强化 Dynasearch & TS 算法求解酸轧生产调度问题

唐立新 [1] 赵 任 [1]

摘 要 酸轧生产调度的主要任务是在满足酸轧机组生产工艺和能力约束下, 考虑下游机组的流向需求, 为保证生产连续性和平滑过渡的要求, 从给定候选池中选择适合的板卷构成一个酸轧调度单元. 针对此问题, 本文建立了以最小化过渡费用和调度单元剩余容量惩罚费用为目标的整数规划模型, 提出了一种嵌入强化 Dynasearch 算法的禁忌搜索混合算法. 该混合算法采用基于最小插入法的两阶段启发式产生初始解, 根据采用邻域结构的不同设计双禁忌表, 为了避免算法陷入局部最优, 在禁忌搜索的每次迭代过程中嵌入 Swap 邻域和 Inner-insert 邻域相结合的多交换 Dynasearch 邻域, 并设计了多项式动态规划算法搜索该邻域. 针对问题的特征, 提出了 Block 分区结构, 基于此分析了多个可行解性质, 有效降低了搜索空间. 与一般禁忌搜索算法比较, 结果表明所提出的强化 Dynsearch & TS (Tabu search) 算法求解效果明显优于一般 TS 算法, 平均改进量为 3.62 %, 算法运行时间大大缩短. 验证了该算法在解决此类问题的有效性.

关键词 酸轧生产调度, 禁忌搜索, Dynasearch 算法, Dynasearch 邻域

DOI 10.3724/SP.J.1004.2010.00304

A New Enhanced-Dynasearch & TS for the Pickling-rolling Scheduling Problem

TANG Li-Xin[1] ZHAO Ren[1]

Abstract Motivated by pickling-rolling production in steel industry, the pickling-rolling scheduling problem is to create a schedule by selecting and sequencing many proper coils from the candidate pool with consideration of the changeovers cost between adjacent coils. Besides the capacity constraints and practical production technical constraints, the flow requirement constraint is also considered so that the total weight of the selected coils belonging to the same downstream unit satisfies the production requirement of the corresponding downstream unit, the objective being to minimize the total transition cost and the penalty cost on the left capacity. The problem is formulated as an integer programming model and a new tabu search (TS) with the enhanced dynasearch algorithm is developed for it. A two-phase heuristic based on nearest insert method is proposed to act as the initial feasible solution to the tabu search. The proposed algorithm designs a double tabu list to fit the different neighborhood structure. For the solution to escape from the local optima, the enhanced dynasearch neighborhood is embedded in each iteration. The dynasearch neighborhood is polynomially searchable by dynamic programming and it can perform a multiple-exchange composite move by combining swap and inner-insert neighborhoods. Based on the analysis of the characteristics of the problem, block structure is defined and several properties of the feasible solution are derived to accelerate the search process. Computational results show that the enhanced-dynsearch & TS algorithm is effective for solving the problem and outperforms the standard one by 3.62 % on average.

Key words Pickling-rolling scheduling, tabu search (TS), dynasearch algorithm, dynasearch neighborhood

随着各钢铁企业的生产规模增加, 国内钢铁市场日益饱和, 竞争也更加激烈. 钢铁企业必须提高产品层次, 增加单位产品的附加值, 才能在激烈的市场竞争中立于不败之地. 冷轧是钢铁生产过程的最后阶段, 经过这些工序加工的钢板都具有较高的附加值. 而冷轧酸洗和轧制联合机组 (下面简称 "酸轧") 是冷轧生产阶段的第一道也是必经工序. 因此, 该工序的生产合理性直接影响其后道各机组的生产. 经过酸轧之后的板卷根据订单合同的需要被分别送至热镀锌机组、连续退火机组或其他机组进行再加工, 如图 1 所示.

酸轧生产调度的主要任务是在满足酸轧机组生产工艺和能力约束下以及下游机组流向需求条件下, 为保证生产连续性和平滑过渡, 从给定候选池中选择适合的板卷构成一个酸轧生产调度单元. 据作者目前所知, 已发表的文献大多是集中在热轧[1−4] 和精整工序[5], 以及冷轧工序[6], 还没有对酸轧问题进行过研究. Lopez 等[1] 针对钢铁企业的热轧生产调度问题建立了一个带选择的旅行商问题 (Prize collecting traveling salesman problem, PCTSP)

收稿日期 2008-12-16 录用日期 2009-07-14
Manuscript received December 16, 2008; accepted July 14, 2009
国家高技术研究发展计划 (863 计划) (2006AA04Z174), 国家自然科学基金 (60674084), 国家杰出青年科学基金 (70425003) 资助
Supported by National High Technology Research and Development Program of China (863 Program) (2006AA04Z174), National Natural Science Foundation of China (60674084), and National Natural Science Foundation for Distinguished Young Scholars of China (70425003)
1. 东北大学物流优化与控制研究所辽宁省制造系统与物流优化重点实验室 沈阳 110004
1. Liaoning Key Laboratory of Manufacturing System and Logistics, The Logistics Institute, Northeastern University, Shenyang 110004

模型. 为了有效地求解该模型, 该文献使用了禁忌搜索算法, 对加拿大一个热轧厂实际生产数据的测试结果表明, 他们所提出的模型和算法要明显优于该热轧厂现有的手工方法. Zhao 等[2] 针对热轧生产调度问题提出了两阶段调度方法, 将主体材批计划描述成带时间窗口的车辆路径问题 (Vehicle routing problem with time windows, VRPTW) 模型, 并采用改进的单亲遗传算法 (Parthenogenetic algorithm, PGA) 进行求解. 然后对于获得的批, 通过使用智能搜索算法调整轧制顺序来获得较高的热轧费用比. Tang 等[3] 针对热轧生产调度问题不同于传统的串行求解策略, 提出了一个并行策略建立该问题的数学模型, 把该问题归结一个多旅行商问题 (Multi-traveling salesman problem, MTSR), 并构造了一个改进遗传算法来获得问题的近优解. Tang 等[4] 以热轧板坯订单为对象, 每个订单包含多个板坯, 考虑了热轧机组的轧制能力和生产下限要求, 对热轧板坯订单的选择和排序集成调度问题进行了研究, 对此问题建立了数学规划模型, 提出了一个基于大规模环交换邻域的迭代局域搜索算法进行求解, 测试结果表明该算法要优于一种同样使用大规模环交换邻域的多点下降算法. Okano 等[5] 研究冷轧精整线板卷调度问题, 提出了一个高性能解决方法使板卷拖期最小. Zhao 等[6] 针对冷轧厂生产调度的复杂性, 将冷轧调度优化分解成板卷合并优化和轧制批优化两个阶段. 对钢卷合并的优化问题建立了多集装箱背包问题, 并采用离散差分进化算法进行求解. 对轧制批计划归结为一个特殊的双旅行商问题并采用基于进化机制和局域搜索的混合启发式进行求解.

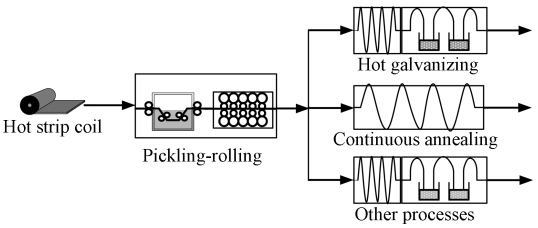


图 1　冷轧厂板卷生产流程

Fig. 1　The production processes of coils

酸轧生产调度问题与已有的研究问题相比有两方面不同: 1) 问题本身: 它在考虑了酸轧机组生产工艺和生产能力的同时, 还考虑了酸轧机组后道机组的流向需求约束. 流向需求约束主要是为平衡酸轧机组后道工序的库存量以及保证生产的连续性, 防止过多或过少的生产某种流向的板卷造成后道机组前库胀库或缺料停产的现象发生. 根据订单合同, 原料库内每个板卷的流向是已知的, 且每个板卷只能进入一个后序流向进行加工; 2) 求解方法: 已有研究的钢铁企业生产调度问题大都采用禁忌搜索 (Tabu search, TS), 遗传算法 (Genetic algorithm, GA), 迭代局域搜索 (Iterated local search, ILS) 或者它们之间的混合算法, 没有研究采用 Dynasearch & TS 算法求解钢铁企业生产调度问题. 本文提出一种强化 Dynasearch & TS 算法来求解酸轧生产调度问题. 该混合算法将 Dynasearch 算法和禁忌搜索算法机制相结合, 为了使禁忌搜索每次迭代获得的解更接近局域最优, 在禁忌搜索的每次迭代过程嵌入强化 Dynasearch 算法, 提高解的质量.

