中国研究生创新实践系列大赛

中国光谷·“华为杯”第十九届中国研究生数学建模竞赛

学　　校	福州大学
参赛队号	22103860012
队员姓名	1.钱政鑫
	2.黄国强
	3.李云祥

中国研究生创新实践系列大赛

中国光谷·“华为杯”第十九届中国研究生数学建模竞赛

题 目　　C 汽车制造涂装-总装缓存调序区调度优化问题

摘　　要：

PBS（Painted Body Store，汽车制造涂装-总装缓存调序区）是汽车制造厂内一种车辆调序设施，为解决涂装车间与总装车间因生产偏好不同而需要对加工车型进行重新排序的问题提出了一种方案。PBS 主要有出车口、接车机、进车道、返回道、送车机和接车口六个部分组成，在涂装车间完成涂装的车身将以一种固定的序列进入出车口，通过接车机和送车机的**调控**和在 PBS 的各区域共同配合下，应使得在接车口输出的车身序列符合总装车间的生产偏好。不乏前人已经对该类有约束情况的车辆排序问题有了一定的研究，其大多是使用**遗传算法**、**基于规则的启发式优化算法**、**粒子群算法**和 **BP 神经网络算法**等优化模型、手段对该问题进行优化处理，得出的结果对提高 PBS 运行效率有着不等的效果。

本文从寻找能使 PBS 调序设备能更加按照优化目标输出车辆序列的角度出发，利用 Maltab 代码建立了一套符合 PBS 内部逻辑和约束的**全过程模拟模型**，在模型中对随机生成的车辆进场序列和车身属性进行模拟运行，观察车辆运行情况并结合题目中所给出的约束说明以验证 Maltab 模拟过程的正确性，为后文的分析和经验总结做基础。

而关于优化目标，题目中已给出详细的量化说明和注解，并提供了它们的权重系数，可将该问题视为单目标优化问题，其总分越高代表该调控手段下 PBS 调序系统的优化能力越好，本文对需量化的分数进行了数学化表示，并将量化分数作为模型优化的优化目标。

针对使用**基于规则的启发式优化算法**需要掌握一定实际经验和生产规律的问题，本文对随机生成的车辆进场序列和车身属性数据的模拟结果进行绘图，并用**遗传算法**对参数进行迭代优化处理得到了比随机生成参数更好的参数情况，在互相对比数据和分析中，总结出得分较高的原因，作为经验和规律，指导作出接车机和送车机的**启发式调度规则**。

针对问题一，在全约束条件下，接车机的调度状态需优先考虑返回道的需要，送车机的调度状态需优先考虑等待时间长的车身。依靠经验和规律，本文提出以下调度规则：接车机以将车辆以进车道 4＞3＞2 = 5＞1 = 6 的优先级送入可进入的进车道；送车机在保证模拟过程不循环的情况下使出车序列不扣分。对使用该调度手段的分数结果与**遗传算法**进行优化后的结果相比较，发现使用该调度手段与通过遗传优化后的优化对 PBS 优化调度模型有同样的促进效果，故使用该调度规则用于 PBS 优化调度模型。最后，该模型应用于附件 1 的得分结果为 **6.457 分**，应用于附件 2 的得分结果为 **30.555 分**，其他调度输出结果存入附件提交。

针对问题二，在去除 6)、7)条约束条件下，接车机可自由选择在出车口或返回道取车，送车机也可自由选择正等待车身进行取车。依靠问题一的经验，本文提出以下调度规则：接车机将同一车型置于同一进车道以供送车机有计划的取车；送车机优先选择不使目标扣分的车辆放入接车口，并保证每辆车身最多通过放回道两次。通过改写代码逻辑嵌入以上调度规则后，完成了对该问题的 PBS 优化调度模型优化，将该更新后的模型应用于附件 1

的得分结果为 **9.093 分**，应用于附件 2 的得分结果为 **32.160 分**，其他调度输出结果存入附件提交。可见，通过去除 6)、7)条约束条件可以提高 PBS 整体调度能力，输出结果更加符合总装车间的车身序列。

关键词：PBS 调度问题；基于规则的启发式优化算法；遗传算法；动态全过程模拟

目 录

1 问题重述 5

1.1 问题背景 5

1.2 问题约束 5

1.3 问题目标 6

1.4 问题描述 6

2 基本假设与符号系统 7

2.1 基本假设 7

2.2 符号系统 7

3 问题一的分析与求解 8

3.1 问题一的分析 8

3.2 模型的搭建 8

3.2.1 车辆实时位置信息模型 8

3.2.2 目标量化计分模型 10

3.2.3 模型处理复杂性的分析 11

3.3 简化 PBS 调度运行的模拟及分析 11

3.3.1 随机生成参数的 PBS 调度运行结果 12

3.3.2 遗传算法优化的 PBS 调度运行结果 13

3.3.3 对调度结果的分析与经验总结 14

3.4 结合历史经验的接送车机启发式规则提出及实现 14

3.4.1 接车机启发式调度规则及其调度方法 14

3.4.2 送车机启发式调度规则及其调度方法 15

3.5 模型求解与分析 16

3.5.1 仅对送车机使用启发式调控规则的结果 16

3.5.2 接车机、送车机皆使用启发式调控规则的结果 18

3.5.4 模型的比较 18

3.6 问题一的结果 18

4 问题二的分析与求解 19

4.1 问题二的分析 19

4.2 接送车机启发式调度规则的更新与实现 19

4.2.1 接车机启发式调度规则的更新 19

4.2.2 送车机启发式调度规则的更新 20
4.2.3 过程模拟代码修改 20
4.3 模型求解与分析 20
4.4 问题二的结果 20
5 模型的评价 21
5.1 模型的优点 21
5.2 模型的缺点 21
5.3 模型的改进 21
6 参考文献 22
附录 程序代码 23

1 问题重述

1.1 问题背景

PBS（Painted Body Store，汽车制造涂装-总装缓存调序区）是汽车制造厂内一种车辆调序设施，解决了涂装车间与总装车间因生产偏好不同而需要对加工的车型进行重新排序的问题。其所生成的满足总装车间生产约束的车辆序列，能够有效调节装涂车间和总装车间由于生产节奏不一致而产生的“阻塞”或“饥饿”现象，保证汽车生产过程的有序平稳，提高汽车制造厂的生产效率[1]。

目前该汽车制造厂所使用的 PBS 系统有如下 7 个区域配置：PBS 由涂装-PBS 出车口（后文均简称为“出车口”）、接车横移机（后文均简称为“接车机”）、进车道 6 条（有 10 个停车位）、返回道 1 条（同有 10 个停车位）、送车横移机（后文均简称为“送车机”）和 PBS-总装接车口（后文均简称为“接车口”），具体位置关系与运行方向见图 1-1 所示。

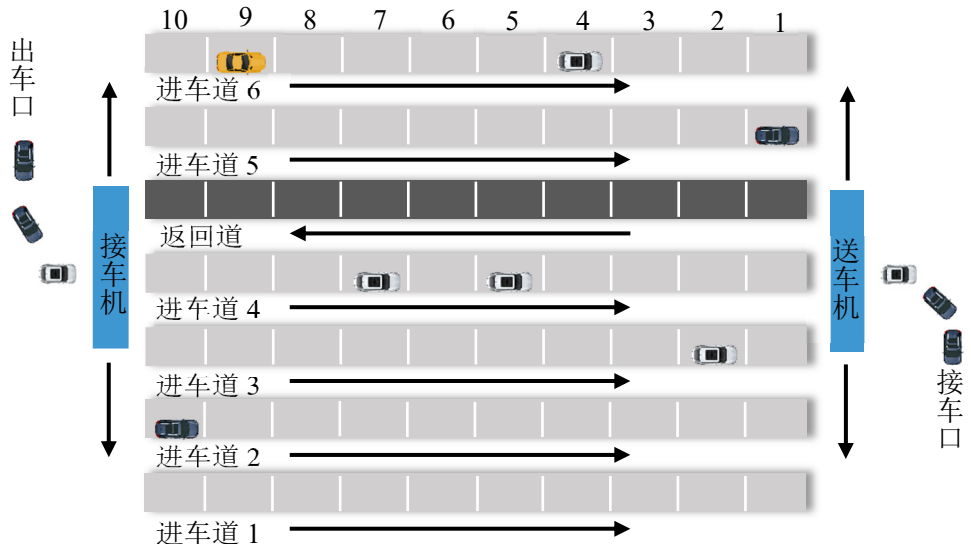


图 1-1　PBS（汽车制造涂装-总装缓存调序区）示意图

各区域能完成的功能如下：

出车口：按照涂装车间完成车辆的顺序不断将车辆放入 PBS 入口。

接车机：将车辆从出车口或者返回道的第 10 停车位送入合适的进车道第 10 停车位，拥有一定的可选调度手段。

进车道：拥有 10 个停车位，并不断将各停车位上的车辆向前移动，如停车位前方无车时，车辆每 9 秒可向前移动一个停车位。

返回道：除运行方向与进车道相反外，其余各功能同进车道相同。

送车机：将车辆从进车道第 1 停车位适当地送入接车口或者返回道的第 1 停车位，同样拥有一定的可选调度手段。

接车口：从送车机上将车辆接过并进入总装车间进入下一步生产工序。

现本文所需解决的问题是，在 PBS 各区域正常运行不中断的情况下，通过调控接车机和送车机的行为，使得重新排序的车辆序列更加符合总装车间的生产偏好，并输出各车辆在各时间下的 PBS 位置代号。

1.2 问题约束

1）各区域功能设定的任务均不可强行变更，只能改变接车机和送车机的行为，其余区域设备按功能不断运行。

2）各停车区域、接车机和送车机工作时，同一时刻最多只能有一个车辆停留。

3）接车机和送车机必须从初始位置（进车道 4 第 10 停车位附近）开始动作并结束动作，并在该动作过程中不可被中断，即任务下达不可被更改和停留。

4）如有进车道中第 1 停车位有车辆等待时，送车机必须处于工作状态。

5）送车机和接车机为 1-6 进车道进行运输车辆并返回初始位置所耗费的时间分别为[18,12,6,0,12,18]秒；送车机和接车机将车辆在 1-6 进车道与返回道进行运送的时间分别为[24,18,12,6,12,18]秒。

