文章编号: 1001-0920(2006)04-0425-05

大规模流水线调度的瓶颈分解算法研究

左　燕, 谷寒雨, 席裕庚
(上海交通大学 自动化研究所, 上海 200030)

摘　要: 为了克服大规模流水线调度问题的计算复杂度, 提出一种瓶颈分解启发式算法. 利用瓶颈特性将流水线分解为瓶颈机和非瓶颈机, 对瓶颈机建立带有到达时间和传递时间约束的单机调度模型, 并优化求解, 而在非瓶颈机上则采用简单的分派规则, 通过不断修正瓶颈机上工件的到达时间和传递时间来协调瓶颈机与非瓶颈机之间的关联. 仿真结果验证了算法的有效性.

关键词: 流水线; 瓶颈; 分解算法; 到达时间; 传递时间
中图分类号: TP18　　**文献标识码**: A

Bottleneck-based Decomposition Algorithm for Large-scale Flow Shop Scheduling Problems

ZUO Yan, GU Han-yu, XI Yu-geng
(Institute of Automation, Shanghai Jiaotong University, Shanghai 200030, China. Correspondent: ZUO Yan, E-mail: leftswallow@sjtu.edu.cn)

Abstract: A bottleneck-based decomposition heuristic algorithm is proposed for dealing with computational complexity in large-scale flow shop scheduling problems. Based on the characteristic of bottleneck, the flow shop is decomposed into bottleneck and non-bottleneck machines. A single-machine scheduling problem with release time and delivery time is built for the bottleneck machine, which is solved optimally, while simple priority rules are used for scheduling the non-bottlenecks. The correlations between the bottleneck and non-bottleneck machines are coordinated by adjusting the release time and delivery time of jobs on the bottleneck machine. Simulation results show that the heuristic algorithm can obtain better solution in quite short time and be suitable to large-scale problems.

Key words: Flow shop; Bottleneck; Decomposition algorithm; Release time; Delivery time

1 引　　言

流水线(Flow shop)调度问题是一类典型的组合优化问题. 当机器个数$m \geq 3$时, 其计算复杂度可归结为NP-hard[1]. 因此, 大规模流水线调度问题的精确求解较为困难, 大量的研究工作主要集中在近似解的获得上.

流水线调度问题的启发式算法主要包括构造式算法和改进式算法. 移动瓶颈方法是一种非常有效的构造式算法[2, 3], 由于所有的子问题均优化求解, 计算时间比较长. 而改进算法, 如模拟退火和遗传算法, 其搜索空间随问题规模的增大而增大, 同样存在计算困难[4, 5]. 文献[3]利用机器之间的负荷差异, 在移动瓶颈方法的框架上提出了一种瓶颈分解算法, 它可求解大规模车间调度问题; 文献[6]针对混合流水线调度问题, 提出了另一种瓶颈迭代算法; 文献[7]基于约束理论思想, 针对流水线调度问题提出了相应的瓶颈规划调度算法. 不同问题算法各异, 但其本质都是通过求解瓶颈机的调度问题来降低原问题的计算复杂度.

本文从分解协调的角度研究瓶颈分解方法, 侧

收稿日期: 2005-01-26; **修回日期**: 2005-05-26.
基金项目: 国家自然科学基金项目(60274013, 60474002); 上海市科技发展基金项目(04DZ11008).
作者简介: 左燕(1980—), 女, 江西井岗山人, 博士生, 从事生产规划调度、系统优化方法等研究; 席裕庚(1946—), 男, 上海人, 教授, 博士生导师, 从事复杂系统控制理论、智能机器人等研究.

重于大规模问题的分解过程和机器之间的协调方式. 针对目标函数为总拖期时间最小的经典流水线调度问题, 给出了一个具体的瓶颈分解算法. 该算法可在较快的时间内得到一个满意的调度, 适用于大规模静态流水线调度问题.

