****

**硕 士 学 位 论 文**

题 目 基于上下文感知的多Cloudlet联合调度策略研究

作 者 2014223040160 完成日期 2017年03月29日

培 养 单 位 四川大学

指 导 教 师

专 业 计算机技术

研 究 方 向

授予学位日期 年 月 日

基于上下文感知的多Cloudlet联合调度策略研究

专业：计算机技术

研究生： 指导教师：

摘要

关键词：

Research on Context-aware Cooperative Muti-Cloudlets Scheduling Strategy

**Major:** Computer Techology

**Graduate student:**  **Supervisor:**

**Abstract**

.

**Key words:**

目录

[第一章 绪论 1](#_Toc467605795)

[1.1 研究背景与意义 1](#_Toc467605796)

[1.2 研究现状 1](#_Toc467605797)

[1.3 本文主要工作 2](#_Toc467605798)

[1.4 本文的组织结构 3](#_Toc467605799)

[第二章 背景知识和相关技术 4](#_Toc467605800)

[2.1 移动云计算 4](#_Toc467605801)

[2.1.1 移动云计算概述 4](#_Toc467605802)

[2.1.2 移动云计算的模式 4](#_Toc467605803)

[2.1.3 移动云计算的优势和挑战 4](#_Toc467605804)

[2.1.4 多Cloudlet联合调度模型 5](#_Toc467605805)

[2.2 上下文感知 5](#_Toc467605806)

[2.2.1 上下文基本概念 5](#_Toc467605807)

[2.2.2 上下文建模技术 8](#_Toc467605808)

[2.2.3 上下文获取技术 10](#_Toc467605809)

[2.2.4 上下文聚合算法 11](#_Toc467605810)

[2.3 优先指派算法 12](#_Toc467605811)

[2.3.1 优先指派算法概述 12](#_Toc467605812)

[2.3.2 优先指派规则 13](#_Toc467605813)

[2.4 遗传算法 13](#_Toc467605814)

[2.4.1 遗传算法概述 13](#_Toc467605815)

[2.4.2 遗传算法的过程和流程 14](#_Toc467605816)

[2.4.3 遗传算法基因操作 14](#_Toc467605817)

[2.4.4 遗传算法控制及特点 15](#_Toc467605818)

[2.5 移动云环境下的任务调度问题 16](#_Toc467605819)

[2.5.1 任务调度问题描述 16](#_Toc467605820)

[2.5.2 任务调度方式 16](#_Toc467605821)

[2.5.3 任务调度问题的求解方法 16](#_Toc467605822)

[第三章 多Cloudlet联合调度系统模型 18](#_Toc467605823)

[第四章 基于优先指派和遗传算法的多目标优化算法(PA-GA) 21](#_Toc467605824)

[4.1 算法概述 21](#_Toc467605825)

[4.2 染色体编码及种群初始化 22](#_Toc467605826)

[4.3 适应度函数 24](#_Toc467605827)

[4.4 选择复制 26](#_Toc467605828)

[4.5 遗传操作 27](#_Toc467605829)

[4.5.1 交叉 27](#_Toc467605830)

[4.5.2 变异 29](#_Toc467605831)

[4.6 算法总结 29](#_Toc467605832)

[4.6.1 算法流程 29](#_Toc467605833)

[第五章 算法仿真和实验对比 30](#_Toc467605834)

[5.1 仿真环境 30](#_Toc467605835)

[第六章 总结和展望 30](#_Toc467605836)

[6.1 工作总结 30](#_Toc467605837)

[6.2 下一步研究工作 31](#_Toc467605838)

[参考文献 31](#_Toc467605839)

[作者在读期间科研成果简介 32](#_Toc467605840)

[声明 33](#_Toc467605841)

[致谢 34](#_Toc467605842)

# 第一章 绪论

## 1.1 研究背景与意义

近些年来,得益于移动智能设备和无线网络技术的快速发展,通过移动终端获取互联网内容已经成为人们生活中的又一主流形式。移动通信网络和互联网的相互融合形成了移动互联网的这种新兴的信息模式，它使得用户可随时随地的使用移动设备通过无线网络(3G/4G/WiFi等)访问互联网，解决了传统PC体积和重量较大、不易携带和可移动性差等缺点。尽管目前移动设备的处理能力和存储能力都得到了很大提升，但是针对日益复杂的移动任务，如计算密集型、实时交互型、高并发型等任务时，移动设备本身固有的一些缺陷包括处理能力、存储容量、电池续航、网络状况、可靠性等将成为性能瓶颈。

为了解决移动设备的固有限制,移动云计算应运而生。传统的解决方案是将移动任务通过无线网络（3G/4G）分流到远端云上，通过结合云计算和移动设备各自的优势，提升移动设备的处理能力和存储能力，延长移动设备的电池周期。然而，这种方式也存在着一些不可忽视的问题，如：1、移动设备和远端云通过无线网络（3G/4G）进行通信，由于距离较远，会产生比较大的通信时延，这对于实时性要求较高的任务来说是致命的。 2、3G/4G这种通信方式，将产生昂贵的通信费用。 3、3G/4G目前来说数据传输速率仍然有很大的提升空间，对于通信量较大的任务将产生较大的传输时延。 4、一些安全性要求较高的移动任务分流的企业的公有云上会带来一定的安全隐患。

移动云计算中一种新的移动终端-cloudlet-远端云的三层计算模型提供了另外一种思路[1]。Cloudlet是位于移动用户附近，由少数服务器，PC，移动设备等组成的一个移动设备和远端云通信的中转站或者直接服务移动设备的代理服务器，它以WiFi方式接入[2]。与终端-云端的两层计算模型相比，这种三层模型在保留了云端无限的计算能力和存储能力的优势下，将任务的执行尽可能的从云端移动到cloudlet上，这带来的好处是不言而喻的，如：接入方式从3G/4G过渡到WiFi，使得传输速度得到极大提升，移动终端保持连接所消耗能量减小，延长了移动终端续航时间。执行体由远端云过渡到位于移动终端附近的cloudlet，通信距离远远降低，使得通信时延和传输时延都得到极大程度的改善。另外，还消除了昂贵的3G/4G通信费用和企业云的租赁服务费用。

## 1.2 研究现状

移动云计算在近些年来得益于移动智能终端和云基础设施的发展受到研究者的广泛关注,使用Cloudlet 策略在移动云计算领域变得越来越流行。在[11]中,作者预见未来Cloudlet基础设施会像今天的WiFi接入点一样分布在我们周围。不出意料地,任务分流[4,5,6,7,8]在移动云环境中也是研究的热点。[5]分析移动云计算相关背景和移动云计算中计算机分流框架的概况和存在的问题。[6]为分流应用和切换服务到其他的Cloudlet上提出了两个决策算法。提出的算法使得当用户在移动过程中也能提供无缝的Cloudlet服务。[7]为移动用户提出一个最优分流算法,考虑用户本地负载和Cloudlet的可用性。并提出一个位于Cloudlet系统中基于分流算法的马尔科夫决策方法(MDP)。[4]提出一个包括在移动设备和它们的云基础设施或者复制体中间的中间层的移动云计算的新的系统架构,这个中间层由Cloudlet组成,所以叫做Cloudlet层。另外,提出一个动态分流算法,创新性在于:当决定向哪儿offload时不止考虑一个分流位置作为参数。[8]提出了一个在这种间歇性连接的Cloudlet系统中基于移动用户的最优的分流算法,考虑用户的本地负载和Cloudlet的可用性。

