文章编号: 1001-0920(2011)01-0037-07

有滞留时间约束的集束型装备在线调度问题研究

李林瑛[1,2], 胡静涛[1]

(1. 中国科学院沈阳自动化研究所 工业信息学重点实验室，沈阳 110016；2. 中国科学院 研究生院，北京 100039)

摘 要: 针对半导体制造中有滞留时间约束的集束型装备, 研究了临时晶圆到达时的在线调度问题, 描述了调度问题域, 建立了问题的数学模型, 并根据模型提出了两层调度方法. 外层算法通过粒子群优化过程求解临时晶圆的加工顺序; 内层算法在给定加工顺序的基础上, 采用前向和后向递推方法获得可行解空间, 并从可行解空间获得最优完工时间. 从理论上证明了算法的可行性, 并通过仿真结果表明, 该方法对求解大规模临时晶圆的调度问题是十分有效的.

关键词: 半导体制造；集束型装备；在线调度；粒子群优化算法；滞留时间约束

中图分类号: TP241.2；TP278 **文献标识码:** A

Online scheduling problem of cluster tools with residency time constraints

LI Lin-ying[1,2], *HU Jing-tao*[1]

(1. Key Laboratory of Industrial Informatics，Shenyang Institute of Automation of Chinese Academy of Sciences, Shenyang 110016， China； 2. School of Graduate， Chinese Academy of Sciences， Beijing 100039， China. Correspondent：LI Lin-ying，E-mail：lilinying@sina.cn)

Abstract: For cluster tools with residency time constraints in semiconductor manufacturing, the online scheduling problem is addressed when temporary wafer arrives. On the basis of the statement domain of scheduling problem, mathematical formulations of the scheduling problems are presented. The two-layer method is proposed. The outer-layer algorithm denotes to solve the processing order of temporary wafer by particle swarm optimization, while the inner-layer one determines the start time of each station based on the given order by using the two-stage recursion method. The feasibility of the inner algorithm is proved theoretically. The simulation results show that the proposed method is effective for solving the large-scale scheduling problem of temporary wafer.

Key words: semiconductor manufacturing；cluster tools；online scheduling；particle swarm optimization algorithm; residency time constraints

1 引 言

半导体制造中, 集束型装备由晶圆加工设备和机械手运输设备构成, 应用于晶圆的刻蚀、光刻以及化学气相淀积等加工过程. 集束型装备在加工晶圆时, 晶圆在加工设备中的停留时间必须限制在规定的时间范围内, 这一要求称为滞留时间约束. 例如, 化学淀积工艺要求在真空度为 0.25 ～ 20 Torr 和高温为 550 ℃～800 ℃ 的条件下对腔内晶圆镀膜, 滞留时间不能超过 20 s, 否则, 晶圆表面会被高温破坏甚至晶圆破损[1-2]. 滞留时间约束增加了调度的困难性, 也使求解可行调度方案变得困难. 半导体生产线存在着来自顾客的紧急订单或返工晶圆, 这些订单 (临时晶圆) 的到达时间、加工时间及数量等信息, 在晶圆进入集束型装备前是未知的, 这种信息的不确定性使离线调度方案失去可行性, 必须采用有效的在线调度方法. 另外, 这类问题与传统的无等待流水线调度问题的区别在于[3-6]: 晶圆的运输由机械手完成, 运输时间不能忽略.

对于有滞留时间约束的集束型装备优化调度问题, 国内外已经做了大量的研究. 文献 [7-8] 提出了一种启发式迭代算法, 该算法利用基于图论方法求解线性规划模型. [9] 采用 P-time 的 Petri 网方法建模, 通过线性规划模型来确定可调度性. 该方法从可行性研究方面而言具有重要意义, 但方法的模型复杂, 线性

收稿日期: 2009-10-27；**修回日期:** 2009-12-28.

基金项目: 沈阳市科技计划项目(108155-2-00).

作者简介: 李林瑛(1975−), 男, 博士生, 从事半导体自动制造系统建模与调度的研究；胡静涛(1963−), 男, 研究员, 博士生导师, 从事远程设备监测与故障诊断等研究.

规划求解也不是多项式算法[10]. [11]利用面向资源Petri网建模, 并对可行调度方案的充要条件进行论证. 这些进展在离线调度中十分重要, 但不适合在线调度问题. [12]提出了基于模拟退火的在线调度方法, 本质上属于线性规划模型的离线调度算法, 实时性较差, 难以保证临时晶圆的在线调度要求. [13]针对晶圆加工路径不同、并行设备及允许晶圆跳过不必要的加工工位等情况, 提出了基于时间约束集的在线调度方法, 但求解问题的规模有限, 不适合较大规模临时晶圆的调度.

针对临时晶圆的加工问题, 当临时晶圆的数目为1时, 可以直接按文献[12-13]中提到的算法计算, 但随着问题规模的增加, 直接计算很困难. 针对以上问题, 本文提出了求解大规模调度问题的两层在线方法. 外层粒子群(PSO)算法迭代地求解晶圆的加工顺序; 内层算法采用两阶段方法, 针对给定的加工顺序, 利用资源的允许区间确定晶圆的加工开始时间, 从而获得最优调度方案. 仿真实验表明, 该方法可以获得仿真算例的最优解.

