生产批量计划问题的RCWW算法验证研究

韩 毅[1] 蔡建湖[1] 周根贵[1] 李延来[2] 缪卫南[1]
(浙江工业大学经贸管理学院 杭州 310023)[1]
(东北大学流程工业综合自动化教育部重点实验室 沈阳 110004)[2]

摘 要 Wagner-Whitin(WW)算法是经典的、求解生产批量计划(Lot-sizing Planning,LSP)问题的最优启发式算法,对于中小规模问题可以有效求得产品的最优生产量。随机累加WW(Randomized Cumulative WW,RCWW)算法是改进了的WW算法,适用于求解具有一般生产结构的、多层级LSP问题。RCWW算法的求解效果已经得到了验证。根据RCWW算法的求解思想,通过采用C语言进行编码实现算法流程。通过对具有一般生产结构LSP问题的标准算例进行求解,验证了RCWW算法的求解效果,发现了原文献的错误,证明了作者对RCWW算法的正确理解。

关键词 Wagner-Whitin算法,生产批量计划问题,随机累加,一般生产结构,多层级,智能优化算法
中图法分类号 TP29 **文献标识码** A

Research on Verification of RCWW Algorithm for Lot-sizing Planning Problem

HAN Yi[1] CAI Jian-hu[1] ZHOU Gen-gui[1] LI Yan-lai[2] MIAO Wei-nan[1]
(College of Economics and Management, Zhejiang University of Technology, Hangzhou 310023, China)[1]
(Key Laboratory of Integrated Automation of Process Industry of Ministry of Education, Northeastern University, Shenyang 110004, China)[2]

Abstract Wagner-Whitin (WW) algorithm is a classical optimization heuristic algorithm for lot-sizing planning (LSP) problem. It can provide the best production volumes of a product effectively for medium- and small-sized problems. Randomized cumulative WW (RCWW) algorithm is a modified WW algorithm, which is very suitable for solving LSP problem with general structure and multiple levels. The performance of RCWW algorithm was proved before. Based on the executive idea of RCWW algorithm, this paper adopted C programming language to implement RCWW algorithm. Through computation on LSP problems with general structure, the effects of RCWW algorithm were verified. Also, the errors from literature were found out and our understanding on RCWW algorithm was proved to be correct.

Keywords Wagner-Whitin algorithm, Lot-sizing planning problem, Randomized cumulative, General structure, Multiple levels, Intelligent optimization algorithm

1 引言

在企业供应链的生产环节部分,生产批量计划(LSP)问题是物料需求计划(Material Requirements Planning,MRP)/制造资源计划(Manufacturing Resources Planning,MRPII)系统的关键问题。根据产品的物料清单(Bill of Materials,BOM)及产品的外部需求,LSP能够确定在有限的计划期内产品在每个时期的最佳生产数量。虽然在当今社会中,许多企业采用功能强大的企业资源计划(Enterprise Resources Planning,ERP)软件(如SAP,BAAN,Oracle9i,Fourth Shift,用友及金蝶等软件)来制定企业生产计划,但是其核心部分仍然是基于MRP和MRPII系统(MRP与MRPII的区别是MRP不考虑企业内可用资源的限制和约束,而MRPII需要考虑各种资源对生产活动的限制)。在现有的所有ERP软件中,制定生产批量计划仅仅是基于最简单的或次优的方法,MRPII仅仅起到预警作用而不是决策支持作用(因为关于生产过程的约束方面的数据是很多企业不愿意花大力气去收集的)[1]。生产批量计划的制定在很多企业中仍然是基于MRP和BOM的,关于无资源约束LSP问题的求解算法方面的研究,仍然具有重要的科学意义[1,2]。

根据层级划分,LSP问题可以分为单级和多级问题。多级问题需要根据产品的BOM来确定每种物料的最佳生产数量。根据复杂性,多级批量计划问题可以分为线性结构问题、装配结构问题和一般结构问题。