移动性导致切换,服务质量退化和服务中断,这阻碍了透明的去访问计算云中服务的目标[9]。因此,移动性也是广大研究者热衷的研究课题[1,10,6,7,12]。[1]使用Cloudlet的访问概率、任务成功率、和任务执行速度来衡量移动性带来的影响。他们发现,Cloudlet访问概率可用来表示。[10]要解决的问题:在一个Cloudlet系统中,由于用户的移动性和cloulet的负载能力情况,移动用户和Cloudlet的连接将可能是间歇性的,这可能会导致用户分流的行为失败。[6]提出一个模型来处理移动云计算中两个比较严峻的挑战:移动性和高效。[7]考虑由用户和Cloudlets的移动性引起的分流失败的细粒度来做分流决策。[12]提出了一种基于Cloudlet的移动云计算模型,这个模型由特定区域内一些分布并彼此连接的Cloudlet集合组成。文中使用此架构来解决移动设备的移动性问题。[14]中首先应用一个细粒度的数据流模型来对移动交互式数据流应用进行建模,然后构建一个统一的优化框架,能够算出所有移动用户的总的使用率,并对该优化问题设计了一个高效的算法,这个框架能够在每个移动设备上权衡吞吐量和能耗。[15]文章提出两种管理模式,集中模式,cloud负责管理和跟踪移动用户信息和非集中模式,移动终端负责自己管理自己的移动信息。根据不同的移动场景进行Cloudlet和云端的选择性任务分流。[16]致力于移动云计算环境中的降低时延和能耗问题,文章提出一个混合本地移动云模型(HLMCM),并根据该模型提出了一个调度算法。[17]提出了一个Cloudlets之间通过特定的协议相互连接的模型。[19]提出一个Cloudlet中间件平台,它能处理应用分发并能动态配置应用,并且管理在网络中发现的资源,包括本地固定的虚拟化基础设施和在这个网络中支持该平台的所有设备。

## 1.3 本文主要工作

针对移动云环境下的任务计算模型及任务分流等问题，本文在深入钻研了大量现有的相关文献及研究成果的基础上，为了解决MCC中面临的诸多问题，如：扩展移动终端的处理能力和存储能力、降低任务开销、延长移动终端电池使用时间、降低响应时延、充分利用闲置资源······, 本文将提出一个移动终端-Cloudlet-云端的三层计算模型，并在此模型基础上，提出一个基于上下文感知的多Cloudlet联合调度算法，具体工作如下：

1）简要概述移动云计算面临的问题，以及当前的研究成果，并探讨本文是如何解决这些问题的。

2）分析目前现有的一些计算模型的不足，提出了移动终端-Cloudlet-云端的三层计算模型，并对该计算模型的优势进行说明。

3）针对在多Cloudlet协同工作的环境下，提出了一个基于上下文感知的多Cloudlet联合调度算法。

4）使用模拟实验对本文提出的计算模型和调度算法进行验证及对比。

## 1.4 本文的组织结构

论文总共包括六个章节，每个章节的主要内容如下：

第一章为绪论，首先本文的相关研究背景和意义，接着概述了目前移动云计算方面的研究成果及相关状态，最后叙述了本文所做的主要研究工作以及每个章节布局及主要内容。

第二章主要介绍移动云计算及移动云计算中Cloudlet的背景知识和相关技术。首先介绍了移动云计算的概念、常见场景以及面临的挑战；接着总结了目前移动云计算中常见的计算模式；然后，介绍了移动终端-Cloudlet-云端三层计算架构的特点、核心思想和要解决的问题。最后，对常见的多Cloudlet联合调度策略进行简单分析和总结。

第三章详细介绍本文提出的移动终端-Cloudlet-云端三层计算模型。

第四章为算法仿真和实验对比。

第五章为总结和展望，先对本文所做的研究工作进行了简单的总结，接着指出本文所提出的计算模型及调度策略的优势和缺陷，最后对下一步将要进行的研究工作进行了展望。

# 第二章 背景知识和相关技术

## 2.1 移动云计算

### 2.1.1 移动云计算概述

近些年来，随着移动智能终端和无线网络通信技术的快速发展和成熟，随时随地通过移动设备接入互联网获取所需信息和服务变成了现实。但是，由于移动设备本身固有的一些限制，如处理和存储能力较弱、电池技术发展遇到瓶颈和无线网络的不稳定性等，移动计算也遇到了很多问题。今天，云计算的概念已经深入人心，云计算产品和应用层出不穷，云计算正在从1.0时代迈向更加成熟、更加实用的2.0时代。云计算的发展和应用很大程度改变了过去任务处理和使用资源的方式，云服务提供商利用分布式计算和虚拟资源管理等技术，通过网络将分散的ICT资源（包括计算与存储、应用运行平台、软件等）集中起来形成共享的资源池，并以动态按需和可度量的方式向用户提供服务，用户只需按照自己的具体需要向云服务提供商购买资源，并可在后期动态扩展所需资源。云计算的服务通常包含IaaS、PaaS、SaaS三个层次。就在2007年“云计算”的概念出现后不久，为了解决移动计算面临的问题，借助云计算强大的资源聚合能力，“移动云计算”这一概念应运而生，引起了很多学者和企业的广泛关注。

移动云计算是云计算技术与移动通信技术结合的产物，利用移动云计算的各种无线互联网的服务将深入到人们的生活中，它的出现和应用将成为通信产业发展的关键一环。移动云计算是指通过移动网络以按需、易扩展的方式获得所需的基础设施、平台、软件等的一种IT资源或服务的交付与使用模式，它是云计算技术在移动互联网中的应用。移动云计算的出现，使得应用的存储、计算等资源从本地转到了云端。用户只需要一台能够安装客户端的终端，无需有超强的性能就能够实现诸如文件存储、移动导航、语音搜索、手机地图等应用，这些服务都是依托云计算来实现的。

### 

### 2.1.2 移动云计算的模式

### 2.1.3 移动云计算的优势和挑战

为了解决传统云计算在移动环境下的诸多限制，移动云计算结合云计算和移动计算带来了很多方面的优势，主要如下：

（1）延长电池使用时间。相对现今移动智能终端在处理能力和存储能力上都得到的飞跃式发展，电池的续航能力仍然是一个非常难以解决的问题。因此，如何延长电池的使用时间成为移动设备最关心的问题之一。移动云计算通过分流技术，将复杂的计算密集型任务从资源有限的移动终端转移到计算能力强大的服务器上或其它资源充足的设备上执行，从而节约了移动终端进行长时间大量计算所消耗的能量，达到延长电池使用时间的目的。

（2）扩展移动终端的处理和存储能力。尽管移动设备的处理和存储能力得到很大程度上的发展，但是相对于云计算环境中无限的计算资源和存储能力来说，移动端的处理和存储能力仍然是一大劣势，当需要执行大量计算或存储大量数据时，这两种能力将成为移动任务的瓶颈。但在移动云计算环境中，通过”云”向用户提供计算和存储功能，移动任务就可以使用云中丰富的计算资源和强大的存储能力，进而完美地扩展了移动终端的处理和存储能力。

（3）增强数据可靠性。移动智能设备为了达到便携，美观等特点，也相应地会在一些方面做出一些牺牲，如易发生故障，易丢失，易发生误操作等。移动云计算把数据处理和数据存储都转移到了云端，通过云服务提供商强大的安全策略，不仅可以对用户数据进行保护和备份，还能为移动用户提供远程的病毒扫描、木马检测，恶意攻击拦截等功能，可达到全方位地保护用户数据安全。

此外，移动云计算还有动态配置、降低成本、智能均衡、易整合和扩展性强等优点。

由于用户的移动性和移动终端固有的一些限制，移动云计算中除了存在着传统云计算中常见的问题外，还引入了一些特殊的问题，主要如下：

（1）移动性。在移动云环境中，移动性是一个最基本的特性。由于移动终端移动的不确定性，网络服务提供商很难保证移动用户始终处于移动网络的覆盖范围内，换言之，移动性使得移动终端的网络连接具有间歇性，这不仅仅会导致网络连接信号强度产生波动，还可能导致网络频繁切换，任务分流失败等情况。另外，移动设备移动过程中的网络吞吐量要明显低于移动设备静止时的网络吞吐量。为了解决这个问题，很多学者提出了一些优秀的解决方案，如数据缓存、预先配置、移动预测等。

（2）低带宽。当前大多移动设备都是使用2G/3G/4G/Wi-Fi无线网络享受网络服务，相对有线网络来说，无线网络资源非常稀缺，因此低带宽也是移动云计算中面临的主要问题之一。低带宽会明显增加延迟，这对一些实时性应用来说是灾难性的。解决该问题的途径一般有两种思路，一种思路是寻找更先进、具有更大带宽和具有更大覆盖面积的无线网络方案或尽可能使用有线网络来代替无线网络传输，另一种思路则是通过其他渠道尽可能避免大量的网络数据传输。