2 问题描述

经典流水线调度问题 $F_m \| \sum T_i$ 可描述如下: n 个待加工工件 $J_1, \cdots, J_n$ 按照相同的工艺路径依次通过 m 台机器 $M_1, \cdots, M_m$, 假设机器按工艺路径依次排列. 每个工件 J_i 都包括 m 道工序 $O_{i1}, O_{i2}, \cdots, O_{im}$, 工件 J_i 在机器 M_j 上的加工时间为 $p_{i,j}$.

假设 1 每个工件在每台机器上只能加工一次;

假设 2 任一时刻, 每台机器最多加工一个工件;

假设 3 每个工件按照相同的工艺路径依次加工;

假设 4 工件一旦开始加工则不允许中断.

调度的目标在于确定每台机器上工件的加工顺序, 使得工件的总拖期(Tardiness)时间最小. 建立整数规划模型(P)如下[1]:

$$
\begin{aligned}
(\mathrm{P}) \quad & \min_{\{x_{i,j,k},\, t_{i,k}\}} \sum_{i=1}^{n} T_i, && (1a)\\
& \text{s.t.}\\
& t_{i,k} - t_{i,k-1} \geq p_{i,k-1},\ i = 1, \cdots, n,\\
& k = 2, \cdots, m; && (1b)\\
& t_{i,1} \geq 0,\ i = 1, \cdots, n; && (1c)\\
& t_{j,k} - t_{i,k} + M(1 - x_{i,j,k}) \geq p_{i,k},\\
& i = 1, \cdots, n-1,\ j = i+1, \cdots, n,\\
& k = 1, \cdots, m; && (1d)\\
& t_{i,k} - t_{j,k} + M x_{i,j,k} \geq p_{j,k},\\
& i = 1, \cdots, n-1,\\
& j = i+1, \cdots, n,\ k = 1, \cdots, m; && (1e)\\
& T_i = \max\{0, t_{i,m} + p_{i,m} - d_i\},\\
& i = 1, \cdots, n; && (1f)\\
& x_{i,j,k} \in \{0, 1\},\ i, j = 1, \cdots, n,\\
& i \neq j,\ k = 1, \cdots, m. && (1g)
\end{aligned}
$$

其中: 决策变量 $x_{i,j,k}$ 为二进制变量, 表示机器 k 上工件 i 和工件 j 的加工顺序. 如果机器 k 上工件 i 在工件 j 前加工, 则 $x_{i,j,k} = 1$; 否则 $x_{i,j,k} = 0$. $t_{i,k}$ 和 $p_{i,k}$ 分别为工件 i 在机器 k 上的开工时间和加工时间. d_i 为工件 i 的交货期限. M 为一充分大的正整数. 工件 i 的拖期时间 T_i 按式(1f)计算. 式(1a)为目标函数, 希望工件的总拖期时间最小; 式(1b)表示工件按照相同的工艺路径依次加工; 式(1c)表示任一工件不能在其到达时间之前开始加工; 式(1d)和(1e)为机器的能力约束, 保证任一时刻每台机器只能加工一个工件.

流水线调度问题 $F_m \| \sum T_i$ 的计算复杂度可归结为强NP-hard[1]. 由整数规模模型(P)可知, 变量和约束的个数均为 $m \cdot n^2$ 个. 如果采用分枝定界算法精确求解, 其计算量与问题的规模呈指数关系, 存在计算困难. 因此, 希望采用一种好的启发式算法在较短的时间内获得满意的调度.

3 瓶颈分解算法研究

能力均衡的生产线容易受到加工产品变化或随机扰动的影响而产生瓶颈移动, 从而使得整条生产线不易于管理, 因此实际生产过程中大多数生产线都设计为能力不均衡. 在订单信息和加工环境确定的条件下, 生产线存在一个稳定的瓶颈. 约束理论指出, 瓶颈主导着整条生产线的性能[6], 因此可通过求解瓶颈机的调度问题来降低原问题的计算复杂度.