*6）返回道第 10 停车位上有车时，接车横移机优先处理该辆车辆。（该约束在问题二被解除）

*7）各进车道第 1 停车位有车时，送车机优先处理最先到达第 1 停车位的车辆。（该约束在问题二被解除）

1.3 问题目标

本题有四个目标需完成，设置每个目标分数初值为 100 分，每违反一次目标规定将扣除一次对应分数，最后将目标 1-4 的分数按加权系数[0.4,0.3,0.2,0.1]相加做为最后的量化成绩。

目标一：需尽量保证各混动车中间隔的非混动车为 2 辆，即每连续两辆混动车之间的非混动车数量不为 2 将扣 1 分。

目标二：四驱车与两驱车在出车序列中应满足 1:1 分布，否则一次将扣 2 分。

目标三：应尽量少的使用返回车道，使用一次放回道将扣 1 分。

目标四：应保证所用调序时间（T）尽量短，因不考虑车辆调序，所有车辆最快通过的时间为 $t=9*$（车辆数）+72，则将对超过该时间的部分（$T-t$）进行罚分，一次性扣除 0.01*（$T-t$）分。

1.4 问题描述

需根据题目所给出的条件和限制，完成如下任务：

问题 1：在全约束条件下完成车辆优化调度

结合 PBS 调度能力和限制，并考虑上述所有条件下，建立起相应的 PBS 优化调度模型，使得总装车间得到的出车序列满足生产需求，即上述四个生产目标。并根据附件 1 与附件 2 所给的涂装车间出车序列，利用模型进行优化，输出得分结果，将各车辆的各时刻下的位置信息进行记录。

问题 2：在去除*6）与*7）约束的情况下完成车辆优化调度

在问题 1 的基础上，去除约束条件*6）与*7），对 PBS 优化调度模型进行调整，使得 PBS 调度过程更加合理。同样，需将附件 1 与附件 2 所给的涂装车间出车序列，利用调整后的模型进行优化，输出得分结果，将各车辆的各时刻下的位置信息进行记录。

2 基本假设与符号系统

2.1 基本假设

为了对分析过程进行有效简化，并合理地完成涂装-总装缓存调序区（PBS）的车辆调序问题，在建立 PBS 优化调度模型时，本文提出以下假设：

假设一：各区域设备正常运行，不会提前不会延迟，各时刻车辆、PBS 区域内工作状态均按照题目要求进行。

假设二：考虑到横移机运动时的速度保持一致，为了更精细的完成 PBS 调度全过程模拟，假设接车机工作时间为其一半时完成放车动作，即[9,6,3,0,6,9]秒，送车机工作时间为其一半时完成提车工作，即[9,6,3,0,6,9]秒。为返回道服务时同理，提车和放车完成的时间为[12,9,6,3,6,9]秒。

2.2 符号系统

表 2.1　符号说明

符号	说明
t	时刻代码，表示该时刻距第一辆车进入 PBS 过去了 t (s)
T	所有车辆从 PBS 通过的总时间
i	某一车辆的进车顺序，i=[1,2,…,318]
E_i	进车顺序为 i 的车辆其动力代码，燃油 E_i=1，混动 E_i=2
D_i	进车顺序为 i 的车辆其驱动代码，两驱 D_i=2，四驱 D_i=4
$S_{i,\ t}$	进车顺序为 i 的车辆在第 t 秒的状态代码
$W_{i,\ t}$	进车顺序为 i 的车辆在第 t 秒的位置代码
L_k	PBS 上各区域、各停车位代码，同附件 3
α	从涂装车间内完成工作后的出车序列，为一向量
β	从 PBS 中调序后的出车序列，为一向量
O	优化结果量化总成绩
O_j	第 j 个优化目标的成绩，j=[1,2,3,4]
Q_j	第 j 个优化目标的权重系数，Q_j=[0.4,0.3,0.2,0.1]
jieche	接车机完成现在的任务还需的时间(s)
songche	送车机完成现在的任务还需的时间(s)
fan	返回道的使用次数(次)

3 问题一的分析与求解

3.1 问题一的分析

本文的问题来自实际生产活动中，在优化的过程中需要依靠掌握大量的生产规律和现场经验赋予计算机实时判断调度方案的能力，但在缺少实际生产过程以供分析的情况下，难以对其中机理和可优化之处进行启发式规则推演和提出。故本文将首先对该 PBS 调度模式进行随机模拟运行（即不考虑返回车道且将每一辆车在进入 PBS 时随机放置在任意进车道上），并对每辆车初始随机放置的进车道参数利用遗传算法，以提高目标分数为目标进行迭代优化。建立可视化车辆状态图，在观察和分析模拟数据中总结生产规律和经验，以供提出接车机入站启发式规则和送车机出站启发式规则做准备。

利用启发式规则推演，在总结优化过程和观察车辆在 PBS 中运行状态后，本文将有针对性地提出接车机和送车机的实时调度手段，将调度手段嵌入 PBS 优化调度模型中。将启发式规则调度手段编入模型后，验证该调度手段对结果的影响，评价该复合 PBS 优化调度模型的效率和合理性。

最后，将附件 1 和附件 2 的生产数据带入模型计算，将结果以得分结果和附件的形式输出并提交。如图 3-1 为问题一的技术路线。

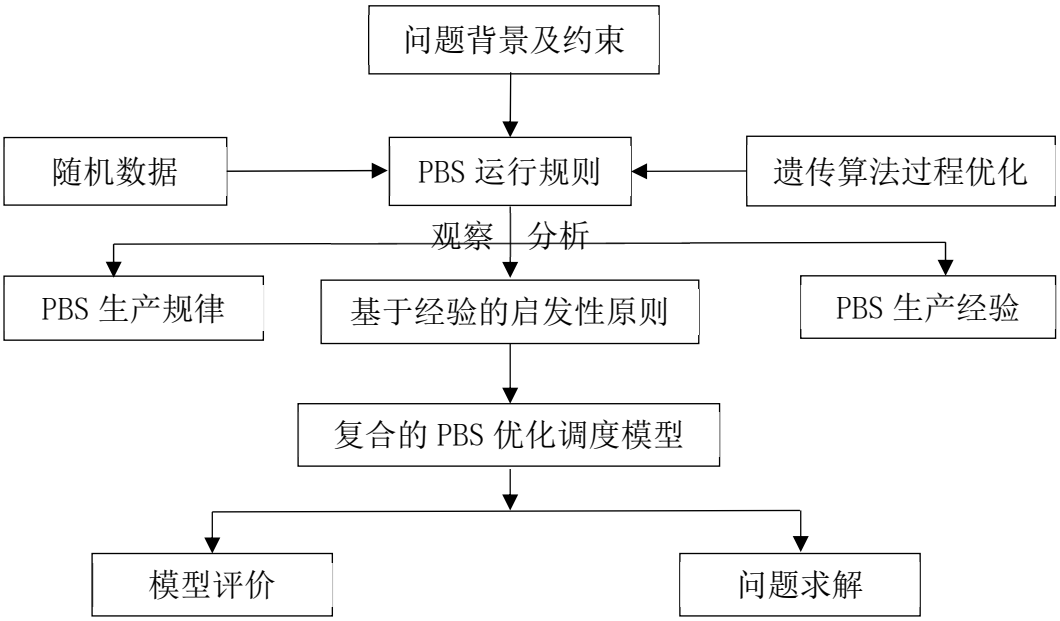


图 3-1　针对解决问题一的技术路线图

3.2 模型的搭建

3.2.1 车辆实时位置信息模型

利用 Matlab 软件用数据的形式将每一车辆通过 PBS 的过程进行全过程模拟，其伪代码如表 3.1 所示。

表 3.1　PBS 全过程模拟算法伪代码

PBS 调度全过程模拟算法伪代码
Input：涂装车间出车序列、各车辆型号数据
Output：总装车间进车序列、各时刻车辆位置信息、该模型下目标优化量化分数
Initialize：读取数据
01.jieche=0,songche=0,time=zeros(6,10), fan_time=zeros(1,10)　各区域工作时间清零

```
02.car1=zeros(6,10),fan_car=zeros(1,10)   各区域清空
03.for   t=1:100000(时间足够大)
04.   各区域时间-1
05.   if   songche=0
06.      判断将在送车机上的车送出，或放入返回道
07.      if   送出
08.         将该车状态改为 1，在序列记录该车出车序号
09.      end
10.      if   放入放回道
11.         返回道 1 车位开始计数，送车机开始计时，将车的序列标记赋值于返回道 1
12.      end
13.   end
14.   for   x=1:6
15.      for   y=2:10
16.         if   该停车位的车辆有条件先前移动
17.            car1(x,y-1)=car(x,y),car1(x,y)=0
18.         end
19.      end
20.   end
21.   if   jieche=0
22.      判断是处理返回道上的车辆还是进车口的车辆
23.      判断将该车置于哪个进车道
24.         if   处理进车口的车
25.            进车道 10 车位开始计数，接车机开始计时
26.            将车的序列标记赋值于返回道 10
27.         end
28.         if   处理返回道上的车
29.            将返回道 10 的数据清除，时间清零
30.            进车道 10 车位开始计数，接车机开始计时
31.            将车的序列标记赋值于返回道 10
32.         end
33.   end
34.   以矩阵形式记录该秒结束时，各车位置
35.   if   各车状态码皆为 1
36.      break   时间循环结束
37.   end
38.end
39.T=t   输出总时间
```

如表 3.1 可见，05~13 行伪代码解决了送车机在该时刻的动作问题，14~20 行伪代码解决了各进车道各停车位在该时刻的运行问题，21~33 行伪代码解决了接车机在该时刻的调度动作问题， 35~37 行代码为大循环的终止条件，最后各时刻车辆的信息和花费总时间会被储存并输出。通过代码全过程模拟的方式解决了车辆实时信息的全过程模拟。