（3）无线网络异构和不稳定。随着无线通信技术的飞速发展，出现了诸如蜂窝移动网络(2G/3G/4G)、无线局域网(Wi-Fi)、无线城域网(WiMAX)、无线传感器网(WSN)等各式各样的采用不同组网技术的异构网络。在这样的异构移动网络环境中，无缝的网络切换服务是一个严峻的挑战。另外，用户移动性和异构的网络环境也是导致网络连接不稳定的因素之一，异构网络的身份验证、允入控制和授权问题等也会导致网络的不稳定，关于这些问题的解决方案还有待做进一步研究。

（4）服务质量。事实上，不管是传统云计算或是移动云计算，服务质量都是移动服务提供商和移动用户共同追求的目标。在移动云环境中，服务质量会涉及延迟、网络传输速率、任务成功率、数据包丢失率、用户体验等诸多方面。一方面，良好的服务质量才能保证用户和移动服务提供商之间的双赢局面，而完善的QoS评价体系和优化模型对于移动用户和移动服务提供商来说也都非常重要，具有很好的理论和实际意义。因此，如何保证移动云计算中的服务质量的研究显得尤为重要。

此外，诸如移动任务分割、安全、移动上下文管理等也是移动云计算中重要的问题和研究热点。

### 2.1.4 多Cloudlet联合调度模型

## 2.2 上下文感知

### 2.2.1 上下文基本概念

大多数的研究人员认为“上下文”（Context）指的是与应用程序相关联的周围环境状况，以及应用程序执行环境特性的组合。目前，被学术界与工业界广泛认同与接受的“上下文”定义是由 K.Dey[15]等人提出的。该定义如下：“任何可以描述一个实体情况或特点的信息都可以称为上下文。这里的实体可以是一个人或一个地点，也可以是用户和应用程序之间的一个交互对象（包括用户和应用程序本身）。” “上下文”信息除了位置和身份之外，还有历史信息，时间等。主要分为计算上下文、用户上下文、物理上下文和时间上下文。1) 计算上下文：主要包括网络连接、通信成本、通信传输速率以及附近的资源（例如：打印机、显示器、工作站等）。2) 用户上下文：包括用户的配置信息、位置、附近的人们和它们所处的社会状况等。3) 物理上下文：包括灯光照明、噪音水平、交通状况以及气温等。4) 时间上下文：包括某天、某周、某月、某季度等。本文所讨论的上下文主要是用户的 QoS 偏好、移动设备的非功能性上下文信息，这些符合 Dey 提出的上下文定义。 虽然“上下文”有一个完整的定义，但是单从定义上理解上下文还是比较模糊，抽象的。Gregory D. Abowd 和 Elizabeth D. Mynatt[16] 就将“5Ws：Who， What， Where， When， Why”作为在普适计算环境中必要的上下文信息最小的集合，以下是对 5Ws 进行简单的介绍： Who：谁，参与系统交互的特定用户身份以及在环境中与该用户发生交流的人们的身份信息。人们将根据身边的人调整各自的活动。 What：干什么，当前系统的通过与用户的交互，感知去判断用户正在干什么或者为什么离开。感知和解释人类的活动是一个棘手的难题，上下文感知驱动设备通过不断地与用户交互，磨合，能够提供有用的信息。 Where：位置，是很多上下文研究的重点。把“Where”与其它上下文信息结合（例如，“When”），就会变得有很大的实际意义。一些旅游向导系统将显示生活中设备移动的历史记录理论化，用于定制用户感兴趣的路线，提供个性化的服务，尽管此类想法还需要更全面的研究和探索。 When：时间，记录用户特定行为发生的时间和持久时间，通过总结时间的规律和变化，有助于解析用户的活动。比如，在一个展览上停留的时间短可以预测用户对该展览缺乏兴趣；另外，根据一位年长的老人没有在常规的时间段内回家，可以给家人发送提醒。 Why：为什么，解析或感知人们为什么做此活动比感知人们做什么活动更加富有挑战。这需要上下文系统去感知用户的情感状态，比如声音、体温、皮肤电反应等。 这些集合主要用于描述用户相关的上下文信息，需要对这些集合进行有机的“上下文”融合后，才能比较准确地解析、推导出用户的活动和行为状态，提供更高质量的用户体验。然而 5Ws 的不足之处就是它并没有将用户周围的计算环境纳入到上下文信息的集合中。 自适应系统要求计算设备能够主动去感知用户所处环境的上下文变化，并自主地调整软件的组织结构或调整软件的功能，使得人机交互过程更加自然（甚至对人们透明），达到“以变应变”的目标，这就要求自适应系统运用上下文感知温等。4) 时间上下文：包括某天、某周、某月、某季度等。本文所讨论的上下文主要是用户的 QoS 偏好、移动设备的非功能性上下文信息，这些符合 Dey 提出的上下文定义。 虽然“上下文”有一个完整的定义，但是单从定义上理解上下文还是比较模糊，抽象的。Gregory D. Abowd 和 Elizabeth D. Mynatt[16] 就将“5Ws：Who， What， Where， When， Why”作为在普适计算环境中必要的上下文信息最小的集合，以下是对 5Ws 进行简单的介绍： Who：谁，参与系统交互的特定用户身份以及在环境中与该用户发生交流的人们的身份信息。人们将根据身边的人调整各自的活动。 What：干什么，当前系统的通过与用户的交互，感知去判断用户正在干什么或者为什么离开。感知和解释人类的活动是一个棘手的难题，上下文感知驱动设备通过不断地与用户交互，磨合，能够提供有用的信息。 Where：位置，是很多上下文研究的重点。把“Where”与其它上下文信息结合（例如，“When”），就会变得有很大的实际意义。一些旅游向导系统将显示生活中设备移动的历史记录理论化，用于定制用户感兴趣的路线，提供个性化的服务，尽管此类想法还需要更全面的研究和探索。 When：时间，记录用户特定行为发生的时间和持久时间，通过总结时间的规律和变化，有助于解析用户的活动。比如，在一个展览上停留的时间短可以预测用户对该展览缺乏兴趣；另外，根据一位年长的老人没有在常规的时间段内回家，可以给家人发送提醒。 Why：为什么，解析或感知人们为什么做此活动比感知人们做什么活动更加富有挑战。这需要上下文系统去感知用户的情感状态，比如声音、体温、皮肤电反应等。 这些集合主要用于描述用户相关的上下文信息，需要对这些集合进行有机的“上下文”融合后，才能比较准确地解析、推导出用户的活动和行为状态，提供更高质量的用户体验。然而 5Ws 的不足之处就是它并没有将用户周围的计算环境纳入到上下文信息的集合中。 自适应系统要求计算设备能够主动去感知用户所处环境的上下文变化，并自主地调整软件的组织结构或调整软件的功能，使得人机交互过程更加自然（甚至对人们透明），达到“以变应变”的目标，这就要求自适应系统运用上下文感知

4.1.1 移动环境的上下文感知

移动云计算中的移动应用载体通过无线网络与其它设备相连，它们的网络环境随着时间和空间的变化而动态变化。因此，执行动态环境下的服务，需要不断且快速地调整移动设备的操作状态来适应环境的变化，即服务需要主动感知移动设备周围环境的变化，进而通过改变自身行为来适应这些变化。这属于上下文感知（Context-aware）自适应的应用范畴。

