实验内容九：炼钢-连铸生产重调度

徐子豪,韩娇娇,姜良泽

2020 年 6 月 20 日

1 炼钢-连铸生产重调度模型

1.1 符号说明

建模过程中对使用符号的定义如下:

i:浇次编号，$i \in (1,2,3)$;

j:炉次编号，$j \in (1,2,3,4,5,6)$;

k:设备类型编号，$k \in (1,2,3)$;

k_{max}:连铸机设备的编号，$k_{max} = max(k) = 3$;

n:同类设备编号，$n \in (1,2,3)$;

$S_{i,j,k}$:第i浇次的第j炉次在第k类设备上的操作开始的时间;

$E_{i,j,k}$:第i浇次的第j炉次在第k类设备上的操作结束的时间;

$U_{i,j,k,n}$:第i浇次的第j炉次在第k类设备上的操作是否在第n台设备上进行，是则$U_{i,j,k,n} = 1$，否则$U_{i,j,k,n} = 0$;

$TT_{k1,n1,k2,n2}$:第$k1$类第$n1$个设备到第$k2$类第$n2$个设备的运输时间;

$MinOT_k$:k类设备上操作的最小处理时间;

$MaxOT_k$:k类设备上操作的最大处理时间;

M:一个足够大的正整数。

1.2 决策变量

需要决策的变量包括：

$$S_{i,j,k} \tag{1}$$

$$E_{i,j,k} \tag{2}$$

$$U_{i,j,k,n} \tag{3}$$

1.3 优化目标

优化目标应该为：所有操作之间除去运输，等待时间最小：

$$MIN \quad \sum_{i=1}^{3}\sum_{j=1}^{6}\sum_{n1=1}^{3}\sum_{n2=1}^{3}\sum_{k=1}^{2}(S_{i,j,k+1}-E_{i,j,k}-TT_{k,n1,k+1,n2})\times U_{i,j,k,n}\times U_{i,j,k+1,n2} \tag{4}$$

但直接将其作为目标函数，会导致无法求解的非线性，将设备分配项的乘法转换为相加后减一再relu，又会使问题非凸。因此决定将等待时间放入约束当中，目标函数处可以选择所有步骤结束时间到下一步骤开始时间的差作为高优先级，最晚的一个操作的结束时间作为低优先级：

$$MIN \quad 1000\sum_{i=1}^{3}\sum_{j=1}^{6}\sum_{k=1}^{2}(S_{i,j,k+1}-E_{i,j,k})+\max_{i\in[1,2,3]}E_{i,6,3} \tag{5}$$

1.4 约束条件

1,2.同一炉次的不同操作受到步骤顺序和运输时间的约束，且不运输运输时间之外的等待时间：

$$s_{i,j,k+1}-E_{i,j,k}-TT_{k,n1,k+1,n2}-M\times(1-U_{i,j,k,n1})-M\times(1-U_{i,j,k+1,n2})\le 0 \tag{6}$$

$$s_{i,j,k+1}-E_{i,j,k}-TT_{k,n1,k+1,n2}+M\times(1-U_{i,j,k,n1})+M\times(1-U_{i,j,k+1,n2})\ge 0 \qquad \forall i,j,k,n1,n2 \tag{7}$$

3.一个炉次操作不能不安排设备，也不能被安排在多个设备上：

$$\sum_{n=1}^{3}U_{i,j,k,n}=1 \qquad \forall i,j,k \tag{8}$$

4.同设备上的不同操作时间范围不能重叠，考虑到建模主要为了供CPLEX求解使用，判断两个操作前后顺序的01变量可以省略，用逻辑约束描述：

$$S_{i1,j1,k}-E_{i2,j2,k}+M\times(1-U_{i1,j1,k,n})+M\times(1-U_{i2,j2,k,n}) \tag{9}$$

$$OR \quad S_{i2,j2,k}-E_{i1,j1,k}+M\times(1-U_{i1,j1,k,n})+M\times(1-U_{i2,j2,k,n}) \qquad \forall i1,i2,j1,j2,k,n \tag{10}$$

5.每个操作的耗时范围约束：

$$E_{i,j,k}-S_{i,j,k}\ge MinOT_k \tag{11}$$

$$E_{i,j,k}-S_{i,j,k}\le MaxOT_k \qquad \forall i,j,k \tag{12}$$

6.连铸机需要和浇次相同：

$$U_{i,j,k_{max},i}=1 \qquad \forall i,j,k \tag{13}$$

7.连铸机要保证连续工作：

$$S_{i,j+1,k_{max}}-E_{i,j,k_{max}}=0 \qquad \forall i,j,k \tag{14}$$

因为同浇次不同炉次整体的互相替换，不影响目标函数，也不影响其他约束，因此按数字顺序排列连铸机上的操作顺序不影响最优性。

8.已经开始的时间节点和相应的设备分配不能变动。

2 启发式调度算法

2.1 启发式算法设计

对延迟炉次操作所在的炉次的所有设备涉及的未开工的炉次操作延迟对应时间；对于已开工的炉次，从O313开始向后延长结束时间，直到连铸机能连续工作；对于未开工的炉次，先按照连铸机上的开工时间，从晚到早选择炉次，对单个炉次按逆向顺序以如下优先级方式安排炉次操作：

