文章编号：1000－8152(2000)01－0031－04

31-34

车间调度算法的研究和开发*

0224

曹承煜 李人厚 樊 健
(西安交通大学系统工程研究所·西安，710049)

摘要：针对车间调度问题，提出了一种改进的拉氏松弛算法．在增加辅助目标函数的基础上，通过对子问题的限制和搜索策略的改变，使拉氏算法的计算量减少，近优解的搜索能力有很大改善．本文还提出了一种基因优化算法，充分利用拉氏算法得到的多个近优解，进一步优化结果．仿真结果表明对车间调度问题得到了较好的结果．本方法也可用于其它有约束的规划问题．

关键词：车间调度问题；拉氏松弛；遗传算法
文献标识码：A

Study on Job Shop Scheduling Problem

CAO Chengyu, LI Renhou and FAN Jian
(Institute of System Engineering, Xi'an Jiaotong University·Xi'an, 710049, P. R. China)

Abstract: In this paper, we propose an improved Lagrangian relaxation algorithm to solve job-shop scheduling problems. Besides the addition of augmented objective, we expand the search scope of near-optimal solutions and improve the computational efficiency greatly by restricting the solution scope of sub-problems and modifying the search method of dual problem. At the same time, we develop a genetic algorithm combining with the LR(Lagrangian relaxation)method. Using the numerous useful solutions we get in the Lagrangian relaxation as the original genes, we can improve the solution further. Test results show that these methods achieve satisfied outcome for job-shop problems. They can also be applyed to other programming problems with constraints.

Key words: job shop scheduling problem; Lagrangian relaxation; genetic algorithm

1 引论(Introduction)

车间调度问题(Job Shop Schedule Problem)一直是调度领域和 FMS 中人们广泛注意的问题．随着现代制造业的发展，小批量、多类型的、有不同完工时间和产品要求的产品需求逐渐取代大批量单类型的产品制造计划．在这种情况下，如何利用现有的资源(加工能力)，满足被加工任务所需的各种约束(加工次序，所需机器)，使所有的任务能尽量按时完成(性能指标最小)，就成为一个十分现实和迫切的问题．另外，实际中的车间调度是一个动态的不断演化过程，在所见的文献中，都是对一个单独的静态问题开展讨论．为了使算法能适应实时的任务调度，Job Shop 问题除了传统的机器负载约束和同一任务间各操作的前后约束外，还要加一个最早开始时间约束．在拉氏松弛方法中，解的震荡与收敛问题一直是该方法应用的一个巨大障碍．Debra Hoitomt 和 Peter Lub[2] 使用 Augmented Lagrangian Relaxation Approach，即在目标函数中加入与约束有关的罚项，通过对违背约束形成一个较大的目标函数代价来加速收敛．但它使操作间有了耦合，不能简单地分解为子问题．本文采用了类似于辅助目标函数法的方法，对每一个任务，计算出每个操作的最早时间和任务有可能按时完成时的各操作最晚完成时间．通过最早时间的计算，可以将任务开始时间约束分散到操作中．对子问题加上此约束还可导致问题的简化．各操作的最晚完成时间作为一个辅助目标函数加入原目标函数中，使任务不能按时完成的信息体现在各操作中，减少了震荡，促使更快的收敛．

为了获得对偶问题的最优解，我们采取了一些措施．一个方面扩大了寻找对偶问题中次优解的范围，改变原来 LR 方法中单独寻找对偶问题最优解的方法；另一方面，以这些次优解为基础，再用基因算法进一步优化解答．

2 问题建立(Formulation)

所有变量的定义见附录 1．

δ_{ijkh} 可由 b_{ij} 与 m_{ij} 很容易得到，其中 b_{ij}，m_{ij} 是决

* 基金项目：国家 863 计划(863－511－9508－003)资助项目．
收稿日期：1997－12－5；收修改稿日期：1999－2－5．

策变量.其余量均是已经给出的量.

