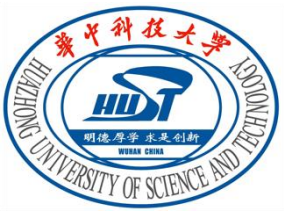

中国研究生创新实践系列大赛

“华为杯”第十九届中国研究生

数学建模竞赛

学　　校	杭州电子科技大学
参赛队号	22103360057
队员姓名	1.陈盟昊
	2.张琳翊
	3.陈利超

中国研究生创新实践系列大赛

“华为杯”第十九届中国研究生

数学建模竞赛

题 目 基于改进贪婪算法的PBS优化调度问题

摘　　要：

汽车制造厂在制造过程中需要对车身的顺序进行调整，以满足总装车间的需求，通常在两车间之间建造用于调整车身顺序的区域 PBS。本文针对 PBS 调序问题以总得分最大为目标并满足 PBS 约束建立混合整数规划模型，对问题一、二分别设计贪婪-模拟退火算法、贪婪算法进行求解，两个附件求解结果均在可行解空间中找到了局部最优解。

针对问题一，本文将车身动力属性间隔要求、车身驱动属性等比例要求、返回道次数最少、调度时间最短的多目标通过线性加权转化为以总得分最大作为目标，考虑移动顺序、优先处理原则等 PBS 约束，建立多约束条件下的**混合整数规划模型**。贪婪算法设计时，分析调整出车顺序可从两个方面进行，一方面是对每个车身的**进车道序号进行选择**，另一方面是对每个车身是否**返回**以及最多返回几次进行判断。为求得更优的解，通过模拟退火在贪婪得到的解的邻近解中寻找最优解。最终解得附件 1 和附件 2 中，**贪婪-模拟退火算法**的**总得分**为 15.394 分，38.794 分；**目标函数 1 得分**为-87 分，-28 分；**目标函数 2 得分**为 81 分，81 分；**目标函数 3 得分**为 78 分，77 分；**目标函数 4 得分**为 102.94 分，102.94 分；**返回道使用次数**为 22 次，22 次；**总调度时间**为 2640 秒，2640 秒。**原始序列**的**总得分**为 13.098 分，35.098 分；**优化比例**为 17.5%，10.53%。将设计的算法结果与原序列直出结果进行对比后发现设计的算法**总得分更大**，从而可知返回道存在的重要性。由于两个附件的进车车身属性分布差异较大，设计的算法在两个附件中的最终的求解结果均较优，证明算法具有一定的普适性，即可对任意进车顺序优化调度。

针对问题二，在问题一的基础上去除横移机的优先处理约束，即横移机可以自主选择接车，建立新的**混合整数规划模型**。求解过程中，去除优先处理约束使出车序列的灵活性更大，通过调整出车序列能够更好地满足目标函数，对**贪婪算法**进行约束限制的调整。贪婪算法设计时分析与问题一相同，都是通过两个方面调整出车序列，第一方面对于进车序列车道的选择，由于送车横移机能够自主选择各车道上第一停车位的车身，故考虑车道的**专车道专用**，即按照车身的属性根据横移机移动时间进行规定车道划分；第二方面每个车身返回时按照尽量满足目标函数的顺序进行判断是否返回，并通过算法设计尽量少走返回道。最终解得附件 1、附件 2，**贪婪算法总得分**为 26.808 分，48.771 分；**目标函数 1 得分**为-69 分，-13 分；**目标函数 2 得分**为 87 分，86 分；**目标函数 3 得分**为 91 分，91 分；**目标函数 4 得分**为 101.08 分，99.71 分；**返回道使用次数**为 9 次，9 次；**总调度时间**为 2826 秒，2963 秒。**原始序列**的**总得分**为 13.098 分，35.098 分；**优化比例**为 104.7%，39.0%。可知问题二中贪婪算法结果比原始序列更高。

本文设计的算法对不同进车序列均可进行调度优化，具有普适性。

关键字：PBS 优化调度；混合整数规划模型；贪婪算法；模拟退火算法

目录

一、 问题重述

1.1 问题背景

随着我国经济的快速增长，人民对于私家汽车的需求也日益增长，当前的汽车行业市场需求已从传统的大批量生产模式向着多样化、品种多、订单批量小等生产模式变化。汽车制造车间[2]由多个车间构成，各个车间之间存在前后需求依赖[1]，且各个车间由于成本等原因，有着不同的生产序列偏好。因此车间生产计划的制订对汽车制造业生产率的提高极为重要。

由于各车间生产偏好的不同导致在连续生产过程中无法按照同一序列，本题以生产序列差异较大的涂装车间与总装车间为例，在两个车间之间建立一个用以调整生产序列以适应总装车间生产需要的缓存区，即 PBS[7]（Painted Body Store，汽车制造涂装-总装缓存调序区），用以提高总装车间的生产效率，并降低产品的成本。

1.2 问题的提出

本次考虑一天内安排上线生产的车身共 318 辆，对于这些车身，存在 2 大类车型，还有混动与燃油的动力区别和两驱与四驱的驱动区别，通过 PBS 将从涂装-PBS 出车口的出车序列调整为更适合总装车间的 PBS-总装进车口的进车序列。其中 PBS 的构造如图 1 所示。

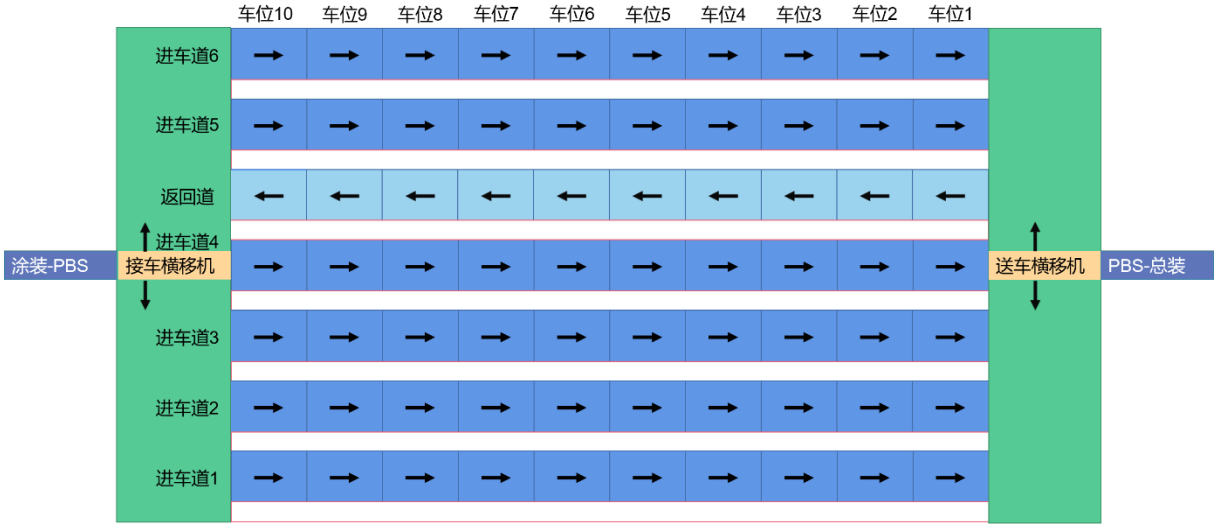

图 1. PBS 结构示意图

PBS 由 1 个涂装-PBS 出车口、1 个接车横移机、6 条进车道、1 个返回道（宽度均为 2 米，各邻车道间隔为 1 米）、1 个送车横移机和 PBS-总装接车口组成。每条车道有 10 个停车位，从某一停车位移动到下一停车位的时间为 9 秒。接车横移机与送车横移机每次运送车身均需从中间初始位置出发，运送完成后返回至初始位置，即为横移机完成一次任务的时间段，横移机从某一车道横移至相邻车道所需时间为 3 秒，且横移机运动时的速度保持一致。

接车横移机从涂装-PBS 出车口或返回道的停车位 10 接到车身，并运送到适合进车道的停车位 10，其中涂装-PBS 出车口的出车顺序如附件 1 与附件 2 所示。

送车横移机从进车道的停车位 1 接到车身，并运送到 PBS-总装接车口或返回道的停车位 1。

对于问题一，使用附件 1 和附件 2 中给出的涂装-PBS 出口车身序列数据，分别制定出满足 PBS 约束规则 1）~12）和时间约束规则的缓存区调度方案（给出记录各个车

身在 PBS 内按照时间轴经过区域的区域代码表），以保证最终的 PBS-总装接车口的接车序列满足尽量多的混动车型间隔 2 台非混动车型、四驱车型与两驱车型尽可能倾向 1：1 出车、返回道尽量少被使用和总调度时间（第一辆车身从涂装-PBS 送车口进到最后一辆车身从 PBS-总装接车口出所经历的时间）尽可能短这四种权重不同的要求。

问题二在问题一的基础上，去除（1）接车横移机优先处理返回道 10 停车位上的车身，（2）送车横移机优先处理各车道中最先到达停车位 1 的车身这两个约束，即使得横移机可选择需运送的车身，建立横移机可选择的 PBS 调度优化模型，使得调度结果满足权重不同的各个优化目标。

二、 问题分析

2.1 问题一的分析

问题一要求制定出 PBS 区的生产调度优化方案，以使送车次序更符合要求，送车次序要求包括满足车身动力属性间隔、车身驱动属性等比例、返回道尽量少使用和总调度时间尽可能短。将这四个目标通过设置权重合并的总得分作为目标。对 12 条 PBS 约束条件进行分析，将可以合并的约束进行合并，建立满足各约束条件的 PBS 的生产调度混合整数规划模型。

考虑 PBS 进车序列车道的选择、出车序列的安排以及返回道上限等约束按照目标函数的优先级进行贪婪算法的设计，从而得到一个满足要求的可行解，在该可行解的基础上使用设计的模拟退火算法寻找该可行解附件邻近解中的最优解，该局部最优解即为 PBS 生产调度优化后的调度方案，并与全部从进车道 4 进入并直接输出的结果进行比较，分析优化目标的满足情况。

2.2 问题二的分析

问题二在问题一的基础上，去除 PBS 约束中关于接车横移机空闲状态下优先处理返回道 10 停车位上的车身与送车横移机空闲状态下优先处理最先到达各进车道停车位 1 上的车身这两个约束，即接车横移机可对来自涂装-PBS 出车口和返回道的车身进行有选择性的运送，送车横移机可对在工作状态下到达进车道停车位 1 上的车身在送车横移机空闲时按照需要有选择性的运送。因此问题二的模型在问题一的模型的基础上除去这两个约束，即可建立满足空间约束与时间约束的横移机自主选择的 PBS 的生产调度混合整数规划模型。

首先根据附件 1 与附件 2 所给出车次序，根据车身属性将所给车身分为 3 种类型：混动两驱、燃油两驱、燃油四驱，据此对进车道进行分配，使其满足“专车道专用”。按照车身的三种类型在 PBS 出车序列的排布中尽可能满足总得分最大的目标。按照目标函数设计出满足要求的贪婪算法，得到最终结果的调度表，并与全部从进车道 4 进入并直接输出的结果进行比较，分析优化目标的满足情况。

三、 模型假设

假设 1：PBS 车间运行顺序完全按照规定状态转移；
假设 2：车身由横移机卸载到停车位的时间与车身由停车位装载到横移机的时间忽略；
假设 3：车身由出车口装载到横移机的时间与车身由横移机卸载到接车口的时间忽略；
假设 4：车身停留在某个区域代码的时间为 0 就转移到下一个区域代码，则该区域代码可忽略。

四、 符号说明

符号	说明	单位
T	总调度时间	秒
t	第t时刻（$t = 1,2,3 \dots T$）	秒
i	PBS 进车顺序中的第i辆车身($i = 1,2,3 \dots 318$)	/
I	PBS 进车顺序的集合	/
j	PBS 出车顺序中的第j辆车身$j \in J$	/
J	PBS 出车顺序的集合（向量）	/
x_{it}	第i辆车身在t时刻所在的 PBS 对应的区域代码	/
$back_{it}$	第i辆车身在t时刻是否返回	/
U	出车顺序的车身动力属性向量	/
u_j	PBS 出车顺序中第j辆车身的动力属性$u_j \in U$	/
H	PBS 出车顺序中混动车的序号集合$H \subset J$	/
V	出车顺序的车身驱动属性向量	/
v_j	PBS 出车顺序中第j辆车身的驱动属性$v_j \in V$	/
$J = (U,V)$	出车顺序中车身包含动力和驱动两种属性	/
$\delta(u_j, u_{j+1}, u_{j+2}, u_{j+3})$	PBS 出车顺序中对于混动车的惩罚分数$j \in H$	/
$block(m)$	出车序列划分的第m块的出车序列集合 $block(m) \subset J$	/
M	出车序列划分的总块数	/
MM	表示出车序列划分的总块数为奇偶时分别对应的集合	/
mm	表示出车序列划分的总块数M是奇数时为 0 否则为 1	
$\varepsilon(m)$	表示第m块和第$m+1$块驱动属性变化的惩罚分数	/
C	出车队列的长度	/
A	PBS 区域代码集合	/
A'	PBS 区域代码除 0 和 3 的集合	/
B	PBS 区域代码除 0、1、2、3 以及每个车道的第一个停车位的集合	/
D	B集合除返回道的停车位	/
Q_a	区域代码a所对应下一时刻的区域代码集合	/
z_{1t}	接车横移机是否空闲	/
z_{2t}	送车横移机是否空闲	/
L	车道内在一个停车位停留的时间长度	秒
l	车道内停留时刻（l=1,2,3 … L）	秒
E	横移机每次移动的时间长度	秒
e	横移机每次移动时刻（e=1,2,3 … E）	秒
S	横移机移动时间集合	秒