1　问题描述及模型

1.1　酸轧生产过程及工艺约束

酸轧所用的原料主要是热轧轧制后的热卷, 热卷在酸轧原料库经过冷却后, 由钢卷运输车将钢卷送到入口段开卷机. 为保证酸轧生产线的连续生产, 经开卷、切头后相邻钢卷焊接在一起. 焊接后的板带经过入口活套后进入酸洗槽去除表面的氧化物. 经过酸洗后的板带按照合同要求进行切边. 再通过出口活套, 板带经过连轧机组进行轧制, 轧制后的板带剪切最终卷曲成卷.

对于轧制工序, 轧制机组一般由 5 个机架组成. 每一个机架上的工作辊, 由于高速轧制, 轧辊磨损很大, 为保证钢卷质量需要经常更换工作辊, 更换前后两次工作辊之间的轧制对象称为轧制单元, 如图 2 所示. 每个轧制单元对应一个酸轧生产调度.

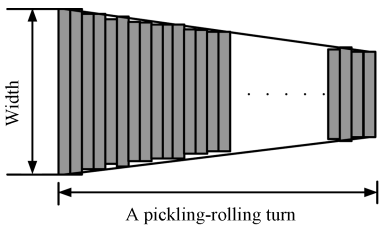


图 2　酸轧生产调度单元构成

Fig. 2　The forming of a pickling-rolling turn

为确保酸轧生产的连续性, 酸轧生产过程需要满足以下约束: 1) 为防止轧制阶段跳跃辊痕的发生, 酸轧生产调度单元中板卷宽度变化趋势应该从宽到窄, 但在允许范围内, 部分板卷可以反跳, 即从窄到宽; 2) 为防止拉伸过程中发生断带, 要求相邻板卷的入口厚度和出口厚度应平滑过渡, 规格跳跃尽可能小; 3) 酸轧卷钢级分为 3 类: 软钢、硬钢和高强钢. 相邻板卷要满足钢种连接要求, 即: 软钢与高强钢不能相邻, 同类钢种的板卷应尽可能编排在一起; 4) 为防止切边剪的频繁启动, 需要切边的板卷应尽可能编排在一起; 5) 相邻两个酸轧生产调度单元切

换时, 需要更换轧制的工作辊, 更换一次轧辊将产生较高的成本费用, 因此一个酸轧生产调度单元应在轧辊的最大轧制能力范围下尽可能多地加工板卷; 6) 为平衡酸轧机组后道工序的库存量以及生产的连续性, 酸轧生产调度单元内不同流向板卷的产量要满足该流向需求约束.

1.2 数学模型

酸轧生产调度的编制主要是从给定板卷集合中, 根据不同流向需求, 选择一定量板卷构成一个酸轧轧制单元并在满足上述酸轧工艺约束 1) ∼ 4) 下编排选中板卷的轧制顺序, 使得产品质量和产能最大化.

在酸轧生产阶段, 产品质量的最大化对应生产调度单元内相邻板卷的规格过渡跳跃的最小化, 因为相邻板卷的规格跳跃越小、过渡越平滑, 轧制时对工作辊的磨损越小, 从而对后面要加工的板卷的质量所产生的影响也就越小. 酸轧实际生产工艺约束包括宽度、厚度、钢种以及切边等规格跳跃约束, 它们具有不同的度量单位, 通过建立量化这些规格变化的数值惩罚结构, 使板卷之间规格过渡费用能够由它们的物理规格变化直接计算出来. 对钢种按照钢的硬度进行顺序编号共分 s 级钢, 其中 $1 \sim s_1$ 级属于软钢, $s_1+1 \sim s_2$ 级属于硬钢, $s_2+1 \sim s$ 级属于高强钢, 将钢种属性转换成数值编号, 通过数值的表达可以描述出相邻板卷钢种连接约束以及计算它们之间的规格变化惩罚值. 此外, 通过对轧制单元内剩余容量的惩罚来体现产能最大化, 因为剩余容量越小说明轧制单元内板卷容量越大. 那么, 通过上述转换, 该问题的目标就是最小化总的惩罚费用 (包括板卷规格过渡惩罚和剩余容量惩罚).

为了更清晰地描述问题, 采用以下参数和变量:
Ω: 给定候选板卷集合, $\Omega = \{0, 1, 2, \cdots, N\}$, 其中 0 表示虚拟板卷;
L: 酸轧后道机组流向集合, $L = \{1, 2, \cdots, P\}$, 其中 P 表示机组流向个数;
Q: 一个酸轧调度单元最大轧制能力, 用重量表示;
B: 一个酸轧调度单元最小轧制能力, 用重量表示;
M: 一个无限大的数;
f_l: 流向 l 的机组板卷需求量 (用重量表示), $l \in L$;
d_i: 板卷 i 的重量;
w_i: 板卷 i 的宽度;
t_i^b: 板卷 i 的入口厚度;
t_i^a: 板卷 i 的出口厚度;
g_i: 板卷 i 的钢级;
h_{il}: 板卷 i 与流向 l 的所属关系, 当板卷 i 的流向为 l 时, $h_{il}=1$, 否则 $h_{il}=0$. 根据订单合同, 原料库内每个板卷的流向已知, 且只能进入一个流向的机组进行加工;
δ: 相邻板卷间规格最大允许跳跃值集合, $\delta = \{\delta_1, \delta_2, \cdots, \delta_5\}$, 包括宽度、入口厚度、出口厚度、钢级跳跃范围;
α: 惩罚系数集合, $\alpha = \{\alpha_1, \alpha_2, \cdots, \alpha_8\}$;
p_{ij}^w: 板卷 i 和 j 的宽度跳跃惩罚, 它由相邻板卷间宽度过渡约束决定.

$$p_{ij}^w = \begin{cases} \alpha_1(w_i - w_j), & 0 \le w_i - w_j < \delta_1 \\ \alpha_2(w_j - w_i), & -\delta_2 < w_i - w_j \le 0 \\ \infty, & \text{其他} \end{cases}$$

p_{ij}^o: 板卷 i 和 j 的出口厚度跳跃惩罚, 它由相邻板卷间出口厚度过渡约束决定.

$$p_{ij}^o = \begin{cases} \alpha_3 \mid t_i^a - t_j^a \mid, & \mid t_i^a - t_j^a \mid \le \delta_5 \\ \infty, & \text{其他} \end{cases}$$

p_{ij}^r: 板卷 i 和 j 的入口厚度跳跃惩罚, 它由相邻板卷间入口厚度过渡约束决定.

$$p_{ij}^r = \begin{cases} \alpha_4 \mid t_i^b - t_j^b \mid, & \mid t_i^b - t_j^b \mid \le \delta_3 \text{且} \dfrac{t_i^b}{t_j^b} \le \delta_4 \\ \infty, & \text{其他} \end{cases}$$

p_{ij}^s: 板卷 i 和 j 的钢级跳跃惩罚, 它由相邻板卷间钢级连接约束决定. 令 O_1 表示软钢集合, O_2 表示硬钢集合, O_3 表示高强钢集合.