从历史发展的角度看,Harris[3]在1913年提出的经济订购批量(Economic Order Quantity,EOQ)方法是最早提出的算法,EOQ平衡了生产准备费用(setup cost)与库存费用(inventory keeping cost)。文献[4]在1958年,提出了求解单级

到稿日期:2010-07-28 返修日期:2010-12-05 本文受国家自然科学基金(70971017),浙江省自然科学基金(Y1100854),浙江省教育厅研究项目(Y201016979),浙江省科技厅软科学研究资助项目(2009C35007),教育部人文社会科学研究项目(10YJC630009),浙江省哲学社会科学规划课题(10CGGL21YBQ)资助。

韩 毅(1979—),男,讲师,主要研究方向为生产计划与调度、智能优化算法等,E-mail:hanyi@zjut.edu.cn;周根贵(1958—),男,教授,博士生导师。

无资源约束 LSP 问题的基于动态规划的著名精确解算法——WW 算法，其依据的是“存在满足条件 $I_{t-1} * X_t = 0$ 的最优解”这个基本性质(简称 WW 性质)，这里 I_{t-1} 是上一个时期的期末库存量(假设没有安全库存)，X_t 是当前时期的生产数量。在随后的研究中，有很多学者提出了求解单级 LSP 问题的启发式和智能优化算法[5-9]。单级问题只是研究以独立产品为单位的计划问题，1968 年，Schussel[10] 讨论了产品结构更加复杂的、基于 BOM 的线型结构 LSP 问题，同时 MRP 系统在生产企业中被广泛接受，极大地促进了多级 LSP 问题的研究。随后很多学者在求解多级 LSP 问题方面进行了广泛的研究[11]。虽然有资源约束多级 LSP 问题比无资源约束多级 LSP 问题更加符合实际情况，但是由于有资源约束多级 LSP 问题的求解要先根据无资源约束多级 LSP 问题的求解工具得到结果，然后进行各时期的资源约束调整，所以关于无资源约束多级 LSP 问题的文献多于有资源约束多级 LSP 问题的文献。目前，关于无资源约束多级 LSP 问题的有效求解工具方面的研究仍然具有很大的研究价值。

纵观所有求解无资源约束多级 LSP 问题的求解算法，文献[2]在 2003 年提出的 RCWW 算法是一种效果很好的启发式算法，它按照一定规则对具有一般产品结构的 BOM 上的所有物料逐一采用 WW 算法进行求解。文献[1]在 2007 年提出的改进 RCWW 算法(RCWW-STVS)是目前为止效果最好的启发式算法，其求解结果已非常接近已知最好解，通过与一些智能优化算法相结合，可以求得一些中规模和大规模问题的最优解。

本文通过采用 C 语言编程实现 RCWW-STVS 算法，对文献[1]给出的求解结果进行验证和比对，具有 3 个方面的研究意义：1) 验证了文献[1]结果的正确性；2) 验证了作者对文献[1]算法理解的正确性；3) 由于隐私性方面的原因，LSP 问题的研究学者无法得到文献[1]算法的源代码，通过自主编程实现了算法，为进一步开发 LSP 问题的有效求解算法奠定了坚实的基础和有力保障。

2 生产批量计划问题数学模型

无资源约束多级 LSP 问题模型中，假设生产提前期为零，并且不允许缺货。总成本由生产准备费用和库存费用组成。

参数定义：

T	计划期的长度；
X_{it}	在第 t 时期物料 i 的生产数量；
I_{it}	第 t 时期末物料 i 的库存量；
Y_{it}	如果物料 i 在第 t 时期进行生产则值为 1，否则值为 0；
d_{it}	在第 t 时期物料 i 的需求数量；
C_{ij}	生产一个单位物料 j 所需要的物料 i 的数量；
l_i	第 i 种物料的生产提前期；
s_i	物料 i 的生产准备费用；
$M_{it}=\sum_{k=1}^{T} d_{ik}$	在第 t 时期物料 i 的生产量的上限；
h_i	物料 i 的库存保管费用；
N	物料总数量；
$\Gamma(i)$	物料 i 的直接后继的集合；
$\Gamma^{-1}(i)$	物料 i 的直接前趋的集合。