Job Shop 问题所要满足的约束为:

1) 负载约束:

$$\begin{cases} \sum_{i=j}\delta_{ijkh} \leqslant M_{kh},\ 1\leqslant k\leqslant N,\ 1\leqslant h\leqslant M, \\ 1\leqslant i\leqslant N,\ 1\leqslant j\leqslant N_i-1. \end{cases} \tag{1}$$

2) 操作前后约束条件:

$$b_{ij}+t_{ij,m_{ij}}-1\leqslant b_{il}, \tag{2}$$

$$1\leqslant i\leqslant N,\ 1\leqslant j\leqslant N_i-1,\ 1\in I_{ij}.$$

3) 开始时间约束:

$$b_{ij}\geqslant e_{ij},\ 1\leqslant i\leqslant N,\ 1\leqslant j\leqslant N_i-1. \tag{3}$$

Job Shop 问题便是在满足这些约束的条件下,使性能指标 J 最小.

$$J=\sum_i W_iT_i^2=\sum_i J_i.$$

引入辅助目标函数项后,目标函数成为:

$$Jaux=\sum_i\Big(J_i+\sum_j\beta_{ij}T_{ij}\Big)=\sum_i Jaux_i,$$

$$1\leqslant j\leqslant N_i,\ 1\leqslant j\leqslant N_i-1.$$

式中: $Jaux_i=J_i+\sum_j\beta_{ij}T_{ij}$,

$T_{ij}=\max(0,b_{ij}-l_{ij})$, $1\leqslant j\leqslant N_i$, $1\leqslant j\leqslant N_i-1$.

l_{ij} 为最迟完成时间, β_{ij} 是辅助目标函数项的系数, $\beta_{ij}=w\times W_i$. w 是给定的参数.

3 拉氏松驰求解方法(Lagrangian relaxation approach)

在此问题中, $Jaux$ 目标函数可以按任务和操作分解,将约束 1)与 2)用拉氏松驰方法分解,约束 3)可以分解代入到各个操作的子问题中.

3.1 拉格朗日松弛方法(The Lagrangian relaxation approach)

将机器负载约束 1)用拉氏乘子 π_{kh} 松驰,前后约束 2) 用拉氏乘子 λ_{ijl} 松驰, $\pi_{kh}\geqslant 0, \lambda_{ij1}\geqslant 0$.

利用(2)式得:

$$\min_{b_{ij}|_{i=1,\cdots,m_{ij}}}\Big\{\sum_i\Big(Jaux_i+\sum_{j,l}(\lambda_{ijl}(b_{ij}+t_{ij,m_{ij}}-b_{il}))+\sum_{kh}\pi_{kh}\Big(\sum_{i,j}\delta_{ijkh}-M_{kh}\Big)\Big\}, \tag{4}$$

$$1\leqslant j\leqslant N,\ 1\leqslant j\leqslant N_i-1,\ 1\in I_{ij},\ 1\leqslant h\leqslant M,$$

$$0\leqslant k\leqslant H.$$

s.t. (1)~(3).

Lagrangion 对偶问题为:

$$\max_{\pi,\lambda\geqslant 0}\Big\{\min_{b_{ij}|_{i=1,\cdots,m_{ij}}}\Big\{\sum_i\Big(Jaux_i+\sum_{j,l}(\lambda_{ijl}(b_{ij}+t_{ij,m_{ij}}-b_{il}))+\sum_{kh}\pi_{kh}\Big(\sum_{i,j}\delta_{ijkh}-M_{kh}\Big)\Big)\Big\}\Big\}, \tag{5}$$

s.t. (1)~(3).

对于给定 π,λ, 求对偶问题可以分解到以操作为单位的子问题:

$$\max_{\pi,\lambda\geqslant 0}\Big\{\sum_{i=1}^{N}\sum_{j=1}^{N_i}\min_{b_{ij}|_{i=1,\cdots,m_{ij}}}\Big(Jaux_i+\sum_{k=0}^{H}\sum_{h=1}^{M}\pi_{kh}\delta_{ijkh}+\sum_{l\in I_{ij}}\theta(i,j)\lambda_{ijl}(b_{ij}+t_{ij,m_{ij}})-\sum_{p\in P_{ij}}\lambda_{ipj}b_{ij}\Big)-\sum_{k=0}^{N}\sum_{h=1}^{M}\pi_{kh}M_{kh}\Big\}. \tag{6}$$

式中 $N_{I_{ij}}$:操作(i,j)的后继操作的个数.

