

## 求解车间作业调度问题的一种改进遗传算法

光 熠, 刘心报, 程 浩

(合肥工业大学, 安徽 合肥 230009)

**摘 要:**针对标准遗传算法收敛速度慢和易陷入局部最优的问题,在总结已有经验的基础上对标准遗传算法提出改进:采用基于工序的编码、解码方式,每一次遗传操作后对种群采用循环选择并保留最优个体,对交叉操作和变异概率的计算提出了一系列改进方法,避免遗传算法产生无用解或陷入局部优化,以提高效率。通过实验验证,改进后的算法具有可行性,并且可以得到十分满意的结果。

**关键词:**遗传算法; Job Shop 调度问题; 组合优化

**中图分类号:** TP301.6; O221.7

**文献标识码:** A

**文章编号:** 1673-629X(2007)11-0171-04

## An Improved Genetic Algorithm in Job-Shop Scheduling Problem

GUANG Yi, LIU Xin-bao, CHENG Hao

(Hefei University of Technology, Hefei 230009, China)

**Abstract:** The standard genetic algorithm is slow and easy to fall into a local optimum. Based on the experiences an improved genetic algorithm was proposed in job shop scheduling problem; acquired on the basis of the standard genetic algorithm proposed improvements: coding and decoding method based on work order, use cycle selection after each operating and save the optimal individual, improve the crossover and calculation methods of mutation probability. This can avoid useless solution or local optimization, so it will be more efficient. Experiments on job shop benchmarks show that this improved algorithm has good performance.

**Key words:** genetic algorithm; job-shop scheduling problem; combinatorial optimization

## 0 引言

车间作业调度问题(JSSP)是最困难的约束组合优化问题,对于  $n$  个作业,  $m$  台处理机的同顺序作业排序问题,全部的可能排序数为  $(n!)^m$ <sup>[1]</sup>。而且,车间作业调度问题也是典型的 NP 难问题,没有一个有效的算法能在多项式时间内求出其最优解<sup>[2,3]</sup>。文献[4]总结出目前研究 JSSP 的方法可以分为两大类:最优化方法和近似/启发式方法。最优化方法有混合整数线性规划(MILP)、分枝定界(B&B)法以及拉氏松弛法等;近似/启发式方法主要包括优先分派规则(PDRs)、人工智能(AI)、神经网络(NN)及邻域搜索法(NS)等。这些研究虽然取得了一定的应用效果,但是却存在着计算规模不可能较大、寻优结果不具备全局特性、计算速度慢等弱点。

20 世纪 80 年代中后期以来,随着人工智能和计算机技术的发展,一些较复杂的优化方法,如神经网络、模拟退火和遗传算法等,都得到了迅速发展,并成为调度理论的研究热点。文献[5]指出,由于遗传算法(GAs)具有处理问题的柔软性和并行处理的自然特性,可在很大状态空间随机高效地采样和搜索,并很快收敛到最优或近似最优解,因此在调度领域受到广泛的关注。实验证明 GAs 在多种判据方面的调度性能均优于启发式方法和随机爬山法,说明 GAs 很适合求解调度问题。Hart 等<sup>[6]</sup>分析了遗传算法的优势和弱点,深入调查遗传算法求解 JSP 的优化性能,并设计了一个可调整难度的可配置问题发生器,通过研究表明遗传算法具有相当好的鲁棒性。

文中对传统的遗传算法进行改进,针对 JSSP 问题的特性,采用基于工序的方式进行编码,并提出相对应的交叉算子、变异算子,避免了无用解的产生。数据模拟的结果表明,改进后的算法收敛较快,而且效果较好。

文中对传统的遗传算法进行改进,针对 JSSP 问题的特性,采用基于工序的方式进行编码,并提出相对应的交叉算子、变异算子,避免了无用解的产生。数据模拟的结果表明,改进后的算法收敛较快,而且效果较好。

## 1 问题描述

JSSP(Job Shop Scheduling Problem)可描述为:  $m$  台机器(用集合  $M = \{M_j\}_{j=1}^m$  表示)加工  $n$  个工件(用

