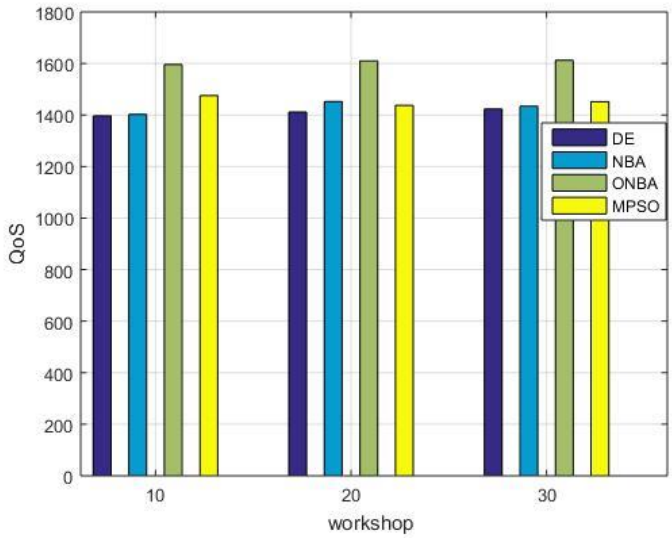

图 6-1 加工工序 50，加工商分别为 10,20,30

Figure 6-1. Processing step 50, the processors are 10, 20, 30 respectively

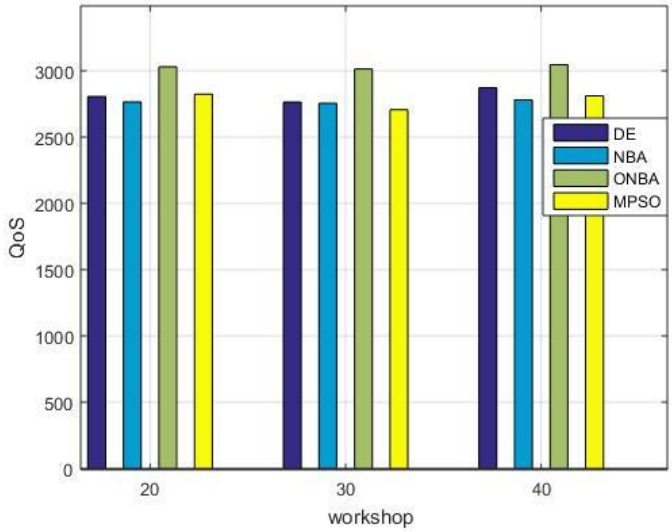

图 6-2 加工工序 100，加工商分别为 20,30,40

Figure 6-2. Processing step 100, the processors are 20, 30, 40 respectively

本文提出的 QoS 由于度量范围在[1,50]其对应的每一个属性值越大则表示加工商在其对应的方面越优，因此综合 QoS 属性值和用户对于属性的权重其值越大越符合用户的需求。从图中可以看出本文提出改进的新型蝙蝠群算法在求解 QoS 值时要优于其他的算法。

为了测试算法在不同云制造任务数和不同工序的搭配下的表现情况，同时也为了体现考虑物流情况下算法调度所得到的总的生产时间的对比，我们将实验分为生产工序分别是 10，20，30 和 40 这四种情况，并分别展示了四种算法在这四

种情况下的对比情况。(n 代表任务数，m 代表工序数)单位为：103 分钟。

表 6-1 算法对比情况

Table 6-1. The algorithm comparison

		MPSO		DE		ONBA		NBA	
n	m	T	Q	T	Q	T	Q	T	Q
30	10	1.5650	1.51	1.5680	1.35	1.4360	1.53	1.5700	1.40
40	10	1.8593	1.43	1.8213	1.44	1.7843	1.51	1.8463	1.42
60	10	2.4942	1.52	2.5332	1.46	2.4433	1.52	2.5044	1.47
70	10	2.9610	1.51	2.9050	1.47	2.8530	1.52	2.9800	1.45
80	10	3.1596	1.50	3.1766	1.49	2.9881	1.51	3.2756	1.49
90	10	3.4848	1.49	3.5248	1.43	3.4458	1.60	3.6198	1.51
50	20	2.9041	2.75	2.8834	2.80	2.8551	3.10	2.9013	2.71
60	20	3.2790	2.73	3.2450	2.72	3.1940	2.84	3.2700	2.83
70	20	3.7010	2.81	3.6580	2.76	3.6190	2.84	3.7130	2.80
80	20	4.0149	2.78	4.0079	2.77	3.9831	2.80	4.1081	2.77
90	20	4.4906	2.78	4.4156	2.79	4.3866	3.01	4.4006	2.81
50	30	3.5263	3.01	3.5763	2.89	3.5003	3.10	3.5923	2.86
60	30	4.0309	2.76	4.0729	2.85	3.9947	2.94	4.0949	2.91
70	30	4.4917	2.90	4.5197	2.91	4.3837	3.01	4.5867	2.91
80	30	4.7104	3.00	4.6944	2.94	4.6444	3.11	4.7274	3.11
90	30	5.0737	2.97	5.1157	2.97	5.0217	3.11	5.0777	2.97
100	30	5.4831	2.71	5.5141	2.80	5.4417	3.15	5.5971	2.82
110	30	5.9173	2.79	5.9114	2.98	5.8824	2.98	5.9973	2.85
120	30	6.1458	3.10	6.2108	3.10	6.1248	3.10	6.2368	3.02
60	40	4.6756	3.26	4.6286	3.34	4.6111	3.50	4.6591	2.89
80	40	5.4536	2.87	5.4176	3.21	5.4086	3.41	5.5016	2.97
100	40	6.0663	3.14	6.1003	3.20	6.0043	3.21	6.0943	2.99
110	40	6.6302	2.97	6.6220	2.86	6.6120	3.12	6.6110	3.11
120	40	6.8880	3.12	6.8640	3.26	6.8610	3.38	6.9090	2.97
130	40	7.3252	2.94	7.2592	2.83	7.1095	3.49	7.3202	3.28

