用遗传算法与自适应神经网络混合方法解 Job-shop 调度问题[1]

杨圣祥　　汪定伟
东北大学 信息科学与工程学院 系统工程系 沈阳 110006

摘要　本文提出了一种用遗传算法结合基于约束满足的自适应神经网络进行 Job-shop 调度问题求解的混合方法。在混合方法中，遗传算法用来进行迭代寻优，迭代中当前代经交叉和变异后生成的染色体对应非可行解，由自适应神经网络运算后得到可行解，对应的染色体作为新一代染色体。仿真表明本文提出的混合求解算法是快速有效的。

关键词　Job-shop 调度问题，遗传算法，约束满足，自适应神经网络

1．引　　言

Job-shop 调度问题是指由 m 台机器加工 n 个有特定加工路线的工件，其目标是确定每台机器上各工件工序的加工顺序和开工时间，使某个性能指标最优，如最小化最大完工时间[1]。由于 Job-shop 调度是典型的 NP 难题，寻找其最优解非常困难，人们开始采用各种启发式算法寻找其次优解，以满足实际问题的需要[2]。Foo S.Y.最早提出用神经网络求解 Job-shop 调度问题[3]，其后又有人对此问题进行了研究[4,5]，取得非常好的成效，近年来遗传算法在 Job-shop 调度问题上也得到了应用[6]。本文提出一种用遗传算法结合基于约束满足的自适应神经网络（CSANN）混合方法来求解 Job-shop 调度问题。遗传算法用来迭代寻优，CSANN 用来在遗传算法迭代过程中求得可行解。仿真表明了混合算法的有效性。

2．问题描述

设 $N=\{1,\cdots,n\}$，$M=\{1,\cdots,m\}$，n_i 是工件 i 的工序数，O_{ikq} 表示工件 i 第 k 道在机器 q 上加工的工序，其开始加工时间和加工时间（加工时间已知）记为 S_{ikq} 和 T_{ikq}，S_{ie_iq}、T_{ie_iq} 分别是工件 i 最后一道工序的开始加工时间和加工时间，r_i、d_i 分别是工件 i 的投料和交货期限，P_i 是工件 i 的有序工序对[O_{ikp}，O_{ilq}]集合，其中 O_{ikp} 优先于 O_{ilq}，R_q 是机器 q 上所有工序的集合。以最小化最大完工时间（即制造周期）为指标 E，则问题数学模型为：

$$\min \quad E=\max_{i\in N}(S_{ie_iq}+T_{ie_iq})$$

$$s.t. \quad S_{ilq}-S_{ikp}\ge T_{ikp} \qquad [O_{ikp},O_{ilq}]\in P_i,\ k,l\in\{1,\cdots,n_i\},i\in N \qquad (1)$$

$$S_{jlq}-S_{ikq}\ge T_{ikq}\ or\ S_{ikq}-S_{jlq}\ge T_{jlq} \qquad O_{ikq},O_{jlq}\in R_q,\ i,j\in N,\ q\in M \qquad (2)$$

$$r_i\le S_{ijq}\le d_i-T_{ijq} \qquad i\in N,\ j\in\{1,\cdots,n_i\},q\in M \qquad (3)$$

其中，（1）式表示同一工件的不同工序不能同时加工，即工序顺序约束；（2）式表示每台机器同一时刻只能加工一个工件，即资源约束；（3）式表示投料和交货期限约束。

3．约束满足自适应神经网络模型

CSANN 是由在通用神经元基础上，分别定义的表示工序开始加工时间的 S 类神经元、工序顺序约束是否满足的 SC 类神经元和资源约束是否满足的 RC 类神经元组成，这三类神经元通过相互连接组成工序顺序约束－SC 模块和资源约束－RC 模块。SC 模块的每个单元