收稿日期:2007-01-10

作者简介:光 熠(1982-),男,安徽合肥人,硕士研究生,研究方向为决策科学与技术;刘心报,博士,教授,博士生导师,研究方向为决策科学与技术。

集合  $J = \{J_i\}_{i=1}^n$  表示), 每个工件包含由多道工序组成的一个工序集合。工件有预先确定的加工顺序, 每道工序的加工时间为  $t_{ij}$ , 在给定的时间内每台机器只能加工一个工件, 并且每个工件只能由一台机器处理。不同工件的加工顺序无限制, 工序不允许中断; 要求在可行调度中确定每个工序的开始时间  $s_{ij}$ , 使总完工时间  $C_{\max}$  最小, 即  $C_{\max}^* = \min(C_{\max}) = \min\{\max(s_{ij} + t_{ij}) : \forall J_i \in J, M_j \in M\}$ , 求解满足以上条件的工件加工顺序即构成 JSSP 调度问题。

Adams<sup>[7]</sup> 提出用集合  $N = \{0, 1, 2, \dots, n\}$  表示工序, 0 和  $n$  表示虚设的起始和完成工序;  $M$  是机器集合;  $A$  是预先确定加工顺序的工序对;  $E_k$  是机器  $k$  上加工的工序对集合; 工序  $i$  的加工时间  $d_i$  是确定的, 完工所用时间  $t_i$  是优化变量, 则 JSSP 问题的线性规划模型表示为:

$$\min t_n$$

$$t_j - t_i \geq d_i, \quad (i, j) \in A$$

$$t_i \geq 0, \quad i \in N$$

$$t_j - t_i \geq d_i \vee t_i - t_j \geq d_j \quad (i, j) \in E_k, k \in M$$

## 2 算法设计

遗传算法是 Holland 基于自然遗传进化的绝对模型提出的并行搜索机制<sup>[8]</sup>, Goldberg 总结出标准遗传算法(SGA)作为统一的算法形式, 是研究遗传算法的基础。SGA 依据适应度函数, 对初始群体中的个体采用遗传操作, 模拟生物基因的遗传和变异, 实现个体结构重组的迭代处理。在这一过程中, 初始群体中的个体一代又一代地优化并逐步逼近最优解。遗传操作主要有选择、交叉和变异。

### 2.1 编码和解码

编码是将问题的解表示成某种形式的排列, 每一个排列被称为一个染色体。遗传算法就是通过对染色体的进化淘汰, 达到优化求解的目的。对 JSP 问题进行编码的关键是必须考虑每个工件的加工顺序, 如果编码时就把工件的加工顺序编进染色体, 可以使解码相对简单, 但会使染色体变长, 增大解空间, 影响遗传算法的求解效率; 如果编码不当, 会在遗传算子操作时产生不可行解, 失去了交叉或变异算子的有效性。因此合适的编码是设计简单高效遗传操作的关键。Cheng 等<sup>[8,9]</sup>把编码表示分为两类共 9 种, 分别是直接的方法(基于工序、基于作业、基于作业对关系、基于完成时间和基于随机键表示法)和间接的方法(基于优先规则、基于偏好列表、基于非连接图和基于机器的表示法)。笔者采用基于工艺约束的个体编码方式, 每个个

体都是由基因元素组成的有序集合, 每个基因代表一道工序, 对所有同一工件的工序使用相同的符号。

以某一 3 工件/4 机床调度问题为例:

工件 1 (1, 1, 4, 1) (1, 2, 2, 2) (1, 3, 3, 4)  
(1, 4, 4, 1)

工件 2 (2, 1, 1, 2) (2, 2, 3, 2) (2, 3, 2, 3)

工件 3 (3, 1, 3, 2) (3, 2, 1, 5) (3, 3, 4, 1)  
(3, 4, 2, 2)

括号内第一个数字代表工件号, 第二个数字代表工件工序号, 第三个数字代表本工序所在机床号, 第四个数字代表该工序的加工时间。

染色体由一组数字组成, 一个数字代表一个工件的某一道工序, 在染色体中不同位置上的同一个数字, 表示该工件的不同工序。则该问题对应的染色体是由 4 个 1, 3 个 2, 4 个 3 等数字组成的一个字符串。如: 2-2-1-3-2-1-3-3-1-3-1, 其中第  $j$  次出现的数字  $i$  代表工件  $i$  的第  $j$  道工序。

解码时, 先将染色体中每个基因转换为所对应的工件, 然后再根据它们在染色体中出现的顺序确定对应的工序, 可以得出每个工件每道工序在机器上的加工时间段, 再依次计算每台机器最后的工作完成时间, 其最大值就是目标函数(即机床的最大完成时间)。