从表 6-1 中可以看出，ONBA 在保证 QoS 值最大的前提下，对于不同任务数和不同工序数的云制造批量生产所产生的时间开销绝大多数情况下均小于 MPSO

和 DE，NBA 算法。在本文的模型中，还加入了物流因素，所以 ONBA 算法所要求解的问题的复杂度更高。从表 6-1 可以看出，除了 ONBA 算法之外其他的三种算法中当模型迭代完成后它们的最优解相差不大，这是由于这些单一的算法虽然有着各自的优点但在问题复杂度较高时，很难发挥出各自的优点，不能完全展示出其在大规模劣质解包围下搜索最优解的能力。ONBA 算法则是结合上面的算法的优点使得搜索能力强于上述的三种算法的搜索能力。

ONBA 算法不仅可以保证具有较快的收敛速度，还可以比较有效的避免陷入局部最优解。以下的三个图分别是云制造任务数为 50，70，130 的情况下 MPSO，DE，ONBA，NBA 对应求得的总的任务执行时间随迭代次数增加的趋势图。

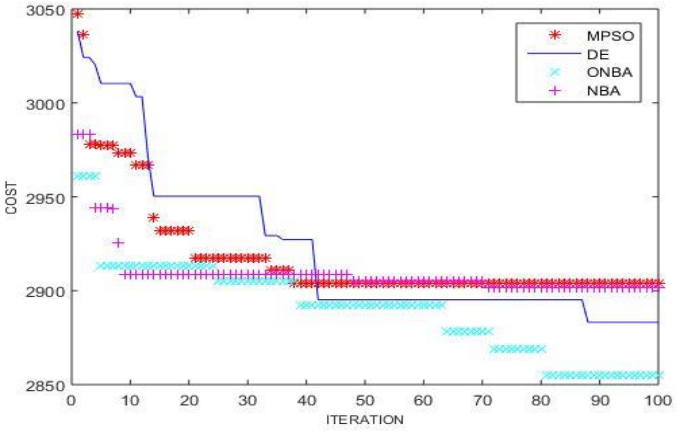

图 6-3 任务数为 50 时

Figure 6-3. When the number of tasks is 50

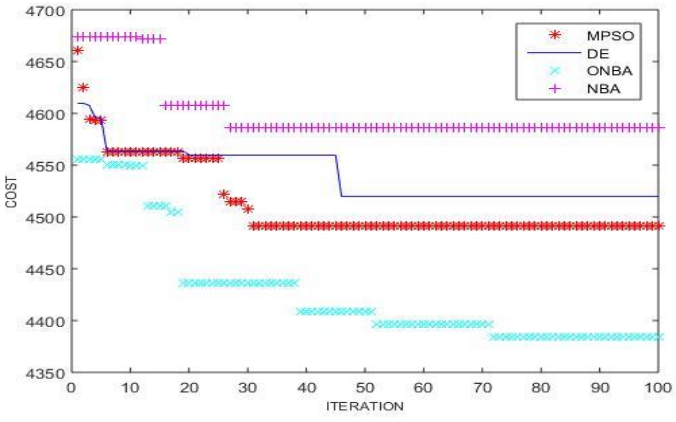

图 6-4 任务数为 70 时

Figure 6-4. When the number of tasks is 70

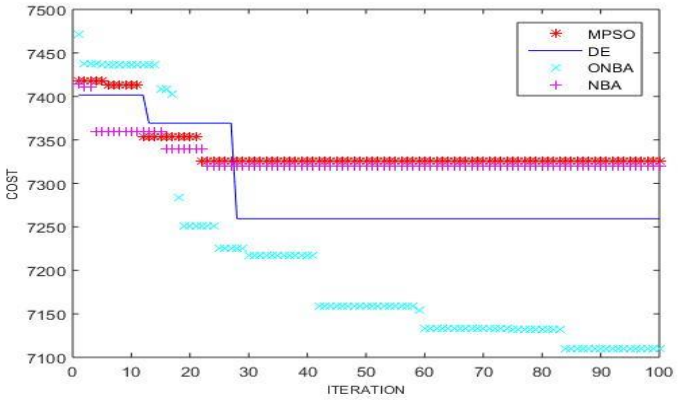

图 6-5　任务数为 130 时

Figure 6-5. When the number of tasks is 130

从上述三幅图中可以看出，三种算法中，ONBA 展现了较强的快速收敛能力和避免陷入局部最优解的能力。ONBA 算法在保持了各自种群的优良性的前提下，不断和其他种群进行信息互换，不但丰富了种群的多样性，而且同时提高了算法最优解的质量。ONBA 算法在 NBA 算法的基础上加入了振荡环节，有效的增强了算法的全局搜索能力，结合 DE 算法中父代粒子杂交产生子代粒子的过程有效的丰富了种群的多样性。从上述图中可以看出，当与 MPSO 和 NBA，DE 这三种算法相比较时，随着解空间范围的增大，在继续保持解的优势情况下，ONBA 的收敛速度逐渐提高。在迭代过程中出现停滞时，ONBA 可以及时丰富种群的多样性和提升种群的优良性，避免陷入局部最优解，同时这些优化也是使得算法在保持解质量的前提下使迭代速度进一步提高的原因所在。并且 ONBA 由于在局部搜索的过程中，不断扩展了最优解集合的范围，所以选取的最优解也比其他三种算法选取的最优解的质量更好。这是因为 ONBA 算法结合 NBA 算法的局部搜索能力强，MPSO 的全局搜索能力强，DE 算法中变异和粒子的杂交等优点使得 ONBA 算法在搜索能力和防止粒子早熟上表现的更优。

6.3　改进的深度学习模型的仿真实验结果

本节的数据集来自第四章改进的新蝙蝠算法仿真得到的数据，然后使用改进的 DBN(IDBN)方法进行训练数据特征的提取，在这采用 RBF 的方法进行预测。

为了评估模型的预测效果，在本文中使用平均绝对误差(MAE)，平均相对误差(MRE)和均方根误差(RMSE)来衡量本文的模型的预测值和真实值之间的差距，上述三种误差衡量方式定义如下：

$$MAE = \frac{1}{n}\sum_{i=1}^{n} | f_i - \hat{f}_i | \tag{6-1}$$

$$MRE = \frac{1}{n}\sum_{i=1}^{n} \frac{| f_i - \hat{f}_i |}{f_i} \tag{6-2}$$

$$RMSE = \left[\frac{1}{n}\sum_{i=1}^{n} (| f_i - \hat{f}_i |)^2\right] \tag{6-3}$$

上式中 f_i 为数据的真实值， $\hat{f}_i$ 为数据的预测值。

该实验主要分为两部分，首先第一部分为 RBM 训练的改进对比，在 RBM 的训练中，设定的初始学习率为 1，误差波动范围为 0.1，训练次数和重构误差关系如下图所示：

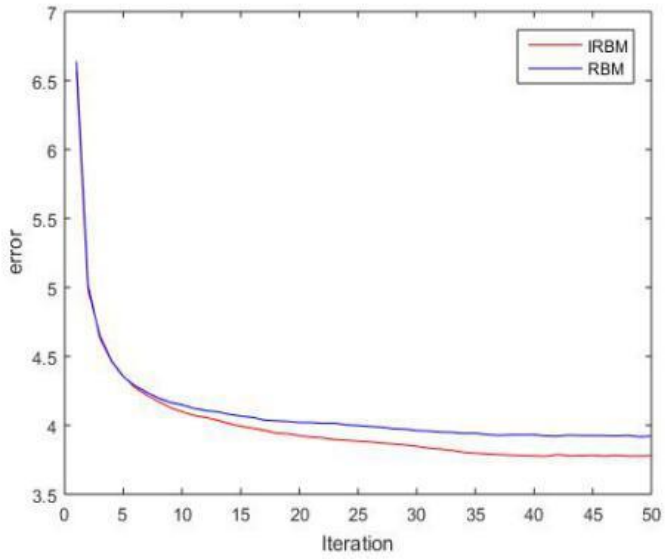

图 6-6 改进 RBM 模型训练对比图

Figure 6-6. Improved RBM model training comparison chart

由上图可以看出，改进的 IRBM 训练不仅在模型的训练收敛速度上比传统的 RBM 快，而且在收敛精度上也比传统的 RBM 训练方法高。从图中可以看出，虽然当收敛次数达到 35 次左右，误差的波动范围就开始变得很小，但是仍然以一个微小的速度在下降，说明本文提出的改进方法对 RBM 模型的训练确实起到了加速的效果。

