

引用格式:张小庆,安春玲,胡亚捷. 移动云计算多目标任务调度进化算法[J]. 微电子学与计算机, 2020, 37(10): 79-86.
[ZHANG X Q, AN C L, HU Y J. Multi-objective task scheduling evolutionary algorithm in mobile cloud computing[J]. Micro-electronics & Computer, 2020, 37(10): 79-86.]

移动云计算多目标任务调度进化算法

张小庆¹, 安春玲², 胡亚捷¹

(1 武汉轻工大学 数计学院, 湖北 武汉 430023;

2 武汉轻工大学 图书馆, 湖北 武汉 430023)

摘 要: 研究了移动云环境中任务调度的多目标优化问题, 提出一种多目标任务调度进化算法 MTSEA. 建立了截止时间、预算及能量约束下的任务调度多目标优化模型, 模型引入执行跨度、执行代价及执行能耗三目标最优化; 设计了一种进化算法对冲突三目标最优化进行求解, 算法重点在种群初始化操作中引入了效率最高、代价最小以及能效最高的三个种群个体, 以此代替随机个体生成; 并利用交叉和变异操作对个体进化迭代, 最终通过非占优排序形式得到满足帕累托最优的调度解集. 通过仿真实验与两种多目标调度算法进行了性能对比. 结果表明, MTSEA 算法调度解的收敛性及解空间距离和分布上是更优的.

关键词: 移动云计算; 任务调度; 进化算法; 多目标优化

中图分类号: TP393

文献标识码: A

文章编号: 1000—7180(2020)10—0079—08

Multi-objective task scheduling evolutionary algorithm in mobile cloud computing

ZHANG Xiao-qing¹, AN Chun-lin², HU Ya-jie¹

(1. School of Mathematics and Computer Science, Wuhan Polytechnic University, Wuhan 430023, China;

2. Library, Wuhan Polytechnic University, Wuhan 430023, China)

Abstract: The multi-objective optimization problem of tasks scheduling in mobile cloud is studied, a multi-objective scheduling evolutionary algorithm MTSEA is proposed. The multi-objective optimization model of tasks scheduling with the constraints of deadline, budget and energy, which introduces the execution makespan, execution cost and execution energy consumption as three-objective optimization. An evolutionary algorithm is presented to solve this three-objective optimization problem with conflict. Our algorithm focuses to introduce three population individual with highest efficiency, minimal cost and highest energy-efficiency in the population initialization instead of randomly generated particles. And, our algorithm uses the evolutionary iteration of the crossover and mutation operation among population individuals. Finally the algorithm can generate the scheduling solution set meeting Pareto optimal by the form of the not-dominant order. Some simulation experiments are constructed and the performance comparison are performed with other two kinds of multi-objective scheduling algorithms. The results show that MTSEA performs better on the convergence of scheduling solutions and the solution space distance and its distribution.

Key words: mobile cloud computing; tasks scheduling; evolutionary algorithm; multi-objective optimization

1 引言

移动云环境中的 DAG 任务(workflow 形式的非独立任务集)调度问题不同于传统的任务调度问题,需要考虑的因素更多更复杂,包括:1)用户 QoS 需求的多样性,如:服务响应时间、服务代价等;2)云服务具有异质性、动态性和弹性等特征;3)联合云服务执行 workflow 任务的方式多样;4)DAG 任务调度时任务间的数据传输更为普遍. 此时的任务调度问题本质上是联合优化问题.

已有的研究内容多数工作集中于优化单一的或两个目标,且没有考虑不同目标间的相互影响,即执行时间与执行代价、或执行时间与执行能耗间的影响^[1-9]. 针对这一问题,本文将设计一种多目标(三个目标)任务调度方法,并在任务的完成时间、任务执行的费用预算以及移动云资源的总能量均有约束的条件下,在任务执行时间、执行代价和执行能耗三个目标上实现同步均衡优化,形成多目标优化调度解集合.

2 系统模型

将一个云应用表示为 workflow 任务模型,以有向无循环图 DAG 表示,图中的每个节点代表一个任务. 将 workflow 任务表示为 DAG $G = (T, E)$, T 为包含 n 个任务的集合, $T = \{T_1, T_2, \dots, T_n\}$, E 为任务间的有向边集合, 每条边 $(T_i, T_j) \in E$ 代表任务间的执行顺序约束, 表示任务 T_i 必须在任务 T_j 开始前执行完成, T_i 为 T_j 的父任务, T_j 为 T_i 的子任务, 且假设仅在某一任务的所有父任务完成并得到所有数据传输后才可以开始执行. 每条边上的权值代表两个任务间的数据传输时间. 若 DAG 模型中的任务无任一父任务, 则称该任务为入口任务 T_{entry} , 若无任一子任务, 则称该任务为出口任务 T_{exit} . 如图 1 为包括 8 个任务的 DAG, 其中, T_1 即为 DAG 的入口任务 T_{entry} , T_8 即为 DAG 的出口任务 T_{exit} , 有向边上的数值表示任务间的数据传输时间.

