

**计算建模**

**大项目结题报告**

题 目 模拟退火算法解决流水车间调度问题

学 院 计算机科学与技术

专 业 计算机科学与技术

学 号 1190200122

学 生 袁 野

任 课 教 师 范晓鹏 刘绍辉

哈尔滨工业大学计算机科学与技术学院

2021.12

# 模拟退火算法解决流水车间调度问题

1. **项目内容梳理（20分）**

如今，为了满足客户多样化与个性化的需求，多品种、小批量生产己经为一种重要的生产方式。与过去大批量、单一的生产方式相比，多品种、小批量生产可以快速响应市场，满足不同客户的不同需求，因此，受到越来越多的企业管理者的重视。特别是以流水线生产为主要作业方式的企业，企业管理者致力于研究如何使得生产均衡化，以实现生产批次的最小化，这样可以在不同批次生产不同品种的产品。在这种环境下，对于不同批次的产品生产进行合理调度排序就显得十分重要。

在传统的生产方式中，企业生产者总是力求通过增加批量来减小设备的转换次数，因此在生产不同种类的产品时，以产品的顺序逐次生产或用多条生产线同时生产。这样，必然会一次大批量生产同一产品，很容易造成库存的积压。在实际生产中如果需要生产A, B, C, D四种产品各100件，各种产品的节拍都是1分钟，如果按照传统的做法，先生产出100件A产品，其次是B，然后是C，最后生产产品D。在这种情况下，这四种产品的总循环时间是400分钟。然而，假设客户要求的循环时间为200分钟(四种产品的需求量为50件)，那么在200分钟的时间内就只能生产出产品A和产品B，因而不能满足客户需求，同时还会过量生产产品A和B，造成库存积压的浪费。这种生产就是非均衡的，如图1所示。

比较均衡的生产方式(图2 )是:在一条流水线上同时将四种产品混在一起生产，并且确定每种品种一次生产的批量。当然，如果在混合生产时不需要对设备进行转换，那么单件流的生产方式是最好的。然而，在实际生产A, B, C， D四种不同产品时，往往需要对流水线上的某些设备进行工装转换。单件流的生产方式在此难以实现，需要根据换装时间来确定每种产品一次生产的批量。同时，由于现实生产中不同产品在流水线上各台机器的加工时间很难相同，因此，流水线的瓶颈会随着产品组合的不同而发生变化。当同一流水线加工多产品，并且每种产品在各道工序(各台机器)的加工时间差异较大时，瓶颈就会在各道工序中发生变化，如何对各种产品的投产顺序进行优化以协调这些变化的瓶颈是生产管理中一个很重要的问题。

图1 图2

因而对流水线调度问题的研究正是迎合这种多品种、小批量生产方式的需要，我们要讨论得是如何对流水线上生产的不同产品的调度顺序进行优要化。

流水车间调度问题一般可以描述为n 个工件要在 m台机器上加工，每个工件需要经过 m 道工序，每道工序要求不同的机器，n个工件在 m台机器上的加工顺序相同。工件在机器上的加工时间是给定的，设为

问题的目标是确定个工件在每台机器上的最优加工顺序，使最大流程时间达到最小。

对该问题常常作如下假设:

(1)每个工件在机器上的加工顺序是给定的;

(2)每台机器同时只能加工一个工件;

(3)一个工件不能同时在不同机器上加工;

(4)工序不能预定;

(5)工序的准备时间与顺序无关，且包含在加工时间中;

(6)工件在每台机器上的加工顺序不同，且分别是确定的。

问题的数学模型：

：工件在机器加工的完工时间。

:第i台机器的工作调度。

个工件，台机器的流水车间调度问题的完工时间：

调度目标：确定j矩阵，最小化

分工如下：

|  |  |
| --- | --- |
| 袁野 | 模拟退火和评价函数的代码编写，测试 |
| 陆任驰 | 此时数据构造，文献查阅 |
| 王一凡 | 流水车间调度问题其他方法学习总结 |

1. **特定主题(topic)或特定文献的内容介绍-发展现状和发展趋势，所研究主体的国内外现状，最新进展等（20分）**

自从Jhonson 1954年发表第一篇关于流水车间调度问题的文章以来，流水车间调度问题引起了许多学者的关注，目前存在着以下几种解决方法：

1. 一种基于扩展采样空间的混合式遗传算法

将邻域搜索与遗传算法相结合求解流水车间调度问题，提出了一种邻域结构，使之更适合求解流水车间问题；设计了一种基于扩展采样空间的混合式遗传并通过计算机模拟验证其有效性。其中，邻域搜索使用定义（由给定的染色体通过随机移动一个基因到一个随机的位置，得到的是染色体的集合）所描述的邻域，采样空间为父代P(t)、改进的父代S(t)、交叉的后代C(t)、变异的后代M(t)，交叉和变异的父代是种群的父代P(t)，而不是改进的父代S(t)。