反向调整时训练误差对比如下图所示：

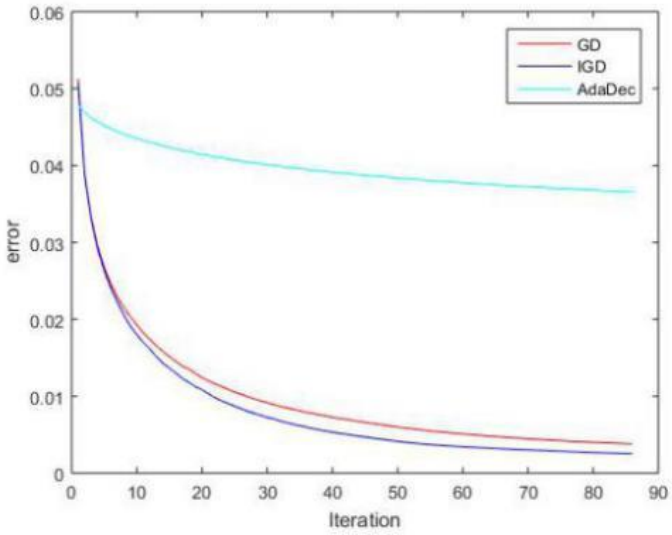

图 6-7　反向调整时训练误差对比

Figure 6-7. Comparison of training errors in reverse adjustment

如图 6-7 所示，与传统的 GD 算法和 AdaDec 算法对比，可见 IGD 算法的收敛速度更快并且拥有更好的效果，而 AdaDec 算法的误差减小速度是最慢的，说明学习率减小对模型的收敛速度有着很大的影响。同时，进行反向微调时，我们整个模型设置的微调迭代次数初始值为 100，从上图可以看出，虽然最大的训练次数为 100 次，但是实际在 90 次不到的时候训练就已经停止了，并且整个模型的训练在后期愈加趋于稳定，说明误差的波动越来越小。但是在收敛的后期，虽然整个模型趋于稳定，但是其仍然处于下降的趋势，上述两点可以说明本文提出的控制整个模型的训练次数的方法是有效的。