I. 炉次操作不能与已经开始的炉次存在冲突；

II. 炉次操作和已经安排的炉次操作的冲突时间最小；

III. 炉次操作尽量最晚开工。

编写Python程序实现算法并得到时间表和甘特图。

2.2 启发式算法调度仿真结果

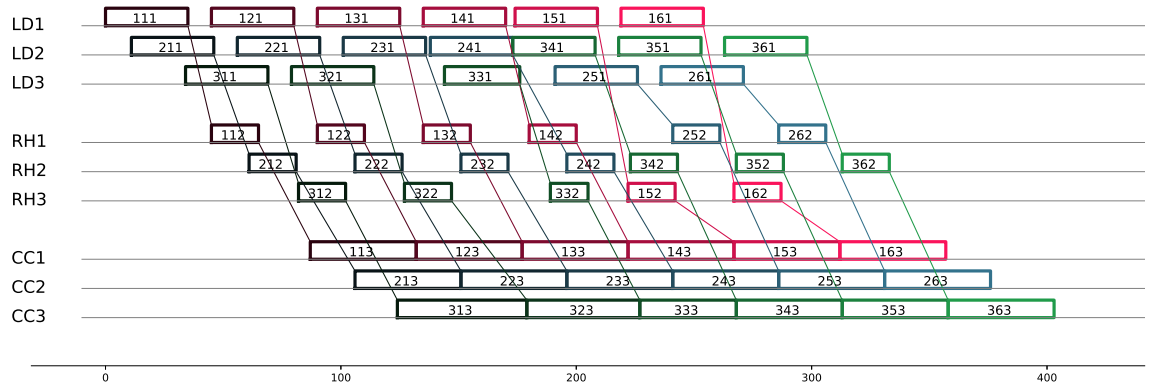

图 1: 启发式算法调度仿真结果甘特图

表 1: 启发式算法调度仿真结果炉次操作开工/完工时间表

浇次	炉次	转炉			精炼			连铸		
		设备	开始	结束	设备	开始	结束	设备	开始	结束
1	1	LD1	0	35	RH1	45	65	CC1	87	132
	2	LD1	45	80	RH1	90	110	CC1	132	177
	3	LD1	90	125	RH1	135	155	CC1	177	222
	4	LD1	135	170	RH1	180	200	CC1	222	267
	5	LD1	174	209	RH3	222	242	CC1	267	312
	6	LD1	219	254	RH3	267	287	CC1	312	357
2	1	LD2	11	46	RH2	61	81	CC2	106	151
	2	LD2	56	91	RH2	106	126	CC2	151	196
	3	LD2	101	136	RH2	151	171	CC2	196	241

表 1 续表

浇次	炉次	转炉			精炼			连铸		
		设备	开始	结束	设备	开始	结束	设备	开始	结束
	4	LD2	138	173	RH2	196	216	CC2	241	286
	5	LD3	191	226	RH1	241	261	CC2	286	331
	6	LD3	236	271	RH1	286	306	CC2	331	376
3	1	LD3	34	69	RH3	82	102	CC3	124	179
	2	LD3	79	114	RH3	127	147	CC3	179	227
	3	LD3	144	176	RH3	189	205	CC3	227	268
	4	LD2	173	208	RH2	223	243	CC3	268	313
	5	LD2	218	253	RH2	268	288	CC3	313	358
	6	LD2	263	298	RH2	313	333	CC3	358	403

3 CPLEX求解

3.1 OPT模型及数据输入

最优化指标部分，使用

```
minimize
  sum(i in Pouring) sum(j in Furnace)
  (StartTime[i][j][2]-EndTime[i][j][1]+StartTime[i][j][3]-EndTime[i][j][2]);
```

计算所有等待时间的总和

```
forall(i in Pouring)
    forall(j in Furnace)
      Operating_order_in_one_furnace:
      {
        StartTime[i][j][3] - EndTime[i][j][2] >= TransportTime[i][j][2];
        StartTime[i][j][2] - EndTime[i][j][1] >= TransportTime[i][j][1];
        }
  forall(i in Pouring)
    forall(j in Furnace)
      forall(k in DeviceClass)
        Operating_time_limit:
        {
          EndTime[i][j][k] - StartTime[i][j][k] >= MinOperatingTime[k];
          EndTime[i][j][k] - StartTime[i][j][k] <= MaxOperatingTime[k];
          }
```

