异构云环境多目标 Memetic 优化任务调度方法

李智勇[1)] 陈少淼[1)] 杨 波[1),2)] 李仁发[1)]

[1)](湖南大学信息科学与工程学院 长沙 410082)

[2)](湖南财政经济学院 信息管理系 长沙 410205)

摘 要 云计算系统的高效能调度优化是当前重要的研究课题,面向异构云环境的多目标优化调度方法研究具有重要意义.云计算环境下的能耗和性能优化管理是 NP-HARD 的多目标组合优化问题,目前一般启发式调度系统大多采用带约束的性能或能耗的单目标优化计算方法,不能完全满足复杂云计算系统资源约束动态性与管理需求多样性的需求.基于传统进化优化的随机搜索算法应用于云环境下的 DAG 任务的多目标调度优化,计算开销大、计算实时性不足,文中提出了新的 Memetic 优化方法以解决异构云环境多目标调度优化问题.首先,文中针对异构云环境多目标调度优化问题,构建了一般性的数学定义;其次,针对该问题设计了多目标 Memetic 优化算法,采用基于解结构相关信息的 Memetic 局部搜索算子加速调度方案的局部优化能力,以提高算法的收敛速度、降低计算开销.实验结果表明,应用所提出的多目标 Memetic 优化算法进行异构云环境能耗和性能多目标调度优化,比传统方法具有更好的计算效率、解集多样性与收敛性能.

关键词 异构云平台;能耗与性能优化;DAG;多目标 Memetic 优化;云计算

中图法分类号 TP18 **DOI 号** 10.11897/SP.J.1016.2016.00377

Multi-Objective Memetic Algorithm for Task Scheduling on Heterogeneous Cloud

LI Zhi-Yong[1)] CHEN Shao-Miao[1)] YANG Bo[1),2)] LI Ren-Fa[1)]

[1)](*College of Computer Science and Electronic Engineering, Hunan University, Changsha* 410082)

[2)](*Department of Information and Management, Hunan University of Finance and Economics, Changsha* 410205)

Abstract Highly efficient scheduling optimization of cloud computing system is an important research subject, and the multi-objective optimal scheduling algorithm on the heterogeneous cloud is meaningful. The problem of energy consumption and performance optimization management on cloud is NP-hard multi-objective combinatorial optimization problem. Currently, most of general heuristic based scheduling algorithm adopt single objective optimization method which binding performance or energy consumption calculation. These methods cannot fully satisfy the dynamics constraints of complex cloud computing system resources, and the diversity of management. The computation overhead of general stochastic search algorithm based on evolutionary optimization for DAG scheduling on cloud is expensive, and real-time calculation is insufficient. For these reasons, a new Memetic optimal algorithm is proposed. In this paper, we define the problem of multi-objective scheduling optimization on the heterogeneous cloud. And then, a Multi-objective Memetic algorithm is proposed, which use memetic local search technique based on the related information of solution structure to improve the local optimization ability, this technique could improve the algorithm convergence speed and reduce the computational overhead of algorithm.

收稿日期:2014-12-21;在线出版日期:2015-07-23.本课题得到国家自然科学基金(61173107)、国家"八六三"高技术研究发展计划项目基金(2012AA01A301-01)、广东省教育部产学研合作专项资金重大专项(2012A090300003)、广东省科技计划项目(2013B090700003)资助.
李智勇,男,1971 年生,博士,教授,主要研究领域为嵌入式系统、多目标进化算法及云计算. E-mail: zhiyong.li@hnu.edu.cn. **陈少淼**,男,1987 年生,博士研究生,主要研究方向为进化计算、云计算. **杨 波**,男,1974 年生,博士研究生,主要研究方向为博弈论、云计算. **李仁发**,男,1957 年生,教授,博士生导师,主要研究领域为网络计算、嵌入式系统.

The experiment results show that the proposed method has better computation efficiency, diversity of solution set and convergence performance than traditional methods.

Keywords heterogeneous cloud platform; energy consumption and performance optimization; DAG; multi-objective memetic optimization; cloud computing

1 引 言

云计算作为一种新的商业计算模型和服务模式,是新一代高速网络计算和服务平台,随着其系统规模与计算性能不断提高,高效能的调度优化日益成为备受关注的课题.据统计,2011 年我国数据中心总耗电量达 700 亿千瓦时,占中国用电总量的 1.5%,相当于 2011 年天津市全年的总用电量.另据世界绿色和平组织的一份报告观测,到 2020 年,全球主要 IT 营运商"云计算"(包括数据处理和电信网络)的能耗将近 2 万亿千瓦时,超过德国、法国、加拿大和巴西四国的能耗总和.能耗开销已成为数据中心运营的最大开销[1-2].高的能源消耗不仅造成了高的计算开销,还会引发系统的不稳定和环境的污染.单一的提高计算性能显然已不再符合云计算发展的需求,结合计算性能与能耗的高效能优化调度是解决高能耗问题的有效途径之一.

大规模异构云计算系统的高效能优化调度是一个具有多目标、动态约束与实时偏好特征的复杂优化问题.一方面,作为一种商业计算平台,需要以实现服务利润最大化为目标之一,但同时需要满足不同的计算任务的客户需求,例如:有的客户更倾向于计算性能(越快越好),而有的客户更倾向于计算开销(越节能越好).另外一方面,复杂的云计算系统具有计算节点异构、可用资源动态变化、管理需求偏好实时变化等特征.传统意义上的简单调度策略,以及基于决策偏好预定义的单目标优化方法都不能满足其复杂的多目标优化、资源约束动态与决策偏好实时变化的计算需求.因此,研究云计算环境下的多目标优化调度方法具有重要意义.

在求解多目标优化问题时通常有 2 大类途径.其一是将多个目标通过加权求和转换为单目标的优化问题,其二是基于 Pareto 排序求解全体 Pareto 最优解集,再利用决策者实时偏好选择其中的某一折衷解.前者在转换为单目标的优化过程中,必须进行预定义的先验决策偏好权重设定,优化解单一,缺乏灵活性.而后者通过多目标优化计算,可获得一个 Pareto 最优解集,具有更灵活、广泛的决策空间.显然,在云计算环境下的能耗和计算性能的多目标优化管理问题中,通过获得该问题的 Pareto 最优解集来实现对不同客户需求下的能耗和计算性能优化与管理更符合系统需要.如图 1 所示(其中 rw 为能源消耗约束,rt 为任务完成时间约束),通过多目标优化算法获得该次调度的 Pareto 最优解集,调度器再根据系统资源的动态状态、用户任务的需求特征以及系统管理的实时需求,选择其中最恰当的调度方案,实现商业化优化管理.显然,实现上述优化调度与管理的核心关键是其中的多目标调度问题建模与高效率优化算法.

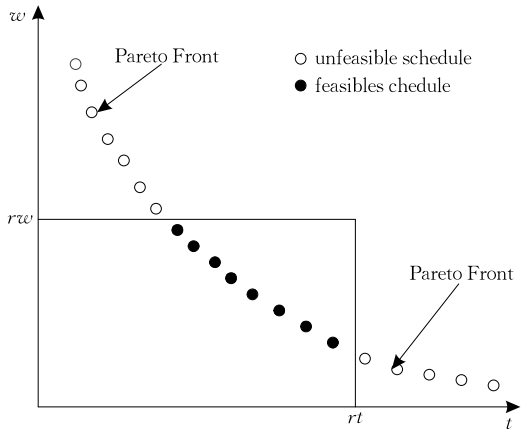


图 1 能耗与计算性能优化管理示意图

基于进化优化的随机搜索算法在求解多目标优化问题有着较好的表现,且该类算法已应用于分布式系统环境下的基于 DAG 任务图的任务调度问题,例如:文献[3-4]采用遗传算法对基于 DAG 任务图的调度问题进行研究,提出了一种满足任务间数据依赖约束关系的交叉和变异算子,取得了较传统调度算法更优的解;Swiecicka 等人[5]将遗传算法和人体免疫模型引入任务调度问题,提出了一种基于细胞自动机的任务调度算法,该算法将任务分为学习、操作和重用操作 3 个互相关联的阶段;文献[6]采用模拟退火算法求解 DAG 调度问题,该算法可避免算法陷入局部最优;Ferrandi 等人[7]将蚁群算法应用于任务到处理机的映射,获得了较好的优化效果; Xu 等人[8]将化学反应算法应用于同构集群

系统下的 DAG 任务调度问题，针对性的设计了分子碰撞、分子合成等进化操作. 上述方法取得了比传统启发式调度算法更好的解，但其计算开销通常太大且这些启发式算法大多应用于求解以提高系统性能为目标的单目标优化问题（没有考虑系统的能耗开销），不能直接应用于多目标优化问题求解.