系统提供的移动云资源表示为集合 R , 资源总量为 m , 表示为 $R = \{R_1, R_2, \dots, R_m\}$. 所有资源间为完全连通拓扑结构, 每种资源的处理能力与执行代价各不相同, 且均有能力执行任务集中的每个任务. 假设任务在移动云资源上执行过程中不能被其他任务抢占, 直到任务完成为止. 如图 2 为包括 5 个资源的结构图.

令 $\text{EST}(T_i, R_j)$ 表示任务 T_i 在资源 R_j 上的最

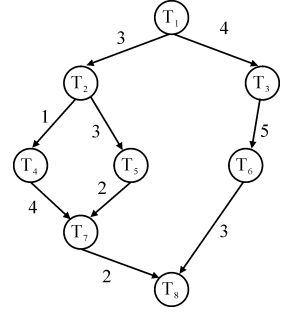


图 1 8 个任务的 task DAG

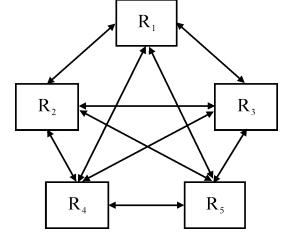


图 2 5 个资源的资源结构

早开始执行时间, $\text{EFT}(T_i, R_j)$ 表示任务 T_i 在资源 R_j 上的最早完成时间. 对于入口任务,

$$\text{EST}(T_{\text{entry}}, R_j) = 0 \quad (1)$$

对于任务 DAG 中的非入口任务, EST 和 EFT 需要从入口任务开始递归计算, 计算方法如式 (2) 和 (3). 为了计算 T_i 的 EFT, T_i 的所有直接前驱任务必须保证已经完成.

$$\text{EST}(T_i, R_j) = \max\{\text{avail}[R_j], \max_{T_m \in \text{pred}(T_i)} (\text{AFT}(T_m) + c_{m,i})\} \quad (2)$$

$$\text{EFT}(T_i, R_j) = \omega_{i,j} + \text{EST}(T_i, R_j) \quad (3)$$

上式中, $\text{pred}(T_i)$ 为任务 T_i 的父任务集合, $c_{m,i}$ 表示任务 T_m 与 T_i 间的通信时间(数据传输时间), $\omega_{i,j}$ 表示任务 T_i 在资源 R_j 上的执行时间, $\text{avail}[R_j]$ 为资源 R_j 对于任务执行的最早就绪时间. 若任务 T_k 是资源 R_j 上最后的调度任务, 则 $\text{avail}[R_j]$ 为资源 R_j 完成 T_k 的时间, 即此时资源 R_j 处于就绪状态, 可执行其他任务. EST 等式中内层 max 即是返回该就绪时间, 即任务 T_i 所需的全部数据到达资源 R_j 的时间.

任务 T_m 调度至资源 R_j 后, T_m 在资源 R_j 上的最早开始时间和最早完成时间分别等于实际开始时间 $\text{AST}(T_m)$ 和实际完成时间 $\text{AFT}(T_m)$. 任务图中的所有任务被调度后, 调度长度 makespan(总体完成时间)即为出口任务 T_{exit} 的实际完成时间, 即:

$$\text{makespan} = \text{AFT}(T_{\text{exit}}) \quad (4)$$

给定每个任务在不同资源上的执行时间, 以及

资源的使用单价,即可得到任务执行代价。

移动云资源执行任务过程中,资源的总能耗 E_{total} 包括处理器执行任务时的动态能耗 E_{dyn} 和处理器空闲时的静态能耗 E_{sta} ,即单个资源的总能耗为:

$$E_{\text{total}} = E_{\text{dyn}} + E_{\text{sta}} \quad (5)$$

处理器资源的功耗计算为:

$$P = ACv^2 f \quad (6)$$

式中, A 表示每个时钟周期内的开关次数; C 表示有效电荷; v 表示处理器的供电电压; f 表示处理器的运行频率. 对于特定的处理器资源而言,参数 A 和 C 均为常量,因此,动态功耗主要被处理器的电压和频率所影响.