分别代表同一炉次需要按操作顺序加工，和操作时间的最大最小值。由于采用了启发算法得到的设备分配和顺序，所以逐条直接输入同设备的使用顺序限制例如：

```
        StartTime[1][2][1] - EndTime[1][1][1] >= 0;StartTime[1][3][1] - EndTime[1][2][1] >= 0;
    StartTime[1][4][1] - EndTime[1][3][1] >= 0;StartTime[1][5][1] - EndTime[1][4][1] >= 0;
    StartTime[1][6][1] - EndTime[1][5][1] >= 0;
```

已经进行到的操作开始（完成）时间，也一并作为约束逐条输入，例如：

```
    StartTime[1][1][1] == 0; EndTime[1][1][1] == 35;
  StartTime[1][2][1] == 45; EndTime[1][2][1] == 80;
  StartTime[1][3][1] == 90; EndTime[1][3][1] == 125;
```

```
StartTime[1][4][1] == 135;
```

因为设备分配固定，所以设备间运输时间直接由炉次操作索引给出，

3.2 CPLEX仿真结果

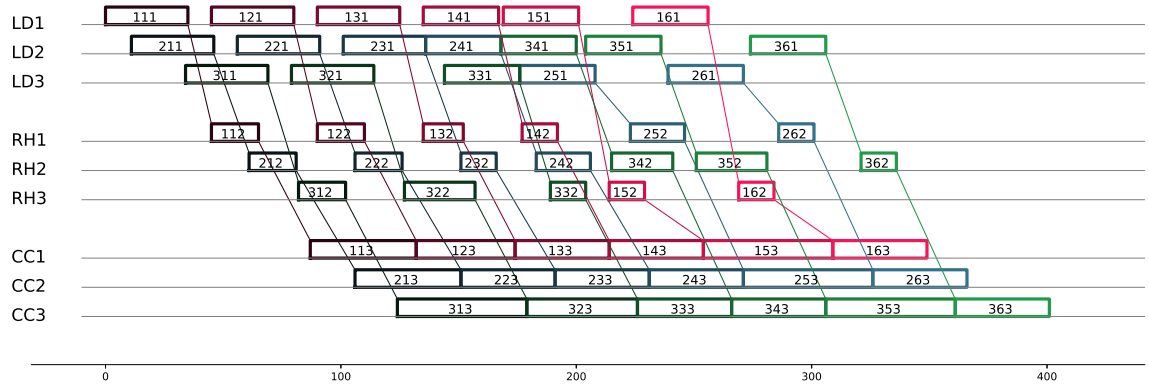

图 2: CPLEX调度仿真结果甘特图

表 2: CPLEX调度仿真结果炉次操作开工/完工时间表

浇次	炉次	转炉			精炼			连铸		
		设备	开始	结束	设备	开始	结束	设备	开始	结束
1	1	LD1	0	35	RH1	45	65	CC1	87	132
	2	LD1	45	80	RH1	90	110	CC1	132	174
	3	LD1	90	125	RH1	135	152	CC1	174	214
	4	LD1	135	167	RH1	177	192	CC1	214	254
	5	LD1	169	201	RH3	214	229	CC1	254	309
	6	LD1	224	256	RH3	269	284	CC1	309	349
2	1	LD2	11	46	RH2	61	81	CC2	106	151
	2	LD2	56	91	RH2	106	126	CC2	151	191
	3	LD2	101	136	RH2	151	166	CC2	191	231
	4	LD2	136	168	RH2	183	206	CC2	231	271
	5	LD3	176	208	RH1	223	246	CC2	271	326
	6	LD3	239	271	RH1	286	301	CC2	326	366
3	1	LD3	34	69	RH3	82	102	CC3	124	179
	2	LD3	79	114	RH3	127	157	CC3	179	226
	3	LD3	144	176	RH3	189	204	CC3	226	266
	4	LD2	168	200	RH2	215	241	CC3	266	306
	5	LD2	204	236	RH2	251	281	CC3	306	361
	6	LD2	274	306	RH2	321	336	CC3	361	401