2 在线调度问题

图1所示为集束型装备. 不同类型的晶圆依照配方从输入装载室进入, 定位后经过每个工位的加工设备(机器), 加工完成后经过冷却返回输出装载室. 集束型装备的工位存在并行设备(加工时间长的工位为了实现负载平衡, 采取多个相同设备完成同一工位的加工, 这些设备称为并行设备), 且为单晶圆加工设备, 工位之间无缓冲; 机械手旋转移动以保证晶圆在装载室和机器之间的运输. 集束型装备的调度就是要合理地安排各加工活动的开始时间, 以使生产周期最短.

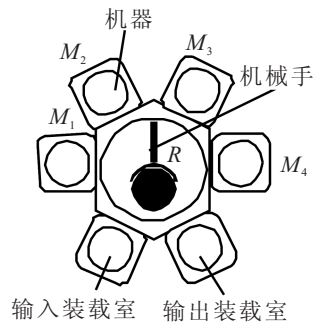

图1 集束型装备

广义的实时调度实现方式分为两类: 一类是使用允许时间在线调度, 即事先不存在静态调度方案, 直接按照生产系统中的晶圆、设备的状况和相关信息, 使用某种计算方法, 确定晶圆或设备的加工决策; 另一类是重调度[14-15], 即在已有静态调度方案的基础上, 根据生产系统的现场状态, 及时进行静态调度方案的调整, 给出适合现场状态的调度方案. 本文考虑第一类调度, 并假设临时晶圆到达时装备内的晶圆继续加工, 装备外等待的晶圆暂停, 调度系统优先考虑临时晶圆的作业安排. 在实时调度系统中, 资源(机器和机械手)可以使用的加工时间和运输时间是一些时间区间值, 能否在这些区间值上加工取决于要加工晶圆的滞留时间及资源的空闲时间. 因此, 合理安排临时晶圆的加工顺序和滞留时间至关重要.

2.1 数学模型

假设有N个工位加工晶圆, 如图2所示. 有以下约束成立:

$$t^s_{n+1} = t^s_n + T^P_n + T^R_n + T^U_n + T^M_n + T^L_n, \quad (1)$$

$$0 \leqslant T^R_n \leqslant T^D_n, \quad (2)$$

$$T^{TR}_{n+1} = T^{M'}_n + T^U_n + T^M_n + T^L_n. \quad (3)$$

其中: t^s_n和t^f_n为在每个工位晶圆加工的开始和结束时间, n为工位序号, T^P_n为晶圆的加工时间, T^R_n为晶圆的滞留时间, T^D_n为晶圆滞留时间的上界, T^{TR}_{n+1}为机械手空载到工位n且运输晶圆到工位$n+1$所用时间. 式(1)表示两个相邻工位加工的时序关系, 式(2)表示滞留时间约束, 式(3)的$T^{M'}_n$,T^U_n,T^M_n和T^L_n分别表示机械手空载、卸载、运输和装载活动所需时间.

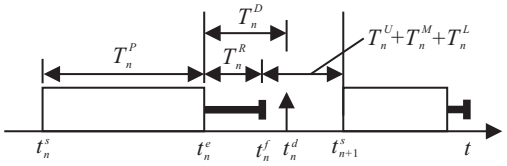

图2 问题参数的时序关系

调度问题可描述为在资源(机器和机械手统称为资源)的允许时间区间内, 寻找临时晶圆在各工位的最优加工开始时间, 以使晶圆的最大完成时间最小. CM_n和CT_n表示机器和机械手的允许区间集合, 即

$$CM_n = \{[L^m_{n,1}, U^m_{n,1}], [L^m_{n,2}, U^m_{n,2}], \cdots, [L^m_{n,\gamma(n)}, U^m_{n,\gamma(n)}]\},$$

$$CT_n = \{[L^t_{n,1}, U^t_{n,1}], [L^t_{n,2}, U^t_{n,2}], \cdots, [L^t_{n,\tau(n)}, U^t_{n,\tau(n)}]\},$$

其中$\gamma(n)$和$\tau(n)$分别表示CM_n和CT_n时间区间的数目.

临时晶圆在各个工位的资源允许时间点t满足以下约束:

$$t \in [L^m_{n,i}, U^m_{n,i}] \text{且} t \in [t^s_n, t^s_n + T^P_n + T^R_n],$$
$$i = 1, 2, \cdots, \gamma(n),\ n = 1, 2, \cdots, N; \quad (4)$$

$$t \in [L^t_{n,i}, U^t_{n,i}] \text{且} t \in [t^s_n - T^{TR}_n, t^s_n],$$
$$i = 1, 2, \cdots, \tau(n),\ n = 1, 2, \cdots, N. \quad (5)$$

式(4)和(5)分别称为加工和运输时间约束[10]. 由式(1)~(5)构成了问题的数学模型.

