DOI:10.13718/j.cnki.xsxb.2020.05.023

基于重复异构最早完成时间的云计算任务调度算法①

蔡昌许

曲靖师范学院 信息工程学院,云南 曲靖 655011

摘要:针对云计算任务调度算法时间成本消耗大的问题,提出了一种新的算法,即重复异构最早完成时间(Duplication based Heterogeneous Earliest Finish Time,DHEFT)任务调度方法.该方法结合异构最早完成时间算法和任务重复算法,可以大大减少任务最早开始时间和最早完成时间.由于任务优先级对于相关任务调度算法的重要性,该算法中提出了乐观成本表的方法来计算任务优先级,根据优先级调度任务,并复制父任务以降低通信成本并获得最佳调度解决方案.实验结果表明,该文提出的 DHEFT 在调度长度比和完成时间性能方面优于其他算法,说明该文方法的可行性与有效性.

关 键 词:云计算;任务调度;重复异构最早完成时间;任务优先级

中图分类号:TP391 **文献标志码**:A **文章编号**:1000-5471(2020)05-0141-07

云计算是一种计算模型,可以访问共享资源,包括计算机设备,应用程序或存储资源[1-2].云计算是一种分布式计算,能够提供安全快速、便捷的网络计算服务[3-4].在云计算中,执行并行应用程序的效率关键取决于用于调度并行应用程序任务的算法,此类应用程序始终由具有优先约束的大量任务组成,并由有向无环图(Directed Acyclic Graph,DAG)表示,其中节点表示应用程序任务,有向边表示依赖性.DAG 调度问题已经表明是完全非确定性多项式(Nondeterministic Polynomially,NP)问题[5-7],因此研究重点是更好地调度求解.

DAG 调度算法大多基于启发式算法,目前的研究提出了多种启发式算法,包括聚类算法、遗传算法、粒子群优化算法、随机调度算法和列表调度算法等.聚类算法[8]必须有从生成的簇到有界数处理器的映射过程,假设处理器的数目过大,则算法此时不适用.文献[9]提出了一种功能强大且改进的遗传算法.该算法利用进化遗传算法和启发式方法的优点采用基于模型检验技术的行为建模方法,该方法能够提供良好的时间表,但执行时间明显高于其他方法.粒子群优化算法也被用到云计算任务调度中[10-11],文献[12]提出一种基于多目标粒子群算法和遗传算法的超启发式资源调度算法作为混合算法,该混合调度算法与现有的基于启发式的调度算法进行比较,在降低成本和提高完工时间方面都有所提高,但该算法的缺点是执行时间仍然较高.

列表调度的基本思想是先计算任务优先级,然后按优先级递减顺序将任务放在列表中,然后根据资源的优先级映射任务.很明显,优先级越高的任务越早执行.常见的列表调度算法之一是异构最早完成时间(Heterogeneous Earliest Finish Time,HEFT)算法,具有复杂度低、性能良好的优点.文献[13]提出一种

① 收稿日期:2018-07-30
基金项目:云南省科技厅高校联合面上项目(2017FH001-060).
作者简介:蔡昌许(1976-),男,硕士,讲师,主要从事信息管理与信息系统研究.

节能工作流任务调度算法，该方法通过回收松弛时间来合并相对低效的处理器，首先计算所有任务的初始调度顺序，并根据HEFT算法获得整个完工时间和期限. 通过使处理器具有运行任务编号和能量利用率，可以关闭最后一个节点并在其上重新分配任务来合并未充分利用的处理器. 该算法的任务调度性能还可以提高.

针对现有云计算调度算法存在的问题，提出了一种重复异构最早完成时间的云计算任务调度算法，该算法基于HEFT和任务复制算法，采用乐观成本表的算法来计算优先级，能够降低通信成本并获得最佳调度解决方案.

1 DAG调度模型

应用程序的相关任务调度问题可以由有向无环图表示(图1)，$G=(V, E)$，其中V是节点集合，每个节点$v_i \in V$表示任务，E是通信边缘集合，表示任务之间的依赖关系，每个边缘$e(i, j) \in E$表示任务依赖约束使得任务v_i的执行应该在执行任务v_j之前完成，任务v_i是任务v_j的前驱任务，任务v_j是任务v_i的后继任务.

