DOI: 10.7641/CTA.2015.40616

可重入混合流水车间调度的拉格朗日松弛算法

周炳海†, 钟臻怡

(同济大学 机械与能源工程学院, 上海 201804)

摘要: 为了有效提升多重入车间的生产效率, 考虑了实际生产中检查和修复过程对于逐层制造的可重入生产系统的重要性, 提出了基于拉格朗日松弛算法的可重入混合流水车间的调度方法. 首先进行了问题域的描述, 并在此基础上以最小化加权完成时间为调度目标, 建立数学规划模型. 针对该调度问题提出了基于松弛机器能力约束的拉格朗日松弛算法, 使松弛问题分解成工件级子问题, 并使用动态规划方法建立递归公式, 求解工件级子问题. 随后, 使用次梯度算法求解拉格朗日对偶问题. 最后, 对各种不同问题规模进行了仿真实验, 结果表明, 所提出的调度算法能够在合理的时间内获得满意的近优解.

关键词: 可重入混合流水车间; 调度; 拉格朗日松弛; 动态规划

中图分类号: TP391 **文献标识码**: A

Lagrangian relaxation algorithm for scheduling problems of reentrant hybrid flow shops

ZHOU Bing-hai†, ZHONG Zhen-yi

(School of Mechanical Engineering, Tongji University, Shanghai 201804, China)

Abstract: To effectively enhance the productivity of multi-reentrant workshops, we consider the importance of inspection and repair processes to reentrant production systems where products are manufactured layer by layer; and then, propose a scheduling method of reentrant hybrid flow shops based on Lagrangian relaxation algorithm. Firstly, a scheduling problem domain of reentrant hybrid flow shops is described and then a mathematical programming model is built with an objective of minimizing the total weighted completion time. A Lagrangian relaxation algorithm with relaxing machine capacity constraints is developed, which decomposes the relaxed problem into job-level sub-problems. Each sub-problem is solved by a dynamic programming method with dynamic programming recursive formulas. After that, the sub-gradient algorithm is used to solve the Lagrangian dual problem. Finally, simulation experiments in different problem scales are carried out to analyze the proposed algorithm. Results indicate that the proposed algorithm can obtain satisfactory near-optimal solutions within a reasonable time period.

Key words: reentrant hybrid flow shop; scheduling; Lagrangian relaxation; dynamic programming

1 引言(Introduction)

可重入车间作为一种新的制造车间类型正引起越来越多的关注. 可重入车间的基本特性是一个工件多次进入某些工作站, 这种特性在半导体制造过程中尤为突出. 在半导体晶圆制造过程中, 晶圆需要一层一层被加工, 有多少个层次就需要重入多少次, 并且每次重入按照相同的顺序依次经过各个工作站完成一系列加工步骤. 此制造过程可归结为可重入混合流水车间(reentrant hybrid flow shop, RHFS)问题, 它表示所有的工件在机器上具有相同的加工路径, 并且按照相同的次序来回多次, 并且至少有一个工作站有超过一台的机器.

近年来, RHFS调度问题引起了许多国内外学者, 尤其是国外学者的关注. Choi等[1]研究了多阶段的RHFS调度问题, 并使用启发式方法解决了具有多种批量的订单问题, 目标函数为最小化多个订单总完成时间. Jiang等[2]对于RHFS调度问题, 使用了拉格朗日松弛(Lagrangian relaxation, LR)算法, 但简化了求解问题, 并没有给出求下界的具体方法. Dugardin等[3]针对两阶段的RHFS多目标问题提出了L–NSGA和

收稿日期: 2014−07−01; 录用日期: 2015−04−24.

†通信作者. E-mail: bhzhou@tongji.edu.cn; Tel.: +86 13564164374.

国家自然科学基金项目(61273035, 71471135), 国家高技术研究发展计划(“863”计划)项目(2009AA043000)资助.

Supported by National Natural Science Foundation of China (61273035, 71471135) and National High Technology Research and Development Program of China (“863” Program) (2009AA043000).

SPEA–II算法. Dugardin等[4]以最大化瓶颈利用率和最小化最大完成时间为目标, 提出了一种新的多目标遗传算法L–NSGA来求解RHFS调度问题. Cho等[5]针对RHFS调度问题, 提出了基于局部搜索的帕累托遗传算法. Hekmatfar等[6]针对两阶段的RHFS调度问题, 提出了以最小化makespan为目标的混合遗传算法.

