智能加工系统RGV小车调度

2019 年 8 月 24 日

摘要

问题为一固定数控机床智能加工系统，已有由8台计算机数控机床（Computer Number Controller，CNC）、1辆轨道式自动引导车（Rail Guide Vehicle，RGV）、1条RGV直线轨道、1条上料传送带、1条下料传送带等附属设备组成。问题要求求解RGV上料小车的最优化的调度方案，使得系统的生产效率更高。

本文采用放松状态转移方程缩小解空间，求解GRV调度问题的近似解，化解为可以运用动态规划思想求解，决策影响环境，环境再影响决策依次循环，直到最终达到系统总体运行时间约束。并最终通过Matlab建立对应模型，求解放松后模型，实现连续的决策和环境反馈。

并且通过程序的设置得到了相应，一道、两道工序，故障等情况下的决策的解，并实现了对于放松后模型的优缺点进行了分析。

关键词: 调度　　启发式算法　　RGV

1　问题重述

已有特定的调度调控智能加工系统，由8台计算机数控机床（Computer Number Controller，CNC）、1辆轨道式自动引导车（Rail Guide Vehicle，RGV）、1条RGV直线轨道、1 条上料传送带、1 条下料传送带等附属设备组成。

针对该智能加工系统进行动态调度设计。根据加工系统已有的相关主体（RGV小车、CNC加工台、传送带等）调度的相关参数数据，设计合适的调度算法，使得该系统在固定时间（8h）中，加工尽可能高效。

其中调度问题主要分为：单道工序、两道工序；无故障、期望故障；同时，加工工件同样存在不同种类，不同种类则有不同相关主体的调度参数。

通过设计所得算法，求出在固定时间段内的调度的步骤情况，需要导出每一个工件的上料和下料时间，并且做出结果的效率的分析。

2　模型假设

1. 一道工序的物料加工作业情况，每台CNC安装同样的刀具，物料可以在任一台CNC上加工完成；
2. 两道工序的物料加工作业情况，每个物料的第一和第二道工序分别由两台不同的CNC依次加工完成；
3. 两道工序情况下，特定CNC加工工序在整个系统运行过程中不改变。
4. 传送带循环上/下料时间短，可以忽略,认为传送带上下料的过程可以不考虑。

3 符号说明

- $STEP_i$ ..... 所有决策阶段的集合
- $step_i$ ..... 第i个决策阶段
- Φ_i ..... 第i个决策的决策范围
- ϕ_i ..... 决策范围状态转移函数
- Ψ_i ..... 第i个决策的决策区间
- ψ_i ..... 决策区间状态转移函数
- $R(Route)$ ..... 决策的集合
- $FR(FeasibleRoute)$ ..... 可行决策的集合
- r ..... 决策的集合中的元素
- $e(r)$ ..... 效率函数
- t ..... 时间维变量
- $\Delta t_{\alpha\beta}$ ..... α到β的时间间隔
- $T(Step_i/R_i)$ ..... 时间函数，某一决策阶段控制/具体决策的时间
- T_{end} ..... 系统可调度时间

4 模型的建立与求解

4.1 系统分析与模型建立

对于该智能系统在8h内的智能调度，本文采用阶段性决策的方法求得近似的调度解，下面说明求解所用启发式算法的结构框架。

下面分析本系统，模型具有一下特殊性质：非时变系统、难分解、离散、分段、具有后效性、马尔可夫链不定长（max=8）。

4.1.1 模型建立

由于目标为求得系统在限定时间里的最大效率产出，故可以列写下面模型数学表达式。其中约束条件分别为：时间约束、决策范围约束、决策时间区间约束；

$$\max \sum_{i \in STEP} e(r_i) \tag{1}$$

$$s.t. \sum_{i} T(Step_i) \leq T_{end} \tag{2}$$

$$T(FR_i) \leq \Psi_i \tag{3}$$

$$FR_i \in \Phi_i \tag{4}$$

$$r_i \in FR_i \tag{5}$$

4.1.2 模型分析

由于这是一个动态的决策模型，即在$Step_i$中ϕ_i 决策范围和ψ_i 决策区间都会因为之间的$Step_j(j<i)$的决策而变化。其中有如下状态转移关系：