在移动计算中，上下文感知计算本身并不是一个新的概念，它已经受到广泛的重视和大量的研究。美国哥伦比亚大学 Schilit 在他的博士论文中提到的移动计算的上下文感知模型应该是上下文感知在移动计算方面较早的研究[49]。移动云计算处于一个动态而复杂的无线网络环境下，移动终端与周遭环境会连续不断地进行隐形交互。在这个过程中，系统是根据用户与应用程序的上下文相关的信息来提供服务，但是由于移动终端设备资源的受限性以及环境的不可预测性，系统和应用都不太可能预先判断或准备好所有可能的服务与资源，现有的服务和资源可能都是暂时的，它们可能随时失去效应，或者随时加入或离开应用程序所处的环境。另外，移动设备自身也是高度动态的，它可能随时到达或者离开去一个不同的环境，也可能随时要求系统提供新的需求支持。与此同时，移动应用本身也可能需要根据它所处上下文运行环境的改变而动态响应。因此，一方面需要上下文感知系统根据它所处环境的改变动态自适应，来确保更好地支持和运行移动应用。另一方面，为确保继续为用户提供尽可能好的服务，移动应用也需要自适应于其运行环境的不断变化。进一步地，用户自身也需要自适应地感知、调整对系统与应用的要求。由于无线通讯链接质量等问题显得比较突出，早期支持的自适应移动系统的研究主要集中在网络服务质量（QoS）上。然而，最近的一些研究也将其它方面的特征考虑进来了。例如，移动应用的需求变化、电池的能量供应等问题，但这些考虑基本上都是针对单个问题来进行独立处理，通过下层的相关机制通知应用，由移动应用本身来负责激活自适应机制并进行自适应处理。目前，将各种自适应问题综合起来研究还处于初级阶段。

根据移动应用的不同位置、时序逻辑和应用逻辑，移动应用中每个计算单元有一个不同的上下文。从原理上讲，一个服务所依赖环境中的任何信息都可以被看作是上下文数据，它们由大量可观察，可控制参数组成。一般说来，移动环境中的上下文具有如下几个基本特点：

1) 时效性：上下文信息可以分为静态上下文和动态上下文两种。静态上下文信息不会随环境和时间的变化而改变，它们往往是由应用程序或者服务明确赋值的，如用户的账户，生日等信息。而移动环境中的绝大多数上下文信息属于动态信息，这些上下文信息的取值不同会随着时间和位置动态改变，例如由硬件传感器获取的行为和声音等。

2) 非完整性：由于技术上的局限，基于传感器获得的上下文信息会缺失完整性而出现误差；上下文信息的发现、产生与使用之间会有时间延缓，从而造成获取上下文信息不能反映当时的真实情况；另外，移动系统网络下的不稳定性和动态性使得某些的上下文信息在通过网络传播时发生丢失，造成部分上下文信息未知。

3) 表达多样性：应用程序使用对象所要求的上下文数据与从物理环境或逻辑环境中采集的原始上下文数据在表现形式上可能有很大不同。于此同时，同一个数据源提供的上下文信息可能被不同使用对象共享，因此上下文信息需要支持各种不同表达方式间的转换。

4) 高度相关性：上下文信息之间可以有继承关系，即有的上下文信息是从其它的上下文中派生过来的。因此，某个上下文信息的变化可能导致另一个上下文信息发生对应变化，即上下文信息之间可能存在着依赖关系。由此得出，很多上下文信息之间是高度相关的。

在研究上下文感知相关技术之前，需要了解什么是上下文。上下文的概念最早由 Schilit 和 Theimer 提出，他们对上下文的定义是：使用位置和周围的人和物的集合以及变化情况[10]。Ryan 等人将上下文定义为用户的位置，用户周围的环境，用户的身份和当前的时间[11]。Brown 将上下文定义成计算机所能感知到的用户环境中的各种元素[34]。Dey 将上下文定义为任何可以用来描述实体周围情形的信息，实体可以是一个人、场所或对象，只要与用户和应用之间的交互有关系，也包括用户和应用本身[8]。学者们对上下文概念的定义都不一样，其侧重点不一样，那么在本文中，对上下文的定义是： 任何移动终端能感知到的周围环境信息，和移动终端自身的各种硬件信息、软件信息。 对于某一些应用来说，单独一个设备的上下文信息可能没有任何意义，但是，当众多设备的所有上下文信息综合在一起的时候，可能就会产生很重要的价值信息。本文中根据移动云计算的研究背景，将上下文综合定义为： 在移动云计算环境下，将大规模的上下文信息聚合成一种更加上层的、对应用直接有用的信息。 在传统的上下文感知计算中，单个设备或少数设备收集到上下文信息之后，就利用这些上下文信息能够得到有意义的信息。但本课题针对有些上下文信息，在聚集了大规模的该数据之后，产生的另一种非常有意义的信息，如一些网站统计软件的整体下载次数来给出软件的受欢迎程度。本课题中重点研究如何从大规模上下文综合信息中聚合出有意义的信息来为用户提供更好的服务，包括如何将众多移动终端收集的上下文信息综合在一起，以及如何对综合的上下文信息进行处理。 在上下文感知技术中，上下文建模技术、上下文获取技术和上下文聚合算法是比较重要的三种技术，其中上下文建模技术涉及到如何对上下文信息建立模型，而且所建立的上下文模型应该具有通用性、跨平台性、一致性等等；上下文获取技术涉及到如何高效地获取上下文信息、如何使得获取的方式具有可移植性等等；上下文聚合算法涉及到如何高效地将大量原始的上下文信息计算处理成对具体应用有用的信息。上下文建模技术是上下文感知技术的基础，其为上下文获取技术和上下文聚合算法提供支撑，一个好的上下文模型能够简化上下文获取和上下文聚合的研究，而且还能提高上下文获取和上下文聚合的效率。上下文获取是上下文感知技术的前提，只有在获取了上下文信息之后才能对所获取到的上下文信息进行研究，从传感器获取到的原始数据采用上下文模型进行统一的描述，上下文获取是上下文聚合的前提条件。上下文聚合是上下文感知技术的核心，将大量原始的上下文信息进行计算分析、处理得到对应用有用的信息，才能使得上下文信息真正有用。与上下文感知技术相关的研究都离不开对这三种技术的研究，本章重点描述这三种技术的研究。

### 2.2.2 上下文建模技术

对上下文进行建模的形式多种多样，学术界根据问题领域的不同，或者期望得到一种统一的上下文建模方式，提出了多种上下文模型，本节将分别描述比较典型的五种上下文模型：标记模型、键值对模型、图形模型、基于逻辑模型和基于本体模型。

2.2.1 标记模型

标记模型一般采用 XML 格式来描述上下文，XML 具有很强的描述性，通过XML 中的标签来表示上下文名称，属性值来表示对应的上下文的值，表 2.1 是Held[35]在研究上下文感知计算中用 XML 方式来描述上下文信息的一个示例。从XML 的特性可以看出标记模型描述上下文信息具有很强的可理解性，而且具有很好的可移植性以及通用性。但其中也存在着一些不足之处，XML 描述一条上下文信息显得有点冗余，特别是 XML 文件头的一些声明对于具体上下文信息的描述不是很有意义，这些因素使得 XML 描述上下文会浪费多余的内存或硬盘空间。

2.2.2 键值对模型

键值对模型是最简单的上下文建模方式，其键一般表示上下文的名称，值对应相应上下文的值，Schilit 是最早采用键值对的方式来对上下文信息进行建模的，表 2.2 是 Schilit 在他的研究中对上下文信息进行建模的一个示例[10]。从键值对的描述方式可以看出这种方式少了很多冗余的附加信息，上下文的主要信息都表达清楚了，和 XML 的标记模型相比当上下文规模较大时，键值对的描述与 XML 相比可读性上会有所损失，但是键值对模型精简的描述信息使得其在内存或硬盘空间的占用上大大地减少了。

2.2.3 图形模型

图形模型一般是采用 UML[36]（Unified Modeling Language，统一建模语言）来进行描述，UML 中具有丰富的建模元素，可以清晰地描述清楚上下文信息。Bauer在空中交通管理中使用了 UML 的扩展来描述上下文相关的一些方面[38]。另外，Henricksen 等人设计了一种面向图形的上下文模型，其主要使用对象规则建模[37]（Object-Rule Modeling，ORM）的方法来建模上下文，如图 2.1 所示，在 ORM 中基本的元素是事实（fact），该示例可以简单地描述一个人能够使用的设备和一个人所处的位置。这种描述方式直观上非常的清晰，能够很容易明白上下文的含义，但是比较不适合用程序来进行处理。2.2.4 基于逻辑模型 基于逻辑的上下文模型中一般会使用事实、描述、规则来定义一个上下文，在系统中上下文信息通常是被添加进去、删除、更新，这些都是根据逻辑推理来实现的。根据应用领域的不同，所使用的逻辑描述方式也会有所区别，图 2.2 是Akman 和 Surav 在他们的扩展情景理论（Extended Situation Theory）[39]中描述一个上下文上的约束信息，即各种规则。基于逻辑的上下文模型对语义的逻辑性要求很高，而且其适合应用在一些推理系统中，在人工智能的领域中如涉及到上下文的使用，则比较适合使用基于逻辑的模型对上下文建模。