2.2 寻找最优加工开始时间的内层算法

单臂具有滞留时间约束的集束型装备调度问题已被证明是NP-hard问题[16], 求解其调度模型有较大困难. 为了有效、快速地求解已建立的调度模型, 本文提出了一个启发式调度算法, 在给定晶圆加工顺序的条件下, 计算临时晶圆在各工位的最优加工开始时间. 算法分为两个阶段: 利用前向和后向递推方法寻找t_n^s和t_n^f的可行解空间; 然后在可行解空间获得最优解.

为了方便算法描述, 做以下定义:

1) 定义加工开始和结束时间的集合CS_n和CF_n, 有

$$CS_n = \{[L_{n,1}^s, U_{n,1}^s], [L_{n,2}^s, U_{n,2}^s], \cdots, [L_{n,\alpha(n)}^s, U_{n,\alpha(n)}^s]\},$$

$$CF_n = \{[L_{n,1}^f, U_{n,1}^f], [L_{n,2}^f, U_{n,2}^f], \cdots, [L_{n,\beta(n)}^f, U_{n,\beta(n)}^f]\}.$$

其中$\alpha(n)$和$\beta(n)$分别表示CS_n和CF_n时间区间的数目. 加工区间按下界点的递增顺序排列, 如果区间的下界点相同, 则按上界点的递增顺序排列. 该排列方法同样适用于资源区间CM_n和CT_n.

2) 定义集合的转换运算、最小和最大运算. 令$C = \{[L_1, U_1], [L_2, U_2], \cdots, [L_m, U_m]\}$为时间集合, 转换运算$\oplus$, 最小运算min和最大运算max分别为

$$C \oplus [a, b] = \{[L_1 + a, U_1 + b], \cdots, [L_m + a, U_m + b]\},$$

$$\min(C) = L_1,\ \max(C) = U_m.$$

寻找晶圆最优加工开始时间t_n^s的算法如下:

$$CS_1 = \left(\{[L_{1,k}^m - T_1^{TR}, U_{1,k}^m - T_1^P]\} \bigcap CT_1\right) \oplus [T_1^{TR}, 0]; \tag{6}$$

$$CF_1 = \{[L_{1,k}^m + T_1^P, U_{1,k}^m]\} \bigcap (CS_1 \oplus [T_1^P, T_1^P + T_1^D]); \tag{7}$$

$$CS_n = \left(\left(\left((CM_n \oplus [0, -T_n^P]) \bigcap (CF_{n-1} \oplus [T_n^{TR}, T_n^{TR}])\right) \oplus [-T_n^{TR}, 0]\right) \bigcap CT_n\right) \oplus [T_n^{TR}, 0],\ n = 2, 3, \cdots, N; \tag{8}$$

$$CF_n = (CM_n \oplus [T_n^P, 0]) \bigcap (CS_n \oplus [T_n^P, T_n^P + T_n^D]),\ n = 2, 3, \cdots, N; \tag{9}$$

$$CS_{N+1} = \left(\left(\left((CM_N \oplus [0, -T_N^P]) \bigcap (CF_N \oplus [T_{N+1}^{TR}, T_{N+1}^{TR}])\right)[-T_{N+1}^{TR}, 0]\right) \bigcap CT_{N+1}\right) \oplus [T_{N+1}^{TR}, 0],$$

$$CM_N = \{[0, \infty]\}; \tag{10}$$

$$t_{N+1}^s = \min(CS_{N+1}); \tag{11}$$

$$t_n^f = t_{n+1}^s - T_{n+1}^{TR},\ n = N, N-1, \cdots, 1; \tag{12}$$

$$t_n^s = \max\left(\{[t_n^f - T_n^P - T_n^D, t_n^f - T_n^P]\} \bigcap CS_n\right),\ n = N, N-1, \cdots, 1. \tag{13}$$

上述算法从CM_n中循环取出$[L_{1,k}^m, U_{1,k}^m](k = 1, 2, \cdots, \gamma(n))$, 然后由式(6)~(10)前向递推构造可行解空间, 根据式(11)~(13)后向递推, 实现从可行解空间中获得最优解. 该算法的时间复杂度取决于时间区间的交集运算效率和次数(运算次数为工位数N), 而离散、有序的两集合交集算法的复杂度为$O(m+n)$[17](m和n表示两集合的区间数目), 所以上述算法的时间复杂度为$O(N(m+n))$.

2.3 算法可行性

定理 1　式(8)和(9)是所有可行的t_n^s和t_n^f的充要条件.

证明　式(6), (7)和(10)是(8)和(9)的特殊情况, 所以仅需证明式(8)和(9)是所有可行的t_n^s和t_n^f的充要条件.

1) 式(8)的充分性.