瓶颈分解算法是一种特殊的机器层分解算法. 它利用机器负荷差异将流水线上的机器分解为瓶颈机和非瓶颈机. 松弛非瓶颈机的能力约束, 建立瓶颈机的调度子模型并精确求解. 非瓶颈机的调度围绕着瓶颈机的调度而建立, 一般采用有效的分派规则. 由于瓶颈机调度模型是在松弛机器能力约束的情况下获得的, 最终得到的调度可能是不可行的, 因此需要不断地协调瓶颈机与非瓶颈机之间的关联, 直到获得一个好的可行调度.

3.1 瓶颈分解过程

经典流水生产线上每个工件的工艺路径固定且相同, 因此加工负荷越大的机器越容易发生拥塞现象而成为生产线的瓶颈. 本文选择加工负荷最大的机器 M_b 为瓶颈机, 工件的加工时间满足

$$\sum_{i=1}^{n} p_{i,b} > \sum_{i=1}^{n} p_{i,j},\ j = 1, \cdots, m,\ j \neq b. \qquad (2)$$

式(2)给出了一个较为鲁棒的瓶颈机选择规则. 即使在动态不确定情况, 如果某个工件的加工时间发生变化, 只要工件的加工时间满足条件(2), 则该机器仍为瓶颈机.

流水线上剩余的机器称为非瓶颈机. 由于流水线特殊的工艺路径, 机器 M_b 上每道工序的前序都在上游机(瓶颈机之前的所有非瓶颈机器)上加工, 其后序都在下游机(瓶颈机之后的所有非瓶颈机器)上加工, 整个流水线分解为上游机、瓶颈机和下游机.

假设非瓶颈机的加工能力无限大, 则非瓶颈机的机器能力约束不再成立, 约束(1d)和(1e)可简化为

$$t_{j,b} - t_{i,b} + M(1 - x_{i,j,b}) \geq p_{i,b},$$
$$i = 1, \cdots, n-1,\ j = i+1, \cdots, n; \quad (3a)$$
$$t_{i,b} - t_{j,b} + M x_{i,j,b} \geq p_{j,b},$$
$$i = 1, \cdots, n-1,\ j = i+1, \cdots, n. \quad (3b)$$

工件 i 在瓶颈机 M_b 上的到达时间和剩余加工时间取决于它在上游机和下游机上的加工. 因为非瓶颈机的加工能力无限大, 所以有

$$t_{i,b} = t_{i,b-1} + p_{i,b-1} = t_{i,b-2} + p_{i,b-2} + p_{i,b-1} = \cdots = t_{i,1} + \sum_{j=1}^{b-1} p_{i,j}, \quad (4a)$$

$$t_{i,m} = t_{i,m-1} + p_{i,m-1} = t_{i,m-2} + p_{i,m-2} + p_{i,m-1} = \cdots = t_{i,b+1} + \sum_{j=b+1}^{m-1} p_{i,j}. \quad (4b)$$

由式(4a), (4b), (1c) 和(1f) 可得

$$t_{i,b} \geq r_i,\ i = 1, \cdots, n; \quad (5a)$$
$$T_i = \max\{0, t_{i,b} + p_{i,b} - d_i'\},\ i = 1, \cdots, n. \quad (5b)$$

其中: $r_i = \sum_{j=1}^{b-1} p_{i,j}$, $q_i = \sum_{j=b+1}^{m} p_{i,j}$, $d_i' = d_i - q_i$. 工件 i 在上游机 (或下游机) 的流动时间用参数 r_i(或 q_i) 表示, 它是工件 i 在上游机(下游机) 的加工时间之和. 式(5a) 和(5b) 暗含了工件 i 经过 r_i 个单位的时间迟延到达瓶颈机, 它在瓶颈机上加工完后再经过 q_i 个单位的时间迟延传递至客户. r_i, q_i 和 d_i' 分别表示工件 i 在瓶颈机 M_b 上的到达时间、剩余加工时间和局部交货期限. 流水线调度模型(P) 可松弛为瓶颈机 M_b 上的调度模型(P_b) 如下:

(P_b)　(1a),

s.t.　(3a), (3b), (5a), (5b), (1g).

