

考虑需求波动和产能调整成本的混流装配线平衡问题

李金霖^{1,2,3}, 高杰^{1,2,3}, 孙林岩^{1,2,3}

(1. 西安交通大学管理学院, 西安 710049;

2. 过程控制与效率工程教育部重点实验室, 西安 710049;

3. 机械制造系统工程国家重点实验室, 西安 710049)

摘要: 混流装配系统面对的市场需求经常受各种因素影响而上下波动, 当实际需求与预期不同时, 企业需要采取加班等临时措施调整产能。然而现有装配线平衡研究大都是按照确定的预期需求量配置的, 少数考虑随机需求的研究也都忽视了平衡方案后续进行产能调整的成本和难易程度。文章针对需求不确定环境下的混装线平衡决策, 考虑了维持日常产能的人工成本和加班带来的产能调整成本, 建立了数学模型, 提出了一种估计总成本下界的方法并设计了启发式算法。计算实验表明算法能在较短时间内获得较好的结果。

关键词: 混装线; 平衡; 不确定需求; 下界; 启发式算法

中图分类号: 请查阅《中国图书馆分类法》

Balancing mixed model assembly line with capacity adjustment

Li Jinlin^{1,2,3}, Gao Jie^{1,2,3}, Sun Linyan^{1,2,3}

(1. School of Management Xi'an Jiao tong University, Xi'an 710049;

2. The Key Laboratory of the Ministry of Education for Process Control and Efficiency Projects, Xi'an 710049;

3. The State Key Lab for Manufacturing, Xi'an 710049)

Abstract: In a mixed model assembly line, the demands for products are usually uncertain and fluctuating. When the actual demands are larger than expectation, demand or capacity adjustments should be made. The difficulty and cost of these adjustments rely heavily on the initial balancing. To the best of knowledge, the convenience of the line configuration for future adjustment and corresponding cost have not been considered in literature. This paper investigate the line balancing problem in a mixed model assembly system which faces uncertain demand and make use of overtime work to meet unexpected demands. The objective is to minimize the expected labor cost, including cost for normal operation and overtime work. A mathematic model is built and a method is proposed to estimate the cost lower bound, based on which a single pass heuristic is designed. The numerical experiment shows that the algorithm is effective and efficient.

Key words: Mixed model assembly line; Balancing; Uncertain demand; Bound; Heuristic

0 引言

在装配线中, 一件产品的加工过程被分为若干个不可再分的工序, 称为任务(task)。每个任务都有对应的标准作业时间, 并需要按装配工艺流程的先后顺序依次进行, 所有任务的优先序关系可表示为优先顺序图(precedence graph)。装配线上有若干个让工人进行任务作业的区域, 称之为工作台(station)。在装配线设计阶段, 决策者需要在满足优先序关系的前提下将所有任务分配到各工作台, 使得装配线的生产能力满足预期市场需求, 并最优化特定目

基金项目: 国家自然科学基金(71001083); 教育部高等学校博士学科点基金(200806981064)

作者简介: 李金霖 (1986-), 男, 博士生, 研究方向: 生产管理

通信联系人: 高杰 (1978-), 男, 山东蒙阴人, 副教授. 主要研究方向: 生产优化、软计算、现代生产服务系统管理. E-mail: gaoj@mail.xjtu.edu.cn

标(效率、成本或负荷均衡性等), 这就是装配线平衡问题^[1]。

装配线平衡问题的研究始于上世纪 50 年代, 其中简单装配线平衡问题(Simple assembly line balancing problem, SALBP)是所有拓展研究的基础。简单装配线是串行、节拍化的单一产品装配线, 任务的作业时间确定, 不考虑除优先序关系和生产节拍约束之外的其他约束条件。根据是否给定生产节拍 c 和工作台数量 m , SALBP 的研究可分大体为四类: SALBP-F(给定 c 和 m , 求可行解), SALBP-1(给定 c , 最小化 m), SALBP-2(给定 m , 最小化 c), SALBP-E(m 和 c 都待定, 最大化 mc)^[2]。这四类问题都是 NP 问题, 其中 SALBP-1 的相关研究最多^[3-9], 其他三类问题都可以通过重复调用 SALBP-1 的算法求解^[2]。文献中存在多种 SALBP-1 问题下界的计算方法^[5, 7, 8, 10], 其中最早由 Johnson 提出的基于单机排序问题的界在多数算例中表现最好^[2], Scholl 和 Klein 改进了原有算法进一步加强了界^[7], Scholl 等则讨论了各种分配约束(资源约束、不相容任务等)对界的影响并提出了相应的修正算法^[11]。