加工开始时间$t(t \in CS_n$, 表示$t \in [L_{1,k}^s, U_{1,k}^s], k = 1, 2, \cdots, \alpha(n))$是$\{[t - T_n^{TR}, t]\} \subset CT_n$和$\{[t, t + T_n^P]\} \subset CM_n$的充分条件. 在式(8)左右两端作$\oplus [-T_n^{TR}, 0]$, 有

$$\{t, t\} \oplus [-T_n^{TR}, 0] \subset CS_n \oplus [-T_n^{TR}, 0] = \left(\left((CM_n \oplus [0, -T_n^P]) \bigcap (CF_{n-1} \oplus [T_n^{TR}, T_n^{TR}])\right) \oplus [-T_n^{TR}, 0]\right) \bigcap CT_n. \tag{14}$$

由式(14)有

$$\{[t - T_n^{TR}, t]\} \subset CT_n,$$

$$\{[t - T_n^{TR}, t]\} \subset \left((CM_n \oplus [0, -T_n^P]) \bigcap (CF_{n-1} \oplus [T_n^{TR}, T_n^{TR}])\right) \oplus [-T_n^{TR}, 0]. \tag{15}$$

由式(15)有

$$(\{[t - T_n^{TR}, t]\} \oplus [T_n^{TR}, 0]) \oplus [0, T_n^P] = \{[t, t + T_n^P]\} \subset CM_n.$$

2) 式(8)的必要性(反证法).

假设存在$t_{n-1}^{f*} \in CF_{n-1}$和$t_n^{s*} \notin CS_n$, 由$t_n^{s*} = t_{n-1}^{f*} + T_n^{TR}$, $t_n^{s*} \in (CM_n \oplus [0, -T_n^P])$和$[t_n^{s*} - T_n^{TR}, t_n^{s*}] \subset CT_n$可知

$$CS_n = \left(\left(\left((CM_n \oplus [0, -T_n^P]) \bigcap (\{[t_{n-1}^{f*}, t_{n-1}^{f*}]\} \oplus [T_n^{TR}, T_n^{TR}])\right) \oplus [-T_n^{TR}, 0]\right) \bigcap CT_n\right) \oplus [T_n^{TR}, 0] =$$

$$\left(\left((CM_n \oplus [0, -T_n^P]) \bigcap (\{[t_n^{s*}, t_n^{s*}]\}) \oplus [-T_n^{TR}, 0]\right) \bigcap CT_n\right) \oplus [T_n^{TR}, 0] =$$

$$\left(\left(\{[t_n^{s*}, t_n^{s*}]\} \oplus [-T_n^{TR}, 0]\right) \bigcap CT_n\right) \oplus [T_n^{TR}, 0] =$$

$$[t_n^{s*}-T_n^{TR},t_n^{s*}]\oplus[T_n^{TR},0]=\{[t_n^{s*},t_n^{s*}]\}.$$

结果与假设矛盾, 故假设不成立.

3) 式 (9) 的充分性.

式 (9) 计算所得 $t(t\in CF_n,t_n^{s*}=t-T_n^P-T_n^R)$ 是 $t\in CM_n$ 和 $t_n^{s*}+T_n^P\leqslant t\leqslant t_n^{s*}+T_n^P+T_n^D$ 的充分条件. 由式 (9) 有 $[t,t]\subset CM_n\oplus[T_n^P,0]$ 且 $CM_n\oplus[T_n^P,0]\subset CM_n$, 所以 $t\in CM_n$. 又根据式 (9) 有

$$[t,t]\subset\{[t_n^{s*},t_n^{s*}]\}\oplus[T_n^P,T_n^P+T_n^D],$$

故 $t_n^{s*}+T_n^P\leqslant t\leqslant t_n^{s*}+T_n^P+T_n^D$.

4) 式 (9) 的必要性 (反证法).

假设存在 $t_n^{s*}\in CS_n$ 和 $t_n^{f*}\notin CF_n$. 由 $t_n^{f*}\in CM_n\oplus[T_n^P,0]$ 和 $t_n^{f*}\in\{[t_n^{s*}+T_n^P,t_n^{s*}+T_n^P+T_n^D]\}$ 可知

$$t_n^{f*}\in(CM_n\oplus[T_n^P,0])\bigcap\{[t_n^{s*}+T_n^P,t_n^{s*}+T_n^P+T_n^D]\},$$

所以 $t_n^{f*}\in CF_n$. 这与假设相矛盾, 故假设不成立. □

定理 1 保证了算法求得的 t_n^s 和 t_n^f 满足约束条件 (1)~(5). 上述算法的目标为求 t_{N+1}^s 的最小值. 由式 (6)~(9) 可以计算 (10) 每个允许区间的上界和下界点. 需要注意的是, 式 (10) 的允许区间可能互相重叠 (即一个区间的上界点落在另一个区间中间) 而产生重叠冗余段. 为了简化分析和消除冗余区间段, 需要将相互重叠的区间进行合并. 为达到上述目的, 将式 (10) 中的每个允许区间按其下界点值从小到大的顺序重新排列, 然后依次对相互重叠的区间进行合并即可.

定理 2 允许区间集 CS_{N+1} 中第 1 个区间的下界点 $t_{N+1}^s=\min(CS_{N+1})$ 为问题的最优解.

证明 首先, 证明区间集合合并前后的等价性. 通过上述分析可知, 区间合并只是将那些相互重叠的小区间合并为一个大区间, 消除了其中的重叠冗余区间段, 而没改变允许区间的分布范围, 所以区间合并前后的两个区间集合完全等价; 其次, 证明 t_{N+1}^s 为问题的可行解. 过程同定理 1, 此略; 最后, 证明 t_{N+1}^s 为问题的全局最小可行解, 鉴于区间集合按升序排列且合并前后区间集的等价性, 显然 $t_{N+1}^s=\min(CS_{N+1})$ 为允许区间集的最小值, 即为全局最优解. □

定理 3 算法求得的 T_n^R 为滞留时间最小值.