### 2.2 选择操作

遗传算法采用优胜劣汰的方式进行优化选择, 染色体的适应度越高, 被遗传下来的概率就越大。采用轮盘赌选择, 设群体大小为  $M$ , 个体  $i$  的适应度为  $F_i$ , 则个体  $i$  被选中的概率  $P_{is}$  为:

$$P_{is} = F_i / \sum_{i=1}^M F_i \quad (i = 1, 2, \dots, M)$$

### 2.3 交叉操作

对于基于排序编码的染色体个体, 若采用传统的单点交叉则可能产生无意义的编码串。文中采用单点交叉和顺序交叉相结合的单点顺序交叉算子<sup>[10]</sup>, 算法如下:

(1) 在任取的两个父本个体编码串 Parent1 和 Parent2(以下简称  $P_1$  和  $P_2$ ) 中, 随机选择一个基因座  $i$ , 将第 1 个基因到第  $i$  个基因的位置作为交叉区域, 并把交叉区域内的基因编码串分别记录到  $T_1$  和  $T_2$  中。

(2) 在父本个体  $P_1$  中从左向右查找  $T_2$  中的所有基因值, 将  $P_1$  中对应位置的基因值置为 0; 同样在父本个体  $P_2$  中从左向右查找  $T_1$  中的所有基因值, 将  $P_2$  中对应位置的基因值置为 0。

(3) 分别对  $P_1$ 、 $P_2$  中的基因值为 0 的基因, 通过左移集中到交叉区域, 而其他非 0 基因的相对次序不变。

(4) 将  $P_1$  交叉区域的内容替换为  $T_2$  的内容, 将

$P_2$  交叉区域的内容替换为  $T_1$  的内容。至此,将分别得到两个子代个体 Child1 和 Child2。

现有两个父本个体  $P_1(1\ 1\ 3\ 5\ 2\ 4\ 1\ 5\ 1\ 4\ 2)$  和  $P_2(1\ 1\ 2\ 1\ 1\ 5\ 4\ 2\ 5\ 4\ 3)$ ,若随机选择的基因座  $i = 4$ ,则交叉后得到的两个子代个体分别为 Child1(1 1 2 1 3 5 4 5 1 4 2) 和 Child2(1 1 3 5 2 1 1 4 2 5 4)。

#### 2.4 可调节的变异概率

遗传算法容易陷入局部优化,此时应以较大的变异概率使算法跳出局部极值点。所以变异概率  $P_m$  的选择应该具有某种可调节性。建立一个变异概率  $P_m$  与遗传代数的函数,第  $i$  代的遗传概率  $P_i$  为:

$$P_i = P_{\min} + (P_{\max} - P_{\min}) * i / \text{Popsiz}e$$

式中  $P_{\max}$  是设定的最大变异率,  $P_{\min}$  是设定的最小变异率,  $\text{Popsiz}e$  是设定的遗传代数。

这样,在运算前期变异率较低,有很强的全局搜索能力;在中后期变异率较高,可以防止局部优化,陷入早熟。

#### 2.5 变异操作

在染色体上随机选取两点,将这两点之间的基因进行倒位操作。如有父体染色体(2 2 1 3 2 1 3 3 1 3 1),随机选取第 2 位到第 6 位进行倒位操作,变异后产生的新个体为(2 1 2 3 1 2 3 3 1 3 1)。这样得到的新个体中工件个数不变,所以生成的新染色体是合法的。

#### 2.6 循环选择

JSP 问题中,某些优化的工件与机床组合会在变异中被破坏,使得在很接近最优解的时候往往还要花费很多资源进行计算,因此进化过程中要尽可能地保存已经形成的优化组合。在此采用一种循环选择的方法有较好的效果。对于种群中新生成的每一个染色体个体,让它们首尾相连形成环状,依次选择其中的基因作为开始,又可生成新的染色体。计算并选择其中适应度高的  $m$  个染色体做为新一代的种群。

#### 2.7 算法步骤

(1) 确定初始种群(群体规模为  $m$ ),随机产生  $m$  个  $1 - n$  的全排列,即  $m$  个染色体。

(2) 计算初始群体中每个个体的适应度值。

(3) 繁殖运算,对上一代群体中的每一个个体进行遗传操作,产生  $m$  个新个体。

(4) 计算  $m$  个新个体的适应度值。因要使所求目标函数作业加工时间(makespan)最小,所以适应度函数可以

表示为:  $F(x) = k - \text{makespan}(x)$ ,  $k$  为一个适当的相对比较大的数。

(5) 从上一代群体中的  $m$  个个体和本次遗传操作产生的  $m$  个新个体所组成的  $2m$  个个体中,选择适应度较大的  $m$  个个体组成新群体。再对每个新个体进行循环选择,得到适应度最大的  $m$  个个体组成下一代新群体。