2.2.5 基于本体（Ontology）模型

本体能够清晰描述事物概念及相互关系，非常适合用来描述日常生活中的一些信息，并转化成一些适合计算机处理的数据结构。在 CoBrA[40]系统中提供了一系列的本体概念来描述与上下文相关的一些实体，并且使用了一个中间代理 agent来使得系统能够在运行时不停地获取上下文信息。这种基于本体的上下文建模被很多研究者广泛使用，但是不同系统之间一般都会采用领域特定的描述方式，一个系统中的上下文建模方式很难在另一个系统中有效使用，因此这种上下文建模方式与特定的领域应用相关，不适合系统之间的通用。 从表 2.3 中可以看出，在描述性和通用性方面，标记模型和键值对模型都好于其他的三种模型，在空间占用性方面键值对模型的表现效果最好，而对于逻辑性来说，基于逻辑的模型效果最好。因此，对于一些逻辑性比较高的系统中，比较适合采用基于逻辑的模型，而系统中不涉及逻辑推理时，从描述性和通用性来考虑，可以考虑采用标记模型或者键值对模型。 在本课题中，研究移动云计算环境下的上下文综合，其各种移动设备之间的异构性和云端对上下文的处理不涉及到逻辑推理等，需要上下文的模型具有很好的描述性和通用性，因此课题中兼容性支持标记模型和键值对模型。

### 2.2.3 上下文获取技术

在上下文感知计算中，对于上下文获取的技术主要有三种方式：应用直接从传感器获取数据，通过软件抽象层的方式获取数据，通过 CS 模式在 Server 处获取数据。这三种方式各有利弊，本节将分别描述这三种上下文获取技术。

2.3.1 直接从传感器获取

在一些应用中，特别是早期的上下文感知计算中，应用直接与传感器进行交互，并从传感器上获取上下文信息，如图 2.3 所示，当应用需要需要某种上下文的时候，直接访问具体的传感器，从其获取数据信息。 在这种方式中，应用需要与具体的传感器进行绑定，也就是说在应用中需要包含传感器的驱动程序，使得应用开发者需要根据应用的特性来专门开发传感器的驱动程序，大大增加了应用开发人员的工作量，同时也使得应用难于维护。但是这种方式有一个好处就是，应用直接与传感器进行交互使得应用与传感器交互的效率具有很大的优势。

2.3.2 软件抽象层

为了使应用具有更强的可移植性，不需要依赖于具体的硬件设施，在传感器的上层开发出一套软件抽象层，将具体的应用与传感器隔离开，使得应用可以直接从软件抽象层获取各种上下文信息，如图 2.4 所示，当应用需要某种上下文的时候，访问软件抽象层，然后由软件抽象层与具体的传感器进行交互获取到相应的数据，最后将数据交给应用。 在这种方式中，应用不需要与具体的传感器进行绑定，只需要熟悉软件抽象层的使用方法，使得开发人员不需要关心底层传感器的硬件特性，只需要开发出与软件抽象层的交互代码即可实现从传感器获取上下文信息的功能，这种方式大大降低了开发人员的工作量，同时也使得应用能够在多种安装有软件抽象层的传感器设备上运行。但这种方式也存在着一个问题就是应用获取上下文信息的效率与软件抽象层的工作效率密切相关，与直接从传感器获取上下文的方式相比在效率上会有所降低。

2.3.3 CS（Client/Server）模式

在移动设备没有足够的计算能力对该设备收集到的上下文信息进行处理操作时，则会考虑将这些上下文信息交给外部设备来进行处理，一般这种情况是专门采用一台中心服务器来处理上下文信息，如图 2.5 所示，移动设备 A，B 和 X 等收集到各自的上下文信息之后并不直接进行处理，而是通过网络将信息发送至中心服务器，然后中心服务器对上下文进行处理之后就将所得到的结果与领域特定的应用相结合。在这种方式中，应用并没有与直接的传感器进行交互，上下文信息是通过网络的方式从其他移动设备上获取，这种方式使得应用并不需要与传感器进行交互，而只需要与接收上下文信息的中心服务器进行交互即可。但是采用这种模式需要中心服务器支持能够接收各种应用需要的上下文信息的能力，而且移动设备将上下文信息传输给中心服务器之间产生的延迟对实时性要求比较高的应用很不适合，其产生的网络带宽也会影响整个应用的效率。

2.3.4 比较与结论

针对三种上下文获取技术的分析，下面从对应用性能的影响，对开发人员工作量的影响，对应用维护及移植性的影响三方面进行比较，结果见表 2.4。 表 2.4 三种上下文获取技术的比较结果表 比较方面

上下文获取技术

直接从传感器获取 软件抽象层 CS 模式

对应用性能的影响 低 中 中

对开发人员工作量的影响 高 低 中

对应用维护及移植的影响 高 低 低

从表 2.4 中可以看出，从开发人员的角度出发，他们最喜欢软件抽象层的上下文获取方式，CS 模式其次，直接从传感器获取的方式虽然性能较好，但随着科技的发展硬件设备的性能将越来越好，这种方式的优势将会越来越不明显。在本课题中，研究移动云计算环境下上下文综合的态势感知，需要考虑两方面的问题，即如何在移动终端更好地获取到上下文信息，然后将各个移动终端的上下文信息汇集到云端，因此结合软件抽象层和 CS 模式两种上下文获取技术，其示意图如图2.6 所示。

### 2.2.4 上下文聚合算法

移动设备通过传感器以及其他手段获取到各种上下文信息之后，需要对这些上下文信息进行处理，将原始的、低级的上下文信息处理层高级的、对具体应用有用的信息的过程称之为上下文聚合。对于上下文聚合的过程，根据不同的上下文类型和上下文数据规模有不同的聚合算法，目前主要有两种方式：基于规则的聚合方式和基于用例的聚合方式，本节将分别描述这两种聚合方式。

2.4.1 基于规则的聚合

基于规则的上下文聚合一般需要先由开发人员自定义好一系列的规则，然后当某个上下文获取之后就由规则匹配器将该上下文与定义好的规则依次匹配下去，直到有一条规则满足条件之后就停止匹配，并触发该规则中定义的行动。图2.7 是 David 等人在 RCSM（Reconﬁgurable Context-Sensitive Middleware）[42]中使用的基于规则的上下文聚合的实现，左边部分是专门定义的一种规则语言CA-IDL，用于描述某一种操作会在上下文得到满足时会被自动触发，而右边部分则是 CA-IDL 语言编译器在 RCSM 系统中的工作方式。从 RCSM 系统以及其他的一些文献中可以看出，基于规则的上下文聚合算法比较容易实现，而且可以由开发人员自定义规则来决定应用在获取不同上下文时所采取的不同的行动，另外，这种方式对应用的结果具有很好的预期效果。

2.4.2 基于用例的聚合

基于用例的上下文聚合方式一般会采用贝叶斯概率法[43]、机器学习[44]、神经网络[45]等主要的研究方法，这些方法一般都会将上下文信息作为输入条件，然后经过聚合系统的计算之后会输出相应的结果信息。图 2.8 是神经网络对上下文进行聚合的算法示意图，输入信息为加速度、压力等各种上下文信息，经过神经网络的黑盒系统之后就会产生结果，并经过特殊的格式转换之后，应用就可以获取到上下文聚合的结果。采用神经网络来聚合上下文信息首先需要大量的样本信息来对网络进行训练，得到一个稳定的网络之后才可以用于上下文聚合。从神经网络的聚合算法可以看出，这种聚合算法的输出结果具有不可预期性，因为该聚合算法的内核是一个完全黑盒的系统，开发人员完全不了解上下文聚合的整个完整过程。类似的，其他如贝叶斯概率法，机器学习法等其他的基于用例的聚合算法，都存在聚合结果不可预期的缺点。