$$p_{ij}^s = \begin{cases} 0, & g_i = g_j \\ \alpha_5, & g_i \ne g_j \text{且} g_i \text{和} g_j \text{属于同类钢种} \\ \infty, & g_i \ne g_j \text{且} g_i \text{和} g_j \text{分别属于} O_1 \text{和} O_3 \\ \alpha_6, & \text{其他} \end{cases}$$

p_{ij}^c: 板卷 i 和 j 的切边跳跃惩罚, 如果相邻板卷都需要切边, $p_{ij}^c = 0$, 否则 $p_{ij}^c = \alpha_7$.
c_{ij}: 相邻板卷 i 和 j 之间生产切换的过渡费用 $c_{ij} = p_{ij}^w + p_{ij}^o + p_{ij}^r + p_{ij}^s + p_{ij}^c$, $i, j \in \Omega$, 其中 $c_{0j} = c_{i0} = 0$.
决策变量:

$$x_{ij} = \begin{cases} 1, & \text{板卷} j \text{紧接板卷} i \text{加工且} i \ne j \\ 0, & \text{其他} \end{cases}, i, j \in \Omega$$

$$y_i = \begin{cases} 1, & \text{板卷} i \text{被编制在计划单元中加工} \\ 0, & \text{其他} \end{cases}, i \in \Omega$$

根据上述模型参数和定义的决策变量, 建立整数规划模型如下:

$$\min \sum_{i \in \Omega} \sum_{j \in \Omega \setminus \{i\}} c_{ij} x_{ij} + \alpha_8 (Q - \sum_{i \in \Omega} d_i y_i) \tag{1}$$

s.t.

$$\sum_{j\in\Omega\backslash\{i\}} x_{ij} \le 1,\ \forall i \in \Omega\backslash\{0\} \tag{2}$$

$$\sum_{i\in\Omega\backslash\{j\}} x_{ij} \le 1,\ \forall j \in \Omega\backslash\{0\} \tag{3}$$

$$\sum_{i\in\Omega\backslash\{0\}} x_{0i} = \sum_{j\in\Omega\backslash\{0\}} x_{j0} = y_0 = 1 \tag{4}$$

$$\sum_{i\in\Omega\backslash\{0\}} x_{ik} = \sum_{j\in\Omega\backslash\{0\}} x_{kj} = y_k, \forall k \in \Omega\backslash\{0\} \tag{5}$$

$$\sum_{i\in S}\sum_{j\in S\backslash\{i\}} x_{ij} \le \mid S \mid -1, S \subset \{1,2,\cdots,n\},\ 2 \le \mid S \mid \le n \tag{6}$$

$$\sum_{i\in\Omega} d_i y_i \le Q \tag{7}$$

$$\sum_{i\in\Omega} d_i y_i \ge B \tag{8}$$

$$\sum_{i\in\Omega} d_i h_{il} y_i \ge f_l,\ \forall l \in L \tag{9}$$

$$0 \le w_i - w_j < \delta_1 + (1 - x_{ij})M, \forall i,j \in \Omega \tag{10}$$

$$0 \le w_j - w_i < \delta_2 + (1 - x_{ij})M, \forall i,j \in \Omega \tag{11}$$

$$\mid t_i^b - t_j^b \mid \le \delta_3 + (1 - x_{ij})M,\ \forall i,j \in \Omega \tag{12}$$

$$\frac{t_i^b}{t_j^b} \le \delta_4 + (1 - x_{ij})M,\ \forall i,j \in \Omega \tag{13}$$

$$\mid t_i^a - t_j^a \mid \le \delta_5 + (1 - x_{ij})M,\ \forall i,j \in \Omega \tag{14}$$

$$x_{ij} \mid g_i - g_j \mid < 1,\ \forall i \in O_1, j \in O_3$$
$$\text{或 } \forall i \in O_3, j \in O_1 \tag{15}$$

$$x_{ij} \in \{0,1\},\ \forall i,j \in \Omega \tag{16}$$

$$y_i \in \{0,1\},\ \forall i \in \Omega \tag{17}$$

目标函数 (1) 由两部分构成: 第一项为相邻板卷间的总的过渡惩罚费用, 第二项为生产调度单元内剩余容量惩罚费用. 约束 (2) 和 (3) 保证每个板卷最多只被轧制一次. 约束 (4) 确保酸轧生产调度单元从虚拟板卷开始轧制, 以虚拟板卷结束. 约束 (5) 确保酸轧生产调度单元内每个选中板卷必有一个紧前和紧后板卷与其连接. 约束 (6) 消除子回环. 约束 (7) 和 (8) 保证酸轧生产调度单元内板卷总重量满足酸轧机组的生产能力限制. 约束 (9) 确保酸轧生产调度单元内不同流向板卷的量要满足该流向需求. 约束 (10) 为生产调度单元内相邻板卷间宽度由宽到窄跳跃约束. 约束 (11) 为生产调度单元内相邻板卷间宽度反跳约束. 约束 (12) 和 (13) 为生产调度单元内相邻板卷间入口厚度跳跃约束. 约束 (14) 为生产调度单元内相邻板卷间出口厚度跳跃约束. 约束 (15) 确保软钢和高强钢不能相邻. 约束 (16) 和 (17) 保证决策变量必须为 0, 1 整数.

如果忽略酸轧实际生产约束以及流向需求约束, 上述模型可以归结为 PCTSP 模型. 由于 PCTSP 模型是 NP 难的, 本文提出的问题模型考虑了更多的实际生产工艺约束, 相对比较复杂, 因此本文所研究的酸轧生产调度问题也是 NP 难的.

2 求解方法

针对问题的特点, 本文提出了一种适合酸轧生产调度问题的嵌入强化 Dynasearch 算法的禁忌搜索 (强化 Dynasearch & TS) 混合算法进行求解, 将强化 Dynasearch 算法嵌入到禁忌搜索算法中.

禁忌搜索算法是由 Glover[7] 提出的一种为寻找混合优化问题近优解而设计的多启发式算法. 它以其独特的短期表和长期表的功能使得算法能成功地实现爬山特性, 防止算法陷入局部最优, 成功地解决了各类经典调度问题[8−9] 和钢铁工业实际生产调度问题[1]. Nowicki[8] 对连续机器间带有有限存储能力的排序 Flowshop 问题, 采用结合结构性质的 TS 搜索算法进行求解, 提出的算法可以对具有 200 个工件和 20 个机器这样的规模得到较好解. Grabowski 等[9] 对目标为最小化 Makespan 的排序 Flowshop 问题, 提出了基于 TS 算法的快速局域迭代搜索算法进行求解, 提出的算法能在较短的时间内获得较好的解. Lopez 等[1] 对热轧生产调度问题建立单 TSP 模型, 并提出了基于块拆分策略的禁忌搜索算法, 有效求解了实际热轧调度问题.

Dynasearch 邻域使用动态规划方法在多项式时间内寻找一些传统邻域 (如: Swap, Insert 等) 的无关独立的组合移动进行每次迭代[10]. Dynasearch 邻域有以下几个特征: 1) 不同于传统单一邻域由一个移动构成, 它是由多个无关的移动构成一次 Dynasearch 移动; 2) 它是通过动态规划的方法来搜索邻域并能够在多项式时间内搜索指数级规模的邻域[10]. Dynasearch 算法是一种基于 Dynasearch 邻域的迭代下降算法, 作为一种新型而有效的近似求解算法已经成功应用于 TSP 问题[10−11]、线性顺序问题和单机调度问题中[10]. Congram[10] 提出求解一般 TSP 的 Dynasearch 策略, 该策略在多项式时间范围内搜索到指数形式大小的邻域. 同时, Congram 在文献 [10] 中应用 Dynasearch 对单机调度问题进行了研究. 李妍峰等[11] 研究了时间依赖型旅行商问题, 建立相应的数学模型, 并提出求解该问题的动态搜索算法. 通过实验仿真, 验证了动态搜索算法优于目前在邻域搜索领域求解该问题最有效的动态规划启发式算法. Congram 等[12] 应用迭代 Dynasearch 算法求解经典的总权重拖期最小单机调度问题, 与传统的第一改进下降算法或是最好改进下

降算法比较, 更有效且更接近最优解. Grosso 等[13] 在文献 [12] 的基础上, 提出了强化 Dynasearch 邻域, 它是通过广义对交换策略获得的, 实验结果表明其显著优于文献 [12].