$$\Delta t_{i-1} = T(step_{i-1}) \tag{6}$$

$$\Phi_i = \phi(r_{i-1}, \Phi_{i-1}, \Delta t_{i-1}) \tag{7}$$

$$\Psi_i = \psi(r_{i-1}, \Delta t_{i-1}, \{\Psi_j | (j<i)\}) \tag{8}$$

由上式可知Φ_i为可分解且无后效性，但是其在运算时有abs的运算(见后面说明)，从而是间断跳跃的，而且其中有时间维度变化但是是可分解的，但是是非时变的；

且由上式可知Ψ_i是有后效性，且滞后作用长度不定，但是其在运算时有abs的运算(见后面说明)，其中有时间维度变化但是是可分解的，非时变的；

从而直接最优求解上式是复杂的，从而通过放松转状态转移约束求解近似解。

4.1.3 近似求解分析

由前面有Ψ_i是有后效性，且滞后作用长度不定，这使得其在运算时不可分解。即**不满足下式**，从而无法使用逐步骤决策的方法求得其最优解。

$$\Psi_i = \psi(r_{i-1}, \Delta t_{i-1}, \{\Psi_j | (j=i-1)\}) \tag{9}$$

但是为了使得模型便于求解，我们仍然可以用逐步决策的方法求得一个可行的近优解。我们Ψ_i通过以1s为单位逐步计算，当需要决策时Ψ_i取出当时时刻的Ψ_i用于该阶段决策。

4.1.4 近似模型

为了求解近优解，我们将模型转变为一下形式。值得说明的是因为$STEP$的大小会随决策变化，所以存在$\sum_{i\in STEP}\max e(r_i) \le \max\sum_{i\in STEP} e(r_i)$，从而我们取每一次决策最优值的和近似整体最优值。目标函数变化如下，约束条件同原来模型。

$$\sum_{i\in STEP}\max e(r_i) \tag{10}$$

$$s.t.... \tag{11}$$

从而可以求解每一步最优的策略，其目标和约束如下.

$$\max e(r_i) \tag{12}$$

$$T(FR_i) \le \Psi_i \tag{13}$$

$$FR_i \in \Phi_i \tag{14}$$

$$r_i \in FR_i \tag{15}$$

4.1.5 近似模型的编程实现

上述求解采用Matlab编程实现，其中主要分为两个模块。

- 模拟系统环境
- 决策部分

模拟系统环境部分，采用时序进行，并以1s为基本间隔循环计算，根据$step_{i-1}$的决策，不断循环更新系统各主体状态（环境）。并在需要决策的$step_i$ 时调用决策部分。

决策部分，根据决策当时刻的系统各主体状态（环境），给出在满足各限制条件的最优的决策。

4.1.6 决策部分算法

在决策部分，因为题目中所面对的决策主体较少（最多8个），且约束条件比较多，从而可以采用带剪枝的遍历方法，找出在决策时间区间Ψ_i内的FR_i中最优调度决策。

因为存在$T(FR_i) \leq \Psi_i$的时间约束，从而调度的可能性大大减小；同时在两道工序的情况下，部分组合是不成立的（如，连续下料两个工序1CNC），从而可行解空间FR_i进一步缩小.