2.4.3 比较与结论

基于用例的上下文聚合算法相对于基于规则的上下文聚合算法在实现上比较复杂，且需要用到其他的一些技术，而且其中存在着结果预期性较差的缺点。 本课题中上下文聚合的工作是在云端完成的，根据云计算现有的技术还不是很成熟的现状，云计算中对于分布式计算的实现主要还是采用简单的 MapReduce[46]模型。那么用 MapReduce 的编程模型来编写基于用例的上下文聚合算法，其在实现上复杂，而且针对现有云计算平台的底层实现，基于用例的上下文聚合算法由于其复杂的实现机制对于云的资源利用率与性能提升的效果都不够理想。而基于规则的上下文聚合算法，与云计算平台的底层实现有比较大的相关性，在云计算中采用基于规则的上下文聚合算法能够更有效地利用云计算的各种资源，使得这种方式能优于基于用例的上下文聚合算法。因此，本课题中重点研究基于规则的上下文聚合算法在云计算中的使用。

## 2.3 优先指派算法

### 2.3.1 优先指派算法概述

2.1 优先指派算法

优先指派算法的核心思想是制定一组优先权规则，根据该规则，每一个工件的工序都有一个权重值，按照权重值的大小降序排列，依次选取一个工序加入到调度序列中，这样就可以得到一个调度方案。本节将介绍一些传统的优先指派规则，并详细阐述 A0算法。

2.1.1 传统优先指派算法概述

Panwalkar 以及 Wafik Iskander[34]在 1977 年对当时超过 100 种的优先规则进行了总结；Oliver Holthaus 以及 Chandrasekharan Rajendran[35]提出了几种评价标准，并给出了以这些标准为目标的优先指派规则算法；Tatsunobu Kawai 以及 Yasutaka Fujimoto[36]提出了几种将优先权规则结合的方法。优先指派算法中最关键的要素就是优先权规则。一般而言，优先指派规则可分为简单规则、复杂规则两大类。

简单优先指派规则可以根据作业时间长短、工序时间长短、已加工的工序数目、未加工的工序数目、已加工的工序时间长度和以及未加工的工序时间长度和等因素为标准来制定优先规则，典型的指派规则包括：最多剩余工序个数优先规则（Most Operation Num Remaining, MONR），最少剩余工序个数优先规则（Least Operation Num Remaining, LONR），剩余处理时间最长优先规则（Longest Work Remaining, LWR），剩余处理时间最短优先规则（Least Work Remaining, LWR），最长处理时间优先规则则（Longest Processing Time, LPT），最短处理时间优先规则（Shortest Processing Time, SPT）等等。

复杂优先指派规则通过设置优先级别来实现，一般会从高到低设定二到三个级别，每一个级别使用相同的比较原则，级别的组合唯一确定一个加工工序。第一级优先权级别高的工序优先加工；如果相等，则第二级优先权级别高的工序优先加工……以此类推。

### 2.3.2 优先指派规则

## 2.4 遗传算法

### 2.4.1 遗传算法概述

遗传算法是一种仿生优化算法,它模仿的机制是一切生命与智慧的产生与进化过程。它通过模拟达尔文“优胜劣汰、适者生存”的原理激励的结构;通过模拟孟德尔遗传变异理论在迭代过程中保持己有的结构,同时寻求更好的结构。作为一种随机的优化与搜索方法,遗传算法有鲜明的特点:1)遗传算法的操作对象是一组可行解,而非单个可行解;搜索轨道有多条,而非单条,因而具有良好的并行性。2)遗传算法只需利用目标的取值信息,而无需梯度等高价值信息,因而适用于任何大规模、高度非线形的不连续多峰值函数的优化以及无解析表达式的目标函数的优化,具有很强的通用性。3)遗传算法择优机制是一种“软”选择,加上良好的并行性,使它具有良好的全局优化性和稳健性。4)遗传算法可行解集是编码化的,因而具有良好的可操作性和简单性。

由子遗传算法是由进化论和遗传学机理而产生的直接搜索优化方法故而在这个算法中要用到各种进化和遗传学的概念。这些概念如下

(1)串(i)

它是个体(divid)的形式,在算法中为字符串,并且对应于遗传学中的染色体(Chromosome)。

(2)群体(Population)

个体的集合称为群体,串是群体中的元素。

(3)群体大小(P即ulationsize)

在群体中个体的数量称为群体购大小。

(4)基因(Gene)

基因是串中的元素,基因用于表示个体的特征。例如有一个串S=1011,则其中的元素l,0,l,l分别称为基因。它们的值称为等位基因(Alleles)。

(5)串结构空间ss

在串中,基因任意组合所构成的串的集合。基囚操作是在串结构空间中进行的。串结构空间对应于遗传学中的基因型(Genoytpe)集合。

(6)参数空间sp

这是串空间在物理系统中的映射,它对应于遗传学中的表现型(PhenoPyte)的集合。

(7)适应度(FitneSS)

表示某一个体对环境的适应度。

### 2.4.2 遗传算法的过程和流程

自从1975年.J.HHollnad系统地提出遗传算法的完整结构和理论以来,人们一直推动遗传算法的发展,对编码方式、控制参数的确定、选择方式和交叉机理等进行深入的研究,提出了各种变形的遗传算法(varinatsofCnaonicalAlgorithms(vCGA)。其基本途径概括起来主要有以下几个方面:.:.改变遗传算法的组成成分或使用技术,如选用优化控制参数、适合问题特性的编码技术;

.:.采用混合遗传算法;

.:.采用动态自适应技术,在进化过程中调整算法控制参数和编码粒度;

.:.采用并行遗传算法;

采用混合遗传算法的三个基本原则是:①尽量采用原有算法的编码;②利用原有算法全

局搜索的优点;③改进遗传算子。

混合遗传算法的流程图基本同于一般遗传算法,其流程图如图2一2所示。由图中可以看

出,混合遗传算法主要包括以下六个步骤:

第一步随机产生群体,个体数目一定,每个个体表示为染色体的基因编码;

第二步 计算个体适应度，并判断是否符合优化准则,若符合,输出最佳个体及其代

表的最优解,并结束计算,否则转向第三步;

第三步依据适应度选择再生个体,适应度高的个体被选中的概率高,适应度低的个体

可能被淘汰;

第四步按照一定交叉概率和交叉方法,生成新的个体;

第五步按照一定的变异概率和变异方法,生成新的个体;

第六步由交叉和变异产生新一代的种群,返回第二步。

遗传算法中的优化准则,一般依据问题的不同有不同的确定方式。例如,可以采用以下

的准则之一作为判断的条件:

.种群中个体的最大适应度超过预先设定值;

.种群中个体的平均适应度超过预先设定值;

.世代数超过预先设定值;

实际工作中,在进行遗传算法程序设计时,需要预先确定以下问题:1、定义终点集;2、

定义初始函数集;3、适应度值评价方法;4、确定运行控制量;5、终止运行的标准。

GP程序设计除了包括基本的选择、交叉和变异外,还可加入次操作来反映遗传程序结构的变化。

### 2.4.3 遗传算法基因操作

### 2.4.4 遗传算法控制及特点

遗传算法中需要选择的参数主要有串长l,群体大小n,交换概率凡,变异概率凡等。这些参数对GA性能影响很大。二进制编码时,串长的选择取决于特定问题解的精度。GoldbegrD.E提出了变长度串的概念,并显示了良好性能。为了选择合适的n,Pc,几,许多学者进行了研究,通常认为:若种群过小,算法就可能收敛于局部最优解,而不能收敛到最优结果或最优结果附近。这主要是因为种群规模过小,导致种群内个体多样性减小,从而可能丢失一些有意义的搜索点或最优点。然而种群过大,每次迭代所需要的计算量就会很大,这又可能导致一个无法接受的慢收敛率。一般,当种群规模增大时,将育利于改善种群的多样性,从而可能有利于使算法收敛到最优解或最优解附近。Shcaeffr.JD建议的最优参数范围是:n二20一30,Pc=.075一.095,凡=.0005一.001。但在某些情况下,当种群达到一定规模时,再增大种群规模,对搜索结果的改善并无多大帮助,甚至有可能变差,因此席欲庚等经研究建议n的选择应与所求问题的非线性程度相关,非线性越大,n越大,目前常用的参数范围是:n二20一200,Pc=0.5一1.0,凡=O一0.05。在简单遗传算法(sG)或标准的遗传算法(CGA)中,这些参数是不变的。目前许多学者认识到这些参数需要随着遗传进程而自适应变化。这种有自组织性能的遗传算法具有更高的鲁棒性、全局优化性和效率,DvasiL提出一种Pc和凡随着基因操作的在线性能自适应变化的有效方法,性能提高则Pc增加,反之则凡增加;Fogarty.TC研究了凡随遗传代数变化的效果,仿真显示凡随指数下降有较好性能,席欲庚等自适应有效基因突变也有类似效果,WhitleyD提出一种自适应变异策略,凡与一对父串间的海明距离成反比,结果显示能有效保持基因的多样性。SrinivasM提出一种Pc,凡随父串的适应度自适应变化的新方法,进行了详细的理论分析和广泛的试验研究,结果显示该法在非线形和多模型问题的优化中性能优异。