本文提出的强化 Dynasearch 算法与已有研究有以下四点不同: 1) 与一般 Dynasearch 算法不同, 它的邻域是由多种不同传统邻域结构构成的一系列无关独立的移动, 这里采用 Swap 邻域和 Inner-insert 邻域相结合的多交换 Dynasearch 邻域; 2) 已有的研究强化 Dynasearch 算法主要求解经典单机调度问题, 而本文提出的强化 Dynasearch 算法应用在实际酸轧生产选择和排序集成优化调度问题; 3) 针对问题的特点, 本文提出了 Block 分区策略, 获得了每个板卷有效的移动区域, 强化 Dynasearch 算法与 Block 区域性质相结合, 降低了 Dynasearch 邻域搜索空间, 加速了多项式动态规划算法搜索 Dynasearch 邻域的过程; 4) 本文将强化 Dynasearch 算法与 TS 算法相结合, 结合了 Dynasearch 算法快速搜索和禁忌搜索防止陷入局部最优的特点, 使算法能够更适应酸轧实际生产中规模大、约束复杂的需要, 能够更快更有效地得到酸轧生产调度. 针对钢铁企业酸轧生产实际调度问题, 本文提出的强化 Dynasearch & TS 算法求解思路如下: 禁忌搜索算法开始于一个初始解, 在每次迭代中, 首先在第 2.2 节中提到的 Delete 邻域、Insert 邻域和 Exchange 邻域中寻找一个最好的邻居 (也就是邻域中的最好解), 执行引导向这个邻居的移动, 得到一个新的当前解. 在新的当前解下调用强化 Dynasearch 算法进行改进, 直到无法改进当前解时, 强化 Dynasearch 算法终止, 得到一个新的当前解, 从这个当前解出发开始禁忌搜索的下一次迭代. 具体流程在第 2.6 节中描述.

2.1 初始解

本文基于最小值插入算法设计了一个两阶段贪婪启发式产生问题的初始解. 第一阶段是构造问题的可行解; 第二阶段是对已获得的可行解在满足约束 (7) 的条件下通过增加适当板卷使得在降低总的惩罚费用的同时增加生产调度容量.

令 Ω' 为把 Ω 中板卷按照从宽到窄的顺序进行排序、编号后得到的新的序列. Ω'_l 为按照从宽到窄的顺序排列的流向为 l 的板卷序列. S 为酸轧生产调度的解. S_l 为 S 内流向为 l 的板卷集合. $D(S_l)=\sum_{i\in S} d_i h_{il}$ 为 S 内流向为 l 板卷的总重量.

针对上述问题和模型的描述, 根据模型中约束 (10) 和 (11) 分析可获得: 对于候选板卷集合中任意未排入 S 中的板卷 i, 在 S 中可插入的有效位置只能是在紧前板卷宽度为 $(w_i+\delta_1, w_i-\delta_2)$ 和紧后板卷宽度为 $(w_i+\delta_2, w_i-\delta_1)$ 之间, 即可插入的位置仅在宽度为 $(w_i+\delta_1, w_i-\delta_1)$ 区间之间. 根据该结论, 可以预先判断出每个板卷可插入轧制单元内的有效位置范围, 能有效减少可行解的搜索空间, 提高算法效率.

结合上述结论, 两阶段启发式步骤如下:

步骤 1. 根据酸轧生产工艺宽度约束的要求, 将给定候选板卷集合内板卷按照从宽到窄的顺序排列, 得到新的顺序序列 Ω', 计算 Ω' 内板卷间 c_{ij} 值.

步骤 2. 构造可行解. 首先对所有需求流向, 判断需求量是否接近于 Ω 中该流向板卷的总容量, 如果存在这样的流向 l, 那么将 Ω' 中该流向板卷按照最小值插入法排序后添加到 S 中. 否则, 从所有需求流向中选择宽度最大的板卷插入到 S 中. 然后, 从剩余需求流向板卷中选择插入后使得过渡费用变化值 $\Delta c = c_{ik}+c_{kj}-c_{ij} < \infty$ 最小的板卷 k 插入到 S 中的相应位置. 当无法从需求流向板卷中找到这样的板卷时, 从非需求流向板卷中选择一个过渡板卷 q, 使其插入 S 后能再从剩余需求流向板卷中找到插入后使过渡费用变化值 $\Delta c' = c_{iq}+c_{qk}+c_{kj}-c_{ij} < \infty$ 或 $\Delta c' = c_{ik}+c_{kq}+c_{qj}-c_{ij} < \infty$ 最小的板卷 k. 重复以上过程直到所有流向需求满足 $D(S_l) \geq f_l$ 为止. 判断此时 S 是否满足约束 (8), 如果满足则 S 即为可行解; 如果不满足则从剩余候选板卷中按照最小值插入法选择板卷插入到 S 中, 重复该过程直到约束 (8) 满足为止.

步骤 3. 完善可行解. 从 Ω' 剩余板卷中按照最小值插入法选择插入 S 后使目标函数变化值 $\Delta F = c_{ik}+c_{kj}-c_{ij}-\alpha_8 d_k < 0$ 的板卷 k, 如果没有这种板卷则终止算法, 否则重复该过程直到约束 (7) 被违反为止.

2.2 邻域

针对研究问题的特点, 本文采用 4 种不同的邻域结构, 分别是 Delete 邻域、Insert 邻域、Exchange 邻域和强化 Dynasearch 邻域.

Delete 邻域是通过从 S 中删除一个板卷产生. Insert 邻域是通过从剩余候选板卷集中选择一个板卷将其插入到 S 中产生. Exchange 邻域是通过分别在 S 中和剩余候选板卷集中选择一个板卷进行交换. 强化 Dynasearch 邻域是通过对 S 中板卷进行一系列由多种不同邻域结构产生的无关独立的移动构成.

强化 Dynasearch 邻域是将 Swap 邻域和 Inner-insert 邻域相结合. 其中 Swap 邻域是通过交换 S 中两个板卷产生; Inner-insert 邻域是通过在 S 中选择某个位置上的板卷插入到 S 中的其他位置上, 包括前向插入和后向插入, 例如给定序列 $S=\sigma i \pi j \omega$,

$i<j$, 向前插入 (i,j) 得到 $\sigma ji\pi\omega$, 向后插入 (i,j) 得到 $\sigma\pi ji\omega$.

设得到的一个解 $S=(\sigma(0),\sigma(1),\cdots,\sigma(n))$, 其中 $\sigma(0)=\sigma(n)$ 表示虚拟板卷, $\sigma(j)$ 为 S 中第 j 个酸轧板卷, $F(j,S)$ 为 S 中从 $\sigma(0)$ 板卷到 $\sigma(j)$ 板卷最小过渡费用, 则 $F(j+1,S)$ 包含板卷 $\sigma(j+1)$. 在第 j 阶段解的子集加入第 $j+1$ 个板卷有三种方式: 1) 板卷 $\sigma(j+1)$ 紧接板卷 $\sigma(j)$ 排入 S 中, 同时保持 S 内原有板卷顺序不变, 如图 3 所示; 2) 加上板卷 $\sigma(j+1)$ 后, 对所得到的 S 内板卷按照 $i\,(i=0,1,\cdots,j-2)$ 执行一系列独立的交换, 即交换 S 内板卷 $\sigma(j)$ 和 $\sigma(i+1)$ 位置, 如图 4 所示; 3) 加入板卷 $\sigma(j+1)$ 后, 对所得到的 S 内板卷根据 $i\,(i=0,1,\cdots,j-2)$ 执行一系列独立的内部插入, 即将板卷 $\sigma(j)$ 插入到紧接 $\sigma(i)$ 之后, 如图 5 所示. 那么强化 Dynasearch 邻域的动态规划算法可以描述如下:

初始条件: $F(0,S)=0$, $F(1,S)=0$, $F(2,S)=c_{\sigma(1),\sigma(2)}$, $F(3,S)=c_{\sigma(1),\sigma(2)}+c_{\sigma(2),\sigma(3)}$. 对 $j=3,\cdots,n$, $F(j+1,S)$ 递推公式可表示为:

$$\begin{aligned}F(j+1,S)=\min\Big\{&F(j,S)+c_{\sigma(j),\sigma(j+1)},\\&\min_{0\le i\le j-2}\big\{F(i,S)+c_{\sigma(i),\sigma(j)}+\\&c_{\sigma(j),\sigma(i+2)}+D_{\sigma(i+2),\sigma(j-1)}+\\&c_{\sigma(j-1),\sigma(i+1)}+c_{\sigma(i+1),\sigma(j+1)}\big\},\\&\min_{0\le i\le j-2}\big\{F(i,S)+c_{\sigma(i),\sigma(j)}+\\&c_{\sigma(j),\sigma(i+1)}+D_{\sigma(i+1),\sigma(j-1)}+\\&c_{\sigma(j-1),\sigma(i+1)}\big\}\Big\}\end{aligned}$$

得到 $F(n,S)$ 后, 通过回溯机制, 可找到相应的移动.

