控制与决策
Control and Decision

基于预防维护的单机调度问题

吴慧， 王冰

引用本文:
吴慧, 王冰. 基于预防维护的单机调度问题[J]. 控制与决策, 2021, 36(2): 395−402.

在线阅读 View online: https://doi.org/10.13195/j.kzyjc.2019.0626

基于预防维护的单机调度问题

吴 慧[1,2], 王 冰[1†]

(1. 上海大学 机电工程与自动化学院，上海 200444；
2. 青岛农业大学 理学与信息科学学院，山东 青岛 266109)

摘 要: 在两种维护约束下,研究完工时间之和最小化的单机调度问题. 第1种维护约束是,固定周期预防维护;第2种维护约束是,机器工作期间可连续加工的最大工件个数受限. 对于这种带有约束的调度问题,根据问题的规模,采用4种方法进行求解. 针对小规模问题,建立一个二值整数规划模型,并根据最优解的特性制定剪枝规则,进而给出分支定界算法. 针对中、大规模问题,采用遗传算法进行求解,为缓解遗传算法中常见的早熟问题,对变异算子进行改进,采用动态变异方法,提出动态遗传算法. 最后通过仿真实验对各种算法进行性能评估.

关键词: 单机调度；预防维护；二值整数规划；分支定界算法；遗传算法；动态遗传算法

中图分类号: TP18 **文献标志码:** A

DOI: 10.13195/j.kzyjc.2019.0626 **开放科学(资源服务)标识码(OSID):**

引用格式: 吴慧, 王冰. 基于预防维护的单机调度问题[J]. 控制与决策, 2021, 36(2): 395-402.

Single-machine scheduling problem with preventative maintenance activities

WU Hui[1,2], *WANG Bing*[1†]

(1. School of Mechatronic Engineering and Automation，Shanghai University，Shanghai 200444，China；2. Science and Information College，Qingdao Agricultural University，Qingdao 266109，China)

Abstract: This paper deals with a single-machine scheduling problem with the objective to minimize the total completion time of jobs subjects to two maintenance constraints. The first maintenance constraint is that the machine will be stopped for maintenance after a periodic interval, and the second one is the constraint on the maximum number of jobs continuously processed. Four methods are used to solve the constrained single-machine scheduling problem. To solve small-scale problems, one binary integer programming model is first proposed, then pruning rules are formulated according to the properties of optimal solutions, and the corresponding branch and bound algorithm is proposed. The genetic algorithm is used to find the near-optimal solutions for medium-scale and large-scale problems. Dynamic genetic algorithm is proposed by adopting the dynamic mutation method to alleviate the precocity problem which is a phenomenon that cannot be ignored in the genetic algorithm. Finally, computational experiments are carried out to evaluate the performance of the proposed algorithms.

Keywords: single machine scheduling；preventative maintenance；binary integer programming；branch-and-bound algorithm；genetic algorithm；dynamic genetic algorithm

0 引 言

传统调度问题通常假设调度期间机器可连续使用,然而实际的加工制造系统并非如此,多种原因会导致机器在调度期间无法连续使用,比如预防维护(确定性)或机器故障(随机性)[1]. 相比于预防维护,机器故障的维修成本更高,引发的后果也更为严重. 实施预防维护,有助于提高机器的稳定性,是降低机器发生故障的有效手段. 因此,在调度问题中考虑预防维护更具有实际意义.

自二十世纪九十年代,基于预防维护的单机调度问题开始引起国内外学者的广泛关注[2-6]. 对于造价昂贵、专业性能较强的大型精密机器,厂家需要派专业技术人员进行上门维护. 为了实现工件加工和设备维护的有效结合,一般采用周期维护,其维护周期和维护时间都是预先确定的. 因此,对工件调度和周期维护进行集成研究,构建更加符合实际的数学模型并进行求解,具有较高的研究价值和实际意义[7-12].

虽然基于预防维护的单机调度问题已得到广泛关注,但大多数研究仅考虑了一种维护约束,即固定周期维护. 实际上,在加工制造系统中,机器的过度使

收稿日期: 2019-05-08；**修回日期:** 2019-09-21.
†通讯作者. E-mail: susanbwang@shu.edu.cn.

用会降低工件的质量. 因此,当工件造价昂贵或者对精度有较高要求时,为确保产品质量,需停机更换零件. 一个较为典型的例子就是印刷电路板的制造系统,钻床作为最重要的设备之一,加工过程中不仅需要停机进行预防维护,使用一定次数后还需要停机更换微钻. 针对此类问题,为确保工件质量,不仅要对机器进行周期维护,每个加工周期内允许加工的最大工件数量也要受到严格限制. 基于这两种约束导致的停机势必会影响工件的完成时间和生产效率. 因此,如何调度工件以提高整体性能,是一个非常重要的科研问题. Low等[13-14]基于这两种维护约束,对目标为最小化最大完成时间的单机调度问题分别关于定期维护和柔性维护进行了研究. 随后,Hsu等[15]对同一问题做了进一步研究,提出了两阶段二值整数规划模型和两个启发式算法;Zade等[16]提出了一个新的数学规划模型和动态遗传算法. 在相同的维护约束下,Shahriari等[17]研究了单机调度的双目标优化问题,目标是使提前/拖期完工惩罚之和与最大完成时间同时最小化,为解决这类问题,他们提出了多目标粒子群优化算法.

目前,基于预防维护的单机调度问题有两大研究热潮:一是在两种维护约束下研究最小化最大完成时间问题[13-16],二是在一种维护约束下研究最小化工件完成时间之和问题[1-4,18-19]. 在两种维护约束下,目标函数为工件完成时间之和的问题还尚未被研究. 为弥补这一领域的研究空白,本文将两种维护约束与工件调度相结合,对最小化完成时间之和的单机调度问题建立集成模型,展开基础研究.