目前的处理器均配置有动态电压 / 频率调整 DVFS 功能,即处理器可运行于不同等级的电压 / 频率上. 设处理器资源的频率运行等级为五级,表示为 $f = \{f_1, f_2, f_3, f_4, f_5\}$,运行电压等级对应为 $V = \{v_1, v_2, v_3, v_4, v_5\}$,频率等级与电压等级一一对应,且依次递增. 令处理器最小频率为 $f_{\min}$,最大频率为 $f_{\max}$,最低电压等级为 $v_{\min}$,最高电压等级为 $v_{\max}$. 则 $f_{\min} = f_1, f_{\max} = f_5, v_{\min} = v_1, v_{\max} = v_5$. 当处理器处于空闲状态时,以最低频率等级保持运行,此时静态功耗为:

$$P_{\text{sta}} = ACv_{\min}^2 f_{\min} \quad (7)$$

当处理器资源执行任务时,以最高频率等级对任务进行处理,此时动态功耗为:

$$P_{\text{dyn}} = ACv_{\max}^2 f_{\max} \quad (8)$$

处理器资源完成最后一个任务后,即可完全关闭,此时能耗为 0. 对于单个处理器资源 R_j 而言,其执行任务 T_i 的动态能耗为:

$$E_{\text{dyn},j} = ACv_{\max}^2 f_{\max} w_{i,j} \quad (9)$$

对于单个处理器资源 R_j 而言,其静态能耗为:

$$E_{\text{sta},j} = ACv_{\min}^2 f_{\min} \Delta w_{\text{idle},j} \quad (10)$$

式中, $\Delta w_{\text{idle},j}$ 表示处理器 T_j 的空闲时间. 则处理器 T_j 的总能耗为:

$$E_{\text{total},j} = E_{\text{dyn}} + E_{\text{sta}} = ACv_{\max}^2 f_{\max} \sum_{i=1}^n w_{i,j} +$$

$$ACv_{\min}^2 f_{\min} \Delta w_{\text{idle},j} = AC(v_{\max}^2 f_{\max} \sum_{i=1}^n w_{i,j} + v_{\min}^2 f_{\min} \Delta w_{\text{idle},j}) \quad (11)$$

资源集合 R 执行任务集合 T 的总能耗为:

$$E_{\text{total}} = \sum_{j=1}^m E_{\text{total},j} = \sum_{j=1}^m AC(v_{\max}^2 f_{\max} \sum_{i=1}^n w_{i,j} + v_{\min}^2 f_{\min} \Delta w_{\text{idle},j}) \quad (12)$$

等式中,仅当任务 T_i 调度至资源 R_j 上执行时,

才以最大执行电压和频率计算此时的任务执行能耗.

3 最优化模型

给定云 workflow 任务集合 T 和移动云资源集合 R ,云任务调度的目标是产生调度解 S ,使得任务的执行总时间、总执行代价以及总执行能耗达到最小. 令 $\text{time}(T_i)$ 表示任务 T_i 的执行时间; $\text{cost}(T_i)$ 表示任务 T_i 的执行代价; $\text{energy}(T_i)$ 表示任务 T_i 的执行能耗; D 表示完成任务的截止时间约束; B 表示完成任务的预算约束; C 表示移动云资源的能量约束. 则多目标任务调度问题可形式化为:

$$\min \text{Time}(S) = \max_{T_i \in T} \text{time}(T_i) = \text{AFT}(T_{\text{exit}}) \quad (13)$$

$$\min \text{Cost}(S) = \sum_{T_i \in T} \text{cost}(T_i) \quad (14)$$

$$\min \text{Energy}(S) = \sum_{T_i \in T} \text{energy}(T_i) \quad (15)$$

约束条件为:

$$\text{Time}(S) \leq D, \text{Cost}(S) \leq B, \text{Energy}(S) \leq C \quad (16)$$

4 调度算法 MTSEA

4.1 编码

进化方法中每个染色体表示一个任务调度方案 S . 在 DAG 表示的任务结构中,一个任务调度方案 S 由任务的调度次序和任务与资源间的映射关系组成. 因此,一个染色体由任务调度序列 TSO 和任务-资源映射序列 TRM 组成. 以图 1 的任务图和图 2 的资源为例,图 3 所示为一个染色体个体 (TSO, TRM).

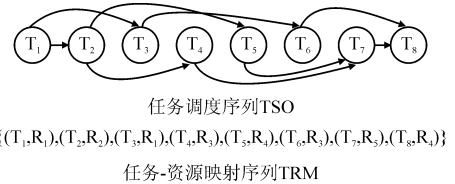


图 3 一个染色体个体

4.2 初始种群生成

常规的完全随机化种群初始化方式会导致解的搜索空间很大,进而导致进化方法收敛速度过慢. 为加速搜索过程,算法的种群初始化方法为:在不考虑其他优化目标的情况下,以三个单目标优化作为目标,分别产生三个种群个体,即以最早完成时间为目

标产生一个种群个体,以最小执行代价为目标产生一个种群个体,以最小执行能耗为目标产生一个种群个体,其余个体以随机化方法产生. 假设种群规模为 P_{size} , 过程具体为:

1) 以最早完成时间最小为目标产生一个种群个体.