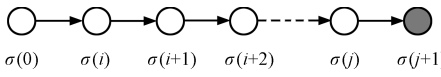


图 3　强化的 Dynasearch 邻域方式 1

Fig. 3　The first way to the enhanced dynasearch neighborhood

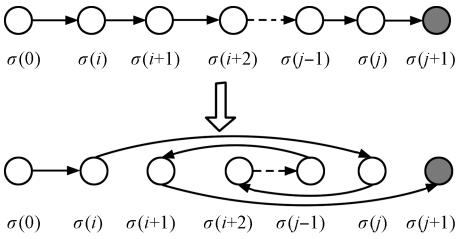


图 4　强化的 Dynasearch 邻域方式 2

Fig. 4　The second way to the enhanced dynasearch neighborhood

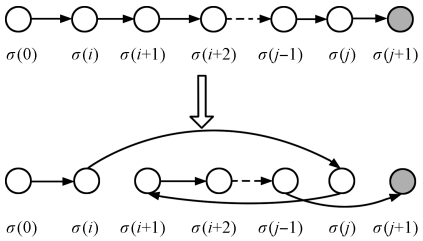


图 5　强化的 Dynasearch 邻域方式 3

Fig. 5　The third way to the enhanced dynasearch neighborhood

本文采用了上述 4 种不同的邻域结构. 如果一些移动被证明为不可行移动, 那么就放弃这样的移动. 针对问题的特点, 本文提出 Block 分区结构. 每个板卷在其有效的 Block 内进行不同的邻域搜索来获得问题的近似最好解, 可以降低搜索空间、加快搜索速度、提高算法的效率.

为了描述方便, 给出以下符号定义: 令 $dw(S_l)_{\min}$ 表示 S 内流向为 l 的板卷中的最小重量; $w(S)_{\max}$ 表示 S 内板卷的最大宽度; $w(S)_{\min}$ 表示 S 内板卷的最小宽度; $Left(\Omega')$ 表示 Ω' 中的剩余板卷按照从宽到窄的顺序排列后的序列集合. 令 S' 表示对 S 内板卷进行移动后得到的新的酸轧生产调度板卷序列; $F(S)$ 表示 S 对应的目标函数值.

具体 Block 分区方法如下: 对 S 中板卷进行分区, 如图 6 所示. 把 S 中板卷分成 R 个 Block, 前 $R-1$ 个 Block 的宽度跨越均为 δ_1, 最后一个 Block 的宽度跨越小于等于 δ_1. $block_r^s$ 表示 S 内第 r 个 Blcok, $\forall r=1,2,\cdots,R$. 每一个 $block_r^s$ 的区域都是从 $block_{r-1}^s$ 的最后一个板卷的紧接板卷开始, 到宽度不小于 $w(S)_{\max}-r\delta_1$ 的最后一个板卷. 其中对于 $block_1^s$ 的区域是 S 的首板卷开始, 到宽度不小于 $w(S)_{\max}-\delta_1$ 的最后一个板卷; 对于 $block_R^s$ 的区域是从 $block_{R-1}^s$ 中最后一个板卷的紧接板卷开始到 S 中的最后一个板卷. R 确定方法如下:

$$R=\lceil\frac{(w(S)_{\max}-w(S)_{\min})}{\delta_1}\rceil \tag{18}$$

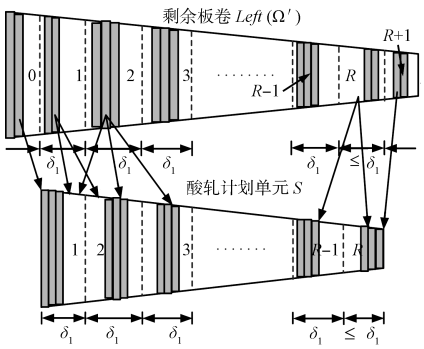


图 6　S 和 $Left(\Omega')$ 中板卷区域划分结构

Fig. 6　The structure of blocks in S and $Left(\Omega')$

对应上述对 S 中板卷分区规则, 相应地将 Ω' 中的剩余板卷 $Left(\Omega')$ 也分成 $R+2$ 个 Block, 划分方法与 S 板卷分区方式类似, 如图 6 所示. $block_r^{Left}$ 表示 $Left(\Omega')$ 中第 r 个 Block, $\forall r=0,1,\cdots,R,R+1$. 具体划分如下: $block_0^{Left}$, 该区域是从 $Left(\Omega')$ 的第一个板卷开始到宽度为 $w(S)_{\max}+\delta_2$ 的最后一个板卷; $block_r^{Left}$, $\forall r=1,\cdots,R-1$, 该区域是从 $block_{r-1}^{Left}$ 的最后一个板卷的紧接板卷开始, 到宽度为 $w(S)_{\max}-r\delta_1$ 的最后一个板卷; $block_R^{Left}$, 该区域是从 $block_{R-1}^{Left}$ 中最后一个板卷的紧接板卷开始到宽度为 $w(S)_{\min}-\delta_2$ 的最后一个板卷; $block_{R+1}^{Left}$, 该区域是从 $block_R^{Left}$ 中最后一个板卷的紧接板卷开始到 $Left(\Omega')$ 中的最后一个板卷.

Block 分区结构具有以下性质, 这些性质能降低邻域搜索空间:

性质 1. 对于 $block_r^S$, $\forall r=1,2,\cdots,R$ 内任意板卷 i, 只有在其所在 $block_r^S$ 内以及相邻 $block_j^S$ $\forall j=r-1,r,r+1$ 区域内移动, 才能保证获得可行解.

证明. 假设该性质不成立, 那么必然至少存在板卷 $i\in block_r^S$ 移动至 $block_j^S$, $\forall j\neq r-1,r,r+1$ 区域内, 使得 $F(S')<\infty$. 那么必然至少存在板卷 $k,q\in block_j^S$ 为板卷 i 移至新位置后的紧前与紧后板卷, 且 $c_{ki}<\infty$, $c_{iq}<\infty$. 存在以下 3 种情况:

1) 当 $i\in block_1^S$ 时, 板卷 i 移动到 $block_j^S$, $\forall j=3,4,\cdots,R$, 由 Block 划分规则可知, $w_k,w_q\in[w(S)_{\max}-(j+1)\delta_1+\delta_2,w(S)_{\max}-(j-1)\delta_1)$, $w(S)_{\max}-\delta_1\leq w_i\leq w(S)_{\max}$. 则有 $w_i-w_q\geq w(S)_{\max}-\delta_1-(w(S)_{\max}-(j-1)\delta_1)=(j-2)\delta_1\geq\delta_1, w_k-w_i\leq 0$ 且 $w_i-w_k\geq\delta_1$. 这违反模型中约束 (10) 和 (11), 此时 $c_{ki}=\infty$, $c_{iq}=\infty$, 必有 $F(S')=\infty$, 这与假设 $F(S')<\infty$ 相矛盾.