在分布式系统的能耗优化研究方面，近年来学者主要进行的系统能耗评估模型和节能调度算法的研究. Elnozahy 等人[9]对计算机功率和处理器工作状态的关系进行了研究，并提出了相应的计算式；Bahsoon[10]总结了云环境下兼顾能源和可靠性的自优化框架所面临的问题，提出用 QoS/能耗作为衡量标准；林闯等人[11]将能量看成一种系统资源，提出了基于随机模型的绿色评价框架，为构建绿色网络和节能机制的评价体系奠定了基础；Li 等人[12-14]分别在同构、异构多处理器环境下研究了系统在一定时间约束下采用连续电压调节技术使得能耗最低的条件，并相应提出了节能调度算法；宋杰等人[15]提出了一种云计算环境下的能效模型和度量方法，定义了能效的数学表达及其测量和计算方法，通过 CPU 使用率和频率来计算能效，简化了能效测量方法；Cao 等人[16]针对云数据中心的能耗分布和负载均衡问题进行了数学定义，分别对能耗约束下的性能优化和时间约束下的能耗优化问题进行了分析，提出了相应的解决方案来优化云数据中心的能耗；谭一鸣等人[6]采用排队论模型 M/M/1 对云计算系统进行建模，分析云计算系统的平均响应时间和平均功率，建立云计算系统的能耗模型，提出了满足性能的最少期望执行能耗的调度算法 ME3PC；Zong 等人[17]对同构集群系统的能耗和性能建模，针对该模型对已有的复制任务调度算法添加能耗感应机制，提出了面向同构集群系统的 EAD 和 PEBD 节能调度算法，李新等人[18]又针对该算法进行了改进，提出了一种启发式处理器合并优化方法 PRO. 目前这些研究大多采用在传统调度算法的基础上添加能耗感知机制，从而达到减少其空闲能耗开销或不合理调度产生的能耗浪费，提高系统的能效比率的目的，但较少考虑云计算环境下不同客户需求等特性.

本文首先对云计算环境下的能耗与计算性能的多目标优化问题进行了数学定义，其中考虑了云计算的异构、动态伸缩特性对系统性能和能耗的影响. 在调度算法研究方面，主要对以往随机搜索算法在 DAG 任务调度中计算开销大、收敛速度慢的缺陷进行了研究，提出了多目标 Memetic 优化算法（Multi-objective Memetic Algorithm，MOMA），采用基于解结构信息的 Memetic 局部搜索算子来进行局部寻优，设计相对应的预评估机制减少对调度策略进行整体评估，提高算法收敛速度，从而达到减少算法计算开销的目的. 此外，提出了基于 DAG 图层次的交叉算子，有利于个体间的信息交互. 模拟实验表明，该算法能实现云计算系统下的能耗和计算性能的优化管理，且本文提出的算法比传统的多目标优化算法具有更好的收敛性能.

本文第 2 节对云环境下的任务多目标调度问题模型进行阐述；第 3 节给出了本文提出的面向异构云环境下能耗与性能的多目标 Memetic 优化算法；第 4 节进行了仿真实验与性能分析；最后一节对本文所做的工作进行总结.

2　云环境下多目标优化调度模型

2.1　云计算环境下任务调度体系结构

云计算系统的任务调度体系结构可如图 2 所示，其包括用户任务、任务调度中间件和异构计算机群. 任务调度中间件是其中核心模块，包括计算任务 DAG 生成、优化调度方案决策及计算任务分配 3 大功能，任务调度中间件的一个重要目标就是尽可能的满足客户的服务水平协议（Server Level Agreement，SLA），同时提高计算系统的服务质量（Quality of Service，QoS）.

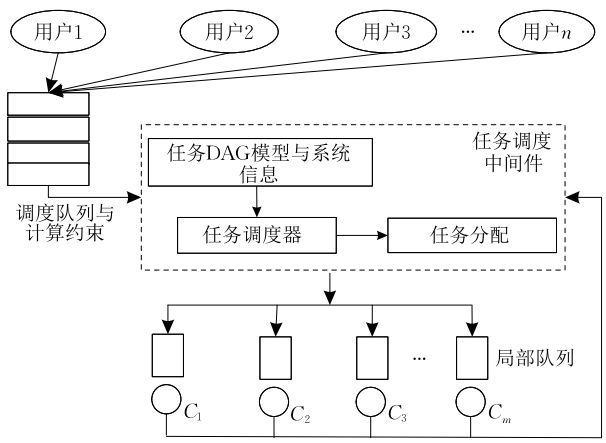
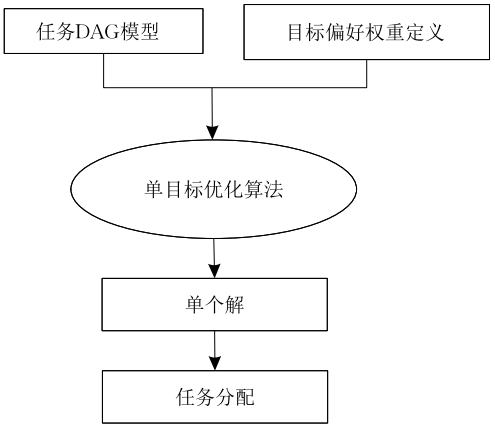


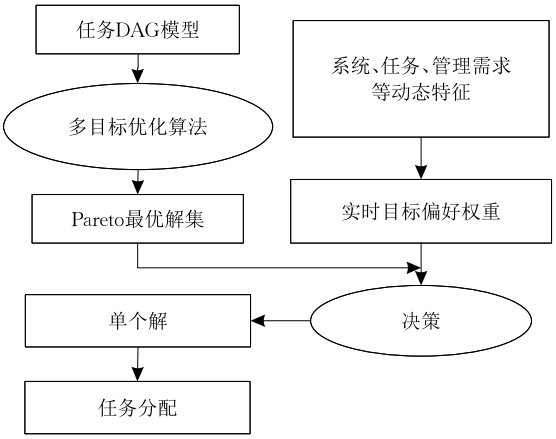
图 2　云环境下的任务调度体系结构

当面向能耗、性能等多个目标的调度优化计算时，任务调度中间件可采用图 3 中的两种计算模式. 图 3(a)是传统的采用预定义偏好加权求和将多个优化目标转化为单目标优化的计算模式，其中的优化目标偏好权重需要预先设定，一般优化结果为单

一解，决策空间有限，无法实现多样化的实时动态偏好决策. 而另一种优化流程可采用图 3(b)的方式，首先将调度问题定义为多目标优化问题，通过高效率的多目标优化算法进行优化计算后，获得整个决策空间的 Pareto 最优调度解集(所有优化目标无偏好的折衷非劣解集合)，然后再根据当前云计算系统的资源动态约束、系统稳定性需求、服务系统利润偏好以及系统管理偏好等产生综合实时决策偏好，决策产生 Pareto 折衷解作为最终调度方案. 这种计算模式较第一种方式具有更灵活的决策选择空间，适合复杂大规模云计算系统的动态特性，但对多目标优化算法性能要求高，是其中的核心关键部分. 以下将重点针对图 3(b)中云环境下能耗和性能的多目标优化算法进行研究.



(a) 基于偏好加权求和的单目标优化的调度决策流程



(b) 基于Pareto多目标优化的调度决策流程

图 3 调度中间件的决策流程

2.2 DAG 任务模型

分布式系统下的静态任务调度问题中，并行应用程序大多以 DAG 任务图模型进行表述，例如，图 4 为具有 10 个子任务的 DAG 任务图. 具有 n 个子任务的 DAG 任务图可定义为

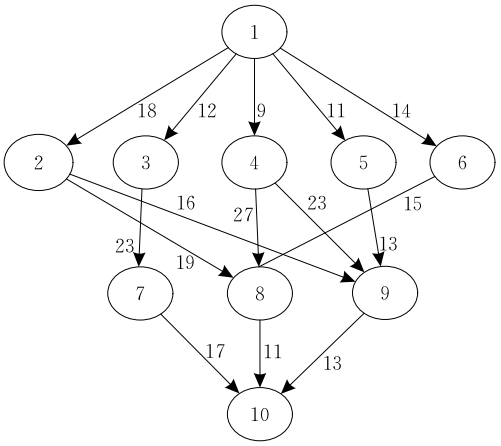


图 4 一个任务数目为 10 的 DAG 任务图

定义 1. DAG 任务图可以表示为一个三元组：$G=\langle V,E,\boldsymbol{L}\rangle$. 其中 $V=\{v_i\mid 1\leq i\leq n\}$表示任务的集合；$E=\{e_{ij}\mid 1\leq i\leq n,1\leq j\leq n\}$表示任务间具有约束关系的边的集合，其中，$e_{ij}$ 边上的数值(W_{ij})表示任务 v_i与任务 v_j的通信需求；$\boldsymbol{L}$ 表示任务的服务长度向量，L_i表示任务 v_i的服务长度. 此外 $succ(v_i)$为任务 v_i的直接后继任务集，$pred(v_i)$为任务 v_i的直接前驱任务集.