首先,定义任务优先级,任务 T_i 在任务 DAG 中的优先级定义为:

$$priority(T_i) = w'_i + \sum_{T_j \in succ(T_i)} (c_{i,j} + \max_{T_j \in succ(T_i)} (priority(T_j))) \quad (17)$$

式中, $succ(T_i)$ 表示任务 T_i 的子任务集合; w'_i 表示任务 T_i 在所有资源上的平均计算时间. 任务优先级的计算需要以递归方式从出口任务开始计算. 而对于出口任务 T_{exit} , 由于其不存在任一子任务, 故其优先级为:

$$priority(T_{exit}) = w'_{exit} \quad (18)$$

然后,根据定义的任务优先级,对任务进行降序排序,即可得到任务的调度次序. 得到任务调度次序后,按照任务调度次序依次为任务选择最优调度资源,选择最优调度资源的原则是使得该任务在资源上能够得到最早完成时间 EFT.

令 $EST(T_i, R_j)$ 表示任务 T_i 在资源 R_j 上的最早开始执行时间, $EFT(T_i, R_j)$ 表示任务 T_i 在资源 R_j 上的最早完成时间. 对于入口任务:

$$EST(T_{entry}, R_j) = 0 \quad (19)$$

对于任务 DAG 中的非入口任务, EST 和 EFT 需要从入口任务开始递归计算,计算方法如式(2)和(3). 为了计算 T_i 的 EFT, T_i 的所有直接前驱任务必须保证已经完成.

$$EST(T_i, R_j) = \max\{avail[R_j], \max_{T_m \in pred(T_i)} (AFT(T_m) + c_{m,i})\} \quad (20)$$

$$EFT(T_i, R_j) = w_{i,j} + EST(T_i, R_j) \quad (21)$$

式中, $pred(T_i)$ 为任务 T_i 的父任务集合, $avail[R_j]$ 为资源 R_j 对于任务执行的最早就绪时间. 若任务 T_k 是资源 R_j 上最后的调度任务, 则 $avail[R_j]$ 为资源 R_j 完成 T_k 的时间, 即此时资源 R_j 处于就绪状态, 可执行其他任务. EST 等式中内层 $\max$ 即是返回该就绪时间, 即任务 T_i 所需的全部数据到达资源 R_j 的时间.

最优资源选择必须满足任务得到最小的最早完成时间 EFT, 即:

$$\min_{R_j \in R} EFT(T_i, R_j), T_i \in T \quad (22)$$

任务 T_m 调度至资源 R_j 后, T_m 在资源 R_j 上的最早开始时间和最早完成时间分别等于实际开始时间 $AST(T_m)$ 和实际完成时间 $AFT(T_m)$. 任务图中所有任务被调度后, 调度长度 $makespan$ (总体完成时间) 即为出口任务 T_{exit} 的实际完成时间, 即:

$$makespan = AFT(T_{exit}) \quad (23)$$

以上调度结果即为最早完成时间最小的任务调度方案, 以此作为一个种群个体.

2) 以最小执行代价为目标产生一个种群个体.

首先,按上文 1) 的方法计算任务优先级,根据任务优先级,对任务进行降序排序,即可得到任务的调度次序. 得到任务调度次序后,按照任务调度次序依次为任务选择最优调度资源,选择最优调度资源的原则是使得该任务在资源上能够得到最小执行代价. 得到每个任务的调度资源后,该调度结果即为最小执行代价的任务调度方案,以此作为一个种群个体.

3) 以最小执行能耗为目标产生一个种群个体.

以上文 1) 产生的种群个体作为初始调度方案,并评估在初始调度方案下的资源能效. 令 EE_j 表示资源 R_j 的能效值,其含义是资源 R_j 实际执行任务的动态能耗占资源 R_j 开启状态下的总体能耗的比例,资源 R_j 开启状态下的总体能耗为资源执行的动态能耗与资源空闲时的静态能耗之和,即:

$$EE_j = \frac{E_{dyn,j}}{E_{total,j}} = \frac{E_{dyn,j}}{E_{dyn,j} + E_{sta,j}} \quad (24)$$

得到每个资源的能效值后,按能效值对资源降序排列,删除能效值最小的资源. 更新资源集合,再次按照 1) 的方法重新计算任务调度方案,关闭能效值最小的资源,以此形成最小执行能耗的任务调度方案,并以此作为第三个种群个体.

4) 以随机方式产生剩余 $P_{size} - 3$ 个种群个体.