近年来, 为了满足人们个性化、多样化、定制化的需求, 可以同时生产多种型号产品的混流装配线(简称: 混装线)逐渐代替了传统的单一产品装配线, 广泛用于汽车制造、消费电子、白色家电等行业。在混装线中, 不同产品的工艺流程不完全相同, 不同产品在相同工序所需作业时间也有差异, 因此平衡决策比 SALBP 更复杂。现有的混装线平衡研究大都采用将混装线转化为单一产品装配线的方法^[1], 即将所有产品的优先顺序图融合为联合装配顺序图(joint precedence graph), 根据各产品的预期需求量计算各任务的加权平均作业时间。然而, 现实中的产品需求量并不确定, 而是经常受价格调整、促销活动、突发事件等各种因素的影响而频繁波动。这种情况在代工企业中尤为普遍, 它们接收不同企业的外包订单, 不同订单的下单时间和交货期限各异, 因此生产计划中每天要加工的产品种类和数量都不尽相同。当生产任务与平衡时的预期需求不同时, 企业可能需要采取一些措施进行产能调整, 例如再平衡、调整生产时间(加班)或者外包等^[12]。然而现有装配线平衡研究几乎都是按照确定的预期需求量进行配置的, 忽视了平衡方案后续进行产能调整的成本和难易程度。混装线中产品需求的不确定直接反映为任务作业时间的随机性, 然而考虑随机任务时间的装配线平衡研究大都采用概率满足生产节拍约束来反映随机性的影响^[13, 14], 极少数明确考虑了需求不确定性的混装线平衡研究也仅仅讨论了对各工作台负荷均衡性的影响^[15, 16], 还没有文献考虑企业对不确定需求的应对措施。

针对这一不足, 本文讨论需求不确定环境下的混装线平衡决策, 并考虑初始平衡对后续调整的影响。平衡方案确定后, 企业在不同需求情境下可以通过加班和改变生产节拍来调整产能。决策目标是 minimized 产线的人工成本, 包括所有工人的日常工资和加班工资。具体来说, 决策者要设置合适的工作台数量、确定任务分配方案并决定不同情境下的加班时间, 以最小化期望总成本。本文建立了此问题的数学模型, 提出了一种估计成本下界的方法, 并在此基础上设计了启发式算法, 计算实验表明算法能在较短时间内获得较好的结果。

1 模型建立

1.1 问题描述

考虑设计一条生产 K 种产品的节拍化混装线。受各种因素影响, 存在 S 种可能出现的需求情境, 情境 s 发生概率为 p_s , 对产品 k 的日需求量为 d_{sk} , 所有情境的需求都必须被满足。总装配流程共有 N 个任务, 产品 k 的任务 i 需要 t_{ki} 的作业时间。任务间的先后序关系用联合装配顺序图 $G(N, E)$ 表示, N 和 E 分别是所有节点和所有边的集合, 每个节点都对应一

个任务，每条边 (i, j) 表示任务 i 必须在任务 j 之前加工，图1是一个示例。若任务 i 必须在任务 j 之前加工，则称任务 i 是任务 j 的前驱，特别当 $(i, j) \in E$ 时任务 i 称为任务 j 的直接前驱，相应的任务 j 称为任务 i 的(直接)后驱。

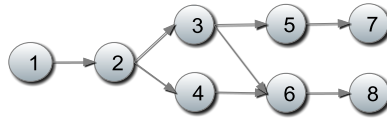


图1 联合装配顺序图示例

每个工作台分配1位工人，每天正常工作时间为 T ，若在此时间内工人不能完成分配的任务则需要加班。假定工人的加班时间必须为最小加班单元 v 的整数倍，且按相关法律规定加班单元数量不超过上限 u_{\max} 。在节拍化生产线中，任何工作台的延误都会影响后续任务的开展，因此一旦加班所有人必须同时加班。此外，将工人正常工作 v 时间的工资标准化为1，每个加班单元应付工资设为 c_o ($c_o > 1$)。其他基本假设为：