2.3 异构云计算系统模型

由于不同客户的计算需求和数据中心设备的不断更新，现有云计算平台的硬件基础设施通常由大量异构的服务器组成，这些异构处理机含有不同的计算性能和能效比，且一般同一个任务在计算性能高的处理机上运行比运行在计算性能较低的处理机上会产生更多的能耗开销. 例如：Intel Pentium G840(C1)的频率为 2.8 Hz，其执行功率为 67.7 W，空闲功率为 42.1 W；而 AMD Fx-6200(C2)的频率为 4.1 Hz，其执行功率为 204 W，空闲功率为 63 W. 如果一个服务长度为 1 的任务分别在这两种处理机上满负载运行，由于频率与处理机的运行速率成正比，则该任务在这两种处理器上的完成时间比例约为 3∶2，能耗比例约为 1∶2，也就是说，如果将任务从 C1 调度到 C2 将提高 33.3%的处理速度，但要增加 100%的能耗开销. 由此可知，高的计算性能是以巨大的能耗开销为代价的，将任务进行合理的调度可以减少能耗开销. 基于此，本文将针对异构云计算环境进行研究. 该异构云计算环境可定义为

定义 2. 具有 m 个处理机的异构云计算系统可定义为一个五元组：$\langle C,S,P,\boldsymbol{B},Q\rangle$. 其中 $C=\{c_i\mid 1\leq i\leq m\}$是异构处理机集合，$c_i$表示第 i 个处理机；$S=\{S_i\mid 1\leq i\leq m\}$是处理机平均服务速率的集合，$S_i$

表示处理机 c_i 的平均服务速率；P 为处理机的功率，其中 P_i 表示处理机 c_i 的平均执行功率，$P_{i,low}$ 为处理机处于空闲状态的功率；$\boldsymbol{B}$ 为处理机间的通信速率矩阵；$Q=(Q_1,\cdots,Q_m)$，其中 Q_i 表示处理机 c_i 开始通信的准备时间.

2.4　任务调度长度

任务调度长度是评价系统计算性能的有效方式，在基于 DAG 任务图的调度问题中，必须考虑任务间的数据依赖约束关系和处理机的可调度性. 设 $EST(v_i)$ 表示任务 v_i 最早开始执行时间，$EFT(v_i)$ 表示任务最早执行完成时间，则 $EST(v_i)$ 与 $EFT(v_i)$ 存在如下关系：

$$EFT(v_i)=EST(v_i)+L_i/S_{proc(v_i)} \tag{1}$$

其中，$proc(v_i)$ 为任务 v_i 的目标处理机. 设任务 v_i 与 v_j 的通信开销为 CC_{ij}，则

$$CC_{ij}=\begin{cases}0, & k=u\\ Q_k+\dfrac{W_{ij}}{B_{ku}}, & k\neq u\end{cases} \tag{2}$$

其中 $k=u$ 表示任务 v_i 与 v_j 在同一个处理机上. 由此，任务 v_i 最早开始执行时间 $EST(v_i)$ 可表示为

$$EST(v_i)=\begin{cases}0, & pre(v_i)=\varnothing\\ \max\{available(proc(v_i)), \\ \quad \max\limits_{v_j\in pre(v_i)}\{EFT(v_j)+CC_{ji}\}\}, & \text{其他}\end{cases} \tag{3}$$

其中，$avaible(proc(v_i))$ 表示 v_i 目标处理机的可调度时间.

最后，任务的调度长度（$makespan$），即任务的完成时间，可表示为

$$makespan=\max_{v_i\in V}\{EFT(v_i)\} \tag{4}$$

2.5　能耗分析

处理机产生的能耗是计算系统产生能耗的主要部分，而处理器产生的能耗主要分为执行能耗 EB（处理机处于运行状态下产生的能耗）和空闲能耗 EI（处理器处于空闲状态下产生的能耗）. 设 EB_j 为处理机 c_j 的执行能耗，则

$$EB_j=\sum_{i=1}^{n}U_{ij}\times\left(\frac{L_i}{S_j}\right)\times P_j \tag{5}$$

其中 $\boldsymbol{U}=\{U_{ij}\mid 1\leq i\leq n,\ 1\leq j\leq m\}$ 表示分配矩阵，$U_{ij}\in\{0,1\}$，为 1 表示任务 v_i 分配到处理机 c_j 上. L_i/S_j 是任务 v_i 在处理机 c_j 上的执行时间. 从而系统总的执行能耗为

$$EB=\sum_{j=1}^{m}\sum_{i=1}^{n}U_{ij}\times\left(\frac{L_i}{S_j}\right)\times P_j \tag{6}$$

处理机处于空闲状态也存在能耗开销（也就是空闲能耗），处理机的空闲能耗等于空闲功率乘以空闲时间，考虑到云计算动态（伸缩）的特征，当没有任务调度到某个节点上时，可放弃对该节点的使用，处理机 c_j 的空闲功耗 EI_j 可表示为

$$EI_j=\begin{cases}\left(makespan-\sum\limits_{i=1}^{n}U_{ij}\times\left(\dfrac{L_i}{S_j}\right)\right)\times P_{j,low}, & \sum\limits_{i=1}^{n}U_{ij}\neq 0\\ 0, & \text{其他}\end{cases} \tag{7}$$

其中，$\sum\limits_{i=1}^{n}U_{ij}\neq 0$ 表示处理机 c_j 上存在调度任务，否则，不存在调度任务. 由此计算系统总的空闲能耗 EI 可定义为

$$EI=\sum_{j=1}^{m}EI_j \tag{8}$$

最终系统的总能耗（$energy$）为

$$energy=EB+EI \tag{9}$$

2.6　问题定义

基于上述计算性能和系统能耗开销的分析，面向云环境下能耗与计算性能的多目标优化问题可定义为

$$\begin{cases}\min:\{makespan,energy\}\\ \text{s.t.}\ \sum\limits_{j=1}^{M}U_{ij}=1,\ i=1,2,\cdots,N\\ \quad EST(v_i)\geq EFT(v_j),v_j\in pred(v_i)\end{cases} \tag{10}$$

其中约束条件 $\sum\limits_{j=1}^{M}U_{ij}=1,i=1,2,\cdots,N$ 保证了所有任务都被调度且每个任务仅被调度一次，约束条件 $EST(v_i)\geq EFT(v_j),v_j\in pred(v_i)$ 保证了任务调度符合任务间的约束关系.

3　面向异构云环境下能耗与性能的多目标 Memetic 优化算法

应用于 DAG 任务调度的随机搜索算法，虽然取得了比传统启发式算法更优的解，但其存在算法计算开销大的缺陷，且大多应用于提高系统性能的单目标优化问题. 如何减少该类算法的计算开销，并能够应用于求解融合能耗和性能的多目标优化调度问题是接下来我们主要探讨的问题.

经过研究发现，造成该类算法在 DAG 任务图调度优化问题中计算开销大的主要原因包括两方面：高的个体（一种调度安排）评估代价和低下的算法搜索效率. 个体的评估在基于进化优化的随机搜索算法中是必不可少的，且每一代个体的更新都要

进行个体适应度的评估.在基于迭代搜索的面向DAG任务图的调度优化中,个体的评估时间复杂度为$O(TN\overline{N})$,其中N为任务的个数,$\overline{N}$为每个任务的平均直接前驱任务个数,T为算法的迭代次数;另一方面,在该问题的解空间搜索过程中,个体必须满足任务间的数据依赖约束关系与处理机的可调度性,使得在搜索过程中产生大量的非法解,严重影响了算法的搜索效率.本文试图采用基于局部预评估的Memetic局部寻优来减少个体的总体评估提高算法的搜索效率,同时采用基于DAG任务图层次的交叉变异操作来避免产生非法解,最终实现减少算法计算开销的目的,为此,提出了面向云计算环境下能耗与性能优化的多目标Memetic调度优化方法(Multi-Objective Memetic Algorithm,MOMA).