2) 当 $i\in block_R^S$ 时, 板卷 i 移动到 $i\in block_j^S$, $\forall j=1,2,\cdots,R-2$, 由 Block 划分规则可知, $w(S)_{\min}\leq w_i<w(S)_{\max}-(R-1)\delta_1$, $w_k,w_q\in[w(S)_{\max}-j\delta_1,w(S)_{\max}-(j-1)\delta_1]$. 则有 $w_q-w_i\geq w(S)_{\max}-(j+1)\delta_1+\delta_2-(w(S)_{\max}-(R-1)\delta_1)=(R-2-j)\delta_1+\delta_2\geq\delta_2$, 且 $w_i-w_q<0$ 这违反模型中约束 (11), 此时 $c_{iq}=\infty$, 必有 $F(S')=\infty$, 这与假设 $F(S')<\infty$ 相矛盾.

3) 当 $i\in block_r^S$ 时, $\forall r=2,3,\cdots,R-1$, 板卷 i 移动到 $block_j^S$, $\forall j\neq r-1,r,r+1$. 由 Block 划分规则可知, $w_i\in[w(S)_{\max}-r\delta_1,w(S)_{\max}-(r-1)\delta_1]$, 板卷 k, q 的宽度 $w_k,w_q\in[w(S)_{\max}-(j+1)\delta_1+\delta_2,w(S)_{\max}-(j-1)\delta_1]$. 当板卷 i 移动到 $block_j^S$, $\forall j=1,2,\cdots,r-2$ 时, 则有 $w_q-w_i\geq w(S)_{\max}-(j+1)\delta_1+\delta_2-(w(S)_{\max}-(r-1)\delta_1)=(r-2-j)\delta_1+\delta_2\geq\delta_2$, 且 $w_i-w_q<0$, 这违反模型中约束 (11), 此时 $c_{iq}=\infty$, 必有 $F(S')=\infty$, 这与假设 $F(S')\leq F(S)<\infty$ 相矛盾; 当板卷 i 移动到 $block_j^S$, $\forall j=r+2,r+3,\cdots,R$ 时, 则有 $w_i-w_q\geq w(S)_{\max}-r\delta_1-(w(S)_{\max}-(j-1)\delta_1)=(j-1-r)\delta_1\geq\delta_1$, 这违反模型中约束 (10), $c_{iq}=\infty$, 必有 $F(S')=\infty$, 这与假设 $F(S')<\infty$ 相矛盾.

综上各种情况均与假设 $F(S')\leq F(S)<\infty$ 相矛盾. 因而假设不成立, 故性质 1 成立. □

性质 2. 对于 $Left(\Omega')$ 中任意板卷 $i\in block_r^{Left}$, $\forall r=0,1,\cdots,R,R+1$, 只有按照下列条件插入 (Insert) 到 S 中, 才能保证获得可行解:

1) 板卷 $i\in block_0^{Left}$, 插入到 $block_1^S$ 的第一个板卷前;

2) 板卷 $i\in block_{R+1}^{Left}$, 插入到 $block_R^S$ 的最后一个板卷后;

3) 板卷 $i\in block_1^{Left}$, 插入到 $block_j^S$, $j=1,2$ 区域中;

4) 板卷 $i\in block_R^{Left}$, 插入到 $block_j^S$, $j=R-1,R$ 区域中;

5) 板卷 $i\in block_r^{Left}$, $\forall r=2,3,\cdots,R-1$, 插入到 $block_j^S(j=r-1,r,r+1)$ 区域中.

证明. 同理性质 1, 根据模型中不等式 (11) 可以得出性质 2 的结论. □

性质 2 中的 3) ∼ 5) 给出的有效区域同样适用于 Exchange 操作. 同理, 对于 S 内任意板卷 $i\in block_r$, $\forall r=1,2,\cdots,R$ 在 S 内进行 Inner-insert 操作, 向前插入时只有在宽度范围小于 $w_i+\delta_2$ 的区域内移动, 才保证获得可行解; 向后插入时只有在宽度范围大于 $w_i-\delta_2$ 的区域内移动, 才能保证得到可行解. 该结论降低了 Dynasearch 邻域中动态规划方法搜索 Inner-insert 邻域的搜索范围.

上述性质和结论表明, 对于 S 和 $Left(\Omega')$ 中的任意板卷对应当前解 S 中的位置都有一些有效位置和一些非有效位置, 在非有效区域内的移动是不能改善目标函数的, 所以在搜索时可以忽略这些非有效区域, 提高搜索效率.

基于上述性质 1 和性质 2, 本文进一步提出加快混合算法速度的加速策略. 令 $[\beta_r^1,\beta_r^2]$ 表示 $block_r^S$ 的范围, 其中 β_r^1 为 $block_r^S$ 中的首板卷, β_r^2 为 $block_r^S$ 中最后一个板卷.

1) Insert 操作

根据性质 2 可知, $block_0^{Left}$ 中的板卷仅能插入 S 中的首板卷前, 因此 $block_0^{Left}$ 中的板卷 i 只有满足 $w_i<w_{\sigma(1)}+\delta_1$ 才能插入. 由于 $Left(\Omega')$ 中的板卷是按照从宽到窄的顺序排列的, 因此搜索时, 从 $block_0^{Left}$ 中的最后一个板卷开始, 直到遇到第一个

宽度使得 $w_i \geq w_{\sigma(1)} + \delta_1$ 的板卷结束. 同理, 对于 $block_{R+1}^{Left}$ 中的板卷, 从 $block_{R+1}^{Left}$ 中的首板卷开始搜索, 直到遇到第一个宽度 $w_i \leq w_{\sigma(n-1)} - \delta_1$ 的板卷结束.

对于任意板卷 $i \in block_r^{Left}$, $\forall r = 1, 2, \cdots, R$, 令板卷 $k, q \in S$ 分别为 i 插入到 S 后的紧前和紧后板卷: 1) 如果 $\mid w_i - w_{\beta_r^1} \mid \leq \mid w_i - w_{\beta_r^2} \mid$, 那么从 $block_r^S$ 中的首板卷 β_r^1 开始, 在性质 2 中规定的有效 Block 区域内分别进行向前搜索和向后搜索, 如图 7 所示. 基于约束 (11), 向前搜索时直到遇到第一个使得 $w_q - w_i \geq \delta_2$ 的板卷 q 时, 停止搜索. 同理, 向后搜索时直到遇到第一个使得 $w_i - w_k \geq \delta_2$ 的板卷 k 时, 停止搜索; 2) 如果 $\mid w_i - w_{\beta_r^1} \mid \geq \mid w_i - w_{\beta_r^2} \mid$, 那么从 $block_r^S$ 中板卷 β_r^2 开始, 在性质 2 中规定的有效 Block 区域内分别进行向前搜索和向后搜索, 具体操作方式同 1).

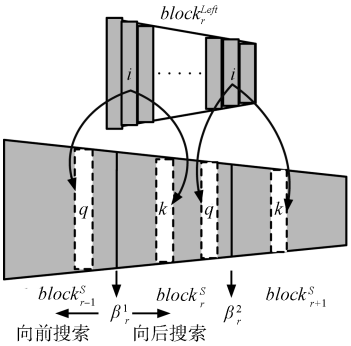


图 7　Insert 加速搜索示意图

Fig. 7　Illustration of the speedup for insert operator

2) Exchange 操作

加速策略 1: 当 S 中某流向的板卷总量与该流向的需求接近时, 只能与同流向的板卷进行交换 (Exchange) 操作, 否则不满足约束 (9).