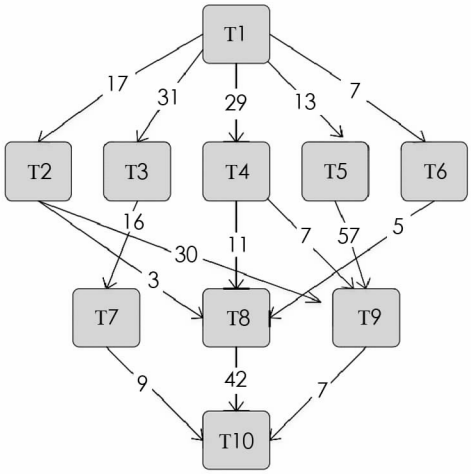

图1 DAG应用过程

没有前驱节点的节点称为入口节点，没有后继节点的节点称为退出节点，如果有多个入口节点或退出节点，则设置一对伪入口节点和退出节点. 此外，伪入口节点和退出节点的计算成本和通信成本也将设置为0. 当所有前驱任务完成并将前体任务的输出传递到任务执行处理器时，可以执行任务. DAG由一个矩阵$\boldsymbol{W}$来补充，它是一个$v \times p$的计算成本矩阵，其中v代表任务的数量，p是系统中处理器的数量. $w_{i,j}$表示在处理器p_j上执行任务v_i的估计时间. 表1给出了图1中每个处理器中任务的不同计算时间矩阵.

表1 每个处理器的任务计算时间

v_i	p_1	p_2	p_3
v_1	22	21	36
v_2	22	18	18
v_3	32	27	43
v_4	7	10	4
v_5	29	27	35
v_6	26	17	24
v_7	14	25	30
v_8	29	23	36
v_9	15	21	8
v_{10}	13	16	33

任务的平均执行时间为

$$\overline{w}_i = \sum_{j \in p} w_{i,j} / p \tag{1}$$

每个边缘$e(i, j) \in E$与调度前的平均通信成本相关，平均通信成本通常为

$$\bar{c}_{i,j} = \overline{L} + \frac{data_{i,j}}{\overline{B}} \tag{2}$$

其中，$\overline{L}$表示处理器的平均延迟时间，$\overline{B}$表示连接p处理器集合的所有链路的平均带宽，$data_{i,j}$表示任务v_i应该传输到任务v_j的通信量. 如果任务v_i和任务v_j被安排在同一处理器上，那么通信成本将被设置为零，因为在本文中该通信成本与处理器间通信成本相比可以忽略不计. 在本文中，假设处理器连接在完全的连接拓扑，且当处理器正在执行任务时不是抢占式的，处理器可以同时与其他人通信.

在介绍目标函数之前，首先描述与以下定义相关联的 $pred(v_i)$ 和 $succ(v_i)$，$pred(v_i)$ 表示 DAG 中任务 v_i 的直接前驱任务集，如果给定多个入口节点，则 DAG 设置具有零计算成本和零通信成本的伪入口节点. 而 $succ(v_i)$ 表示一组任务 v_i 的后继任务，如果 DAG 中存在多个出口节点，则将在 DAG 中建立具有零计算成本和零通信成本的伪出口节点，其前驱任务是当前多个出口节点.

EST 和 EFT 两个重要的定义，$EST(v_i, p_j)$ 表示处理器 p_j 上任务 v_i 的最早开始时间，入口节点的 EST 表示为 $EST(v_{entry}, p_j)=0$，$EFT(v_i, p_j)$ 表示处理器 p_j 上任务 v_i 的最早执行完成时间. 节点的 EST 和 EFT 计算为

$$EST(v_i, p_j)=\max\left\{\begin{matrix} r(v_i, p_j), \\ \max\{EFT(v_i, p_j)+c_{j,i}\}\end{matrix}\right\} \tag{3}$$