(1)遗传算法从问题解集开始搜索,而不是从单个解开始。

这是遗传算法与传统优化算法的最大区别。传统优化算法是从单个初始值开始迭代求最优解的,容易误入局部最优解。遗传算法从串集开始搜索,覆盖面大,利于全局择优。

(2)遗传算法求解时使用特定问题的信息极少,容易形成通用算法程序。

在标准的遗传算法中,基本上不用搜索空间的知识或其它辅助信息,而仅用适应度函数值来评估个体,并在此基础上进行遗传操作。需要着重提出的是,遗传算法的适应度函数不仅不受连续可微的约束,而且其定义域可以任意设定。对适应度函数的唯一要求是,对于输入可计算出加以比较的正确输出。遗传算法的这一特点使它的应用范围大大扩展。

(3)遗传算法有极强的容错能力。

遗传算法的初始串集本身就带有大量与最优解甚远的信息,通过选择、交叉、变异操作能迅速排除与最优解相差极大的串;这是一个强烈的滤波过程,并且是一个并行滤波机制。故而,遗传算法有很高的容错能力。

(4)遗传算法中的选择、交叉和变异都是随机操作,而不是确定的精确规则。

这说明遗传算法是采用随机方法进行最优解搜索,选择体现了向最优解逼近,交叉体现了最优解的产生,变异体现了全局最优解的覆盖。

(5)遗传算法具有隐含的并行性。

## 2.5 移动云环境下的任务调度问题

### 2.5.1 任务调度问题描述

### 2.5.2 任务调度方式

### 2.5.3 任务调度问题的求解方法









# 第三章 多Cloudlet联合调度系统模型

集中式指派算法优点 缺点

分布式指派算法优点 缺点

优先级组





**3.1 Cloudlet系统网络模型**

本文使用图来描述Cloudlet系统网络，给出一个图, 代表了图中的K个Cloudlet节点，他们的处理能力分别为，每边上有非负权值代表两个直接相连的Cloudlet之间的通信开销。定义序列为上的任务队列中的任务序列，，任务从用户终端到的传输时延为。定义为网络时延矩阵，为到之间任务的传输时延,若和直接相连，则，否则。定义为Cloudlet的转发时延矩阵，为转发任务到的转发时延，同样地，若和直接相连，则，否则。

多用户移动任务分流模型可用一个排队网络来描述，假定中放置了K个Cloudlet,每个用户将任务分流到Cloudlet，对任意以的速率接收任务，代表K个Cloudlet任务到达速率,。

使用代表Cloudlet的平均排队时间，该Cloudlet由个速度为的主机构成，任务到达速率为，函数为著名的Erlang公式[18]。

设任务被调度到，且任务传输路径为，中间共经过m跳，在上执行时间为。

任务的响应时延为：

则上任务的平均响应时延为：

系统的平均等待时间为：

设任务序列中每个任务分别对应任务执行松弛度权重为，设为松弛度函数，则上基于松弛度优先的任务的平均响应时延为：

系统基于松弛度优先的平均等待时间为：

代表K个Cloudlet的平均负载，

使用K个Cloudlet负载的标准差作为负载均衡因子 (load balancing factor, LBF)：

综上，在Cloudlet网络G中的任务调度问题即是一个多目标优化问题，可描述如下，给定系统参数(，对系统任务进行调度，以使得系统平均响应时间，系统基于松弛度的平均响应时间和K个Cloudlet负载标准差最小，即：

# 第四章 基于优先指派和遗传算法的多目标优化算法(PA-GA)

## 4.1 算法概述

设在任意时刻，任意上存在待处理的任务序列序列，系统的调度策略分为两步：首先，对执行基于上下文的优先指派算法处理，任务序列发生变化为，接着，对各个调整后的序列采用基于遗传算法的多目标优化混合启发式算法进行任务调度到相应的cloudlet上进行处理。

本文采用的基于上下文的优先指派算法的优先指派规则为rule-X：，对于任务而言，TL表示的任务量，p(x)为的松弛度，C表示所需的资源大小，U表示所属的移动终端资源消耗。设规则rule-X的单一映射函数为,

根据问题描述，对任意任务采用基于遗传算法的多目标优化混合启发式算法调度到合适cloudlet，是一个属于NP完全问题的多目标优化问题，

## 4.2 染色体编码及种群初始化

遗传算法中常见的染色体编码方式有很多种形式，常见的如二进制编码和浮点数编码等，考虑到移动云环境下特殊的任务调度问题，本文采用直接路径表示编码方式，染色体用一维字符串表示，染色体长度为从任务接收Cloudlet节点到任务处理节点的传输路径上Cloudlet个数，即||，每个基因取值为该位置所对应的Cloudlet的字符编号，具体编码如下图所示：



该示例是一条符合编码要求的染色体，它代表编号为S的Cloudlet接收到任务，并依次通过e,F,B,a,h传输到编号为E的Cloudlet上执行。这样的染色体反应出了任务的传输路径和待分配的目标Cloudlet。

通常一般的遗传算法都会采用随机的方法产生初始种群，以保证初始种群的多样性。采用这种方式产生初始种群一般都是基于种群内的个体都是独立的个体，然而在本文中，种群中的个体并非是独立的个体，是和Cloudlet系统的网络拓扑相关的，因此，为了提高算法的效率，我们将根据高度拟合网络拓扑的方式产生初始种群，具体如下：

设当前选中Cloudlet为,与其直接相连的Cloudlet为,所对应的边的数量分别为(),则选择为下一跳的概率为：

从上式可以看出，Cloudlet相连的边越少，选中的几率越大。另外，设最后的执行目标Cloudlet为，距离的最小跳数为,很明显越大的选中几率越小，但是越大的的个数通常也会相应更多。综合考虑，本文随机生成一个随机数，则生成子个体的长度为，为网络拥塞因子，可根据网络实时状况进行调节。算法选择当前Cloudlet为第一个Cloudlet，从第二个开始，根据上一个已选Cloudlet对应边表中Cloudlet的边数，按照以上选择概率函数选择下一个编码。同时，为了避免循环，当选择了一个Cloudlet之后，将该Cloudlet从边表中移除。这样的过程反复执行次，就得到一个新个体。

如，假定=5,第一步，选择a。第二步，选择b,c,d的概率分别为9/28,9/28,10/28，假设选择d。第三步，选择c,h,j的概率分别为1/3,1/3,1/3，此处选择h。第四步，选择c,g,i,j的概率分别为16/63，15/63，16/63，16/63，假设此处选择i。第五步，选择g,j,l,m的概率分别为16/66，17/66，15/66，18/66，此处假设选择m。结束，由此产生了一个新个体为(a,d,h,i,m)。



## 4.3 适应度函数

遗传算法通过使用各个解的适应度函数值来评价该解的优劣程度，从而决定下一代的进化方向，因此适应度函数设置是否恰当尤为重要。大部分的遗传算法都是使用单一目标来设计适应度函数，然而，本文所研究的移动云环境下的多任务调度问题是一个多目标优化问题，显然传统的遗传算法适应度函数的设计思路并不适用。本文通过引入任务调度效率函数这一概念，结合移动云环境下的任务调度目标，选取任务响应时延、任务执行开销、任务成功率以及负载均衡这4个目标来量化系统对不同任务的调度效率。

