第21卷 第6期 自 动 化 学 报 Vol. 21, No. 6
1995 年 11 月 ACTA AUTOMATICA SINICA Nov., 1995

解 Job-shop 调度问题的神经网络方法[1)]

张长水 阎平凡
(清华大学自动化系 北京 100084)

摘 要

研究用神经网络方法解决 Job-shop 调度问题. 首先描述解 Job-shop 调度问题的算法，然后给出这一算法及其网络性质的理论结果. 仿真实验结果证明了该方法是可行的. 最后，针对几类典型调度问题的解决进一步说明了这一方法的优势.

关键词：组合优化，神经网络，调度问题.

1 引言

调度问题的一个简化形式就是：有M台机器及N个工件，根据工件的加工工艺要求、每个工件使用M台机器的顺序及每道工序所需时间已知，如何安排每台机器上工件的加工顺序，使得某个指标(如总的完工时间)最小. 当N个工件的加工路线不要求相同时，该问题被称为 job-shop 调度问题. 最初人们致力于寻找问题的最优解[1,2]，但遇到了很大的困难[3]. 之后，开始研究问题的次优解，因为实际上好的次优解也可以满足要求. 特别是使用神经网络模型解优化问题，引起研究人员很大兴趣[4]. 本文就是采用神经网络方法解 job-shop 调度问题.

2 方法

2.1 问题的数学描述

假设s_{ik}和t_{ik}分别表示工件i的第k个操作起始时间和加工时间. 花费函数是所有工件最后一个操作的起始时间之和$\sum_{i=1}^{N} s_{ik_i}$，这里N是工件数，k_i是工件的最后一个操作，所有变量s_{ik}的数值满足下列不等式：

$$s_{ik} - s_{ik+1} + t_{ik} \leqslant 0, \quad i = 1, \cdots, N, k = 1, \cdots, k_i - 1; \tag{1}$$

$$s_{i1} \geqslant 0, \qquad i = 1, \cdots, N; \tag{2}$$

$$s_{ik} - s_{jp} + t_{ik} \leqslant 0 \text{ 或 } s_{jp} - s_{ik} + t_{jp} \leqslant 0. \tag{3}$$

---

1) 本项目得到中国科学院自动化所国家模式识别实验室开放课题基金的支持.
本文于 1993 年 6 月 4 日收到.

其中式(1)表示任何一个工件只能在加工完前一道工序以后，才能加工后一道工序；式(2)表示所有工件的第一道工序起始时间要大于 0；式(3)中的两个操作 O_{ik}，O_{ip} 是在同一机器上的两个操作，它表示对于同一机器上的两个操作 A 与 B，要么先加工完 A 以后再加工 B，要么先加工完 B 以后再加工 A．调度问题就是在式(1)，(2)，(3) 约束下的最小化 $\sum_{i=1}^{N} s_{iki}$，这就构成了 Job-shop 调度问题的一种数学描述．

在给出的神经网络模型中，s_{ik} 表示神经元，网络的能量函数为

$$E = \sum_i s_{iki} + \sum_i \sum_k H_1 \cdot F_1(s_{ik} - s_{ik+1} + t_{ik}) + \sum_i H_2 \cdot F(-s_{i1})$$
$$+ \Sigma\Sigma H_3 \cdot \min(F_1(s_{ik} - s_{jp} + t_{ik}), F_1(s_{jp} - s_{ik} + t_{jp}))．\qquad (4)$$

这里 H_1, H_2 和 H_3 是参数；

$$F_1(x) = \begin{cases} e^x & x > 0 \\ 0 & \text{else}. \end{cases}$$

2.2 算法 A．

对于由式(4)定义的神经网络，神经元 s_{ik} 按下面顺序方式演变：

$$s_{ik}(t+1) = \begin{cases} s_{ik}(t) - \Delta, & \dfrac{\partial E}{\partial s_{ik}} > 0; \\ s_{ik}(t) + \Delta, & \dfrac{\partial E}{\partial s_{ik}} < 0; \\ s_{ik}(t), & \dfrac{\partial E}{\partial s_{ik}} = 0. \end{cases}$$