五、 模型的建立与求解

5.1 问题一模型的建立与求解

5.1.1 问题一模型的建立

数据处理：

（1） PBS 区域代码重排：

将停车位 1-10 重新编码为 0-9，则第一车道上的第一停车位由代码 11 更改为代码 10，第一车道上的第十停车位由 110 更改为代码 19，以此类推。

（2）动力：

当该车身动力属性为燃油时，数据替换为 0，否则替换为 1。

（3）驱动：

当该车身驱动属性为二驱时，数据替换为 2，否则替换为 4。

参数说明：

x_{it}表示 PBS 进车顺序中第i辆车身在第t时刻所在 PBS 区域的代码，出车顺序j可通过该矩阵计算得到，如图 2：

i \ t	112	113	114	115	116	117	118	119
4	41	3	3	3	3	3	3	3
5	31	31	31	31	2	2	2	3

图 2. PBS 进车顺序和出车顺序序号对应示意图

每一辆车出 PBS 的标志均为第一次出现$x_{it}=3$时，则可按照这个思路从x_{it}构成的矩阵中读取出车顺序，这里需要满足：

$$t_{(J(1))}^{x_{J(1)t_{first}}=3} < t_{(J(2))}^{x_{J(2)t_{first}}=3} < \cdots < t_{(J(318))}^{x_{J(318)t_{first}}=3}, \forall j \in J, J(j) = f\big(I(i)\big) \tag{1}$$

假设只有两辆车身，车身序号为 4 和 5，调度时间从 112-119 秒时，可知进车顺序为 4-5，则出车顺序按照图示为 4-5。

目标函数：

目标 1：出车顺序车身需满足动力属性间隔要求，该目标权重为 0.4

$$u_j = \begin{cases} 0 & ,\text{第 j 辆车身的动力属性为燃油} \\ 1 & ,\text{第 j 辆车身的动力属性为混动} \end{cases} \tag{2}$$

其中，u_j为出车序列中第j辆车身的动力属性。

$$\delta\big(u_j, u_{j+1}, u_{j+2}, u_{j+3}\big) = \begin{cases} 0 & ,u_j = u_{j+3}\text{且}u_{j+1} = u_{j+2} \ \ \big(j \in H, u_j \in U\big) \\ 1 & ,else \ \ \ (j \in H) \end{cases} \tag{3}$$

其中，$\delta\big(u_j, u_{j+1}, u_{j+2}, u_{j+3}\big)$为出车序列中对于每一个混动车的惩罚分数。

$$Z_1 = 100 - \sum_{j \in H} \delta\big(u_j, u_{j+1}, u_{j+2}, u_{j+3}\big) \tag{4}$$

目标 2：出车顺序车身需满足驱动属性等比例要求，该目标权重为 0.3

$$v_j = \begin{cases} 2 & ,\text{第}j\text{辆车身的驱动属性为二驱} \\ 4 & ,\text{第}j\text{辆车身的动力属性为四驱} \end{cases} \tag{5}$$

其中，v_j为出车序列中第j辆车身的驱动属性。

$$\begin{aligned} &if\ v_j = v_{j+1} = \cdots = v_{j+length}, v_j \neq v_{j-1}, v_{j+length+1} \neq v_j \\ &block(m) = \{v_j, v_{j+1}, \dots, v_{j+length}\}, block(m) \subset J, v_j \in V \\ &\varepsilon(m) = \begin{cases} 0 & ,else \\ 1 & ,length_{block(m)} \neq length_{block(m+1)} \end{cases} \quad m \in MM \end{aligned} \tag{6}$$

其中，$\varepsilon(m)$表示第m块和第$m+1$块驱动属性变化的惩罚分数。

$$MM = \begin{cases} \{m = 1,3,5, \dots M\}, M\text{是奇数} \\ \{m = 1,3,5, \dots (M-1)\}, M\text{是偶数} \end{cases} \tag{7}$$

$$mm = \begin{cases} 1, M\text{是奇数} \\ 0, M\text{是偶数} \end{cases} \tag{8}$$

其中，M表示将出车序列划分的总块数。当M是奇数时，$m = 1,3,5, \dots M$；当M是偶数时，$m = 1,3,5, \dots (M-1)$。MM为出车序列划分的总块数为奇偶时分别对应的集合，mm表示出车序列划分的总块数M是奇数时为 0 否则为 1。

$$Z_2 = 100 - \sum_{m \in MM} \varepsilon(m) - 1 \times mm \tag{9}$$

目标 3：返回道尽量不使用，该目标权重为 0.2

$$back_{it} = \begin{cases} 0, & else \\ 1, & x_{i(t-1)} \neq 70 \text{ 且} x_{it} = 70 \end{cases} \tag{10}$$

其中，$back_{it}$表示进车顺序第i辆车身是否在t时刻刚好进入返回道的第一个车位，如果是则将为 1，否则为 0。

$$Z_3 = 100 - \sum_{i=1}^{318} \sum_{t=1}^{T} back_{it} \tag{11}$$

目标 4：总调度时间尽量短，该目标权重为 0.1

$$Z_4 = 100 - (0.01 \times T - 9C - 72) \tag{12}$$

综上，可将各目标函数通过线性加权的方式将得到总得分的目标：

$$Z = \max(0.4Z_1 + 0.3Z_2 + 0.2Z_3 + 0.1Z_4) \tag{13}$$

约束条件：

约束 1（PBS 约束说明 3、9）：某一时刻，一种区域代码下(除区域代码 0 与 3)只能存在一个车身

$$x_{it} \neq x_{i't}, if\ \forall t, i \neq i', x_{it}, x_{i't} \in A' \tag{14}$$

其中，i'为除i外任一进车序号，其中A'为 PBS 区域代码除 0 和 3 的集合。这样即满足 PBS 的时间位置原则。

约束 2（PBS 约束说明 1、2、12）：车身在进车道和返回道上的移动方向恒定

$$x_{it}=a, x_{i(t+1)}=P_a, if\, a\in A, P_a\in Q_a \tag{15}$$

其中，Q_a为区域代码a所对应下一时刻的区域代码集合，P_a为区域代码a所对应下一时刻的区域代码。这样即满足 PBS 的内部顺序原则。

约束 3（PBS 约束说明 6）：若返回道停车位 10 有车身，当接车横移机空闲时，则优先运送返回道停车位 10 上的车身

定义 0-1 变量z_{1t}：

$$z_{1t}=\begin{cases}0 & ,\forall i\in I, x_{it}\neq 1\\ 1 & ,else\end{cases} \tag{16}$$

其中，在t时刻，所有车身的区域代码均不为 1 时，则接车横移机工作状态为空闲，即$z_{1t}=0$，否则不空闲，$z_{1t}=1$。

$$x_{i(t+1)}=1, z_{1(t+1)}=1, if\ x_{it}=79, z_{1t}=0 \tag{17}$$

当车身i在t时刻的区域代码为 79 时，此时接车横移机工作状态为空闲时，则该车身i在下一时刻的区域代码为 1，此时接车横移机工作状态为不空闲。这样即满足接车横移机的优先处理原则。

约束 4（PBS 约束说明 7）：如果若干进车道停车位 1 有车身，当送车横移机空闲时，则优先运送最先进入进车道停车位 1 上的车身

定义 0-1 变量z_{2t}：

$$z_{2t}=\begin{cases}0 & ,\forall i\in I, x_{it}\neq 2\\ 1 & ,else\end{cases} \tag{18}$$

其中，在t时刻，所有车身的区域代码均不为 2 时，则送车横移机工作状态为空闲，即$z_{2t}=0$，否则不空闲，$z_{2t}=1$。

$$x_{i(t+1)}=2, z_{2(t+1)}=1\ if, x_{it}\in\{10,20,30,40,50,60\}, z_{2t}=0 \tag{19}$$

当车身i在t时刻的区域代码为若干进车道停车位 1 时，此时送车横移机工作状态为空闲时，即送车横移机刚完成一次运送任务，则该车身i在下一时刻的区域代码为 2，此时送车横移机工作状态为不空闲。这样即满足送车横移机的优先处理原则。

约束 5（PBS 约束说明 8）：若任意进车道停车位 1 存在车身，则送车横移机不能空闲

$$z_{2t}=1,\ if\ x_{it}\in\{10,20,30,40,50,60\} \tag{20}$$

当车身i在t时刻的区域代码为若干进车道停车位 1 时，则送车横移机工作状态设置为不空闲。

约束 6（PBS 约束说明 10、11）：当在同一车道内，连续两辆车身的可以不同步移动时间约束

$$\begin{gathered}\prod_{k=1}^{9}\left(x_{i(t-k)}-x_{it}+1\right)=0, if\ x_{i't}\neq x_{it}-1, x_{it}\in D\\ \prod_{k=1}^{9}\left(x_{i(t-k)}-x_{it}-1\right)=0, if\ x_{i't}\neq x_{it}+1, x_{it}\in B-D\end{gathered} \tag{21}$$

当第i个车身在t时刻区域代码属于集合B且下一个停车位没有车身时，则第i个车身将在九秒内某一时刻移动至下一停车位。

约束 7：车道内车身移动的时间约束

$$L\geq 9, if x_{i(t-1)}\neq x_{it}, \sum_{l=0}^{L}\left(x_{i(t+l)}-x_{it}\right)=0, x_{it}\neq x_{i(t+L+1)}, x_{it}\in B \tag{22}$$

当第i辆车身在t时刻刚到达B集合包含的区域代码时，在$(t+L+1)$时刻该车身到达

下一区域代码，所花费的时间L大于等于九秒。

约束 8（PBS 约束说明 4、5）：横移机上车身移动时间约束

$$E = time\left(x_{i(t-1)}, x_{it}, x_{i(t+E)}\right)$$

$$,if\ x_{i(t-1)} \neq x_{it}\ ,\sum_{e=0}^{E-1}\left(x_{i(t+e)} - x_{it}\right) = 0\ ,x_{it} \neq x_{i(t+E)}\ ,time\left(x_{i(t-1)}, x_{it}, x_{i(t+E)}\right) \in S, x_{it} \in \{1,2\} \quad (23)$$

其中，e表示横移机每次移动时刻，当车身i在$(t-1)$时刻处于返回道第十停车位或各进车道第一停车位或涂装-PBS 接车口，在t时刻该车身装载入横移机，必须经过E秒到达各进车道第十停车位或返回道第一停车位或 PBS-总装送车口。

约束 9：PBS 进车顺序与出车顺序约束（i与j对应关系）

$$\begin{gathered} t = 1,2,3 \ldots T \\ J = (U, V) \\ j \in J, u_j \in U, v_j \in V \\ t_{(J(1))}^{x_{J(1)t_{first}}=3} < t_{(J(2))}^{x_{J(2)t_{first}}=3} < \cdots < t_{(J(318))}^{x_{J(318)t_{first}}=3}\ ,\forall j \in J, J(j) = f\left(I(i)\right) \end{gathered} \quad (24)$$

其中，T表示调度 318 辆车身进入出车序列所使用的总时间，x_{it}中的$t = 1,2,3 \ldots T$。J表示出车序列的向量，U表示出车序列的动力属性向量，V表示出车序列的驱动属性向量，$J = (U, V)$表示出车序列中每一个车身的两个属性与车身序号一一对应。$t_{(J(j))}^{x_{J(j)t_{first}}=3}$表示排在出车序列$J$向量的第$j$个车身在该车身区域代码矩阵中当出现第一个 3 的时间，即第j个车身 PBS-总装接车口的时刻。$J(j) = f\left(I(i)\right)$表示出车序列与入车序列具有对应关系，出车序列向量$J$中存放的是按照时刻排序的进车序列编号，时刻小的排在出车序列J中靠前的位置。