具体的混合式算法框架

BEGIN

t=0

初始化P(t)

WHILE不满足终止条件DO

① 下降搜索， 应用多点最速下降法改进p (t)，得到改进的父代S(t):

② 用P(t)进行单点交叉生成C(t);

③ 用P(t)进行移动变异生成M(t);

④ 采样从P(t)、S(t)、C(t)、M (t)中选出最好的不重复的下一代染色体：

t =t+1

END

1. 改进的DNA进化算法

改进的DNA 进化算法中引入了交换操作（交换操作就是在DNA单链中随意产生一个位置， 然后将位置前的DNA链与位置后的DNA链相交换，组成一条新的链）以更好地搜索解空间，并采用黄金分割率控制变异个体的数目。同时为了进一步提高搜索性能，采用一种新颖的启发式规则。具体算法如下：对于每个工件都有3 个时间指数：为工件j在所有机器上的加工时间之和; 为工件j在第一台机器上的加工时间；为工件j 在最后一台机器上的加工时间;为T件j 的加权加工时问。B、C 是[0,1]之间的数。当随机生成一个A,再在[0,1-A]之间随机产生一个B便能确定的大小，然后每个工件按照的降序排列，这样就会产生一个可行解。生成不同的A, 就会得到不同的可行解。将启发式算法得到的可行解作为DNA进化算法的初始群体。具体算法如下：

1. 计算每个工件的及；
2. for( I = 1, 2, …, n)(n 表示要产生的可行解的个数）；

A = random(0,1);

B = random(0,1-A):

B;

End

1. 根据每个工件计算出的 进行降序排列．得到对应的工件排序，即可行解。通过仿真可以验证，加入启发式算法能够快速地接近最优解，提高算法的收敛速度，产生初始种群。
2. 一种基于遗传算法的求解方法

一种基于遗传算法的求解方法，在由染色体转换成可行调度的过程中引入工件插入方法，同时设计了一种新的交叉算子（这里设计了一种新的交叉算子，从种群中按交叉概率随机选取两个个体作为父体，对于每个个体随机寻找两个不同的基因位置．选择这两个位置及其之间的基因作为交叉部分，两个交叉部分的长度可以不同。首先将两个交叉部分进行交换．然后按照父体中原来基因排列的顺序补齐交叉部分没有包含的基因，经过交叉之后产生的子代个体一部分基因保留了在一个父辈个体中的绝对位置，另一部分基因则保留了在另一个父輩中的相对位置。该操作具有较好的遗传特性，同时也能够产生足够的搜索空间。计算表明该算子优于PMX交叉算子。）通过大量的数值计算表明，该算法优化质量大大优于传统的遗传算法和NEH 启发式算法。

1. 一个无等待流水车间调度启发式算法

采用一个经典的全局任务插入算法构造初始解，应用局部搜索方法对其进行改进。通过4000个不同规模实例将提出算法与目前求解该问题最好的几个算法从性能和计算时间方面进行全面比较。实验结果表明：提出算法的性能是目前最好的，多项式复杂度的计算时间适合实际生产需求。此启发式算法包括两个阶段：初始序列的产生阶段和改进阶段。

(1) 在初始序列的产生阶段，采用任务插入的方法，它类似于NEH[3]算法。

(2) 在初始序列的改进阶段，定义v= (x, y) 为序列S 中的一对位置，其中：。v的移动将S中第x个任务插入到第y个位置，位置对集合：, 其中包括个位置对。算法描述如下：

①令k=1, 计算所有任务的F2 值。选择最小值对应的任务放入S中，将其余 个任务放入R中；

②;

③从R中任意取出一个任务j, 将其插入到S的k个不同位置，产生k个不同的序列，计算这k个序列的F1值，选择最小值对应的序列作为一个候选序列，将任务j从R中移除；

④如果R不为空集，返回第3步，否则转到第5步；

⑤在产生的个候选序列中，计算各自的F值，选择最小值对应的序列替换S. 将序列S以外的所有任务存放到集合R中；

⑥如果, 结束，S即为最终初始序列；否则回到第2步继续；

⑦生成序列S的位置对集合井进行插入操作，产生个新的任务序列，计算所有新产生序列的F1值，将最小值对应序列记为S'；

⑧如果，则，返回第7步重新开始，否则转入第9步；

⑨序列S即为最终任务序列。

1. 混合禁忌搜索算法(HTS)