虽然上述文献的研究成果能够为RHFS调度问题提供良好的借鉴, 但仍具有一定的局限性. 首先, 上述文献仅仅考虑了晶圆的加工和检查, 而没有考虑到实际生产过程中, 晶圆的某一个层次制造所产生的错误会被新的一层覆盖, 不能被纠正的问题[7]. 本文研究的RHFS调度问题, 从晶圆在加工完一层之后, 考虑及时进行晶圆的检查和修复, 对于部分修复不成功的晶圆还要继续重入, 再进行检查和修复, 随后才能开始对下一层进行加工.

其次, 上述文献多使用智能优化算法, 且算法中没有体现出可重入调度问题中不同工件的不同层次之间对于机器的耦合这一重要特性. 而类似拉格朗日松弛(Lagrangian relaxation, LR)这类能够在合理的时间内找到实际规模问题的可量化指标的近优解的算法更受青睐. 例如, 在混合流水车间(hybrid flow shop, HFS)调度问题以及炼钢等问题中LR算法就已广泛应用[8–11]. 但有关这些文献, 并不考虑可重入调度问题. 本文提出了基于松弛机器能力约束的LR算法, 不仅能刻画多重入调度问题的重要特性, 并且通过松弛机器能力约束这一耦合约束, 能将原本复杂的RHFS问题分解成较容易求解的工件级子问题.

2 数学建模(Mathematical modeling)

2.1 问题描述(Problem statement)

在本文所研究的RHFS调度问题中, 由两个主要的阶段组成, 共有N个工件, 不同的工件具有不同的层次数L_i(重入次数), 示意图如图1所示.

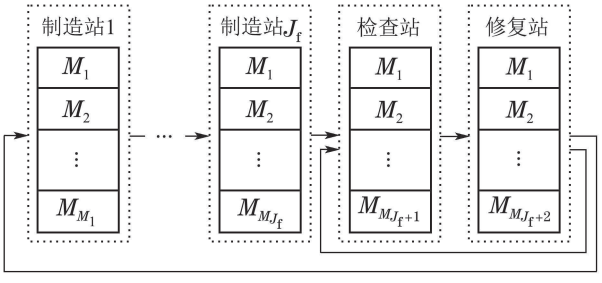

图 1 带有重入系统的RHFS问题

Fig. 1 The problem of RHFS with a reentrant system

第1阶段为制造阶段, 其中有$J_{\rm f}$个工作站, 并且每个工作站j由一组相同的平行机M_j组成. 第2阶段中, 由检查和修复工作站组成一个小的可重入系统, 每个工作站j由一组相同的平行机M_j组成, 每个工件i的每一层l都需要重入这一阶段L_i'次, 不同的工件都有不同的重入次数. 在本问题中, 每个工件每个层次按照相同的顺序依次经过各个工作站进行加工, 每个工件总共被加工$(J_{\rm f}+2L_i')L_i$次.

2.2 数学模型(Mathematical formulation)

基于上述问题描述, 建立所研究的RHFS调度问题的数学模型如下:

1) 模型假设. 机器在任何时候都可用, 并且没有故障和维护延迟; 工件加工没有优先级, 即一旦工件的一个工序开始加工, 则此工序不允许中断; 各个工作站之间没有有限缓冲; 各个工作站之间没有搬运时间; 所有机器都是平行机且能力和加工速率均相同; 一台机器在同一时刻至多只能加工一个工件, 且一个工件在任意时刻至多只能在一台机器上加工.

2) 参数. N为工件数; L_i为工件i的层次数(重入次数); L_i'为工件i的每一层重入检查修复阶段的次数; $J_{\rm f}$为制造阶段的工作站数; J_i为工件i的每一层所经过的工作站数, 每个工作站由一组平行机组成, 且$J_i=J_{\rm f}+2L_i'$; M为总机器数; K为时间水平; w_i为工件i的权重; $p_{i,j}$为工件i在工作站j的加工时间; M_j为工作站j的一组平行机集合, 且

$$M_1\cup M_2\cup\cdots\cup M_{J_{\rm f}+2}=M,$$
$$M_j=M_{j-2},\ \forall J_{\rm f}+2<j\leqslant J_i,$$
$$M_{j1}\cap M_{j2}=\phi,\ \forall 1\leqslant j_1,\ j_2\leqslant J_{\rm f}+2.$$

3) 决策变量. $C_{i,l,j}$为工件i的第l层在工作站j的完成时间; $m_{i,l,j}$为工件i的第l层在工作站j所分配的机器.