5 Case分析与求解

5.0.7 Case1

Case1 要求，在智能加工系统上加工一道工序的产品，且不考虑故障的发生，求解合适的RGV的调度方案使得系统工作效率尽量高。

通过设置系统的初始条件，可以按照前述思路，求解出合适的近优的调度方法。

设置初始条件：

- 所有CNC上无工件
- 所有CNC都处理工序1
- 目前处理第1(2/3)组数据
- 目前故障率为0

5.1 Case2

Case2 要求，在智能加工系统上加工需要两道道工序的产品，且不考虑故障的发生，求解合适的RGV的调度方案使得系统工作效率尽量高。

通过设置系统的初始条件，可以按照前述思路，求解出合适的近优的调度方法。

设置初始条件：

- 所有CNC上无工件
- 安排各CNC，负责的工序
- 目前处理第1(2/3)组数据
- 目前故障率为0

（其中工序的安排，通过实践得知为按照两道工序的时间比列进行数量的选择，并且不同工序机器面对排布。）

5.2 Case3

Case2 要求，在智能加工系统上加工需要两道道工序的产品，且不考虑故障的发生，求解合适的RGV的调度方案使得系统工作效率尽量高。

通过设置系统的初始条件，可以按照前述思路，求解出合适的近优的调度方法。

设置初始条件：

- 所有CNC上无工件
- 安排各CNC，负责的工序
- 目前处理第1(2/3)组数据
- 目前故障率为1%
- 故障停机时间为10 20间随机数

（其中工序的安排，通过实践得知为按照两道工序的时间比列进行数量的选择，并且不同工序机器面对排布。）

5.2.1 模型结果

CASE	Group1	Group2	Group3
Case1	382-0	359-0	392-0
Case2	252-0	210-0	240-0
Case3-1	370-4	342-4	386-3
Case3-2	231-6	205-3	236-2

结果：8小时内工件数目-故障的数目

6 模型的分析

6.0.2 模型缺点

设置初始条件：

- 求解的为近似解模型，偏离最优解
- 没有考虑传送带的最佳调度
- 停机时间为均匀分布，没有分析其具体分布
- 难以测定距离最优解的距离

6.0.3 模型优点

- 模型可以比较快速的求解
- 模型可以仿真出每一个时刻的系统状态
- 模型可以模拟测试出合适的参数

6.0.4 模型改进

- 可以考虑后效性的影响，可以采用两部最优解的一致性来尝试解决这个问题

6.0.5 Matlab程序构成

Matalb程序由一下几部分组成：

- caculate-time.m (用时计算)
- plan-route.m (路线规划)
- Decsion.m (决策部分)
- Operation.m (模拟系统环境)

其中调用关系如下：caculate-time$\Longleftrightarrow$plan-route$\Longleftrightarrow$Decsion$\Longleftrightarrow$Operation

References

[1] 全国大学生数学建模竞赛组委会, 高教社杯全国大学生数学建模竞赛论文格式规范, 北京，2009。

[2] 韩中庚，数学建模竞赛获奖论文精选与点评，北京：科学出版社，2007。

附录

Matalb程序由一下几部分组成：

- caculate-time.m (用时计算)
- plan-route.m (路线规划)
- Decsion.m (决策部分)
- Operation.m (模拟系统环境)

其中调用关系如下：caculate-time$\Longleftrightarrow$plan-route$\Longleftrightarrow$Decsion$\Longleftrightarrow$Operation

附录1　Operation.m

```
%2018_Mathmatic_Modling_Problem-B
%%
%模拟8h，仿真；
%Author:YXP
%Email:yxp189@foxmal.com
%Please feel free to contact us for any questions,thank you!
%%
%DATA
Move_Duration = ...%移动用时;行-不同距离；列-不同组别
    [20 23 18;...
    33 41 32;...
    46 59 46;];

Process_Duration = ...%加工用时;行-不同工序安排；列-不同组别
    [560 580 545;...
    400 280 455;...
    378 500 182;];

Load_Duration = ...%上下料（装载）用时；第一行-1/3/5/7；第二行-2/4/6/8;
    [28 30 27;...
    31 35 32;];

Clean_Duration = ...%清洗用时；列-不同组别
    [25 30 25];

%%
%Defined
%GROUP
Group = 1;  %目前组数;
```

```
%TIMER
Global_Timer = 0;  %全局计时器；从0s开始，持续8h(3600*8s);最短时间间隔1s
Gap = 1;    %离散化；时间间隔1s;
End_Time = 3600*8;  %结束时间8h(3600*8s);

%CNC加工台
CNC_State = zeros(1,8);     %CNC加工机床状态；空闲(0)/加工倒计时(remain);
All_free = zeros(1,8);
All_busy = ones(1,8);
CNC_Artifacts = zeros(1,8); %CNC上是否有工件；0-无/1-有;
CNC_Process = ones(1,8);    %CNC加工工序；1-Case1///2-工序1；3-工序2;

%RGV小车
%%%%%%%%%%%%%%%%%%%
%RGV STATE TABLE%
% Order   0     1    2     3          4            5             6
% State 空闲移动1 移动2 移动3 上下料1/3/5/7 上下料2/4/6/8 清洗
%%%%%%%%%%%%%%%%%%%
RGV_State = zeros(1,2); %RGV小车目前所处状态；(STATE,REMAIN)/(状态,剩余时间)
RGV_position = 1;       %RGV小车所在位置；(1-4)
RGV_destination = 1;    %RGV小车目的地；(1-4)

%CONVEYER传送带
Conveyer_State = ones(1,8); %Conveyer状态(0-无货；1-有货)
%%
clear;clc
%%
%Operation
Start_Position = 1;
Node = [1,2,3,4,5,6,7,8];
Node_num = 8;
Free = [1,2,3,4,5,6,7,8];
Time_Limit = 2000;
Group = 1;
CNC_Artifacts = [0,0,0,0,0,0,0,0];
CNC_Process =   [1,1,1,1,1,1,1,1];
plan = [];
duration = [];
```

```
while Global_Timer <= End_Time
    if isequal(All_busy,CNC_State)
        Time_Limit = 1500;
    else
        if isequal(All_free,CNC_State)
            Time_Limit = 1500;
            Free = Node;
        else
            Time_Limit = min(CNC_State(find(CNC_State)));
            Free = Node(find(~CNC_State));
            [plan,duration] = Decsion(Start_Position,Free,Time_Limit,Group,CNC_Artifac
        end
    end

    Size_free = size(Free);
    Start =  zeros(1,8);
    if  ~isempty(plan)  %有安排则计算开始时间
        for i=1:Size_free(2)
            Start(plan(i)) = duration(i);
            if CNC_Artifacts(plan(i)) == 1
                Start(plan(i)) = Start(plan(i)) - Clean_Duration(Group);
            end
        end
    end

    if ~isequal(Start,All_free)
        for i=1:Node




    for i=1:Node_num
         if CNC_State(i)~= 0
            CNC_State(i) = CNC_State(i) - 1;
         end
    end

    Global_Timer = Global_Timer + 1;
end
```