[1] 该研究由国家自然科学基金(No.69684005)和国家 863 计划 CIMS 主题(No.863-511-9609-003)共同资助

由一个 SC 类、两个 S 类神经元组成（见图 1），用来判断同一工件的工序顺序约束是否得到满足，RC 模块的每个单元由一个 RC 类、两个 S 类神经元组成（见图 2），用来判断资源约束是否得到满足，当工序顺序约束或资源约束条件不满足时，通过适当的反馈调节来消除约束冲突。通用神经元由线性加权求和函数与非线性函数 $f(\bullet)$ 级联组成，定义如下：

$$O_i = f(I_i) = f(\sum_{j=1}^{n}(W_{ij} * O_j) + B_i) \tag{4}$$

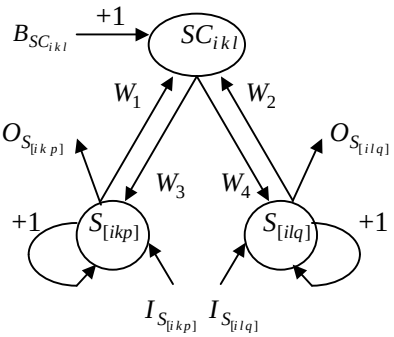

图 1　SC 模块单元

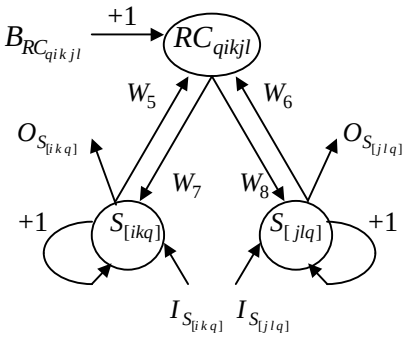

图 2　RC 模块单元

S 类神经元用于约束各工件的工序加工活动开始时间，其定义如下：

$$I_{Si}(t+1) = \sum_j (W_{ij} * O_{SCj}(t)) + \sum_k (W_{ik} * O_{RCk}(t)) + O_{S_i}(t) \tag{5}$$

$$O_{S_i}(t+1) = \begin{cases} r_i & I_{S_i}(t+1) \le r_i \\ I_{S_i}(t+1) & r_i \le I_{S_i}(t+1) \le d_i - T_{S_i} \\ d_i - T_{S_i} & I_{S_i}(t+1) \ge d_i - T_{S_i} \end{cases} \tag{6}$$

其中，（5）式右边的第一、第二项分别是相关的 SC 类和 RC 类神经元的反馈调节作用，第三项是 S 类神经元 S_i 上一时刻的输出，（6）式中 T_{S_i} 是 S_i 所对应工序的加工时间。

SC 类、RC 类神经元采集 S 类神经元的输出，分别用来判断（1）式表示的工序顺序约束、（2）式表示的机器资源约束是否满足，定义形式相同，如下所示：

$$I_{C_i}(t) = \sum_j (W_{ij} * O_{S_j}(t)) + B_{C_i} \tag{7}$$

$$O_{C_i}(t) = \begin{cases} 0 & I_{C_i}(t) \ge 0 \\ -I_{C_i}(t) & I_{C_i}(t) \le 0 \end{cases} \tag{8}$$

其中，$C_i = SC_i$ 或 RC_i，B_{C_i} 是神经元 SC_i 或 RC_i 的偏置设定（见图 1 和图 2）。

图 1 中，$[O_{ikp}, O_{ilq}] \in P_i$，各连接权值与偏置的取值如下所示：

$$W_1 = -1, \quad W_2 = 1, \quad W_3 = -W, \quad W_4 = W, \quad B_{SC_{ikl}} = -T_{ikp} \tag{9}$$

图 2 中，O_{ikq}、$O_{jlq} \in R_q$，各连接权值与偏置视网络运行过程中 t 时刻具体情况自适应取定，当 $S_{ikq}(t) \le S_{jlq}(t)$ 时，由（10）式确定，当 $S_{ikq}(t) \ge S_{jlq}(t)$ 时，由（11）式确定：

$$W_5 = -1, \quad W_6 = 1, \quad W_7 = -W, \quad W_8 = W, \quad B_{RC_{qikjl}} = -T_{ikq} \tag{10}$$

$$W_5 = 1, \quad W_6 = -1, \quad W_7 = W, \quad W_8 = -W, \quad B_{RC_{qikjl}} = -T_{jlq} \tag{11}$$