综上，可得到问题一的**混合整数规划模型**，如下所示：

$$Z_1 = 100 - \sum_{j\in H}\delta(u_j,u_{j+1},u_{j+2},u_{j+3})$$

$$Z_2 = 100 - \sum_{m\in MM}\varepsilon(m) - 1\times mm$$

$$Z_3 = 100 - \sum_{i=1}^{318}\sum_{t=1}^{T} back_{it}$$

$$Z_4 = 100 - (0.01\times T - 9C - 72)$$

$$Z = \max(0.4Z_1 + 0.3Z_2 + 0.2Z_3 + 0.1Z_4)$$

$$s.t.\begin{cases}
\delta(u_j,u_{j+1},u_{j+2},u_{j+3}) = \begin{cases}0 & ,u_j = u_{j+3}且u_{j+1} = u_{j+2}\ \ (j\in H,u_j\in U)\\ 1 & ,else\ \ \ (j\in H)\end{cases} & \textbf{(1-1)}\\
\varepsilon(m) = \begin{cases}0 & ,else\\ 1 & ,length_{block(m)} \neq length_{block(m+1)}\end{cases}\ m\in MM & \\
block(m) = \{v_j,v_{j+1},\dots,v_{j+length}\}, block(m)\subset J, v_j\in V & \textbf{(1-2)}\\
back_{it} = \begin{cases}0, else\\ 1,\ \ x_{i(t-1)}\neq 70\ 且 x_{it} = 70\end{cases} & \textbf{(1-3)}\\
t = 1,2,3\dots T & \textbf{(1-4)}\\
t_{(J(1))}^{x_{J(1)t_{first}}=3} < t_{(J(2))}^{x_{J(2)t_{first}}=3} < \cdots < t_{(J(318))}^{x_{J(318)t_{first}}=3}, \forall j\in J, J(j) = f(I(i)) & \textbf{(1-5)}\\
J = (U,V) & \textbf{(1-6)}\\
x_{it}\neq x_{i't}, if\ \forall t, i\neq i', x_{it}, x_{i't}\in A' & \textbf{(1-7)}\\
x_{it} = a, x_{i(t+1)} = P_a, if a\in A, P_a\in Q_a & \textbf{(1-8)}\\
x_{i(t+1)} = 1, z_{1(t+1)} = 1, if\ x_{it} = 79, z_{1t} = 0 & \textbf{(1-9)}\\
x_{i(t+1)} = 2, z_{2(t+1)} = 1\ if, x_{it}\in\{10,20,30,40,50,60\}, z_{2t} = 0 & \textbf{(1-10)}\\
z_{2t} = 1,\ if\ x_{it}\in\{10,20,30,40,50,60\} & \textbf{(1-11)}\\
\prod_{k=1}^{9}(x_{i(t-k)} - x_{it} + 1) = 0, if\ x_{i't}\neq x_{it} - 1, x_{it}\in D & \\
\prod_{k=1}^{9}(x_{i(t-k)} - x_{it} - 1) = 0, if\ x_{i't}\neq x_{it} + 1, x_{it}\in B - D & \textbf{(1-12)}\\
E = time(x_{i(t-1)}, x_{it}, x_{i(t+E)}) & \textbf{(1-13)}\\
, if\ x_{i(t-1)}\neq x_{it}, \sum_{e=0}^{E-1}(x_{i(t+e)} - x_{it}) = 0, x_{it}\neq x_{i(t+E)}, time\left(x_{i(t-1)}, x_{it}, x_{i(t+E)}\right)\in S, x_{it}\in\{1,2\} &
\end{cases}$$

约束中说明：

（1-1）出车序列相邻四个车身动力属性是否满足混动燃油燃油混动，如果满足扣 0 分，否则扣 1 分。

（1-2）按照出车序列车身驱动属性分块并统计块内的数据个数，对比第一块和第二块，第二块和第三块的长度，依此类推，块长度相同则扣 0 分，否则扣 1 分。

（1-3）构造和状态矩阵相同的矩阵用于存放是否走返回道，当车身刚刚到返回道的第一停车位时则填 0，否则填 1。

（1-4）车身状态矩阵的列是从 1 到 T 的，列的长度为总调度时间。

（1-5）出车序列可由状态矩阵中进车序列车身刚刚出现区域代码 3，即进入总装环节的先后顺序导出，最先进入总装环节的排在出车序列第一个，第二个进入总装环节的排在出车序列的第二个，依此类推即可由状态矩阵推出出车序列。

（1-6） 进车序列的车身具有两种属性，即动力属性和驱动属性。
（1-7）在某车身的某一时刻状态矩阵中填入的区域代码（除涂装-PBS 和 PBS-总装）不能在同一时刻的其他车身状态矩阵中填入该区域代码。
（1-8）如果某一时刻某一车身状态矩阵中的区域代码已填入，按照运行顺序规则的集合对该车身下一时刻状态矩阵代码进行填写。
（1-9）如果某一时刻某一车身的区域代码为 79，即在返回道的第十个车位上，且接车横移机处于空闲状态时，那么该车身下一时刻装载入接车横移机且下一时刻接车横移机不空闲。这样就满足了接车横移机的优先原则。
（1-10）如果某一时刻某一车身的区域代码为{10,20,30,40,50,60}，即在 1-6 进车道的第一个车位上，且送车横移机处于空闲状态时，优先处理进车道第一停车位等待时间最长的车身，那么该车身下一时刻装载入送车横移机且下一时刻送车横移机不空闲。这样就满足了送车横移机的优先原则。
（1-11）如果某一时刻某一车身的区域代码为{10,20,30,40,50,60}，则送车横移机不能设置为空闲状态。
（1-12）当某一时刻某一车身的所处位置的下一个车位没有车时，则该车身可以在九秒内的某一时刻移动到下一个车位。这里就满足了车道内车身可以不同步的要求。
（1-13）如果某一时刻某一车身处于横移机上时，则其按照横移机的移动时间在状态矩阵中填入相应的时间长度。

5.1.2 问题一模型的求解

通过对问题的分析可知，该问题属于 NP 难问题[3]，同时对附件 1 和附件 2 数据的观察可以发现设计的算法需要同时适用于两种差异较大的进车序列，一般的算法无法得到较优的结果[4][5]，这里设计贪婪-模拟退火算法对该问题进行求解。对于问题一模型的算法设计可首先通过设计的贪婪算法求得一个可行解，之后再通过模拟退火的方法将初始解进行优化，从而实现得分最大化的目标。

（1）贪婪算法的设计：

本文所设计的贪婪算法主要分为两个步骤，第一个步骤是给 PBS 进车序列中的车身按照一定要求分配入进车道序号的第十个停车位，第二个步骤是当车身出进车道时，根据一定规则对该车身是否遣返进行判断，对于每一个车身当其自身属性不满足目标需求时需要进行遣返操作进入返回道。

模块 1：车身送入 PBS 进车道

车身在选择进车道时，如果优先级高的进车道通畅则选择优先级高的进入，否则选择次优的进车道，优先顺序按照接车横移机上装载车身的两种来源车身移动的两种方式分别列出。

方式一：当车身来源于涂装-PBS 出车口，为满足总调度时间尽可能少的目标，通过考虑横移机将车身送入进车道所耗时间的长短，对进车道的优先选择顺序进行排列，经过接车横移机进入进车道时对于车道选择的优先顺序按照消耗时间从短到长进行排序，排列结果为：4>3>2>5>1>6。

方式二：当车身来源于返回道，通过考虑横移机将车身送入进车道所耗时间的长短与避免出现拥堵的情况，对进车道的优先选择顺序进行排列，对于车身从返回道经过接车横移机进入进车道对于车道选择的优先顺序按照消耗时间从短到长进行排序，排列结果为：4>5>3>6>2>1。

本问题的进车道选择优先级方案如上。当接车横移机需要工作时，需要判断车身来自于何处，根据该车身的来源，按照对应优先级选择进入车道，当所选车道发生拥堵时，

则判断次级优先级车道是否发生拥堵，直到找到未发生拥堵车道，接车横移机将车身送入该进车道。

模块 2：车身驶出 PBS 进车道

本问题为满足目标函数一对于出车次序尽可能为“1 0 0 1”，即混动 燃油 燃油 混动的要求。为满足上述要求，需要满足 1（混动）：0（燃油）=1：2，分析附件 1 与附件 2 的数据可得，燃油车身的数量均小于该要求，考虑定义一个参数来记录 PBS 内混动：燃油的比例，因此定义$\mu=\frac{\text{剩余燃油车身数量}}{\text{剩余混动车身数量}}$来记录。根据$\mu$处于不同范围来判断混动少还是燃油少，以此来动态调整返回的车身类型。并对可能输出序列进行分析，分析三种模式下可接受的排列方式，三种模式对应的排列方式如下表所示。

表 1. 模式列表

模式一 $1.55 \le \mu \le 2.45$	模式二 $\mu \le 1.55$	模式三 $\mu \ge 2.45$
{100,110,010,001}	{101,111,011,001}	{100,110,010,000}

通过判断当前情况处于何种模式，并根据所给表格判断所输出的车身序号满足该模式下的可接受排列方式，判断该车身是否需要被遣返，从而达到满足目标函数一的目的。其具体操作思路如下所示。

首先查看已安排进出车序列倒数两个车身的动力属性与该车身的动力属性组成的序列是否属于模式一中的集合，如果是则无需遣返，直接将该车身安排进出车序列。否则，看已安排进出车序列倒数两个车身的动力属性与该车身的下一个将要出来的车身组成的序列是否属于模式一中的集合，如果是则将该车身遣返。否则，计算除已安排的出车序列中车身外的剩余车身动力属性比值μ，看此时的μ是在模式一、模式二还是模式三中，当选定模式后，观察已安排进出车序列倒数两个车身的动力属性与该车身的动力属性组成的序列是否属于该模式中的集合，如果是则将该车安排进出车序列。否则，进行下一步判断。

为保证满足目标函数二对于出车驱动车型的四驱与二驱之比倾向于 1：1 的要求，加入判断，若满足该要求则输出。

为保证满足目标函数三对于使用返回车道次数尽量少的要求，设定每辆车身走返回车道的次数上限为 3。

因此在送车横移机选择输出车身时，具体步骤如下：

Step1. 送车横移机开始工作。

Step2. 寻找首个到达进车道停车位 1 的车身，判断动力类型是否为“1”，若不为“1”，则送车横移机将该车身送入返回道，并重复该步操作；若为“1”，则送车横移机将该车身输出，并执行下一步。

Step3. 寻找第二个到达进车道停车位 1 的车身，判断动力类型是否为“0”，若不为“0”，则送车横移机将该车身送入返回道，并重复该步操作；若为“0”，则送车横移机将该车身输出，并执行下一步。

Step4. 判断是否完成出车动力次序为“10”的要求，若未完成，则返回 Step1，否则执行下一步。

Step5. 送车横移机开始进入模式判断模块。

Step6. 判断此时优先级最高的车身是否满足返回次数限制，若满足，则将该车身输出，并返回 Step5；否则执行下一步。

Step7. 判断此时优先级最高的车身是否满足模式一的车身动力类型排序要求，若满足，则将该车身输出，并返回 Step14；否则执行下一步。

Step8. 判断当前 PBS 内剩余车身比例是否在模式一，若在，输出该车身，并返回 Step14；否则执行下一步。

Step9. 判断当前 PBS 内剩余车身比例位于何种模式下，若位于模式一，则跳到 Step10；若位于模式二，则跳到 Step11；若位于模式三，则跳到 Step12。