4 CPLEX直接求解

因为不确定是否需要优化使用了启发算法得到的设备分配和顺序的模型，所以也尝试直接用CPLEX求解问题的方法。

4.1 OPT模型及数据输入

此时OPL模型的约束部分和第一节建模部分的模型几乎一致。但在求解需要耗费一小时以上的时间。经过实验发现可以在同一炉次的操作顺序时间约束一项中增加约束：

```
operation_order_in_one_furnace:
forall(i in Pouring)
    forall(j in Furnace)
        forall(k in 1..2)
            forall(n1 in DviceNumber)
                forall(n2 in DviceNumber)
        {
        StartTime[i][j][k+1] - EndTime[i][j][k]
        - TransportTime[k][n1][k+1][n2]
        + 500*(1-UseDevice[i][j][k][n1])
        + 500*(1-UseDevice[i][j][k+1][n2])>= 0;
        StartTime[i][j][k+1] - EndTime[i][j][k] >= 10;
          }
```

即下一步骤的开始时间和上一步骤的结束时间至少间隔10。因为最小运输时间大于10，所以满足原问题约束的解集是该约束解集的子集，优化问题和原优化问题等价。但增加的约束可以明显加快CPLEX求解的速度，从一小时以上减小到数秒。

4.2 CPLEX直接求解仿真结果

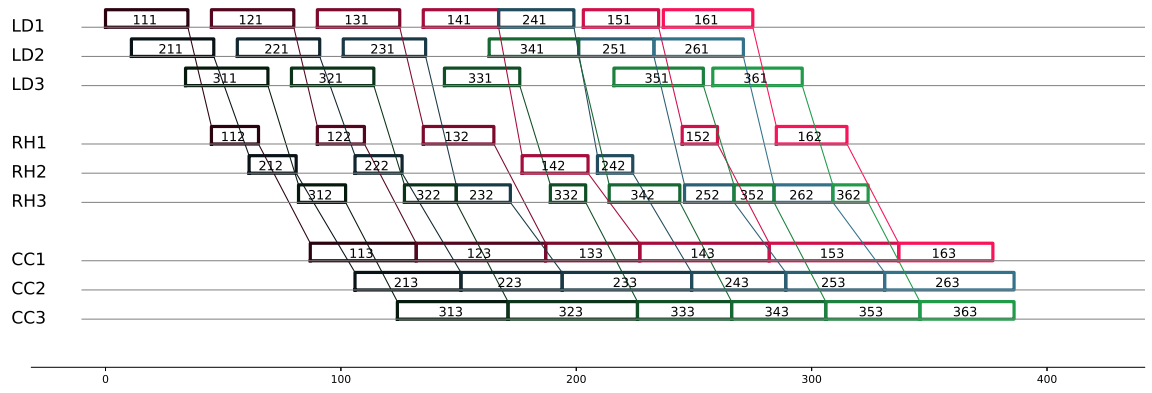

图 3: CPLEX直接求解调度仿真结果甘特图

表 3: CPLEX直接求解调度仿真结果炉次操作开工/完工时间表

浇次	炉次	转炉			精炼			连铸		
		设备	开始	结束	设备	开始	结束	设备	开始	结束
1	1	LD1	0	35	RH1	45	65	CC1	87	132
	2	LD1	45	80	RH1	90	110	CC1	132	187
	3	LD1	90	125	RH1	135	165	CC1	187	227
	4	LD1	135	167	RH2	177	205	CC1	227	282
	5	LD1	203	235	RH1	245	260	CC1	282	337
	6	LD1	237	275	RH1	285	315	CC1	337	377
2	1	LD2	11	46	RH2	61	81	CC2	106	151
	2	LD2	56	91	RH2	106	126	CC2	151	194
	3	LD2	101	136	RH3	149	172	CC2	194	249
	4	LD1	167	199	RH2	209	224	CC2	249	289
	5	LD2	201	233	RH3	246	267	CC2	289	331
	6	LD2	233	271	RH3	284	309	CC2	331	386
3	1	LD3	34	69	RH3	82	102	CC3	124	171
	2	LD3	79	114	RH3	127	149	CC3	171	226
	3	LD3	144	176	RH3	189	204	CC3	226	266
	4	LD2	163	201	RH3	214	244	CC3	266	306
	5	LD3	216	254	RH3	267	284	CC3	306	346
	6	LD3	258	296	RH3	309	324	CC3	346	386

5 方法对比

因为使用CPLEX求解时，使用了启发算法得到的设备分配和顺序，所以两者结果下相差不大，最后一个浇次的最后一个操作，CPLEX结果仅仅好了2s。但启发式算法求解重调度存在一些问题：

I. 炉次操作实践有弹性，导致需要判断是否时间冲突的情况激增，程序冗长；

II. 由于已经结束的炉次操作的限制，且未开始炉次调度顺序为按时间倒序，所以存在循序靠前的操作安排的过于贪心或者随机导致顺序靠后的操作没有可行的解，并且无法判断是该优化优先级下真的无解，还是随机选择导致有几率无解；

III. 限制条件增加，反而比从零开始的调度问题更加复杂。