接下来本文用改进的深度学习模型对调度算法模型进行了结果的预测，预测的准确度主要和传统的神经网络预测模型进行对比：

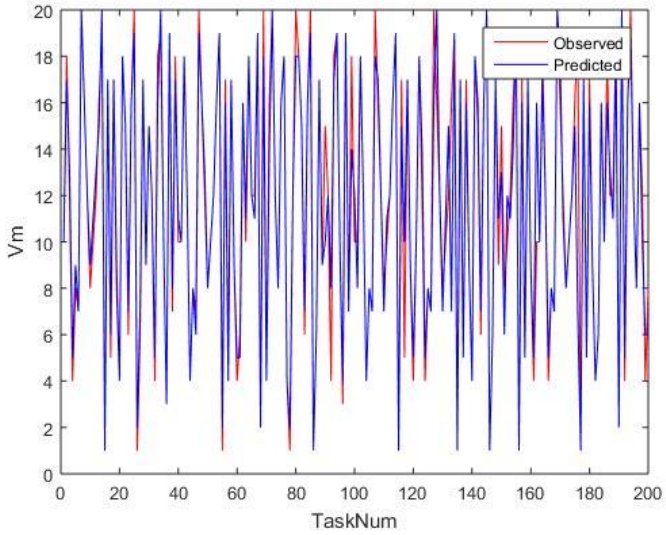

图 6-8 加工商数为 20，任务数为 200 时 IDBN 预测

Figure 6-8. IDBN forecast with 20 processors and 200 tasks

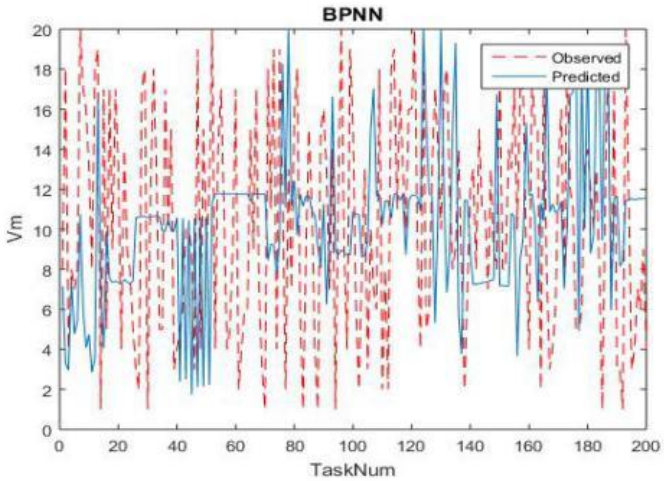

图 6-9 加工商数为 20，任务数为 200 时 BP 预测

Figure 6-9. BP prediction with 20 processors and 200 tasks

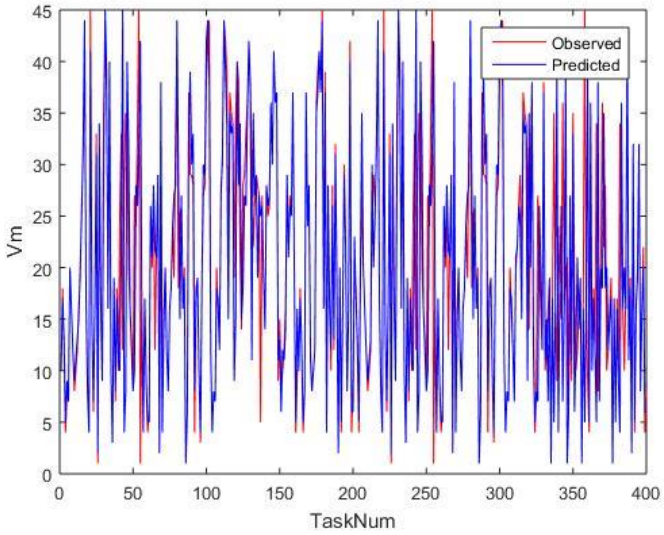

图 6-10　加工商数为 45，任务数为 400 时 IDBN 预测

Figure 6-10. IDBN forecast with 45 processors and 400 tasks

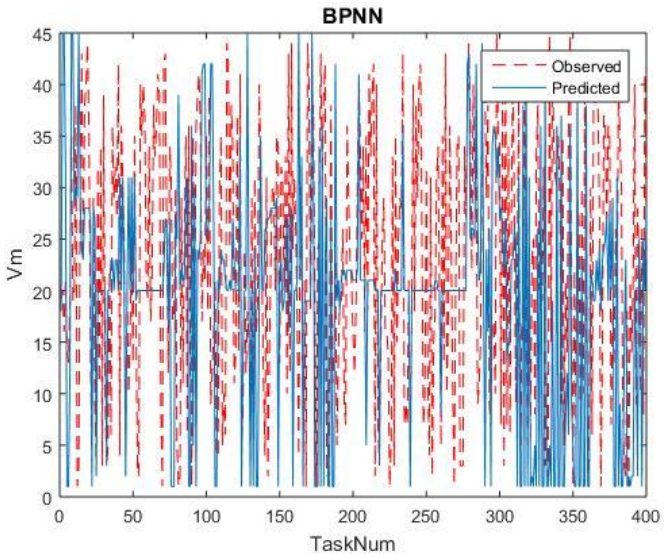

图 6-11　加工商数为 45，任务数为 400 时 BP 预测

Figure 6-11. BP prediction with 45 processors and 400 tasks

其中图 6-8 和图 6-10 为改进的深度学习模型的预测结果，图 6-9 和图 6-11 为 BP 神经网络的预测结果。从图中可以明显看出改进的深度学习模型的预测效果要比传统的 BP 神经网络的预测结果要好的多。

表 6-2　ONBA 与 IDBN 耗时对比

Table 6-2. Time-consuming comparison between ONBA and IDBN

m	n	ONBA	IDBN
200	20	5.202336	0.468154

200	45	6.964589	0.685310
300	20	7.023486	0.631514
300	45	9.363982	0.492413
400	20	8.783663	0.814632
400	45	12.107095	0.899639