根据上述问题描述、模型假设、参数以及决策变量, 对所研究的RHFS调度问题建模如下:

$$\min\sum_{i=1}^{N}w_iC_{i,L_i,J_i},\tag{1}$$

$$\text{s.t. } C_{i,l,j}\geqslant p_{i,l,j},\tag{2}$$
$$i=1,\cdots,N,\ l=1,\cdots,L_i,\ j=1,\cdots,J_i,$$

$$C_{i,l,j-1}\leqslant C_{i,l,j}-p_{i,l,j},\tag{3}$$
$$i=1,\cdots,N,\ l=1,\cdots,L_i,\ j=2,\cdots,J_i,$$

$$C_{i,l-1,J_i}+p_{i,l,1}\leqslant C_{i,l,1},\tag{4}$$
$$i=1,\cdots,N,\ l=2,\cdots,L_i,$$

$$\sum_{(i,l,j)\in O_u}(\varphi(k-C_{i,l,j}+p_{i,l,j}-1)-\varphi(k-C_{i,l,j}-1))\leqslant 1,\tag{5}$$
$$k=1,\cdots,K,\ u=1,\cdots,M.$$

上述模型中, 目标函数(1)是最小化总加权完成时间; 约束(2)表明了加工开始时间要求; 约束(3)表明了工件加工顺序要求, 直到某个工件的某一层的前一工序完成后, 后一工序才能开始加工; 约束(4)表明了工件加工顺序要求, 直到某个工件的某一层的最后一道

工序完成后, 后一层的重入工序才能开始加工; 约束(5)定义了机器能力要求, 表明一台机器在同一时刻至多只能加工一个工件, 其中$\varphi(k)$ 表示一个函数, 并且满足

$$\varphi(k)=\begin{cases}1,\ k\geqslant 0,\\ 0,\ k<0,\end{cases}$$

$$O_u=\{(i,l,j)|m_{i,l,j}=u\}.$$

3 拉格朗日松弛算法(Lagrangian relaxation algorithm)

3.1 松弛机器能力约束(Relaxing machine capacity constraints)

从数学模型中可知, 只有约束(5)机器能力约束耦合了不同的工件, 为此, 引入拉格朗日乘子$\lambda_{k,u}(k=1,\cdots,K,\ u=1,\cdots,M)$, 把约束(5)松弛到目标函数中, 形成如下松弛问题:

$$\begin{aligned}L(\lambda)=\min\{&\sum_{i=1}^{N}w_iC_{i,L_i,J_i}+\sum_{u=1}^{M}\sum_{k=1}^{K}\lambda_{k,u}\times\\&[\sum_{(i,j,l)\in O_u}(\varphi(k-C_{i,l,j}+p_{i,l,j}-1)-\\&\varphi(k-C_{i,l,j}-1))-1]\},\end{aligned}\tag{6}$$

$$\lambda_{k,u}\geqslant 0,\ k=1,\cdots,K,\ u=1,\cdots,M,\tag{7}$$

s.t. 式$(2)-(4)(7)$.

由于如下两个关系式成立:

$$\begin{aligned}&\sum_{u=1}^{M}\sum_{(i,j,l)\in O_u}(\varphi(k-C_{i,l,j}+p_{i,l,j}-1)-\\&\varphi(k-C_{i,l,j}-1))=\\&\sum_{i=1}^{N}\sum_{l=1}^{L_i}\sum_{j=1}^{J_i}(\varphi(k-C_{i,l,j}+p_{i,l,j}-1)-\\&\varphi(k-C_{i,l,j}-1)),\\&\sum_{k=1}^{K}\lambda_{k,u}\times(\varphi(k-C_{i,l,j}+p_{i,l,j}-1)-\\&\varphi(k-C_{i,l,j}-1))=\sum_{k=C_{i,l,j}-p_{i,l,j}+1}^{C_{i,l,j}}\lambda_{k,u}.\end{aligned}$$

因此, 松弛问题的目标函数可以写成如下形式:

$$\begin{aligned}&\sum_{i=1}^{N}\min\{w_iC_{i,L_i,J_i}+\\&\sum_{l=1}^{L_i}\sum_{j=1}^{J_i}\sum_{k=C_{i,l,j}-p_{i,l,j}+1}^{C_{i,l,j}}\lambda_{k,m_{i,l,j}}\}-\sum_{u=1}^{M}\sum_{k=1}^{K}\lambda_{k,u}.\end{aligned}\tag{8}$$

那么松弛问题就可以分解成独立的工件级子问题, 工件级子问题可以表示为

$$\begin{aligned}L_i(\lambda)=\min\{&w_iC_{i,L_i,J_i}+\\&\sum_{l=1}^{L_i}\sum_{j=1}^{J_i}\sum_{k=C_{i,l,j}-p_{i,l,j}+1}^{C_{i,l,j}}\lambda_{k,u}\},\end{aligned}\tag{9}$$

s.t. 式$(2)-(4)(7)$.