4.3 适应度评估

适应度函数用于在给定的优化目标时度量解的质量. 算法将适应度函数分解为目标函数和惩罚函数,目标函数使得方法寻求使得目标函数值最小的调度解,对于调度解 S 的目标函数为:

$$\text{时间目标函数: } f_{Time}(S) = Time(S)/D$$

$$\text{代价目标函数: } f_{Cost}(S) = Cost(S)/B$$

$$\text{能量目标函数: } f_{Energy}(S) = Energy(S)/C$$

惩罚函数用于处理违背约束条件时对适应度的调整. 设 $P(S)$ 表示惩罚函数, 定义为:

$$P(S) = P_{Deadline}(S) + P_{Budget}(S) + P_{Capacity}(S) \quad (25)$$

式中, $P_{\text{Deadline}}(S)$ 表示截止时间惩罚函数, 定义为:

$$P_{\text{Deadline}}(S) = \begin{cases} f_{\text{Time}}(S), & \text{Time}(S) > D \\ 0, & \text{otherwise} \end{cases} \quad (26)$$

P_{Budget} 表示预算惩罚函数, 定义为:

$$P_{\text{Budget}}(S) = \begin{cases} f_{\text{Cost}}(S), & \text{Cost}(S) > B \\ 0, & \text{otherwise} \end{cases} \quad (27)$$

P_{Capacity} 表示能量惩罚函数, 定义为:

$$P_{\text{Capacity}}(S) = \begin{cases} f_{\text{Energy}}(S), & \text{Energy}(S) > C \\ 0, & \text{otherwise} \end{cases} \quad (28)$$

当调度解未满足三个约束中的任意一个约束时, 即视为不可行解, 因此, 同时满足三个约束是任务调度的基本目标. 最后, 适应度函数定义为:

时间适应度函数: $F_{\text{Time}}(S) = f_{\text{Time}}(S) + P(S)$

代价适应度函数: $F_{\text{Cost}}(S) = f_{\text{Cost}}(S) + P(S)$

能量适应度函数: $F_{\text{Energy}}(S) = f_{\text{Energy}}(S) + P(S)$

4.4 种群进化

种群个体进化包括个体交叉与变异操作. 个体交叉包括染色体中任务调度序列 TSO 的交叉操作和任务-资源映射序列 TRM 的交叉操作. 任务调度序列 TSO 的交叉操作的具体方法为: 随机选择两个任务调度序列作为父代 1 和父代 2, 并在两个任务调度序列的相同位置随机选择一个交叉点, 该交叉点将任务调度序列分割为两部分, 将分割的右半部分进行互换, 得到两个新的染色体 TSO 成份, 作为子代 1 和子代 2. 图 4 为一个任务调度序列 TSO 的交叉示例.

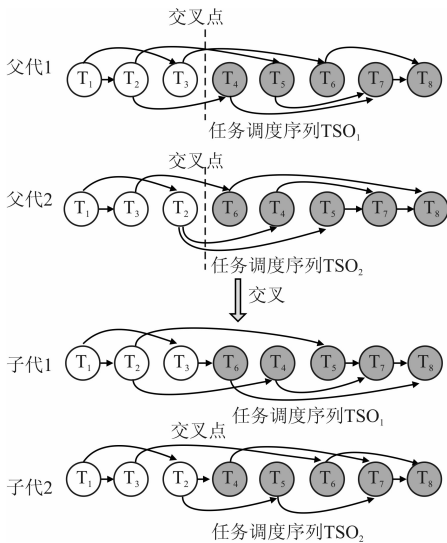


图 4 任务调度序列 TSO 的交叉

任务-资源映射序列 TRM 的交叉操作具体方法为: 随机选择两个任务-资源映射序列作为父代 1 和父代 2, 并在两个任务-资源映射序列的相同位置随机选择一个交叉点, 该交叉点将任务-资源映射序列分割为两部分, 将分割的右半部分进行互换, 得到两个新的染色体 TRM 成份, 作为子代 1 和子代 2. 图 5 为一个任务-资源映射序列 TRM 的交叉示例.



图 5 任务-资源映射序列 TRM 的交叉

个体变异包括染色体中任务调度序列 TSO 的变异操作和任务-资源映射序列 TRM 的变异操作. 任务调度序列 TSO 的变异操作的具体方法为: 对于一个给定的任务调度序列, 随机选择两个变异任务, 在不违背任务间的执行顺序约束的情况下, 互换两个任务在任务调度序列中的位置. 图 6 为一个任务调度序列 TSO 的变异示例.