3.2.2 目标量化计分模型

针对 PBS 调序区的排序优化问题 P 可以表示为一个多元组[2]：

$$P = P(\Theta, \alpha, \beta, E, D, T, fan) \tag{3.1}$$

其中，Θ 为 PBS 调序区约束逻辑，涂装车间和总装车间两区的车辆通过 PBS 进行重排序，车辆的配置和型号没有发生改变，故使用了 Θ 代表 PBS 在排序中起到的作用；α为从涂装车间内完成工作后的出车序列，为一向量；β为从 PBS 中调序后的出车序列，同为一向量；E 表示所有车辆的动力配置集合；D 表示所有车辆的驱动（两驱/四驱）配置集合；T 表示完成所有车辆经过 PBS 完成重排序的总时间；fan 表示在重排序中使用返回道的次数。

该问题 P 的目标是利用 PBS 调序区 Θ 将涂装车间内完成工作后的出车序列α转化为一个进入总装车间的重排序序列β，可以记为$\beta = \Theta(\alpha)$。易知，序列α与序列 β 中车辆数量相同，即 $n = \|\alpha\| = \|\beta\|$。那么，$E_i$ 可以表示为$(E_1, \ldots, E_i, \ldots, E_n)$，$E_i \in \Theta$ 显然可能存在 $E_i = E_j$（$i \neq j$），即 E_i 与 E_j 为动力（燃油或混动）配置相同的车辆；同理，集合 $D = (D_1, \ldots, D_i, \ldots, D_n)$ 表示所有车辆的驱动（两驱/四驱）配置集合。以上，i，$j = 1,2,\ldots,n$，n 为计划生产的车辆数量。

通过 PBS 调序后的出车序列β中，若遇到一辆混动车辆(E_i=2)发现与前一辆混动车(E_j=2)之间非混动车(E=1)的数量不是 2 时，表示违反目标一，记$\delta(E_i)$=1，式中 $E_i \in E$。反之记$\delta(E_i)$=0。即：

$$\delta(E_i) = \begin{cases} 0 & , E_i = 2\text{且与前一辆混动车中间恰有2辆非混动车} \\ 1 & , E_i = 2\text{且与前一辆混动车中间非混动车数不为2} \end{cases} \tag{3.2}$$

通过 PBS 调序后的出车序列β中，将出车序列按照两驱车和四驱车进行分块，假设可将出车序列β分为 l 块，若每一块区域中四驱车和两驱车的比例不为 1:1，则记$\tau(l) = 2$，反之记$\tau(l) = 1$。即

$$\tau(l) = \begin{cases} 0 & ，\text{区域块内四驱车和两驱车的比例为1:1} \\ 2 & ，\text{区域块内四驱车和两驱车的比例不为1:1} \end{cases} \tag{3.3}$$

问题 P 所要求的优化量化分数为 $O(\beta, fan, T) = Q_1O_1(\beta) + Q_2O_2(\beta) + Q_3O_3(fan) + Q_4O_4(T)$，即目标 1-4 的加权分数最大值，其中 Q_i 为权重系数 Q_i=[0.4,0.3,0.2,0.1]，$i = 1,2,3,4$。其他相关公式如下：

$$O_1 = 100 - \sum_{i=2}^{n} \delta(E_i) \tag{3.4}$$

$$O_2 = 100 - \sum_{l=1}^{n} \tau(l) \tag{3.5}$$

$$O_3 = 100 - fan \tag{3.6}$$

$$O_4 = 100 - 0.01(T - 9n - 72) \tag{3.7}$$

其中，*fan* 为使用返回车道的次数，*n* 为需通过 PBS 的车辆总数。

那么，问题 *P* 的数学模型可表达为：

$$\max \quad O = 0.4O_1 + 0.3O_2 + 0.2O_3 + 0.1O_4$$

$$s.t. \quad \begin{cases} \beta = \Theta(\alpha) \\ E = (E_1, ..., E_i, ..., E_n) \\ D = (D_1, ..., D_i, ..., D_n) \\ E \in \beta, \ D \in \beta \end{cases} \tag{3.8}$$

3.2.3 模型处理复杂性的分析

对于附件中所给出的排序序列长度 $n = ||\beta|| = 318$，进车道规模为 6 个的排序问题，问题的规模至少为 $6^{318} \approx 2.83 \times 10^{247}$。显然这种问题不可能通过遍历解析的方法求解的。从问题的复杂性和可操作性的角度上考虑，利用启发式算法提出启发式原则进行优化较为有利。

不可否认的是，单纯地使用启发式优化算法将会使得优化结果太过依赖于人机交互，使得优化结果与人的判断好坏相关，故本文将结合遗传算法不断迭代优化种群的特点与启发式优化算法结合使用，帮助人脑突破发现更优秀的启发式规则。

本文将对随机生成参数模拟 PBS 的运行情况，并对运行结果进行遗传算法优化，在优化过程中观察成绩较好的子代的运行情况，并以此为经验和生产规律，提出启发式规则，以求更加地优化该 PBS 调序区模型。

3.3 简化 PBS 调度运行的模拟及分析

为了深化对 PBS 运行内部逻辑的理解，本节在不考虑使用返回车道并对每一车辆随机选择进车道进入的运行结果进行模拟。随后，对随机结果利用遗传算法进行迭代优化，得出量化分数更为优秀的例子，如图 3-2 所示为不断迭代后子代量化分数变化的示意图。

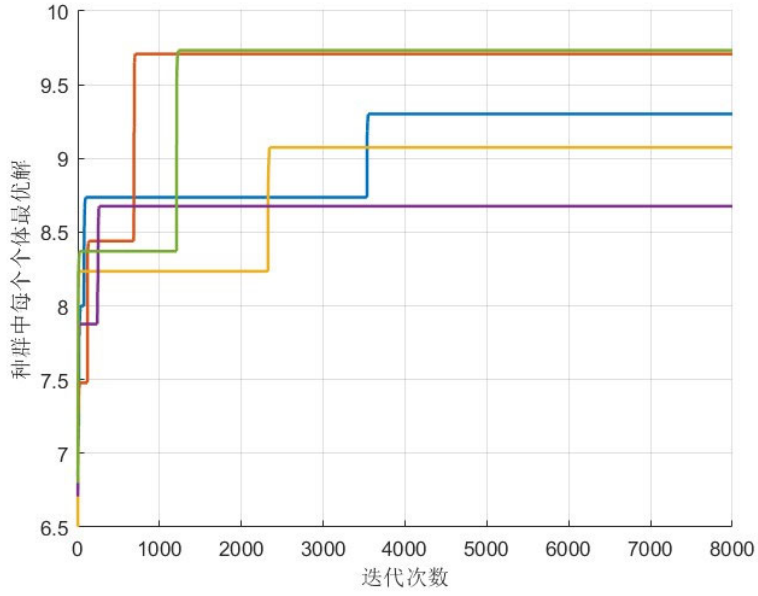


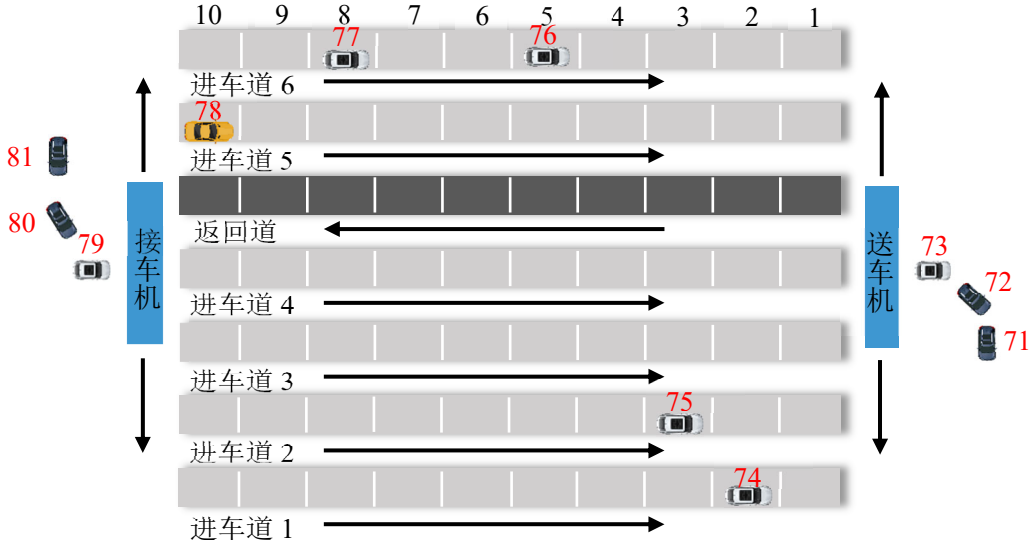
图 3-2　遗传算法优化过程

在图 3-2 中，每条不同颜色的线表示每一种种群在迭代过程中的最优解个体的分数。通过观察随机数据和算法优化后生成的 PBS 模拟结果，总结分数更优的因素，为后文提

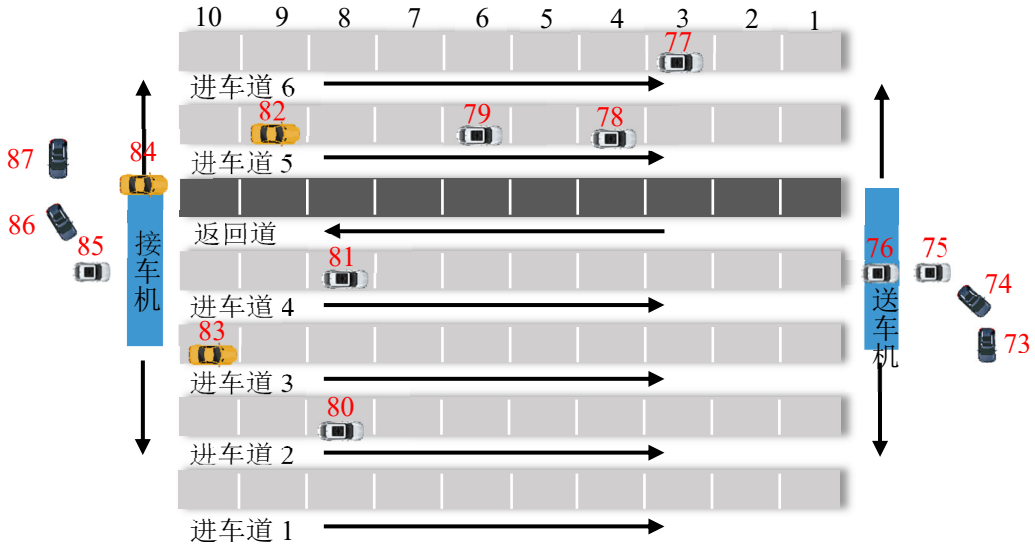
出启发式调度方案做指导。

3.3.1 随机生成参数的 PBS 调度运行结果

在每一车辆进入 PBS 时，将其进入进车 1-6 设置为随机数，即其随机进入一条进车道的情况利用 Matlab 进行动态全过程模拟，在模拟进行至 1000s 和 1050s 时它们的结果如图 3-3 所示：