拉格朗日对偶问题为

$$\begin{aligned}\max_{\{\lambda_{k,u}\}}\{&\sum_{i=1}^{N}\min\{w_iC_{i,L_i,J_i}+\\&\sum_{l=1}^{L_i}\sum_{j=1}^{J_i}\sum_{k=C_{i,l,j}-p_{i,l,j}+1}^{C_{i,l,j}}\lambda_{k,m_{i,l,j}}\}-\\&\sum_{u=1}^{M}\sum_{k=1}^{K}\lambda_{k,u}\},\end{aligned}\tag{10}$$

s.t. 式$(2)-(4)(7)$.

拉格朗日对偶问题的解为原问题提供一个下界.

3.2 求解工件级子问题(Solving job-level subproblems)

松弛后的工件级子问题用动态规划方法解决, 工件同一层次间和不同层次间的加工顺序要求包括在递归公式中. 对于求解工件级子问题的动态规划递归公式最早在job shop问题中给出[12–13]. 为此, 结合本文的多重入调度问题的特点, 给出动态规划过程中用到的变量和公式.

$h_{i,l,j}(u,t)$表示成本函数, 定义如下:

$$h_{i,l,j}(u,t)=\begin{cases}\sum\limits_{x=t-p_{i,j}+1}^{t}\lambda_{x,u}+w_it,\ l=L_i,\ j=J_i,\\ \sum\limits_{x=t-p_{i,j}+1}^{t}\lambda_{x,u},\qquad\text{其他},\end{cases}\tag{11}$$

$$i=1,\cdots,N;\ l=1,\cdots,L_i;\ j=1,\cdots,J_i;$$

$$t=T_{i,l,j},\cdots,K;\ u\in M_j,$$

其中: $T_{i,l,j}$表示工件i的第l层在工作站j的最早可完成时间, $h_{i,l,j}(u,t)$表示工件i的第l层在工作站j在时间t内在机器u上完成的成本.

$f_{i,l,j}(u,t)$表示工件i的第l层在工作站j在(u,t)状态下的最优评价指标. 解决每个工件级子问题的动态规划递归公式如下所示:

$$f_{i,l,j}(u,t)=\begin{cases}h_{i,l,j}(u,t),\ l=1,\ j=1,\\ h_{i,l,j}(u,t)+\\ \min\limits_{v\in M_{i,l-1,J_i},T_{i,l-1,J_i}\leqslant x\leqslant t-p_{i,j}}f_{i,l-1,J_i}(v,x),\\ \qquad l\neq 1,j=1,\\ h_{i,l,j}(u,t)+\\ \min\limits_{v\in M_{i,l,j-1},T_{i,l,j-1}\leqslant x\leqslant t-p_{i,j}}f_{i,l,j-1}(v,x),\\ \qquad\text{其他},\end{cases}\tag{12}$$

$$i=1,\cdots,N;\ l=1,\cdots,L_i;$$

$$j=1,\cdots,J_i;\ t=T_{i,l,j},\cdots,K;\ u\in M_j.$$

对工件i的第L_i层在工作站J_i的机器和完成时间

的最优决策可以由

$$(u^*_{i,L_i,J_i}, t^*_{i,L_i,J_i}) = \arg \min_{u\in M_{i,L_i,J_i}, T_{i,L_i,J_i}\leqslant t\leqslant K} f_{i,L_i,J_i}(u,t)$$

循环得到. 对工件i的第$L_i-1, L_i-2, \cdots, 1$层在工作站$J_i-1, J_i-2, \cdots, 1$的机器和完成时间的最优决策可以由下式循环得到:

$$(u^*_{i,l,j}, t^*_{i,l,j}) = \begin{cases} \arg \min\limits_{u\in M_{i,l,j}, T_{i,l,j}\leqslant t\leqslant t^*_{i,l+1,0}-p_{i,0}} f_{i,l,j}(u,t), \\ \quad l\neq L_i;\ j=J_i, \\ \arg \min\limits_{u\in M_{i,l,j}, T_{i,l,j}\leqslant t\leqslant t^*_{i,l,j+1}-p_{i,j+1}} f_{i,l,j}(u,t), \\ \quad \text{其他}. \end{cases} \tag{13}$$

3.3 构造可行解(Construction of a feasible solution)

对偶问题得到的解通常是不可行解, 因为可能违反了一部分机器能力约束(5), 因此需要构造可行解以获得原问题的上界, 以下为从松弛问题得到可行解的一个两阶段启发式方法: 第1阶段中, 每个工件的第1层在第1个工作站的完成时间$C_{i,l,j}$可以由松弛问题的解得到, 按照$C_{i,l,j}\times\sum\limits_{l=1}^{L_i}\sum\limits_{j=1}^{J_i} p_{i,j}/w_i$对每个工件进行升序排序, 依次分配给第1个工作站中最早空闲下来的机器. 随后每个工件的每1层的每个工作站都按照之前一个工作站的完成时间的顺序, 分配给最早空闲下来的机器. 第2阶段中, 对上述得到的可行解进行领域交换, 以得到更好的上界.