(1) 混合禁忌搜索HTS算法的主要思路为：

通过一个有效的启发式算法为TS算法提供一个较好初始解，并可加快TS算法的收敛速度；采用禁忌搜索算法改进初始解以搜索到更好的近优解。初始解生成算法：

1. 任意产生一个初始序列;
2. 利用双插入启发式算法[5](DIH)对序列Q进行改进获取一个序列。 DIH 基于全局插入操作和局部插入操作的思想来产生局部种子序列井对当前调度进行改进。该算法具有较高效率的搜索能力，得到一个较好的近优解；
3. 将序列进行一次全局成对交换，得到初始序列P。

(2) HTS算法描述：

基于已得到的序列P作为初始解T和以上禁忌搜索算法，关键参数的设置，下面给出HTS 算法：

① 调用初始解生产算法产生初始解P并赋予;

② 将初始解T作为当前解利用成对交换(Swap) 产生的邻域结构得到多个邻域解；

③ 将所有邻域解对应的目标函数值从小到大排序，然后选取前e个邻域解作为候选解；

④ 从第1个候选解开始，如果满足藐视准则，则将此邻域解作为当对的序列；否则在候选解中选非禁忌的最佳状态序列作为当前序列;

⑤ 保存每个当前序列, 及其目标函数值，并找出其中最优的目标函数值及对应的序列 ；

⑥ 若满足终止条件，则比较最后得到的当前序列与序列 所对应的目标函数值大小，选取目标函数值小的序列作为算法最终所得到的近优解，算法停止；若不满足终止条件则，则转向2。

1. 混合规划

针对不确定条件下流水车间调度问题(Flow shop scheduling)，研究了含有随机参数和灰色参数的混合机会约束规划模型的建立及求解方法。提出了灰色模拟的概念和方法，为含有灰色参数的机会约束规划提供了求解途径。通过理论推导及仿真实例，结合遗传算法，验证了基于随机模拟和灰色模拟的混合机会约束规划的调度模型及求解方法的有效性。

1. **应用方案\应用系统\实验设置和结果展示（已有方法，改进的方法等，以及实验分析部分）（30分）**

使用模拟退火算法可以很好的求得流水车间调度问题的较优解。首先我们来介绍模拟退火算法。

模拟退火算法是一种模拟自然界物理过程的算法。通过模仿固体从高温徐徐冷却，内部分子从无序逐渐变成有序，内能减为最小的过程，使目标函数的值也达到最小。也就是说，我们可以设置一个变量来表示温度，而温度随循环不断减小，便是降温的过程。大量研究和实践保证了算法的较优性质，在收敛时有可观的效果。对于流水车间调度问题，这里针对目标函数特殊设计了计算的方法，利用了工件加工的有序性，前驱性，将约束条件转化为了 DAG 图上的边，通过遍历图的方式即可得出每个状态对应目标函数的值。

具体而言，我们可以按照以下几步实现模拟退火算法。

步骤一：设置一个初始温度 T，一个结束温度零温 T0，还有指数衰减率 rate

步骤二：如果初始温度 T<=零温 T0，则算法结束，将目前最优解作为答案

步骤三：随机产生一个当前状态邻域内的新状态，并用 DAG 评价函数估值

步骤四：如果新状态更优则接受；否则取两个状态估值的差 delta=new\_value-old\_value，以指数分布的概率接受新状态

步骤五：温度下降，转到步骤二

接下来我们介绍一下如何具体处理流水车间调度问题的状态。

首先我们可以对于每一台设备i都确定一个该设备的加工顺序 ,其中表示设备i加工的第j个工件。

其次很显然的是，如果我们确定了每一台设备中所有工件的加工顺序，那么这样情况下加工完所有工件所需的最短时间就是确定的，最短时间的求解方式如下：

对于n个工件和m台设备（即步骤），我们可以将其抽象为nm个点，每个点 代表由第i台设备加工第j个工件的任务。显然这些任务是有依赖关系的，而这些依赖关系可以转换为一张有向无环图。建图方式如下：

1. 对于同一台设备i，其加工序列中相邻的两个工件 和 对应的任务向连一条有向边，表示同一台设备的加工列表中的先后顺序。
2. 对于同一工件j，显然要由 向 连一条有向边，表示同一个工件的加工步骤先后顺序。