模型：

$$\text{Min. } Z=\sum_{i=1}^{N}\sum_{t=1}^{T}(h_i I_{it}+s_i X_{it}) \tag{1}$$

Subject to

$$X_{it}+I_{i,t-1}-I_{it}=d_{it},\ i=1,\cdots,N,t=1,\cdots,T \tag{2}$$

$$d_{it}=\begin{cases} d_{it}, & \text{if } \Gamma(i)=\phi \\ \sum_{j\in\Gamma(i)} C_{ij}X_{j,t+lj}, & \text{else} \end{cases} \tag{3}$$

$$X_{it}\leqslant M_{it}Y_{it},\ i=1,\cdots,N,t=1,\cdots,T \tag{4}$$

$$Y_{it}\in\{0,1\},\ i=1,\cdots,N,t=1,\cdots,T \tag{5}$$

$$X_{it},I_{it}\geqslant 0,\ i=1,\cdots,N,t=1,\cdots,T \tag{6}$$

目标函数式(1)表示总费用包括生产准备费用及库存费用；约束(2)表示物流守衡方程；约束(3)表示需求公式，如果物料 i 是最终项目，则需求为外部需求，否则需求由当前物料 i 的所有后继物料的生产量决定；约束(4)表示只有在生产数量大于 0 时才能进行生产调整；约束(5)表示二进制决策变量，1 表示生产，0 表示不生产；约束(6)表示生产量和库存量不能为负数。

3 无资源约束 LSP 问题的 3 种启发式

3.1 WW 算法[4]

WW 算法是基于动态规划的、求解单级无资源约束 LSP 问题的最有效算法。为帮助其他学者更好地理解 WW 算法，WW 算法的执行过程将以算例形式进行解释和阐述。表 1[4] 给出了某个产品在 12 个生产时期内各个时期的外部需求量，假设产品的生产准备费用在各个时期都是 54，库存费用在各个时期都是 0.4，第 1 个时期的期初库存量为零。

表 1 某产品生产计划期和需求量

月份	1	2	3	4	5
需求量	10	62	12	130	154
月份	6	7	8	9	10
需求量	129	88	52	124	160
月份	11	12			
需求量	238	41			

定义 $F(t)$ 为产品最好生产策略的总成本，定义 $f(t)$ 是产品最好生产策略的二进制决策向量。

在第 1 个时期，必然要进行生产才能至少保证满足产品需求，所以这个时期的最优生产决策 $f(1)=(1)$，生产费用 $F(1)=54$。在第 2 个时期，为了确定 $f(2)$ 和 $F(2)$，面临 2 个选择：第 1 个选择是在第 1 个时期生产 10 个产品，在第 2 个时期生产 62 个产品(这种情况下的生产决策 $f(2)=f(1)+1=(1,1)$，生产费用 $F(2)=F(1)+54=108$)；第 2 个选择是在第 1 个时期生产 72 个产品、存储 62 个产品，在第 2 个时期不生产(这种情况下的生产决策 $f(2)=f(1)+0=(1,0)$，生产费用 $F(2)=F(1)+62*1*0.4=78.8$)。这里很容易看出，第 2 种选择是最优的，生产费用最小。

在第 3 个时期，为了确定 $f(3)$ 和 $F(3)$，面临 3 个选择：第 1 个选择是在第 1 个时期的生产量足够满足 2 个月的需求量，在第 3 个时期生产 12 个产品(这种情况下的生产决策 $f(3)=f(2)+1=(1,0,1)$，生产费用 $F(3)=F(2)+54=132.8$)；第 2 个选择是在第 1 个时期的生产量足够满足 1 个月的需求量，在第 2 个时期生产量足够满足 2 个月的需求量