上述各式中，W 是正的可调参数（例如 0.5），用来控制反馈调整的强度。

为了消除 CSANN 运行过程中可能出现的“死锁”现象，保证得到可行解，对同一机器上的相邻工序，满足一定条件时，互换其开始加工时间以互换其顺序，即网络运行 t 时刻，机器 q 上加工的工序 O_{ikq} 和 O_{jlq} 已经连续调整其开工时间的次数达到设定常数 H 时，取

$$S_{ikq}(t+1) = S_{jlq}(t), \quad S_{jlq}(t+1) = S_{ikq}(t) \tag{12}$$

针对实际的 Job-shop 调度问题，CSANN 的求解步骤：

步骤 1：建立 CSANN 模型：确定 S 类神经元的数目 $\sum_{i=1}^{n} n_i$，为每个 S 类神经元对应工序 O_{ikq} 编号为 $S_{[ikq]}$；确定 P_i 和 R_q 集合，构造 SC 模块和 RC 模块；设定 W、H 的值；

步骤 2：初始化各工序开始加工时间 $I_{S_{[ikp]}}$，作为相应的 S 神经元的初始输出；

步骤 3：运行 SC 模块各单元，按（7、8、9）式计算 $O_{SC_{ikl}}(t)$，若 $O_{SC_{ikl}}(t) \neq 0$，表示对应的工序顺序约束不满足，则按（9）式和（13）式修改 $S_{ikp}(t+1)$ 和 $S_{ilq}(t+1)$；

$$S_{ikp}(t+1) = S_{ikp}(t) + W_3 * O_{SC_{ikl}}(t), \quad S_{ilq}(t+1) = S_{ilq}(t) + W_4 * O_{SC_{ikl}}(t) \qquad (13)$$

步骤 4：运行 RC 模块各单元，按（7、8、10 或 11）式计算 $O_{RC_{qikjl}}(t)$，若 $O_{RC_{qikjl}}(t) \neq 0$，表示对应的资源约束不满足，则按（10 或 11）式和（14）式或（12）式修改 $S_{ikq}(t+1)$ 和 $S_{jlq}(t+1)$，重复步骤 3 和步骤 4，直至得到可行调度解。

$$S_{ikq}(t+1) = S_{ikq}(t) + W_7 * O_{RC_{qikjl}}(t), \quad S_{jlq}(t+1) = S_{jlq}(t) + W_8 * O_{RC_{qikjl}}(t) \qquad (14)$$

4． 遗传算法及混合算法描述

4.1 遗传算法描述

本文所采用的遗传算法主要要素有：

● 染色体编码方式：采用自然码排列进行染色体编码，染色体长度为总的工序数。每个染色体由一组子染色体组成，每个子染色体对应一台机器。每个子染色体由一组代表该机器上加工的工序所属的工件序号的自然数排列组成，子染色体长度为在该台机器上加工的工序数，基因的位置排序代表该台机器上工件的加工次序，例如某子染色体为“645312”表示该台机器上工件的加工次序为：6 →4→ 5 →3 →1→ 2。

● 适值函数：用一个足够大的正数（例如 10000）减去目标函数（即最大完工时间）作为染色体的适应值，染色体的适应值越大，对应的解的性能指标越好，即制造周期越小。

● 选择策略：采用回转轮法和精英选择策略，精英选择是指在新一代种群中任意选一个染色体用上一代中最好的染色体代替。在第 K 代中第 i 条染色体的选取概率为：

$$P_K(i) = \frac{f_K(i) - f_K}{\sum_{j=1}^{PN}[f_K(j) - f_K]}, \quad f_K = \min\{f_K(i),\ i = 1, \cdots, PN\} \qquad (15)$$

其中，$f_K(i)$ 为第 K 代中第 i 条染色体的适应值，PN 为种群规模。

● 遗传算子：交叉算子采用 PMX（Partial Mapped Crossover）和 UX（Uniform Crossover）交叉算子；变异算子采用 CM（Converse Mutation）逆序变异算子、RSM（Right Shift Mutation）右移变异算子和 EM（Exchange Mutation）互换变异算子。