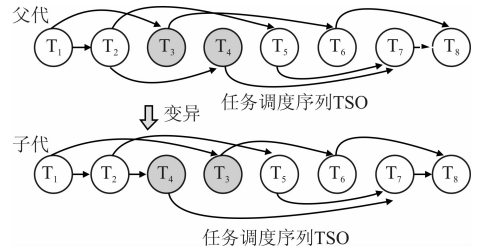


图 6 任务调度序列 TSO 的变异

任务-资源映射序列 TRM 的变异操作采用的表现形式为 $DE/x/y/z$, x 表示变异操作中差异向量进行重组的基准个体是随机个体还是当前种群中的最优个体, y 表示参与变异重组的差异向量的个数, z 表示重组采用的方式, 主要有指数重组方式和二项式重组方式. 算法采用的变异策略为 $DE/\text{rand}/1/\text{bin}$, 即:

$$P = P_1 + F(P_2 - P_3) \quad (29)$$

式中, P_1 为基准个体; P_2 和 P_3 为参考个体, 新的个体 P 由基准个体与两个参考个体 P_2 与 P_3 的差异向量重组形成. 仅考虑映射资源的不同, 假设个体 P_2 的任务-资源映射序列 $P_2(\text{TRM}) = \{(T_1, R_1), (T_2, R_2), (T_3, R_3), (T_4, R_4), (T_5, R_5), (T_6,$

$R_1), (T_7, R_2), (T_8, R_3)\}$, 个体 P_3 的任务-资源映射序列 $P_3(\text{TRM}) = \{(T_1, R_1), (T_2, R_3), (T_3, R_2), (T_4, R_4), (T_5, R_5), (T_6, R_2), (T_7, R_1), (T_8, R_3)\}$, 将 P_2 称为模式矢量, P_3 称为无序矢量, $P_2 - P_3$ 表示两个任务-资源映射序列的差距, 即 P_3 可以通过最少几次在资源上插入操作可还原至 P_2 , 可以得出, 此时 $P_2 - P_3 = 2$, 即将 $P_3(\text{TRM})$ 中的 (T_3, R_2) 中的 R_2 插入 (T_2, R_3) 中的 R_3 , R_3 顺延, (T_7, R_1) 中 R_1 插入 (T_6, R_2) 中的 R_2 , 并将 R_1 顺延, 即可还原为 P_2 . 然后, 以得到最少插入操作次数对变异个体 P_1 进行变异操作. 令 $D = P_2 - P_3$, 以 D 作为变异次数, 对个体 P_1 中 TRM 实施变异, 得到一个新的 TRM', 按前文的 TSO 变异方法对 P_1 中的 TSO 实施变异得到一个新的 TSO', 将随机选择的一个父代个体的 TSO 与 TSO' 进行交叉操作即可得到新个体的 TSO, 将父代个体的 TRM 与 TRM' 进行交叉操作即可得到新个体的 TRM, 以此得到一个新的候选个体.

4.5 MTSEA 算法实施步骤

步骤 1: 以种群初始化方法, 产生 P_{size} 个种群个体, 标记为 $P = \{P_1, P_2, \dots, P_{\text{size}}\}$;

步骤 2: 当迭代次数小于最大次数时, 执行步骤 3;

步骤 3: 对于每一个种群个体, 随机选择一个父代个体, 并按前文变异操作产生一个候选个体, 并评估候选个体的适应度; 若候选个体占优父代个体, 则以候选个体取代父代个体保留至种群; 若父代个体占优候选个体, 则以父代个体取代候选个体保留至种群; 若两者之间不存在占优关系, 则将候选个体直接保留至种群, 且种群规模加 1;

注: 个体占优关系定义为: 若个体 P_1 代表的调度解 S 的三个优化目标 $\text{Time}(S)$ 、 $\text{Cost}(S)$ 和 $\text{Energy}(S)$ 均小于个体 P_2 , 则 P_1 占优 P_2 ; 否则, P_1 与 P_2 为非占优关系, 所代表的解为非占优解.

步骤 4: 若最终的种群规模超过 P_{size} , 则将适应度低的个体移出种群, 此时得到的种群为任务调度解集合.

5 实验分析

5.1 实验环境与性能指标

利用 CloudSim 仿真平台^[10]对算法进行实验分析. 选择 Montage 科学工作流进行实验测试. 所构建的云环境中, 云资源的处理器速率为 $[1000, 10000]$ MIPS 间的随机值, 单位 MIPS 是 CloudSim

平台中对于资源处理能力度量的单位, 表示处理器每秒钟可以执行的百万个指令数量. 虚拟机资源定价参考 Amazon EC2 的资源定价机制.

三种约束采用了非固定值的设置方式, 以便考察算法在同步满足约束条件下的能力. 截止时间约束定义方式为:

$$D = \text{LB}_D + \alpha \times (\text{UB}_D - \text{LB}_D) \quad (30)$$

式中, LB_D 表示初始种群生成方法中以最早完成时间最小为目标生成的种群个体所得到的任务执行跨度时间; UB_D 设置为 3 倍的 LB_D ; α 表示截止时间因子, $\alpha \in [0, 1]$.