1 问题描述和符号表示

单机环境下,n个工件$J_1, J_2, \dots, J_n$相互独立且在零时刻可用,工件在加工过程中若因机器维护导致中断则须重新加工. 机器一次只能加工一个工件,此外,还需满足两种维护约束:第1种维护约束是周期维护,机器运行T时间必须停机进行预防维护,维护时长为t;第2种维护约束是每个加工周期T内允许加工的工件数量受限,不能超过K. 维护期间,机器不能加工工件. 调度问题的目标是将工件的完成时间之和最小化.

为表述方便,将连续加工的工件集合看作一个批次,记为B. 在任意一个批次内的工件个数不超过K,加工时间之和不超过T,在固定周期T后对机器进行维护,维护时长为t. 该问题可通过图1表示,其中M_l表示第l个维护活动,B_l表示第l个批次. 可行调度π记为$\pi = (B_1, M_1, B_2, M_2, \dots, M_{L-1}, B_L)$,表示一共有$L$个批次.

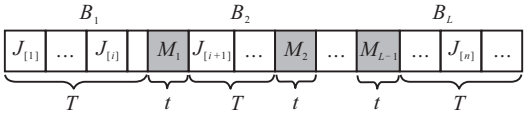

图1 基于两种维护约束的单机调度问题

根据Graham等[20]的三元组表示法,本文研究的问题可记为$1|\mathrm{nr}, \mathrm{pm}-\mathrm{mn}|\sum C_i$. 其中:1表示单机系统,nr表示加工过程中工件不可中断,$\mathrm{pm}-\mathrm{mn}$表示两种维护约束,$\sum C_i$表示目标函数.

其他符号说明如下:

p_i: 工件J_i的加工时间;

C_i: 工件J_i的完成时间;

n_l: 批次B_l中的工件个数($n_l \leqslant K$);

I_l: 批次B_l结束与维护活动M_l开始之间的闲置时间,$I_l = T - \sum\limits_{J_i \in B_l} p_i, l = 1, 2, \dots, L-1$;

$J_{[i]}$: 排在第i个位置的工件;

$p_{[i]}$: 工件$J_{[i]}$的加工时间;

$C_{[i]}$: 工件$J_{[i]}$的完成时间;

$J_{[jl]}$: 位于批次B_l中第j个位置的工件;

$p_{[jl]}$: 工件$J_{[jl]}$的加工时间;

M: 非常大的正数.

2 二值整数规划模型

由于不能预先确定最优的批次个数以及每个批次内工件的个数,用最大可能的批次个数及每个批次内最大可能的工件个数作为取值上界. 假设有n个工件,则最多有n个批次,每个批次所含工件个数最多为K. 因为某些批次可能含多个工件,这将导致部分批次为空,在下述规划模型中将对这些空批次忽略不计. 记该问题的决策变量为

$$x_{ijl} = \begin{cases} 1, & J_i\text{位于}B_l\text{中第}j\text{个位置;} \\ 0, & \text{否则.} \end{cases}$$

$1|\mathrm{nr}, \mathrm{pm}-\mathrm{mn}|\sum C_i$的二值整数规划模型为

$$\min\ z = \sum_{i=1}^{n} C_i. \tag{1}$$

$$\text{s.t.} \sum_{j=1}^{K}\sum_{l=1}^{n} x_{ijl} = 1,\ i = 1, 2, \dots, n; \tag{2}$$

$$\sum_{i=1}^{n} x_{ijl} \leqslant 1,\ j = 1, 2, \dots, K,\ l = 1, 2, \dots, n; \tag{3}$$

$$\sum_{i=1}^{n}\sum_{j=1}^{K} p_i x_{ijl} \leqslant T,\ l = 1, 2, \dots, n; \tag{4}$$

$$\sum_{i=1}^{n}\sum_{j=1}^{K} x_{ijl} \leqslant K,\ l = 1, 2, \dots, n; \tag{5}$$

$$C_{ijl}=$$

$$(T+t)(l-1)x_{ijl}+\Big(\sum_{i'=1}^{n}\sum_{j'=1}^{j}p_{i'}x_{i'j'l}\Big)x_{ijl},$$

$$i=1,2,\ldots,n,\ j=1,2,\ldots,K,\ l=1,2,\cdots,n; \tag{6}$$

$$C_i=\sum_{j=1}^{K}\sum_{l=1}^{n}C_{ijl},\ i=1,2,\ldots,n; \tag{7}$$

$$\sum_{i=1}^{n}\sum_{j=1}^{K}x_{ijl}\cdot M\geqslant\sum_{i=1}^{n}\sum_{j=1}^{K}x_{ijl+1},$$

$$l=1,2,\ldots,n-1; \tag{8}$$

$$\sum_{i=1}^{n}x_{ijl}\geqslant\sum_{i=1}^{n}x_{ij'l},$$

$$j\leqslant j',\ j,j'=1,2,\ldots,K,\ l=1,2,\ldots,n; \tag{9}$$

$$x_{ijl}\in\{0,1\},$$

$$i=1,2,\ldots,n,\ j=1,2,\ldots,K,\ l=1,2,\ldots,n. \tag{10}$$

其中: 模型以式(1)最小化完成时间之和为目标; 约束(2)表示一个工件只能在一个位置上加工; 约束(3)表示每个位置最多分配一个工件; 约束(4)表示每个批次内工件的加工时间之和不超过T; 约束(5)表示每个批次内允许加工的工件个数不超过K; 约束(6)表示位于批次B_l中第j个位置的工件J_i的完成时间; 约束(7)表示工件J_i的完成时间; 约束(8)表示若批次B_l为空, 则批次B_{l+1}一定为空; 约束(9)表示在同一批次中一个位置为空, 则其后续位置也为空; 约束(10)表示决策变量x_{ijl}为0-1变量.

3　最优解的性质

性质1　$1|\mathrm{nr},\mathrm{pm}-\mathrm{mn}|\sum C_i$是强NP-hard问题.