(6) 若满足终止条件则停止,否则转到(3)。

### 3 模拟分析

针对参考文献[10]中的具体实例,采用文中改进的遗传算法,用 VB 编程进行比较现有 6 工件/5 机床 JSP 问题如下:

工件 1 (1,1,1,2) (1,2,2,3) (1,3,4,4)  
(1,4,1,6)  
工件 2 (2,1,5,7) (2,2,5,8) (2,3,2,3)  
(2,4,3,9) (2,5,1,4)  
工件 3 (3,1,2,6) (3,2,4,7) (3,3,1,1)  
(3,4,5,2) (3,5,4,7) (3,6,1,9)  
工件 4 (4,1,3,1) (4,2,2,10) (4,3,1,2)  
工件 5 (5,1,2,13) (5,2,3,15) (5,3,4,1)  
(5,4,3,1)

工件 6 (6,1,2,3) (6,2,4,3) (6,3,5,6)

括号内第一个数字代表工件号,第二个数字代表工件工序号,第三个数字代表本工序所在机床号,第四个数字代表该工序的加工时间。

使用与参考文献相同的参数:种群规模为  $M = 50$ ,进化代数  $\text{Popsiz}e = 20$ ,交叉概率为  $P_c = 0.6$ ,一般情况下,变异概率的取值范围是  $0.01 \sim 0.1$ ,则第  $i$  代的变异概率  $P_m$  为  $P_i = 0.01 + (0.1 - 0.01) * i / \text{Popsiz}e$ ,经过 20 代进化,参考文献[10]中所得最优目标函数值为 52,而文中的改进遗传算法所求得的最优目标函数值为 48(见图 1、图 2)。