证明 由式 (11) 和 (12) 可知, t_n^f 是晶圆在机器的最早结束时间, 又由式 (13) 可知, t_n^s 是满足滞留时间约束的最晚开始时间, 所以 v 为滞留时间最小值. □

算法求得的 T_n^R 为滞留时间最小值, 最大限度地保证了晶圆的加工质量, 防止腔内残留的腐蚀气体或热量破坏已完成加工的晶圆.

定理 4 如果算法求得的 CS_n 或 CF_n 为空, 则算法继续求解将得不到问题的可行解.

证明 式 (11) 和 (12) 通过前向递推法逐步求得 CS_n 和 CF_n, 若 CS_n 或 CF_n 为空, 则后续的 CS_n 或 CF_n 也为空, 所以调度方案不可行. □

定理 4 表明, 如果 CS_n 或 CF_n 为空, 则用包含该区间的任何后续区间构造的开始加工和结束时间均不是可行解. 当 CS_n 或 CF_n 为空时, 必须重新寻找资源的可加工区间, 所以需要利用 CM_1 和 CT_1 的下一个允许加工区间, 并利用上述算法重新构造 CS_n 和 CF_n.

令 $\varphi(n)$ 表示工位 n 的并行设备数目; $CM_n^{\varphi(n)}$, $CS_n^{\varphi(n)}$ 和 $CF_n^{\varphi(n)}$ 表示在工位 n 的机器 $\varphi(n)$ 允许、加工开始和结束时间的区间集合. 无需改变内层算法的逻辑结构, 用 $CM_n^{\varphi(n)}$, $CS_n^{\varphi(n)}$ 和 $CF_n^{\varphi(n)}$ 更换式 (6)~(13) 中的 CM_n, CS_n 和 CF_n, 算法同样能够解决具有并行设备的情况.

3 在线调度方法

随着待加工晶圆数目的增加, 为了尽快完成这些临时晶圆的加工任务, 在线调度方法使用粒子群算法对临时进入集束型装备的晶圆加工顺序进行迭代; 在给定的加工顺序下, 利用第 2.2 节的内层算法获得晶圆在每个工位的最优加工开始时间.

3.1 外层粒子群算法

3.1.1 编 码

粒子群算法是一种典型的群体智能实现模型, 是由鸟群运动模拟抽象而成的算法和模型. 粒子群算法由 James等首次提出. 该算法最初用于连续空间的优化, 其优化性能通过大量的实验已得到验证. 粒子位置和速度矢量的表达以及粒子更新策略等均具有连续本质, 而本文的调度问题是复杂的离散问题, 故需要设计位置矢量编码.

本文调度问题的关键是如何安排晶圆的加工顺序, 直接的编码方法是用位置矢量的一维代表一个晶圆, 这样粒子本身就可以表示所有晶圆的一个排列, 即一个调度. 例如, 6 维粒子向量和 6 晶圆的排列对应关系为

粒子向量 : $(2.80,1.71,-0.93,3.24,-2.34,-1.30)$.

晶圆排列 : $(5,6,3,2,1,4)$.

其中粒子向量中元素最小值 -2.34 对应的晶圆 5 最早加工, 依次类推.

3.1.2 惯性权重

惯性权重 w 调整个体历史信息对粒子的影响, 较大的 w 使粒子保留较多的历史信息, 有利于群体的多样性; 反之, 较小的 w 则有利于群体集中性. 因此, 可先令 w 取较大值, 使得 PSO 算法能够搜索较大的区

域. 随着搜索过程的深入, w 逐渐减少, 开始精细搜索. 惯性权重 w 的初始值和最终值分别为 0.4 和 1.4.

3.1.3 局部搜索

从晶圆排列中取出一个晶圆, 将其插入另外一个位置, 得到该排列的一个邻居, 所有的这些邻居构成该排列的插入邻域. 本文采用文献 [6] 提出的插入邻域局部搜索算法, 具体步骤如下:

Step 1: 令 $k=1$, 从当前排列中取出第 k 个晶圆, 将该晶圆分别插入到另外 $n-1$ 个位置, 并分别按式 (6)∼(13) 计算晶圆的加工完成时间.

Step 2: 令 $k=k+1$, 若 $k \leqslant n$, 则返回 Step 1.

Step 3: 如果邻域内的最优排列优于当前排列, 则用邻域最优排列置换当前排列, 并返回 Step 1; 否则算法结束.

为了方便说明, 本文将有局部搜索算法的 PSO 称为改进 PSO, 反之称为基本 PSO.

3.2 基于改进 PSO 的在线调度方法

Step 1: 对调度问题进行分析, 初始化所有粒子, 在允许范围内随机设置粒子的初始速度和位置, 每个粒子的最优位置设置为其初始位置, 各粒子最优位置中的最好值设为群的最优位置.

Step 2: 由粒子位置矢量得到晶圆的加工顺序, 按加工顺序选出进入集束型装备的晶圆.

Step 3: 由式 (6)∼(13) 计算选出晶圆的加工完成时间 t_{N+1}^{s}, 并作为粒子的适应值.