证明　Qi等[3]借助于三划分问题证明了问题$1|\mathrm{nr},\mathrm{pm}|\sum C_i$是强NP-hard问题, 具体证明过程参见文献[3], 而$1|\mathrm{nr},\mathrm{pm}-\mathrm{mn}|\sum C_i$的复杂度更高. □

性质2　在$1|\mathrm{nr},\mathrm{pm}-\mathrm{mn}|\sum C_i$的最优解中, 同一批次内的工件是按照SPT (shortest processing time)规则排列的.

证明　反证法. 通过交换同一批次内的任意两个工件, 比较其完成时间之和, 即可证明. □

性质3　在$1|\mathrm{nr},\mathrm{pm}-\mathrm{mn}|\sum C_i$的最优解中, 对于任意批次$B_r$与$B_k$, 如果$r<k$, 则$n_r\geqslant n_k$.

证明　反证法. 假设在最优调度π中, 存在B_r与$B_k, r<k, n_r<n_k$, 如图2所示.

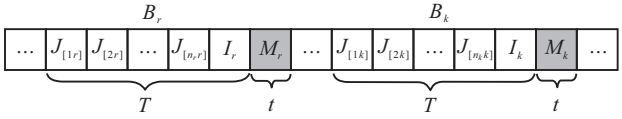

图2　调度π(性质3)

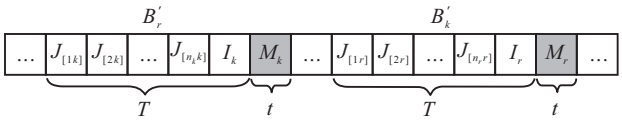

图3　调度π'(性质3)

将批次B_r中所有工件与B_k中所有工件进行整体交换, 得到可行调度π', 如图3所示.

分析可得

$$z(\pi')=$$
$$z(\pi)-n_k(k-r)(T+t)+n_r(k-r)(T+t)=$$
$$z(\pi)-(n_k-n_r)(k-r)(T+t).$$

由$r<k, n_r<n_k$, 可知$(n_k-n_r)(k-r)(T+t)>0$, 因此有$z(\pi')<z(\pi)$, 这与假设矛盾. □

性质4　在$1|\mathrm{nr},\mathrm{pm}-\mathrm{mn}|\sum C_i$的最优解中, 如果$n_l<K$, 则对于任意$J_k\in B_r, r=l+1,l+2,\ldots,L$, 有$I_l<p_k$.

证明　反证法. 假设在最优调度π中, $n_l<K$, 存在$J_k\in B_r, r>l$使得$I_l\geqslant p_k$. 根据性质2, 最优调度中同一批次内的工件是按照SPT规则排列的, 因此可假定J_k位于B_r中第1个位置, 如图4所示.

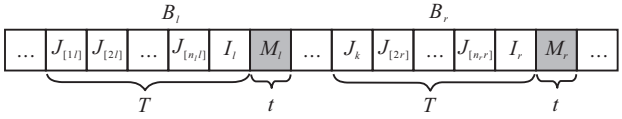

图4　调度π(性质4)

由$n_l<K, I_l\geqslant p_k$可知, 将J_k从B_r中移除并插入到B_l的尾部是可行的, 记为可行调度π', 如图5所示.

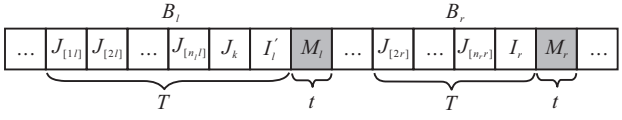

图5　调度π'(性质4)

分析可得

$$z(\pi')=$$
$$z(\pi)-(r-l)(T+t)+(T-I_l)-(n_r-1)p_k=$$
$$z(\pi)-(r-l)(T+t)-(n_r-1)p_k+(T-I_l)=$$
$$z(\pi)-[(r-l)(T+t)-T+I_l-p_k+n_rp_k].$$

由$r>l, I_l\geqslant p_k$, 可知$(r-l)(T+t)-T+I_l-p_k+n_rp_k>0$. 因此$z(\pi')<z(\pi)$, 这与假设矛盾. □

4　求解算法

4.1　分支定界算法

调度π的总完成时间$z(\pi)$由如下两部分组成:

$$z(\pi)=\sum_{l=2}^{L}n_l\cdot(l-1)\cdot(T+t)+$$

$$\sum_{l=1}^{L}\sum_{j=1}^{n_l}(n_l-j+1)\cdot p_{[jl]}. \tag{11}$$

第1部分由固定周期和维护时间组成,第2部分由工件的加工时间组成.

4.1.1 下 界

在分支定界树中,每个结点对应一个局部调度. 结点D将工件分成两部分:已调度工件和未调度工件. 假设有n_D个工件已调度,用$S_D=(J_{[1]},J_{[2]},\ldots,J_{[n_D]})$表示对应的序列,fn和ft分别表示$S_D$中最后一个批次内的工件个数和工件的加工时间之和,sn表示S_D中倒数第2个批次所含工件个数,$z_1(D)$表示n_D个已调度工件的总完成时间,$z_2(D)$表示$n-n_D$个未调度工件的总完成时间,US_D表示$n-n_D$个未调度工件组成的集合. 显然

$$z_1(D)=\sum_{i=1}^{n_D}C_{[i]},$$

而$z_2(D)$需要估算. 假设π'是US_D对应的一个序列,n_i'是π'中批次B_i'所含工件个数. 显然,在两种维护约束下,π'是SPT调度,且机器无闲置时$z_2(D)$最小,此时加工时间对应部分为

$$(n-n_D)C_{[n_D]}+(n-n_D)p_{[n_D+1]}+$$
$$(n-n_D-1)p_{[n_D+2]}+\ldots+p_{[n]},$$