其中 E 是式(4)给出的表示，Δ 是一个正的常量．

3 算法 A 的理论结果

对算法 A 进行了理论分析，它主要包括对算法收敛性、解的有效性和网络的吸引子的研究．收敛性是算法能够实现的基本保证．对于调度问题，解的有效性是指解在实际中的可执行性[1)]．而网络的吸引子则直接影响到算法的使用．因此，这些理论分析是重要的．由于分析过程要使用一些定义、概念和结论(其证明较复杂)，限于篇幅，在此只给出三个重要结论，其中细节，请参阅有关文章[1)]．

定理 1(收敛性定理)．算法 A 是收敛的．

定理 2(解的有效性定理)．算法 A 的解是有效解．

定理 3(吸引子定理)．算法 A 中网络的平衡状态与调度问题的有效解是一一对应的．

4 算法 B

算法 A 保证了上述三个定理，使网络收敛到了局部极小值，但在解的最优性方面却有

1) 张长水．用神经网络方法解决作业调度问题的研究，清华大学博士论文，1992．

所牺牲. 尽管这个局部极小值已经比较令人满意(参见图 1),但仍希望得到更好的解.算法 B 可以做到这一点.

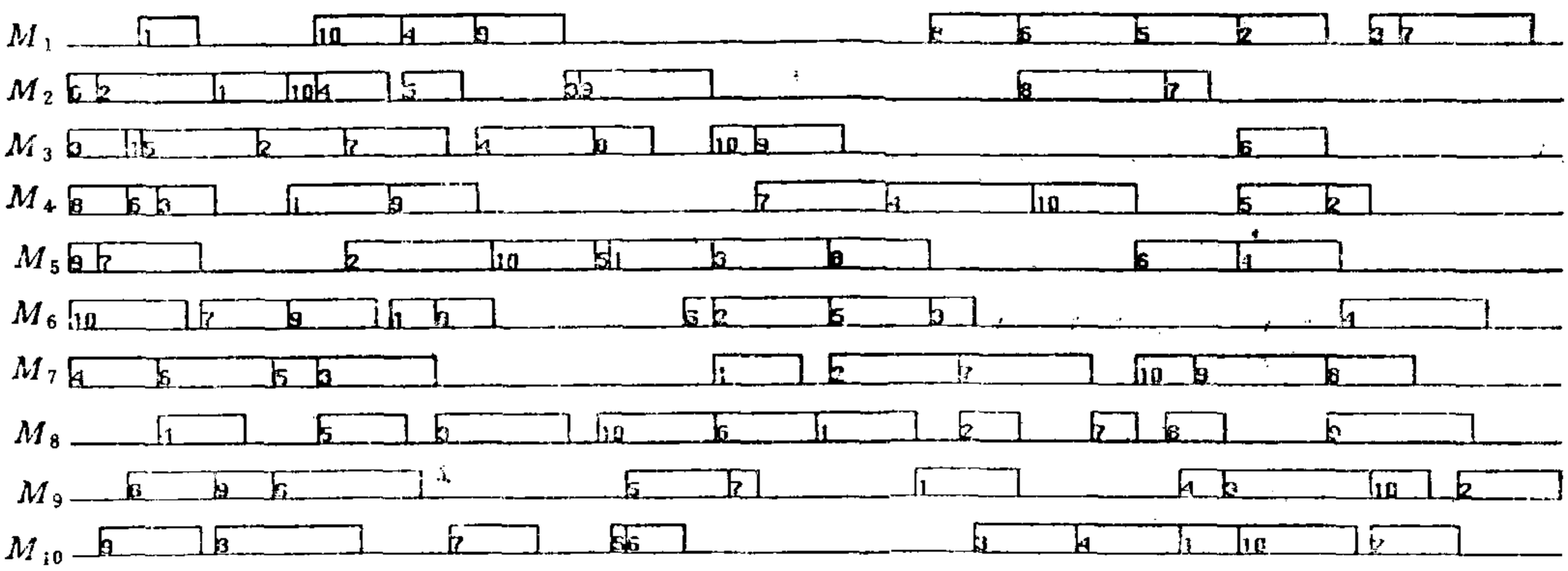

图 1 算法 A 的调度结果

算法 B 的思路是这样的,对于算法 A 给出的一个解,依次将每个工件的后一道工序的起始时间重新赋值为前一道工序的完工时间. 这样, 让网络在一个可能是新的解空间中再次演化,网络稳定后,将算法 A 的这一结果与上一次结果相比较,保留更好的;然后继续这一过程,直至最后不能再得到更好的解. 下面给出具体算法.

算法 B.