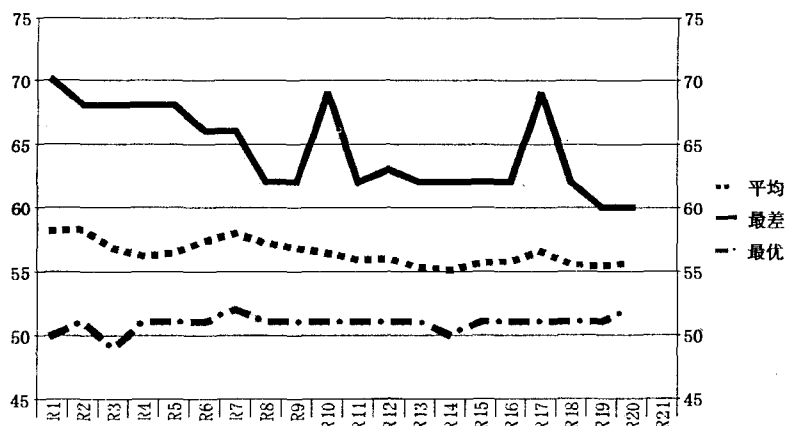


图 1 标准遗传算法收敛曲线

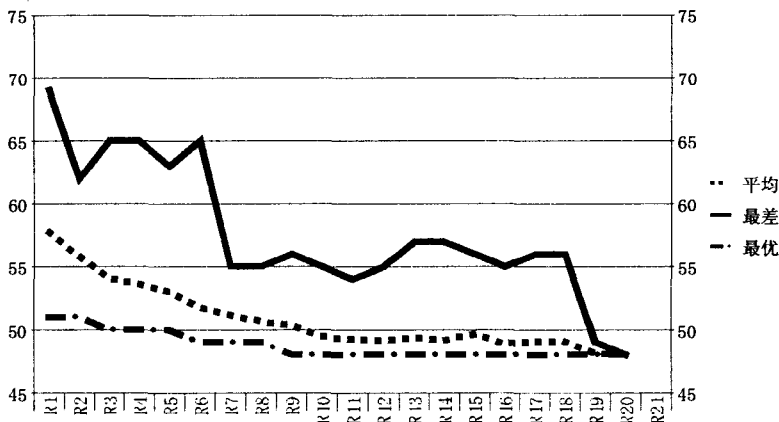


图 2 改进遗传算法收敛曲线

表 1 实例的机器加工过程表

机器序号	工序	加工起 始时间	加工结 束时间
M <sub>1</sub>	(1,1,1,2)	0	2
	空闲		
	(3,3,1,1)	22	23
	空闲		
	(1,4,1,6)	26	32
	(3,6,1,9)	33	42
	(4,3,1,2)	42	44
M <sub>2</sub>	(2,5,1,4)	44	48
	(6,1,2,3)	0	3
	(1,2,2,3)	3	6
	(3,1,2,6)	6	12
	空闲		
	(2,3,2,3)	15	18
	(5,1,2,13)	18	31
M <sub>3</sub>	(4,2,2,10)	31	41
	(4,1,3,1)	0	1
	空闲		
	(2,4,3,9)	18	27
	空闲		
	(5,2,3,15)	31	46
	空闲		
M <sub>4</sub>	(5,4,3,1)	47	48
	空闲		
	(6,2,4,3)	3	6
	空闲		
	(2,2,4,8)	7	15
	(3,2,4,7)	15	22
	(1,3,4,4)	22	26
M <sub>5</sub>	(3,5,4,7)	26	33
	空闲		
	(5,3,4,1)	46	47
	(2,1,5,7)	0	7
	空闲		
	(6,3,5,6)	7	13
	空闲		
	(3,4,5,2)	23	25

通过收敛曲线图的对比可以看出,图 1 中标准遗传算法在第 10 代就已经陷入了局部最优,出现早熟现象;图 2 中改进的遗传算法三条曲线均成均匀收敛

趋势,并没有陷入局部优化。对比结果表明,改进后的算法可以在较短时间内得到较优解。(13323532415562 13643245126)、(223265553 4132311136423546)、(3221 363425131552362651344)、(463222135331225131365 5644)、(6142213235115332 634354562)等都是可以得到最优解的染色体。(614221323511533 2634354562)对应的机器加工过程

见表 1。

#### 4 结束语

针对标准遗传算法收敛速度慢和易陷入局部最优的问题,文中在总结已有经验的基础上,采用了基于工序的编码、解码的方式,在进行遗传操作后,采用循环选择并保留最优个体,在交叉和变异概率的计算方法等方面提出了一系列改进方法,在优化性能上有了很大提高,实例结果证明了该改进算法的有效性。作业车间调度是复杂的组合优化问题,需要考虑的因素非常多,对求解 JSSP 的遗传算法进行改进还有以下几点建议:

(1)遗传算法的局部搜索能力不强,因此可以将局部搜索的方法引入到遗传算法中的重组和选择部分。在这种混合遗传算法中,遗传算法主要进行种群中的全局搜索,局部搜索方法则负责染色体间的局部搜索,二者互补,可以获得更好的优化效果。

(2)遗传算法对初始种群很敏感,因此若用启发式的方法生成性能良好的初始种群,将比随机生成的初始种群有更好的优化效果。

(3)为防止遗传陷入局部优化,采用可调节的变异概率  $P_m$  时,还需要注意适当保持全局搜索和局部搜索之间的平衡,这样才能尽快收敛到全局最优解。

#### 参考文献:

[1] 唐恒永,赵传立. 排序引论[M]. 北京:科学出版社,2002: 81-82.  
[2] Garey M, Johnson D, Serhi R. The complexity of flow shop and job shop scheduling[J]. Mathematics of Operations Research, 1976(1): 117-129.  
[3] Blazewicz J, Ecker K H, Schmidt G, et al. Scheduling in Computer and Manufacturing Systems[M]. Second Revised Edition. Berlin: Springer-Verlag, 1996.  
[4] 王书锋,邹益仁. 车间作业调度(JSSP)技术问题简明综述

表 2 镜像数据库初始化步骤

发送端步骤	接收端步骤
1. 设置数据库故障还原模型为“非简单模式”	1. 接收端程序启动, 侦听网络
2. 禁用跟踪触发器	2. 接收文件一
3. 清空跟踪记录表记录	3. 接收文件二
4. 截断数据库日志	4. 获取镜像数据库的排他访问权
5. 完整备份数据库至文件一	5. 从文件一还原数据库(此时数据库处于加载状态)
6. 设立事务标记 EnableTrigger, 启用跟踪触发器	6. 从文件二还原日志到指定事务标记 EnableTrigger
7. 备份数据库日志至文件二	7. 删除镜像数据库的所有触发器、约束
8. 关闭数据库连接	8. 设置数据库故障还原模型为“简单模式”
9. 连接接收端	9. 启动同步模块接收端
10. 传输文件一	10. 向发送端返回镜像数据库初始化成功标志(信号 0)
11. 传输文件二	
12. 等待接收端返回镜像数据库初始化成功标志(信号 0)	
13. 启动同步模块发送端	