从上述的预测图中我们可以看出使用深度学习模型预测的结果和使用传统调度算法得出的结果的偏差很小，从这方面可以说明深度学习模型较为准确的预测到了传统的调度算法产生的结果。从表 6-2 中我们可以知道，使用深度学习模型预测所耗费的时间要大大小于使用传统调度算法所耗费的时间。结合图和表看来，使用深度学习模型预测可以有效的提高云制造平台对调度任务的时效。虽然在任务数和加工商数增加的同时，深度学习模型的预测时间也在增长，但是其增长的时间跟传统的调度算法所耗费的时间相比几乎可以忽略不计。这种现象的出现主要是因为任务数量和加工商数量的增加会给云制造平台进行服务匹配时带来更多选择的可能性，数据之间组合的难度也会大大增加，所以会导致预测时间的变长。

6.4 本章小结

本章主要对第四章和第五章介绍的本文所使用到的算法进行了仿真实验。从实验结果可以看到本文改进的新蝙蝠算法在 QoS 约束上的效果要比二阶震荡粒子群算法(MPSO)、差分进化算法(DE)和新蝙蝠群算法(NBA)都要好。同时，改进的新蝙蝠算法在跳出局部最优解、防止粒子早熟和收敛速度上都明显要优于其他三个算法，说明本文改进的新蝙蝠算法效果是十分不错的。最后对本文提出的深度学习模型对云制造调度算法中的任务调度结果预测进行了验证，上述的实验可以说明，当深度学习的模型训练完成之后，其对调度的预测速度不仅显著快于传统的调度算法，还可以尽可能保证解的质量。因此可以证明本文的研究是具有一定的实际应用价值的。

第七章 总结与展望

7.1 总结

制造业是中国国民经济的主体，是国家的立国之本，强国之基。从十八世纪的工业文明以来，世界各个国家的发展史和人民的奋斗史可以说明，强大的制造业是一个国家和民族强盛的根本。强大的制造业实力是提升国际竞争力和综合国力的基础，因此打造具有强大实力的制造业是中华民族复兴的重要一步。新中国成立以来，尤其是在改革开放之后，我国的制造业飞速发展，逐渐建立起来了一个完整的制造业体系，国家也在大力推动工业化和现代化进程。但是与世界上拥有强大制造业实力的强国相比，我们国家的制造业水平在自主创新能力、资源的利用率、产业的结构化水平和信息化程度等方面差距还是十分明显的，因此对制造业的转型升级和进一步发展提出了更大的要求。

在这一背景之下，云制造的出现让我们看到了更大的希望，云制造模式为制造业信息化提供了一种崭新的理念与模式。云制造模式融合了云计算、物联网和大数据等新兴技术，是一种面向需求服务的高效率、低消耗模式，云制造模式不仅可以提高生产效率，更大可能的减少制造业的开销，更加合理的利用资源，更可以为节能减排出一份力，而且云制造模式更加面向用户需求，给了客户更多的选择和个性化定制，是未来我国发展制造业的一种重要的制造模式。