3.4 次梯度算法(Subgradient algorithm)

本文引入次梯度算法来求解拉格朗日对偶问题, 每次迭代过程中根据如下公式进行拉格朗日乘子更新:

$$\lambda_{k,u} = \max\{0, \lambda_{k,u} + \alpha\frac{\mathrm{UB}-\mathrm{LB}}{\sum\limits_{u=1}^{M}\sum\limits_{k=1}^{K} h^2_{k,u}}\times h_{k,u}\}, \tag{14}$$

$$h_{k,u} = \sum_{(i,l,j)\in O_u} (\varphi(k-C_{i,l,j}+p_{i,j}-1) - \varphi(k-C_{i,l,j}-1)) - 1, \tag{15}$$

$$k=1,\cdots,K;\ u=1,\cdots,M, \tag{16}$$

其中UB和LB分别表示当前迭代次数下的上界和下界.

3.5 拉格朗日松弛算法流程(Steps for Lagrangian relaxation algorithm)

步骤 1 初始化, 令当前迭代次数$n=0$, 令所有拉格朗日乘子$\lambda_{k,u}=0$.

步骤 2 用动态规划求解每个工件级子问题, 计算得到下界.

步骤 3 对松弛问题的解进行可行化, 计算得到上界.

步骤 4 判断是否符合停止准则, 若符合则算法停止, 否则继续.

步骤 5 进行拉格朗日乘子更新, 且$n=n+1$, 并回到步骤2.

4 实验与分析(Experiments and analysis)

本实验结果通过对偶间隙和CPU时间来衡量, 对偶间隙$=(\mathrm{UB}^*-\mathrm{LB}^*)/\mathrm{LB}^*\times 100\%$, 其中: UB^*表示所有迭代次数中的最优上界值, LB^*表示所有迭代次数中的最优下界值. 且设定最大迭代次数为300, 次梯度优化算法中取参数$\alpha=0.2$.

本实验测试了所提出的LR算法求解各种不同规模RHFS调度问题的性能. 问题实例随机产生如下:

$$N=\{10,20,30,40,50\},$$
$$L_i=\{[1,2],[2,3]\},\ J_{\mathrm{f}}=\{2,3,4\},$$
$$M_j=\{3,4,5,6,7,8,9,10,11,12,13\},$$

其中$L_i=\{[1,2],[2,3]\}$表示工件的层次数取两种不同的规模, 且每种规模下, 每个工件的层次数不同且在给定范围内按离散均匀分布随机产生; $J_{\mathrm{f}}=\{2,3,4\}$, 且由于检查修复阶段的重入次数L'_i由$[1,2]$离散均匀分布随机产生, 则可得工件i经过的工作站数$J_i=\{[4,6],[5,7],[6,8]\}$; 每个工件的加工时间和权重均分别由$[10,20]$和$[1,10]$离散均匀分布随机产生. N, L_i, J_{f}, M_j共有$5\times2\times3\times3=90$种规模, 对每种规模随机产生和测试10个不同的实例, 故本实验共产生和使用900个测试实例.

本文算法采用以下计算机配置进行实现: CPU: 英特尔, 奔腾, 双核, T2130, 1.86 GHz; 内存: 896 MB; 电脑系统: Windows XP Professional; 编程语言: C++.

实验结果如表1所示, 表1中的数据(除最后一行外)都是实验中相同规模的10个实例的平均性能. 表1中: $M=$工件数×层次数×工作站数×机器数.

从表1可以得出如下结论:

1) 对于各种不同规模, 在相同工件数、层次数和工作站数条件下, 随着机器数增加, 对偶间隙总会变小. 这是由于随着机器数增加, 工件之间对于机器的竞争就减少, 从而使问题变得简单一些, 解的质量就有所提高.

2) 除了工件数为10的小规模问题外, 对于其他各种不同规模, 在相同工件数、层次数和机器数条件下, 随着工作站数增加, 对偶间隙总会变小. 这是由于随着工作站数增加, 各个工件的不同层次之间对于机器的竞争会越小, 因而对偶间隙会变小.