Step 4: 更新允许区间集合 CM_n 和 CT_n.

Step 5: 完成所有晶圆的加工, 转至 Step 6; 否则转至 Step 3.

Step 6: 根据所有晶圆的完成时间, 计算适应值, 对粒子群的粒子进行插入邻域局部搜索, 确定新的个体最优粒子, 局部搜索完毕后确定全局最优粒子.

Step 7: 对于所有粒子, 利用标准粒子群算法的位置和速度公式, 产生新粒子群的速度和位置.

Step 8: 若满足中止条件 (最大迭代次数或足够好的适应值), 则输出最优晶圆加工顺序和完工时间; 否则, 转至 Step 3.

4 算　例

算例 1　为了验证在线调度方法的有效性, 首先给出小规模的在线调度问题. 临时晶圆的加工数为 3, 每个晶圆经 4 个工位完成加工. 机器和机械手的允许加工区间和运输区间为

$$CM_1=\{[26,120],[156,+\infty]\},$$
$$CM_2=\{[0,16],[40,82],[106,+\infty]\},$$
$$CM_3=\{[14,28],[70,158],[182,+\infty]\},$$
$$CM_4=\{[0,22],[50,104],[132,+\infty]\},$$
$$CT_1=\{[0,14],[16,20],[22,26],[28,40],[42,50],$$
$$[52,70],[72,82],[84,102],[104,118],$$
$$[120,132],[134,156],[158,182],[184,+\infty]\}.$$

临时晶圆的 [加工时间, 滞留时间] 集合为

晶圆1 : $\{[16,10],[24,30],[24,30],[24,50]\}$;

晶圆2 : $\{[10,12],[28,20],[22,50],[30,40]\}$;

晶圆3 : $\{[18,15],[16,24],[20,40],[30,40]\}$.

下面以临时晶圆 1 为例说明内层算法的计算过程: 取 $CM_1=[26,120]$, 根据式 (6) 和 (7) 计算 CS_1 和 CF_1, 有

$$CS_1=$$
$$\big(\{[L_{1,k}^{m}-T_1^{TR},U_{1,k}^{m}-T_1^{P}]\}\bigcap CT_1\big)\oplus[T_1^{TR},0]=$$
$$\big\{[24,104]\bigcap CT_1\big\}\oplus[2,0]=$$
$$\{[26,26],[30,40],[44,50],[54,70],[74,82],[86,102]\}.$$
$$CF_1=$$
$$\{[L_{1,k}^{m}+T_1^{P},U_{1,k}^{m}]\}\bigcap(CS_1\oplus[T_1^{P},T_1^{P}+T_1^{D}])=$$
$$\{[42,120]\}\bigcap(CS_1\oplus[16,26])=\{[42,120]\}.$$

当 $n=2,3,4,5$ 时, 根据式 (8) 和 (9) 计算 CS_n 和 CF_n, 有

$$CS_2=\big((CM_2\oplus[0,-24])\bigcap(CF_1\oplus$$
$$[2,2])\big)\oplus[-2,0]\bigcap CT_2\oplus[2,0]=$$
$$\{[44,50],[54,58],[106,118],[122,122]\},$$
$$CF_2=(CM_2\oplus[24,0])\bigcap(CS_2\oplus[24,54])=$$
$$\{[68,82],[130,176]\},$$
$$CS_3=\big((CM_3\oplus[0,-24])\bigcap(CF_2\oplus[2,2])\big)\oplus$$
$$[-2,0]\bigcap CT_3\oplus[2,0]=$$
$$\{[70,70],[74,82],[132,132]\},$$
$$CF_3=(CM_3\oplus[24,0])\bigcap(CS_3\oplus[24,54])=$$
$$\{[94,136],[156,158]\},$$
$$CS_4=\big((CM_4\oplus[0,-24])\bigcap(CF_3\oplus[2,2])\big)\oplus$$
$$[-2,0]\bigcap CT_4\oplus[2,0]=$$
$$\{[132,132],[136,138],[160,160]\},$$
$$CF_4=(CM_4\oplus[24,0])\bigcap(CS_4\oplus[24,74])=$$
$$\{[156,234]\},$$
$$CS_5=\big((CM_5\oplus[0,-24])\bigcap(CF_4\oplus[2,2])\big)\oplus$$
$$[-2,0]\bigcap CT_5\oplus[2,0]=$$
$$\{[160,182],[186,236]\}.$$

当 $n=5,4,3,2,1$ 时, 根据式 (11)~(13) 计算 t_n^s 和 t_n^f, 有

$$t_5^s=\min(CS_5)=160,\ t_4^f=t_5^s-T_5^{TR}=158,$$
$$t_4^s=\max(\{[132,132]\})=132,$$
$$t_3^f=t_4^s-T_4^{TR}=130,$$
$$t_3^s=\max(\{[76,82]\})=82,\ t_2^f=t_3^s-T_3^{TR}=80,$$
$$t_2^s=\max(\{[44,50],[54,56]\})=56,$$
$$t_1^f=t_2^s-T_2^{TR}=54,\ t_1^s=\max(\{[30,38]\})=38.$$

因此, 临时晶圆 1 在每个工位 [开始时间,结束时间] 的集合为 $\{[38,54],[56,80],[82,130],[132,158]\}$.