加速策略 2: 与 Insert 操作加速策略类似. $block_0^{Left}$ 中的板卷仅能与 S 中的首板卷进行 Exchange. 因此搜索时, 从 $block_0^{Left}$ 中的最后一个板卷开始, 直到遇到第一个宽度为 $w_i \geq w_{\sigma(2)} + \delta_1$ 的板卷 i 时结束; 同理, 对于 $block_{R+1}^{Left}$ 中的板卷仅能与 S 中板卷 $\sigma(n-1)$ 进行 Exchange. 从 $block_{R+1}^{Left}$ 的首板卷开始搜索, 直到遇到第一个宽度使得 $w_i \leq w_{\sigma(n-2)} - \delta_1$ 的板卷 i 时结束; 对于任意板卷 $i \in block_r^{Left}$, $\forall r = 1, 2, \cdots, R$, 令板卷 $k, q \in S$ 分别为 i 交换到 S 后的紧前和紧后板卷. 在 $block_r^S$ 中寻找能使 $\mid w_i - w_p \mid$ 最小的板卷 p, 从 p 开始在性质 2 规定的有效 Block 区域内分别进行向前搜索和向后搜索, 如图 8 所示. 基于约束 (11), 向前搜索时直到遇到第一个使得 $w_q - w_i \geq \delta_2$ 的板卷 q 时, 停止. 同理, 向后搜索时直到遇到第一个使得 $w_i - w_k \geq \delta_2$ 的板卷 k 时, 停止.

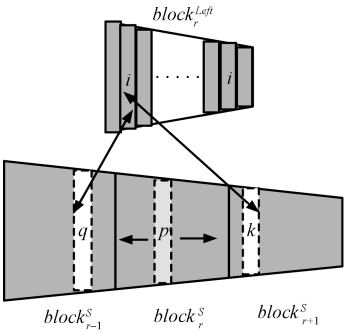


图 8　Exchange 加速搜索示意图

Fig. 8　Illustration of the speedup for exchange operator

此外, 对于 Delete 操作, 根据模型中的不等式 (9), 可以推断出: 如果 S 内流向为 l 板卷的总重量 $D(S_l)$ 满足 $D(S_l) - f_l \leq dw(S_l)_{\min}$, 那么对流向为 l 的板卷进行 Delete 操作无法获得可行解. 也就是说, 当酸轧生产调度内某流向的板卷总量与该流向的需求接近时, 不能再通过删除该流向板卷的方式改进最好解.

2.3　禁忌表

针对问题特点以及邻域结构的不同, 本文提出的禁忌搜索算法采用 2 个禁忌表. 禁忌表 TL1 记录由 Delete 邻域、Insert 邻域和 Exchange 邻域产生的移动. 禁忌表 TL2 记录由强化 Dynasearch 邻域产生的一系列无关移动. TL1 和 TL2 由两部分构成: 禁忌的移动以及该移动对应的邻域结构类型. 对于 TL1 存在的邻域结构类型有 Delete、Insert 和 Exchange, 对于 TL2 存在的邻域结构类型有 Swap 和 Inner-insert.

一旦一个移动被接受, 那么它的反移动就会被记录到禁忌表中. 对于 TL1 禁忌表采用固定长度, 在禁忌搜索的每次迭代中选择上述 3 种邻域中最好的一个移动, 如果这个最好的移动是被禁忌的, 就从其他的邻域中选择不被禁忌的最好的移动. 由于 Dynasearch 算法是一个下降邻域搜索方法, 当无法找到更好解时, Dynasearch 算法停止. 因此, 对于 TL2 禁忌表本文采用动态的禁忌表策略. TL2 禁忌表的长度取决于禁忌搜索每次迭代时第一次 Dynasearch 移动产生的一系列无关移动的个数.

如果某个移动被禁, 但该移动能改进当前最好解, 这时该移动可以破禁.

2.4　长期记忆表

长期记忆表可以使得搜索不完全丧失以前搜索所具有的好特性, 同时还能避免陷入局部最优. 本文采用长期表形式为 (S^*, N^*, T^*), 其中 S^* 记录当前最好解序列, N^* 记录 3 个与当前最好解相关的最好未实施移动 (这 3 个移动分别对应 Delete 邻

域、Insert 邻域和 Exchange 邻域各自产生的一个最好未实施移动), T^* 记录与当前最好解序列相关的 TL1 禁忌表.

2.5 终止准则

本文采用 3 个终止准则:

1) 达到最大迭代次数时, 算法终止;

2) 达到最大未改进当前最优解的循环次数时, 算法终止;

3) 长期记忆表 N^* 为空时, 算法终止.

2.6 算法流程

求解带流向需求约束的酸轧生产调度问题的强化 Dynsearch & TS 算法, 具体步骤如下.

步骤 1. 运用第 2.1 节中所示的两阶段启发式生成初始解, 并设为当前最好解 S^{best} 和当前解 S^{cur}. 初始化相关参数, 最大迭代次数 $iter_{\max} = 200$, 最大未改进迭代次数 $Unimpiter_{\max} = 30$, 禁忌表 TL1 和 TL2 空, 禁忌表 TL1 的长度为 7, 记录未改进最优解的循环次数 $ret = 0$, 记录循环次数 $iter = 0$.

步骤 2. 令 $iter = iter + 1$, $ret = ret + 1$. 结合上述加速性质和结论, 采用 Delete 邻域、Insert 邻域和 Exchange 邻域类型进行一次禁忌搜索, 从这 3 种搜索中找出一个最好的未被禁忌的移动或者最好的被禁但其移动可以改进 S^{best} 的移动, 对当前解应用这个移动, 得到新的 S^{cur}. 将应用移动的相反移动记录在禁忌表中, 更新 TL1. 如果 $S^{cur} < S^{best}$, 则令 $S^{best} = S^{cur}$, $ret = 0$.

步骤 3. 对 S^{cur} 执行强化 Dynasearch 算法: 将第一次强化 Dynasearch 邻域移动产生的一系列无关移动的反移动记录在 TL2 中, 更新 S^{cur}; 对 S^{cur} 执行第二次强化 Dynasearch 邻域搜索, 搜索过程中选择未被禁移动或可破禁的移动构成一次 Dynasearch 移动, 执行该次强化 Dynasearch 移动得到新的 S^{cur}, 更新 TL2; 重复执行上述过程, 直到 S^{cur} 不再被改进为止. 强化的 Dynasearch 搜索算法终止时, 清空 TL2, 如果 $S^{cur} < S^{best}$, 则令 $S^{best} = S^{cur}$, $ret = 0$.

步骤 4. 如果 $ret < Unimpiter_{\max}$ 且 $iter < iter_{\max}$, 则转到步骤 2. 否则, 转到步骤 5.

步骤 5. 如果 $ret > Unimpiter_{\max}$ 且 $iter < iter_{\max}$, 如果 N^* 非空, 则令 $S^{cur} = S^*$, $TL1 = T^*$, $ret = 1$, $iter = iter + 1$, 从 N^* 中找出一个最好的的移动, 对当前解应用这个移动, 得到新的 S^{cur}, 更新 TL1. 往 N^* 添加 0 元素, 转至步骤 2. 否则, 转到步骤 6.

步骤 6. 如果 N^* 为空或 $iter \geq iter_{\max}$, 算法终止.

3 实验及算法性能比较

本文算法采用 C++ 编程, 并在 512 M RAM 的 IBM 机器运行. 测试实例是模拟实际酸轧生产情况随机产生数据. 为了验证算法的有效性, 对不同规模问题随机产生 10 组数据进行实验, 共有 60 组数据. 产生方法如下: 根据表 1 中板卷规格范围要求按照均匀分布的方式产生随机数据. 候选板卷个数为 300 个. 酸轧生产调度单元最大生产能力 $Q = \beta \sum_{i \in \Omega} d_i$, 最低生产需求 $B = \gamma Q$. 各流向的板卷需求 f_l 是在小于候选板卷中该流向板卷总量范围内随机产生, 要求 $\sum_{l \in L} f_l < 0.9Q$. 禁忌表 TL1 长度为 7.