其中决策变量 $x_{i,j,b}$ 和 $t_{i,b}$ 定义同上. 调度模型(P_b) 的变量和约束明显减少, 计算复杂度降低.

引理 1　令 Z 和 Z_b 分别为问题(P) 和(P_b) 的最优解, 则有 $Z_b \leq Z$.

证明　令 S 和 S_b 分别为问题(P) 和(P_b) 解的可行域, 则有 $S \subseteq S_b$. 由于目标函数相同且为最小化问题, 有 $Z_b \leq Z$, 即瓶颈机调度问题的最优解是原问题最优解的下界. □

3.2　瓶颈机与非瓶颈机的调度

针对瓶颈机调度模型(P_b), 可以采用精确算法或启发式算法进行求解. 假设瓶颈机 M_b 上工件的排序为 σ_b, $\sigma_b(i)$ 表示瓶颈机 M_b 上第 i 个加工的工件, 添加瓶颈机 M_b 的能力约束

$$t^*_{\sigma_b(i),b} - t^*_{\sigma_b(i-1),b} \geq p_{\sigma_b(i-1),b},\ i = 2, \cdots, n. \quad (6)$$

其中 $t^*_{\sigma_b(i),b}$ 为工件 $\sigma_b(i)$ 在瓶颈机 M_b 上的开工时间.

一旦获得瓶颈机上的调度, 则非瓶颈机上的调度围绕着瓶颈机上的调度而建立. 瓶颈机的调度为上游机上的每个工件提供了最迟完工时间, 同时为下游机上的每个工件提供了最早开工时间. 为了保证调度的可行性(即非瓶颈机的调度不破坏瓶颈机的调度), 设置工件 i 在机器 M_{b-1} 上的交货期限等于它在机器 M_b 上的开工时间, 工件 i 在机器 M_{b+1} 上的到达时间等于它在机器 M_b 上的完工时间, 有

$$d_{i,b-1} = t^*_{i,b},\ r_{i,b+1} = C^*_{i,b},\ i = 1, \cdots, n. \quad (7)$$

非瓶颈机上未建立详细的数学规模模型, 它根据工件和机器的参数采用有效的分派规则对每个工序设置优先级, 当机器可用时, 从已到达且未调度的工件集中选择优先级最高的工件. 具体分派规则参见文献[8].

3.3　瓶颈机与非瓶颈机之间的协调

由于瓶颈机调度模型是在松弛非瓶颈机能力约束的情况下获得的, 最终得到的调度不一定可行. 因此需要考虑非瓶颈机上工件的加工状态, 对所得到的解进行调整. 瓶颈机上工件 i 的到达时间 r_i 和剩余加工时间 q_i 取决于工件 i 在上游机和下游机的开工时间, 因此瓶颈机与非瓶颈机之间的协调体现在瓶颈机上工件的到达时间和传递时间参数的调整上. 类似文献[6] 的参数修正方法, 给出瓶颈机上工件到达时间和传递时间的调整规则如下:

$$r_i = \begin{cases} \max\left\{\sum_{j=1}^{b-1} p_{i,j},\ \text{if } C_{i,b-1} > t^*_{i,b}\right\}; \\ \max\left\{\sum_{j=1}^{b-1} p_{i,j},\ r_i - (d_i - C_{i,m})\right\},\ \text{if } C_{i,m} < d_i; \\ r_i, \text{otherwise.} \end{cases} \quad (8a)$$

$$q_i = \begin{cases} \max\left\{\sum_{j=b+1}^{m} p_{i,j},\ q_i + (C_{i,m} - d_i)\right\},\ \text{if } C_{i,m} > d_i; \\ q_i, \text{otherwise.} \end{cases} \quad (8b)$$