1. 串行装配线，不允许并行工作台/支线；
2. 联合装配图给定且不存在回路；
3. 不同产品的相同任务必须分配到同一工作台；
4. 不考虑除任务先后序之外的其他约束。

决策者需要确定设置的工作台数量，每个工作台分配的任务以及各情境下的加班时间，以最小化总成本（所有工人日常工资和加班工资之和）的期望值。若设置的工作台数量过多，则需雇佣很多工人，导致产能维护成本升高，并容易出现产能闲置；若设置的工作台数量过少，则工人经常需要加班，导致产能调整成本升高，因此产能维护成本和调整成本之间的权衡是本研究的一个核心问题。

1.2 数学模型

定义以下符号：

x_{ij} 若任务 i 被分配到工作台 j ，则为1，否则为0

u_s 需求情境 s 下的加班时间

z_j 工作台 j 若启用，则为1，否则为0

根据上节的问题描述，在需求情境 s 下每个工作台的实际作业时间为 $T + u_s v$ ，应支付给每个工人的工资为 $T/v + u_s c_o$ 。本问题的数学模型可表示为：

$$\min \sum_{s=1}^S \sum_{j=1}^N p_s z_j (T/v + u_s c_o) \quad (1)$$

$$\text{s.t.} \quad \sum_{k=1}^K d_{sk} \sum_{i=1}^N t_{ki} x_{ij} \leq z_j T + u_s v, \forall s, j \quad (2)$$

$$\sum_{j=1}^N x_{ij} = 1, \forall i \quad (3)$$

$$\sum_{k=1}^N k x_{ik} \leq \sum_{h=1}^N h x_{jh}, \forall (i, j) \in E \quad (4)$$

$$x_{ij} \leq z_j, \forall i, j \quad (5)$$

$$x_{ij}, z_j \in \{0, 1\}, \forall i, j \quad (6)$$

$$u_s \in \{0, \dots, u_{\max}\}, u_s \text{ int}, \forall s \quad (7)$$

目标函数(1)表示最小化各情境下总成本的期望值。约束(2)表示在任意需求情境下，每个工作台的任务作业时间不应超过该工作台总生产时间。约束(3)表示每个任务都被分配且只能分配到一个工作台。约束(4)表示必须满足任务先后序关系。约束(5)表示任务只能分配到被启用的工作台。约束(6)，(7)是决策变量的取值范围约束。

2 算法设计

120 SALBP-1 是 NP 难题^[7]。当 $S=1, u_s=0$ 时, 本问题转化为 SALBP-1, 因此也是 NP 难题。考虑到问题复杂性, 本节提出一种启发式算法, 基本思想是从第 1 个工作台开始, 穷举所有可以分配到当前工作台的集合, 估计每个分支对应的总成本下界, 选择下界最小的任务集合分配到当前工作台, 再考虑下一个工作台, 如此往复直至所有任务都被分配完。为方便, 引入以下符号:

- 125 D_s 需求情境 s 下各种产品的日需求量之和, $D_s = \sum_{k=1}^K d_{sk}$.
- m 工作台数量, $m = \sum_{j=1}^N z_j$.
- c_s 需求情境 s 的生产节拍, $c_s = (T + u_s v) / D_s$.
- C 由各情境生产节拍组成的向量, $C = (c_1, c_2, \dots, c_S)$. 由于 u_s 取值在 $[0, u_{\max}]$ 之间, 当所有情境 $u_s=0$ 时, 对应的 C 记为 C_{\min} , 当所有情境 $u_s = u_{\max}$ 记为 C_{\max} .
- 130 a_{is} 情境 s 安放任务 i 所有后驱任务所需工作台数量(称为 *tail*, 可为小数)。
- b_{is} 情境 s 安放任务 i 所有前驱任务所需工作台数量(称为 *head*, 可为小数)。
- q_{is} 情境 s 任务 i 的加权作业时间与生产节拍的比值, $q_{is} = \sum_{k=1}^K t_{ik} d_{sk} / D_s / c_s$.