2000年以来Memetic算法逐渐受到研究者们的认可,IEEE、CEC、GECCO、PPSN等进化计算领域的主流国际会议陆续的组织了MA(Memetic Algorithm)的专题,MA作为一个开放式的进化计算框架,易于把多种局部/全局搜索策略结合起来,具有算法收敛速度快的优点.在离散优化领域,MA算法的研究较少,现有的一些研究主要集中于Memetic局部搜索策略的研究[19-21],其方法大多采用对特定的离散问题(例如:车间调度优化、路径优化等)设计针对性的Memetic局部搜索策略的方式来提高算法的收敛性能,较少应用于面向DAG任务图的调度优化问题,且这些方法主要针对单目标优化问题,较少应用于多目标优化.

融合Memetic局部优化与多目标进化优化方法的多目标Memetic算法的总体框架可如图5所示.其中,$A(t)$是第t代的归档集,$R(t)$和$R'(t)$为第t代种群.该框架主要包括个体进化、局部优化、归档集更新等操作.

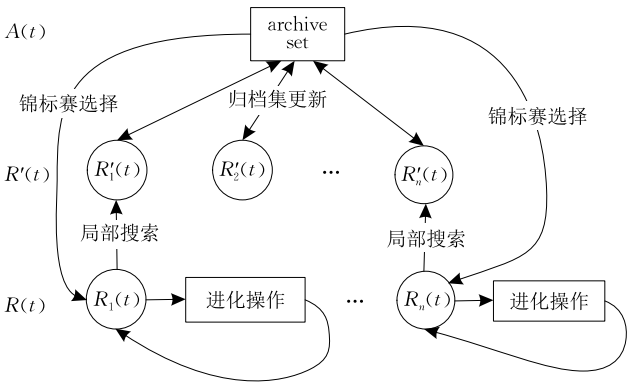


图5 多目标Memetic优化算法总体框架

面向异构云环境下性能和能耗优化管理的多目标Memetic优化算法总体流程如下:

1. 初始化云计算环境参数(产生任务DAG图,获取云处理机参数,提取DAG任务图各层次任务子集;

2. 随机产生可调度序列,初始化种群$R(0)$,外部归档集$A(0)$,进化代数$t=0$;

3. 判断是否满足终止条件,如果满足则输出归档集中的最终解,算法结束;

4. 对种群$R(t)$进行适应度值评估;

5. 对种群$R(t)$中非支配解进行Memetic局部搜索,产生$R'(t)$;

6. 种群$R'(t)$和$A(t)$更新生成$A(t+1)$,$t=t+1$;

7. 对$A(t)$执行锦标赛选择操作,并将结果保存到$R(t)$中.然后对种群$R(t)$进行基于DAG层次的交叉操作和变异操作,产生新的子种群$R(t)$,转到步3.

该流程中的终止条件可为一定的个体评估次数或运行时间.步6中的归档集更新操作采用文献[22]中的非支配集拥挤距离排序方法.步2中的种群初始化,步5中的Memetic局部搜索策略和步7中基于DAG层次的交叉和变异算子将在接下来的内容中详细介绍.

3.1 个体编码与种群初始化

在静态调度问题中,任务的一次调度包括确定任务的调度顺序和目标处理机的映射.本文中,种群中的个体(一种调度方案)的一个基因(s_i)由任务编号和其目标处理机共同组成(如图6所示),图中$Z(s_i)$表示基因s_i位置对应的任务编号,$Y(s_i)$为基因s_i的目标处理机编号.

$Z(s_1)$	$Z(s_2)$	...	$Z(s_{n-1})$	$Z(s_n)$
$Y(s_1)$	$Y(s_2)$	...	$Y(s_{n-1})$	$Y(s_n)$

图6 个体编码示意图

在种群初始化过程中,先采用文献[23]中的任务优先级计算方法,对DAG任务图中的任务根据优先级进行降序排序.当任务v_i存在直接后继任务时,任务v_i的优先级可表示为

$$rank(v_i)=\frac{1}{m}\left(\sum_{j=1}^{m}\frac{L_i}{S_j}\right)+\max_{v_j\in succ(v_i)}(\overline{C_{ij}}+rank(v_i)) \tag{11}$$

其中$\overline{C_{ij}}$为任务v_i与直接后继任务的平均通信开销.当任务v_i没有直接后继任务时,任务v_i的优先集为

$$rank(v_i)=\frac{1}{m}\left(\sum_{j=1}^{m}\frac{L_i}{S_j}\right) \tag{12}$$

对任务优先级降序排序后,依照任务调度顺序,依次将任务随机调度到处理机上,产生种群规模大小的满足任务间数据依赖约束关系的随机个体.

3.2　Memetic 局部搜索策略

Memetic 局部搜索策略主要通过利用解的结构相关信息进行领域的局部搜索来提高随机搜索算法的搜索效率. 目前虽然已存在较多的 Memetic 局部搜索策略，但它们并没有考虑 DAG 任务图多目标调度问题的独有特性，本文对 DAG 任务图调度问题进行分析，提出了一种针对 DAG 任务图调度问题的 Memetic 局部搜索策略，并采用预评估方法，减少对个体的评估次数.

在面向 DAG 任务图的调度问题中，一次调度安排的任务调度长度由部分子任务决定，我们称这部分任务为关键任务(设关键任务集为 CP)，关键任务生成的路径称为关键路径，而 DAG 任务图中的其它子任务称为非关键任务. 如图 7(图 4 中的 DAG 任务图在两种不同调度情况下生成的不同关键路径). 关键任务决定着 DAG 任务图在该次调度中的完成时间，降低关键任务的最早完成时间，将极大可能减少 DAG 任务图在该次调度中的完成时间，而非关键任务将对 DAG 任务图的完成时间相对影响较少. 由此可知，在一个个体中，部分变量(关键任务映射的基因)对任务的完成时间较为敏感，适合于对任务的完成时间和空闲能耗进行局部优化，而另外一部分变量(非关键任务映射的基因)对任务完成时间的影响相对较弱，可以用于对不合理调度产生的能耗进行局部优化.

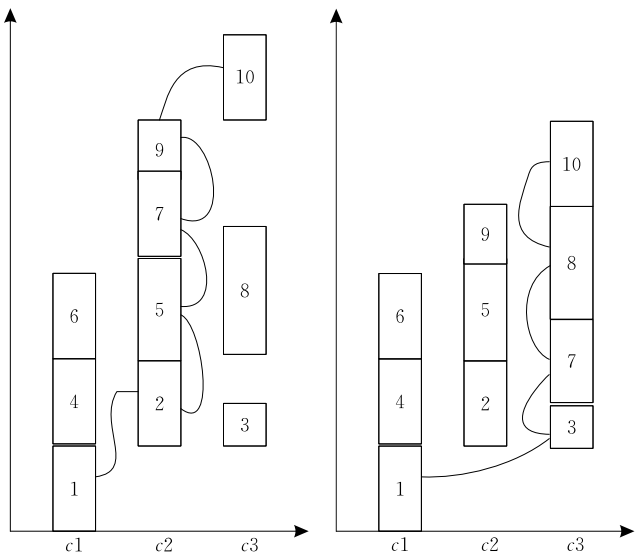


图 7　不同调度下的关键路径

受此启发，本文将对关键任务和非关键任务进行相对应的局部寻优. 对于关键任务调度调节到能减少其最早任务完成时间的处理机上，也就是对关键任务 v_i 对应的目标处理机进行调节，当 $EFT(v_i,c_j)<EFT(v_i,c_{\text{new}})$ 时，再进一步进行 DAG 任务图的整体评估，如果新的个体支配原来的个体，则接受该次调节；否则，不接受. 其中 $EFT(v_i,c_j)$ 表示任务 v_i 调度到处理机 c_j 上的的最早完成时间；而非关键任务调节到使其能耗开销减少又不增加其直接后继中关键任务的最早开始时间的处理机上，也就是对非关键任务 v_i 的目标处理机进行调节，当 $EC(v_i,c_{\text{new}})<EC(v_i,c_j)\ \&\&\ EST(v_i,v^*,c_{\text{new}})\leq EST(v_i,v^*,c_j)$，$\forall v^*\in\{succ(v_i)\cap CP\}$ 时，再进一步进行 DAG 任务图的整体评估，如果新的个体支配原来的个体，则接受该次调节；否则，不接受. 其中 $EC(v_i,c_i)$ 表示任务 v_i 调度到处理机 c_j 上的能耗开销，$EST(v_i,v^*,c_j)$ 表示任务 v_i 调度到处理机 c_j 上时直接后继任务 v^* 的最早开始时间.