附录2　Decsion.m

```
function [plan,duration] = Decsion(Start_Position,Free,Time_Limit,Group,CNC_Artifacts,
%UNTITLED 此处显示有关此函数的摘要
%    此处显示详细说明
%2018_Mathmatic_Modling_Problem-B
%%
%Author:YXP
%Email:yxp189@foxmal.com
%Please feel free to contact us for any questions,thank you!
%%
%Clean
% clear;clc;
%%
%Data
% Start_Position = 1;
% Free = [1,2,3,4,5,6,7,8];
% Time_Limit = 800;
% Group = 1;
% CNC_Artifacts = [0,1,0,1,1,1,1,1];
% CNC_Process =   [2,3,3,3,2,3,2,3];
%%
%Operation Function
[Plan,Duration_matrix] = plan_route(Start_Position,Free,Time_Limit,Group,CNC_Artifacts
Duration = max(Duration_matrix')';
%%
%Various Index
%%%%%%%%%%
Pluse_order = find(Duration);
[~,T_I] = min(Duration(Pluse_order));%总时间计算;运用总时间判断
%%%%%%%%%%
BW_matrix = sum((~~Plan)')';
AVE = Duration./BW_matrix;%平均时间计算;
[~,A_I] = min(AVE);
%%%%%%%%%
%全部运行完成的最大值
Size_Route = size(Duration);
Size_Free = size(Free);
num = 1;
Max_idex = [];
```

```
Max_Achive = max(BW_matrix);
for i=1:Size_Route(1)
    if BW_matrix(i) == Max_Achive
        Max_idex(num,:) = i;
        num = num +1;
    end
end
[~,F_I] = min(Duration(Max_idex,:));
%%
%OUTPUT
%总量判据
disp('Pluse-TOTAL-Choice');
plan(1,:) = Plan(Pluse_order(T_I),:);
disp (plan(1,:));
disp (Duration(Pluse_order(T_I),:));
duration(1,:) = Duration_matrix(Pluse_order(T_I),:);
disp (duration(1,:));
%平均判据
disp('AVERAGE-Choice');
plan(2,:) = Plan(A_I,:);
disp (plan(2,:));
disp (Duration(A_I,:));
duration(2,:) = Duration_matrix(A_I,:);
disp (duration(2,:));
%完成判据/完成度判据
disp('FULL-Choice');
if isempty(F_I)
    disp('No Achive Case!');
else
    plan(3,:) = Plan(Max_idex(F_I,:),:);
    disp (plan(3,:));
    disp (Duration(Max_idex(F_I,:),:));
    duration(3,:) = Duration_matrix(Max_idex(F_I,:),:);
    disp (duration(3,:));
end

end
```