2.1 总成本下界

定义: 对给定 SALBP, 记 $LM(c)$ 为给定 c 对应 SALBP-1 所需工作台数量的下界, 若组合 (m, c) 满足 $m \geq LM(c)$, 则称 (m, c) 在界的意义上可行, 简称“界可行”, 否则称界不可行。

本问题中存在多需求情境, 在每个单独情境中都是 m 和 c_s 都待定的 SALBP 问题。类似的, 定义 (m, C) 组合的可行性: 记 $LM(C)$ 为给定 C 本问题所需工作台数量的下界, 若 $m \geq LM(C)$ 则称组合 (m, C) 界可行, 否则称界不可行。显然, 界可行是存在可行解的必要条件, 若组合 (m, C) 界不可行, 则必然不存在工作台数量为 m 、各情境生产节拍为 c_s 的解。 $LM(C)$ 的估计直接影响算法的效率, 若界太松, 会浪费许多时间在搜索不可行的解。

2.1.1 给定 C 计算所需工作台数量的下界 $LM(C)$

若给定 u_s (或 C), 模型优化目标(1)即等价于最小化工作台数量, 而生产时间约束(2)与线性资源约束的数学表达相同, 这样原问题即退化为有资源约束的 SALBP-1^[11]。通过改进文献^[11]的方法, 计算 $LM(C)$ 的具体步骤为:

- 145 1. 给优先顺序图添加虚拟的起始任务 0 作为所有任务的前驱, 以及虚拟的任务 $(N+1)$ 作为所有任务的后驱, 任务 0 和任务 $(N+1)$ 的作业时间都为 0。
2. 根据优先顺序图, 按任务倒序, 依次计算 a_{is} 。 a_{is} 的初始值不低于任务 i 所有后驱任务构成的装箱问题的界, 即文献^[7]中的 LB1, LB2, LB3, LB6。
3. 若 $a_{is} + q_{is} > [a_{is}] + 1$, 则令 $a_{is} = [a_{is}] + 1$, 其中 $[x]$ 表示将 x 向下取整。
- 150 4. 若任务 i 存在 h 个直接后驱任务, 将这些任务按对应 *tail* 值从大到小排序, 假设序列为 $\{i_1, \dots, i_h\}$, 则 a_{is} 必须满足 $a_{is} \geq \max_{k=1, \dots, h} \{ \sum_{j=1}^k q_{i_j s} + a_{i_k s} \}$ 。若 a_{is} 为小数, 则说明 $[a_{is}]$ 个工作台不够放下任务 i 的所有后驱, 至少有一个直接后驱还未分配, 因此
- $$a_{is} \geq [a_{is}] + \min_{k=1, \dots, h} \{ q_{i_k s} \}$$
5. 记 $a_i = \max \{ a_{is} \}$, 在情境 s 任务 i 之后最少有 $[a_i]$ 个工作台, 所有情境的平衡方案相同, 因此若 $a_{is} < [a_i]$, a_{is} 必须增大到 $[a_i]$ 。若 a_i 为小数, 则与 4 同理, 有
- $$a_{is} \geq [a_i] + \min_{k=1, \dots, h} \{ q_{i_k s} \},$$
- 155 6. 将步骤 1-5 的方法应用于逆向的优先顺序图, 按任务顺序依次计算 b_{is} 。类似的定义

$b_i = \max\{b_{is}\}$ 表示执行步骤 4 之后最大的 b_{is} 值。

7. 记 $LM_s(C) = \max_{0 \leq i \leq N+1} \{\lceil a_{is} + q_{is} + b_{is} \rceil\}$, $\lceil x \rceil$ 表示将 x 向上取整, $LM(C) = \max\{LM_s(C)\}$ 。

160 按以上步骤得到 $LM(C)$ 之后, 再使用文献^[7]的 LB7 计算情境 s 需要的工作台数量, 取各情境的最大值与 $LM(C)$ 比较, 最终 $LM(C)$ 取值为两者的较大值。在本问题中若仅考虑需求情境 s , 记 $LM_s(c_s)$ 为给定 c_s 所需工作台数量的下界。 $LM_s(c_s)$ 的计算也可采用以上步骤, 令 $S=1$, 或跳过步骤 5, 所得结果与文献^[7] 的 LB7 的较大值即为 $LM_s(c_s)$ 。在与 LB7 比较取大值之前, 显然 $LM_s(C) \geq LM_s(c_s)$ 。由于最终 $LM(C)$ 也不低于各情境 LB7 的最大值, 因此必然有 $LM(C) \geq \max\{LM_s(c_s)\}$ 。当 $LM(C) > \max\{LM_s(c_s)\}$, 即对每个单独情境 s 的 SALBP 问题(m , c_s)组合界可行, 但组合(m , C)界不可行时, 需要调整 C , 增大某些情境下的生产节拍 c_s 使得组合(m , C)界可行并且增加的成本最小, 具体方法见 2.1.2 节。