预算约束定义方式为:

$$B = \text{LC}_B + \beta \times (\text{UC}_B - \text{LC}_B) \quad (31)$$

式中, LC_B 表示初始种群生成方法中以最小执行代价为目标生成的种群个体所得到的任务执行代价, UC_B 设置为将任务调度至代价最高的资源上执行时得到的执行代价, β 表示预算因子, $\beta \in [0, 1]$.

能量约束定义方式为:

$$C = \text{LE}_C + \gamma \times (\text{UE}_C - \text{LE}_C) \quad (32)$$

式中, LE_C 表示初始种群生成方法中以最小执行能耗为目标生成的种群个体所得到的任务执行能耗; UE_C 设置为 3 倍的 LE_C ; γ 表示能量因子, $\gamma \in [0, 1]$.

选择世代距离 GD 和间隔 spacing ^[8] 两个指标评估算法性能. 世代距离 GD 是评估相对边界时算法收敛性的指标, 表示为:

$$\text{GD} = \left(\sum_{i=1}^{|Q|} d_i^2 \right) 1/2 / |Q| \quad (33)$$

式中, d_i 为 Q 的解与相对边界的最近解间的欧氏距离; Q 为计算 GD 指标时算法得到的边界.

间隔 spacing 指标则用于评估调度解的多样性, 可表示为:

$$\text{spacing} = \sqrt{1/Q} \sum_{i=1}^{|Q|} (d_i - \bar{d})^2 \quad (34)$$

式中, d_i 为该解与 Q 的最近解间的距离, 不同于欧氏距离; $\bar{d}$ 为距离度量 d_i 的平均值. 对于进化算法而言, GD 和 spacing 指标均越小对算法越有利.

选择多目标最优化进化遗传工作流调度算法 MOEGA^[4]、基于非占优排序的混合多目标粒子群优化调度算法 HPSO^[8] 作为性能的对比基准算法. 与本文相似, 基准算法也均是基于智能种群优化的调度算法, 因此可以进行相同的参数配置, 包括种群大小均设置为 50, 交叉概率均设置为 0.6, 变异概率均设置为 0.4.

5.2 实验分析

实验观察了三个约束因子在不同取值下算法两个性能指标的表现情况,结果如图 7~图 9 所示. 图中的横坐标为相应约束因子的不同取值,左侧的纵坐标为指标 GD 的均值,对应的图形为柱状图,右侧的纵坐标为指标 spacing 均值,对应的图形为折线图. 结果显示,放宽约束因子的取值,算法的性能会变得更好,原因在于若约束因子取值更大,则相应约束更显宽松,调度算法在为任务选取执行资源时更可能选择更为合适更为高效的资源进行任务调度,其得到的调度解的性能也越好. 相比而言,本文设计的 MTSEA 算法得到了更小的世代距离 GD 和间隔 spacing,这是由于 MTSEA 算法基于进化模式的均衡调度得到了非占优的解集合,初始解中融入三种更加有效的调度初始方案也使得后续解的生成过程中保持非占优解的概率更大,其收敛性能会更好. MOEGA 算法因为并未考虑预算和能量的约束问题,因此改变预算因子和能量因子对其调度解的性能不产生直接影响,所以在两个指标上基本是平滑的曲线. HPSO 算法考虑了三个目标的优化,但没有考虑约束,因此在较宽松的约束条件下依然可以得到性能表现不错的调度解,但若约束较紧,即较小的约束因子值,其调度方案选择将极大受限,得到帕累托解集的分布不如本文算法紧致.

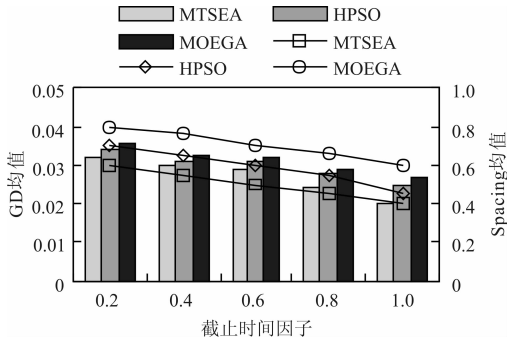


图 7 截止时间因子变化时算法性能变化

6 结束语

提出了一种移动云计算中的多目标任务调度进化算法. 算法的主要步骤是: 首先, 进行种群初始化; 其次, 以每一个初始化种群个体作为父代个体, 在其基础上产生一个候选解, 并根据适应度函数评估候选解的适应度; 然后, 比较父代个体与候选个体在三个适应度函数上的占优关系, 具体为: 若候选个体占优父代个体, 则以候选个体取代父代个体进入种群; 若父代个体占优候选个体, 则以父代个体取代候选