维护时间对应部分为

$$n_2'\cdot t+n_3'\cdot 2t+\ldots\geqslant n_2'\cdot t+n_3'\cdot t+\ldots=$$
$$(n-n_D-n_1')\cdot t\geqslant(n-n_D-\mathrm{sn}+\mathrm{fn})\cdot t.$$

显然,

$$z_2(D)\geqslant(n-n_D)C_{[n_D]}+(n-n_D)p_{[n_D+1]}+$$
$$(n-n_D-1)p_{[n_D+2]}+\ldots+p_{[n]}+$$
$$(n-n_D-\mathrm{sn}+\mathrm{fn})\cdot t.$$

因此,结点D处的下界可表示为

$$z_L(D)=\sum_{i=1}^{n_D}C_{[i]}+(n-n_D)C_{[n_D]}+(n-$$
$$n_D)p_{[n_D+1]}+(n-n_D-1)p_{[n_D+2]}+$$
$$\ldots+p_{[n]}+(n-n_D-\mathrm{sn}+\mathrm{fn})\cdot t. \tag{12}$$

这里$p_{[n_D+1]},p_{[n_D+2]},\ldots,p_{[n]}$是按照非减顺序排列.

4.1.2 剪枝规则

在分支定界算法中,合理的剪枝规则能减少不必要的搜索,有效提高计算效率. 根据最优解的性质,可得到如下几条剪枝规则.

规则1 如果结点D处的下界大于当前上界,则将结点D删除.

将$J_i\in\mathrm{US}_D$放在序列S_D后得到新结点D_i.

规则2 如果$\mathrm{ft}+p_i\leqslant T,\mathrm{fn}+1\leqslant K$,但是$p_i<p_{[n_D]}$,则将结点$D_i$删除.

规则3 如果$\mathrm{ft}+p_i\leqslant T,\mathrm{fn}+1\leqslant K$,但是$\mathrm{fn}=\mathrm{sn}$,则将结点$D_i$删除.

规则4 如果$\mathrm{ft}+p_i>T,\mathrm{fn}+1\leqslant K$,但是$\mathrm{ft}+p_{\min}\leqslant T$,其中$p_{\min}=\min\limits_{J_j\in US_D}\{p_j\}$,则将结点$D_i$删除.

规则2~规则4分别根据最优解的性质2~性质4得到.

4.1.3 算法步骤

step 1: 将SPT调度的目标函数值作为初始上界.

step 2: 初始化根结点D,设置$\mathrm{ft}=0,\mathrm{fn}=0,\mathrm{sn}=0,S_D=\Phi,\mathrm{US}_D=\{J_1,J_2,\ldots,J_n\}$.

step 3: 如果搜索树已空,则停止搜索,否则选择一个未搜索结点D.

step 4: 如果D是叶子结点,即$\mathrm{US}_D=\Phi$,则得到一个完整的调度,根据式(11)计算目标函数值. 如果目标函数值小于当前上界,则用其更新当前上界,转入step 3.

step 5: 将$J_i\in\mathrm{US}_D$放在S_D后,得到新结点D_i.

step 5.1: 若$\mathrm{fn}+1\leqslant K$成立,则判断D_i是否满足规则2~规则4. 若满足其中任意一条,则将D_i删除;否则,根据式(12)计算D_i处下界. 若下界满足规则1,则将D_i删除;否则,将D_i放在搜索树上. 令$\mathrm{fn}=\mathrm{fn}+1,\mathrm{ft}=\mathrm{ft}+p_i$,转入step 3.

step 5.2: 若对于任意$J_i\in\mathrm{US}_D,\mathrm{fn}+1>K$或$\mathrm{ft}+p_i>T$成立,则将$J_i$放入下一个批次中,得到新结点$D_i$,根据式(12)计算$D_i$处下界. 若满足规则1,则将$D_i$删除;否则,将$D_i$放在搜索树上,令$\mathrm{sn}=\mathrm{fn},\mathrm{ft}=p_i,\mathrm{fn}=1$,转入step 3.

4.2 遗传算法

由于$1|\mathrm{nr},\mathrm{pm}-\mathrm{mn}|\sum C_i$是强NP-hard问题,为求解中、大规模问题,本文采用遗传算法与动态遗传算法求解.

4.2.1 普通遗传算法

遗传算法的主要设置如下.

1) 染色体结构:每个可行调度被编码成一个含有n个整数的染色体,每个整数代表一个工件,在染色体中,值i在位置j表示J_i位于调度中第j个位置.

2) 初始种群:为提高搜索效率,随机生成500个解,择优选取200个作为初始种群,种群规模$n_{\mathrm{pop}}=200$.

3) 适应度函数:个体π_s的适应度函数定义为

$$\mathrm{fitness}(\pi_s)=\max_{t=1}^{n_{\mathrm{pop}}}(z(\pi_t))-z(\pi_s)+\varepsilon, \tag{13}$$

这里ε是一个正常数.

4) 选择策略:轮盘赌方法,π_s的选择概率是

$$P_{\pi_s}=\mathrm{fitness}(\pi_s)\Big/\sum_{t=1}^{n_{\mathrm{pop}}}\mathrm{fitness}(\pi_t). \tag{14}$$

5) 遗传算子:交叉算子,采用两点交叉,交叉概率$P_c=0.9$;变异算子,随机选取两点互换位置,变异概率$P_m=0.05$.

6) 停止准则:最大迭代次数Maxgen = 250.

4.2.2 动态遗传算法

尽管遗传算法有诸多优点,但早熟问题是不可忽视的现象,为缓解这一问题,提出动态遗传算法,对变异算子采用动态变异方法.

1) 计算当前种群中与所选变异染色体相同的染色体个数n_p.

2) 若$n_p<n_{\mathrm{pop}}/5$,则对所选染色体执行$n/3$次变异,否则转入3).

3) 若$n_p<n_{\mathrm{pop}}/3$,则对所选染色体执行$n/2$次变异,否则转入4).

4) 对所选染色体执行n次变异.