附录3　plan-route.m

```
function [Plan,Duration] = plan_route(Position,Free,Limit_Time,Group,CNC_Artifacts,CNC
```

```
%Plan_Route 找到指定点中，最优的安排。
% 给出车辆所在位置，空闲的CNC，决策时间限制；
%2018_Mathmatic_Modling_Problem-B
%%
%Author:YXP
%Email:yxp189@foxmal.com
%Please feel free to contact us for any questions,thank you!
%%
%Import data
Move_Duration = ...%移动用时;行-不同距离；列-不同组别
    [20 23 18;...
    33 41 32;...
    46 59 46;];

Process_Duration = ...%加工用时;行-不同工序安排；列-不同组别
    [560 580 545;...
    400 280 455;...
    378 500 182;];

Load_Duration = ...%上下料（装载）用时；第一行-1/3/5/7；第二行-2/4/6/8;
    [28 30 27;...
    31 35 32;];

Clean_Duration = ...%清洗用时；列-不同组别
    [25 30 25];

%Decision
Size_free = size(Free);
num = 1;
Choice = Free;

for i=1:Size_free(2)
    if  (CNC_Artifacts(i)==1)%&&(CNC_Process(i)==2)
        Choice(Size_free(2)+num) = Free(i);
        num = num+1;
    end
end

Test = perms(Free);
Size_Test = size(Test);
```

```
Duration = zeros(factorial(Size_free(2)),Size_free(2));
Result = zeros(factorial(Size_free(2)),Size_free(2));
i = 1;

while i <= Size_Test(1)
    [Duration(i,:),Result(i,:),Skip] = caculate_time(Position,Test(i,:),Limit_Time,Gro
    %用于减少调用
    if Skip~=0
        Temp_Duration = Duration(i,:);
        Temp_Result = Result(i,:);
        TT = i+factorial(Skip)-1;
        for j=i+1:TT
            Duration(j,:)= Temp_Duration;
            Result(j,:) = Temp_Result;
            i = i+1;
        end
    end
    i = i+1;
end

Plan = Result;

end
```