$$\theta(i,j)=\begin{cases}0, & j=N_i,\\ 1, & \text{其它}.\end{cases}$$

p_{ij} 是操作(i,j)的所有前继操作

$$P_{ij}=\{x\mid(i,j)\in I_{ix}\}.$$

对给定的 π,λ, $\sum_{k=0}^{N}\sum_{h=1}^{M}\pi_{kh}M_{kh}$, 相当于一常量.这样,每个单独的子问题为:

$$\min_{l_{ij}|_{i=1,\cdots,m_{ij}}}\Big(Jaux_i+\sum_{k=0}^{H}\sum_{h=1}^{M}\pi_{kh}\delta_{ijkh}+\sum_{l\in I_{ij}}\theta(i,j)\lambda_{ijl}(b_{ij}+t_{ij,m_{ij}})-\sum_{p\in P_{ij}}\lambda_{ipj}b_{ij}\Big). \tag{7}$$

s.t. (1)~(3).

3.2 解子问题(Solving subproblem)

在解决子问题(7)时,以往的方法总是对 m_{ij} 和 b_{ij} 的所有组合选择一个最优解,当时间跨度很长时,迭代次数是相当多的.

(7)式又可以写作:

$$\min_{m_{ij}|_{i=1,\cdots,b_{ij}}}\Big(Jaux_i+\sum_{k=0}^{H}\sum_{h=1}^{M}\pi_{kh}\delta_{ijkh}+\Big(\sum_{l\in I_{ij}}\theta(i,j)\lambda_{ijl}-\sum_{p\in P_{ij}}\lambda_{ipl}\Big)b_{ij}+\sum_{p\in P_{ij}}\theta_{ij}\lambda_{ijl}t_{ij,m_{ij}}\Big). \tag{8}$$

b_{ij} 的选取受

$$\Big(\sum_{l\in I_{ij}}\theta(i,j)\lambda_{ijl}-\sum_{P\in P_{ij}}\lambda_{ipj}\Big) \tag{9}$$

的影响相当严重.当(9)式值为正时,解中 b_{ij} 倾向于变小;反之,为负时, b_{ij} 倾向于变大.(9) 式的符号变换会对 b_{ij} 的求解带来巨大的震荡.假如机器负载乘子 π_{kh} 比较均衡,那么(9)式为负时,求解(8)式得到的会是最大值 $(H-t_{ij,m_{ij}}+1)$;而(9)式为正时,解又会是最小值 0.因为时间区域的选取总是能够完成所有的操作,当 k 比较大,即接近 H 时, π_{kh} 总为 0.所以,每当(9) 式刚为负时, b_{ij} 不可避免要选择最大值,它马上又返回一个很大的误差项,于是乘子大

幅度修正，进而又使 b_{ij} 很小，形成震荡.

为了克服 b_{ij} 的剧烈变动和前后震荡，我们通过限制 b_{ij} 的变动范围来解决这个问题. 子问题求解时，限制 b_{ij} 只能前后变化若干步，以克服 b_{ij} 大幅度震荡. 如果 b_{ij} 确实需要大幅度后移，通过几次迭代，也很容易实现.

3.3 求解对偶问题(Solving dual problem)

解决对偶问题的实质是找到一组合适的拉氏乘子. 现采用次梯度法来迭代，对机器负载的约束来说：

$$\pi^{n+1} = \pi^n + a^n S^n.$$

S^n 为当前的搜索方向，a^n 为当前的步长.

$$a^n = \beta \frac{L^u - L^n}{(S^n)^T(S^n)},\ 0 < \beta \leqslant 1.$$

L^u 是对偶问题最优解的估计上界，L^u 是第 n 次迭代时 L 函数值.

当前的搜索方向如下：

$$S^0 = g(\pi^0),\ S^n = g(\pi^n) + \gamma^n S^{n-1},\ n = 1,2,\cdots,$$

$$\gamma^n = \max\left[0, -\varepsilon \frac{(S^{n-1})^T(g(\pi^n))}{(S^{n-1})^T(S^{n-1})}\right],$$

ε 为参数，$g(\pi^n)$ 是第 n 次迭代的次梯度. 在第 h 个机器类型和第 k 时刻其分量为：

$$\sum_{ij}(\delta_{ijkh} - M_{kh}).$$

4 遗传算法(Genetic algorithm)

为了得到更好的目标函数值，最后用基因方法来进一步寻优. 因为对偶问题的计算并不能完全对等于原问题，所以对原问题直接寻优是很重要的

由于基因算法主要是用来处理无约束的参数优化，而车间调度问题的困难正在于大量的约束，为了使基因算法能用于有约束的规划问题，可以在基因表示中就将约束关系考虑在内. 另外，我们改变变异与变换策略，以保证所得到的基因不违反约束.