在该DAG上做topo排序并通过动态规划的方法计算关键路径长度，则到达 的关键路径长度与m设备加工n设备的时间之和即为当前状态下的最短加工时间。

为了满足模拟退火初始阶段的随机性，我们需要将某一个设备加工序列中的随机两个位置的工件序号交换，重复交换n\*m次后开始进行退火过程。

在每次降温之后，我们需要得到一个由当前状态产生的新状态，产生方式如下：

随机一个设备，再随即两个位置，交换该设备加工序列的这两个位置的工件标号即可。

重复上述过程直至温度将为0。

下面给出算法伪代码：

function Simulated-Annealing(problem, schedule) returns a solution state

inputs: problem, a problem

T, highest temperature

T0, lowest temperature

rate, temperature's down rate

local variables: current, current state

next, next state

current <- Random-initialization

while T>T0 begin

next <- a randomly selected successor of current

DAG-evaluate next

delta <- new-value - old-value

if delta>0 then current <- next

else current <- next only with probability exp(delta/T)

T <- T\*(1-rate)

end

实际执行效果如下：

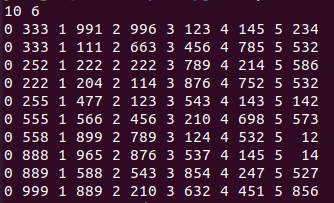


图 1

分别有10个工件和6台设备，对应加工时长如下图10\*12的矩阵所示。我们将其输入到程序中后，每降温1000次输出温度和当前状态的最小花费：

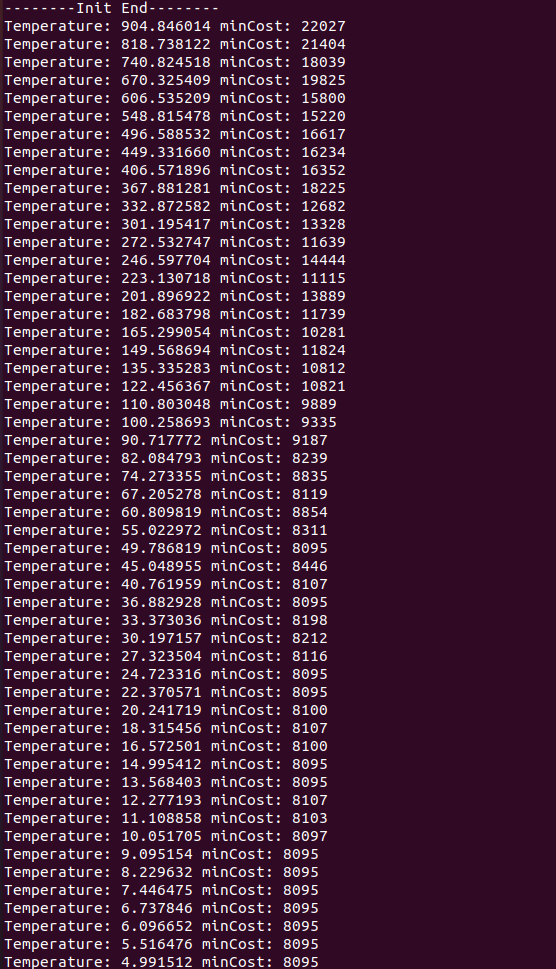


图 2

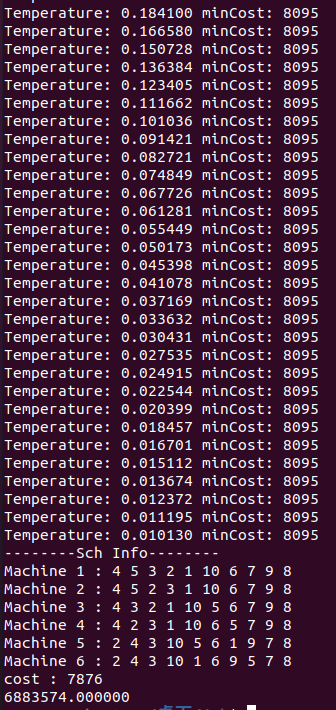


图 3

最终得到的最小代价为7876。执行的总时长为6.88s。最终的方案也如最后输出所示。

分析算法时间复杂度。可以看到算法的主体是一个循环，设循环执行了𝑘次，则有𝑇0 = 𝑇(1 − 𝑟𝑎𝑡𝑒)𝑘，可得 ，同时对于DAG-evaluate函数，建图复杂度，进行拓扑序动态规划复杂度，