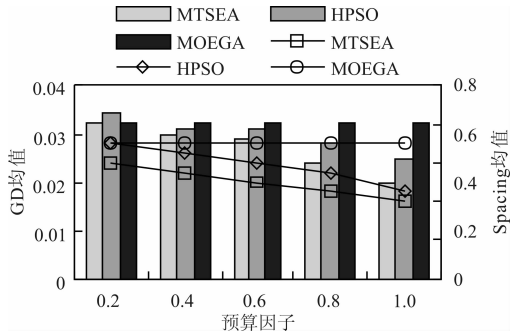


图 8 预算因子变化时算法性能变化

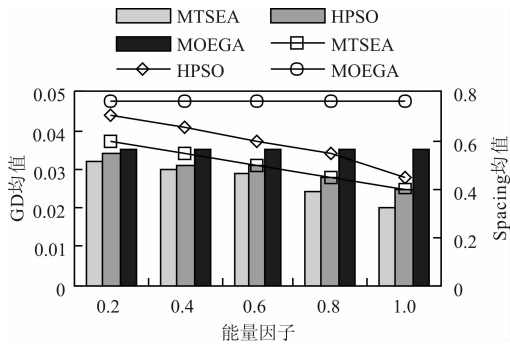


图 9 能量因子变化时算法性能变化

个体; 若两者之间不存在占优关系, 则将候选个体保留至种群中. 算法可以在任务执行时间、执行代价和执行能耗三个目标上实现同步均衡的优化, 形成多目标优化的调度解集合. 仿真实验结果证明了算法的先进性.

参考文献:

[1] ZHENG W, SAKELLARIOU R. Budget-deadline constrained workflow planning for admission control [J]. Journal of Grid Computing, 2013, 11(4): 633-651.

[2] ARABNEJAD H, BARBOSA J G. A budget constrained scheduling algorithm for workflow applications [J]. Journal of Grid Computing, 2014, 12(4): 1-15.

[3] LI K W, ZHANG G X, ZHU Z M. A decomposition-based multi-objective workflow scheduling algorithm in cloud environment [J]. Computer Engineering & Science, 2016, 38(8): 1588-1594.

[4] WANG G H, LI Q H, LIU A F. Evolution genetic algorithm of multi-objective optimization scheduling on cloud workflow [J]. Computer Science, 2018, 45(5): 31-37.

[5] YUAN Y W, BAO Z Q, YU D J, et al. Multi-scientific workflow scheduling algorithm based on multi-objective in cloud environment [J]. Journal of Software, 2018, 29(11): 1-15.

[6]

HUANG Q, SU S, LI J, et al. Enhanced Energy-Efficient Scheduling for Parallel Applications in Cloud [C]//IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, USA, IEEE, 2013; 781-786.

[7]

THANAVANICH T, UTHAYOPAS P. Efficient energy aware task scheduling for parallel workflow tasks on hybrids cloud environment[C]//International Computer Science and Engineering Conference, USA, IEEE, 2013; 37-42.

[8]

DU Y M, XIAO J H. Scientific workflow scheduling algorithm based on hybrid multi-objective particle swarm optimization in cloud environment[J]. Computer Science, 2017, 44(8): 252-259.

[9]

XIE Y, HE T, NI Q, et al. Scheduling for improving the energy efficiency of cloud workflow execution [J]. Systems Engineering-Theory & Practice, 2017, 37(4): 1056-1071.

[10]

RODRIGO N, RAJIV R, ANTON B, et al. Cloud-Sim: A Toolkit for Modeling and Simulation of Cloud Computing Environments and Evaluation of Resource Provisioning Algorithms [J]. Software Practice & Experience, 2010, 41(1): 23-50.

作者简介:

张小庆 男, (1984-), 博士, 讲师. 研究方向为移动计算.
E-mail: 51449902@qq. com
安春玲 女, (1987-), 硕士, 中级馆员. 研究方向为数字化图书馆建设.
胡亚捷 男, (1997-), 学士. 研究方向为软件工程。

(上接第 78 页)

作者简介:

袁艺丹 女, (1986-), 硕士, 讲师. 研究方向为模拟集成电路与功率集成电路设计.
罗 谦 男, (1976-), 博士, 副教授. 研究方向为化合物半导体异质结物理, GaN 基高电子迁移率晶体管器件物理及制

备工艺等.
贺江平 男, (1976-), 博士, 讲师. 研究方向为模拟集成电路与功率集成电路的设计.
吴克军(通讯作者) 男, (1986-), 博士, 博士后. 研究方向为模拟数字混合集成电路设计.
E-mail: kjwu@uestc. edu. cn