当上游机调度不可行时, 则至少存在一个工件 i 无法在瓶颈机规划的开工时间之前到达, 即 $C_{i,b-1} > t^*_{i,b}$, 修正瓶颈机上工件 i 的到达时间为 $r_i = C_{i,b-1}$; 当下游机调度完获得最终调度, 工件 i 不能在预定的交货期限内完工, 即 $C_{im} > d_i$, 则希望该工件更早调度, 提高瓶颈机上工件 i 的传递时间为 $q_i = q_i + (C_{i,m} - d_i)$; 当工件 i 在预定的交货期限之前完工, 即 $C_{i,m} < d_i$, 则降低瓶颈机的到达时间为 $r_i = r_i - (d_i - C_{im})$.

针对目标函数为总拖期时间最小的流水线调度

问题, 给出一个具体的瓶颈分解算法如下:

Step 1: 辨识瓶颈. 选择加工负荷最大的机器为瓶颈机M_b.

Step 2: 参数估计. 按 3.1 节方法估计瓶颈机上工件 i 的初始到达时间 r_i, 传递时间 q_i 和交货期限 d_i'.

Step 3: 瓶颈机的调度. 对瓶颈机M_b建立子问题(P_b), 采用精确算法或启发式算法求解.

Step 4: 上游机的调度. 对上游非瓶颈机采用分派规则调度. 如果上游机得到的调度不可行, 则按式(8a) 调整工件的到达时间, 转 Step 3; 否则转 Step 5.

Step 5: 下游机的调度. 对下游非瓶颈机采用分派规则调度. 如果算法无改进, 则停止; 否则当存在工件拖期, 按式(8b) 调整该工件的传递时间, 转 Step 3; 若存在工件早到, 则按式(8a) 调整该工件的到达时间, 转 Step 3.

4 仿真研究

为了测试瓶颈分解算法在调度问题$F \| \sum T_i$中的性能, 按文献[5] 的方式随机生成大量测试数据, 实验参数设计如表 1 所示.

表 1 测试数据生成参数表

问 题	参 数
机器个数 m	10, 20, 30
交货期范围 R	0.8, 1.8
瓶颈机位置 b	第 1 个 1/4, 第 2 个 1/4, 第 3 个 1/4, 最后一个 1/4
工件个数 n	50, 100, 200
交货期滞后程度 T	0.1, 0.5
瓶颈机与非瓶颈负荷差异 L	0.3(高), 0.5(中), 0.8(低)

所有工件均在零时刻到达, 瓶颈机上每个工件的加工时间服从[1, 50] 均匀分布, 非瓶颈机上每个工件的加工时间服从$[1 \cdot L, 50 \cdot L]$均匀分布, 工件 i 的交货期限 d_i 服从$[P'(1 - T - R/2), P'(1 - T + R/2)]$均匀分布, 其中$P'$是最大完工时间$C_{\max}$的下界. 所有参数组合生成 432 种类型的问题, 每种类型分别生成 10 组数据, 共 4 320 组测试数据.

由于瓶颈机上的调度问题 $1|r|\sum T_i$ 为强 NP-hard, 目前还没有一种高效的分枝定界算法可以在较短的时间内获得最优解. 因此, 本文对瓶颈机采用MOD 规则调度, 上游非瓶颈机和下游非瓶颈机也采用MOD 规则, 记作BOT. 将瓶颈分解算法与另一种启发式算法列表调度进行比较[6]. 其中分派规则分别为[7]: 先到先加工(FCFS), 最短加工时间先加工(SPT), 最早交货期限先加工(EDD), 最早工序交货期限先加工(MOD), 拖期费用最大者先加工(ATC) 和改进型PSK 规则(MPSK). 采用以下两个评价指标[2]:

令 $T(H, I)$ 为对给定实例 I 下采用启发式算法 H 求得的 $\sum T_i$ 值, 且

$$\mathrm{BEST}(I) = \min_H \{T(H, T)\};$$

定义一组满足相同特征的问题 S, 对每类问题 S 计算相对性能指标为

$$\rho(H, S) = \frac{\sum_S T(H, I)}{\sum_S \mathrm{BEST}(I)}.$$

为了比较不同方法的计算时间, 计算一类问题所有实例的CPU 平均计算时间. 对问题S 采用启发式方法H 得到的平均 CPU 计算时间为ACT(H, S). 所有算法均采用 C 语言编程, 运行在 Pentium Ⅳ, CPU 为 2 GHz 的机器上, 测试结果如表 2～ 表 5 所示.