本文在这一大背景下对云制造平台的调度模式进行了研究，首先对云制造进行了一些基础的介绍，并对一些知名的云制造案例进行了举例分析，可以初步了解云制造模式的优点。然后本文对国内外的云制造研究现状进行了分析，并详细介绍了云制造领域目前的研究现状，又介绍了云制造领域需要用到的一些重要技术，再对云制造模型中使用到的这些重要技术的详细情况进行了说明介绍。

从第三章开始，本文首先提出了一个云制造的模型，提出的模型考虑了基于 QoS 的服务资源匹配和总任务执行时间最短两个方面。由于云制造服务的特点是面向服务需求，所以本文的模型考虑 QoS 的服务资源匹配可以更好的满足客户的需求，在此基础上使总任务的执行时间最短可以缩短任务的完成时间，在最大限度上控制云制造平台的成本和提高客户的满意度。第四章文章首先介绍了新蝙蝠算法，然后根据本文的模型需要对新蝙蝠算法进行了优化改进，在改进的新蝙蝠

算法中引入了动态因子，并加入了二阶振荡环节和差分进化。第五章文章引入了深度学习模型并深度学习模型进行了介绍，对深度学习模型的学习率和学习次数进行改进，使得深度学习模型可以更好更快的收敛，然后通过改进的新蝙蝠算法获得云制造的调度数据集对深度学习模型进行训练，最后使用训练完成的深度学习模型对调度结果进行预测并与传统的调度算法在时间上进行比较。第六章是本文的仿真实验部分，其中第一部分先对改进的新蝙蝠算法进行了仿真，并与改进前的算法、DE 算法、MPSO 算法进行了对比，证明改进的新蝙蝠算法在 QoS 约束和快速收敛上都有更好的效果。第二部分验证了改进后的深度学习模型在训练速度的模型的收敛速度上都有比较明显的提升，然后利用训练完成的模型对调度结果的预测与改进的新蝙蝠算法调度结果进行对比，证明了本文提出的方法是具有实际的应用价值的。

7.2 下一步工作与展望

本文虽然对云制造平台的调度做了一些优化，但是还存在很多的不足之处，本文的云制造调度模型还可以进行更进一步的完善，在云制造平台的实际应用中影响任务完成的因素还有很多，下一步完善模型时可以将经济因素、质量因素以及一些会对调度产生干扰的因素都加入进来，使得模型更具有实际应用意义。本文提出的改进的新蝙蝠算法在求解多目标问题时耗费的时间虽然对比其他算法有所进步，但是耗时仍旧较长，这方面还可以做出更多的改进。本文将深度学习模型引入了进来，并取得了较好的结果，但是仍需要对其他的调度情况进行更深一步的研究分析，进一步的去使深度学习模型适应其他不同的调度环境和调度算法，增强深度学习模型在云制造调度环境中的实用性。

参考文献

[1] Li D X, Xu E L, Ling L. Industry 4.0: state of the art and future trends[J]. International Journal of Production Research, 2018, 56(8):1-22.

[2] 周济. 智能制造——“中国制造 2025”的主攻方向[J]. 中国机械工程, 2015, 26(17):2273-2284.

[3] 李伯虎, 张霖, 王时龙,等. 云制造——面向服务的网络化制造新模式[J]. 计算机集成制造系统, 2010, 16(1):1-7.

[4] 周龙飞, 张霖, 刘永奎. 云制造调度问题研究综述[J]. 计算机集成制造系统, 2017(6).

[5] Cheng Y, Zhao D, Hu A R, et al. Multi-view Models for Cost Constitution of Cloud Service in Cloud Manufacturing System[M]// Advances in Computer Science and Education Applications. 2011.

[6] Xu X . From cloud computing to cloud manufacturing[J]. Robotics and Computer-Integrated Manufacturing, 2012, 28(1):75-86.

[7] Tao F , Zuo Y , Xu L D , et al. IoT-Based Intelligent Perception and Access of Manufacturing Resource Toward Cloud Manufacturing[J]. IEEE Transactions on Industrial Informatics, 2014, 10(2):1547-1557.

[8] Wang X V , Xu X W . An interoperable solution for Cloud manufacturing[J]. Robotics and Computer-Integrated Manufacturing, 2013, 29(4):232-247.

[9] Tao F , Cheng J , Cheng Y , et al. SDMSim: A manufacturing service supply–demand matching simulator under cloud environment[J]. Robotics & Computer Integrated Manufacturing, 2017, 45(6):34-46.

[10] Xu Z , He J , Chen Z . Design and actualization of IoT-based intelligent logistics system[C]// IEEE International Conference on Industrial Engineering & Engineering Management. IEEE, 2014.

[11] Lu Y , Xu X , Xu J . Development of a Hybrid Manufacturing Cloud[J]. Journal of Manufacturing Systems, 2014, 33(4):551-566.

[12] Liu N , Li X . A Resource Virtualization Mechanism for Cloud Manufacturing Systems[C]// International IFIP Working Conference on Enterprise Interoperability. Springer Berlin Heidelberg, 2012.