2.1.2 C 的调整算法

性质 1: 若 $LM(C) > \max\{LM_s(c_s)\}$, 则对某些任务 i , 存在情境 s 满足 $a_{is} < [a_i]$ 或者 $b_{is} < [b_i]$ 。

170 **证明:** 用反证法。若对所有任务 i , 在所有情境 s 都有 $a_{is} \geq [a_i]$ 和 $b_{is} \geq [b_i]$, 则步骤 5 没有起作用。根据 $LM(C)$ 与 $LM_s(c_s)$ 的计算方法, 必定有 $LM(C) = \max\{LM_s(c_s)\}$, 与前提条件矛盾。证明完毕。

当组合(m , C)界不可行时, 用向量 $(\Delta u_1, \Delta u_2, \dots, \Delta u_S)$ 表示各情境应该增加的加班时间数量, 并称之为调整向量。对任务 $i=0, 1, \dots, N$, 记 $A_i = \{s | a_{is} \geq [a_i]\}$; 对任务 $i=1, 2, \dots, N+1$, 记 $A_{-i} = \{s | b_{is} \geq [b_i]\}$ 。另外定义 $A_{N+2} = \{s | LM_s(C) = LM(C)\}$, $A = \{A_j | j = -N, \dots, 0, 1, \dots, N+2\}$ 。

性质 2: 若 $LM(C) > \max\{LM_s(c_s)\}$, 集合 $A^* \subseteq \{1, 2, \dots, S\}$, 若只增大 A^* 中元素对应情境的生产节拍, 即 $\forall s \notin A^*, \Delta u_s = 0$, 则仅当 A^* 满足以下条件才有可能使 $LM(C)$ 减小: $\forall j \in \{-N, \dots, N+2\}$, $\exists A_j \in A$, 使得 $A_j \subseteq A^*$ 。

证明: 若不存在 $A_j \in A$ 满足 $A_j \subseteq A^*$, 则 $A_j \cap A^* \neq \emptyset, \forall j \in \{-N, \dots, N+2\}$ 。当 A^* 中元素对应情境的生产节拍增加后, $\forall i \in \{0, 1, \dots, N+1\}$, $\exists s \in A_i \setminus A^*$, 使得 $a_{is} \geq [a_i]$, 因此 $[a_i]$ 取值不会发生变化, 进而在执行 4.1.1 节算法步骤 5 时, $\forall s \notin A^*, a_{is}$ 值不会受到生产节拍调整的影响。同理 $[b_i]$ 和 b_{is} 值也不会受到影响, 故 $\forall s \notin A^*, LM_s(C)$ 的值不会发生变化。又由 $A_{N+2} \cap A^* \neq \emptyset$, 可知 $\exists s \in A_{N+2} \setminus A^*, LM_s(C)$ 的值没有变化, 因此 $LM(C)$ 的取值不会减小。证明完毕。

185 若组合(m , C)界不可行, 性质 2 排除了许多不可行的调整方向, 但满足条件的 A^* 依然很多, 寻找增加成本最小的调整向量可按以下方法进行:

1. 建立空的调整向量列表 L ;
2. 对集合 A 每个元素 A_j , 向列表 L 添加向量 $(\Delta u_1, \Delta u_2, \dots, \Delta u_S)$, 使得 $\forall s \in A_j, \Delta u_s = 1$; $\forall s \notin A_j, \Delta u_s = 0$ 。计算每个调整向量增加的成本。
3. 按增加成本最小的调整向量 $(\Delta u_1^*, \Delta u_2^*, \dots, \Delta u_S^*)$, 验证增加生产节拍后(m , C)组合是否可行, 若可行, 则结束, 否则转到 5;
4. 更新集合 A 和所有 $A_j, \forall j \in \{-N, \dots, N+2\}$ 。对集合 A 每个元素 A_i , 向列表 L 添加向量 $(\Delta u_1, \Delta u_2, \dots, \Delta u_S)$, 使得 $\forall s \notin A_i, \Delta u_s = \Delta u_s^*, \forall s \in A_i, \Delta u_s = 1 + \Delta u_s^*$ 。删除原调整向量 $(\Delta u_1^*, \Delta u_2^*, \dots, \Delta u_S^*)$, 计算每个新增调整向量增加的成本, 转到 3;
5. 重复步骤 3 和 4 直到(m , C)组合可行。