使用在线方法对该算例用 Matlab 7.0 进行计算机仿真, 初始种群大小根据问题的规模设定, 当临时晶圆规模较小时, 可选取较少的粒子进行迭代. PSO 参数设置如下: 种群规模为 10, 迭代次数为 10, 学习系数 $c_1=c_2=2$. 仿真结果显示, 最短生产周期为 273 s. 仿真实验表明, 10 个初始粒子在迭代 10 次之后, 均得到生产周期为 273 s 的仿真结果.

下面给出其中 1 个最优调度结果: 晶圆加工顺序为 [2,3,1]. 晶圆 2 在 4 个工位的加工时间段分别为 [26, 38], [40,68], [70,130], [132,162], 使用机械手的时间段分别为 [37,40], [67,70], [129,132]; 晶圆 3 在 4 个工位的加工时间段分别为 [156,174], [176,192], [194, 214], [216,246], 使用机械手的时间段分别为 [173, 176], [191,194], [213,216]; 晶圆 1 在 4 个工位的加工时间段分别为 [174,195], [197,221], [223,247], [249,273], 使用机械手的时间段分别为 [194,197], [220,223], [246,249].

算例 2 本例给出较大规模的在线调度问题: 临时晶圆的加工数为 10, 晶圆经过 6 个工位, 每个工位存在多个相同加工能力的并行设备. 算法需要的机器和机械手的可加工区间和运输区间分别为

$$CM_1=\{[0,27],[2,25],[6,30],[7,33],[8,35],[10,36],[13,39],[17,45],[21,49],[30,57],[39,+\infty),[40,+\infty),[45,+\infty),[49,+\infty)\};$$
$$CM_2=\{[2,26],[3,27],[4,29],[7,30],[9,34],[12,39],[15,45],[20,46],[27,50],[32,67],[42,69],[43,+\infty),[50,+\infty),[53,+\infty),[55,+\infty)\};$$
$$CM_3=\{[4,30],[5,29],[5,33],[7,39],[11,36],[12,37],[18,49],[30,+\infty),[35,+\infty),[39,+\infty),[42,+\infty),[44,+\infty),[47,+\infty)\};$$
$$CM_4=\{[3,27],[4,27],[4,29],[7,36],[8,32],[10,38],[14,39],[20,46],[29,56],[32,59],[42,79],[55,+\infty),[57,+\infty),[60,+\infty),[62,+\infty)\};$$
$$CM_5=\{[4,28],[5,34],[7,34],[9,32],[10,36],[13,39],[18,46],[21,52],[31,59],[40,67],[49,+\infty),[53,+\infty),[58,+\infty),[60,+\infty)\};$$
$$CM_6=\{[7,39],[8,33],[8,38],[9,35],[13,39],[21,48],[24,50],[30,57],[32,60],[43,77],[53,+\infty),[60,+\infty),[63,+\infty),[69,+\infty),[80,+\infty)\}.$$

机械手可以使用的时间区间为

$$CT_1=\{[6,15],[13,24],[15,30],[26,37],[30,50],[40,55],[51,67],[52,66],[65,77],[72,88],[80,90],[85,99],[100,120],[101,+\infty),[108,+\infty),[120,+\infty),[125,+\infty)\}.$$

临时晶圆的 [加工时间, 滞留时间] 集合为

晶圆1 $\{[20,40],[30,40],[30,40],[20,30],[40,50],[30,50]\}$;

晶圆2 $\{[20,30],[30,50],[20,30],[20,30],[40,50],[40,60]\}$;

晶圆3 $\{[20,40],[40,50],[20,30],[20,30],[30,40],[20,40]\}$;

晶圆4 $\{[30,50],[20,40],[30,40],[20,30],[30,40],[30,50]\}$;

晶圆5 $\{[40,50],[20,40],[20,30],[20,40],[30,50],[30,40]\}$;

晶圆6 $\{[20,40],[30,50],[40,50],[20,40],[40,60],[30,40]\}$;

晶圆7 $\{[40,50],[30,50],[20,40],[20,30],[20,30],[20,40]\}$;

晶圆8 $\{[30,40],[30,40],[30,40],[20,40],[30,50],[30,40]\}$;

晶圆9 $\{[30,50],[30,40],[20,30],[30,40],[20,40],[30,40]\}$;

晶圆10 $\{[40,50],[30,40],[20,30],[20,40],[30,40],[30,50]\}$.

对于算例 2, 种群规模设为 20, 迭代次数为 50, 多次仿真结果显示, 最短生产周期为 245 s. 表 1 记录了取得最优调度方案时的晶圆加工顺序以及在每个机器的开始时间.