$$EFT(v_i, p_j)=EST(v_i, p_j)+w(v_i)/w(p_j) \tag{4}$$

$$c_{j,i}=w(e_{j,i})/w(l_{n,m}) \tag{5}$$

其中，$r(v_i, p_j)$ 是处理器 p_j 最早的就绪时间，$w(\cdot)$ 表示在处理器上执行任务的估计时间. 式(3)中的内部最大块是任务 v_i 所有前驱任务已经执行并且所需的所有数据已经被传送到处理器 p_j 的时间，并且当任务 v_i 和任务 v_j 都安排在同一处理器 p_j 上时，$c_{j,i}$ 为零. 调度长度定义为 $makespan=EFT(v_{exist})$，表示退出节点的 EFT，如果有多个退出节点，调度长度是伪出口节点的 EFT.

任务调度问题的目标函数是确定给定应用程序的任务分配给处理器，使得其调度长度最小化.

2　基于 DHEFT 任务调度方法

本文 DHEFT 算法在云计算中被称为完全连接的异构处理器算法，有 3 个主要阶段：①任务优先化阶段，用于计算任务优先级；②处理器选择阶段，用于选择最合适的处理器来执行当前任务；③复制阶段，用于更早地执行当前任务的执行起始时间.

2.1　优先级定义

给定 DAG，可以根据任务优先级按降序构造任务列表. 与 HEFT 算法相比，有更好的方法来计算任务优先级：Lookahead 算法和有限数量异构处理器算法(PEFT 算法)，两个算法的主要特点是处理器选择策略. Lookahead 算法和 PEFT 算法最强大的特点是，都有能力预测当前任务和所有子任务分配的影响. 虽然 Lookahead 可以在选择处理器方面做出更好的决策，但是计算优先级的复杂性显著增加. 因此，在本文中采用 PEFT 方法，引入一种新的基于列表的调度算法，即乐观成本表(optimistic cost table，OCT) 方法. OCT 是一个矩阵，其中行的数目是任务数目，列表示处理器数目. $OCT(v_i, p_k)$ 表示处理器 p_j 上任务 v_i 的 OCT 值，通过将 DAG 从出口节点向上遍历到入口节点，通过等式(6) 递归地计算.

$$OCT(v_i, p_k)=\max_{v_j\in succ(v_i)}\left\{\min_{p_w\in P}\left\{\begin{matrix}OCT(v_j, p_w)+ \\ w(v_j, p_w)+c_{j,i}\end{matrix}\right\}\right\} \tag{6}$$

其中，$c_{j,i}$ 表示通信成本，如果在处理器 p_k 上安排任务 v_j，则 $c_{j,i}$ 为零. $w(v_j, p_w)$ 表示任务 v_j 在处理器 p_w 上的执行时间. $OCT(v_i, p_k)$ 表示最大乐观处理时间任务 v_i 的后继节点，因为后继任务是在处理器中执行，与处理器可用性无关的最小化处理时间(执行和通信)，在调度开始之前计算 OCT. 因为 OCT 是递归定义的，并且子节点已经具有退出节点的乐观成本，所以仅考虑第一级子节点. 对于在任何处理器上调度的出口节点，OCT 值为零.

通过平均每项任务的 OCT，可以计算本文中的任务优先级.

$$\mathrm{rank}_{\mathrm{oct}}(v_i)=\frac{\sum_{k=1}^{P}OCT(v_i, p_k)}{P} \tag{7}$$

设 HEFT 算法计算的优先级为 $\mathrm{rank_h}$，表 2 显示了通过 OCT 和 HEFT 两种方式计算的图 1 中 DAG 样本的优先级.

与HEFT优先级相比，OCT优先级在任务的顺序中是不同的. 例如，T5的优先级低于T4，因此首先选择T4进行调度. 相反，基于$rank_h$首先选择T5进行调度. 本优先级排序算法的主要特征是OCT，反映每个任务和处理器执行后继任务直到退出节点的成本.