表 1 酸轧板卷规格范围

Table 1 The ranges of gauges of the coils

属性	范围
宽度 (整数)	[1 000, 1 700]
入口厚度 (小数)	[2.0, 5.0]
出口厚度 (小数)	[0.5, 2.0]
钢级 (整数)	[1, 18]
重量 (小数)	[16.0, 32.0]
流向 (整数)	[1, 3]
切边标记 (整数)	[0, 1]

表 2 给出了本文所提出的强化 Dynsearch & TS 算法与一般 TS 算法 (邻域由 Delete、Add、Insert、Exchange、Swap 构成, 禁忌表固定长度) 的结果对比. 其中 $f^{\bullet}_{\rm obj}$ 表示方法 $\bullet$ 得到的目标函数值, $f^{\bullet}_{\rm trans}$ 表示方法 $\bullet$ 得到的过渡费用, $f^{\bullet}_{\rm weight}$ 表示方法 $\bullet$ 得到的选中板卷总重量, $Ratio = (f^{\rm TS}_{\rm obj} - f^{\rm EDTS}_{\rm obj})/f^{\rm TS}_{\rm obj}$, 时间单位为秒.

表 2 强化 dynsearch &TS 算法 (EDTS) 与一般 TS 算法结果比较

Table 2 Comparison of computational results between enhanced-dynsearch & TS and TS

参数			强化 Dynsearch & TS (EDTS)				一般禁忌搜索 (TS)				Ratio (%)
β	Q	γ	$f^{\rm EDTS}_{\rm trans}$	$f^{\rm EDTS}_{\rm weight}$	$f^{\rm EDTS}_{\rm obj}$	Time	$f^{\rm TS}_{\rm trans}$	$f^{\rm TS}_{\rm weight}$	$f^{\rm TS}_{\rm obj}$	Time	
0.3	2 060.6	0.5	1 642.36	1 785.58	1 916.88	25.72	1 644.38	1 740.13	1 964.25	28.2	2.41
		0.7	1 659.77	1 756.96	1 962.81	32.04	1 718.87	1 746.40	2 032.47	33	3.42
0.5	3 589.32	0.5	2 384.27	2 859.34	3 114.25	56.25	2 436.26	2 823.68	3 204.9	74.4	2.83
		0.7	2 553.37	3 391.25	2 751.37	70.2	2 610.71	3 305.25	2 894.78	102.2	4.95
0.8	5 771.48	0.5	3 249.29	3 810.05	5 210.63	127.3	3 419.87	3 706.90	5 484.45	157.38	4.99
		0.7	4 285.53	5 322.73	4 734.31	165	4 363.12	5 249.07	4 885.53	185.8	3.1
平均											3.62

表 3 给出了表 2 中不同算法目标函数中分项归约值. 其中 $Rf_{\text{weight}} = f_{\text{weight}}^{\text{EDTS}}/f_{\text{weight}}^{\text{TS}}$, $Rf_{\text{trans}} = f_{\text{trans}}^{\text{EDTS}}/f_{\text{trans}}^{\text{TS}}$.

表 3　强化 Dynsearch & TS 算法与一般 TS 算法分项归约

Table 3　Results of ratios of EDTS to TS

β	Q	γ	Rf_{trans}	Rf_{weight}
0.3	2 060.6	0.5	0.998	1.026
		0.7	0.965	1.006
0.5	3 589.32	0.5	0.978	1.012
		0.7	0.978	1.026
0.8	5 771.48	0.5	0.95	1.027
		0.7	0.982	1.014
	平 均		0.975	1.0185

由表 2 和表 3 结果表明: 本文所提出的强化 Dynsearch & TS 算法得到的目标函数要优于一般禁忌搜索算法, 并且平均改进量为 3.62 %. 这说明 Dynsearch 算法每次迭代产生的一系列无关移动, 加速了搜索, 同时使解更逼近了局域最优解. 强化 Dynsearch & TS 算法的运行时间比一般禁忌搜索算法时间短, 说明算法有效地利用了 Dynasearch 邻域搜索的特点, 大大缩短了运行时间.

4　结束语

本文针对带有流向需求约束的酸轧实际生产调度问题进行了研究, 建立了整数规划模型. 针对问题的 NP 难和复杂的模型结构等特征, 提出了一种嵌入强化 Dynasearch 算法的禁忌搜索混合算法进行近似求解. 通过模拟实际生产情况随机产生的数据测试, 得出以下结论: 1) 与一般禁忌搜索算法比较, 本文所提出的强化 Dynsearch & TS 算法求解效果明显优于一般 TS 算法; 2) 强化 Dynsearch & TS 算法有效地利用了 Dynasearch 邻域搜索的特点, 运行时间大大缩短. 上述结论验证了该算法在解决此类问题的有效性, 它能够适应酸轧实际生产中规模大、约束复杂的需要, 能够在合理的计算时间内得到满意的酸轧生产调度.

References

1 Lopez L, Carter M W, Gendreau M. The hot strip mill production scheduling problem: a tabu search approach. *European Journal of Operational Research*, 1998, **106**(2-3): 317−335

2 Zhao J, Wang W, Liu Q L, Wang Z G, Shi P. A two-stage scheduling method for hot rolling and its application. *Control Engineering Practice*, 2009, **17**(6): 629−641

3 Tang L X, Liu J Y, Rong A Y, Yang Z H. A multiple traveling salesman problem model for hot rolling scheduling in Shanghai Baoshan Iron and Steel Complex. *European Journal of Operational Research*, 2000, **124**(2): 267−282

4 Tang L X, Wang X P. Iterated local search algorithm based on very large-scale neighborhood for prize-collecting vehicle routing problem. *The International Journal of Advanced Manufacturing Technology*, 2006, **29**(11-12): 1246−1258

5 Okano H, Davenport A J, Trumbo M, Reddy C, Yoda K, Amano M. Finishing line scheduling in the steel industry. *IBM Journal of Research and Development*, 2004, **48**(5-6): 811−830

6 Zhao Jun, Liu Quan-Li, Wang Wei. Models and algorithms of production scheduling in tandem cold rolling. *Acta Automatica Sinica*, 2008, **34**(5): 565−573

7 Glover F. Tabu search — part I. *ORSA Journal of Computing*, 1989, **1**(3): 190−206

8 Nowicki E. The permutation flow shop with buffers: a tabu search approach. *European Journal of Operational Research*, 1999, **116**(1): 205−219

9 Grabowski J, Wodecki M. A very fast tabu search algorithm for the permutation flowshop problem with makespan criterion. *Computers and Operations Research*, 2004, **31**(11): 1891−1909

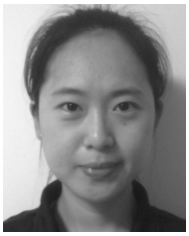
10 Congram R K. Polynomially Searchable Exponential Neighbourhoods for Sequencing Problems in Combinatorial Optimization [Ph. D. dissertation]. University of Southampton, UK, 2000

11 Li Yan-Feng, Li Jun, Zhao Da. Dynasearch algorithms for solving time dependent traveling salesman problem. *Control and Decision*, 2009, **24**(2): 274−278
(李妍峰, 李军, 赵达. 动态搜索算法求解时间依赖型旅行商问题研究. 控制与决策, 2009, **24**(2): 274−278)

12 Congram R K, Potts C N, van de Velde S L. An iterated dynasearch algorithm for the single-machine total weighted tardiness scheduling problem. *INFORMS Journal on Computing*, 2002, **14**(1): 52−67

13 Grosso A, Croce F D, Tadei R. An enhanced dynasearch neighborhood for the single-machine total weighted tardiness scheduling problem. *Operations Research Letters*, 2004, **32**(1): 68−72

唐立新　东北大学物流优化与控制研究所、辽宁省制造系统与物流优化重点实验室教授. 主要研究方向为生产调度、物流与供应链管理和组合最优化. 本文通信作者.
E-mail: qhjytlx@mail.neu.edu.cn
(**TANG Li-Xin**　Professor at Liaoning Key Laboratory of Manufacturing System and Logistics, and the Logistics Institute, Northeastern University. His research interest covers production scheduling, logistics and supply chain management, and combinational optimization. Corresponding author of this paper.)



赵　任　东北大学物流优化与控制研究所博士研究生. 主要研究方向为智能优化和钢铁企业生产计划与调度.
E-mail: zhaoren113@126.com
(**ZHAO Ren**　Ph. D. candidate at the Logistics Institute, Northeastern University. His research interest covers intelligent algorithm and production plan and scheduling in the iron steel industry.)