Step10. 判断此时优先级最高的车身是否满足模式一的车身动力类型排序要求，若满足，则将该车身输出，并返回 Step14；否则执行 Step13。

Step11. 判断此时优先级最高的车身是否满足模式二的车身动力类型排序要求，若满足，则将该车身输出，并返回 Step14；否则执行 Step13。

Step12. 判断此时优先级最高的车身是否满足模式二的车身动力类型排序要求，若满足，则将该车身输出，并返回 Step14；否则执行 Step13。

Step13. 判断此时优先级最高的车身是否满足车身驱动类型排序要求，若满足，则将该车身输出，并返回 Step14；否则将车身送入返回道，并返回 Step5。

Step14. 判断是否完成所有车身输出，若未完成，则返回 Step5；否则排序完成，输出结果。

问题一贪婪算法流程图如图 3 所示：

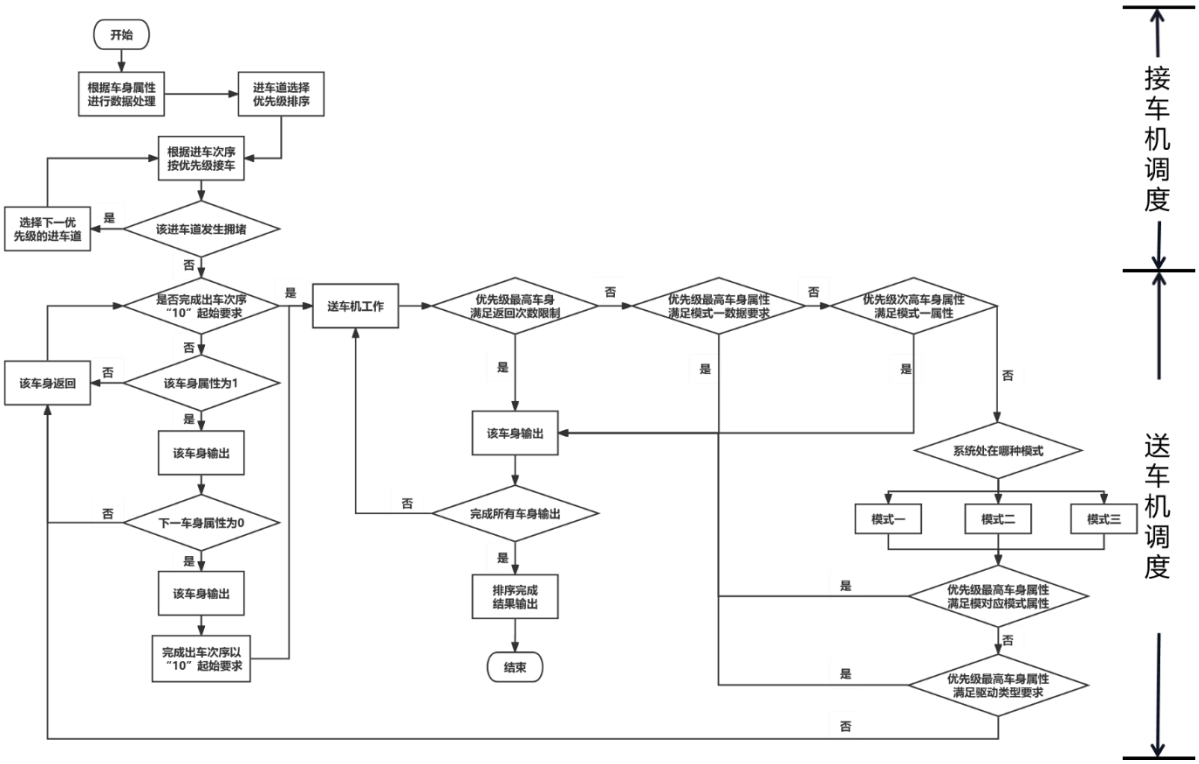

图 3. 问题一贪婪算法流程图

（2）模拟退火算法的设计：

由之前所设计的贪婪算法可得到 318 行 5 列的矩阵X_0，该矩阵中 5 列分别为进车序列序号、对应的两种属性、第一次进入车道的序号以及返回道的次数，接下来设计模拟退火的解更改原则，即更改进车序列对应的第一次进车道序号或返回道的次数，从而在修改后寻找最优解。

步骤 1：输入初始矩阵X_0，参数设定初始温度T_0，终止温度T_r，冷却速率v_c，迭代记数$T_{0_{count}}$。

步骤 2：当前矩阵$X_w = X_0$，当前温度$T_w = T_0$。

步骤 3：进入循环

1）当此时温度$T_w > T_r$，对X_w进行部分更改，首先在 318 个车身中随机选定一个车身序号，该车身对于进车道的选择进行随机更改，随机从 1-6 车道第十个停车位中除原本选择的车道外任选一个进车道与原矩阵中的第一次进车道序号替换，其次在 318 个车身中随机选定一个车身序号，更改该车道的返回次数上限，生成一个相邻解X_{w+1}。

2）计算该矩阵X_{w+1}对应的出车序列总得分f_{w+1}。

3）判断$\Delta f = f_{w+1} - f_w$。若$\Delta f > 0$，则接受为新解；若$\Delta f < 0$，则需要进行进一步的判断，对$random(0,1) - e^{\frac{\Delta f}{T_0}}$的大小进行判断，若大于 0，则舍弃，并回到步骤 3 的 1）；若小于 0，则也接受为新解。只要出现新解，$T_{0_{count}}++$，并判断$T_{0_{count}}$是否大于设定迭代次数，若未超过，则进行下一步，否则，退出循环。

4）计算此时温度$T_{w+1} = v_c * T_w$，并判断此时温度是否小于终止温度，若大于，进行下一次循环；否则，退出循环。图 4 为问题一模拟退火的算法流程图：

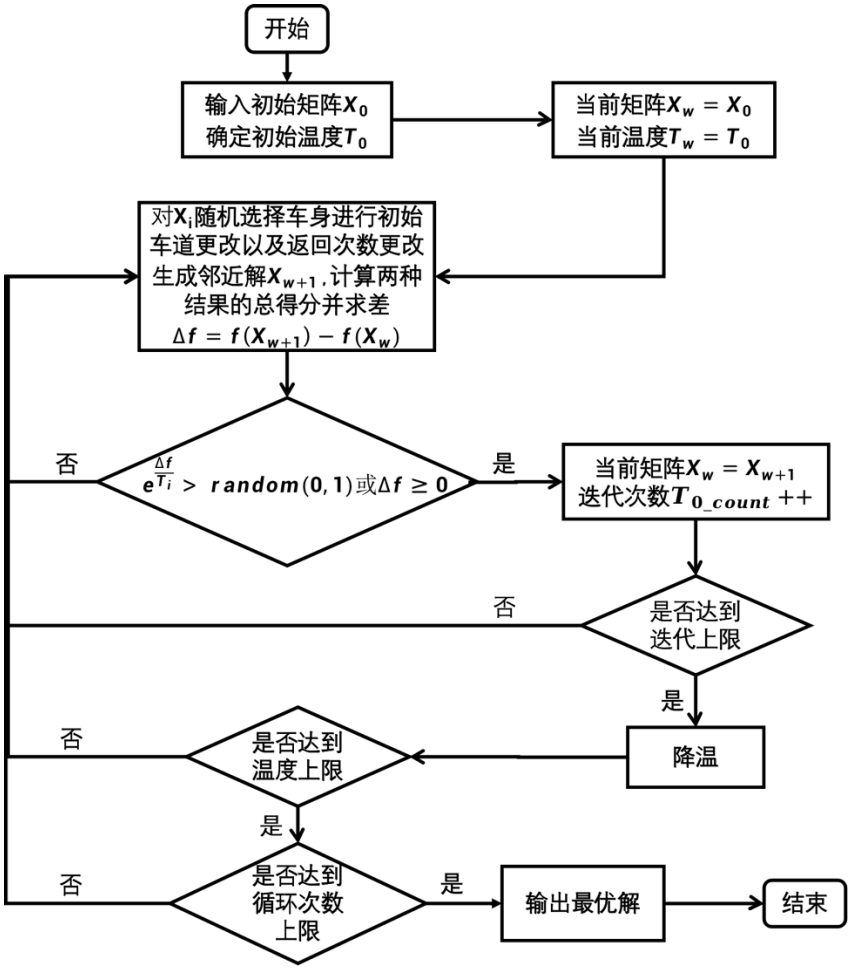

图 4. 问题一模拟退火算法流程图

5.1.3 问题一结果的分析

（1） 两种方式总得分与各目标函数得分情况比较

问题一中按照两种方法求解：①计算所有车身均只走四车道且没有返回道情况时计

算各目标得分及总得分②贪婪算法求得的解的基础上使用模拟退火算法计算各目标得分及总得分。表 2 中比较了两种方式下的得分情况

表 2. 问题一两种方式附件 1 和附件 2 的得分情况

方式	附件编号	总得分 (100 分)	目标函数 (100 分)	目标函数 2 (100 分)	目标函数 3 (100 分)	目标函数 4 (100 分)
原始顺序	附件 1	13.098	-103	81	100	99.98
	附件 2	35.098	-48	81	100	99.98
贪婪+退火	附件 1	15.394	-87	81	78	102.94
	附件 2	38.794	-28	81	77	102.94

由上表 2 可得贪婪加模拟退火得到的两个附件的总得分均比原始顺序的得分要高，这里的原始顺序表示出车顺序与进车顺序保持一致，结果显示通过返车道调整出车序列具有较好的效果，这里贪婪-模拟退火得到的结果比原始顺序的总得分提高的不多可能是因为附件 1 和附件 2 的进车序列的属性分布差异较大。设计的算法对两种不同分布情况均计算出较好的结果，表明算法有普适性。附件 1、2 中设计的算法较原始序列总得分的优化比例分别为 17.5%，10.53%。

（2） 模拟退火寻得最优解

表 3. 问题一两种算法附件 1 和附件 2 的得分情况

算法	附件编号	总得分 (100 分)	目标函数 1 (100 分)	目标函数 2 (100 分)	目标函数 3 (100 分)	目标函数 4 (100 分)
贪婪算法	附件 1	12.967	-93	80	78	105.67
	附件 2	35.915	-37	80	80	107.15
贪婪+退火	附件 1	15.394	-87	81	78	102.94
	附件 2	38.794	-28	81	77	102.94

贪婪算法是走六个车道且有一个返车道时求得的一个可行解，模拟退火在这个解的邻近解中寻找最优解，在表 3 中可以观察到贪婪算法之后用模拟退火的结果比只用贪婪算法总得分高。这里主要是提高了目标函数 1 的分数，适当降低目标函数 3 和目标函数 4 的分数，即尽量满足动力属性要求，从而达到总得分最高的结果。附件 2 的总得分一直比附件 1 的总得分大的原因可能是附件 2 中的数据更加均匀，更容易满足目标函数。

（3） 两种算法使用返回道情况

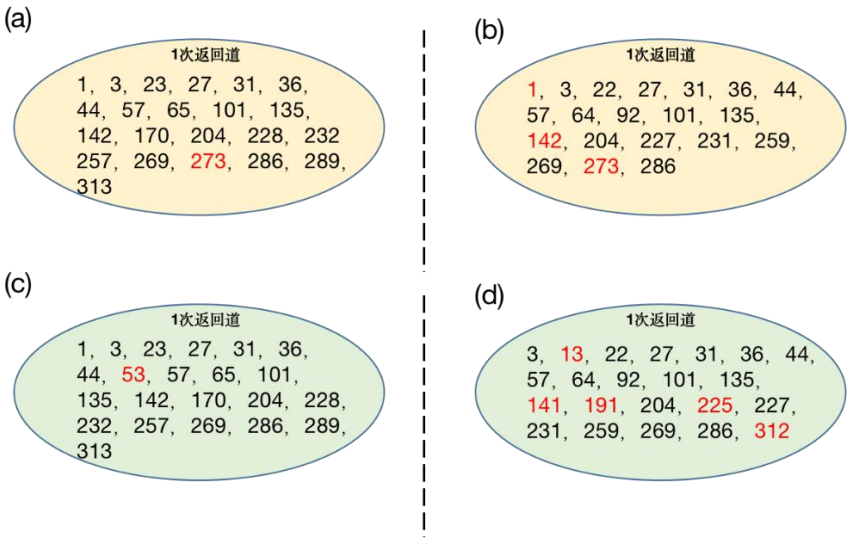

图 5. 问题一使用返回道车身序号集合

由贪婪算法的结果可以得出附件 1 共使用返回道 22 次，附件 2 共使用返回道 20 次，且这些使用返回道的车身均只使用一次返回道；由贪婪-模拟退火算法的结果可以得到两个附件均使用 20 次返回道。上图展示了两种算法中两个附件只走返回道一次的车身序号集合，其中图 5（a-b）表示用贪婪算法得到的附件 1、2 的走返回道车身序号集合。图 5（c-d）表示用贪婪-模拟退火算法得到的附件 1、2 的走返回道车身序号集合。图中标注红色的是对比上下集合中不相同的序号，黑色表示上下集合中相同的序号。

（4） 两种算法总调度时间情况

表 4. 问题一两种算法附件 1 和附件 2 总调度时间

算法	附件编号	总调度时间（秒）
贪婪算法	附件 1	2367
	附件 2	2219
贪婪+退火	附件 1	2640
	附件 2	2640

表 4 显示出贪婪算法中附件 1 的总调度时间是 2367 秒，附件 2 的总调度时间为 2219 秒；贪婪加模拟退火算法的附件 1 和附件 2 的总调度时间均是 2640 秒。已知贪婪-模拟退火算法计算得到的总得分比贪婪算法的总得分高，则可以看出贪婪-模拟退火算法为满足总得分尽量高牺牲了一部分的调度时间。

（5） 两种算法附件 1 出车序列部分结果

表 5.问题一附件 1 两种算法出车动力属性对比

原出车序列	贪婪算法调度后出车序列	贪婪+退火调度后出车序列	原序列出车动力属性	贪婪算法调度后出车动力属性	贪婪+退火调度后出车动力属性
224	**225**	**223**	**1**	**1**	**1**
225	**226**	**227**	**1**	**0**	**0**
226	**227**	**226**	**0**	**0**	**0**
227	**229**	**230**	**0**	**1**	**1**
228	**230**	**231**	**0**	**1**	**0**
229	**231**	**229**	**1**	**0**	**1**
230	**233**	**233**	**1**	**0**	**0**
231	**204**	**236**	**0**	**1**	**0**
232	**234**	**204**	**1**	**0**	**1**
233	**235**	**235**	**0**	**0**	**0**
234	**236**	**234**	**0**	**0**	**0**
235	**237**	**239**	**0**	**0**	**1**
236	**238**	**238**	**0**	**0**	**0**
237	**239**	**237**	**0**	**1**	**0**
238	**240**	**242**	**0**	**0**	**1**
239	**241**	**241**	**1**	**1**	**1**
240	**242**	**240**	**0**	**1**	**0**