上述步骤省略了错误处理,如果运行成功的话,对于 1G 数据量的数据库整个过程耗时约八分钟左右,在同步模块启动前,跟踪记录表的记录数约有几千至几万条,由于采用了事务标记还原技术,上述过程保证了初始的镜像数据库加上跟踪记录表的信息与源数据库中的对应信息一致。对此过程的几点说明如下:

(1) 因为网闸的存在,笔者采用了整数来标志状态,如数字 0 表示镜像数据库初始化成功,确保从接收端返回的状态信息不会超过四个字节。

(2) 由于需要做日志备份,所以需要源数据库开启故障还原模型为完全,而镜像数据库在必要的时候可以由源数据库重新初始化,因而镜像数据库的故障还原模型为简单,以提升镜像数据库的性能。

(3) 数据库中的数据一致性、完整性、正确性完全由源数据库来保证,各同步表的信息均各自同步更新,所以必须要删除镜像数据库中的所有触发器、约束。

如果该过程中出现错误,如网络错误、数据库连接错误等,则需要人工排除错误,重新启动该模块。

### 3.5 同步模块的设计

同步模块是该系统的另一个核心模块,分为发送端和接收端两部分,发送端定时地从源数据库中的跟

踪记录表中提取跟踪信息,经 SQL 语句优化模块、压缩模块处理,交给发送模块打包发往接收端,接收端在收到数据包后先进行解包,解压缩,然后在镜像数据库上执行,成功后返回状态信息,发送端在接收到镜像服务器端执行成功的信息后,删除已提取部分的跟踪信息记录,再定时从跟踪记录表中提取新的跟踪信息,如此周而复始完成镜像数据库数据的增量同步。

### 4 小 结

根据文中的设计思路,笔者主导实施了某电厂的二次系统安全防护项目,整个同步系统对源数据库中的运行几乎没有影响,而且因为将源数据库的查询应用转移到了镜像服务器上,源数据库的性能在某些情况下甚至会比原来有所提升。镜像数据库中的数据时延最低可为 3 秒左右,完全达到了电厂远动数据的应用需要,经受了每分钟上万条记录同步的考验,运行良好。另外本系统配置灵活,模块划分清晰,可扩展性好,如经过对触发器生成模块和镜像数据库初始化模块的简单修改可以使本系统适用 Oracle 等其他数据库。由于应用的实际情况,本系统在设计时并未考虑数据库字段类型为 ntext, binary, varbinary, image 的情况。

#### 参考文献:

[1] Liu Peng, Jajodia S, McCollum C D. Intrusion confinement by isolation in information systems[J]. Journal of computer security, 2000(7): 23-28.

[2] 代卫星, 章 俊. 物理隔离在电力调度专用网中的应用[J]. 广东电力, 2006(7): 28-30.

[3] 肖永田, 章 军. 基于内存交换的网闸系统的研究与实现[J]. 计算机工程与应用, 2005(36): 140-142.

[4] 刘 伟, 佟俐鹏. 异构数据库集成中的变化捕获方案设计[J]. 计算机应用研究, 2005(7): 213-215.

[5] 黄晓微, 陈 玲, 魏 伟, 等. 基于快照日志分析的数据同步方法[J]. 后勤工程学院学报, 2006(2): 59-62.

(上接第 174 页)

[J]. 系统工程理论与实践, 2003(1): 49-55.

[5] 方红雨, 崔逊学. 基于遗传算法的调度问题研究[J]. 电脑与信息技术, 2001(2): 1-5.

[6] Hart E, Ross P. A systematic investigation of GA performance on job shop scheduling problems[J]. Lecture Notes in Comp Sci, 2000, 1803: 277-286.

[7] Adams J, Balas E, Zawack D. The shifting bottle neck procedure for job shop scheduling[J]. Management Science, 1988, 34(3): 391-401.

[8] 玄光男, 程润伟. 遗传算法与工程设计[M]. 北京: 清华大学出版社, 2000.

[9] Cheng R, Gen M, Tsujimura Y. A tutorial survey of job-shop scheduling problems using genetic algorithms - representation[J]. Computers & Industrial Engineering, 1996, 30(4): 983-997.

[10] 谢胜利, 黄 强, 董金祥. 求解 JSP 的遗传算法中不可行调度的方案[J]. 计算机集成制造系统, 2002(11): 902-906.