5 仿真实验与分析

本节通过仿真实验对上述算法进行评估. 所有仿真均在Intel(R) Core(TM) i7-5500U@2.4 GHz 8 GB RAM计算机上实施. 二值整数规划模型利用Lingo软件求解,分支定界算法、遗传算法和动态遗传算法利用Matlab软件实现. 由于此类调度问题属首次涉及,本节所有测试实例均为随机生成,参数取值借鉴文献[15-16].

1) 工件个数:小规模问题{6, 8, 10, 15},中等规模问题{20, 25, 30, 35, 100},大规模问题{1 000, 2 000}.

2) 加工时间:由整数均匀分布[5, 15]随机生成.

3) 机器的工作时间周期

$$T=\max\left\{\left\lceil a\sum_{i=1}^{n}p_i\right\rceil,\ \max_{1\leqslant i\leqslant n}p_i\right\},\ a=1/3,1/5.$$

4) 机器的维护时长

$$t=\lceil bT\rceil,\ b=1/5,1/10.$$

5) 每个加工周期T内允许加工的最大工件个数

$$K=\lceil cn\rceil,\ c=1/3,1/5.$$

根据上述取值规则,随机生成88个问题实例. 每个问题用不同方法运行5次,表1和表2给出了每种方法的平均运行时间和最优解. 可以看出,二值规划模型和分枝定界算法求解的问题规模相对有限,而遗传算法和动态遗传算法能够在合理的计算时间内求解多达2 000个工件的大型问题. 对同一问题的计算时间,在工件个数不超过20时,分枝定界算法所需的计算时间最少,随着问题规模扩大,分枝定界算法和二值规划模型所需的计算时间快速增长,尤其是二值规划模型,属于爆炸性枚举,对任何规模的问题,其计算时间都比较长,且当工件个数大于10时,已无法在合理的时间内给出问题的最优解. 遗传算法和动态遗传算法随着问题规模的增长运行时间稳步增长. 在动态遗传算法中,染色体的变异次数由n_p决定,因此在求解同一问题时,动态遗传算法所需的时间一般比普通遗传算法稍有增加,但得到的结果较为理想. 动态遗传算法增加了种群的多样性,进而增大了生成更好个体的可能性. 利用误差百分比可评估各种算法的性能,下式给出了计算误差百分比的公式:

$$\text{误差比}=\frac{\text{最好解}-\text{最优解}}{\text{最优解}}\times 100\%. \tag{15}$$

从表1和表2可以看出,遗传算法和动态遗传算法的性能相当可靠,针对中小规模问题,误差百分比不超过0.02 %. 分枝定界算法对于较大的T或K,所需时间较少,这是因为T与K越大,同一批次内可加工的工件个数越多,越有利于剪枝. T与K对遗传算法和动态遗传算法在运行时间上有轻微影响,但是对目标函数值有较大的影响. 在相同条件下,T或K越大,目标函数值越小,这是因为T或K越大,相应调度所含批次越少,所需的维护时间也会变少. 维护时间t对算法的运行时间影响不大. 从图6和图7可以看出算法的收敛性,遗传算法和动态遗传算法对大规模问题($n=2\,000, T=4\,008, t=802, K=667$)依然能够求出近似最优解. 由图8和图9可知,对同一问题($n=100, T=203, t=21, K=34$),动态遗传算法的适应度范围相对于遗传算法更广,说明在陷入局部最优时,动态遗传算法为种群引入了更多新的个体,保持了种群的多样性,得到的结果相对更优.

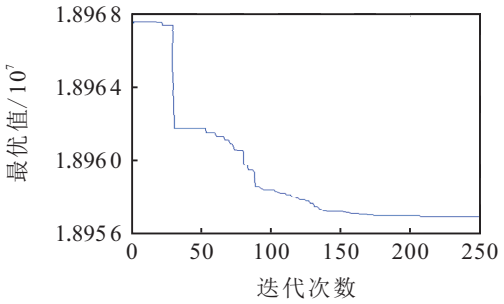

图 6　遗传算法

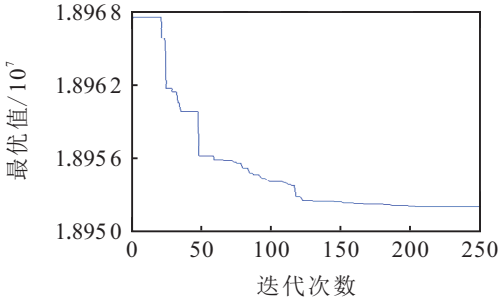

图 7　动态遗传算法

表1 实验结果($K=\lceil n/3\rceil$)