[13] Laili Y , Tao F , Lin Zhang⋯. A study of optimal allocation of computing resources in cloud manufacturing systems[J]. International Journal of Advanced Manufacturing Technology, 2012, 63(5-8):671-690.

[14] Laili Y , Tao F , Zhang L , et al. A Ranking Chaos Algorithm for dual scheduling of cloud service and computing resource in private cloud[J]. Computers in Industry, 2013, 64(4):448–463.

[15] Zhang L , Guo H , Tao F , et al. Flexible management of resource service composition in cloud manufacturing[C]// IEEE International Conference on Industrial Engineering & Engineering Management. IEEE, 2010.

[16] Li F , Zhang L , Liu Y , et al. A clustering network-based approach to service composition in cloud manufacturing[J]. International Journal of Computer Integrated Manufacturing, 2017, 30(3):1-12.

[17] Li W, Zhu C, Yang L T, et al. Subtask scheduling for distributed robots in cloud manufacturing[J]. IEEE Systems Journal, 2017, 11(2): 941-950.

[18] Zhou L, Zhang L, Laili Y, et al. Multi-task scheduling of distributed 3D printing services in cloud manufacturing[J]. The International Journal of Advanced Manufacturing Technology, 2018, 96(9-12): 3003-3017.

[19] Cheng Y , Tao F , Zhao D , et al. Modeling of manufacturing service supply–demand matching hypernetwork in service-oriented manufacturing systems[J]. Robotics and Computer-Integrated Manufacturing, 2016:S0736584516301491.

[20] Li H , Chan K C C , Liang M , et al. Composition of Resource-Service Chain for Cloud Manufacturing[J]. IEEE Transactions on Industrial Informatics, 2017, 12(1):211-219.

[21] Xiang F , Hu Y , Yu Y , et al. QoS and energy consumption aware service composition and optimal-selection based on Pareto group leader algorithm in cloud manufacturing system[J]. Central European Journal of Operations Research, 2014, 22(4):663-685.

[22] Beloglazov A, Abawajy J, Buyya R. Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing[J]. Future generation computer systems, 2012, 28(5): 755-768.

[23] Tao F , Feng Y , Zhang L , et al. CLPS-GA: A case library and Pareto solution-based hybrid genetic algorithm for energy-aware cloud service scheduling[J]. Applied Soft Computing, 2014, 19:264-279.

[24] 李乔, 郑啸. 云计算研究现状综述[J]. 计算机科学, 2011, 38(4):32-37.

[25] 云计算白皮书(2012 年)[M]. 工业和信息化部电信研究院,2012.

[26] Birke R, Podzimek A, Chen L Y, et al. Virtualization in the Private Cloud: State of the Practice[J]. IEEE Transactions on Network & Service Management, 2017, 13(3):608-621.

[27] Burihabwa D, Pontes R, Felber P, et al. On the Cost of Safe Storage for Public Clouds: An Experimental Evaluation[C]// Reliable Distributed Systems. 2016.

[28] Baig R, Freitag F, Moll A, et al. Cloud-based community services in community networks[C]// International Conference on Computing. 2016.

[29] 李伯虎, 张霖, 任磊, et al. 云制造典型特征、关键技术与应用[J]. 计算机集成制造系统, 2012, 18(07):0-0.

[30] 张霖, 罗永亮, 陶飞, et al. 制造云构建关键技术研究[J]. 计算机集成制造系统, 2010, 16(11):2510-2520.

[31] Wu Z, Ni Z, Gu L, et al. A Revised Discrete Particle Swarm Optimization for Cloud Workflow Scheduling[C]// International Conference on Computational Intelligence & Security. 2011.

[32] Pandey S, Wu L, Guru S M, et al. A Particle Swarm Optimization-Based Heuristic for Scheduling Workflow Applications in Cloud Computing Environments[C]// IEEE International Conference on Advanced Information Networking & Applications. 2010.

[33] Xu L, Wang K, Ouyang Z, et al. An improved binary PSO-based task scheduling algorithm in green cloud computing[C]// International Conference on Communications & Networking in China. 2014.

[34] Xue S J, Wu W. Scheduling workflow in cloud computing based on hybrid particle swarm algorithm[J]. TELKOMNIKA Indonesian Journal of Electrical Engineering, 2012, 10(7): 1560-1566.

[35] Wen X, Huang M, Shi J. Study on Resources Scheduling Based on ACO Allgorithm and PSO Algorithm in Cloud Computing[C]// International Symposium on Distributed Computing & Applications to Business. 2012.

[36] Ramezani F, Lu J, Hussain F. Task scheduling optimization in cloud computing applying multi-objective particle swarm optimization[C]//International Conference on Service-oriented computing. Springer, Berlin, Heidelberg, 2013: 237-251.

[37] Liu J, Luo X G, Zhang X M, et al. Job scheduling algorithm for cloud computing based on particle swarm optimization[C]//Advanced Materials Research. Trans Tech Publications, 2013, 662: 957-960.

[38] Verma A, Kaushal S. Bi-Criteria Priority based Particle Swarm Optimization workflow scheduling algorithm for cloud[C]// Engineering & Computational Sciences. 2014.