表2 不同算法的优先级

v_i	p_1	p_2	p_3	$rank_{oct}$	$rank_h$
v_1	22	21	36	72.7	169
v_2	22	18	18	41	114.3
v_3	32	27	43	37	102.7
v_4	7	10	4	43.7	110
v_5	29	27	35	31	129.7
v_6	26	17	24	41.7	119.3
v_7	14	25	30	17	52.7
v_8	29	23	36	20.7	92
v_9	15	21	8	16.3	42.3
v_{10}	13	16	33	0	20.7

2.2 处理器选择和复制阶段

在按优先级构造任务序列后，将任务分配给处理器，并应用复制来减少完成时间. 任务序列中的第一个未调度任务v_i被选择并在处理器p_j上调度. $r(v_i, p_j)$和$EST(v_i, p_j)$之间可能有空闲时间，其定义为

$$slot(v_i, p_j) = EST(v_i, p_j) - r(v_i, p_j) \tag{8}$$

当选择任务序列中的任务v_i在处理器p_j上调度时，如果任务$v_j \in pred(v_i)$满足以下条件：任务v_j在处理器p_j上执行，$slot(v_j, p_j) \geqslant w(v_j, p_j)$，同时任务$v_j$从未在处理器$p_j$上被调度，则可以在处理器$p_j$上复制任务$v_j$，以使任务$v_i$更早地执行，因此任务$v_i$可以最早使用前驱任务复制完成其执行. 本文算法只是在前两个或3个级别上进行复制，因为如果在所有节点上执行操作，复杂性将显着增加，并且前两个或3个级别起到决定性作用，以使总体完工时间减少.

3 实验结果与分析

为了验证本文DHEFT任务调度算法的可行性，进行了与其他算法相比较的仿真实验. 为了测试调度算法的性能，在墨尔本大学开发的CloudSim-3.0.3平台上进行模拟实验.

本文使用任务完成时间和调度长度比(scheduling length ratio，*SLR*)来对本文算法进行验证，*SLR*是将DAG与不同的拓扑结构进行比较，表示标准化下限的完工时间，*SLR*定义为

$$SLR = \frac{makespan}{\sum_{v_i \in CP_{MIN}} \min_{p_j \in P}(w_{i,j})} \tag{9}$$

*SLR*方程中的分母是关键路径CP_{MIN}上任务计算成本的最小和. 给定DAG的完工时间大于等式的分母，因此较低的*SLR*表示算法更好. CP_{MIN}表示当任务节点权重被评估为所有有能力的处理器中最小计算成本时的DAG关键路径，分母代表调度长度的下限.

为了评估启发式的相对性能，使用随机生成的应用图，采用DAG Generation Program合成DAG生成程序，主要参数决定DAG形状，具体参数为

n：DAG中的计算节点数(即应用任务).

fat：确定DAG的宽度，DAG的宽度是可以同时执行的最大任务数. 较小的值将导致具有低任务并行性的窄DAG，而较大的值导致具有高度并行性的宽DAG.

密度：该参数确定DAG两个级别任务之间的依赖性数量，低值表示少量边缘，大值表示许多边缘.

跳转：任务间通信跨越的最大级别数，允许生成具有不同长度执行路径的DAGs.

本文中，在合成DAG生成器中使用不同的参数值来创建DAG，其中包括特定数量的节点及其依赖

性. 有两个重要的定义是通信计算比(Communication-to-computation ratio，*CCR*)和处理器上计算成本的范围百分比 β 解释如下：

CCR 表示边缘权重之和与 DAG 中节点权重之和的比率. β 表示处理器上计算成本的范围百分比，处理器速度的异质性因素. β 值高意味着更高的异质性和不同处理器之间的计算成本，低 β 值表示给定任务的计算成本在处理器之间几乎相等. 图 2 显示了本文算法 DHEFT 和 HEFT 样本 DAG 的调度结果.

通过比较 HEFTD 和 HEFT 的时间表，可以看到 $T1$ 被分配给 $P1$，虽然不能保证 $T1$ 的最早完成时间，但是 $P1$ 最小化了预期的 *EFT*，说明本文方法平均比现有 HEFT 算法产生更好的调度.