Step 1. 使用算法 A,得到一个解,并计算其总的加工时间 total-time;

Step 2. 指定当前要考虑改变状态的神经元为 s_{jp};

Step 3. 令 $s_{jp} = s_{jp-1} + t_{jp-1}$;

Step 4. 使用算法 A 再调度,得到一个新解;

Step 5. 计算新解的总加工时间 new-total-time;

Step 6. 如果 new-total-time < total-time,则

1) 新解作为当前解;

2) 下一个神经元作为当前要考虑改变状态的神经元 s_{jp};

3) go to Step 3;

Step 7. 如果算法未结束, 则令下一个神经元作为当前要考虑改变状态的神经元 s_{jp},go to step 3;

Step 8. 结束.

在算法 B 的 Step 7 中,判断算法结束的标准是连续 L 次没有得到更优解, 其中 L 是网络中神经元个数. 事实上,由于问题存在最小解,所以存在时刻 t ,在 t 时刻之后,网络将不会得到更优解. 而当网络连续 L 次没有得到更优解之后, 算法又周而复始重复前 L 次过程. 所以上述判断算法结束标准是好的.

5 实验模拟结果

对于大规模的调度问题,不仅解决时很困难,就是对其调度结果的观察和显示也是困

难的．因此，实验数据不容易获得．这里对文献[5]给出的图进行测量之后，得到了 10 台机床加工 10 个工件调度问题的最初数据．图 1 和图 2 分别是算法 A 和算法 B 的调度结果甘特图．图中方块里 i 表示加工第 i 个工件．图 1 的总加工时间是 4615，图 2 的总加工时间是 3995．