本文提出的 Memetic 局部搜索策略主要通过上述的局部预评估方式来减少对 DAG 任务图的整体评估次数，同时提高算法的收敛速度. Memetic 局部搜索策略的具体流程算法 1 所示，其中 pop，$newpop$ 为种群，$Drank(x)=1$ 表示个体 x 为非支配解，x' 为 x 局部寻优后产生的新个体，$x'\succ x$ 表示个体 x' 支配 x. 同时考虑到算法的计算开销，和已有相关研究证明了在好的解的周围有更大的概率产生好的解，本算法只对当前种群的非支配解进行局部搜索.

算法 1.　Memetic 局部搜索策略.

输入：pop，CP

输出：$newpop$

```
For each x∈pop && Drank(x)=1
  随机选择一个基因 s_i
  IF V(s_i)∈CP THEN
    FOR each c_j∈C
      IF EFT(V(s_i),c_j))<EFT(V(s_i),Y(s_i)) THEN
        IF x'≻x
          x←x'
        END IF
      END IF
    END FOR
  ELSE
    FOR each c_j∈C
      IF EC(V(s_i),c_j)<EC(V(s_i),Y(s_i))&
       EST(V(s_i),v*,c_j))<EST(V(s_i),v*,Y(s_i)),
       ∀v*∈{succ(V(s_i))∩CP} THEN
       IF x'≻x
         x←x'
       END IF
      END IF
    END FOR
  END IF
END FOR
newpop←pop
```

3.3 交叉与变异算子

交叉操作主要是通过个体间信息的交换来产生更好的个体. 本文提出的基于 DAG 任务层次的交叉算子,首先需对 DAG 图进行分层处理(该操作可在算法初始化阶段完成),图 4 中的 DAG 任务图分层处理如图 8 所示. 同一层次中的任务间不存在数据依赖关系,且对相邻层次的任务有着较大的影响. 同一层次的任务所产生的调度序列可视为总的任务调度序列的一个信息单元. 在交叉算子中,随机选取两个不相邻的信息单元进行交叉操作. 例如,假设图 4 中的 DAG 任务图调度到 3 个处理机上,其交叉流程如图 9 所示,s_1、s_2分别为父代的两个个体,在这次操作中随机选取了 DAG 任务图的层次 2 和层次 4 进行交叉操作,即选择层次任务对应的处理机进行交换,得到新的个体 s_1' 和 s_2'. 这种交叉方式将促使个体间更好的进行信息交互.

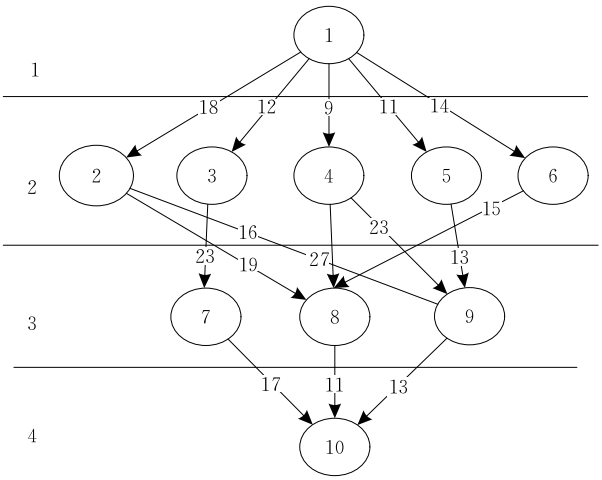


图 8 DAG 任务图的分层处理

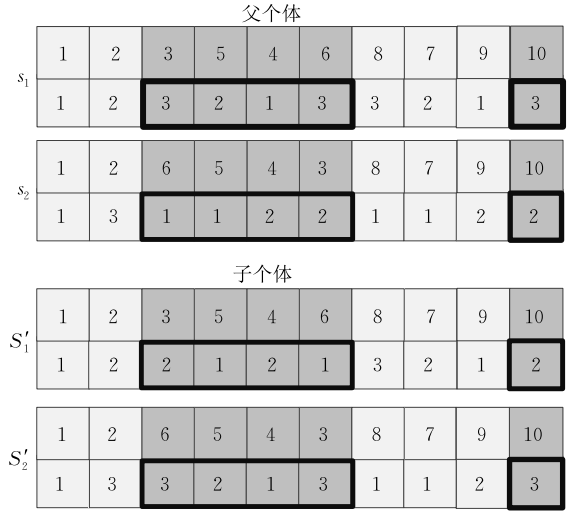


图 9 交叉操作示意图

变异操作主要是为了保持种群的多样性. 在本文提出变异算子中,以一定变异概率选中某个基因,如果该基因对应的任务存在其他同一层次的任务,则随机选择一个,交换两者的位置,并随机将该任务调节到其他处理机上,如果没有,就仅随机改变该任务的目标处理机.

3.4 算法复杂度及收敛性分析

3.4.1 算法复杂性分析

在算法复杂度方面,MOMA 算法以一定的个体评估次数作为终止条件,所以在与传统的多目标优化算法[22]进行算法复杂度对比分析时,个体评估带来的算法复杂度可以不计算其中. MOMA 算法相对传统的多目标算法主要多了 DAG 任务图分层处理和局部搜索两个操作,其中,DAG 任务图分层处理的时间复杂度为 $O(N)$,其中 N 为任务的个数. 局部搜索算子中主要增加了局部优化预评估带来的计算开销,其计算复杂度为 $O(\overline{N}PM)$,小于种群适应度评估的计算复杂度 $O(N\overline{N}P)$,因为 $M\ll N$,其中 P 为种群规模,M 为计算资源数目. 由此可知,MOMA 算法虽然相对传统多目标进化算法增加了一定计算开销,但数量级别上是一样的.

3.4.2 算法收敛性分析

定义 3. 称 x'是从 x 通过进化过程可达的,如果

$$Pr(CM\_LS(x)=x')>0 \tag{13}$$

$CM_LS(x)$为 x 通过进化过程产生的个体,$Pr(A)$表示随机事件 A 发生的概率.

引理 1. 若某进化算法满足以下两个条件[24],则该算法以概率 1 收敛到全局最优解集.

(1) $\forall x,x'\in I$(I 为可行域)x'可以通过 x 进化过程可达.

(2) 整个种群的序列 $R(t)$是单调的.

定理 1. 全局收敛性. MOMA 算法以概率 1 收敛到问题的全局最优解集.

证明. (1) MOMA 算法中任意两个个体 x 和 x'通过进化过程可达的.

令 $x=\{x_1,x_2,\cdots,x_n\}$,设 $x'=\{x_1',x_2',\cdots,x_n'\}$,设交叉和变异概率分别 τ 和 σ,y 是 x 进行交叉变异后产生的个体,m 为处理机的个数. 当 $m>1$ 时

$$Pr(x_i=y_i)=\tau(1-\sigma)\frac{1}{m-1}+(1-\tau)\sigma\frac{1}{m-1}+\tau\sigma\frac{m-2}{m-1}\frac{1}{m-1}=\frac{1}{m-1}\left(\tau-\tau\sigma+\sigma-\frac{\tau\sigma}{m-1}\right) \tag{14}$$

因为 $0\ll 1$,$0<\sigma<1$,所以,$\tau-\tau\sigma=\tau(1-\sigma)>0$,

$\sigma-\frac{\sigma\tau}{m-1}=\frac{(m-\tau)\sigma}{m-1}>0$，所以 $Pr(x_i=y_i)>0$，从而

$$Pr(CM(x)=y)=\prod_{i=1}^{n}\frac{1}{m-1}\left(t-ts+s-\frac{ts}{m-1}\right)>0 \tag{15}$$

又局部搜索概率为 $0<\beta<1$，则

$$Pr(LS(y)=x')=\prod_{i=1}^{n}\frac{\beta}{m-1}>0 \tag{16}$$

所以

$$Pr(CM\_LS(x)=x')\geq Pr(CM(x)=y)\times Pr(LS(y)=x')=\prod_{i=1}^{n}\frac{\beta}{(m-1)^2}\left(\tau-\tau\sigma+\sigma-\frac{\tau\sigma}{m-1}\right)>0 \tag{17}$$

又 $m=1$ 时，$Pr(CM\_LS(x)=x')=1>0$ 所以 x' 可以通过 x 进化过程可达.

(2) 再证整个种群的次序 $R(t)$ 是单调的. $A(t)$ 和 $R(t)$ 分别是第 t 迭代的归档集和种群，由于 $A(t+1)=M_f(R(t)\bigcup A(t),\succ)$，$R(t+1)\subseteq A(t+1)$ 可知 $A(t)$ 和 $R(t)$ 都是单调的.

综合(1)、(2)可知，MOMA 算法以概率 1 收敛到问题的全局最优解集. 证毕.