表 5 中展示了原出车序列部分车身对应的动力属性和两种算法调度后的动力属性比对，在该部分中这一段原序列没有满足 “1001”，贪婪算法调度后满足两次“1001”，贪婪-模拟退火算法满足三次“1001”，从而能够使得调度出车序列比原出车序列扣分少。

（6） 贪婪-模拟退火算法附件 1 车身在 PBS 中运动的结果可视化

贪婪-模拟退火算法附件 1 车身在 PBS 中运动的结果可视化：

以附件 1 中涂装-PBS 出车口进车序列中的第一辆车身为例，其从开始到最终到达 PBS-总装接车口的结果过程如图所示。其中红六角星表示序号一车身，其它数字的位置表示当前时刻 PBS 内所包括所有车身的序号处于何处[6]。

下面为由模拟退火算法得到的最终结果可视化：

① 当$t = 0s$时，第 1 车身选进车道 4，则此时第 1 车身位于进车道 4 的停车位 10。

② 当$t = 82s$时，此时 PBS 内所包含车身所处位置如图 6 所示。

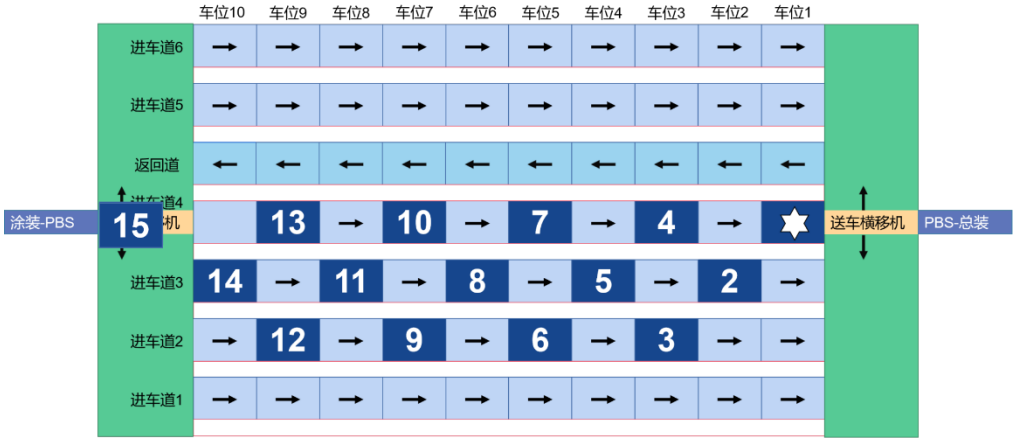

图 6. 问题一当$t = 82s$时 PBS 内车身位置示意图

此时第 1 车身刚刚进入进车道 4 的停车位 1 的位置，2 表示第 2 车身所处位置为进车道 3 的停车位 2，之后以此类推。进车道 2 与进车道 3 内的车身均已在所在停车位停留 6s，停车道 4 内车身均刚刚到达所在停车位，第 15 车身已在接车横移机上 3s。

③ 当$t = 86s$时，此时 PBS 内所包含车身所处位置如图 7 所示。

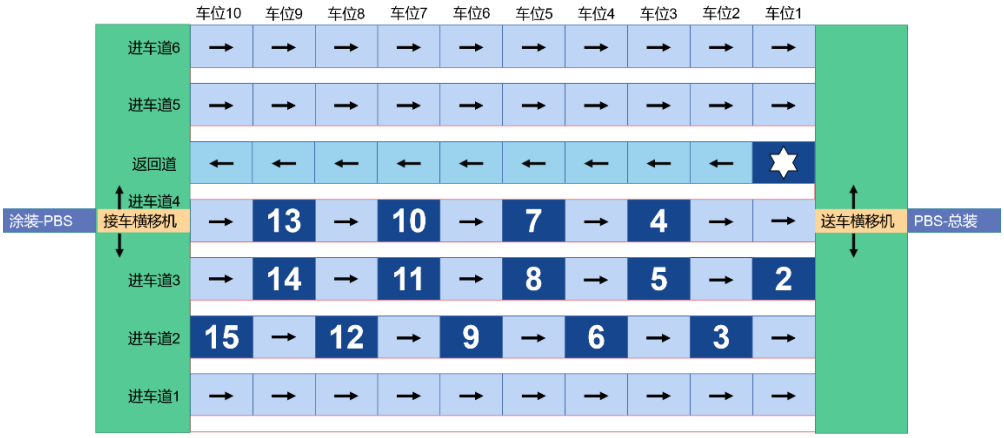

图 7. 问题一当$t = 86s$时 PBS 内车身位置示意图

此时第 1 车身刚刚进入返回道的停车位 1 的位置，进车道 2 与进车道 3 内的车身均已在所在停车位停留 1s，进车道 4 内车身已在所在停车位停留 4s。

④ 当$t = 178s$时，此时 PBS 内所包含车身所处位置如图 8 所示。

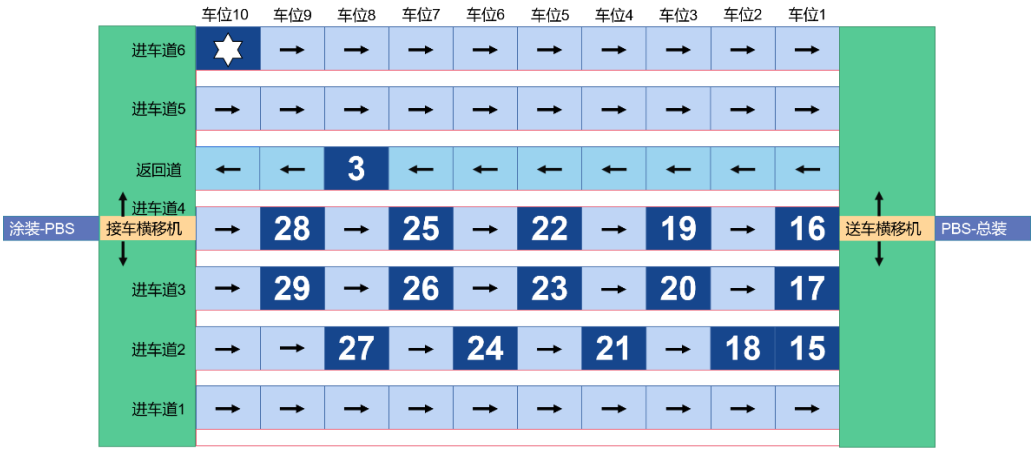

图 8.问题一当$t = 178s$时 PBS 内车身位置示意图

此时第 1 车身根据轮盘赌法加上输出车身选择优先级来对进车道进行选择，选择进车道 6 作为第 1 车身再次往总装车间送的进车道。第 1 车身刚进入进车道 6，进车道 2 除第 15 车身与进车道 3 内的车身已在所在停车位停留 3s，第 15 车身已经在进车道 2 的停车位 1 处等待送车机完成其它任务后来运送该车身，进车道 4 内的车身均已到达所在停车位 6s。此时车身 15、16、17 均在停车位 1，此时根据送车横移机优先选择先到车身，来装载第 15 车身。第 3 车身在返回道停车位 8 停留了 5s。

⑤ 当$t = 293$时，此时 PBS 内所包含车身所处位置如图 9 所示。

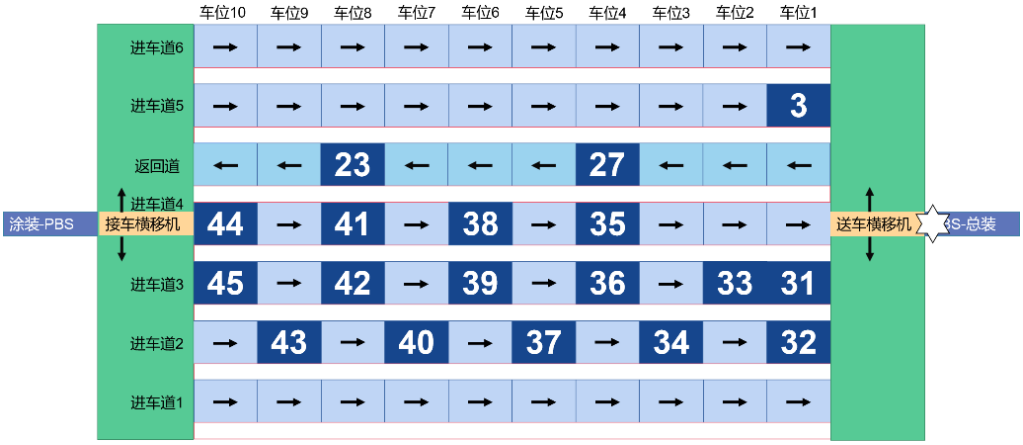

图 9.问题一当$t = 293s$时 PBS 内车身位置示意图

此时第 1 车身刚刚进入 PBS-总装接车口，即完成了整个 PBS 过程。第 3 车身经由返回道后，根据轮盘赌法，再次进入进车道 5 停车位 1，已停留 14s，在等待送车横移机空闲。第 23 车身与第 27 车身在返回道中进行返回，其中第 23 车身刚刚到达返回道停车位 8，第 27 车身刚刚到达返回道停车位 4。第 31 车身比第 32 车身早到达停车位 1，正在等待送车横移机空闲，进车道 2 与进车道 3 内的车身均已到达所在停车位 2s，进车道 4 内的车身均已到达所在停车位 5s。

5. 2 问题二模型的建立与求解

5. 2. 1 问题二模型的建立

问题二中目标函数没有改变，约束中问题二是在问题一的基础上去除 PBS 约束说明 6 和 7，即去除问题一模型中约束 3 和约束 4，即两种横移机均无需在横移机空闲时对车身进行优先处理原则，横移机可以在停车位有车身时进行自主选择。

由问题一模型去除两条约束可建立问题二**混合整数规划模型：**

$$Z_1 = 100 - \sum_{j\in H}\delta(u_j,u_{j+1},u_{j+2},u_{j+3})$$

$$Z_2 = 100 - \sum_{m\in MM}\varepsilon(m) - 1\times mm$$

$$Z_3 = 100 - \sum_{i=1}^{318}\sum_{t=1}^{T} back_{it}$$

$$Z_4 = 100 - (0.01\times T - 9C - 72)$$

$$Z = \max(0.4Z_1 + 0.3Z_2 + 0.2Z_3 + 0.1Z_4)$$

$$s.t.\begin{cases}
\delta(u_j,u_{j+1},u_{j+2},u_{j+3}) = \begin{cases}0 & ,u_j = u_{j+3}\text{且}u_{j+1} = u_{j+2}\ \ (j\in H,u_j\in U)\\ 1 & ,else\ \ \ (j\in H)\end{cases} & \textbf{(2-1)}\\
\varepsilon(m) = \begin{cases}0 & ,else\\ 1 & ,length_{block(m)} \neq length_{block(m+1)}\end{cases}\ m\in MM & \\
block(m) = \{v_j,v_{j+1},\dots,v_{j+length}\}, block(m)\subset J, v_j\in V & \textbf{(2-2)}\\
back_{it} = \begin{cases}0, else\\ 1,\ \ x_{i(t-1)}\neq 70\ \text{且}x_{it}=70\end{cases} & \textbf{(2-3)}\\
t = 1,2,3\dots T & \textbf{(2-4)}\\
t_{(J(1))}^{x_{J(1)t_{first}}=3} < t_{(J(2))}^{x_{J(2)t_{first}}=3} < \cdots < t_{(J(318))}^{x_{J(318)t_{first}}=3}, \forall j\in J, J(j) = f(I(i)) & \textbf{(2-5)}\\
J = (U,V) & \textbf{(2-6)}\\
x_{it}\neq x_{i't}, if\ \forall t, i\neq i', x_{it}, x_{i't}\in A' & \textbf{(2-7)}\\
x_{it} = a, x_{i(t+1)} = P_a, if a\in A, P_a\in Q_a & \textbf{(2-8)}\\
z_{2t} = 1,\ if\ x_{it}\in\{10,20,30,40,50,60\} & \textbf{(2-9)}\\
\prod_{k=1}^{9}(x_{i(t-k)} - x_{it} + 1) = 0, if\ x_{i't}\neq x_{it} - 1, x_{it}\in D & \\
\prod_{k=1}^{9}(x_{i(t-k)} - x_{it} - 1) = 0, if\ x_{i't}\neq x_{it} + 1, x_{it}\in B - D & \textbf{(2-10)}\\
E = time(x_{i(t-1)}, x_{it}, x_{i(t+E)}) & \textbf{(2-11)}\\
, if\ x_{i(t-1)}\neq x_{it}, \sum_{e=0}^{E-1}(x_{i(t+e)} - x_{it}) = 0, x_{it}\neq x_{i(t+E)}, time\left(x_{i(t-1)}, x_{it}, x_{i(t+E)}\right)\in S, x_{it}\in\{1,2\} &
\end{cases}$$