表 2 不同加工负荷差异下算法/规则的性能

问题 S	启发式方法 H 的相对性能指标 $\rho(H, S)$						
	BOT	FCFS	SPT	EDD	MOD	ATC	MPSK
L = 0.3	1.000	1.359	1.213	1.338	1.211	1.280	1.219
L = 0.5	1.000	1.545	1.348	1.432	1.282	1.373	1.323
L = 0.8	1.000	3.254	2.665	1.756	1.507	1.648	2.507

表 3 瓶颈机位于生产线不同位置下算法/规则的性能

问题 S	启发式方法 H 的相对性能指标 $\rho(H, S)$						
	BOT	FCFS	SPT	EDD	MOD	ATC	MPSK
第 1 个 1/4	1.000	2.020	1.719	1.505	1.330	1.432	1.731
第 2 个 1/4	1.000	2.090	1.762	1.515	1.335	1.437	1.718
第 3 个 1/4	1.000	2.121	1.799	1.506	1.330	1.431	1.705
最后 1/4	1.000	1.980	1.688	1.509	1.338	1.435	1.579

表 4 不同交货期限范围 R 和滞后程度 T 下算法/规则的性能

问题 S (R, T)	启发式方法 H 的相对性能指标 $\rho(H, S)$						
	BOT	FCFS	SPT	EDD	MOD	ATC	MPSK
(0.8, 0.1)	1.000	2.355	1.890	1.650	1.496	1.606	2.208
(1.8, 0.5)	1.000	1.450	1.273	1.396	1.250	1.334	1.253
(0.8, 0.1)	1.000	3.095	2.602	1.728	1.427	1.588	2.112
(1.8, 0.5)	1.000	1.311	1.203	1.262	1.161	1.207	1.161

由表 2～ 表 4 可以看出, 瓶颈分解算法优于列表调度算法. 随着瓶颈机与非瓶颈机之间的负荷差异程度增大(即对应L 减小), 瓶颈分解算法和列表调度算法的调度质量均提高. 瓶颈机的位置对瓶颈分解算法的调度质量影响不大, 但影响其计算时间,

表 5　不同问题规模下算法/规则性能比较和瓶颈分解算法 CPU 平均计算时间

问题 S	启发式方法 H 的相对性能指标 $\rho(H,S)$							ACT
(M,N)	BOT	FCFS	SPT	EDD	MOD	ATC	MPSK	BOT
(10 50)	1.000	1.867	1.533	1.584	1.355	1.456	1.493	0.001
(20 50)	1.000	1.497	1.344	1.474	1.326	1.382	1.327	0.002
(30 50)	1.000	1.380	1.282	1.388	1.280	1.314	1.279	0.003
(10 100)	1.000	2.580	2.071	1.616	1.370	1.514	1.914	0.005
(20 100)	1.000	1.712	1.480	1.487	1.330	1.423	1.426	0.008
(30 100)	1.000	1.513	1.361	1.430	1.310	1.379	1.329	0.010
(10 200)	1.000	3.938	3.156	1.612	1.362	1.536	3.065	0.010
(10 200)	1.000	2.134	1.813	1.486	1.318	1.443	1.748	0.030
(10 300)	1.000	1.852	1.638	1.503	1.351	1.458	1.568	0.040

越靠近瓶颈机中间位置, 瓶颈迭代的次数越多, CPU 计算时间越长. 交货期限越松(滞后因子 T 取 0.1), 瓶颈分解算法的性能越好. 由表 5 可知, 瓶颈分解算法的调度性能随着问题规模的增大相对性能指标提高, 而且该算法计算速度非常快(不超过 1s), 适用于大规模调度问题.