195 2.1.3 计算总成本下界

本问题中 m 与 C 都待定, 在每个单独的情境中都是类似 SALBP-E 的决策。借鉴 SALBP-E 问题的求解方法^[18], 本文采用类似思路计算总成本下界: 穷举所有界可行的 (m , C) 组合, 找出对应总成本最小的作为总成本下界。由于 C 是 s 维向量, 难以一一穷举, m 的取值范围

相对要小的多, 因此搜索沿着遍历 m 进行。

- 200 以便于理解, 结合示例说明具体计算步骤。以图 1 为例, 考虑两种产品 A 和 B, 假设存在 5 种需求情境, 各情境发生概率和对应产品需求如表 1 所示, 各产品的任务作业时间和在各情境下任务作业时间的加权平均值(单位秒)如表 2 所示。假定每天工作时间为 8 小时, 最小加班单元长度是 10 分钟, 最多加班 3 小时, 加班需支付双倍工资, 即 $T=28800$, $v=600$, $u_{\max}=18$, $c_o=2$ 。
- 205 1. 求 m 取值范围的下界。按照最大加班时间对应的生产节拍 C_{\max} , 利用 2.1.1 节算法计算所需工作台数量的下界 m_{\min} , $m=m_{\min}$ 。在示例中 $C_{\max}=(39.4, 45.3, 39.4, 40.2, 43.0)$, $m_{\min}=2$;
2. 计算给定 m 每个单独情境下所需最小生产节拍 c_s 。从 $u_s=0$ 开始, 利用 2.1.1 节算法计算 $LM_s(c_s)$, 若 $LM_s(c_s)>m$ 则增大 u_s 直至 $LM_s(c_s)=m$ 。在示例中, $m=2$, $C=(37.6, 37.7, 37.6, 37.8, 37.8)$, 对应的各情境加班单元数量为(15, 7, 15, 14, 10);
- 210 3. 利用 2.1.1 节算法验证 (m, C) 组合是否可行, 即 $LM(C)=m$ 是否成立。若成立, 则跳过 4, 直接进入 5。在示例中, $LM(C)=3>2$, 因而进入 4;
4. 利用 2.1.2 节算法调整 C 使得调整后 $LM(C)=m$ 且增加成本最小。在示例中, 执行计算 $LM(C)$ 的步骤 5 时任务 2 的 tail 和 head 都向上取整了, 有两个关键情境集合, $A_2=\{2\}$, $A_2=\{1,3,4,5\}$, 因此列表 L 中建立两个调整向量(0,1,0,0,0)和(1,0,1,1,1), 它们对应的增加成本都是 2。先尝试(0,1,0,0,0), 即令 $u_2=8$, $C=(37.6, 38.4, 37.6, 37.8, 37.8)$, 再按 2.1.1 节
- 215 算法计算得到 $LM(C)=2$, 已满足条件, 调整结束;
5. 计算 (m, C) 组合对应的成本下界 $f(m)=m(T+c_o\sum_{s=1}^S p_s u_s)$, 更新目前得到的最好的总成本下界 f_{\min} 。在示例中, $f_{\min}=f(2)=148$;
- 若 $f_{\min}>(m+1)T$, 则令 $m=m+1$, 进入 2; 否则程序结束。在示例中, $f(2)>3\times 48=144$, 因此
- 220 需要尝试 $m=3$, 转到 2, 按照类似步骤计算得到 $f(3)=144.8$, 对应各情境加班单元数量为 (0, 0, 1, 0, 0), 由于 $f(3)<4\times 48$, 算法结束, 最终总成本下界 $f_{\min}=144.8$ 。

表 1 各情境需求量和发生概率

需求 情境	发生 概率	产品需求量	
		A	B
1	0.1	591	394
2	0.5	804	201
3	0.1	670	335
4	0.2	175	700
5	0.1	828	92

表 2 各情境下任务作业时间的加权平均值

任务	作业时间		各情境中的加权平均值				
	A	B	1	2	3	4	5
1	11	12	11.4	11.2	11.3	11.8	11.1
2	17	18	17.4	17.2	17.3	17.8	17.1
3	9	8	8.6	8.8	8.7	8.2	8.9
4	5	4	4.6	4.8	4.7	4.2	4.9
5	8	7	7.6	7.8	7.7	7.2	7.9
6	12	9	10.8	11.4	11.0	9.6	11.7
7	10	10	10.0	10.0	10.0	10.0	10.0
8	3	2	2.6	2.8	2.7	2.2	2.9