n	T	t	Lingo		分支定界算法		遗传算法			动态遗传算法		
			最优解	时间/s	最优解	时间/s	最优解	时间/s	平均误差/%	最优解	时间/s	平均误差/%
6	$\lceil \mathrm{sum}/3\rceil$	$\lceil T/5\rceil$	266	3.021	266	0.126	266	18.050	0	266	22.253	0
		$\lceil T/10\rceil$	252	6.035	252	0.121	252	17.931	0	252	21.786	0
	$\lceil \mathrm{sum}/5\rceil$	$\lceil T/5\rceil$	251	5.102	251	0.120	251	18.506	0	251	21.125	0
		$\lceil T/10\rceil$	241	3.918	241	0.120	241	18.071	0	241	23.193	0
8	$\lceil \mathrm{sum}/3\rceil$	$\lceil T/5\rceil$	502	64.567	502	0.138	502	18.929	0	502	24.078	0
		$\lceil T/10\rceil$	475	84.434	475	0.136	475	18.940	0	475	23.324	0
	$\lceil \mathrm{sum}/5\rceil$	$\lceil T/5\rceil$	607	75.591	607	0.138	607	19.916	0	607	22.282	0
		$\lceil T/10\rceil$	565	81.290	565	0.141	565	19.941	0	565	24.314	0
10	$\lceil \mathrm{sum}/3\rceil$	$\lceil T/5\rceil$	548	3 752.622	548	0.122	548	18.760	0	548	22.478	0
		$\lceil T/10\rceil$	521	4 180.775	521	0.107	521	18.410	0	521	23.824	0
	$\lceil \mathrm{sum}/5\rceil$	$\lceil T/5\rceil$	627	2 967.908	627	0.133	627	19.091	0	627	24.448	0
		$\lceil T/10\rceil$	585	3 567.543	585	0.120	585	19.117	0	585	25.607	0
15	$\lceil \mathrm{sum}/3\rceil$	$\lceil T/5\rceil$	–	–	1 500	0.307	1 500	23.503	0	1 500	33.160	0
		$\lceil T/10\rceil$	–	–	1 410	0.302	1 410	23.132	0	1 410	33.631	0
	$\lceil \mathrm{sum}/5\rceil$	$\lceil T/5\rceil$	–	–	1 466	0.555	1 466	24.244	0	1 466	36.843	0
		$\lceil T/10\rceil$	–	–	1 382	0.417	1 382	23.571	0	1 382	34.242	0
20	$\lceil \mathrm{sum}/3\rceil$	$\lceil T/5\rceil$	–	–	2 012	4.691	2 012	27.415	0	2 012	41.707	0
		$\lceil T/10\rceil$	–	–	1 898	4.893	1 898	27.300	0	1 898	37.338	0
	$\lceil \mathrm{sum}/5\rceil$	$\lceil T/5\rceil$	–	–	1 913	10.854	1 913	28.064	0	1 913	43.148	0
		$\lceil T/10\rceil$	–	–	1 781	7.762	1 781	28.042	0	1 781	41.432	0
25	$\lceil \mathrm{sum}/3\rceil$	$\lceil T/5\rceil$	–	–	3 422	43.651	3 422	28.028	0	3 422	50.667	0
		$\lceil T/10\rceil$	–	–	3 214	39.995	3 214	28.220	0	3 214	50.511	0
	$\lceil \mathrm{sum}/5\rceil$	$\lceil T/5\rceil$	–	–	3 282	46.348	3 282	28.116	0	3 282	48.984	0
		$\lceil T/10\rceil$	–	–	3 081	53.053	3 081	28.178	0	3 081	49.615	0
30	$\lceil \mathrm{sum}/3\rceil$	$\lceil T/5\rceil$	–	–	5 776	3 936.702	5 776	31.954	0	5 776	47.204	0
		$\lceil T/10\rceil$	–	–	5 445	4 287.131	5 446	32.069	0.02	5 446	55.110	0.02
	$\lceil \mathrm{sum}/5\rceil$	$\lceil T/5\rceil$	–	–	5 362	4 537.756	5 362	32.619	0	5 362	48.325	0
		$\lceil T/10\rceil$	–	–	5 004	4 908.874	5 005	32.919	0.02	5 004	51.046	0
35	$\lceil \mathrm{sum}/3\rceil$	$\lceil T/5\rceil$	–	–	–	–	6 833	29.573	–	6 833	53.325	–
		$\lceil T/10\rceil$	–	–	–	–	6 428	29.607	–	6 428	51.341	–
	$\lceil \mathrm{sum}/5\rceil$	$\lceil T/5\rceil$	–	–	–	–	6 381	29.841	–	6 381	51.961	–
		$\lceil T/10\rceil$	–	–	–	–	5 972	30.070	–	5 973	55.836	–
100	$\lceil \mathrm{sum}/3\rceil$	$\lceil T/5\rceil$	–	–	–	–	54 447	70.959	–	54 432	90.276	–
		$\lceil T/10\rceil$	–	–	–	–	51 067	71.579	–	51 118	89.457	–
	$\lceil \mathrm{sum}/5\rceil$	$\lceil T/5\rceil$	–	–	–	–	48 714	70.874	–	48 575	88.379	–
		$\lceil T/10\rceil$	–	–	–	–	45 551	72.179	–	45 416	89.573	–
1 000	$\lceil \mathrm{sum}/3\rceil$	$\lceil T/5\rceil$	–	–	–	–	5 457 789	934.666	–	5 437 328	1 911.574	–
		$\lceil T/10\rceil$	–	–	–	–	5 077 950	1 292.431	–	5 074 001	1 815.823	–
	$\lceil \mathrm{sum}/5\rceil$	$\lceil T/5\rceil$	–	–	–	–	4 755 908	1 071.002	–	4 755 906	1 655.868	–
		$\lceil T/10\rceil$	–	–	–	–	4 438 708	1 381.433	–	4 438 710	1 702.359	–
2 000	$\lceil \mathrm{sum}/3\rceil$	$\lceil T/5\rceil$	–	–	–	–	21 920 051	2 825.535	–	21 899 860	5 721.150	–
		$\lceil T/10\rceil$	–	–	–	–	20 517 836	2 779.562	–	20 517 789	3 117.713	–
	$\lceil \mathrm{sum}/5\rceil$	$\lceil T/5\rceil$	–	–	–	–	18 954 631	4 065.803	–	18 951 958	2 467.866	–
		$\lceil T/10\rceil$	–	–	–	–	17 691 024	2 431.974	–	17 693 701	2 492.294	–