故总体复杂度为，远低于寻找准确最优解所需的指数级复杂度。

分析空间复杂度。我们仅在 DAG-evaluate 函数中和状态的存储中使用到了一部分内存，并且复杂度均为，故在空间复杂度上也是可以接受的。

1. **思考、自己的见解、改进的基本想法、结论（20分）**

根据实验结果我们可以得到以下模拟算法的优缺点：

1.可以突破爬山算法的局限性，获得全局最优解 (以一定的概率接受较差解，从而跳出局部最优解）

2.初始解与最终解都是随机选取的，它们毫无关联，因此具有很好的鲁棒性，即抵御外界不稳定因素的能力。

3.其最优解常常受迭代次数k的影响，若k值越大，则搜索时间越长．获得的最优解更可靠；k值越小，则搜索时间越短，有可能就跳过了最优解。

4.模拟退火算法受温度冷却速率的影响，若冷却速率较慢，则搜索时间较长，可以获得更优的解，但是会花费大量时间；如冷却速度过快，可能较快的搜索到更优的解，但也有可能直接跳过最优解。

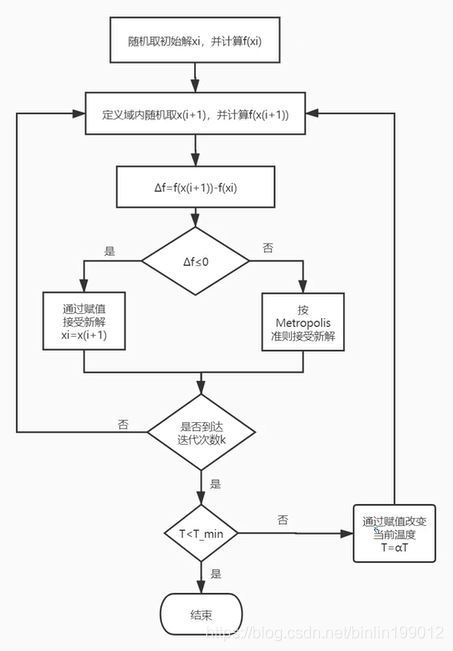


图 4

模拟退火算法的应用很广泛，可以高效地求解NP完全问题，如货郎担问题(Travelling Salesman Problem，简记为TSP)、最大截问题(Max Cut Problem)、0-1背包问题(Zero One Knapsack Problem)、图着色问题(Graph Colouring Problem)等等，但其参数难以控制，不能保证一次就收敛到最优值，一般需要多次尝试才能获得（大部分情况下还是会陷入局部最优值）。观察模拟退火算法的过程，发现其主要存在如下三个参数问题：

(1) 温度T的初始值设置问题

温度T的初始值设置是影响模拟退火算法全局搜索性能的重要因素之一、初始温度高，则搜索到全局最优解的可能性大，但因此要花费大量的计算时间；反之，则可节约计算时间，但全局搜索性能可能受到影响。

(2) 退火速度问题，即每个T值的迭代次数

模拟退火算法的全局搜索性能也与退火速度密切相关。一般来说，同一温度下的“充分”搜索是相当必要的，但这也需要计算时间。循环次数增加必定带来计算开销的增大。

(3) 温度管理问题

温度管理问题也是模拟退火算法难以处理的问题之一。实际应用中，由于必须考虑计算复杂度的切实可行性等问题，常采用如下所示的降温方式：

注：为了保证较大的搜索空间，一般取接近于1的值，如0.95、0.9。

本代码中为了尽量减少局部最优值的影响，会重复进行多次降温过程，每次降温的初始阶段为上一次退火过程的结果。

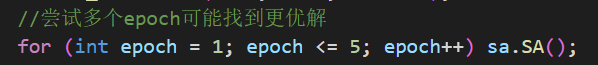


图 5

流水线问题是模拟退火算法的一个较为具象的应用。实际上，模拟退火算法已经在包括图像处理、VLSI设计、神经网络等多个领域实现应用。

1. **参考文献（10分）**

[1]冯爱芬,闻博卉,黄宇.基于模拟退火算法仓内拣货的优化问题[J].洛阳理工学院学报(自然科学版),2021,31(04):73-77.  
[2]曲媛, 杨晓伟. 关于流水车间调度问题的综述[J]. 中小企业科技, 2007(8):24-25.

[3]包波. A公司自动化总装车间AGV路径优化问题研究[D].北京交通大学,2020.DOI:10.26944/d.cnki.gbfju.2020.002468.

[4]Ribas Imma,Companys Ramon,Tort-Martorell Xavier. An iterated greedy algorithm for the parallel blocking flow shop scheduling problem and sequence-dependent setup times[J]. Expert Systems With Applications,2021,184:  
[5]曹瑞瑞,孔建寿.改进GA算法求解自动开票流水线生产调度问题[J].工业控制计算机,2021,34(06):4-7+9.  
[6]王艺霖,郑建国.离散蝙蝠算法在三阶段装配流水线调度问题的应用[J].控制与决策,2021,36(09):2267-2278.DOI:10.13195/j.kzyjc.2020.0054.