附录4　caculate-time.m

```
function [Duration,Result,Skip] = caculate_time(Position,Route,Limit_Time,Group,CNC_Ar
%caculate_time 计算特定路线的时间花费
%2018_Mathmatic_Modling_Problem-B
%%
%Author:YXP
%Email:yxp189@foxmal.com
%Please feel free to contact us for any questions,thank you!
%%
%Import data
Move_Duration = ...%移动用时;行-不同距离; 列-不同组别
    [20 23 18;...
    33 41 32;...
    46 59 46;];
```

```
Process_Duration = ...%加工用时;行-不同工序安排; 列-不同组别
    [560 580 545;...
    400 280 455;...
    378 500 182;];

Load_Duration = ...%上下料（装载）用时; 第一行-1/3/5/7; 第二行-2/4/6/8;
    [28 30 27;...
    31 35 32;];

Clean_Duration = ...%清洗用时; 列-不同组别
    [25 30 25];

%Caculate
Size_Free = size(Route);%这里的Free是经过了特别安排后的路线

Duration = zeros(1,Size_Free(2));    %设置初值0;
Result = zeros(1,Size_Free(2));
Step_Duration = 0;
Move = 0;
Temp_Position = Position;
Skip = 0;
num = 1;


LoadClean_Time = zeros(1,Size_Free(2));  %不含移动距离的装载+清洗时间
Case1 = ones(1,8);

if isequal(Case1,CNC_Process)%如果是Case1
    for i=1:Size_Free(2)     %不含移动距离的装载+清洗时间
        LoadClean_Time(i) = LoadClean_Time(i) + Load_Duration(2-mod(Route(i),2),Group)
序的上下料时间;
        if CNC_Artifacts(Route(i)) == 1
            LoadClean_Time(i) = LoadClean_Time(i) + Clean_Duration(Group);     %清
洗时间
        end
    end

    for i=1:Size_Free(2)
        Step_Duration = 0;
```

```
        Move = abs(Temp_Position - (Route(i)+mod(Route(i),2))/2);
        if Move ~= 0
            Step_Duration = Step_Duration + Move_Duration(Move,Group);
        end
        Step_Duration = Step_Duration + LoadClean_Time(i);
            if i ~= 1
                Gap = Duration(i-1)+Step_Duration;
            else
                Gap = Duration(i);
            end
        if (Gap) <= Limit_Time
            if i==1
                Duration(i) =  Step_Duration;
            else
                Duration(i) = Duration(i-1) + Step_Duration;
            end
            Temp_Position = (Route(i)+mod(Route(i),2))/2;
            Result(num) = Route(i);
            num = num+1;
        else
            Skip = (Size_Free(2)-num)+1;
            break;
        end
    end
else
    for i=1:Size_Free(2)     %不含移动距离的装载+清洗时间
        LoadClean_Time(i) = LoadClean_Time(i) + Load_Duration(2-mod(Route(i),2),Group)
序的上下料时间;
        if (CNC_Artifacts(Route(i)) == 1)&&(CNC_Artifacts(Route(i)) == 3)
            LoadClean_Time(i) = LoadClean_Time(i) + Clean_Duration(Group);     %清
洗时间
        end
    end

    Last_CNC = 0;    %上一个CNC状态0-空/1-Case2-1道工序工件/2-Case2-2道工序工件;

    %这里连续1工序后必须是2工序!
    for i=1:Size_Free(2)

        Move = abs(Temp_Position - (Route(i)+mod(Route(i),2))/2);
```

```
        if Move ~= 0
            Step_Duration = Step_Duration + Move_Duration(Move,Group);
        end
        Step_Duration = Step_Duration + LoadClean_Time(i);

        if CNC_Artifacts(Route(i))==1   %如果有货

            if (CNC_Process(Route(i))==2) && (i~=Size_Free(2))%如果恰好为1工
序
                Last_CNC = 1;
                if CNC_Process(Route(i+1))==2   %如果恰好为1工序,如果后一道也
为1工序

                    if i==1
                        Duration(i) =  Step_Duration;
                    else
                    Duration(i) = Duration(i-1) + Step_Duration;
                    end
                    Temp_Position = (Route(i)+mod(Route(i),2))/2;
                    Result(num) = Route(i);
                        Skip = (Size_Free(2)-num)+1;
                    break;
                else%有货，如果恰好为1工序,如果后一道为2工序或者是最后一道工序
                        if i ~= 1
                            Gap = Duration(i-1)+Step_Duration;
                        else
                            Gap = Duration(i);
                        end
                    if (Gap) <= Limit_Time
                        if i==1
                            Duration(i) =  Step_Duration;
                        else
                            Duration(i) = Duration(i-1) + Step_Duration;
                        end
                        Temp_Position = (Route(i)+mod(Route(i),2))/2;
                        Result(num) = Route(i);
                        num = num+1;
                    else
                        Skip = (Size_Free(2)-num)+1;
                        break;
```

```
                end
            end

        else%如果当前为2工序
                Last_CNC = 0;
                if i ~= 1
                    Gap = Duration(i-1)+Step_Duration;
                else
                    Gap = Duration(i);
                end
                if (Gap) <= Limit_Time
                    if i==1
                        Duration(i) =  Step_Duration;
                    else
                        Duration(i) = Duration(i-1) + Step_Duration;
                    end
                    Temp_Position = (Route(i)+mod(Route(i),2))/2;
                    Result(num) = Route(i);
                    num = num+1;
                else
                    Skip = (Size_Free(2)-num)+1;
                    break;
                end

        end
    else  %如果没有货物
        if (CNC_Process(Route(i))==3) && (Last_CNC==0)    %如果为2工序,且
之前没有1工序
            Skip = Size_Free(2) - i;
            break;
        else   %当前正好是工序1，或者之前有工序1
                if (CNC_Process(Route(i))==3)
                    Last_CNC=0;
                end
                if i ~= 1
                    Gap = Duration(i-1)+Step_Duration;
                else
                    Gap = Duration(i);
                end
                if (Gap) <= Limit_Time
```

```
                        if i==1
                            Duration(i) =  Step_Duration;
                        else
                            Duration(i) = Duration(i-1) + Step_Duration;
                        end
                        Temp_Position = (Route(i)+mod(Route(i),2))/2;
                        Result(num) = Route(i);
                        num = num+1;
                    else
                        Skip = (Size_Free(2)-num)+1;
                        break;
                    end
            end
        end
    end

end

end
```