225

2.2 启发式算法

借鉴 Hoffman 算法^[3]的思想, 启发式算法的基本步骤是:

1. $m=1$;
2. 穷举所有可分配到工作台 m 的任务集合, 要求在任意情境 s 集合中所有任务的总作业时间不超过最大加班时间对应生产节拍, 且集合中任务满足先后序约束;
3. 估计每个部分解的总成本下界。计算方法与 2.1 节方法类似, 只是在 2.1.3 节步骤 2 要求得到的 c_s 不低于已分配工作台在对应情境下的工作负荷;
4. 选取总成本下界最低的作为当前工作台的分配方案。若存在多个选择, 选择工作负荷较大和直接后驱较多的。
5. 若所有任务都已被分配, 算法结束, 否则进入 6;
6. $m=m+1$, 转到步骤 2。

为减少步骤 2 搜索不必要的任务集合, 算法中采用了几种占优规则。

1. 最大负荷规则(maximal load rule)。假设所有已分配工作台在需求情境 s 的最大工作负荷为 w_s , 给定当前工作台已分配的任务集合, 若还能继续添加可分配任务 i 到当前集合使得添加后当前工作台在情境 s 的工作负荷不超过 w_s , 则此集合被占优。
2. 放松任务不可再分的约束, 预判部分解的总成本下界。假设所有未分配任务都任意可分, 在调用 2.1 节算法之前, 利用类似思想在极短时间内即可获得约束放松后的总成本下界。若该值高于目前最好的部分解的总成本下界, 则此集合被占优。
3. 若当前任务集合还可继续添加可分配任务 i , 使得添加后当前工作台的工作负荷不超过成本下界对应的最优生产节拍, 则当前集合被占优。

3 数值实验

为验证算法有效性, 利用 SALBP 标准算例库 (www.assembly-line-balancing.de) 随机生成了 25 个测试案例。对每个算例, 假定混装线装配两种产品 A 和 B, 产品 A 的各任务作业时间即标准算例中的作业时间, 产品 B 的各任务作业时间为产品 A 该任务的时间的 γ 倍, γ 服从区间为 [0.8, 1.2] 的均匀分布, 各任务的优先序关系与标准算例相同。所有算例中 $s=5$, $T=288000$, $v=6000$, $u_{\max}=18$, $c_o=2$, 各情境下两种产品的需求比例随机产生, 需求总量服从区间为 $[T/c_{\min}, T/c_{\max}]$ 的均匀分布, 其中 c_{\min} 和 c_{\max} 是该算例在文献中测试过的最小节拍时间和最大节拍时间。

用 C 语言实现上述算法程序, 并在一台 CPU 为 T8200, 内存 2G 的 PC 上运行。在 4.2 节穷举可分配在当前工作台的所有可行任务集合时采用了 60s 的计算时间限制, 即运行时间超过 60s 就不再继续穷举。数值实验结果如表 3 所示, 可以看到有 4 个算例启发式算法找到的解就等于下界, 因此必定是最优解。在所有算例中启发式算法找到的解与界的平均偏差为 3.02%, 最大偏差为 12.97%, 所有算例的 CPU 时间都不超过 3 分钟。

表 3 数值实验结果

算例	任务数量 N	总成本下界 f_{\min}	启发式算法的解	与界的差距
Arc1	83	566.6	572.6	1.06%
Arc2	111	566.28	570.9	0.82%
Barthold1	148	446.94	446.94	0.00%
Barthold2	148	1501.64	1590.92	5.95%
Bowman	8	144.72	144.72	0.00%
Buxey	29	339.92	356.58	4.90%
Gunther	35	446.76	496.4	11.11%