在本文中 DHEFT 与 HEFT 相比具有不同 *CCR* 值的性能，设置任务数量的范围为[50，100，200，300，400，500]，根据实验结果得到计算能力异质性因子值 β 取值为 0.2 时性能最佳，此时选择 β 为 0.2. 利用不同的 *CCR* 值，可以从这些数据中了解到 DHEFT 算法的平均 *SLR* 低于 HEFT 算法. 图 3－图 6 给出了不同 *CCR* 下的 *SLR* 性能.

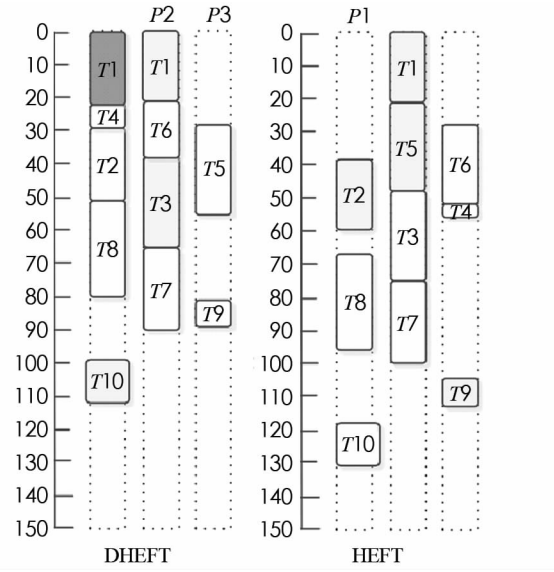

图 2　不同算法在
图 1 中样本任务图的时间表

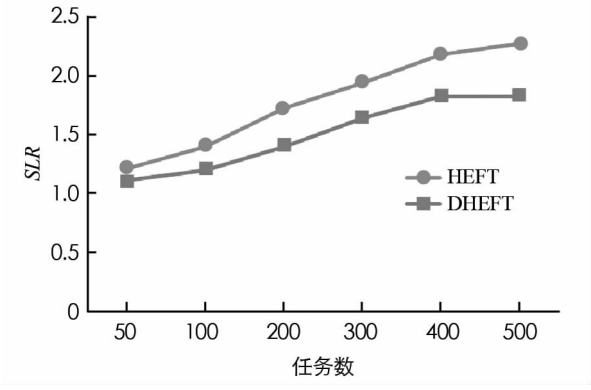

图 3　*CCR*＝0.1 时 *SLR* 性能对比

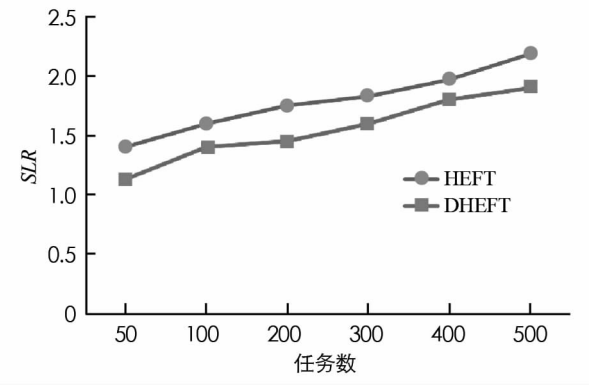

图 4　*CCR*＝0.5 时 *SLR* 性能对比

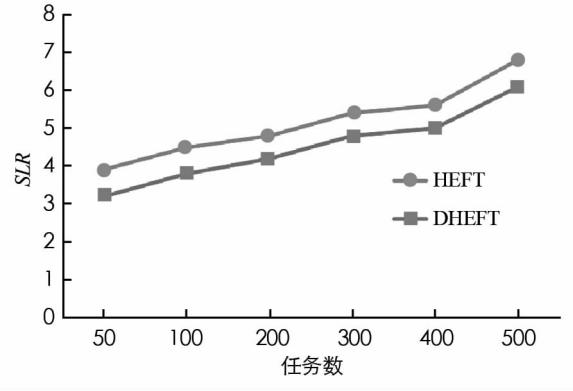

图 5　*CCR*＝1 时 *SLR* 性能对比

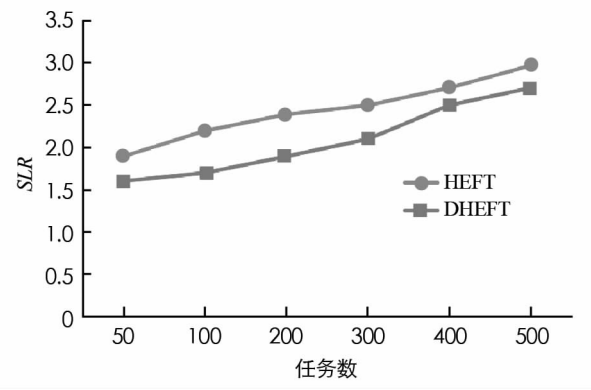

图 6　*CCR*＝5 时 *SLR* 性能对比

从图 3－图 6 中不同 *CCR* 下的 *SLR* 值可以观察到，当 *CCR*＝0.1 时 *SLR* 最小，*CCR*＝1 时 *SLR* 最大，对于所有情况，平均 *SLR* 分别随着任务数量的增加而呈现增加趋势，这是因为除了关键路径上的任务

节点之外，任务节点的比例随着任务图增大而增加，使得实现下限更加困难. 在对时间的性能比较实验中，选择 CCR 等于 0.1，β 取值 0.2 进行实验.

图 7 给出本文算法与其他不同算法任务完成时间的比较. 其他算法有处理器关键路径算法(Critical Path ON a Processor，CPOP)，异构最早完成时间(Heterogeneous Earliest Finish Time，HEFT)算法，文献[14]中任务分层和时间约束的关联任务调度算法(related Task Scheduling Algorithm Based On Task Hierarchy And Time Constraint，RTS-TH-TC)，文献[15]中任务复制的 QoS 调度算法(task replication of QoS，TR-QoS). 