表 1 不同规模问题的实验结果

Table 1 Experimental results of different scales of problems

问题	M	对偶间隙/%	CPU时间/s	问题	M	对偶间隙/%	CPU时间/s
1	$10\times[1,2]\times[4,6]\times3$	6.52	16.22	46	$30\times[2,3]\times[4,6]\times7$	19.34	199.04
2	$10\times[1,2]\times[4,6]\times4$	3.65	14.41	47	$30\times[2,3]\times[4,6]\times8$	15.84	181.6
3	$10\times[1,2]\times[4,6]\times5$	2.62	16.02	48	$30\times[2,3]\times[4,6]\times9$	13.1	186.97
4	$10\times[1,2]\times[5,7]\times3$	5.51	23.74	49	$30\times[2,3]\times[5,7]\times7$	15.47	268.34
5	$10\times[1,2]\times[5,7]\times4$	3.54	21.58	50	$30\times[2,3]\times[5,7]\times8$	10.39	251.1
6	$10\times[1,2]\times[5,7]\times5$	2.46	22.94	51	$30\times[2,3]\times[5,7]\times9$	7.4	247.84
7	$10\times[1,2]\times[6,8]\times3$	5.54	31.9	52	$30\times[2,3]\times[6,8]\times7$	9.58	352.51
8	$10\times[1,2]\times[6,8]\times4$	3.17	31.05	53	$30\times[2,3]\times[6,8]\times8$	8.56	340.31
9	$10\times[1,2]\times[6,8]\times5$	2.47	28.64	54	$30\times[2,3]\times[6,8]\times9$	5.94	345.84
10	$10\times[2,3]\times[4,6]\times3$	11.53	45.68	55	$40\times[1,2]\times[4,6]\times9$	14.55	107.33
11	$10\times[2,3]\times[4,6]\times4$	5.49	38.96	56	$40\times[1,2]\times[4,6]\times10$	12.78	112.31
12	$10\times[2,3]\times[4,6]\times5$	3.24	39.93	57	$40\times[1,2]\times[4,6]\times11$	10.51	107.38
13	$10\times[2,3]\times[5,7]\times3$	10.86	66.86	58	$40\times[1,2]\times[5,7]\times9$	12.12	149.64
14	$10\times[2,3]\times[5,7]\times4$	3.78	59.27	59	$40\times[1,2]\times[5,7]\times10$	10.63	156.61
15	$10\times[2,3]\times[5,7]\times5$	2.49	57.02	60	$40\times[1,2]\times[5,7]\times11$	8.09	147.81
16	$10\times[2,3]\times[6,8]\times3$	4.51	82.81	61	$40\times[1,2]\times[6,8]\times9$	11.11	204.35
17	$10\times[2,3]\times[6,8]\times4$	2.4	78.1	62	$40\times[1,2]\times[6,8]\times10$	8.54	208.25
18	$10\times[2,3]\times[6,8]\times5$	1.5	81.22	63	$40\times[1,2]\times[6,8]\times11$	6.86	196.05
19	$20\times[1,2]\times[4,6]\times5$	10.72	41.58	64	$40\times[2,3]\times[4,6]\times9$	21.9	289.5
20	$20\times[1,2]\times[4,6]\times6$	6.82	40.13	65	$40\times[2,3]\times[4,6]\times10$	16.57	289.97
21	$20\times[1,2]\times[4,6]\times7$	5.54	38.43	66	$40\times[2,3]\times[4,6]\times11$	14.07	266.37
22	$20\times[1,2]\times[5,7]\times5$	9.09	61.36	67	$40\times[2,3]\times[5,7]\times9$	16.56	390.25
23	$20\times[1,2]\times[5,7]\times6$	6.77	58.59	68	$40\times[2,3]\times[5,7]\times10$	14.79	403.16
24	$20\times[1,2]\times[5,7]\times7$	5.11	53.1	69	$40\times[2,3]\times[5,7]\times11$	9.51	367.3
25	$20\times[1,2]\times[6,8]\times5$	7.86	80.84	70	$40\times[2,3]\times[6,8]\times9$	13.22	523.8
26	$20\times[1,2]\times[6,8]\times6$	5.07	78.19	71	$40\times[2,3]\times[6,8]\times10$	9.92	522.33
27	$20\times[1,2]\times[6,8]\times7$	4.45	71.84	72	$40\times[2,3]\times[6,8]\times11$	7.62	510.91
28	$20\times[2,3]\times[4,6]\times5$	17.06	116.73	73	$50\times[1,2]\times[4,6]\times11$	16.47	151.94
29	$20\times[2,3]\times[4,6]\times6$	10.35	102.87	74	$50\times[1,2]\times[4,6]\times12$	13.31	156.85
30	$20\times[2,3]\times[4,6]\times7$	8.18	98.38	75	$50\times[1,2]\times[4,6]\times13$	10.85	149.48
31	$20\times[2,3]\times[5,7]\times5$	13.83	166.67	76	$50\times[1,2]\times[5,7]\times11$	13.27	207.94
32	$20\times[2,3]\times[5,7]\times6$	7.37	146.46	77	$50\times[1,2]\times[5,7]\times12$	10.84	211.72
33	$20\times[2,3]\times[5,7]\times7$	5.48	140.41	78	$50\times[1,2]\times[5,7]\times13$	9.52	203.32
34	$20\times[2,3]\times[6,8]\times5$	10	213.85	79	$50\times[1,2]\times[6,8]\times11$	9.47	270.1
35	$20\times[2,3]\times[6,8]\times6$	5.02	198.03	80	$50\times[1,2]\times[6,8]\times12$	9.6	289.13
36	$20\times[2,3]\times[6,8]\times7$	3.71	191.81	81	$50\times[1,2]\times[6,8]\times13$	7.26	259.84
37	$30\times[1,2]\times[4,6]\times7$	14.47	75.94	82	$50\times[2,3]\times[4,6]\times11$	22.05	388.36
38	$30\times[1,2]\times[4,6]\times8$	11.23	68.89	83	$50\times[2,3]\times[4,6]\times12$	19.33	387.16
39	$30\times[1,2]\times[4,6]\times9$	9.22	74.19	84	$50\times[2,3]\times[4,6]\times13$	16.34	367.95
40	$30\times[1,2]\times[5,7]\times7$	10.8	101.52	85	$50\times[2,3]\times[5,7]\times11$	17.04	529.13
41	$30\times[1,2]\times[5,7]\times8$	9.25	99.1	86	$50\times[2,3]\times[5,7]\times12$	14.99	540.25
42	$30\times[1,2]\times[5,7]\times9$	6.87	100.72	87	$50\times[2,3]\times[5,7]\times13$	12.12	514.6
43	$30\times[1,2]\times[6,8]\times7$	9.91	135.6	88	$50\times[2,3]\times[6,8]\times11$	13.72	707.41
44	$30\times[1,2]\times[6,8]\times8$	7.56	130.64	89	$50\times[2,3]\times[6,8]\times12$	10.36	713.72
45	$30\times[1,2]\times[6,8]\times9$	6.12	134.81	90	$50\times[2,3]\times[6,8]\times13$	8.12	686.2
				平均		9.65	195.99