5　结　论

本文针对大规模静态流水线调度问题, 提出了一种基于瓶颈的分解协调方法. 该方法通过求解瓶颈机的调度问题, 降低了原问题的计算复杂度. 仿真结果显示, 瓶颈分解算法优于列表调度算法, 它可在很短的时间内获得一个好的调度. 同时关联项的估计对最终调度的影响很大, 到达时间和传递时间的修正方法尚有待改进. 本文研究的对象是典型的流水线调度问题, 当考虑复杂情况, 如重入和批处理机等, 则瓶颈机与非瓶颈机之间的协调方式需要进一步研究.

参考文献(References)

[1] Pinedo M. *Scheduling: Theory, Algorithms, and Systems*[M]. NJ: Prentice Hall, 1995.

[2] Demirkol E, Mehta S V, Uzsoy R. A Computational Study of Shifting Bottleneck Procedures for Shop Scheduling Problems[J]. *J of Heuristic*, 1997, 3(2): 111-137.

[3] Uzsoy R, Wang C S. Performance of Decomposition Procedures for Job Shop Scheduling Problems with Bottleneck Machines[J]. *Int J of Production Research*, 2000, 38(6): 1271-1286.

[4] Sameer H, Chandrasekharan R. Scheduling in Flowshops to Minimize Total Tardiness of Jobs[J]. *Int J of Production Research*, 2004, 42(12): 2289-2301.

[5] 唐立新, 吴亚萍. 混合流水车间调度的遗传下降算法[J]. *自动化学报*, 2002, 28(4): 637-641.
(Tang L X, Wu Y P. A Genetic Descent Algorithm for Hybrid Flow Shop Scheduling[J]. *Acta Automatica Sinica*, 2002, 28(4): 637-641.)

[6] Lee G G, Kim Y D, Choi S W. Bottleneck-focused Scheduling for a Hybrid Flowshop[J]. *Int J of Production Research*, 2004, 42(1): 165-181.

[7] Zhang H Y, Xi Y G, Gu H Y. A Decomposition and Coordination Scheduling Method for Flow-shop Problem Based on TOC[J]. *Acta Automatica Sinica*, 2005, 31(2): 182-187.

[8] Vepsalainen, A P J, Morton T E. Priority Rules for Job Shops with Weighted Tardiness Costs[J]. *Management Science*, 1987, 33(8): 1035-1047.

(上接第 424 页)

[4] Gandhi P, Kassam S A. Analysis of CFAR Processors in Nonhomogeneous Background[J]. *IEEE Trans on AES*, 1988, 24(2): 427-445.

[5] Viswanathan R, Eftekhari A. A Selection and Estimation Test for Multiple Targets in Clutter Detection[J]. *IEEE Trans on AES*, 1992, 28(1): 505-519.

[6] Barkat M, Varshney P K. Adaptive Cell-averaging CFAR Detection in Distributed Sensor Networks[J]. *IEEE Trans on AES*, 1991, 27(2): 424-429.

[7] Ritcey J A. Analysis of the Censored Mean-level Detector[J]. *IEEE Trans on AES*, 1986, 22(4): 443-454.

[8] Uner M K, Varshney P K. Decentralized CFAR Detection Based on Order Ststistics[A]. *In Proc of 36th Midwest Symposium on Circuits and Systems*[C]. Detroit, 1993: 146-149.

[9] Xiu Y H, Norihiko M, Toshihiko N. Divect Evalution of Radar Detection Probabilities[J]. *IEEE Trans on AES*, 1987, 23(4): 418-423.

[10] Hamid A, Viswanathan R. A New Distributed Constant False Alarm Rate Detector[J]. *IEEE Trans on AES*, 1997, 33(1): 85-97.