5. 2. 2 问题二模型的求解

（1）贪婪算法的设计：

贪婪算法主要分为两个步骤。

首先对车身属性进行分类和数据分析，根据附件 1 与附件 2 所给的数据，所有车身可以分为 3 种类型：“04”为燃油四驱、“02”为燃油两驱、“12”为混动两驱。且附件 1 中类型“12”为 212 辆，类型“02”为 77 辆，类型“04”为 29 辆；附件 2 中类型“12”为 159 辆，类型“02”为 130 辆，类型“04”为 29 辆。

之后开始运送车身，车身运送主要分为两个阶段:

步骤 1：车身接入进车道

根据类型车身的数量，对不同类型的车身分配其对应的进车道，根据分配可得：类型“12”分配进车道 4、2，类型“02”分配进车道 3、1，类型“04”分配进车道 5。

本问题考虑“专车道专用“的思想，分配方案如上述分配。当接车横移机在涂装-PBS 出车口接到车身后，判断该车身属于哪种类型，根据该车身的类型，选择进入对应进车道，当对应车道都发生拥堵时，则接车横移机进行等待，直到对应车道停车位 10 出现

空位为止。

步骤 2：车身送出 PBS-总装接车口

此时送车横移机可以选择车身，为满足优化目标 1 与优化目标 2 的输出车身次序要求，先对可能输出序列进行分析。输出序列形式为下面三种情况：

情况 1：

驱动类型：	2	2	2	2	2	2	2	2	2	2	2	…
动力类型：	1	0	0	1	0	0	1	0	0	1	0	0 …

情况 2：

驱动类型：	2	4	2	2	4	4	2	4	2	2	4	4 …
动力类型：	1	0	0	1	0	0	1	0	0	1	0	0 …

情况 3：

驱动类型：	2	2	2	2	2	2	2	2	2	2	2	2 …
动力类型：	1	1	1	1	1	1	1	1	1	1	1	1 …

其中情况 1 与情况 3 均只需驱动类型 2，情况 2 则需要驱动类型 2 与 4，且驱动类型 4 的车身仅 29 辆，即情况 2 发生的次数比较少，则优先考虑情况 2。不难发现情况 2 以 6 辆车身为一次循环，一次循环中包括类型“04”为 3 辆，类型“12”为 2 辆，类型“02”为 1 辆。因此为保证在尽可能多的满足优化目标 1 的情况下去满足优化目标 2 时，在输出车身序列时，类型“04”应尽量以情况 2 的方式输出。

因此在送车横移机选择输出车身时，步骤如下：

Step1. 送车横移机在初次运送前，首先分别计算出情况 1（12 02 02）、情况 2（12 04 02 12 04 04）和情况 3（12 12）分别需要的运送量。

Step2. 送车横移机第一次运送车身时，必须保证该车身的类型为“12”。当发现存在进车道停车位 1 上有类型“12”的车身时，送车横移机装载该车身。

Step3. 送车横移机开始按照不同情况挑选车身进行运送。

Step4. 判断已完成情况 2 的数量，若未达到规定数量，则执行下一步；若达到规定数量，则跳至 Step6。

Step5. 优先判断类型“04”对应的进车道停车位 1-6 是否存在 3 辆车身。

若未存在，则执行下一步；

若存在，则进一步判断类型“02”对应的进车道停车位 1-4 是否存在车身，若未存在，则执行下一步；若存在，则按照情况 2 依次挑选对应类型车身进行运送，当需要运送某辆车身时，该车身还未处于停车位 1，则送车横移机将此时处于停车位 1 的车身运送到返回道上，直到该车身处于停车位 1，完成该情况，并返回 Step3。

Step6. 判断已完成情况 1 的数量，若未达到规定数量，则执行下一步；若达到规定数量，则跳至 Step8。

Step7. 判断类型“02”对应的进车道停车位 1-4 是否一共存在 2 辆车身，若未存在，则执行下一步；若存在，则按照情况 1 依次挑选对应类型车身进行运送，当需要运送某辆车身时，该车身还未处于停车位 1，则送车横移机将此时处于停车位 1 的车身运送到返回道上，直到该车身处于停车位 1，完成该情况，并返回 Step3。

Step8. 判断已完成情况 3 的数量，若未达到规定数量，则执行下一步；若达到规定数量，则跳至 Step10。

Step9. 选择类型“12”的车身进行运送。运送完成后执行下一步。

Step10. 判断是否完成所有车身的运送，若未完成，则返回 Step3，若完成，排序完成，结果输出。

问题二贪婪算法流程图如图 10 所示：

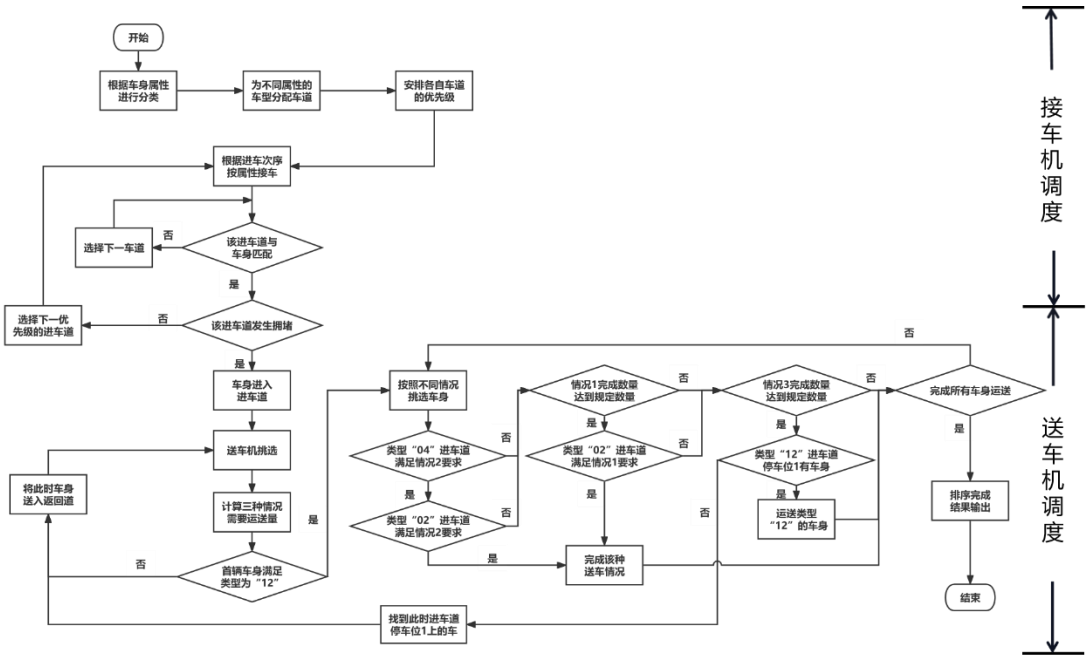

图 10. 问题二贪婪算法流程图

5. 2. 3 问题二结果的分析

（1）两种方式总得分与各目标函数得分情况比较

表 6. 问题二两种方式附件 1 和附件 2 的得分情况

方式	附件编号	总得分 (100 分)	目标函数 (100 分)	目标函数 2 (100 分)	目标函数 3 (100 分)	目标函数 4 (100 分)
原始顺序	附件 1	13.098	-103	81	100	99.98
	附件 2	35.098	-48	81	100	99.98
贪婪	附件 1	26.808	-69	87	91	101.08
	附件 2	48.771	-13	86	91	99.71

问题二比较原始顺序和贪婪算法最终的总得分，表 6 展示了两种方式下的得分情况。由表 6 可得贪婪算法计算的两个附件的总得分均比原始顺序的高，尤其是目标函数 1 的得分有较大的提升，这可能是由于问题二中去除了横移机优先处理原则，使得横移机可以自主选择车身，放宽了约束使得目标函数更容易满足。附件 1、2 中设计的算法较原始序列总得分的优化比例分别为 104.7%，39.0%。

（2）贪婪算法使用返回道的情况

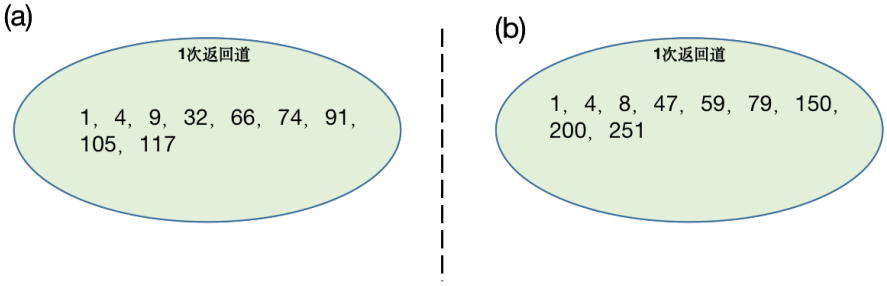

图 11. 问题二使用返回道车身序号集合

问题二中附件 1 和附件 2 均使用 9 次返回车道，且每辆返回的车身只返回了一次。图 11（a）表示附件 1 中走返回道的车身序列集合，图 12（b）表示附件 2 中走返回道序列集合。

（3）贪婪算法总调度时间情况

表 7. 问题二贪婪算法附件 1 和附件 2 总调度时间

算法	附件编号	总调度时间（秒）
贪婪算法	附件 1	2826
	附件 2	2963

表 7 显示出贪婪算法中附件 1 的总调度时间是 2826 秒，附件 2 的总调度时间为 2963 秒。已知贪婪算法计算得到的总得分比原始序列的总得分高，则可以看出贪婪算法为实现尽量高的总得分牺牲了一部分的调度时间。

（4）贪婪算法附件 1 出车序列部分结果

表 8.问题二附件 1 贪婪算法出车动力属性对比

原出车序列	贪婪算法调度后出车序列	原序列出车动力属性	贪婪算法调度后出车动力属性
1	2	0	1
2	3	1	1
3	6	1	0
4	7	0	0
5	5	1	1
6	8	0	0
7	11	0	0
8	10	0	1
9	12	1	0
10	13	1	1
11	16	0	0
12	15	0	1
13	14	1	1
14	18	1	1
15	17	1	1
16	20	0	1
17	21	1	0
18	23	1	0
19	24	1	1

表 8 中比较了问题二中原出车序列以及贪婪算法调度后部分车身对应的动力属性满足情况，在截取的片段中原序列满足了一次 “1001”，贪婪算法调度后满足了三次“1001”，从而能够实现经过设计的算法调度的出车序列比原出车序列目标函数 1 扣分少。

（5）贪婪算法附件 1 车身在 PBS 中运动的结果可视化

仍以附件 1 中涂装-PBS 出车口进车序列中的第一辆车身为例，其从开始到最终到达 PBS-总装接车口的结果过程如图所示。

下面为由贪婪算法得到的最终结果可视化：

① 当$t=0s$时，第 1 车身选择进车道 3，则此时第 1 车身位于接车横移机上。

② 当$t=85s$时，此时 PBS 内所包含车身所处位置如图 12 所示。

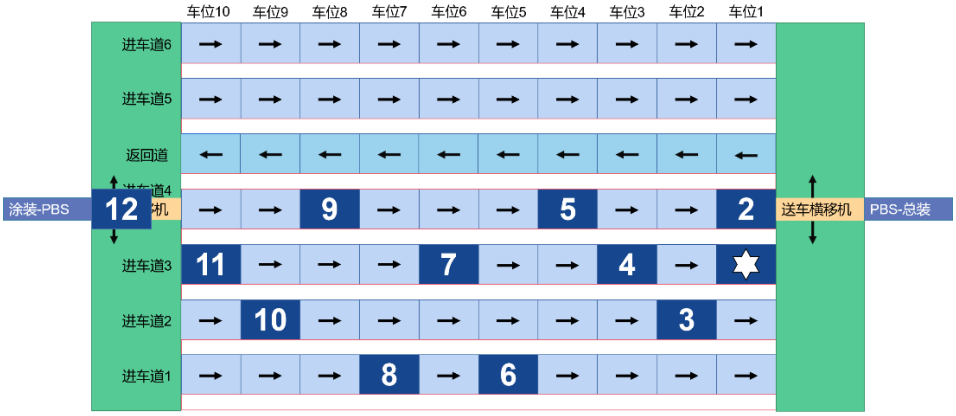

图 12. 问题二当$t=85s$时 PBS 内车身位置示意图

此时第 1 车身刚刚进入进车道 3 的停车位 1 的位置，2 表示第 2 车身所处位置为进车道 3 的停车位 2，之后以此类推。进车道 1 内存在的车身为 6、8，进车道 2 内存在的车身为 3、10，进车道 3 内存在的车身为 1、4、7、11，进车道 4 内存在的车身为 2、5、9，车身 12 位于接车横移机上。