4　实验与分析

本文采用模拟实验来验证和评价 MOMA 算法的性能. 在实验过程中建立了由 Intel、AMD 等 32 种处理机构成的可扩展的异构模拟环境①，每组实验都是从 32 种计算资源中随机挑选若干种类作为计算资源.

实验中的 DAG 任务图由应用程序随机产生. 在 DAG 任务图生成过程中，先给定任务的数目和 DAG 图的并行因子(λ). 任务的计算需求是 $L_{\min}$(最少计算需求)到 $L_{\max}$(最大计算需求)中的一个随机数；DAG 任务图的高度由均值为 $\frac{\sqrt{N}}{\lambda}$ 的均匀分布随机生成，λ 的值越少 DAG 任务图的并行度越低；而任务间的通信开销由符合均值为 $\frac{1}{m}\left(\sum_{j=1}^{m}\frac{L_i}{S_j}\right)$ 的均匀分布随机生成；DAG 任务图中的子任务与下一层子任务间采用 0.5 的概率随机产生依赖关系.

在每次实验中，考虑任务的个数、处理机的个数和 DAG 任务图的并行因子对调度算法性能的影响. 算法得相关参数和模拟场景相关参数如表 1 所示.

表 1　仿真实验参数设定

参数		参数设置	定义
DAG 生成相关参数	N	32,64,128,256	任务数量
	M	4,8,16,32	处理机个数
	λ	0.5,1,2,5	并行因子
算法参数	τ	0.9	交叉概率
	σ	0.1	变异概率
	Gen	30000	解评估次数
	$Popsize$	30	种群大小

4.1　MOMA 算法与单目标优化算法比较

在该部分实验中，主要将 MOMA 算法与经典的 HEFT 算法[23]和较新颖的能耗感知调度算法 PRO 算法[18]进行比较，来验证 MOMA 算法的优化效果. 由于 MOMA 算法是多目标优化算法，寻优结果为一个 Pareto 解集，而 HEFT、PRO 算法的寻优结果只有一个解，为了更公正的比较算法之间的性能，将不直接对 3 个算法的结果进行比较. 其实验结果比较示意图如图 10 所示. 实验中主要进行 3 个方面的考虑：(1) Pareto 解集中支配单目标算法最优解的解的个数，如图 10 中虚线与坐标轴所围成区域中解的个数；(2) 单目标算法(HEFT 或 PRO)最优解 x 与 Pareto 解集中参考解 y 进行比较，来验证 MOMA 算法与目标算法(HEFT、PRO)在相似能耗开销下优化任务调度长度的能力，其中 y 是 Pareto 解集中能耗开销小于 x 且能耗开销与 x 最相近的解；(3) x 与 Pareto 解集中的参考解 z 进行比较，来验证 MOMA 算法与目标算法 HEFT、PRO 在相似的任务完成时间下算法对系统能耗开销优化的能力，其中 z 是 Pareto 解集中调度长度小于 x 且调度长度与 x 最相近的解.

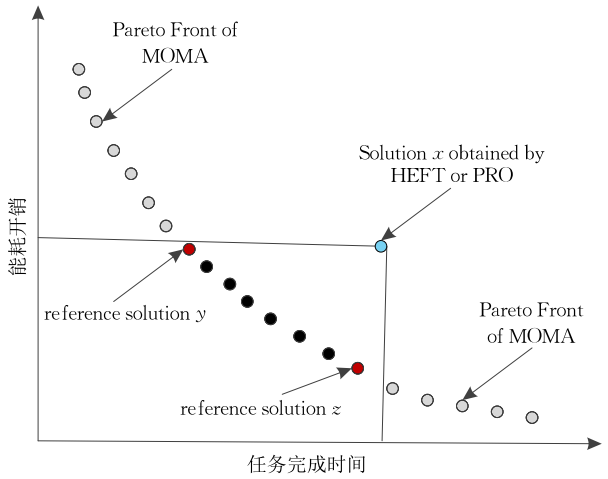


图 10　MOMA 与 HEFT、PRO 优化效果比较示意图

① http://www.xbitlabs.com/articles/cpu/

按照上述方法针对不同的随机 DAG 任务图进行反复 50 次实验，其实验结果如表 3 所示. 表中的实验数据都是 50 次实验后统计的平均值. 从表 3 中可以发现，在任务的完成时间方面，MOMA 算法在不同 DAG 任务图规模和并行因子、处理机个数下，都能获得比 HEFT 和 PRO 算法更好的优化效果，相对于 HEFT 算法，MOMA 算法任务完成时间平均减少 11.46%，相对于 PRO 算法，MOMA 算法任务完成时间平均减少 11.63%. 在能耗开销方面，相对于没有考虑能耗开销的 HEFT 算法，MOMA 算法存在较大的优势（平均减少 14%），而对于 PRO 算法，MOMA 算法也平均减少了 7.84%的能耗开销，这主要是由于 MOMA 算法考虑了通过自适应进化的方式来调节处理机的数量，以及异构计算环境下合理调度对能耗开销的影响，而 PRO 主要是通过减少处理机的数量来减少能耗开销. 由此可知 MOMA 算法相对传统启发式算法（HEFT、PRO）具有更好的优化能力和普适性.

在算法的计算开销方面，我们在各个不同环境下，对 HEFT、PRO、MOMA 算法的计算开销进行统计分析，其平均值（sec.）分别为 0.007 756、13.536 845、75.358 3，可以发现 HEFT 算法的计算开销是最少的，同时 PRO 和 MOMA 算法的计算时间都远远高于 HEFT 的，这主要是由于 PRO 算法属于复制任务调度算法，它需对各个子任务的调度状态进行扫描且存在大量迭代计算，而 MOMA 算法是随机搜索算法，需对解空间反复搜索和评估，但 MOMA 是基于群智能优化的，具有较好的并行性，其服务的对象也是一种并行计算系统，采用并行编码技术可以数倍甚至数十倍的减少算法的计算时间，从而弥补算法在计算开销方面的不足.

4.2 MOMA 算法与多目标优化算法比较

本组实验主要目的是验证 MOMA 算法的多目标收敛性能与算法的计算复杂度. 由于目前面向云环境下性能和能耗优化的多目标优算法尚不多见，所以本研究实验中，将经典的多目标算法 MOEA[25] 进行改进（其中 MOEA 算法的进化操作采用文献[4]的交叉变异策略），然后再与 MOMA 算法进行比较.

算法的收敛性能评估指标采用 S 指标和 C 指标[26]. S 指标主要依据 $S(\chi')$ 来衡量非支配解集 χ' 的收敛性能，$S(\chi')$ 为 χ' 对应目标空间的超立方体的体积，由于各个算法获得的非支配解的个数不同，本文采用平均 $S(\chi')$，即为 $S(\chi')$ 除以算法获得非支配解的个数. 收敛指标 C 可定义为

$$C(\chi'',\chi') := \frac{|\{\alpha''\in\chi'';\exists\,\alpha'\in\chi':\alpha'\succeq\alpha''\}|}{|\chi''|} \tag{18}$$

其中 $\chi',\chi''\in\chi$ 为决策变量的两个解集. C 的取值在 0～1 之间，当 $C(\chi'',\chi')=1$，说明 χ'' 中的解都被 χ' 支配或相等，当 $C(\chi'',\chi')=0$，说明 χ'' 中的所有解都不被 χ' 支配或相等. 在对比实验中，本文将 $C(\chi'',\chi')$ 与 $C(\chi',\chi'')$ 同时考虑. 实验结果如图 11～图 13 所示，图中的数据都是 50 次实验的平均值与均值 95%的置信区间.