(a) t=1000s



(b) t=1050s

图 3-3　随机生成参数的 PBS 在 1000s、1050s 时的运行结果

可见，在第 1000s 时，各进车道 1 号停车位内无车辆，送车机处于空闲状态。仅 1、2、5、6 号进车道内有车辆依次向各车道 1 号停车位行驶。79 号车辆在出车口等待接车机送入车道，此时返回车道内没有车辆。

在第 1050s 时，与第 1000s 相比各车辆都向前移动了若干距离，序号分别为 74、75 的车辆被送出了 PBS，送车机正在为序号为 76 的车辆做送出工作。各进车道内车辆分布较为均匀，85 号车辆在出车口等待接车机送入合适的进车道，77-84 号车辆依次向不同车道的 1 号停车位行驶，此时返回车道内没有车辆。

3.3.2 遗传算法优化的 PBS 调度运行结果

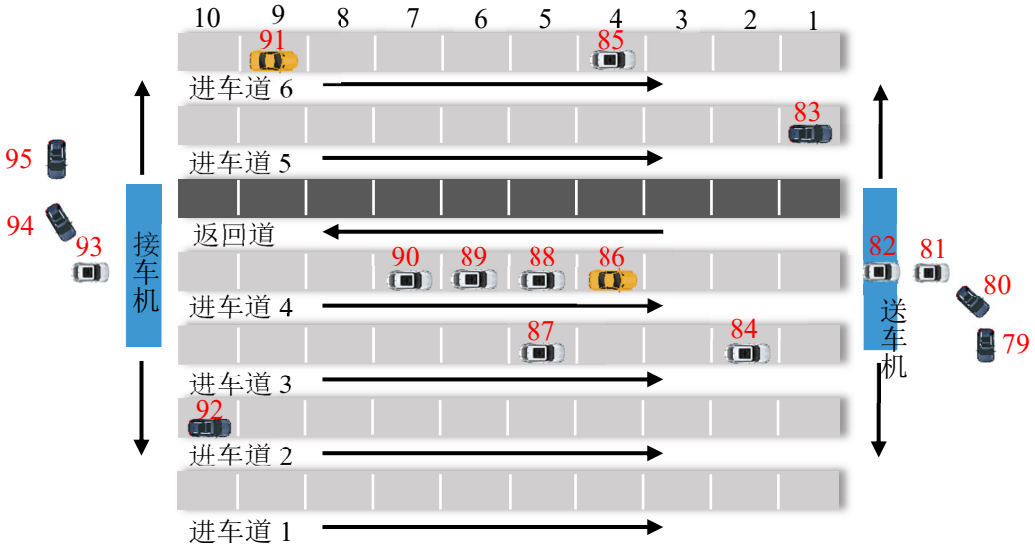
遗传算法(GA)[3]是人工模拟自然界优胜劣汰过程的一种优化算法，通过算子选择、交叉和变异三个核心步骤计算出适应度更高的个体。

将各车辆进入进车道 1-6 的随机数序列利用遗传算法对量化分数进行优化。易知，任意随机数序列皆为 PBS 运行模型的可行解，故可以选择进车道的随机序列作为染色体编码，每一车辆进入的进车道即为基因；将 PBS 模型的量化分数作为该个体的适应度，以分数尽量高为目标，进行迭代，遗传算法中各参数设定如表 3.2 所示。

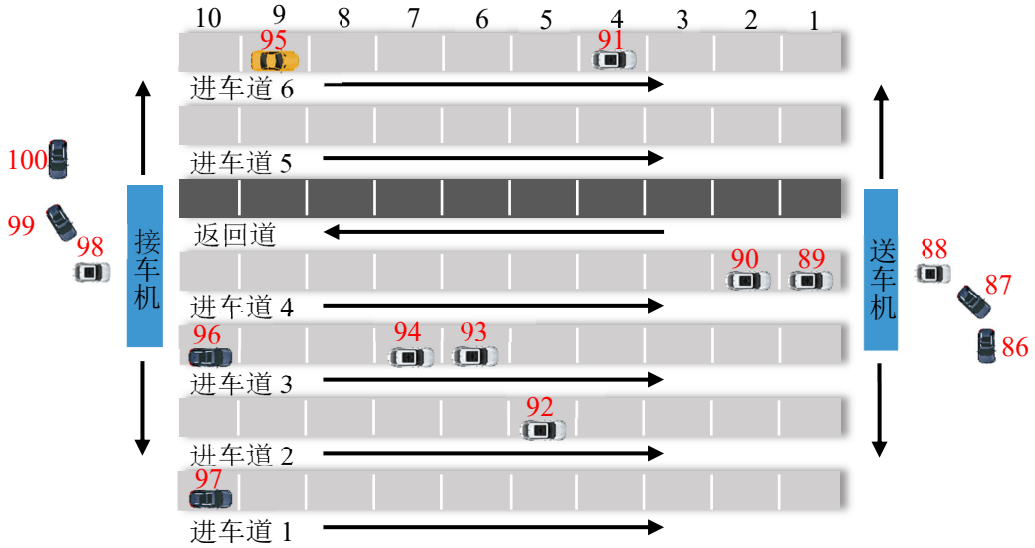
表 3.2　遗传算法参数设置

最大迭代次数	种群规模	交叉率	变异率
8000	5	0.45	0.1

下面给出优化过后优秀个体序列进入 PBS 系统的模拟情况，其模拟进行至 1000s 和 1050s 时它们的结果如图 3-4 所示：



(a) t=1000s



(b) t=1050s

图 3-4　输入遗传优化参数的 PBS 系统在 1000s、1050s 时的运行结果

可见，在第 1000 秒时， 82 号车辆正由送车机送至接车口，83 号车辆到达 5 号进车道的 1 号停车位，等待送车机运送。93 号车辆在出车口等待接车机送入合适的进车道，83-92 号车辆依次向不同车道的 1 号停车位行驶，此时返回车道内没有车辆。

在第 1050 秒时，此时 4 号进车道内的 89 号车到达 1 号停车位，等待送车机运送。送车机将根据出车口内已有的车辆序列决定接收 89 号车辆或将 89 号车辆送入返回道。98 号车辆等待接车机选择合适的进车道，90-97 号车辆依次向不同车道的 1 号停车位行驶，此时返回车道内没有车辆。

3.3.3 对调度结果的分析与经验总结

从宏观上看，同为 t=1000s，随机生成的参数与优化后的参数模拟结果对比，后者车辆运行更快，序列为 81 的车辆已经完成调序，但前者才进行到序列为 73。相同时间节点下由遗传算法优化的 PBS 调度系统的车辆传送速度明显快于随机生成参数的 PBS 调度系统的调度结果。可见，经过遗传算法后的结果优化效果明显。

从微观上看，由遗传算法优化的 PBS 调度模拟情况在进车道 4 的车辆分布最为密集，进车道 3、进车道 2、进车道 5 次之，进车道 1 和进车道 6 车辆最少，车辆基本按照中间车道密集，两边车道稀疏的规律排布，这样的调度规律可以有效的节约总调度时间，与优化目标 3 是一致的。距离接车机远处的进车道 1 和进车道 6 的车辆则可以用作优化目标 1 和 2 的考虑使用。

最后比较，随机数据和优化数据得出结果的最终分数为 4.230 分和 8.934 分，有一定的提升。

3.4 结合历史经验的接送车机启发式规则提出及实现

本节的主要目的是：基于问题的目标和可供调配的资源，利用启发式算法指定接送车机的调度规则。启发式算法可以获得局部最优解，但无法保证一定得到模型最优解，故对规则的制定是关键，本文经过提出验证在不断改进取优的过程后提出以下接车机和送车机的启发式调度规则。

3.4.1 接车机启发式调度规则及其调度方法

在对 PBS 模拟中，本文发现车辆在进入 PBS 调序系统时越接近中间进车道（即进车道 4、进车道 3 等），最后得出的分数越高，符合了目标 4 节约时间的优化目标，故本文针对优化目标 4 为主要目标，制定了以下接车启发式规则，按优先级由高到低的顺序如下：