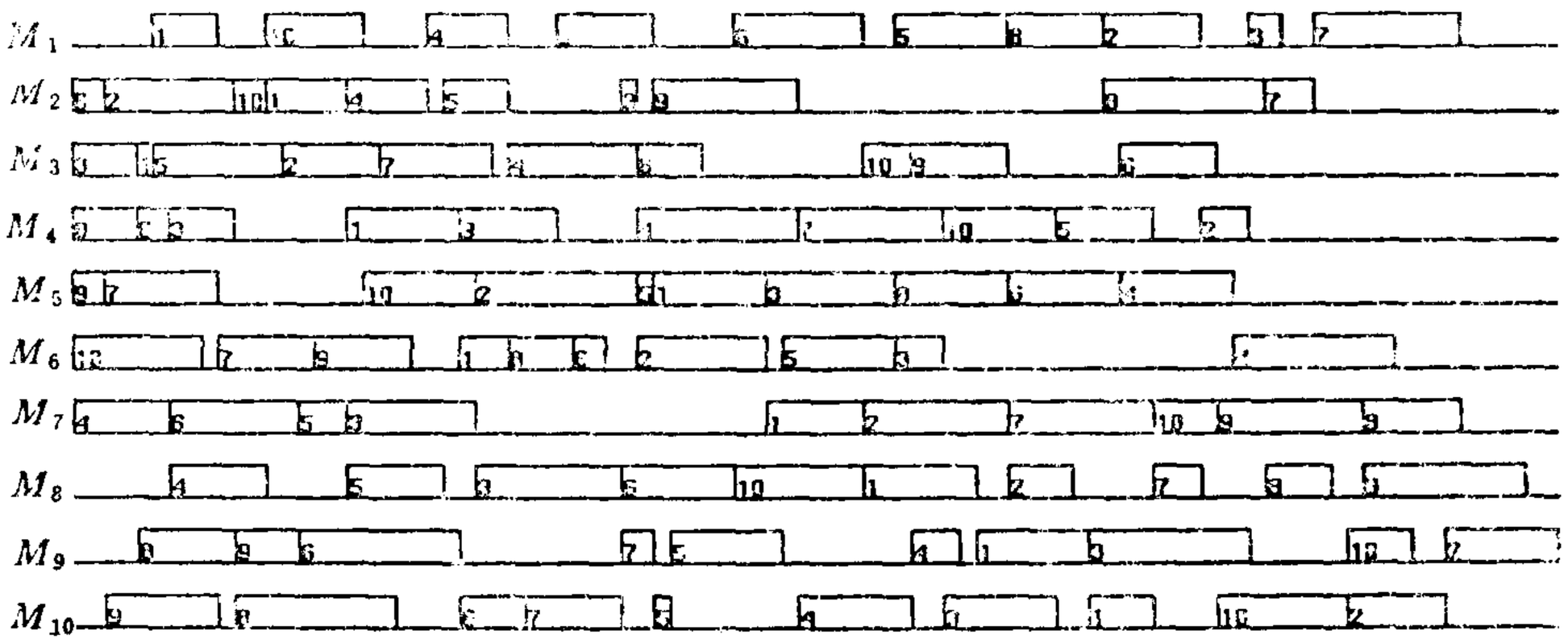

图 2　算法 B 的调度结果

6　对典型问题的解决

作业调度问题在实际应用中的困难，不仅表现为它是一个 NP 难题，而且由于研究人员对调度问题的高度提炼和简化也导致了它难以适用于复杂多变的实际场合．因此，一种好的调度方法应该能够适用于各种类型的调度问题．在实际应用的调度问题中，有优先级存在、交换期限存在和自动小车存在的几种调度问题，被认为是最重要和困难的．目前已有的调度方法难以解决这类问题，而用本文提出的方法则可以解决．

6.1　有优先级的调度

由于各个工件的急需程度和重要程度不同，因此有些工件就必须得到优先安排，这样就构成了优先顺序的约束．如果需要加工的一批工件都具有相同的优先级，那么所有的工件被同等看待，这样就无所谓优先顺序的存在．因此优先顺序约束的 Job-shop 调度问题一定是几部分工件之间具有优先级的差异存在．这种差异表现为，一批是特急任务，必须优先安排加工；而另一批是次急任务，被安排次优先加工；对于一般任务，优先级较低，安排在最后加工．由此看来，这一类问题是有着很强的实际背景的．

由于优先级高的工件具有绝对的优先加工权，因此采用下面方案是合理的：

1）首先对优先级最高的工件使用算法 A 调度；

2）逐次对较低级的工件使用算法 A．在对于较低级的工件排序时，不改变已经调度好的较高级工件的加工时间．

假定 d_i 表示第 i 个工件的优先级（d_i 取正数，d_i 越大，表示工件 i 的优先级越高），则式

$$E = \sum_i s_{ik_i} + H_1 \times \sum_i F_1(s_{ik} - s_{ik+1} + t_{ik}) + H_2 \times \sum_i F_1(-s_{i1})$$

$$+ H_3 \times \Sigma\Sigma\min(F_1(s_{ik} - s_{jp} + t_{ik}) \times F_3(d_j - d_i), F_1(s_{jp} - s_{ik} + t_{jp}) \times F_3(d_i - d_j))$$