③ 当$t=104s$时，此时 PBS 内所包含车身所处位置如图 13 所示。

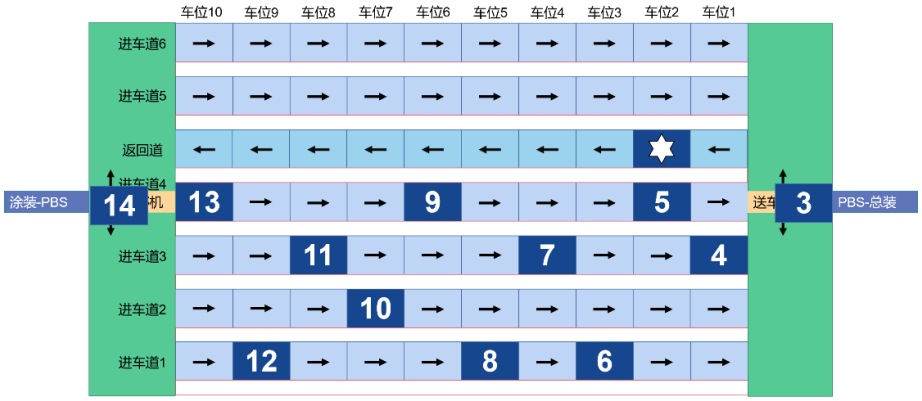

图 13. 问题二当$t=104s$时 PBS 内车身位置示意图

此时第 1 车身刚刚进入返回道的停车位 2 的位置，进车道 1 内存在的车身为 6、8、12，进车道 2 内存在的车身为 10，进车道 3 内存在的车身为 4、7、11，进车道 4 内存在的车身为 5、9、13，车身 3 位于送车横移机上，车身 14 位于接车横移机上。

④ 当$t=197$时，此时 PBS 内所包含车身所处位置如图 14 所示。

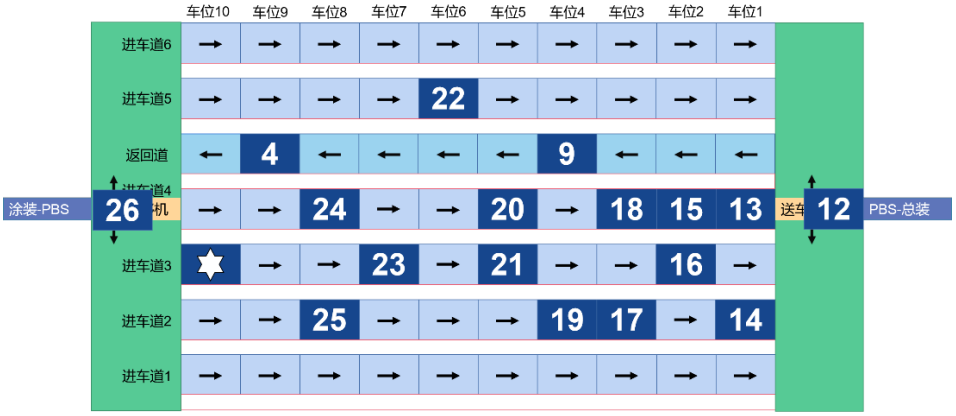

图 14. 问题二当$t=197s$时 PBS 内车身位置示意图

此时第 1 车身位于进车道 3 停车位 10 上，进车道 2 内存在的车身为 14、17、19、25，进车道 3 内存在的车身为 1、16、21、23，进车道 4 内存在的车身为 13、15、18、20、24，进车道 5 内存在的车身为 22，车身 4 与车身 9 位于返回道上，车身 12 位于送车横移机上，车身 26 位于接车横移机上。

⑤ 当$t = 304$时，此时 PBS 内所包含车身所处位置如图 15 所示。

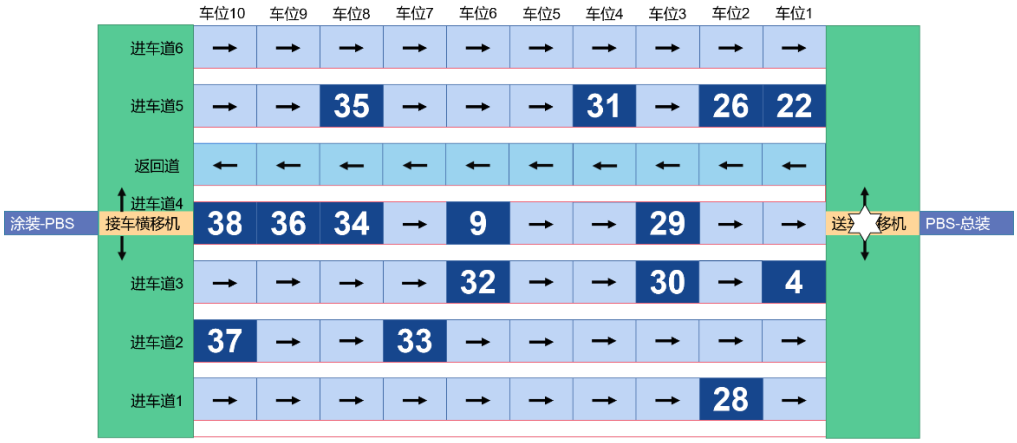

图 15. 问题二当$t = 304s$时 PBS 内车身位置示意图

此时第 1 车身位于送车横移机上，进车道 1 内存在的车身为 28，进车道 2 内存在的车身为 33、37，进车道 3 内存在的车身为 4、30、32，进车道 4 内存在的车身为 29、9、34、36、38，进车道 5 内存在的车身为 22、26、31、35。

六、 模型的评价与改进

本问题从车身状态转移矩阵出发，从该矩阵中推出出车序列，书写与出车序列相关的目标函数以及与车身状态转移矩阵相关的约束条件，很好将较为复杂的模型中的目标函数与约束条件使用状态转移矩阵联系起来，模型具有较好的整体性。

6.1 模型的优点

（1） 模型结构清晰，目标函数与约束条件均使用状态矩阵连接，有较好的连贯性；
（2） 模型中使用集合和向量两种方式来表示一一对应的关系，从而使约束更清晰；
（3） 模型约束条件清晰，问题一将 PBS 约束说明的 12 条合并为 9 条，约束的减少使得求解高效且易于实现；
（4） 模型中设定一些参数可根据数据的长度进行改变，具有良好的拓展性；
（5） 设计的算法在两个附件上均得到较优的结果表明算法具有一定的普适性。

6.2 模型的缺点

（1） 模型求解的结果对输入数据的车身属性比例依赖性较高，当车身属性比例有较大变动时需要对模型中的参数进行适当调整；
（2） 模型只考虑了车身有两种属性，而实际生产中可能存在更多属性，此时需要对模型的目标函数和约束条件进行调整。

七、 参考文献

[1]李晋航. 混流制造车间物料配送调度优化研究[D].华中科技大学,2012.
[2]陈正茂. 基于排序缓冲区的多车间关联排序研究[D]. 华中科技大学, 2008.
[3] A. Lim and Zhou Xu, "Searching optimal resequencing and feature assignment on an automated assembly line," 17th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'05), 2005, pp. 8 pp.-498, doi: 10.1109/ICTAI.2005.114.
[4]冷泯伶. 基于深度强化学习的汽车涂装生产排序研究[D].大连理工大学,2021.
[5]付玉婷. 基于强化学习的带返工汽车涂装重排序方法[D].大连理工大学,2021.
[6] Zhipeng T, Xinyu S, Haiping Z, et al. Small-world optimization algorithm and its application in a sequencing problem of painted body storage in a car company[J]. Mathematical Problems in Engineering, 2015, 2015.
[7]C. E. Speck and M. G. Reynolds, "Electrostatic Processes Affecting Attraction of Dirt to Metallic Vehicle Bodies During Conventional (Nonelectrostatic) Painting," in IEEE Transactions on Industry Applications, vol. IA-23, no. 5, pp. 810-819, Sept. 1987, doi: 10.1109/TIA.1987.4504988.

附录

附录材料的列表

附录 1：文中参数对应表格

（1）Q_a为当区域代码为a时其对应的可走序列的集合。

$x_{it}=a$	Q_a
0	0, 1, 49
1	$back_{i(t-1)}=0$, {1, 19, 29, 39, 59, 69} $back_{i(t-1)}=1$, {1, 19, 29, 39, 49, 59, 69}
2	2, 3, 70
3	3
10	10, 2
11	11, 10
12	12, 11
13	13, 12
14	14, 13
15	15, 14
16	16, 15
17	17, 16
18	18, 17
19	19, 18
20	20, 2
21	21, 20
22	22, 21
23	23, 22
24	24, 23
25	25, 24
26	26, 25
27	27, 26
28	28, 27
29	29, 28
30	30, 2
31	31, 30
32	32, 31
33	33, 32
34	34, 33
35	35, 34
36	36, 35
37	37, 36
38	38, 37
39	39, 38

40	40, 2
41	41, 40
42	42, 41
43	43, 42
44	44, 43
45	45, 44
46	46, 45
47	47, 46
48	48, 47
49	49, 48
50	50, 2
51	51, 50
52	52, 51
53	53, 52
54	54, 53
55	55, 54
56	56, 55
57	57, 56
58	58, 57
59	59, 58
60	60, 2
61	61, 60
62	62, 61
63	63, 62
64	64, 63
65	65, 64
66	66, 65
67	67, 66
68	68, 67
69	69, 68
70	70, 71
71	71, 72
72	72, 73
73	73, 74
74	74, 75
75	75, 76
76	76, 77
77	77, 78
78	78, 79
79	79, 1

（2）S为横移机的移动时间集合

$(x_{i(t-1)}, x_{it}, x_{i(t+E)})$:区域代码转移	$S(x_{i(t-1)}, x_{it}, x_{i(t+E)})$:花费时间（秒）
(0,1,19)	18
(0,1,29)	12
(0,1,39)	6
(0,1,49)	0
(0,1,59)	12
(0,1,69)	18
(10,2,3)	18
(20,2,3)	12
(30,2,3)	6
(40,2,3)	0
(50,2,3)	12
(60,2,3)	18
(79,1,19)	24
(79,1,29)	18
(79,1,39)	12
(79,1,49)	6
(79,1,59)	12
(79,1,69)	18
(10,2,70)	24
(20,2,70)	18
(30,2,70)	12
(40,2,70)	6
(50,2,70)	12
(60,2,70)	18