**定义1** 任务调度效率函数。当任务实际调度效果与任务期望调度效果相近时，则称任务调度效率达标。设某一任务的某一个单一目标的调度效果为,任务期望的调度效果为,则任务的调度效率函数定义为：

式中为均衡常量，且，当任务实际调度效果与任务期望调度效果相等时，函数值为0，此时刚好达标。若，则说明实际调度效果值高于期望的调度效果值，反之若，则说明实际调度效果值比较期望的调度效果值低。

为了保证适应度函数非负，我们对以上介绍的调度效率函数做出进一步的变换：

设为任务的用户期望完成时间，根据任务调度效率函数的定义，任务基于响应时延的调度效率函数为：

任务执行开销会很大程度影响到系统吞吐量，是常见的QoS考虑目标之一。设对任务所使用的资源量化开销，总的执行开销为：

式中为使用的各个量化资源数量，分别为CPU，内存和带宽单位开销。设任务的期望执行开销为相应地，根据任务调度效率函数的定义，任务基于执行开销的调度效率函数为：

设根据系统监控获得的某个资源平均故障率为,任务期望的任务成功率为, 则任务基于任务成功率的调度效率函数为：

设任务的任务量为，则任务基于负载均衡的调度效率函数为：

移动云环境下的任务调度需要同时考虑多个目标，本文考虑以上列出的4个目标，从用户和系统综合考虑来说，任务响应时延越短越好，任务执行开销越少越好，任务成功率越高越好，系统负载均衡程度越高越好，因此，本文选用多目标加权法来决定任务的调度适应度函数：

式中任务各目标权重系数，且，，根据不同的任务需求，可设置相对应的权重系数，以使系统整体性能最优。

## 4.4 选择复制

遗传算法通过选择操作来促使优良染色体个体得以遗传到下一代，本文采用“轮盘赌”和精英保留机制相结合的策略。对于一个适应度为的染色体个体，其选择概率为：

累积概率为：

具体步骤如下：



## 4.5 遗传操作

### 4.5.1 交叉

交叉操作是遗传算法产生新个体的主要途径。设交叉概率为，当前种群规模为*M*，则有个个体被选择进行交叉操作。遗传算法的参数中的交叉概率和变异概率的选择是影响遗传算法行为和性能的关键所在，直接影响算法的收敛性，因此我们使用Srinvivas等提出的自适应交叉率和变异率的计算方法，交叉概率计算如下：

式中，——群体中最大的适应度值，——每代群体的平均适应度值；要交叉的两个个体中较大的适应度值；；

本文不仅要考虑Cloudlet的位置和顺序，还要考虑Cloudlet之间的连接关系，Cloudlet之间的连接关系定义为边(edge)，为了让子个体能够继承父个体中边的信息，本文使用改进的Whitley等提出的边重组(Edge Recombination,ER)交叉操作，ER操作根据继承两个父个体中Cloudlet之间的连接情况生成子个体。

考虑下面两个父个体表示：

其中(g i)属于重复的边，故在边表中将重复的边打上标记“\*”，另外，再将在两个父个体中只出现一次的边打上标记“+”，根据生成的边表，ER操作产生子个体的时候最优先考虑选择打有标记“+”的Cloudlet，次优先考虑打有标记“\*”的Cloudlet,最后才考虑其它的Cloudlet，这样可以让子个体能够充分继承父个体中边的信息。在该例子中，每个Cloudlet的边表为:

Cloudlet a: 通向其他Cloudlet的边有b +c +d;

Cloudlet b: 通向其他Cloudlet的边有a c e f g;

……

Cloudlet g: 通向其他Cloudlet的边有b c f h \*i l;

……

Cloudlet i: 通向其他Cloudlet的边有g h j +l +m;

……

Cloudlet q: 通向其他Cloudlet的边有n o p;

设随机生成的交叉产生的子个体的长度为k，子个体的第一个Cloudlet和父个体一样，从第二个开始，根据以上概率函数选择下一个编码，如果有打有标记的Cloudlet则优先选择，这样可以让子个体能够继承父个体中边的信息。这样的过程反复执行k次，就得到了子个体。

### 4.5.2 变异

变异操作是遗传算法中为了避免过早收敛，增加种群多样性的遗传算子，是产生新个体的一种辅助手段。设变异概率为，当前种群规模为*M*，则有个个体被选择进行变异操作。同理以上自适应交叉率的计算，自适应变异率的计算如下：

式中：——要变异的个体的适应度值；；

为了保证变异操作的效果，在执行变异操作的时候，我们在边表中将父个体中出现的边打上标记“-”，根据生成的边表，变异操作产生子个体的时候优先没有任何有标记“+”的Cloudlet，再考虑打有标记“-”的Cloudlet，这样可以保证子个体的多样性。

## 4.6 算法总结

### 4.6.1 算法流程



**终止条件**

# 第五章 算法仿真和实验对比

## 5.1 仿真环境

# 第六章 总结和展望

## 6.1 工作总结

## 6.2 下一步研究工作

参考文献

[1] Niroshinie Fernando,Seng W.Loke and Wenny Rahayu. Mobile cloud computing:A survey[J].Future Generation Computer Systems, 2013,29(1):184-106.

[2] Magurawalage, C. M. S., Yang, K., Hu, L. and Zhang, J. Energy- efficient and network-aware offloading algorithm for mobile cloud computing[J]. Computer Networks, 2014, 74: 22-33.

作者在读期间科研成果简介

在校期间发表论文：

1. \*\*\*，\*\*\*. 公有云环境基于路径聚簇的工作流费用优化算法[J]，现代计算机，2016（3）：8-12.
2. \*\*\*，\*\*\*，\*\*\*，\*\*\*. 基于分层的河流水下传感器网络路由算法[J]，计算机应用，已录用.

投稿论文：

1. \*\*\* ，\*\*\* ，\*\*\* ，\*\*\*.基于科学工作流分层概率聚簇的资源优化配置方法[J].系统工程理论与实践 (外审中).

声明

本人郑重声明所呈交的学位论文是本人在导师指导下进行的研究工作以及取得的研究成果。

据本人所知，除了文中特别加以标注的地方外，论文中不包含其它人已经发表或撰写过的研究成果，也不包含为获得四川大学或其它教育机构的学位或证书而使用过的材料。与本人一同工作的老师或同学对本研究所做的任何贡献均已在论文中作了明确的说明，并表示谢意。

本学位论文成果是本人在四川大学读书期间在导师指导下取得的，论文成果归四川大学所有，特此声明。

指导教师（签名）：

学生（签名）：

致谢

光阴似箭，研究生学习和生活正走向尾声。回首这三年的教育经历，我获益良多，令我终生难以忘怀。我忘不了“海纳百川，有容乃大”的博大胸襟，忘不了老师们的谆谆教诲，忘不了同学们的殷切关怀。在家人、老师、同学和朋友的鼓励和帮助下，我逐渐成为一位优秀的川大人，期望成为一位优秀的社会人。

在论文付梓之际，我借此向所有人表示致谢。

感谢川大老师，特别是导师—\*\*教授的指导和教诲。在求学过程中，我遇到过许多难题，但在老师们的帮助下，我顺利地将其解决。实验室老师教会我如何开展研究，如何进行论文写作。特别是，导师的创新思维和严瑾做事，令我心悦诚服，对我影响很深。这些研究方法和治学态度，让我终生受用。在此，感谢老师这几年来给我的帮助与关心，感谢老师对我研究方向上的指导以及对我毕业论文所付出的时间与精力。

感谢家人，特别是父母的默默奉献和支持。父亲对我说，家里的事情不用担心，让我努力学习。母亲经常对我嘘寒问暖，让我保重身体。这些叮呤嘱咐，让我多远都有家的温暖。

感谢川大同学，特别是实验室的兄弟姐妹的照顾和关心。共同学习和一起进步，让我在这三年体会到学生生活的美好。另外，感谢所有给我的安慰、劝解和鼓励的朋友们，希望未来的人生一直都有你们。

最后，感谢各位论文评审专家和答辩委员会的老师，感谢您们对我的论文的悉心指正，感谢您们的辛勤劳动。