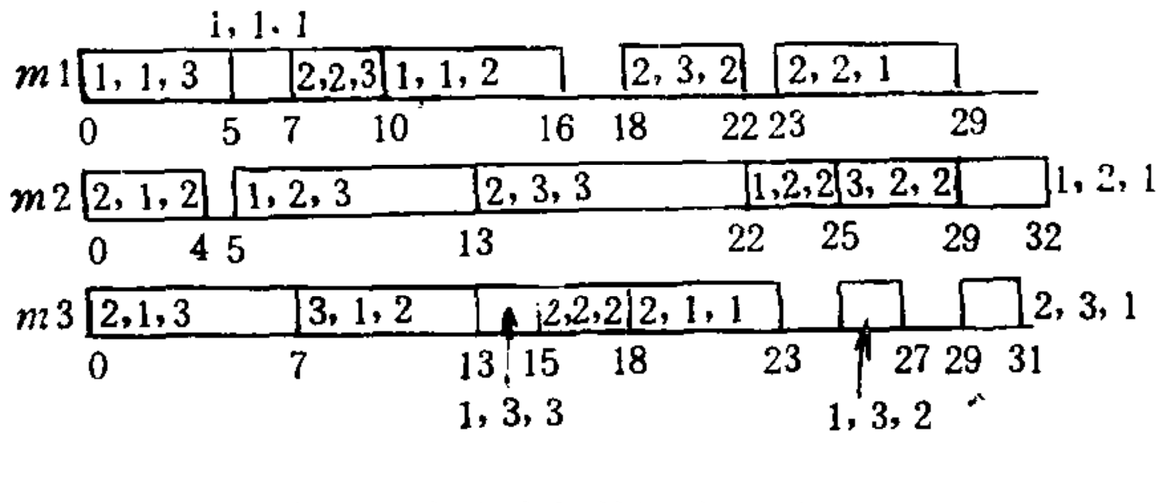

图 3 有优先级的调度结果

是这一问题所对应网络的能量函数，其中

$$F_3(x) = \begin{cases} c & x > 0, \\ 1 & \text{else}, \end{cases}$$

c 表示一个大的正常数；能量函数中第四项则反映了工件优先级的加工限制。图 3 给出了解这类问题的仿真结果，其中 (i,k,d) 表示优先级为 d 的工件 i 的第 k 个操作。

6.2 有交货期限要求的调度问题

生产总是与交货期限 (due date) 相联系的，这一类型的问题是调度问题中最难的问题之一。解决这一问题的一个想法，就是具有较早交货期限的工件应该有较高的优先级。但这一想法不尽合理。事实上，只有快到交货期限时，需要交货的工件才表现出较高的优先级。而在最初的加工时间里，一般来说，所有工件是被同等看待的。因此，这种情况下，工件的优先级是时间的函数，也就是说，工件的每一个操作由于完工时间不同，因而所对应的时间的函数是肯定不同的。所以，必须要知道一个排序结果，根据这一结果确定每个工件的每个操作的完工期限。根据这一想法，下面给出解决有交货期限调度问题的算法。

算法 C.

Step 1. 按交货期先后顺序确定工件的优先级，交货期早的工件具有较高优先级；

Step 2. 置每个工件的交货期限无穷大；

Step 3. 令 d_0 为最高优先级；

Step 4. 用神经网络方法调度；

Step 5. 确定每台机器上优先级大于等于 d_0 的工件的加工顺序，并根据交货期限确定这些工件的每一个操作的最晚开工时间。已确定了最晚开工时间的工件，其优先级函数就已确定；

Step 6. 再调度，再调度结果将使得优先级高于 d_0 的工件满足交货期限；

Step 7. 如果 d_0 是倒数第 2 级，则结束；

Step 8. 置 d_0 为下一优先级，go to Step 4.