表 2　实验结果($K = \lceil n/5 \rceil$)

n	T	t	Lingo		分支定界算法		遗传算法			动态遗传算法		
			最优解	时间/s	最优解	时间/s	最优解	时间/s	平均误差/%	最优解	时间/s	平均误差/%
6	$\lceil \mathrm{sum}/3 \rceil$	$\lceil T/5 \rceil$	266	4.038	266	0.126	266	18.050	0	266	22.253	0
		$\lceil T/10 \rceil$	252	4.065	252	0.121	252	17.931	0	252	21.786	0
	$\lceil \mathrm{sum}/5 \rceil$	$\lceil T/5 \rceil$	251	6.402	251	0.120	251	18.506	0	251	25.125	0
		$\lceil T/10 \rceil$	241	7.318	241	0.120	241	18.071	0	241	23.193	0
8	$\lceil \mathrm{sum}/3 \rceil$	$\lceil T/5 \rceil$	617	76.532	617	0.152	617	18.839	0	617	23.101	0
		$\lceil T/10 \rceil$	581	65.456	581	0.148	581	18.868	0	581	24.031	0
	$\lceil \mathrm{sum}/5 \rceil$	$\lceil T/5 \rceil$	607	74.552	607	0.136	607	19.612	0	607	24.258	0
		$\lceil T/10 \rceil$	565	87.003	565	0.131	565	19.689	0	565	23.468	0
10	$\lceil \mathrm{sum}/3 \rceil$	$\lceil T/5 \rceil$	934	2 988.656	934	0.122	934	19.428	0	934	23.872	0
		$\lceil T/10 \rceil$	874	3 180.742	874	0.118	874	18.948	0	874	25.021	0
	$\lceil \mathrm{sum}/5 \rceil$	$\lceil T/5 \rceil$	627	4 267.990	627	0.193	627	19.191	0	627	26.142	0
		$\lceil T/10 \rceil$	585	3 567.567	585	0.183	585	19.257	0	585	25.290	0
15	$\lceil \mathrm{sum}/3 \rceil$	$\lceil T/5 \rceil$	–	–	2 407	0.456	2 407	23.313	0	2 407	33.790	0
		$\lceil T/10 \rceil$	–	–	2 227	0.477	2 227	23.793	0	2 227	33.308	0
	$\lceil \mathrm{sum}/5 \rceil$	$\lceil T/5 \rceil$	–	–	1 585	1.146	1 585	24.419	0	1 585	34.229	0
		$\lceil T/10 \rceil$	–	–	1 492	0.943	1 492	24.308	0	1 492	34.901	0
20	$\lceil \mathrm{sum}/3 \rceil$	$\lceil T/5 \rceil$	–	–	3 361	17.201	3 361	27.568	0	3 361	37.420	0
		$\lceil T/10 \rceil$	–	–	3 121	18.681	3 121	27.593	0	3 121	37.506	0
	$\lceil \mathrm{sum}/5 \rceil$	$\lceil T/5 \rceil$	–	–	2 201	44.096	2 201	28.506	0	2 201	42.896	0
		$\lceil T/10 \rceil$	–	–	2 041	39.597	2 041	27.556	0	2 041	38.037	0
25	$\lceil \mathrm{sum}/3 \rceil$	$\lceil T/5 \rceil$	–	–	5 903	102.905	5 903	28.509	0	5 903	50.849	0
		$\lceil T/10 \rceil$	–	–	5 453	94.072	5 453	28.154	0	5 453	50.951	0
	$\lceil \mathrm{sum}/5 \rceil$	$\lceil T/5 \rceil$	–	–	3 804	339.744	3 804	28.866	0	3 804	51.299	0
		$\lceil T/10 \rceil$	–	–	3 553	361.183	3 553	28.926	0	3 553	53.875	0
30	$\lceil \mathrm{sum}/3 \rceil$	$\lceil T/5 \rceil$	–	–	9 332	3 567.843	9 332	31.150	0	9 333	55.924	0.01
		$\lceil T/10 \rceil$	–	–	8 672	5 291.882	8 672	32.187	0	8 672	54.492	0
	$\lceil \mathrm{sum}/5 \rceil$	$\lceil T/5 \rceil$	–	–	6 033	5 865.901	6 033	32.339	0	6 033	53.220	0
		$\lceil T/10 \rceil$	–	–	5 614	6 183.003	5 614	32.215	0	5 614	55.405	0
35	$\lceil \mathrm{sum}/3 \rceil$	$\lceil T/5 \rceil$	–	–	–	–	11 313	29.556	–	11 313	54.834	–
		$\lceil T/10 \rceil$	–	–	–	–	10 474	29.523	–	10 473	58.596	–
	$\lceil \mathrm{sum}/5 \rceil$	$\lceil T/5 \rceil$	–	–	–	–	7 332	29.942	–	7 332	56.842	–
		$\lceil T/10 \rceil$	–	–	–	–	6 856	30.415	–	6 856	59.246	–
100	$\lceil \mathrm{sum}/3 \rceil$	$\lceil T/5 \rceil$	–	–	–	–	90 061	71.393	–	90 061	87.459	–
		$\lceil T/10 \rceil$	–	–	–	–	83 303	71.420	–	83 300	87.323	–
	$\lceil \mathrm{sum}/5 \rceil$	$\lceil T/5 \rceil$	–	–	–	–	57 913	71.940	–	57 913	87.445	–
		$\lceil T/10 \rceil$	–	–	–	–	53 900	73.544	–	53 898	93.523	–
1 000	$\lceil \mathrm{sum}/3 \rceil$	$\lceil T/5 \rceil$	–	–	–	–	8 856 998	852.791	–	8 852 911	1 484.968	–
		$\lceil T/10 \rceil$	–	–	–	–	8 198 859	1 213.303	–	8 197 226	1 483.698	–
	$\lceil \mathrm{sum}/5 \rceil$	$\lceil T/5 \rceil$	–	–	–	–	5 737 161	1 280.252	–	5 734 096	1 550.913	–
		$\lceil T/10 \rceil$	–	–	–	–	5 332 261	1 296.861	–	5 331 967	1 755.345	–
2 000	$\lceil \mathrm{sum}/3 \rceil$	$\lceil T/5 \rceil$	–	–	–	–	35 632 536	2 138.115	–	35 641 021	5 840.161	–
		$\lceil T/10 \rceil$	–	–	–	–	32 868 002	2 136.469	–	32 860 320	2 369.889	–
	$\lceil \mathrm{sum}/5 \rceil$	$\lceil T/5 \rceil$	–	–	–	–	23 035 280	2 328.164	–	23 034 716	2 438.867	–
		$\lceil T/10 \rceil$	–	–	–	–	21 459 138	2 905.581	–	21 436 810	2 432.110	–