Hahn	53	298.8	298.8	0.00%
Heskia	28	190.38	191.7	0.69%
Jackson	11	308.16	308.16	0.00%
Jaeschke	9	166.74	173.94	4.32%
Kilbrid	45	240	240	0.00%
Lutz1	32	310.08	319.8	3.14%
Lutz2	89	1384.88	1463.4	5.67%
Lutz3	89	815.68	859.86	5.42%
Mansoor	11	140.76	141.76	0.71%
Mertens	7	100.8	106.48	5.64%
Mitchell	21	147	147.72	0.49%
Mukherje	94	636.48	646.1	1.51%
Roszieg	25	274.4	282.6	2.99%
Sawyer	30	336	340.76	1.42%
Scholl	297	2332.8	2381.4	2.08%
Tonge	70	594.24	612.48	3.07%
Warnecke	58	918	1037.02	12.97%
Wee-Mag	75	2976	3024	1.61%

260

4 结语

尽管现实中的产品需求频繁波动,混装线平衡文献中通常假定在平衡阶段已知各产品确定的市场需求量,忽视了平衡方案后续进行产能调整的难易程度和带来的成本。针对这一缺陷,本文讨论了需求不确定的混装线平衡决策,假定企业采用加班这种最常用的产能调整手段应对需求波动,考虑了日常作业和临时加班产生的人工成本。模型突出了产能的维持成本和调整成本之间的权衡,并采用了情景规划的思想,帮助决策者决定最合适的工作台数量以及不同需求情境下的加班时间,以最小化期望总成本。为有效求解此问题,本文提出了估计成本下界的方法,并设计了启发式算法,计算实验表明算法能在较短时间内获得较好的解。

265

[参考文献] (References)

- [1] Becker C, Scholl A. A survey on problems and methods in generalized assembly line balancing[J]. European Journal of Operational Research, 2006,168(3): 694-715.
- [2] Scholl A, Becker C. State-of-the-art exact and heuristic solution procedures for simple assembly line balancing[J]. European Journal of Operational Research, 2006,168(3): 666-693.
- [3] Hoffman TR. Assembly line balancing with a precedence matrix[J]. Management Science, 1963,9(4): 551-562.
- [4] Hoffmann TR. Eureka: a hybrid system for assembly line balancing[J]. Management Science, 1992,38(1): 39-47.
- [5] Johnson RV. Optimally balancing large assembly lines with Fable[J]. Management Science, 1988,34(2): 240-253.
- [6] Nourie F, Venta E. Finding optimal line balances with OptPack[J]. Operations Research Letters, 1991,10(3): 165-171.
- [7] Scholl A, Klein R. SALOME: a bidirectional branch-and-bound procedure for assembly line balancing[J]. INFORMS Journal on Computing, 1997,9(4): 319-334.
- [8] Fleszar K, Hindi KS. An enumerative heuristic and reduction methods for the assembly line balancing problem[J]. European Journal of Operational Research, 2003,145(3): 606-620.
- [9] Sewell EC, Jacobson SH. A branch, bound, and remember algorithm for the simple assembly line balancing problem[J]. INFORMS Journal on Computing, 2011.
- [10] Berger I, Bourjolly J-M, Laporte G. Branch-and-bound algorithms for the multi-product assembly line balancing problem[J]. European Journal of Operational Research, 1992,58(2): 215-222.
- [11] Scholl A, Flidner M, Boysen N. ABSALOM: balancing assembly lines with assignment restrictions[J].

270

275

280

285

- 290 European Journal of Operational Research, 2010,200(3): 688-701.
- [12] Boysen N, Fliedner M, Scholl A. Production planning of mixed-model assembly lines: overview and extensions[J]. Production Planning & Control, 2009,20(5): 455-471.
- [13] Carraway RL. A dynamic programming approach to stochastic assembly line balancing[J]. Management Science, 1989,35(4): 459-471.
- 295 [14] Sarin SC, Erel E, Dar-El EM. A methodology for solving single-model, stochastic assembly line balancing problem[J]. Omega, 1999,27(5): 525-535.
- [15] Xu W, Xiao T. Robust balancing of mixed model assembly line[J]. COMPEL: Int J for Computation and Maths in Electrical and Electronic Eng, 2009,28(6): 1489-1502.
- [16] Xu W, Xiao T. Strategic Robust Mixed Model Assembly Line Balancing Based on Scenario Planning[J]. Tsinghua Science & Technology, 2011,16(3): 308-314.
- 300 [17] Wee TS, Magazine MJ. Assembly line balancing as generalized bin packing[J]. Operations Research Letters, 1982,1(2): 56-58.
- [18] Rosenblatt MJ, Carlson RC. Designing a production line to maximize profit[J]. IIE Transactions, 1985,17(2): 117-122.
- 305