在 Step 4 和 Step 6 中，神经网络的能量函数第四项为

$$H_3 \times \Sigma\Sigma\min(F_1(s_{ik} - s_{jp} + t_{ik}) \times F_3(L_{ik} - s_{ik}), F_1(s_{jp} - s_{ik} + t_{jp}) \times F_3(L_{jp} - s_{jp})).$$

其中

$$F_3(x) = \begin{cases} c & x > 0, \\ 1 & \text{else}, \end{cases}$$

c 是一个大的正常数；L_{ik} 记为 i 工件第 k 个操作的最晚开工时间。

图 4(a),(b) 分别给出没有交货期限和有交货期限的仿真调度结果.

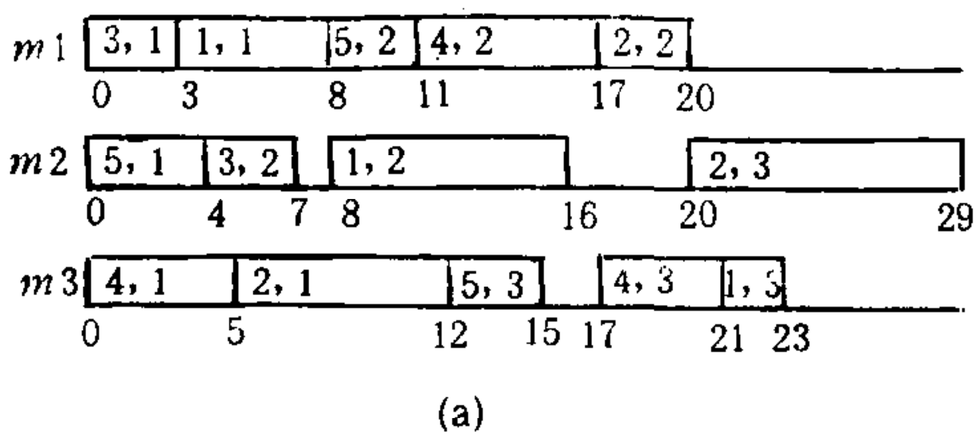

图 4

6.3 有自动活动车辆的调度

在高度自动化的工厂中,自动活动车辆 (automated moving vehicle) 越来越多地用于机器之间工件、材料的传送. 尤其是在计算机集成制造系统 (CIMS) 中，自动活动车辆更是必不可少. 怎样调度活动车辆使得机器的工作效率更高，历来被认为是一个十分困难的问题.

在本文的方法中，活动车辆被看成一台“机器”. 车辆将工件 i 从机床 A 运送到机床 B 的时间看成工件 i 在活动车辆这台“机床”上加工所需的时间. 假定工件在两台机床之间一定要由活动车辆传送,那么原来工件 i 的加工路线(表 1(a)所示)就变成了表 1(b)所示的加工路线,其中 $M_i(i=1,2,3)$ 表示机床，M_4 表示活动车辆. 这样,问题就变成了一般的 Job-shop 调度问题. 但问题远没有这么简单,事实上，车辆在运送工件时的时间花费是依赖于顺序的. 也就是说,当工件 i 要从机床 A 运送到机床 B 时,车辆首先要从其当前所在位置(机床 C)返回到机床 A,然后再运送 i 工件. 因此,如果假定小车先运送工件 i 从机床 A 到机床 B. 再运送工件 j 从机床 C 到机床 D,那么应有