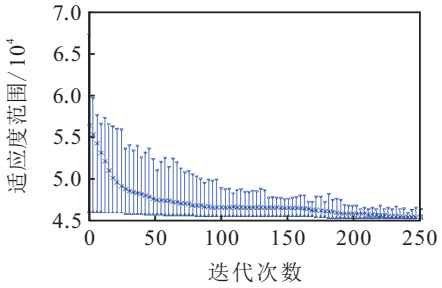

图 8 遗传算法

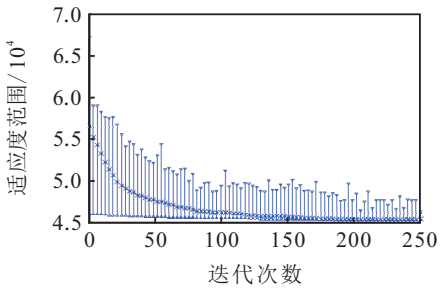

图 9 动态遗传算法

6 结 论

本文研究了两种维护约束下目标函数为完成时间之和的单机调度问题,提出了二值整数规划模型、分支定界算法、遗传算法和动态遗传算法. 仿真结果表明,在合理的计算时间内,二值规划模型最多可求解10个工件的调度问题,分支定界算法最多可求解30个工件的调度问题,而遗传算法和动态遗传算法能够解决多达2 000个工件的大型问题. 同时,计算结果也表明了各种算法的有效性. 在今后的研究中,可考虑其他性能指标或将问题扩展到更复杂的机器环境上,如并行机、流水车间等.

参考文献(References)

[1] Batun S, Azizoğlu M. Single machine scheduling with preventive maintenances[J]. International Journal of Production Research, 2009, 47(7): 1753-1771.

[2] Lee C Y, Liman S D. Single machine flow-time scheduling with scheduled maintenance[J]. Acta Informatica, 1992, 29(4): 375-382.

[3] Qi X T, Chen T, Tu F S. Scheduling the maintenance on a single machine[J]. Journal of the Operational Research Society, 1999, 50(10): 1071-1078.

[4] Qi X T. A note on worst-case performance of heuristics for maintenance scheduling problems[J]. Discrete Applied Mathematics, 2007, 155(3): 416-422.

[5] Wu H P, Huang M, Wang X W. Single-machine scheduling problem with multi-RMAs[J]. Control and Decision, 2014, 29(12): 2253-2258.

[6] Gan J, Zeng J C. Integrated model of single-machine scheduling and maintenance decision for degrading state systems[J]. Control and Decision, 2016, 31(3): 513-520.

[7] Liao C J, Chen W J. Single-machine scheduling with periodic maintenance and nonresumable jobs[J]. Computers & Operations Research, 2003, 30(9): 1335-1347.

[8] Chen W J. Minimizing number of tardy jobs on a single machine subject to periodic maintenance[J]. Omega, 2009, 37(3): 591-599.

[9] Ji M, He Y, Cheng T C E. Single-machine scheduling with periodic maintenance to minimize makespan[J]. Computers & Operations Research, 2007, 34(6): 1764-1770.

[10] Ruan M Z, Li D W, Zhang Z H. Analysis of optimal combined maintenance strategy based on performance deterioration[J]. Systems Engineering — Theory & Practice, 2018, 38(4): 1035-1042.

[11] Perez-Gonzalez P, Framinan J M. Single machine scheduling with periodic machine availability[J]. Computers & Industrial Engineering, 2018, 123(9): 180-188.

[12] Shen J Y, Zhu K. An uncertain single machine scheduling problem with periodic maintenance[J]. Knowledge-based Systems, 2018, 144(5): 32-41.

[13] Low C, Ji M, Hsu C J, et al. Minimizing the makespan in a single machine scheduling problems with flexible and periodic maintenance[J]. Applied Mathematical Modelling, 2010, 34(2): 334-342.

[14] Low C, Hsu C J, Su C T. A modified particle swarm optimization algorithm for a single-machine scheduling problem with periodic maintenance[J]. Expert Systems with Applications, 2010, 37(9): 6429-6434.

[15] Hsu C J, Low C, Su C T. A single-machine scheduling problem with maintenance activities to minimize makespan[J]. Applied Mathematics and Computation, 2010, 215(11): 3929-3935.

[16] Zade A E, Fakhrzad M B. A dynamic genetic algorithm for solving a single machine scheduling problem with periodic maintenance[J]. Isrn Industrial Engineering, 2013, 2013: 1-11.

[17] Shahriari M, Shoja N, Zade A E, et al. JIT single machine scheduling problem with periodic preventive maintenance[J]. Journal of Industrial Engineering International, 2016, 12(3): 299-310.

[18] Gan J, Zeng J C. Single-machine integrated scheduling model considering maintenance activities[J]. Computer Integrated Manufacturing Systems, 2014, 20(5): 1099-1105.

[19] Wang G N, Yu B H, Pan E S, et al. Single-machine group scheduling policy with preventive maintenance planning considering learning and forgetting effects[J]. Journal of Shanghai Jiaotong University, 2013, 47(5): 723-727.

[20] Graham R L, Lawler E L, Lenstra J K, et al. Optimization and approximation in deterministic sequencing and scheduling: A survey[J]. Annals of Discrete Mathematics, 1979, 5(1): 287-326.

作者简介

吴慧 (1980−), 女, 讲师, 博士生, 从事机器调度、鲁棒优化等研究, E-mail: wuhui0925@163.com;

王冰 (1966−), 女, 教授, 博士生导师, 从事生产调度、优化方法等研究, E-mail: susanbwang@shu.edu.cn.

(责任编辑: 齐 霁)