3) 除了工件数为10的小规模问题外, 对于其他各种不同规模, 在相同工件数、工作站数和机器数条件下, 随着层次数增加, 对偶间隙总会变大. 这是由于, 随着层次数增加, 各个工件的不同层次之间对于机器的竞争会变大.

4) 对于各种不同规模, 在相同层次数、工作站

数和机器数条件下, 当工件数增加时, 对偶间隙总会变大, 这是由于随着工件数增加, 工件之间对机器的竞争就越大.

5) 所有实例的平均对偶间隙和平均CPU时间分别为9.65%和195.99 s. 这说明对于本文所研究的带有重入系统的RHFS调度问题, 所设计的LR算法能够在较为合理的时间内得到比较满意的近优解.

LR不仅能够为实际问题提供可量化指标, 同时也能得到该问题的解. 本文将LR所得出的解与较为高效的智能优化算法, 遗传算法(genetic algorithm, GA)和粒子群算法(particle swarm optimization, PSO)进行比较. 图2表示各个算法在不同工件规模下180个实例的δ平均值, 其中$\delta = (\mathrm{UB}-\mathrm{LB}^*)/\mathrm{LB}^* \times 100\%$. 对于LR, UB即为最优上界值$\mathrm{UB}^*$, 对于GA和PSO, UB表示两个算法200次迭代后所得的目标函数值. 本文所使用的GA和PSO均为标准流程. 从图2可以看出, PSO的解的质量在各个不同工件规模下均略优于GA, 而LR的解在各种规模下均明显由于GA和PSO.

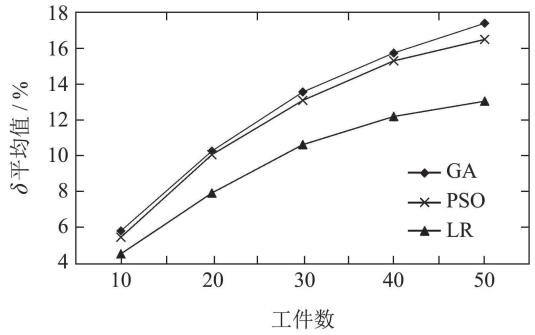

图 2 不同工件规模下各算法的δ平均值

Fig. 2 δ average in different number of jobs of different algorithms

5 结论(Conclusion)