表 3 MOMA 与 HEFT、PRO 算法优化效果比较

		HEFT			PRO		
		支配解个数	makespan 平均减小/%	energy 平均减少/%	支配解个数	makespan 平均减小/%	energy 平均减少/%
任务数量	32	17.6	10.06	23.20	12.4	10.18	14.50
	64	8.7	13.29	11.78	5.3	13.64	5.61
	128	7.7	7.67	10.39	5.6	6.77	5.73
	256	8.1	8.41	12.51	4.6	8.44	7.62
处理机个数	4	6.5	8.61	15.46	4.2	8.58	10.31
	8	7.7	7.67	10.39	4.8	6.51	7.14
	16	8.5	23.86	25.91	6.0	24.93	12.92
	32	9.4	17.31	23.98	7.2	18.43	11.47
并行因子	0.5	7.0	9.81	10.05	4.0	10.67	4.47
	1.0	7.7	7.67	10.39	6.0	8.91	4.95
	2.0	8.6	22.24	13.33	6.5	21.69	8.69
	5.0	2.2	0.96	1.50	1.8	0.83	0.71

图 11 是不同规模 DAG 任务图的调度模拟实验结果. 从图中可以发现 C(MOMA，MOEA) 的平均值一直大于 C(MOEA，MOMA) 的，且 C(MOMA，MOEA) 的置信区间都大于 C(MOEA，MOMA) 的. 这说明不同任务数目（对应问题的维度）下，MOMA 算法都产生了更优的非支配解，且具有更加稳定的性能. 对应不同的任务数目（32，64，128，256），MOEA 非支配解集中的解被 MOMA 的解支配的

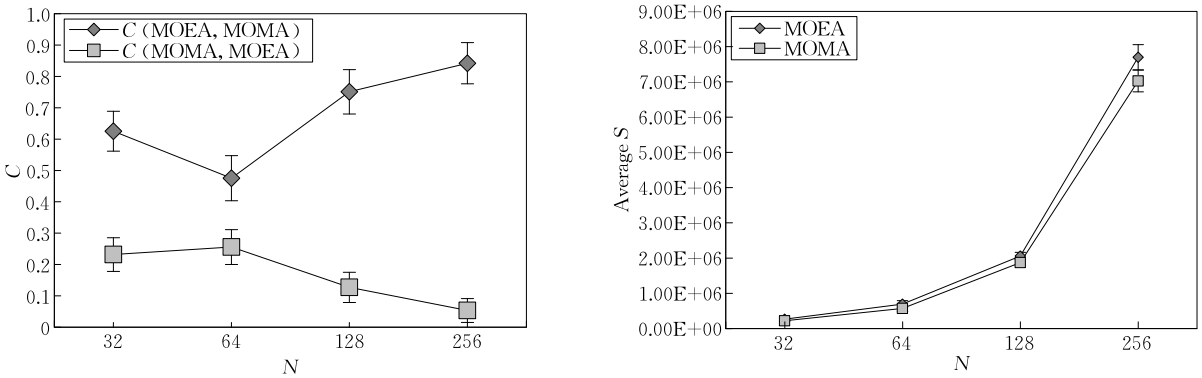


图 11　不同任务数目下的 C 指标和 S 指标

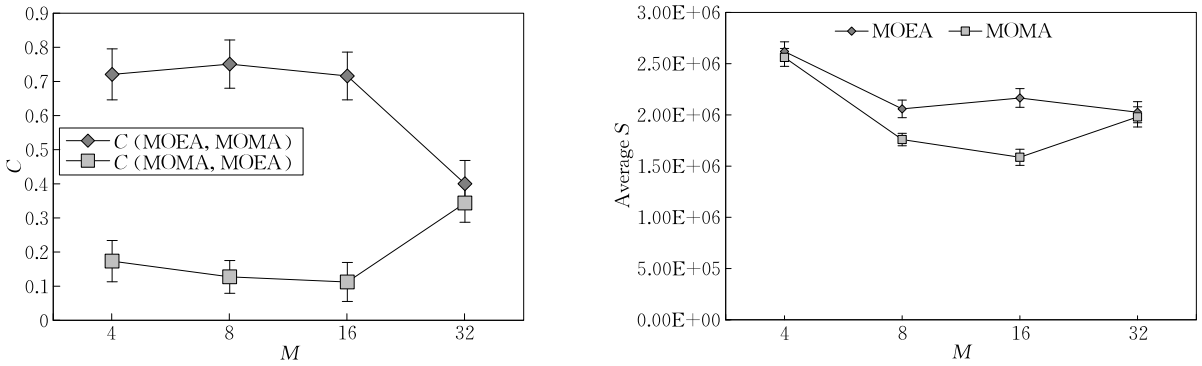


图 12　不同处理机数目下的 C 指标和 S 指标

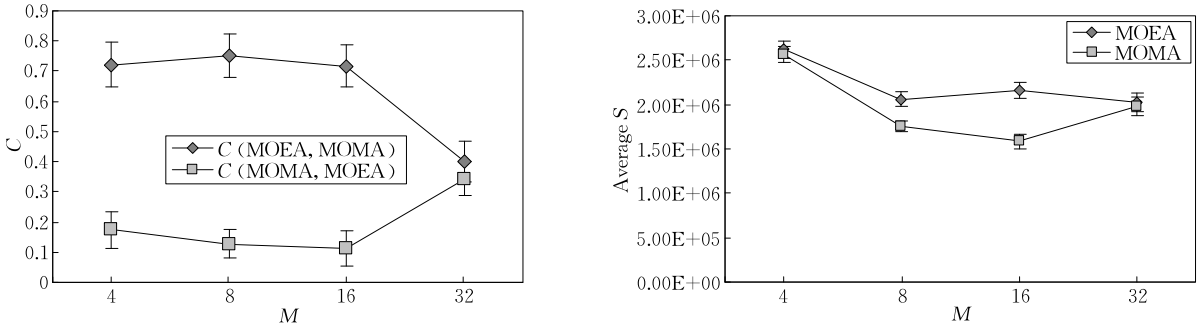


图 13　不同并行系数下的 C 指标和 S 指标

的，结合两者可知，不同问题维度下，MOMA 算法比 MOEA 算法都具有更好的收敛性能.

图 12 是在一定的任务数目下不同数量的处理机的调度模拟实验结果. 对应不同的处理机数目(4，8，16，32)，MOEA 非支配解集中的解被 MOMA 的解支配的平均比率分别为 72%，75%，71%，40%，MOMA 非支配解集中的解被 MOEA 的解支配的平均比率分别为 17%，12%，11%，34%，又 MOMA 算法的平均 S 指标都小于 MOEA 的，由此可知，不同处理机数目下，MOMA 算法相对 MOEA 算法也具有更好的收敛性能.

在不同的并行因子的 DAG 任务图调度模拟实验中，实验结果如图 13 所示，对应不同的 DAG 任务图并行因子(0.5，1，2，5)，MOEA 非支配解集中的解被 MOMA 的解支配的平均比率分别为 71%，75%，74%，74%，MOMA 非支配解集中的解被 MOEA 的解支配的平均比率分别为 11%，12%，10%，12%，且 MOMA 的平均 S 指标的都比 MOEA 的少. 可以发现，DAG 任务图不同的并行因子对 MOMA 算法的信能影响较少，MOMA 算法比 MOEA 算法具有更好的收敛性能.

算法的计算开销是算法评价的一个重要指标，我们采用 CEC2006 竞赛评价指标[①]进行算法计算

① Problem definitions and evaluation criteria for the CEC 2006 special session on constrained real-parameter optimization. http://www3.ntu.edu.sg/home/EPNSugan/

开销与计算复杂性的评价,具体定义如下:

$$T1=\left(\sum_{i=1}^{12}t1_i\right)\Big/12,$$

$$T2=\left(\sum_{i=1}^{12}t2_i\right)\Big/12 \tag{19}$$

其中 $t1_i$ 是问题 i 进行 10000 次评估的计算时间,本文主要考虑了 12 种不同场景参数设置下的 DAG 调度问题,$t2_i$ 是算法完成问题 i 10000 次迭代评估的完成时间. 当 $(T2-T1)/T1$ 越少时,说明算法的计算复杂度越低. 本文的实验结果如表 4 所示. 从表 4 我们发现,在相同的适应度评估次数下,MOMA 算法在计算复杂度方面略优于 MOEA 算法. 这主要是由于 MOMA 算法进行了局部寻优,减少了种群拥挤距离与支配等级排序的次数,又局部寻优的计算开销低于种群的拥挤距离与支配等级排序的,因此 MOMA 算法总的时间开销少于 MOEA 算法的开销.

表 4 MOMA 与 MOEA 算法计算复杂性比较

	$T1$(sec.)	$T2$(sec.)	$(T2-T1)/T1$
MOEA	20.51911	24.69846	0.1875
MOMA		24.31743	0.1692

从上述实验可以看出,MOMA 算法在不增加算法的计算开销的前提下较 MOEA 算法具有更好的收敛性能.

5 结束语

本文对异构云计算环境下能耗和性能优化管理的多目标优化问题进行定义,针对性的提出了包含 Memetic 局部优化策略的多目标优化方法,对当前调度的关键任务和非关键任务进行相对应的局部寻优,可以减少调度策略的评估计算,提高算法收敛速度,最终达到减少算法计算开销的作用. 一系列的模拟实验结果表明,MOMA 能够获得较好的优化效果,与经典的 MOEA 算法比较,具有更好的收敛性能与计算开销优势.