1)尽量将车送入进车道 4 以节约时间；

2)若进车道 4 第 10 停车位被占用，则选择进车道 3 第 10 停车位；

3)若进车道 3 第 10 停车位被占用，则选择进车道 2 或进车道 5 中车数少的一条进车道，以此类推，选择进车道 1 或进车道 6 的情况相同。

则基于 PBS 调序区的接车机送车逻辑判断流程如图 3-5。

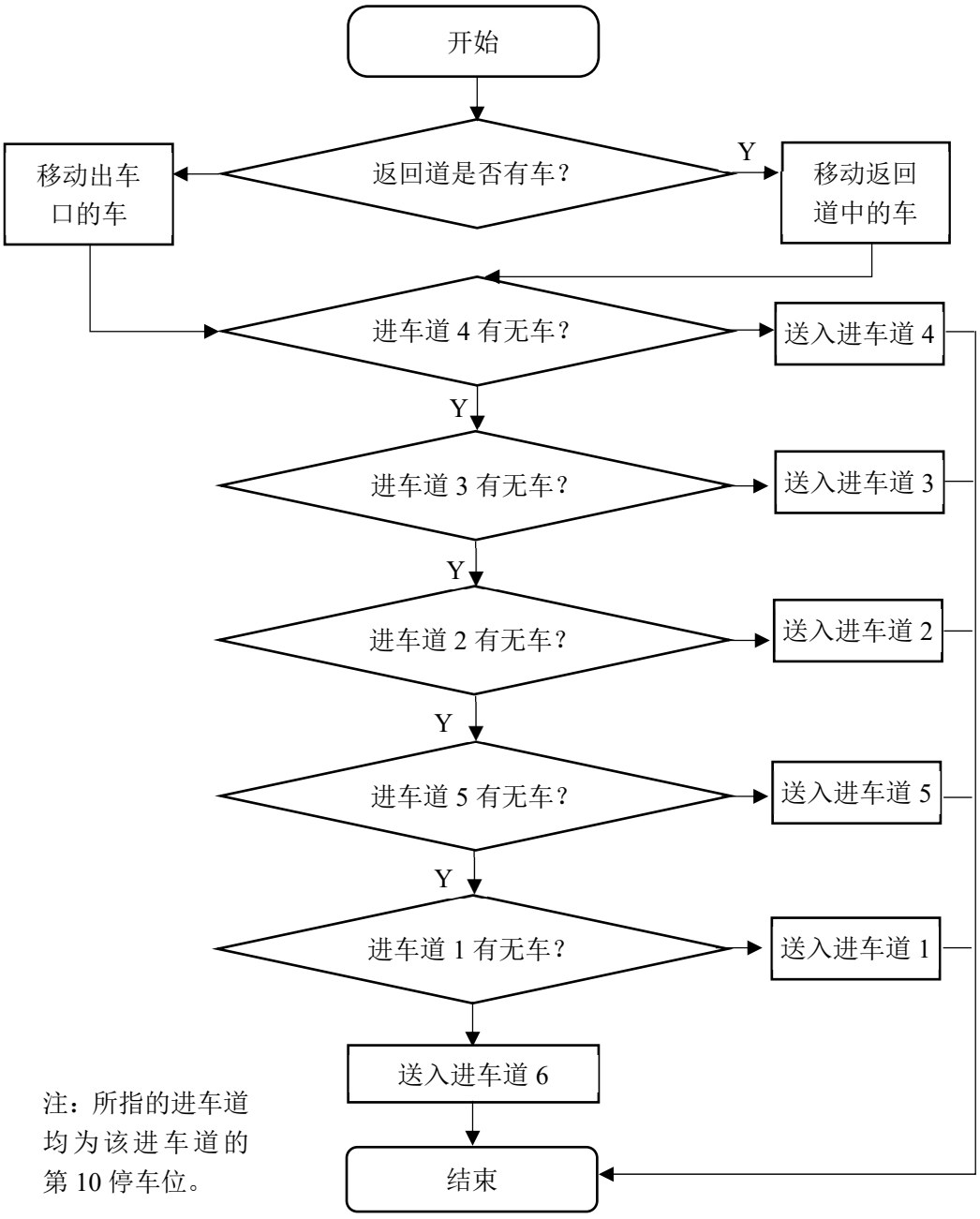


图 3-5　接车机调度方案

3.4.2 送车机启发式调度规则及其调度方法

送车机送出的车辆序列即为目标分数评分的具体依据，故本文认为对送车的调度原则应重点考虑，通过合理的调度方案使得车辆送出序列符合条件约束，从而减少扣分的产生以提高目标优化的分数。

在上述分析中，本文得出以下送车机调度经验：

混动车型及燃油车型进入接车口序列逻辑：如果出库车前一辆车为混动车型且车道上有燃油车型，若送车横移机接到混动车型，则进入返回车道，直至接到燃油车型，将其出库。如果出库车前两辆车第一辆车为混动车型，第二辆车为燃油车型且车道上有燃油车型，若送车横移机接到混动车型，则进入返回车道，直至接到燃油车型，将其出库。如果出库车前两辆车均为燃油车型且车道上有混动车型，若送车横移机接到燃油车型，则进入返回车道，直至接到混动车型，将其出库。

四驱车型及两驱车型进入接车口序列逻辑：由两驱车型变为四驱车型直至变为两驱车型或由四驱车型变为两驱车型直至变为四驱车型为一小块；考虑每一块中相同驱动方式车辆的个数，若车道上有相同数量的不同种驱动方式的车辆，且送车横移机接到相同种驱动方式车辆，则将车辆送至返回到，直至接到不同种驱动方式车辆，此时送车横移机将车辆送至接车口；当接到的不同种驱动方式车型的车辆数量与相同种驱动方式车型的车辆数量相等时，若接收到不同种驱动方式的车型时，送车横移机将车辆送入返回道，直至接到相同种驱动方式车型的车辆为止，并分块；当四驱车型及两驱车型进入接车口序列逻辑与混动车型及燃油车型进入接车口序列逻辑发生冲突时，考虑到优化目标 1 的权重大于优化目标 2 的权重，优先考虑混动车型及燃油车型进入接车口序列逻辑。

故本文综合考虑各个优化目标，在不至于车辆在 PBS 中不断循环的前提下，结合约束条件，制定了以下送车启发式规则，按优先级由高到低的顺序如下：

1)若该车辆不至于造成扣分(即驱动方式与上一出站车辆相同且符合优化目标 1)，则将该车放入接车口；

2)若返回道第 1 停车位被占用，则将车放入接车口；

3)为避免最后有车进入死循环，不考虑最后 10 辆车的返回问题，将其全部放入接车口；

4)将有扣分风险的车辆放入返回道。

则基于 PBS 调序区的送车机送车逻辑判断流程如图 3-6。

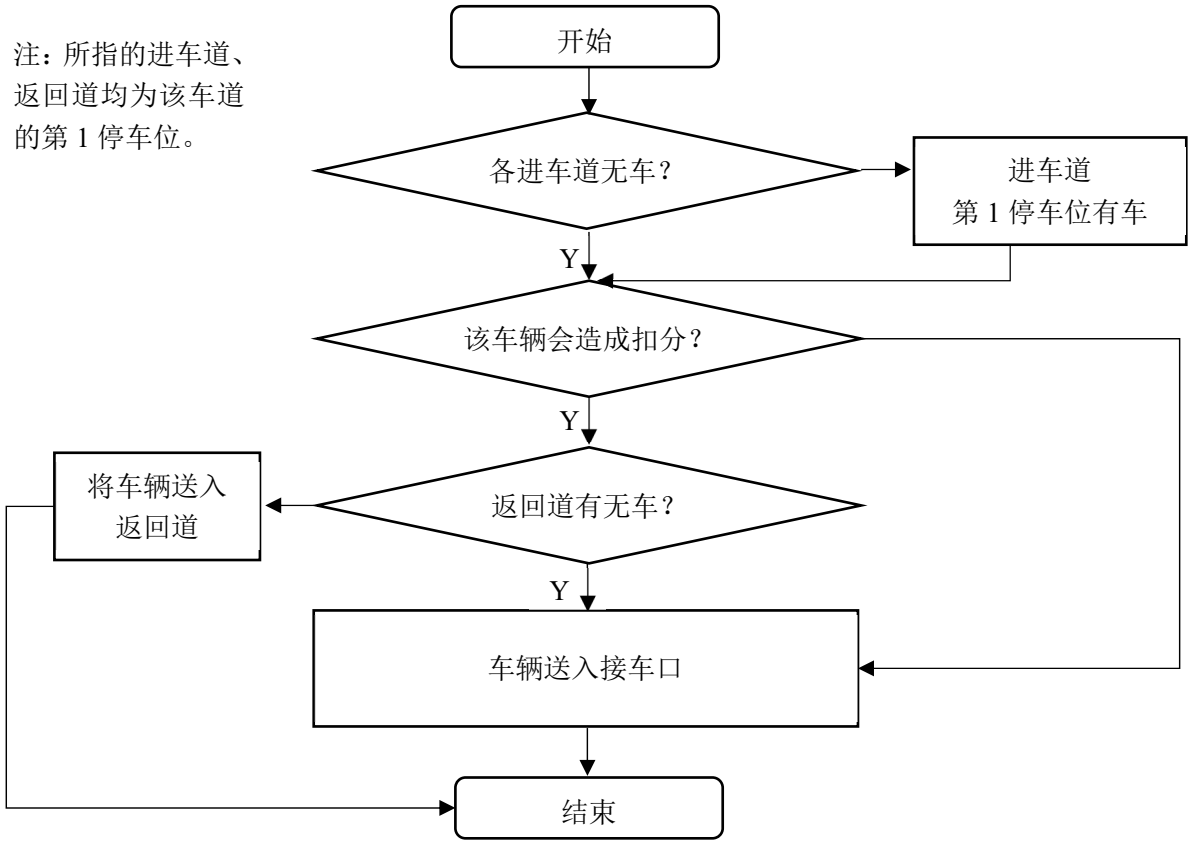


图 3-6　送车机调度方案

3.5 模型求解与分析

3.5.1 仅对送车机使用启发式调控规则的结果

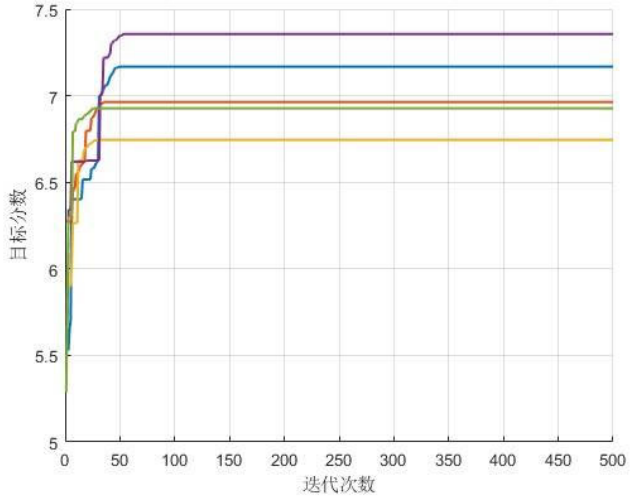
对送车机使用启发式调控规则后，PBS 返回道则将开始使用，运行情况变得复杂，故对接车机的送车进入进车道的选择仍按随机规则，观察，返回道的使用和返回规则对模型优化结果的影响。