在遗传过程中，交叉或变异都是在子染色体中进行，以确保得到的子代染色体有意义。在子染色体的基因位置进行交叉或变异而发生改变时，基因所对应的工序的开工时间也随着作相应的改变，例如某子染色体经遗传（交叉或变异）处理前为“346512”，其对应的各工序开工时间为 0、2、5、9、11、14，经遗传处理后变为“314652”，其对应的各工序开工时间为 0、2、5、9、11、14。这样由代表可行解的染色体经遗传运算后，得到的染色体所对应的解不一定可行，此时把得到的非可行解作为 CSANN 的初始状态来运行，求得新的可行解。由新的可行解在每台机器上把各工序对应的工件号按工序开工时间由小到大顺序转换成子染色体，例如某机器上各工序开工时间满足：$S_{J_1} \leq \cdots \leq S_{J_i} \leq \cdots \leq S_{J_n}$，$J_i \in \{1, \cdots, n\}, i \in \{1, \cdots, n\}$，$S_{J_i}$ 是工件号为 J_i 对应的工序的开始加工时间，则对应的子染色体为：$J_1 \cdots J_i \cdots J_n$，这样所得的染色体作为新一代染色体。

● 停止准则：本文采用最大迭代代数—MG 作为遗传算法的停止准则。

4.2 混合算法描述

整个混合 Job-shop 调度算法是由遗传算法和 CSANN 组成，遗传算法被用来迭代寻优，迭代中当前代经交叉和变异后生成的染色体对应非可行解，由自适应神经网络运算后得到可行解，对应的染色体作为新一代染色体。混合调度算法的基本步骤为：

步骤 1：设定参数值：*PN*、*MG*、交叉概率 P_C 和变异概率 P_M；

步骤 2：随机产生 *PN* 个染色体作为初始种群，每个初始染色体对应的初始解如下求得：设某台机器对应的子染色体为 $J_1\ J_2 \cdots J_n$，$J_i \in \{1,\cdots,n\}, i \in \{1,\cdots,n\}$，$J_i$ 表示该台机器上第 i 个工序所属的工件号，则该台机器上各工序的开工时间为：$S_1 = 0$，$S_{i+1} = S_i + T_i$，其中 S_i 和 T_i（$i \in \{1,\cdots,n-1\}$）是该机器上第 i 个工序的开工时间和加工时间。所得到的初始解作为 CSANN 的初始状态进行运算，求得可行解，并把它所对应的染色体作为第一代染色体；

步骤 3：根据适值函数计算每条染色体的适应值；

步骤 4：按照每条染色体的适应值计算出其选取概率，并按照选取概率的大小，随机地选取 *PN* 个染色体进行自我复制；

步骤 5：随机选取的染色体在交配池中进行交叉和变异运算，并把生成的染色体对应的非可行调度解，用神经网络求得可行解，求得的可行解转换得到新一代的染色体，并随机地选择一条染色体用上一代中最好的染色体代替；

步骤 6：判断是否满足停止准则，如果满足则运算结束，否则返回步骤 3。

5． 仿真实验结果

我们以一个 6x6 Job-shop 调度问题为例（问题的最小制造周期是 55），在 PC 586/133 上 VC++ 5.0 开发环境下进行了仿真研究。表 1 给出了只用 CSANN 进行求解的 100 个实验的统计结果，其中第一个实验是在零初始条件下进行的，即所有工序初始开工时间都设为零，而其他 99 个实验中，初始解的各工序初始开工时间取[0,100]之间的均匀分布。仿真中 CSANN 的参数设置为：$H = 5$ 和 $W = 0.5$。表 1 中程序运行时间为 0 秒表示不到 1 秒。各工件投料时间设为 0，即在 0 时刻所有工件都已释放或可加工。完工时间约束是事先设定的，分别为较大的正数 300 和近优值 58，仿真中看作是所有工件的交货期约束。

完工时间约束	工序初始开工时间设定	制造周期 E			运行时间（秒）		
		平均	最小	最大	平均	最小	最大
300	0	76			1		
300	随机产生	105	89	117	1	0	1
58	0	58			50		
58	随机产生	58	58	58	48	25	97