4.1 基因编码(Genetic encoding)

因为具有众多的约束条件，如果将一个可行解转化为二进制表示的话，特别是 b_{ij}，基因的交换与变异产生的只会是众多的无用的违反约束条件的解. 为此，我们用各操作的开始次序和它们所使用的机器类型来表示一个可行的原问题解. 基因串的长度是总的操作数，它分为若干个单元，每一个单元唯一的对应一个操作(i,j). 单元包含三个元素，分别为：任务号 i；操作号 j；操作(i,j) 所使用的机器类型 m，而操作(i,j) 的起始时间 b_{ij} 隐含地保存在该元素整条基因中的位置上.

对于从可行解构造算法中得到的一组 $\{b_{ij}, m_{ij}\}$，构造一个基因的方法是按照 b_{ij} 从大到小的顺序依次写入一条基因中，如图 1：

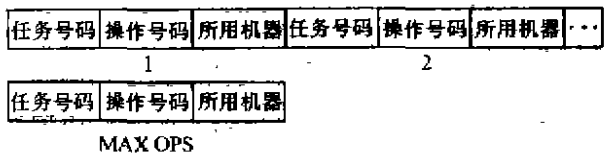

图 1　基因编码

Fig. 1　Genetic encoding

按各操作在基因中的排序构造可行解，在满足机器负载约束和操作前后负载约束的条件下，每个操作只取它最早可能的开始时间写入 b_{ij}，而所用的机器类型便是基因中记录的 m_{ij}.

4.2 基因变异乘子(Genetic mutation factory)

基因中 P 位置操作(i,j) 所用机器类型突变就是在 H_{ij} 中按一定概率选一新的类型并对 m_{ij} 赋值.

基因中 P 位置操作(i,j) 次序变异就是当前位置 P 处操作与 $P-1$ 位置操作不具有前后约束关系时，交换 $P-1$ 与 P.

4.3 基因交换算子：$g1$，$g2$，$posi$，$posj$ (Genetic crossover operator: $g1$, $g2$, $posi$, $posj$)

即交换两条基因 $g1$，$g2$ 中从起始位置 $posi$ 到结束位置 $posj$ 中的元素. 它的主要思想是在不违反前后约束的前提下，交换两基因中 $posi$ 与 $posj$ 之间的各操作的相对位置信息，从而隐含着对 b_{ij} 的交换. 步骤如下(见图 2)：

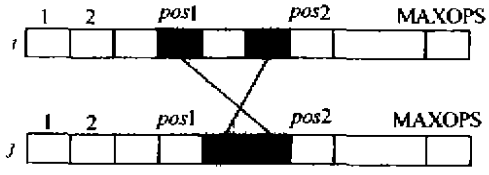

图 2　基因交换算子

Fig. 2　Genetic crossover operator

1) 找出 $posi$ 与 $posj$ 间两基因所有的共同操作；

2) 在两基因中分别记录下 $posi$ 与 $posj$ 中所有操作的当前位置 pos；

3) 交换两基因中 $posi$ 与 $posj$ 间的共同操作，包括(i,j,m_{ij},pos)；

4) 对两个基因中 $posi$ 与 $posj$ 间的区域分别进行重排序.

通过该基因交换算子的作用，首先保证得到的基因中各操作仍然满足任务中原来的偏序关系. 另外，又将两基因中操作位置次序与所用机器的信息进行了互换，故它们对基因中其它部分的偏序关系仍然保证.

MAXOPS 为操作总数.有连线的两个操作为进行交换的共同操作.

4.4 构造可行解与性能评估(Constructing feasible solution and performance valuation)

得到基因后,要对基因的性能进行评估.就可直接用基因构造出一个可行解(用 4.1 中的步骤),然后计算该结果的目标函数 J,作为评价原则.