在上述情况下，通过模拟得出如表 3.3 所示的结果：

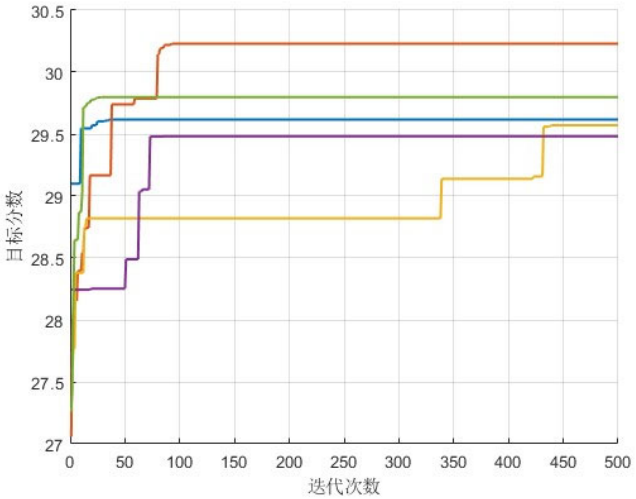
表 3.3　仅对送车机使用启发式调控规则的结果

	目标一分数	目标二分数	目标三分数	目标四分数	总优化目标分数
附件一	-100	64	82	88.93	4.493
附件二	-45	62	85	89.48	26.548

由于缺乏对该结果的多重认识，下面本文对接车机随机送入车道的序列进行遗传迭代取优，如图 3-7 为迭代过程，得到表 3.4 的结果：



(a)附件一数据的迭代过程



(a)附件二数据的迭代过程

图 3-7　使用遗传算法进行优化迭代的过程

表 3.4　遗传算法取优后的结果

	目标一分数	目标二分数	目标三分数	目标四分数	总优化目标分数
附件一	-99	68	87	91.56	7.356
附件二	-39	64	88	90.29	30.229

可见，在接车机随机放置车辆至任意车道的情况下，其分数较低，经过遗传优化后有进步的空间，分数不断上涨。

3.5.2 接车机、送车机皆使用启发式调控规则的结果

由于接车机、送车机皆使用启发式调控规则进行工作，则该模拟过程完全固定，结果也固定。故利用 OBS 优化调度模型对该种情况下的模拟结果的目标分数见表 3.5。

表 3.5　接车机、送车机皆使用启发式调控规则的结果

	目标一分数	目标二分数	目标三分数	目标四分数	总优化目标分数
附件一	-100	64	83	106.57	6.457
附件二	-44	64	91	107.55	30.555

与表 3.3 和表 3.4 比较后，可以发现其结果（接车机、送车机皆使用启发式调控）接近于遗传算法取优后输出的结果，且较大幅度超过仅对送车机使用启发式调控规则的结果，可见通过对接车机、送车机使用启发式调控规则可以起到预期优化效果。

3.5.4 模型的比较

综合以上考虑，选用接车机、送车机皆使用启发式调控规则的结果作为问题一的最终结果。

3.6 问题一的结果

通过 3.5 节的求解，利用 PBS 优化调度模型，对附件一和附件二内的数据得出的优化结果为：

表 3.6　问题一附件 1 的得分结果

附件 1 目标分数
6.457

表 3.7　问题一附件 2 的得分结果

附件 2 目标分数
30.555

其余调度输出结果整理在附件 result11.xlsx 和 result12.xlsx 中展示。

4 问题二的分析与求解

4.1 问题二的分析

去除 PBS 约束说明中的*6)、*7)两条约束后，接车机和送车机可以更加自由的选择接送车辆的序列。对这两条约束进行分析可知，*6)主要约束了接车机的运行进程，使得接车机在返回道第 10 停车位上有车辆时必须放弃下一任务进而将该车辆进行移动，去除*6)约束后接车机的行为将可设定为有利于 PBS 排序的动作；*7)主要约束了送车机的运行进程，使得送车机必须处理一辆最先到达第 1 停车位的车辆，而不是处理最优于 PBS 排序问题的车辆，去除*7)后送车机的行为将被大大解放，可以对在第 1 停车道的任意车辆进行移动，从而得到 PBS 优化调序模型的更加优化的解。

针对以上分析，下文将对送车机和接车机的启发式调度规则进行更新和调整，通过代码实现的方式完成问题的求解和结果的输出。

4.2 接送车机启发式调度规则的更新与实现

4.2.1 接车机启发式调度规则的更新

根据分析，需要在更新的调度规则中体现接车机的能动性，综合 PBS 内各区域当前实况全局地考虑运行方式，使其为 PBS 调序系统的调度做出有利贡献。以下为本文提出的新调度规则：

Step1：计算接车机序列中各车型的数量。

Step2：数量最多的车型，走耗时最少的 4 车道，若四车道满，走最远的 1 或 6 车道；数量次多的车走 3 道；数量最少的车走 5 车道或 2 车道。

Step3：返回道停车位 10 有车时，根据返回道停车位 10 中的车型和出车口的下一辆车的车型，优先选择车型数量少的车运送，若车型相同则优先运送返回道停车位 10 的车辆，遵循 Step2 的车道运送规则。

基于上述描述接车机送车逻辑判断流程如图 4-1 所示。

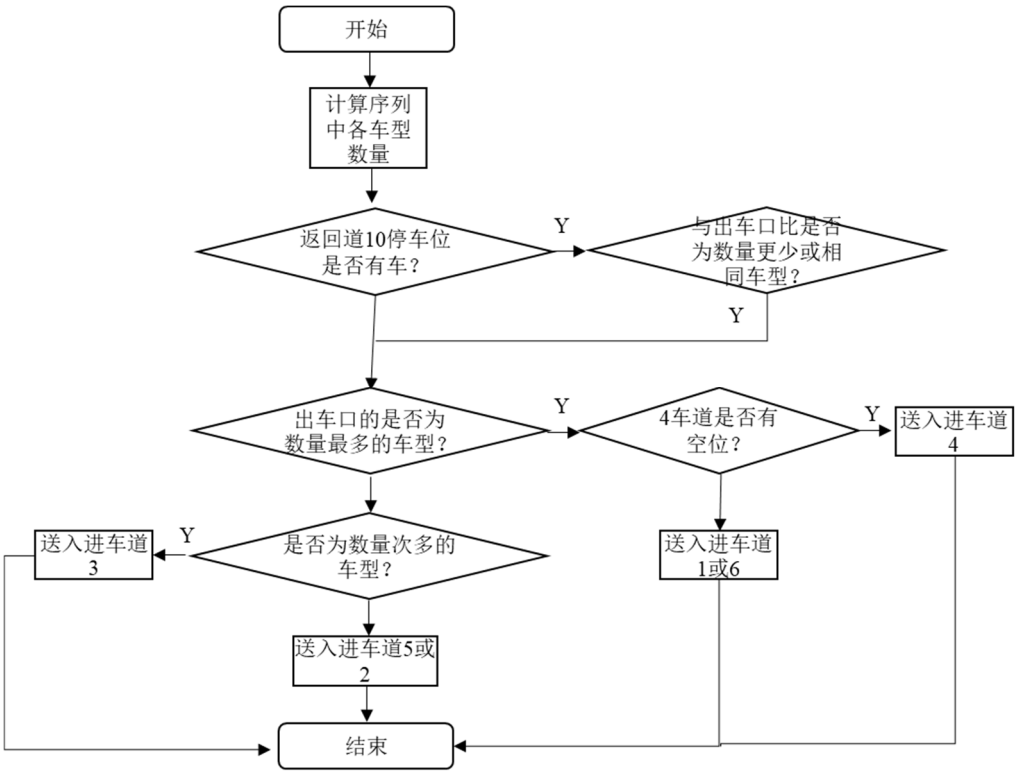


图 4-1　接车机送车逻辑判断流程图

4.2.2 送车机启发式调度规则的更新

根据分析，因去除了*7）约束，则 PBS 赋予了送车机更多的行动空间，若合理地利用好这些行动空间，送车机的行为将会对出车序列的优化起到帮助。目前，对送车机提出的行动准则是使得做决策的时刻尽量地少扣分，即不违反四个目标条件，以下为本文提出的新调度规则：

Step1：送车机优先按照优化目标 1 选择接送顺序，满足优化目标 1 的前提下尽可能满足优化目标 2。

Step2：当到达停车位 1 的车中没有能够满足优化目标 1 的车型时，将车辆送至返回车道，在新到达停车位 1 的车辆中继续筛选车辆，每辆车最多被返回 2 次。有两辆同时在停车位 1 的车，优先选择返回 4 车道其次 3 车道其次 2/5 车道其次 1/6 车道的车。

Step3：被返回两次的车辆和其他车辆同时到达停车位 1 时，优先选择被返回两次的车辆进入总装车间序列。

4.2.3 过程模拟代码修改

在表 3.1 的伪代码中，将上述更新的启发式调度方案进行写入，调整若干参数后完成代码的修改，代码源码见附录 2。

4.3 模型求解与分析

将附件一和附件二的数据带入模型和代码中进行全过程模拟，对数据进行整理和验证后，得出如表 4.1 所示的调控分数结果。

表 4.1　基于更新后的调度规则输出的 PBS 调控结果

	目标一分数	目标二分数	目标三分数	目标四分数	总优化目标分数
附件一	-96	66	85	106.93	9.093
附件二	-40	64	91	107.60	32.160

可见，更新过的调控方案使得 PBS 优化调度模型的结果相比更新前的结果（表 3.5）更加优秀，主要在目标一的方向上做出了调整，减少了混动车型与非混动车型之间排序错误。综上，本文将此答案作为问题二的结果。