表 1　只用 CSANN 求解的实验统计结果

交叉算子	PMX				UX			
变异算子	制造周期（目标值）			运行时间（秒）	制造周期（目标值）			运行时间（秒）
	平均	最小	最大		平均	最小	最大	
CM	64	60	72	137	62	58	75	114
RSM	69	63	82	150	66	61	81	135
EM	67	61	82	145	65	60	80	130

表 2　用具有不同遗传算子的混合调度算法求解的结果比较

表 2 是用不同遗传算子的遗传算法和 CSANN 混合算法进行求解的结果，遗传算法参数设置为： $PN=20$ 、 $MG=200$ 、 $P_C=1$ 和 $P_M=0.4$ ，CSANN 的参数设置为： $H=5$ 和 $W=0.5$ ，各工件投料时间设为 0，完工时间约束事先设定为较大的数 500（等效于无穷大）。

6. 结　论

仿真表明，针对 Job-shop 调度问题，只用 CSANN 求解时，对大多数的实际问题能够快速得到可行调度解，但性能指标严重依赖于最大完工时间约束的设定。约束设定较松时，得到的解的性能指标较差；约束设定较合理时，得到的解的性能指标较好；约束设定太紧，甚至小于最优值时，得不道可行解。用遗传算法和 CSANN 混合算法进行求解时，CSANN 的完工时间约束可以设定得非常宽松或为无穷大，用不同的遗传算子一般都能得到较好的可行调度解或近优解，其中采用 UX 交叉算子较 PMX 交叉算子为好，采用逆序变异算子较其他两种算子效果为好。仿真结果表明本文提出的求解 Job-shop 调度问题的混合算法是有效的。

参 考 文 献

[1] Conway R. W., *Theory of Scheduling*, Reading Mass: Addison-Wesley ,1967

[2] French, S., *Sequencing and scheduling: An introduction to the mathematics of the Job-shop*, New York: Wiley, 1982

[3] Foo S. Y. and Takefuji Y., Integer-linear programming neural networks for job-shop scheduling, *Proc. IEEE IJCNN'88*, San Diago, 341-348, 1988

[4] Willems T. M. and Brandts L. E. M. W., Implementing heuristics as an optimization criterion in neural networks for job-shop scheduling, *J. of Intelligent Manufacturing*, **6**, 377-387, 1995

[5] D. N. Zhou, V. Charkassky, T. R. Baldwin and D. W. Hong, "Scaling neural network for job-shop scheduling," *Proc. IEEE IJCNN'89*, New York, **3**, 889-894, 1989

[6] Federico Della Croce, Roberto Tadei and Giuseppe Volta, Agenetic algorithm for the job-shop problem, *Computers and Operations Research*, **22(1),** 15-24, 1995

Genetic algorithm and adaptive neural network hybrid method for job-shop scheduling problems

Yang Shengxiang　　Wang Dingwei

Abstract　This paper proposes a hybrid method of genetic algorithm (GA) and constraint satisfaction adaptive neural network (CSANN) for solving job-shop scheduling problems. In the hybrid method GA is used to iterate for searching optimal solutions, CSANN is used to solve feasible solutions during the iteration of GA. Computer simulations have shown the good performance of the proposed hybrid method for job-shop scheduling problems.

Keywords　job-shop scheduling, genetic algorithm, constraint satisfaction, adaptive neural network

作 者 简 介

杨圣祥　男，1972 年 5 月生。1996 年于东北大学获工业自动化专业工学硕士学位，现在东北大学信息科学与工程学院系统工程系攻读博士学位。主要研究神经网络、智能优化和智能调度等。

汪定伟　男，1948 年 11 月生。东北大学博士，曾在美国北卡罗来拉州立大学作博士后。现为东北大学教授、博士导师。中国自动化学会管理与系统专业委员会委员、《控制与决策》杂志编委、国家 863 计划 CIMS 主题 09 专题专家。主要研究生产计划与调度理论、建模与决策、智能优化方法。