(这种情况下的生产决策 $f(3)=f(1)+1+0=(1,1,0)$，生产费用 $F(3)=F(1)+54+12*1*0.4=112.8$)；第 3 个选择是在第 1 个时期的生产量足够满足 3 个月的需求量(这种情况下的生产决策 $f(3)=1+0+0=(1,0,0)$，生产费用 $F(3)=54+62*2*0.4+12*1*0.4=88.4$)。因此可以看出，第 3 种选择是最优的，生产费用最小。算法递归进行求解，最后得到 12 个时期的最小生产成本 $F(12)=501.2$，最佳的生产决策向量 $f(12)=(1,0,0,1,1,0,1,0,1,1,1,0)$，所有计算过程中的中间结果由表 2[4] 给出。

表 2　WW 算法的中间结果

月份	1	2	3	4	5
需求量	10	62	12	130	154
选择 1	**54**	108	132.8	244.4	490.8
选择 2		**78.8**	112.8	216.8	401.6
选择 3			**88.4**	184.8	308
选择 4				**142.4**	204
选择 5					**196.4**
月份	6	7	8	9	10
需求量	129	88	52	124	160
选择 1	748.8	960	1105.6	1502.4	2078.4
选择 2	608	784	908.8	1256	1768
选择 3	462.8	603.6	707.6	1005.2	1453.2
选择 4	307.2	412.8	496	744	1128
选择 5	**248**	318.4	380.8	579.2	899.2
选择 6	250.4	**285.6**	327.2	476	732
选择 7		302	**322.8**	422	614
选择 8			339.6	389.2	517.2
选择 9				**376.8**	440.8
选择 10					**430.8**
月份	11	12			
需求量	238	41			
选择 1	3030.4	3210.8			
选择 2	2624.8	2788.8			
选择 3	2214.8	2362.4			
选择 4	1794.4	1925.6			
选择 5	1470.4	1585.2			
选择 6	1208	1306.4			
选择 7	994.8	1076.8			
选择 8	802.8	868.4			
选择 9	631.2	680.4			
选择 10	526	558.8			
选择 11	**484.8**	**501.2**			
选择 12		538.8			

3.2　RCWW 算法[1]

RCWW 算法是改进的 WW 算法，适用于求解多级无资源约束 LSP 问题。图 1[1] 给出了具有一般产品结构的 BOM，其中物料左边带下划横线的数字表示产品生产提前期，物料右边的数字表示生产准备费用，物料之间的需求关系是 1 对 1 的。WW 算法在执行时，需要根据确定的生产准备费用和库存费用来制定生产决策和确定生产成本。RCWW 算法是基于修改的虚拟生产准备费用和原始库存费用来执行的，具体的执行过程分为 5 个步骤：1)根据 BOM 来确定一个计算物料生产决策的先后顺序；2)确定当前物料在每个时期修改后的虚拟生产准备费用；3)基于当前物料的虚拟生产准备费用和库存费，使用 WW 算法逐时期从前到后进行计算，得到虚拟修改费用下的最优的 0-1 生产决策向量；4)采用原始的生产准备费用(未被修改的生产准备费用)，根据 0-1 生产决策向量计算得到生产成本；5)将第 2)－4)步骤循环进行，直到所有的 BOM 上的物料都被计算过为止，最后输出最优的生产量矩阵和总生产成本。

RCWW 算法是一种基于修改的虚拟费用的 WW 算法，采用修改的虚拟费用的思想是源于“在某个时期的物料生产量，会导致生产该物料所需的所有物料都必须进行生产的结果”。根据图 1 来看，如果在某个生产时期 t 要生产物料 1，那么就需要在第 $t-1$ 个时期生产物料 3、4 和 6(都是物料 1 的前驱)。因此可以看出，物料 1 的生产导致了物料 3、4 和 6 的生产，物料 1 需要承担一部分物料 3、4 和 6 的生产准备的责任，责任的体现以累加随机虚拟生产准备费用的方式来实现。