4.4 问题二的结果

通过上述分析和求解，通过 PBS 优化调度模型，得出问题二的结果如下表所示：

表 4.2　问题二附件 1 的得分结果

附件 1 目标分数
9.093

表 4.3　问题二附件 2 的得分结果

附件 2 目标分数
32.160

其余调度输出结果整理在附件 result21.xlsx 和 result22.xlsx 中展示。

5 模型的评价

5.1 模型的优点

本文采用了较为创新的研究方法，首先利用遗传算法对简化参数后的车辆序列进行生产模拟，观察生产过程总结经验和规律，再通过得到的经验启发性地提出调度策略，在 PBS 优化建模中取得了理想的效果。本文认为本模型存在以下优点：

优点一：模型算法简单，不涉及上万次上亿次的迭代过程，单次运行模型即可得到优化后的结果，并可掌握 PBS 中各时刻的车辆信息。

优点二：利用遗传算法得到大量生产排序过程中成绩优秀的例子，对该例子进行查看和经验积累，从而得出启发式规则，启发式规则对模型的适用性良好，可有效提高模型优化效率。

优点三：利用遗传算法和启发式规则优化算法互相演进，得出合理的结论，从而不断改进 PBS 优化调度模型，使得模型具有良好的求解效果。

5.2 模型的缺点

第一，本文在优化 PBS 模型时对于启发式优化算法过于依赖，易导致优化结果同时依赖于人机交互的结果，使得最终的结果仅为局部最优解。

第二，缺乏多种优化算法使用后的优化对比，使用的优化算法较为单一，缺乏算法间效率的有力验证。

5.3 模型的改进

考虑到启发式优化算法过于依靠人机交互的问题[4-12]，后期可以采用 BP 神经网络模型对接车机和送车机的实时调控能力进行继续优化，使用当前时刻下的 PBS 内部位置信息作为输入，行动代码进行输出，并对神经网络模型进行不断地训练，使得 PBS 优化调度模型减少人机交互的干扰。

6 参考文献

[1] 喻代权. 汽车混流装配线动态生产调度研究及应用[D]. 重庆理工大学, 2014.
[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]司守奎, 孙兆亮主编, 数学建模算法与应用. 北京:国防工业出版社, 329-333, 2015.
[4]赵萌. 基于深度强化学习的作业车间调度方法研究[D].华中科技大学,2020.
[5]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.
[6]李浩. 基于遗传算法的冲压车间调度算法研究与系统实现[D].华中科技大学,2009.
[7]张文慧.基于遗传算法的生产车间调度方法研究[J].电子测试,2017(12):45-46.
[8]邓宇巍,吕勇哉,陈玉旺.基于混合遗传算法的热轧生产调度优化方法[J].控制工程,2007(S1):67-69+87.
[9]李燚. 面向汽车混流生产的多级流水车间排产与调度问题研究[D].重庆大学,2021.
[10]喻代权. 汽车混流装配线动态生产调度研究及应用[D].重庆理工大学,2014.
[11]李栓子,施国洪.基于启发式规则的物流配送车辆排班优化[J].中国集体经济,2022(03):112-116.
[12]李之晨, 任秀. PBS 输送线在汽车生产企业总装工程中的应用研究[J].科技创新与应用,2017(16):161.

附录 程序代码

附录 1

Fenshu.m

```
function [fen] = fenshu(che,sj1,t,fan_t)
    if(size(sj1,1)<318)
        fen = -100*ones(1,5);
    else
        for i=1:318
            for j=1:318
                if sj1(i,1)==che(j,1)
                    sj1(i,2)=che(j,2);
                    sj1(i,3)=che(j,3);
                    break;
                end
            end
        end
        fen(1)=100;
        fen(2)=100;
        fen(3)=100;
        fen(4)=100;
        %目标 1 分数
        tag2=0;
        for i=1:318
            if sj1(i,2)==2
                if tag2~=2
                    fen(1)=fen(1)-1;
                end
                tag2=0;
            else
                tag2=tag2+1;
            end
        end
        %目标 2 分数
        tag2=[0,0,0];j=1;
        ii=1;%ii 为变化次数记录
        for i=1:318
            if sj1(i,3)==sj1(j,3)
                tag2(1,ii)=tag2(1,ii)+1;
            else
                j=i;
                ii=ii+1;
                tag2(1,ii)=tag2(1,ii)+1;
                if ii>=3
```

```
                    if tag2(1,2)~=tag2(1,1)
                        fen(2)=fen(2)-2;
                    end
                    ii=1;
                    tag2=[1,0,0];
                end
            end
        end
        fen(3)=fen(3)-fan_t;%目标 3 分数
        fen(4)=fen(4)-(0.01*(t-9*size(sj1,1)-72));%目标 4 分数
        fen(5)=0.4*fen(1)+0.3*fen(2)+0.2*fen(3)+0.1*fen(4);%总分
    end
end
```

附录 2

Model.m

```
function [fen che] = model
    [che,txt]=xlsread('附件 1.xlsx');
    for i=2:1:size(txt,1)
        if strcmp(txt(i,3),'燃油')==1 che(i-1,2)=1; end
        if strcmp(txt(i,3),'混动')==1 che(i-1,2)=2; end
        if strcmp(txt(i,4),'两驱')==1 che(i-1,3)=2; end
        if strcmp(txt(i,4),'四驱')==1 che(i-1,3)=4; end
    end
    che(:,5)=0;
    ttt=[1,2,3,4,5,6];
    che(1:6,4)=4;
    %初始变量区
    time1=[18,12,6,0,12,18];
    time2=[24,18,12,6,12,18];
    jieche=0;songche=0;
    tutu=1;
    tag=0;
    shijian=zeros(6,10);
    fan_shijian=zeros(1,10);
    car01=zeros(6,10);
    fan_car=zeros(1,10);
    fan_t=0;
    sj1=0;
    nn=1;
    x=0;k=0;ran=0;
    flag1=0;flag2=0;
    for t=0:6000
```

```
if(fan_shijian(1,10)<0 && tag>300)
    a=1;
end
jieche(t+2)=jieche(t+1)-1;
songche(t+2)=songche(t+1)-1;
shijian=shijian-1;
fan_shijian=fan_shijian-1;
for oo=1:6
    if car01(oo,1)==0
        shijian(oo,1)=shijian(oo,1)+1;
    end
end
%计数
if ((flag2==1) && (songche(t+2)==0))
    fan_car(1,1)=tag;
    fan_shijian(1,1)=9;
    flag2=0;
end
if ((flag2==2) && (songche(t+2)==0)) %记录出车序列
    if(find(sj1(:,1)==tag))
        a=1;
    end
    sj1(nn,1)=tag;
    che(tag,5)=1;
    nn=nn+1;
    x=0;
    flag2=0;
end
%当送车横移机到时候才把车带走
if((flag2==2) && (x~=0)&&(songche(t+2)==(time1(1,x)/2)+1))
    car01(x,1)=0;
    shijian(x,1)=0;
end
%此时将车标记在送车横移机上
if((flag2==2) && (x~=0)&&(songche(t+2)>1)&&(songche(t+2)<=(time1(1,x)/2)+1))
    songcheTable(t+1,1)=tag;
    songcheTable(t+1,2)=time1(che(tag,4));
else
    songcheTable(t+1,:)=0;
end
%送车机
if songche(t+2)<=0 && sum(car01(:,1))~=0
    x=find(shijian(:,1)==min(min(shijian(:,1))));
    if sum(x(:,1)==6) y=6; end
```

```
            if sum(x(:,1)==1) y=1; end
            if sum(x(:,1)==5) y=5; end
            if sum(x(:,1)==2) y=2; end
            if sum(x(:,1)==3) y=3; end
            if sum(x(:,1)==4) y=4; end
            x=y;
            tag=car01(x,1);
            order1=0;order2=0;
            if(size(sj1,1)<=size(che,1))
            if size(sj1,1)>4
            if che(tag,2)==2
                if che(size(sj1,1),2)==1&&che(size(sj1,1)-1,2)==1&&che(size(sj1,1)-
2,2)==2
                    order1=0;
                end
                order1=1;
            else
                if che(size(sj1,1),2)==1&&che(size(sj1,1)-1,2)==1&&che(size(sj1,1)-
2,2)==2
                    order1=1;
                end
                order1=0;
            end
            if che(size(sj1,1),3)~=che(tag,3)
                order2=1;
            else order2=0;
            end
            end
            end
            if(tag<=size(txt,1))
                if((order1&&order2)&&tutu<size(che,1)-10)
                    if(fan_shijian(1,1)<=0 && fan_car(1,1)==0)
                        flag2=1;
                        fan_t=fan_t+1;
                        songche(t+2)=time2(1,x)-1;
                        car01(x,1)=0;
                        shijian(x,1)=0;
                    else
                        flag2=0;
                    end
                end
            end
            if(flag2~=1)
                songche(t+2)=time1(1,x)-1;
```

```
        flag2=2;
        if(x==4)
            car01(x,1)=0;
            shijian(x,1)=0;
            if(find(sj1(:,1)==tag))
                a=1;
            end
            sj1(nn,1)=tag;
            che(tag,5)=1;
            nn=nn+1;
            x=0;
        end
    end
end
%车辆移动
for i=1:6
    for j=2:10
        if shijian(i,j)<=0 && car01(i,j-1)==0 && car01(i,j)~=0
            car01(i,j-1)=car01(i,j);
            car01(i,j)=0;
            if j==2
                shijian(i,j-1)=-1;
            else
                shijian(i,j-1)=9;
            end
        end
    end
end
if tutu<=(size(che,1)+1)
    if (jieche(t+2)<=0)
        ran = rand < 0.5;
        if(ran &&(flag1~=1) && (fan_shijian(1,10)<=0 )&& (fan_car(1,10)>0))
            flag1=1;
            if shijian(6,10)<=0 k=6;end
            if shijian(1,10)<=0 k=1;end
            if shijian(5,10)<=0 k=5;end
            if shijian(2,10)<=0 k=2;end
            if shijian(3,10)<=0 k=3;end
            if shijian(4,10)<=0 k=4;end
            jieche(t+2)=time2(1,k);
            tag2=fan_car(1,10);
            fan_car(1,10)=0;
        end
        if(flag1~=1)
```

```
                    flag1=2;
                    if(tutu<=(size(che,1)) && (shijian(che(tutu,4),10)<=0) &&
(che(tutu,4)~=-1))%设置接车状态
                        jieche(t+2)=time1(1,che(tutu,4));
                        tutu=tutu+1;
                    end
                end
            end
            if (flag1==1)&&(jieche(t+2)<0)&& (fan_shijian(1,10)<=0 )&&
(fan_car(1,10)>0)
                a=1;
            end
            if ((flag1==1)&&(jieche(t+2)<=0))
                shijian(k,10)=9;
                car01(k,10)=tag2;
                flag1=0;
            end
            if ((flag1==2)&&((jieche(t+2)==(time1(1,che(tutu-1,4))/ 2))||(che(tutu-
1,4)==4)) && (shijian(che(tutu-1,4),10)<=0))%卸车
                if(find(car01(che(tutu-1,4),:)==tutu-1))
                    tutu=tutu+1;
                    if(tutu>size(che,1))
                        continue;
                    end
                end
                shijian(che(tutu-1,4),10)=9;
                car01(che(tutu-1,4),10)=tutu-1;
                che(tutu-1,5)=-1;
            end
        end
        %返回车移动
        for j=9:-1:1
            if fan_shijian(1,j)<=0 && fan_car(1,j+1)==0 && fan_car(1,j)~=0
                fan_car(1,j+1)=fan_car(1,j);
                fan_car(1,j)=0;
                if j==9
                    fan_shijian(1,j+1)=-1;
                else
                    fan_shijian(1,j+1)=9;
                end
            end
        end
        for i=1:318
            if che(i,5)==-1
```

```
            m=i;
        end
    end
    if shijian(6,10)<=0 l=6;end
    if shijian(1,10)<=0 l=1;end
    if shijian(5,10)<=0 l=5;end
    if shijian(2,10)<=0 l=2;end
    if shijian(3,10)<=0 l=3;end
    if shijian(4,10)<=0 l=4;end
    che(m+1,4)=l;
    if sum(che(:,5)==1)==size(che,1)   %t 结束条件
        break;
    end
    jieguo1(:,:,t+1) = shijian(:,:);
    jieguo2(:,:,t+1) = car01(:,:);
    fan_car1(t+1) =fan_car(1,10);
    fan_car10(t+1) =fan_car(1,10);
    flag_t1(t+1) = flag1;
    flag_t2(t+1) = flag2;
end
fen = fenshu(che,sj1,t,fan_t);
end
```