表 1　10 晶圆调度的最优方案

晶圆顺序	工位的加工开始时间					
	1	2	3	4	5	6
1	6	32	64	96	118	160
2	17	43	75	97	119	161
8	39	71	103	135	157	189
5	40	82	104	126	148	180
10	45	87	119	141	163	195
6	30	55	87	129	159	201
9	49	81	114	146	178	201
4	69	101	123	155	187	219
3	79	103	145	167	193	225
7	80	122	154	176	199	225

算例 3　半导体制造刻蚀设备为一类典型的集束型装备. 本文采用国内某刻蚀机厂商的实际数据进行测试和验证. 该刻蚀设备包括 4 台机器, 1 个机械手和 2 个分别起输入和输出作用的装载室. 刻蚀工艺过程包括 3 个步骤: 刻蚀加工腔 A 或 B(氢氟酸或氨氟化物)→ 除酸清洗腔 → 干燥腔. 表 2 列举了晶圆类型 A 和 B 的加工参数.

表 2　刻蚀机的加工参数　　s

晶圆类型 A			晶圆类型 B		
工位	加工时间	滞留时间	工位	加工时间	滞留时间
刻蚀腔 A/B	35	15	刻蚀腔A	50	10
清洗腔	30	20	清洗腔	30	20
干燥腔	30	60	干燥腔	30	60

临时有一批待加工晶圆 25 片, A 和 B 两种类型晶圆的混合比为 13:12. 经过多次仿真结果显示, 最优的加工顺序为 [A B A B A B A B A B A B A B A B A B A B A B A B A], 加工的最短生产周期为 920 s, 图 3 是调度的甘特图.

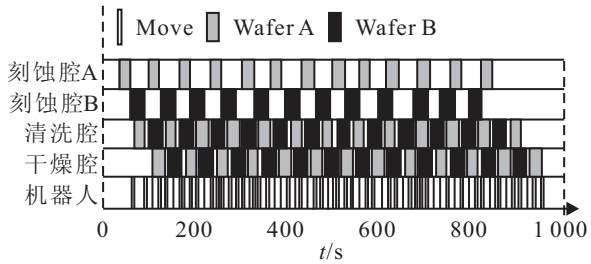

图 3　刻蚀机调度甘特图

分别用基本 PSO 和改进 PSO 对 3 个算例随机计算 10 次, 得到最优调度方案的平均运行时间如表 3 所示. 由表 3 可知, 改进 PSO 所得结果与标准 PSO 相比有较大幅度提高, 表明邻域搜索算法是提高算法性能的有效途径, 并且算例 3 的平均运行时间 (28.30 s) 达到了国内某刻蚀机厂商的性能要求 (小于 30 s).

表 3　算例的平均运行时间　　s

	算例 1	算例 2	算例 3
基于基本 PSO 的两层算法	5.36	18.18	35.42
基于改进 PSO 的两层算法	5.11	15.27	28.30

5　结　论

本文提出了临时晶圆到达时在线调度问题的两层调度方法. 外层算法采用 PSO 优化过程求得临时晶圆的最优调度序列; 内层算法分为前向和后向递归两阶段, 用于寻找临时晶圆的最优开始加工时间. 通过仿真实验表明, 该方法在问题规模较大时, 仍能在使用较少计算代价的情况下迅速得到最优调度方案. 另外, 采用国内某刻蚀机厂商的实际数据进行评测, 表明了方法的有效性和实际应用价值.

在线调度方法是处理不确定性问题的有效算法. 下一步的工作是利用该方法处理设备故障和加工时间不确定等意外事件.

参考文献(References)

[1] Lin K K, Spanos C J. Statistical equipment modeling for VLSI manufacturing: An application for LPCVD[J]. IEEE Trans on Semiconductor Manufacturing, 1990, 3: 216-229.

[2] Moslehi M, Chapman R A, Wong M. Single-wafer integrated semiconductor device processing[J]. IEEE Trans on Electron Devices, 1992, 39: 4-32.

[3] 朱夏, 李小平, 王茜. 基于目标增量的无等待流水调度快速迭代贪婪算法[J]. 计算机学报, 2009, 31(1): 132-141.
(Zhu X, Li X P, Wang X. Objective increment based iterative greedy heuristic for no-wait flowshops with total flowtime minimization[J]. Chinese J of Computers, 2009, 31(1): 132-141.)

[4] 李建祥, 唐立新, 吴会江. 带运输和设置时间的无等待并行流水车间调度问题研究[J]. 系统工程理论与实践, 2006, 26(1): 18-24.
(Li J X, Tang L X, Wu H J. No-wait parallel flowshop scheduling with transfer and setup times[J]. Systems Engineering-Theory and Practice, 2006, 26(1): 18-24.)

[5] 轩华, 唐立新. 实时无等待 HFS 调度的一种拉格朗日松弛算法[J]. 控制与决策, 2006, 21(4): 376-380.
(Xuan H, Tang L X. Lagrangian relaxation algorithm for real-time hybrid flowshop scheduling with no-wait in process[J]. Control and Decision, 2006, 21(4): 376-380.)

[6] 潘全科, 王文宏, 朱剑英. 解决无等待流水车间调度问题的离散粒子群优化算法[J]. 计算机集成制造系统, 2007, 13(6): 1127-1130.
(Pang Q K, Wang W H, Zhu J Y. Modified discrete particle swarm optimization algorithm for no-wait flow shop problem[J]. Computer Integrated Manufacturing Systems, 2007, 13(6): 1127-1130.)

(下转第48页)