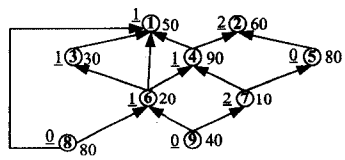

图 1　一般结构产品 BOM 图

在生产物料 1 时，需要考虑 2 种情况：1)物料 3、4 和 6 在生产时期 $t-1$ 已经因其他物料的生产而导致生产准备活动的发生(在这种情况下，物料 1 不需要承担该生产时期其所需所有物料的生产准备责任)；2)物料 3、4 和 6 在生产时期 $t-1$ 没有因其他物料的生产而导致生产准备活动的发生(在这种情况下，物料 1 需要承担该生产时期其所需所有物料的生产准备责任)。物料的虚拟生产准备费用需要根据式(7)来计算。在给出式(7)之前，需要解释几个变量。S_{it} 表示物料 i 在时期 t 修改后的生产准备费用；$T(i,j,t)$ 表示二进制 0－1 变量，当其值为 1 表示物料 i 在 $t-l_i$ 时期引发了物料 j 产生准备活动，其值为 0 表示在 $t-l_i$ 时期已经有其他物料引发了物料 j 产生准备活动；r 表示值为大于等于 0 和小于等于 1 之间一个随机数；$|\Gamma(i)|$ 表示所有需要物料 i 来进行生产的物料(所有物料 i 的后继)的个数：

$$S_{it}=s_i+r\sum_{j\in\Gamma^{-1}(i)}T(i,j,t)\frac{S_j}{|\Gamma(j)|} \tag{7}$$

式中，S_j 需要按照式(8)来递归求得：

$$S_i=s_i+\sum_{j\in\Gamma^{-1}(i)}\frac{S_j}{|\Gamma(j)|} \tag{8}$$

式(7)中的 S_j 是过渡变量；变量 $T(i,j,t)$ 不是事先确定好的，而是需要根据物料生产准备费用的修改顺序来确定，比如第 1 个能够进行生产费用修改的物料只能是物料 1 或物料 2。变量 $T(i,j,t)$ 的引入就是用来控制某物料引发了其前驱物料的生产准备活动后是否需要承担相应责任。变量 r 用来控制责任的承担量，也就是说物料不必承担引起其他物料生产准备活动的全部责任。对于每种物料，变量 r 的值固定不变。

3.3　RCWW-STVS 算法[1]

这个方法是 RCWW 算法的一个修改版本，通过对每种物料分配不同的 r 值来计算修改后的生产准备费用。r 值根据式(9)来确定：

$$r_i=R\left(1+\frac{P-2\Phi_i+1}{P-1}u\right) \tag{9}$$

式中，r_i 是第 i 种物料的生产准备费用累加系数，R 表示大于等于 0、小于等于 0.5 的小数，P 表示物料总数，$\Phi_i\in\{1,\cdots,P\}$ 表示物料 i 在物料序列中的位置，u 表示大于等于－1 和小于等于 1 之间的实数。

注意到对于物料顺序中的第一个物料(产品,$\Phi_i=1$),根据式(9)可以得到 $r_i=R(1+u)$;对于物料顺序中的最后一个物料(生产材料,$\Phi_i=P$),根据式(9)可以得到 $r_P=R(1-u)$。因此,R 是 r_i 的平均值;u 确定了斜率,修改的生产准备费用累加式如式(10)所示:

$$S_u=s_i+r_i\sum_{j\in\Gamma^{-1}(i)}T(i,j,t)\frac{S_j}{|\Gamma(j)|} \tag{10}$$

根据 u 的符号,可以区分两种情况:

1)如果 u 是正数,r_i 的值随产品序号的增加而减小;2)如果 u 是负数,r_i 的值随产品序号的增加而增加。

表 3 $R=0.431183$ 和 $u=0.956399$ 时各物料修改后的生产准备费用和生产量

i	s_i	r_i		1	2	3	4	5	6	7	8	9
2	60	0.84	$S_{2,t}$						195.7	195.7	195.7	195.7
		(0.43)							(129.4)	(129.4)	(129.4)	(129.4)
			X_{2t}						50	60		30
									(50)	(40)	(20)	(30)
5	80	0.74	$S_{5,t}$				91.1	91.1	91.1	91.1	91.1	91.1
		(0.43)					(86.5)	(86.5)	(86.5)	(86.5)	(86.5)	(86.5)
			X_{5t}				50	60		30		
							(50)	(60)		(30)		
1	50	0.64	$S_{1,t}$					128.6	128.6	170.6	128.6	170.6
		(0.43)						(103.2)	(103.2)	(103.2)	(103.2)	(131.6)
			X_{1t}					10	25		12	20
								(10)	(15)	(10)	(12)	(20)
3	30	0.53	$S_{3,t}$				44.3	30	30	44.3	30	30
		(0.43)					(41.5)	(30)	(30)	(30)	(30)	(30)
			X_{3t}				10	25		12	20	
							(10)	(25)		(12)	(20)	
4	90	0.43	$S_{4,t}$				96.5	90	90	96.5	90	96.5
		(0.43)					(96.5)	(90)	(90)	(90)	(90)	(96.5)
			X_{4t}				60	85		62		
							(60)	(55)	(30)	(62)		
6	20	0.33	$S_{6,t}$			39.7	39.7	26.6	26.6	39.7	26.6	26.6
		(0.43)				(45.9)	(45.9)	(28.6)	(28.6)	(28.6)	(28.6)	(28.6)
			X_{6t}			70	120	25	74	32	20	
						(70)	(90)	(45)	(84)	(32)	(20)	
8	80	0.22	$S_{8,t}$		80	80	80	80	80	80	80	80
		(0.43)			(80)	(80)	(80)	(80)	(80)	(80)	(80)	(80)
			X_{8t}		70	155		131		52		
					(70)	(145)		(141)		(52)		
7	10	0.12	$S_{7,t}$			12.4	10	10	10	10	10	10
		(0.43)				(18.6)	(10)	(10)	(10)	(10)	(10)	(10)
			X_{7t}			60	135	60	62	30		
						(60)	(105)	(90)	(62)	(30)		
9	40	0.02	$S_{9,t}$	40	40	40	40	40	40	40	40	40
		(0.43)		(40)	(40)	(40)	(40)	(40)	(40)	(40)	(40)	(40)
			X_{9t}	60	205	180	87	104	52			
				(60)	(175)	(180)	(107)	(114)	(52)			

注:加括号部分是文献[2]的 r 值和计算结果。

3.4 算例描述[1]

由于算法的关键部分是修改过的生产准备费用的计算过程,为了让其他学者能够更好地理解 RCWW-STVS 算法进行生产准备费用修改的过程,这里采用一个算例进行描述。我们采用一个与图 1 一样的一般结构产品 BOM 图,原始的生产准备费用和生产提前期与图 1 所示的一致。这里,计划期的长度为 9;物料 1(产品)在每个时期的需求量为{0,0,0,0,10,15,10,12,20};物料 2(产品)在每个时期的需求量为{0,0,0,0,0,50,40,20,30};从物料 1 至物料 9 的库存保管费用为{13,8,4,4,3,3,2,1,1}。在修改生产准备费用之前,假设费用修改顺序为{2,5,1,3,4,6,8,7,9}(该序列根据 BOM 图从上层到下层按先后关系来决定。对于 RCWW 算法,为了确定顺序中的第一个物料,有物料 1 和物料 2 可以选择。根据各物料的原始生产准备费用,给物料 1 分配选择概率为 50/(50+60)=0.45,给物料 2 分配选择概率为 60/(50+60)=0.55。在 0 和 1 之间产生一个随机数 0.638,根据类似遗传算法的轮盘赌选择方式,由于该随机数大于物料 1 的选择概率 0.45,因此选择物料 2 作为序列的第一个元素),$R=0.431183$ 以及 $u=0.956399$(由此得到对应于费用修改顺序的 r_i 分别为{0.84,0.74,0.64,0.53,0.43,0.33,0.22,0.12,0.02})。