附录 2 代码

介绍：问题一代码

```
%% 六车道贪婪算法
function [Fenshu,Line_ru,Line_chu,XX,Csm1,Csm2,T]=Tanlan1(nx)
% XX 为结果矩阵，Line_ru 为入车次序，Line_chu 出车次序，Csm1 横移车 1 运动轨迹，
% Csm2 横移车 2 运动轨迹，Fenshu 得分，T 总调度时间
%nx 附件几
Line_ru=[xlsread(['附件',num2str(nx),'.xlsx'],'A2:C319'),zeros(318,2)];%入车顺序
XX=-1*ones(318,9*318*3);%结果矩阵
Time_ting=zeros(6,2);%进车道一号位停留时间
Csm1=ones(3,9*318*3).*[0,-2,-2]';%横移机 1
Csm2=Csm1;%横移机 2
Line_chu=zeros(318,3);%出车顺序
```

```
t=1;num_chu=0;
need(:,:,1)=[0,0;0,0];need(:,:,2)=[1,0;0,0];need(:,:,3)=[1,1;1,1];
while(1-isempty(find(XX(:,8586)~=2,1)))
    XX=gengxinxx(XX,t);
    choose1=0;
    while(Csm1(1,t)==0&&choose1~=100)%横移机 1 可进行工作时
        Body_che1=find(XX(:,t)==-1|XX(:,t)==79);
        if isempty(Body_che1)
            break
        end
        Body_che1=Body_che1(1);
        site_che1=XX(Body_che1,t);
        choose1=next_dao1(site_che1,XX(:,t));
        if choose1~=100
            if Line_ru(Body_che1,5)==0
                Line_ru(Body_che1,4)=fix(choose1/10);
            end
            if site_che1==-1&&choose1==49
                XX(Body_che1,t:t+8)=49;
            else
                [XX,Csm1]=gengxinxc1(XX,Csm1,choose1,site_che1,Body_che1,t);
            end
        end
    end

    while(Csm2(1,t)==0&&sum(Time_ting(:,2))>0)%横移机 2 可进行工作时
        Body_che2=find(Time_ting(:,2)==max(Time_ting(:,2)));
        Body_che2=Time_ting(Body_che2(1),1);
        site_che2=XX(Body_che2,t);

choose2=next_dao2(Time_ting,XX,t,Body_che2,site_che2,Line_ru,Line_chu,num_chu,nee
d);
        if site_che2==40&&choose2==2
            XX(Body_che2,t:9*318*3)=2;
            Time_ting(4,:)=0;
        else
            [XX,Csm2]=gengxinxc2(XX,Csm2,choose2,site_che2,Body_che2,t);
        end
        if choose2==70
            Line_ru(Body_che2,5)=Line_ru(Body_che2,5)+1;
        else
            num_chu=num_chu+1;
            Line_chu(num_chu,:)=Line_ru(Body_che2,1:3);
        end
```

```
    end
    Time_ting=gengxintt(Time_ting,XX,t);
    t=t+1;
end
T=t-1;
Fenshu=Defen(Line_chu(:,2:3),Line_ru(:,5),T);
end
%% 横移车 1 移动选择
function next=next_dao1(site_che,XXt)
a=100;
if site_che==-1
    n=[49,39,29,59,19,69];
    for i=1:6
        if isempty(find(XXt==n(i),1))
            a=0;
            break
        end
    end
else
    n=[49,59,39,69,29,19];
    for i=1:6
        if isempty(find(XXt==n(i),1))
            a=0;
            break
        end
    end
end
next=max(n(i),a);
end
%% 横移车 2 移动选择
function next=next_dao2(tt,XX,t,bc,sc,lr,lc,nc,need)
pow_now=lr(bc,2);
prm_now=lr(bc,3);
n=[2,70];
no=fix(sc/10);
ttt=[0,0];
n_pw11=sum(lr(:,2));
n_pw22=318-n_pw11;i=2;
for ii=1:6
    if (ii~=no)&&(ttt(2)<tt(ii,2))
    ttt=tt(ii,:);
    end
end
if nc>=2
```

```
    n_pw1=sum(lc(1:nc,2));
    n_pw2=nc-n_pw1;
    n_pw1=n_pw11-n_pw1;
    n_pw2=n_pw22-n_pw2;
    nn_pw=n_pw2/n_pw1;
    if nn_pw<1.55%选择模式
        F=2;%混动直接过
    elseif nn_pw>2.45
        F=1;%燃油直接过
    else
        F=2;%尽量满足
    end
    pow_bef=lc(nc-1:nc,2);
    prm_bef=lc(nc,3);
    good_need=need(pow_bef(1)+1,pow_bef(2)+1,2);
    need=need(pow_bef(1)+1,pow_bef(2)+1,F);
    if pow_now==good_need
        i=1;
    elseif ttt(1)~=0
        if lr(ttt(1),2)==good_need
        i=2;
        end
    elseif pow_now==need
        i=1;
    end
    if i==2&&prm_now==prm_bef
        i=1;
    end
elseif nc==0
    if pow_now==1
        i=1;
    else
        i=2;
    end
else
    if pow_now==0
        i=1;
    else
        i=2;
    end
end
if lr(bc,5)>=3
    i=1;
end
```

```
if 1-isempty(find(XX(:,t)==70,1))
    i=1;
end
next=n(i);
end
%%模拟退火
function
[F_good,Line_ru_good,Line_chu_good,XX_good,Csm1_good,Csm2_good,T_good,t]=Tui
huo(nx,times,chushit,jieshut,rate)
%nx 表示附件几，XX_good 最后的最优的结果矩阵,F_good 目标函数，t 为 times 模拟
退火计算时间
%times 退火次数.chushit 退火初始温度,jieshut 退火结束条件,rate 冷却速率
[F_good,Line_ru_good,Line_chu_good,XX_good,Csm1_good,Csm2_good,T_good]=Tanla
n1(nx);
Line_ru=Line_ru_good;
Fenshu=F_good;
tic
for i=1:times
    previousLr=Line_ru;previousF=Fenshu;
    temperature=chushit;                %初始化温度
    iterations=1;                       %初始化初值
    temperature_iterations=1;           %初始化温度迭代次数，当迭代温度达到 100
时，进行降温
    NEWLr_iterations=1;                 %总迭代次数

    while jieshut<temperature
        temp_Lr=newLr(previousLr);  %随机生成相邻解

[new_XX,new_Line_ru,new_Line_chu,new_Csm1,new_Csm2,new_F,new_T]=SSf(temp_
Lr);%计算相邻解的目标分数
        diFF=previousF(1)-new_F(1);         %计算改变的目标分数
        %diFF<0 说明当前的分配比之前的更优，换。不满足则以一定概率选中此解
        if (diFF<=0)||(rand<exp(-diFF/(temperature)))
            previousLr=new_Line_ru;             %接受新分配方案
            previousF=new_F;
            temperature_iterations=temperature_iterations+1; %更新温度和计数迭代
次数
            NEWLr_iterations=NEWLr_iterations+1;
            iterations=iterations+1;
        end
        if new_F(1)>=F_good(1)
            XX_good=new_XX;
            Line_ru_good=new_Line_ru;
            Line_chu_good=new_Line_chu;
```

```
                Csm1_good=new_Csm1;
                Csm2_good=new_Csm2;
                F_good=new_F;
                T_good=new_T;
            end
            if temperature_iterations>=100 %当温度迭代次数达到 100 时，进行降温
                temperature=rate*temperature;
                temperature_iterations=0;
            end
        end
end
t=toc;
end

% 生成相邻解
function previousLr=newLr(previousLr)
n=ceil(rand*318);
nn=ceil(rand*318);
P=[6,9,10,11,7,5];
pn=previousLr(n,4);
for i=1:6
    P=P/sum(P);
end
Pc=cumsum(P);
newn=pn;
while(newn(1)==pn)
    newn=find(Pc>rand);
end
previousLr(n,4)=newn(1);
if previousLr(nn,5)==3
    newnn=2;
elseif previousLr(nn,5)==0
    newnn=1;
else
    P=[0.6,1];
    nnn=find(P>rand);
    num=[-1,1];
    newnn=previousLr(nn,5)+num(nnn(1));
end
previousLr(nn,5)=newnn;
end
```

介绍：问题二代码

```
%%
%% 问题二的贪婪算法
```

```
function [Fenshu,Line_ru,Line_chu,XX,Csm1,Csm2,T]=Tanlan2(nx)
% XX 为结果矩阵，Line_ru 为入车次序，Line_chu 出车次序，Csm1 横移车 1 运动轨迹，
% Csm2 横移车 2 运动轨迹，Fenshu 得分，T 总调度时间
% nx=2;附件几
Line_ru=[xlsread(['附件',num2str(nx),'.xlsx'],'A2:C319'),zeros(318,3)];%入车顺序
for i=1:318
    if Line_ru(i,2)==1&&Line_ru(i,3)==0
        Line_ru(i,6)=3;
    elseif Line_ru(i,2)==0&&Line_ru(i,3)==0
        Line_ru(i,6)=2;
    else
        Line_ru(i,6)=1;
    end
end
mode_limit=modelimit(nx);%入车各个模式次数上限
XX=-1*ones(318,9*318*3);%结果矩阵
Num_ting=zeros(3,10);%每种车型的各个停车点的车具数
Csm1=ones(3,9*318*3).*[0,-2,-2]';%横移机 1
Csm2=Csm1;%横移机 2
Line_chu=zeros(318,3);%出车顺序
t=1;num_chu=0;%已出车数量
mode_choose=5;%出车模式
Che_need=[3,3,3];%所需车辆
while(1-isempty(find(XX(:,8586)~=2,1)))
    XX=gengxinxx(XX,t);
    choose1=0;
    while(Csm1(1,t)==0&&choose1~=100)%横移机 1 可进行工作时
        Body_che1=find(XX(:,t)==-1|XX(:,t)==79);
        if isempty(Body_che1)
            break
        end
        Body_che1=Body_che1(1);
        site_che1=XX(Body_che1,t);
        choose1=next_dao1(Line_ru,Body_che1,XX(:,t));
        if Line_ru(Body_che1,5)==0
            Line_ru(Body_che1,4)=fix(choose1/10);
        end
        if site_che1==-1&&choose1==49
            XX(Body_che1,t:t+8)=49;
        elseif choose1~=100
            [XX,Csm1]=gengxinxc1(XX,Csm1,choose1,site_che1,Body_che1,t);
        end
    end
```

```
    if mode_choose==0&&sum(Num_ting(:,1))>0&&Csm2(1,t)==0

[mode_choose,mode_limit,Che_need]=modechoose(mode_choose,mode_limit,Che_need,N
um_ting);
    end
    choose2=0;
    while(Csm2(1,t)==0&&mode_choose>0&&choose2~=100)%横移机 2 可进行工作时
        [choose2,Body_che2,site_che2]=next_dao2(Che_need,XX(:,t),nx,Line_ru);
        if site_che2==40&&choose2==2
            XX(Body_che2,t:9*318*3)=2;
        elseif choose2~=100
            [XX,Csm2]=gengxinxc2(XX,Csm2,choose2,site_che2,Body_che2,t);
        end
        if choose2==70
            Line_ru(Body_che2,5)=Line_ru(Body_che2,5)+1;
        end
        if choose2==2
            num_chu=num_chu+1;
            Line_chu(num_chu,:)=Line_ru(Body_che2,1:3);
            Che_need(1)=Che_need(1)+1;
            if Che_need(1)>Che_need(2)
                mode_choose=0;
            end
        end
    end
    Num_ting=gengxinnt(Num_ting,XX(:,t));
    t=t+1;
end
T=t-1;
Fenshu=Defen(Line_chu(:,2:3),Line_ru(:,5),T);
end
%% 模式选择函数 √
function [modech,modelt,need]=modechoose(modech,modelt,need,Nt)
if modelt(1)>0&&Nt(1,3)>=3&&Nt(2,2)>=1&&Nt(3,2)>=2
    modech=1;
    modelt(1)=modelt(1)-1;
    need=[3,8,3,1,2,3,1,1];
else
    if Nt(2,3)>=2&&Nt(3,3)>=1&&modelt(2)>0
        modech=2;
        modelt(2)=modelt(2)-1;
        need=[3,5,3,2,2];
    else
        if
```

```
modelt(1)==0&&modelt(2)==0&&Nt(1,1)>=1&&Nt(2,1)>=1&&Nt(3,2)>=1&&modelt(3)
>0
                modech=3;
                modelt(3)=modelt(3)-1;
                need=[3,5,3,1,2];
            else
                if
modelt(1)==0&&modelt(2)==0&&modelt(3)==0&&modelt(4)>0&&Nt(1,1)>=1&&Nt(3,1
)>=1
                    modech=4;
                    modelt(4)=modelt(4)-1;
                    need=[3,4,3,1];
                else
                    if Nt(2,1)>0&&Nt(3,1)>0&&modelt(5)>0
                        modech=5;
                        modelt(5)=modelt(5)-1;
                        need=[3,4,3,2];
                    else
                        if Nt(3,1)>0&&modelt(6)>0
                            modech=5;
                            modelt(6)=modelt(6)-1;
                            need=[3,3,3];
                        end
                    end
                end
            end
        end
end
end
%% 模式使用上限生成函数 √
function mode_limit=modelimit(nx)
if nx==1
    nn=[5,25,7,7,15,148]';
elseif nx==2
    nn=[4,30,7,10,60,44]';
end
mode_limit=[nn,[3,1,2;0,2,1;1,1,1;1,0,1;0,1,1;0,0,1]];
end
%% 横移车 1 移动选择  √
function next=next_dao1(lr,bc,XXt)
a=100;
n=[49,29,39,19,59,69];
if lr(bc,6)==3
    for i=1:2
```

```
            if isempty(find(XXt==n(i),1))
                a=0;
                break
            end
        end
elseif lr(bc,6)==2
        for i=3:4
            if isempty(find(XXt==n(i),1))
                a=0;
                break
            end
        end
else
        for i=5:6
            if isempty(find(XXt==n(i),1))
                a=0;
                break
            end
        end
end
next=max(n(i),a);
end
%% 横移车 2 移动选择
function [next,bc,sc]=next_dao2(cneed,XXt,nx,Lr)
need=cneed(cneed(1));
next=100;
if nx==1
        n=[1,4,9,32,66,74,91,105,117];
else
        n=[1,4,8,47,59,79,150,200,251];
end
if need==1
        for i=50:10:60
            bb=find(XXt==i);
            if 1-isempty(bb)
                next=2;sc=i;bc=bb;
                break
            end
        end
end
if need==2
        for i=30:-20:10
            bb=find(XXt==i);
            if 1-isempty(bb)
```

```
                next=2;sc=i;bc=bb;
                break
            end
        end
end
if need==3
        for i=40:-20:20
            bb=find(XXt==i);
            if 1-isempty(bb)
                next=2;sc=i;bc=bb;
                break
            end
        end
end
a=0;
for i=1:9
        if XXt(n(i))>9&&mod(XXt(n(i)),10)==0
            a=1;break
        end
end
if a&&Lr(n(i),5)==0
        next=70;bc=n(i);sc=XXt(n(i));
end
if next==100
        bc=0;sc=0;
end
end
```