[39] Zhao C, Zhang S, Liu Q, et al. Independent tasks scheduling based on genetic algorithm in cloud computing[C]//2009 5th International Conference on Wireless Communications, Networking and Mobile Computing. IEEE, 2009: 1-4.

[40] Liu J, Luo X G, Zhang X M, et al. Job scheduling model for cloud computing based on multi-objective genetic algorithm[J]. International Journal of Computer Science Issues (IJCSI), 2013, 10(1): 134.

[41] Jang S H, Kim T Y, Kim J K, et al. The study of genetic algorithm-based task scheduling for cloud computing[J]. International Journal of Control and Automation, 2012, 5(4): 157-162.

[42] Gu J, Hu J, Zhao T, et al. A new resource scheduling strategy based on genetic algorithm in cloud computing environment[J]. Journal of computers, 2012, 7(1): 42-52.

[43] Kaur S, Verma A. An efficient approach to genetic algorithm for task scheduling in cloud computing environment[J]. International Journal of Information Technology and Computer Science (IJITCS), 2012, 4(10): 74.

[44] Li J F, Peng J. Task scheduling algorithm based on improved genetic algorithm in cloud computing environment[J]. Jisuanji Yingyong/ Journal of Computer Applications, 2011, 31(1): 184-186.

[45] Ye Z, Zhou X, Bouguettaya A. Genetic algorithm based QoS-aware service compositions in cloud computing[C]//International Conference on Database Systems for Advanced Applications. Springer, Berlin, Heidelberg, 2011: 321-334.

[46] Hinton G E, Salakhutdinov R R. Reducing the dimensionality of data with neural networks [J].

science, 2006, 313(5786): 504-507.

[47] Yang X S. A new metaheuristic bat-inspired algorithm[M] Germany: Springer, Berlin, Heidelberg, 2010: 65-74.

[48] Meng X B, Liu Y, Liu Y, et al. A novel bat algorithm with habitat selection and Doppler effect in echoes for optimization [J]. Expert Systems with Applications An International Journal, 2015, 42(17–18):6350-6364.

[49] 简琤峰,王斌,张美玉,等.一种求解云服务组合全局 QoS 最优问题的改进杂交粒子群算法[J]. 小型微型计算机系统, 2017, 38(7):1562-1567.

[50] Y. Bengio. Learning Deep Architectures for AI [J]. Foundations and Trends in Machine learning, 2009, 2(1): 1-127.

[51] Y. Bengio, P. Lamblin, D. Popovici, et al... Greedy Layer-Wise Training of Deep Networks [C]. Advances in Neural Information Processing Systems, 2006, 19: 153-160

[52] Hinton G E. Learning to represent visual input [J]. Philosophical Transactions of the Royal Society B: Biological Sciences, 2010, 365(1537): 177-184.

[53] Smolensky. Information processing in dynamical systems: foundations of harmony theory [R]. DTIC, 1986,194-281.

致　谢

作者简介

1 作者简历

××××年××月出生于×××××。

××××年××月——××××年××月，××大学××院（系）××专业学习，获得××学硕士学位。

2 攻读硕士学位期间发表的学术论文

[1] 面向云制造资源调度预测的学习模型[J]. 小型微型计算机系统, 2019, 40(2). （第二导师为第一作者，本人为第二作者）

[2] 一种基于改进 ACF 特征的手部检测方法[J]. 小型微型计算机系统, 2018, v.39(07):200-204. （导师为第一作者，本人为第三作者）

3 参与的科研项目及获奖情况

[1] 面向设计意图在线交换的语义云粒元重组方法研究, 国家自然科学基金项目. 编号: 61672461.

[2] 图像与视频的不变性局部结构特征描述及应用研究, 国家自然科学基金项目. 编号: 61672463.

4 发明专利

[1] 一种基于 ONBA 的云制造任务最短生产时间调度方法. 中国, 2018 1 1237571.1 [P]. （第二导师为第一作者，本人为第二作者）

学位论文数据集

密　级*	中图分类号*	UDC*	论文资助
公开	TP391	004	
学位授予单位名称	学位授予单位代码	学位类型*	学位级别*
浙江工业大学	10037	工程硕士	硕士
论文题名*	面向云制造多目标优化资源调度结果的预测方法研究		
关键词*	云制造，调度模型，新蝙蝠算法，深度学习，调度结果预测		论文语种*
并列题名*	无		中文
作者姓名*		学　号*	
培养单位名称*	培养单位代码*	培养单位地址	邮政编码
浙江工业大学计算机科学与技术学院	10037	杭州市潮王路 18 号	310032
学科专业*	研究方向*	学　制*	学位授予年*
软件工程	云制造	2.5	2019
论文提交日期*	2019 年 06 月		
导师姓名*		职　称*	教授
评阅人	答辩委员会主席*	答辩委员会成员	
电子版论文提交格式：文本（　）图像（　）视频（　）音频（　）多媒体（　）其他（　）			
电子版论文出版（发布）者	电子版论文出版（发布）地	版权声明	
论文总页数*	55		
注：共 33 项，其中带*为必填数据，为 22 项。			

附件 2：学位论文书脊示例