从物料 2 开始进行费用修改,$T(i,j,t)$ 的所有取值为 1(因为此时没有任何物料的生产量已经被确定下来),物料 8 和 9(生产材料)的修改后的费用与原始生产准备费用一样(因为它们没有前驱物料)。为了计算物料 2 修改后的生产准备费用,需要根据式(8)从产品 BOM 的最底层到最顶层来计算物料 2 的所有前驱物料的过渡变量值(为了使用式(10)而需要根据式(8)计算临时修改的生产准备费用)。因此根据图 1 先计算 $S_8=s_8=80$ 及 $S_9=s_9=40$,再计算 $S_6=s_6+S_8/2+S_9/2=80$ 及 $S_7=s_7+S_9/2=30$,然后计算 $S_4=s_4+S_6/3+S_7/2=131.7$ 及 $S_5=s_5+S_7/2=95$,最后计算 $S_{2t}=s_2+0.84*(S_4/2+S_5)=195.1$(这个值与图 1 给出的值有点差别,因为图 1 中的 r_2 实际上大于 0.84)。表 3 列出了所有物料修改后的生产准备费用的值以及根据这些值而计算出来的每种物料的生产量,根据不同的 r_i 值而计算出来的总费用为 2217,而

根据文献[2]的固定 r 值为 0.431183 的 RCWW 算法的结果为 2257。为了证明 R 和 u 能够对总费用产生影响，这里采用了不同的 R=0.36898 和 u=−0.87656 重新进行了计算，总生产成本为 2257，如表 4 所列。

表 4　R=0.36898 和 u=−0.87656 时各物料修改后的生产准备费用和生产量

i	s_i	r_i		1	2	3	4	5	6	7	8	9
2	60	0.05	$S_{2,t}$						67.3	67.3	67.3	67.3
			X_{2t}						50	40	20	30
5	80	0.13	$S_{5,t}$				81.9	81.9	81.9	81.9	81.9	81.9
			X_{5t}				50	60		30		
1	50	0.21	$S_{1,t}$					75.6	75.6	75.6	75.6	89.2
			X_{1t}					10	15	10	12	20
3	30	0.29	$S_{3,t}$				37.7	30	30	30	30	30
			X_{3t}				10	15	10	12	20	
4	90	0.37	$S_{4,t}$				95.5	90	90	95.5	90	95.5
			X_{4t}				60	55	30	62		
6	20	0.45	$S_{6,t}$			47	47	29	29	29	29	29
			X_{6t}			70	80	55	84	32	20	
8	80	0.53	$S_{8,t}$		80	80	80	80	80	80	80	80
			X_{8t}		70	145		141		52		
7	10	0.61	$S_{7,t}$			22.2	10	10	10	10	10	10
			X_{7t}			60	105	90	62	30		
9	40	0.69	$S_{9,t}$	40	40	40	40	40	40	40	40	40
			X_{9t}	60	175	170	117	114	52			

现在考虑 $T(i,j,t)$的修改过程。假设已经确定了物料 2 和物料 5 的各时段的生产量，接着要确定物料 1 的生产量。物料 3、6 和 8 是物料 1 的前驱物料，但它们都不是物料 2 和 5 的前驱物料，所以 $T(1,3,t)$、$T(1,6,t)$和 $T(1,8,t)$在所有时期都为 1。物料 4 既是物料 1 的前驱物料又是物料 2 的前驱物料。物料 2 在第 6、7 和 9 个时期有生产量安排，它有 2 个时期的生产提前期，物料 2 的生产导致物料 4 在第 4、5 和 7 个时期要进行生产准备活动，因此，如果物料 1 也需要物料 4 在第 4、5 和 7 个时期进行生产准备，物料 1 不用承担任何责任。则 $T(1,4,5)$、$T(1,4,6)$和 $T(1,4,8)$的值都为 0，对于其他时期 $T(1,4,t)$等于 1(也就是说，物料 1 的修改后的生产准备费用在第 5、6 和 8 个时期不加入物料 4 的部分过渡性的修改后费用，而在其他时期物料 1 的修改后的生产准备费用需要加入物料 4 的部分过渡性的修改后费用)。