5 实验结果(Experimental results)

举例:有八个相同权值的任务($W = 10$),其中包括分支和多后继的任务类型.有三台不同的机器,构成任务的各操作可在任一台机器上运行,并且有不同的处理时间.任务的开始时间均为 0,预定完成时间设为 10.

为解此问题,使用改进的 LR 方法,并在最后用基因算法进一步优化.结果见表 1:

表 1 例 1 运行的结果(改进 LR 方法)

Table 1 The operation results of the example

LR 底层迭代次数	533
LR 上层迭代次数	17
LR 方法得到的原问题目标函数	24310
基因算法迭代次次数	20
基因算法得到的原问题目标函数	22000

6 结论(Conclusion)

在本文中,通过将拉氏松驰算法与基因算法相结合,针对车间调度问题,我们提出了一套有效的解决算法.通过限制对偶问题局部渐变来克服震荡,并极大的降低了计算量.在上层算法中,通过扩大搜索范围,得到众多可行解作为基因算法的寻优基础.这也可以弥补子问题中局部寻优的缺陷.从数值例子中可以看出,算法是有效和快速的.这种思想也完全可以应用于其它有约束的规划问题中.

参考文献(References)

[1] Chang Shu-Chuang and Liao Da-Yin. Scheduling flexible flow shops with no setup effects[J]. IEEE Trans. on Robotics and Automation, 1994, 10(2): 112 - 121

[2] Debra J H and Peter B L. A practical approach to job-shop scheduling problems[J]. IEEE Trans. on Robotics and Automation, 1993, 9(1):

[3] Christopher S C and Peter B L. Scheduling products with bills of materials using an Improved Lagrangian relaxation technique[J]. IEEE Trans. on Robotics and Automation, 1994, 10(2):

[4] Guan Xiaohong and Peter B L. A new parallel algorithm for optimal control problem of interconnected systems[J]. Int J. Control, 1992, 56 (6): 1275 - 1297

附录 变量定义(Appendix Definition of variables)

任务 i 的 j 操作被简写为操作 (i,j);

N:任务总数;

N_i:任务 i 的操作数目;

M:机器总数;

b_{ij}:操作 (i,j) 的开始时间;

C_{ij}:操作 (i,j) 的结束时间;

H:调度区间范围;

C_i:任务 i 的完成时间;

D_i:任务 i 的预定完成时间;

I_{ij}:任务 i 中操作 (i,j) 的后继操作集;

P_{ij}:任务 i 中操作 (i,j) 的前继操作集;

H_{ij}:能处理操作 (i,j) 的机器类型集;

t_{ijh}:操作 (i,j) 在机器类型 h 上的处理时间, $h \in H_{ij}$;

$N_{I_{ij}}$:任务 i 中操作 (i,j) 的后继操作个数;

$N_{P_{ij}}$:任务 i 中操作 (i,j) 的前继操作个数;

m_{ij}:被选择用来处理操作 (i,j) 的机器类型, $m_{ij} \in H_{ij}$;

M_{hk}:机器类型 h 在时间 k 上的负载能力;

T_i:任务 i 的延时,定义为 $\max[0,(C_i - D_i)]$;

B_i:任务 i 的最早开始时间;

e_{ij}:操作 (i,j) 的最早开始时间;

l_{ij}:操作 (i,j) 在不延时时最晚结束时间;

W_i:任务 i 的权值;

δ_{ijhk}:整数变量,如果操作 (i,j) 在时刻 k 在机器类型 h 处是激活的, $\delta_{ijhk} = 1$;否则, $\delta_{ijhk} = 0$;

J:原问题性能指标.

本文作者简介

曹承煜 1970 年生.1995 年为交通大学系统所硕士研究生,1996 年攻读博士学位,1997 年去美国 MIT 继续学习.主要研究方向为:智能控制,离散事件系统,CIMS.

李人厚 1935 年生.1986 年任教授,博士生导师.目前主要研究方向为:大系统理论与应用,智能控制,CSCW.编著有教材,专著 10 部,发表论文 150 多篇.

樊 健 1971 年生.西安交通大学系统工程研究所硕士生 现在深圳工作.