本文研究以半导体制造为背景的RHFS调度问题, 考虑了实际生产中检查和修复对于可重入生产系统的重要性, 提出了带有重入系统的RHFS调度问题, 并且针对该问题进行问题描述, 以最小化加权完成时间为目标, 建立数学模型. 本文使用LR算法, 通过松弛机器能力约束, 将松弛问题转化为工件级子问题, 并建立动态规划的递归公式求解该问题. 实验分析了各种规模问题的实验结果, 得出了相应的结论. 结果表明, 本文所设计的LR算法能够在合理的时间内获得满意的近优解. 后续研究可以考虑如何减少算法运行时间以及如何进一步增加约束条件以提高大规模问题的解的质量.

参考文献(References):

[1] CHOI S W, KIM Y D, LEE G C. Minimizing total tardiness of orders with reentrant lots in a hybrid flowshop [J]. *International Journal of Production Research*, 2005, 43(11): 2149 – 2167.

[2] JIANG S, TANG L. Lagrangian relaxation algorithms for re-entrant hybrid flowshop scheduling [C] *//2008 IEEE International Conference on Information Management, Innovation Management and Industrial Engineering*. Taipei: IEEE, 2008: 78 – 81.

[3] DUGARDIN F, AMODEO L, YALAOUI F. Multiobjective scheduling of a reentrant hybrid flowshop [C] *//2009 IEEE International Conference on Computers & Industrial Engineering*. Troyes: IEEE, 2009: 193 – 195.

[4] DUGARDIN F, YALAOUI F, AMODEO L. New multi-objective method to solve reentrant hybrid flow shop scheduling problem [J]. *European Journal of Operational Research*, 2010, 203(1): 22 – 31.

[5] CHO H M, BAE S J, KIM J, et al. Bi-objective scheduling for reentrant hybrid flow shop using Pareto genetic algorithm [J]. *Computers & Industrial Engineering*, 2011, 61(3): 529 – 541.

[6] HEKMATFAR M, FATEMI GHOMI S M T, KARIMI B. Two stage reentrant hybrid flow shop with setup times and the criterion of minimizing makespan [J]. *Applied Soft Computing*, 2011, 11(8): 4530 – 4539.

[7] RAU H, CHO K H. Genetic algorithm modeling for the inspection allocation in reentrant production systems [J]. *Expert Systems with Applications*, 2009, 36(8): 11287 – 11295.

[8] NISHI T, HIRANAKA Y, INUIGUCHI M. Lagrangian relaxation with cut generation for hybrid flowshop scheduling problems to minimize the total weighted tardiness [J]. *Computers & Operations Research*, 2010, 37(1): 189 – 198.

[9] 轩华. 运输能力有限混合流水车间调度的改进拉格朗日松弛算法 [J]. 计算机集成制造系统, 2013, 19(7): 1633 – 1639.
(XUAN Hua. Improved LR algorithm for hybrid flowshop scheduling with transportation consideration [J]. *Computer Integrated Manufacturing Systems*, 2013, 19(7): 1633 – 1639.)

[10] MAO K, PAN Q, PANG X, et al. A novel Lagrangian relaxation approach for a hybrid flowshop scheduling problem in the steelmaking-continuous casting process [J]. *European Journal of Operational Research*, 2014, 236(1): 51 – 60.

[11] 刘国莉, 张博, 唐立新. 可延迟供货的冷轧生产库存问题之建模与优化 [J]. 控制理论与应用, 2012, 29(11): 1512 – 1516.
(LIU Guoli, ZHANG Bo, TANG Lixin. Modeling and optimization for a cold rolling production-inventory problem with backlogging [J]. *Control Theory & Applications*, 2012, 29(11): 1512 – 1516.)

[12] CHEN H, CHU C, PROTH J M. A more efficient Lagrangian relaxation approach to job-shop scheduling problems [C] *//1995 IEEE International Conference on Robotics and Automation*. Nagoya: IEEE, 1995: 496 – 501.

[13] CHEN H, CHU C, PROTH J M. An improvement of the Lagrangean relaxation approach for job shop scheduling: a dynamic programming method [J]. *IEEE Transactions on Robotics and Automation*, 1998, 14(5): 786 – 795.

作者简介:

周炳海 (1965–), 男, 博士, 同济大学工业工程研究所所长, 博士生导师, 主要从事离散系统建模、调度与仿真等方向的研究, E-mail: bhzhou@tongji.edu.cn;

钟臻怡 (1990–), 女, 博士研究生, 主要从事智能优化和生产系统建模、调度等方向的研究, E-mail: 986032363@qq.com.