本文重点研究了面向复杂异构云计算系统的多目标调度优化算法,为构建灵活的优化调度器提出了可行的计算方法. 为使多目标优化调度更具实用性,针对具体的分布式计算系统动态特性与管理需求,采用合适的实时偏好决策同样非常重要,这也将是本方法应用研究的重要工作.

参 考 文 献

[1] Barroso L A, Hölzle U. The case for energy-proportional computing. IEEE Computer, 2007, 40(12): 33-37

[2] Raghavendra R, Ranganathan P, Talwar V, et al. No power struggles: Coordinated multi-level power management for the data center. ACM SIGARCH Computer Architecture News, 2008, 36(1): 48-59

[3] Hou E S H, Ansari N, Ren H. A genetic algorithm for multiprocessor scheduling. IEEE Transactions on Parallel and Distributed Systems, 1994, 5(2): 113-120

[4] Wu A S, Yu H, Jin S, et al. An incremental genetic algorithm approach to multiprocessor scheduling. IEEE Transactions on Parallel and Distributed Systems, 2004, 15(9): 824-834

[5] Swiecicka A, Seredynski F, Zomaya A Y. Multiprocessor scheduling and rescheduling with use of cellular automata and artificial immune system support. IEEE Transactions on Parallel and Distributed Systems, 2006, 17(3): 253-262

[6] Kashani M, Jahanshahi M. Using simulated annealing for task scheduling in distributed systems//Proceedings of the International Conference on Computational Intelligence, Modelling and Simulation. Brno, Czech Republic, 2009: 265-269

[7] Ferrandi F, Lanzi P, Pilato C, et al. Ant colony heuristic for mapping and scheduling tasks and communications on heterogeneous embedded systems. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2010, 29(6): 911-924

[8] Xu Y, Li K, He L, et al. A DAG scheduling scheme on heterogeneous computing systems using double molecular structure-based chemical reaction optimization. Journal of Parallel and Distributed Computing, 2013, 73(9): 1306-1322

[9] Elnozahy E N M, Kistler M, Rajamony R. Energy-Efficient Server Clusters. Berlin: Springer, 2003

[10] Bahsoon R. A framework for dynamic self-optimization of power and dependability requirements in Green Cloud architecture//Babar M A, Gorton I eds. Software Architecture. Berlin: Springer, 2010: 510-514

[11] Lin Chuang, Tian Yuan, Yao Min. Green network and green evaluation: Mechanism, modeling and evaluation. Chinese Journal of Computers, 2011, 34(4): 593-612(in Chinese)
(林闯, 田源, 姚敏. 绿色网络和绿色评价: 节能机制, 模型和评价. 计算机学报, 2011, 34(4): 593-612)

[12] Li K. Performance analysis of power-aware task scheduling algorithms on multiprocessor computers with dynamic voltage and speed. IEEE Transactions on Parallel and Distributed Systems, 2008, 19(11): 1484-1497

[13] Li K. Scheduling precedence constrained tasks with reduced processor energy on multiprocessor computers. IEEE Transactions on Computers, 2012, 61(12): 1668-1681

[14] Li K. Optimal power allocation among multiple heterogeneous servers in a data center. Sustainable Computing: Informatics and Systems, 2012, 2(1): 13-22

[15] Song Jie, Li Tian-Tian, Yan Zhen-Xing, et al. Energy-efficiency model and measuring approach for cloud computing. Journal of Software, 2012, 23(2): 200-214(in Chinese)
(宋杰，李甜甜，闫振兴等. 一种云计算环境下的能效模型和度量方法. 软件学报，2012，23(2)：200-214)

[16] Cao J, Li K, Stojmenovic I. Optimal power allocation and load distribution for multiple heterogeneous multicore server processors across clouds and data centers. IEEE Transactions on Computers, 2014, 63(1): 45-58

[17] Zong Z, Manzanares A, Ruan X, et al. EAD and PEBD: Two energy-aware duplication scheduling algorithms for parallel tasks on homogeneous clusters. IEEE Transactions on Computers, 2011, 60(3): 360-374

[18] Li Xin, Jia Zhi-Ping, Ju Lei, et al. Energy efficient scheduling and optimization for parallel tasks on homogeneous clusters. Chinese Journal of Computers, 2012, 35(3): 591-602 (in Chinese)
(李新，贾智平，鞠雷等. 一种面向同构集群系统的并行任务节能调度优化方法. 计算机学报，2012，35(3)：591-602)

[19] Ishibuchi H, Yoshida T, Murata T. Balance between genetic search and local search in Memetic algorithms for multiobjective permutation flowshop scheduling. IEEE Transactions on Evolutionary Computation, 2003, 7(2): 204-223

[20] Tang K, Mei Y, Yao X. Memetic algorithm with extended neighborhood search for capacitated arc routing problems. IEEE Transactions on Evolutionary Computation, 2009, 13(5): 1151-1166

[21] Neri F, Cotta C. Memetic algorithms and memetic computing optimization: A literature review. Swarm and Evolutionary Computation, 2012, 2: 1-14

[22] Deb K, Pratap A, Agarwal S, et al. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation, 2002, 6(2): 182-197

[23] Topcuoglu H, Hariri S, Wu M. Performance-effective and low-complexity task scheduling for heterogeneous computing. IEEE Transactions on Parallel and Distributed Systems, 2002, 13(3): 260-274

[24] Bäck T. Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms. Britain, Oxford: Oxford University Press, 1996

[25] Deb K. Multi-Objective Optimization Using Evolutionary Algorithms. Britain, Chichester: John Wiley & Sons, 2001

[26] Zitzler E, Thiele L. Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach. IEEE Transactions on Evolutionary Computation, 1999, 3(4): 257-271

LI Zhi-Yong, born in 1971, Ph. D., professor. His research interests include embedded computing system, dynamic multi-objective evolutionary algorithm and cloud computing.

CHEN Shao-Miao, born in 1987, Ph. D. candidate. His research interests include evolutionary computation, cloud computing.

YANG Bo, born in 1974, Ph. D. candidate. His research interests include resource management in cloud computing system and game theory.

LI Ren-Fa, born in 1957, professor, Ph. D. supervisor. His research focuses on network computing, embedded computing.

Background

Our research works on the multi-objective optimal tasks scheduling and resource management on cloud computing system. For the past several years, most of heuristic scheduling optimization algorithms focused on the single-objective optimization on performance or energy efficiency, and less consider multiple optimization features of complex cloud computing system simultaneously. In this paper, the optimal management of energy consumption and computation performance on heterogeneous cloud is considered as a multi-objective optimization problem. In order to solve this problem, a novel multi-objective memetic algorithm (MOMA) is proposed, which has less computational load and better convergence capability than those compared multi-objective optimization methods.

This issue is a part of the project "New Generation High Performance Computer System 'Tianhe'" (National High Technology Research and Development Program (863 Program) of China, No. 2012AA01A301-01), and our method is based on project "Auantum-Inspired Memetic Computation Strategies and Algorithms for Dynamic Multi-Objective Optimization" (National Natural Science Foundation of China, No. 61173107). This research will improve the resource management and tasks scheduling capability of cloud computing system on 'Tianhe', meanwhile this optimization problem accelerate us designing novel multi-objective optimization algorithms.

In recent years, our team mainly works on high performance computing system (include the performance, energy efficiency and security on distributed computing system) and intelligent optimization methods (include scheduling and management on distributed computing system, multi-objective optimization algorithms). We have published research papers in international journals and conferences. Some of them are listed as follows:

[1] Nguyen T T, Li Z, Zhang S, et al. A hybrid algorithm based on particle swarm and chemical reaction optimization. Expert Systems with Applications, 2014, 41(5): 2134-2143

[2] Li Z, Chen H, Xie Z, et al. Dynamic multiobjective optimization algorithm based on average distance linear prediction model. The Scientific World Journal, 2014, 2014

[3] Tang Xiaoyong, Li K, Zeng Z, et al. A novel security-driven scheduling algorithm for precedence-constrained tasks in heterogeneous distributed systems. IEEE Transactions on Computers, 2011, 60(7): 1017-1029

[4] Li K. Scheduling precedence constrained tasks with reduced processor energy on multiprocessor computers. IEEE Transactions on Computers, 2012, 61(12): 1668-1681

[5] Tang X, Li K, Liao G, Li R. List scheduling with duplication for heterogeneous computing systems. Journal of Parallel and Distributed Computing, 2010, 70(4): 323-329

[6] Xu Y, Li K, He L, et al. A DAG scheduling scheme on heterogeneous computing systems using double molecular structure-based chemical reaction optimization. Journal of Parallel and Distributed Computing, 2013, 73(9): 1306-1322