图 7 中曲线自上而下分别为 CPOP，HEFT，[15]，[14]，DHEFT，从图 7 中可以看出，本文算法在任务完成时间方面的性能要优于其他几种任务调度方法，说明本文方法的可行性与有效性.

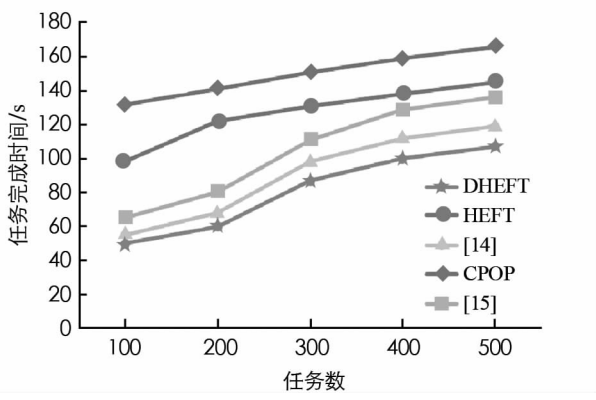

图 7 不同算法的任务完成时间

4 结 语

本文提出了一种新的任务调度算法，称为基于复制的异构最早完成时间 DHEFT 任务调度方法. 该算法结合异构最早完成时间算法和任务复制算法，可以减少任务的最早开始时间和最早完成时间，从而缩短任务完成时间. 采用乐观成本表方法来计算任务优先级，该表考虑了后继任务的总体成本以获得更好的任务序列. 实验结果表明，本文 DHEFT 调度算法比 HEFT 算法具有更低的 SLR，且在任务完成时间性能方面，本文算法优于其他算法.

参考文献：

[1] WANG B，LI J，WANG C. Cost-Effective Scheduling Precedence Constrained Tasks in Cloud Computing [C]//2017 IEEE 2nd International Conference on Cloud Computing and Big Data Analysis (ICCCBDA). Chengdu：IEEE，2017.

[2] XIA Z H，WANG X H，ZHANG L G，et al. A Privacy-Preserving and Copy-Deterrence Content-Based Image Retrieval Scheme in Cloud Computing [J]. IEEE Transactions on Information Forensics and Security，2016，11(11)：2594-2608.

[3] ALI M，KHAN S U，VASILAKOS A V. Security in Cloud Computing：Opportunities and Challenges [J]. Information Sciences，2015，305：357-383.