$$s_i + d(\mathrm{A},\mathrm{B}) + d(\mathrm{B},\mathrm{C}) - s_j \leqslant 0,$$

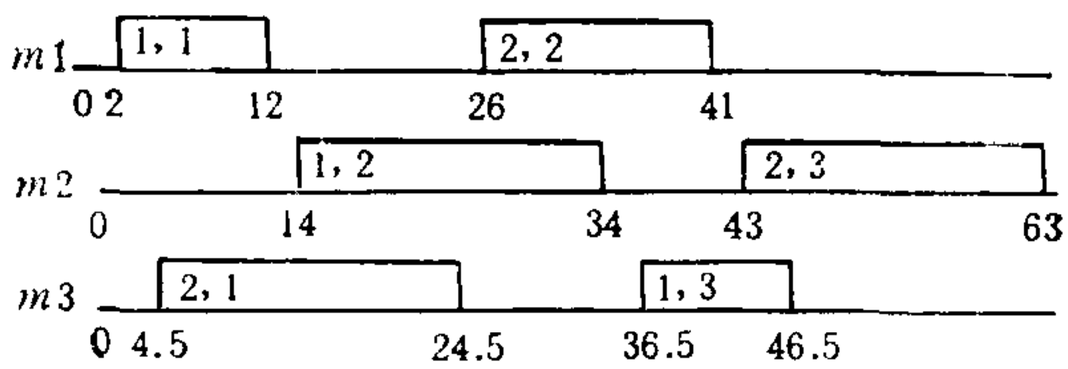

图 5　有活动车辆的调度结果

表　1(a)

M_1	M_2	M_3

表　1(b)

M_4	M_1	M_4	M_2	M_4	M_3	M_4

表　2

	仓库	M_1	M_2	M_3
仓库	0	2	4	2.5
M_1	2	0	2	1.5
M_2	4	2	0	2.5
M_3	2.5	1.5	2.5	0

其中 s_i, s_j 分别表示车辆运送工件 i, j 的起始时间; $d(\mathrm{A},\mathrm{B})$, $d(\mathrm{B},\mathrm{C})$ 分别表示车辆从机床 A 到 B 和从机床 B 到 C 的时间花费. 这样,有活动车辆这一限制就可以表示如下:

$$\Sigma\min(F_1(s_{ik}-s_{jp}+t_1+t_2),F_r(s_{jp}-s_{ik}+t_3+t_4)).$$

表 2 给出对各机器间的路程车辆所行走的时间表，图 5 给出仿真调度结果.

致谢： 感谢国家模式识别实验室全体人员的热情帮助和资助.

参 考 文 献

[1] 韩继业. 排序问题的一个判别条件和一类特殊的$m\times n$排序问题. 应用数学学报，1980，(4).

[2] Rodammer F A, Whit K P. A recent survey of production scheduling. *IEEE Trans. on System, Man and Cybern.* 1968, **18**(6): 841—851.

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

[4] Hopfield J J, Tank D W. Neural computation of decisions in optimization problems. *Biological Cybernetics*, 1985, **52**(3): 141—152.

[5] Zhou D N etc. Scaling neural network for Job-shop scheduling. Proc. IJCNN'90, Washington: Dec., 1990, **3**: 889—892.

NEURAL NETWORK METHOD OF SOLVING JOB-SHOP SCHEDULING PROBLEM

ZHANG CHANGSHUI YAN PINGFAN

(*Department of Automation, Tsinghua University, Beijing* 100084)

ABSTRACT

The improved methods for Job-shop scheduling problem (JSP) with neural network are described. And the theorem of the convergence of the networks, the theorem of the efficiency of the solutions and the theorem of the attractors of the networks are given. Then computer simulation result shows that our methods are good. In the end, three kinds of scheduling problems —— JSP with priority, JSP with due dates and JSP with automated moving vehicle —— are solved by our methods. The corresponding simulations are successfully done.

Key words: Neural network, combinatorial optimization, scheduling problem.

张长水 1986 年毕业于北京大学数学系，获学士学位. 1992 年获清华大学自动化系模式识别与智能控制专业博士学位. 现任清华大学自动化系副教授，目前从事模式识别、图像处理和人工智能方面的研究工作.

阎平凡 简介及照片见本刊第 19 卷第 4 期.