附录 3

Main.m

```
clear all
close all
clc
tic
Data.Function = 3;
% Problem size
switch Data.Function
    case 1
        Sett.Type    = 'min';
        Sett.LengthX = 1;
    case 2
        Sett.Type    = 'min';
        Sett.LengthX = 2;
    case 3
        Sett.Type    = 'max';
        Sett.LengthX = 318;
    case 4
```

```
        Sett.Type    = 'max';
        Sett.LengthX = 2;
end
% 决策变量界限
XLim(1,:) = 1*ones(1,Sett.LengthX);
XLim(2,:) = 7*ones(1,Sett.LengthX);
% 设置
Sett.NumPop    = 5;%群体数
Sett.NumChr    = 10;%染色体数
Sett.NumIter   = 500;%迭代数
Sett.FlagPlots = true;
Sett.Fprintf = false;
%% 运行 GA
[Results.Xpbest,Results.ObjFunpbest,Results.ObjFuncbestPlot,Results.Scorepbest,Data]
= Optimization_GA_v01(XLim, Sett, Data);
toc
```

附录 4

GA.m

```
function [Xpbest, ObjFunpbest,ObjFuncbestPlot,Scorepbest, Data] =
Optimization_GA_v01(XLim, Sett, Data)
%初始化
X      = zeros(Sett.NumChr,Sett.LengthX);%每一行为染色体，
Xcbest = zeros(1,Sett.LengthX);%个体最优，对比每个染色体
Xpbest = zeros(1,Sett.LengthX);%种群最优，对比每个个体
ObjFun = zeros(Sett.NumChr,1);
ObjFuncbestPlot = zeros(Sett.NumPop,Sett.NumIter);
switch Sett.Type
    case 'min'
        ObjFunpbest     = +inf;
        ObjFunpbestPlot = +inf*ones(1,Sett.NumIter);
    case 'max'
        ObjFunpbest     = -inf;
        ObjFunpbestPlot = -inf*ones(1,Sett.NumIter);
end
for IndexPop = 1:1:Sett.NumPop
    fprintf('- IndexPop:    %3i out of %3i\n',IndexPop,Sett.NumPop);
    % X initialization 初始化一个个体
    for IndexChr = 1:1:Sett.NumChr
        for IndexX = 1:1:Sett.LengthX
            X(IndexChr,IndexX) = floor(XLim(1,IndexX) + (XLim(2,IndexX) -
XLim(1,IndexX))*rand(1));
```

```
    end
end
% Iteration 进行迭代
for IndexIter = 1:1:Sett.NumIter
    fprintf('-- IndexIter:  %3i out of %3i\n',IndexIter,Sett.NumIter);
    % ObjFun calculation
    for IndexChr = 1:1:Sett.NumChr
        fprintf('--- IndexChr:  %3i out of %3i\n',IndexChr,Sett.NumChr);
        [Score(IndexChr,:) ObjFun(IndexChr)] = ObjFun_fun(X(IndexChr,:),Data);
    end
    % 选择算子
    % Natural selection
    switch Sett.Type
        case 'min'
            [~,Indexc]  = sort(ObjFun,'ascend');
        case 'max'
            [~,Indexc]  = sort(ObjFun,'descend');
    end
    % 随机选择一个作为个体最优
    Scorecbest = Score(Indexc(1),:);
    ObjFuncbest = ObjFun(Indexc(1));%记录个体最优决策值
    Xcbest      = X(Indexc(1),:);%记录个体最优决策变量
    % 输出个体最优决策值与最优决策变量
    % Xcbest and ObjFuncbest output
    if(Sett.Fprintf==true)
        for IndexS = 1:4
            fprintf('-- Scorecbest(%d): %-+1.10e,',IndexS,Scorecbest(IndexS));
        end
        fprintf('-- Scorecbest(5): %-+1.10e,',Scorecbest(5));
        fprintf('-- ObjFuncbest: %-+1.10e,',ObjFuncbest);
        for IndexX = 1:1:Sett.LengthX
            if(IndexX < Sett.LengthX)
                fprintf(' Xcbest(%i) = %-+1.10d,',IndexX,Xcbest(IndexX));
            else
                fprintf(' Xcbest(%i) = %-+1.10d\n',IndexX,Xcbest(IndexX));
            end
        end
    end
    % 均匀杂交
    % Uniform crossover
    for IndexChr = 1:1:Sett.NumChr
        for IndexX = 1:1:Sett.LengthX
            Coin = round(rand(1));
            IndexParent = randi(2);
```

```
                if(IndexChr ~= Indexc(1) && IndexChr ~= Indexc(2) && Coin == true)
                    X(IndexChr,IndexX) = X(Indexc(IndexParent),IndexX);
                end
            end
        end
        % 变异
        % Mutation
        for IndexX = 1:1:Sett.LengthX
            Coin = round(rand(1));
            if(Coin == true)
                X(Indexc(end),IndexX) = floor(XLim(1,IndexX) + (XLim(2,IndexX) -
XLim(1,IndexX))*rand(1));
            end
        end
        % Plot
        ObjFuncbestPlot(IndexPop,IndexIter) = ObjFuncbest;
        switch Sett.Type
            case 'min'
                if(ObjFuncbest < ObjFunpbestPlot(IndexIter))
                    ObjFunpbestPlot(IndexIter) = ObjFuncbest;
                end
            case 'max'
                if(ObjFuncbest > ObjFunpbestPlot(IndexIter))
                    ObjFunpbestPlot(IndexIter) = ObjFuncbest;
                end
        end
    end
    % Xpbest and ObjFunpbest determination
    switch Sett.Type
        case 'min'
            if(ObjFuncbest < ObjFunpbest)
                Xpbest      = Xcbest;
                ObjFunpbest = ObjFuncbest;
                Scorepbest = Scorecbest;
            end
        case 'max'
            if(ObjFuncbest > ObjFunpbest)
                Xpbest      = Xcbest;
                ObjFunpbest = ObjFuncbest;
                Scorepbest = Scorecbest;
            end
    end
    % Xpbest and ObjFunpbest output
    for IndexS = 1:4
```

```
        fprintf('-- Scorepbest(%d): %-+1.10e,',IndexS,Scorepbest(IndexS));
    end
    fprintf('-- Scorepbest(5): %-+1.10e,',Scorepbest(5));
    fprintf('-  ObjFunpbest: %-+1.10e,',ObjFunpbest);
    for IndexX = 1:1:Sett.LengthX
        if(IndexX < Sett.LengthX)
            fprintf(' Xpbest(%i) = %-+1.10d,',IndexX,Xpbest(IndexX));
        else
            fprintf(' Xpbest(%i) = %-+1.10d\n',IndexX,Xpbest(IndexX));
        end
    end
    toc
end
%% Plot
if(Sett.FlagPlots == true)
    figure;
    semilogy(ObjFunpbestPlot,'LineWidth',10);
    for IndexPop = 1:1:Sett.NumPop
        hold on;
        semilogy(Results.ObjFuncbestPlot(IndexPop,:),'LineWidth',1.5);
        title('Objective function optimization');
        xlabel('NumIter');
        ylabel('ObjFun_{c,best}(Pop)');
        grid on;
    end
end
end
```

附录 5

ObjFunction.m

```
function [fen ObjFun] = ObjFun_fun(X, Data)
switch Data.Function
    case 1
        ObjFun = +(X-27)^2+25;
    case 2
        ObjFun = +(X(1)-50)^2+(X(2)-25)^2+100;
    case 3
%         fen = model1(X');
        fen = model2(X');
        ObjFun = fen(5);
    case 4
        ObjFun = -(X(1)-50)^2-(X(2)-25)^2-100;
```

```
end
end
```