[4] 张少辉，崔仲远，韩秋英. 云计算环境下基于非均匀窗口蚁群行为的负载平衡算法 [J]. 重庆邮电大学学报(自然科学版)，2016，28(4)：567-574.

[5] AIZAD S，ANJUM A，SAKELLARIOU R. Representing Variant Calling Format as Directed Acyclic Graphs to Enable the Use of Cloud Computing for Efficient and Cost Effective Genome Analysis [C]//2017 17th IEEE/ACM International Symposium on Cluster，Cloud and Grid Computing (CCGRID). Madrid：IEEE，2017.

[6] Musial J，Guzek M，Bouvry P，et al. A note on the complexity of scheduling of communication-aware directed acyclic graph [J]. Bulletin of the Polish Academy of Sciences，Technical Sciences，2018，66(2)：187-191.

[7] MEIJER R J，GOEMAN J J. A Multiple Testing Method for Hypotheses Structured in a Directed Acyclic Graph [J]. Biometrical Journal，2015，57(1)：123-143.

[8] 王李彧，孙 斌，秦 童. 改进的 DBSCAN 聚类算法在云任务调度中的应用 [J]. 北京邮电大学学报，2017，40(S1)：68-71.

[9] KESHANCHI B，SOURI A，NAVIMIPOUR N J. An Improved Genetic Algorithm for Task Scheduling in the Cloud Environments Using the Priority Queues：Formal Verification，Simulation，and Statistical Testing [J]. Journal of Systems and Software，2017，124：1-21.

[10] ZUO X Q, ZHANG G X, TAN W. Self-Adaptive Learning PSO-Based Deadline Constrained Task Scheduling for Hybrid IaaS Cloud [J]. IEEE Transactions on Automation Science and Engineering, 2014, 11(2): 564-573.

[11] AL-MAAMARI A, OMARA F A. Task Scheduling Using PSO Algorithm in Cloud Computing Environments [J]. International Journal of Grid and Distributed Computing, 2015, 8(5): 245-256.

[12] KUMARI K R, SENGOTTUVELAN P, SHANTHINI J. A Hybrid Approach of Genetic Algorithm and Multi Objective PSO Task Scheduling in Cloud Computing [J]. Asian Journal of Research in Social Sciences and Humanities, 2017, 7(3): 1260-1271.

[13] TANG Z, QI L, CHENG Z Z, et al. An Energy-Efficient Task Scheduling Algorithm in DVFS-enabled Cloud Environment [J]. Journal of Grid Computing, 2016, 14(1): 55-74.

[14] 陈　曦，毛莺池，接　青，等. 云计算中基于任务分层和时间约束的关联任务调度算法 [J]. 计算机应用，2014，34(11)：3069-3072.

[15] 张巧龙，张桂珠，吴德龙. 基于任务复制的多维 QoS 云计算任务调度 [J]. 计算机应用，2014，34(9)：2527-2531.

Cloud Computing Task Scheduling Algorithm of Duplication Based Heterogeneous Earliest Finish Time

CAI Chang-xu

College of Engineering, Qujing Normal University, Qujing Yunnan 655011, China

Abstract: Aiming at the problem of large time cost of cloud computing task scheduling algorithm, a new algorithm, Duplication based Heterogeneous Earliest Finish Time task scheduling method has been proposed. In the method the heterogeneous earliest finish time algorithm and the task repetition algorithm have been combined, which can greatly reduce the earliest start time and the earliest completion time of the task. Due to the importance of the task priority to the related task scheduling algorithm, the method of calculating the optimistic cost table has been proposed in the algorithm, and task scheduling been completed based on the priority calculated in this method. Then replicating the parent task to reduce communication cost and obtain the best scheduling solution. The experimental results show that the DHEFT proposed in this paper is superior to other algorithms in terms of scheduling length ratio and completion time performance, indicating the feasibility and effectiveness of the proposed method.

Key words: cloud computing; task scheduling; Duplication based Heterogeneous Earliest Completion Time; task priority

责任编辑　夏　娟