4　算例验证

为验证文献[1]算法的正确性和作者对该文算法理解的正确性，采用 C 语言对 RCWW-STVS 算法进行了验证。本文的编程环境为：电脑采用 IBM Thinkpad R400；编程工具采用 VC++6.0；操作系统是 Window XP；内存大小为 1GB；CPU 为酷睿 2 双核 P8700、主频 2.53GHz。

本文具体的实现细节是采用结构体来记录、表达和存储 BOM 图，具体代码如下：

```
struct MATERIAL //物料结构
{
int info;//当前节点信息
int level;//当前节点所处层级
int processable;//当前节点是否可以处理
float s;//Setup cost
float S1;//过渡变量
float h;//库存保管费用
int leadtime;//物料生产提前期
float r;
int predecessor_number;//前驱节点个数
int successor_number;//后继节点个数
int predecessor[M+1];//当前节点前驱
int successor[M+1];//当前节点后继
};
```

本文算例来自文献[1]中的 3 个算例，产品 BOM 图与图 1 相同。算法的具体执行结果由图 2、图 3 和图 4 给出，图 2 对应文献[1]的 R=0.431183 以及 u=0.956399 的计算结果；图 3 对应文献[2]的 r=0.431183 的计算结果；图 4 对应文献[1]的 R=0.36898 和 u=−0.87656 的计算结果。通过与表 3 和表 4 的结果进行对比发现，本文算法对文献[2]的计算结果与文献[1]的结果在生产量方面不一致，对于其他两个算例的结果与文献[1]的结果完全一致。这是因为文献[1]给出的计算结果不正确，从而导致生产量的计算是错误的。

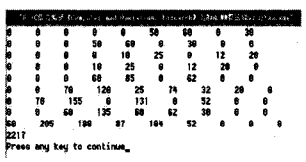

图 2　文献[1]中第一个算例的计算结果

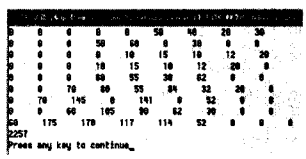

图 3　文献[2]中算例的计算结果

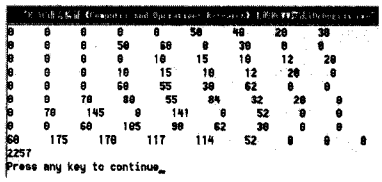

图 4　文献[1]中算例二的计算结果

细心的学者可以发现本文物料 3 的库存保管费用为 4，物料 3 的需求量为{10，15，10，12，10}。如果按照文献[1]的结果，在第 5 个时期生产 25 个，而在第 6 个时期不生产，则需要在第 5 个时期存储 10 个单位物料 3，其库存保管费用为 40。由于物料 3 的生产准备费用在第 6 个时期为 30，可以很容易地发现第 6 个时期生产 10 个单位物料 3 的费用要比在第 5 个时期存储 10 个单位物料 3 的费用要低。因此，本文算法的计算结果是完全正确的。

结束语 本文基于 C 语言，采用 VC++6.0 对 RCWW-STVS 算法进行编程实现，验证了其正确性和作者对该算法理解的正确性，发现了原文献的错误。本文成果为进一步开发有效的、基于智能优化算法的 LSP 问题求解工具，提供了有力的代码保障和基础支撑，具有重要的科学意义和研究价值。

参 考 文 献

[1] Pitakaso R, Almeder C, Doerner K F, et al. A MAX-MIN ant system for unconstrained multi-level lot-sizing problems [J]. Computers & Operations Research, 2007, 34(9): 2533-2552

[2] Dellaert N, Jeunet J. Randomized multi-level lot-sizing heuristics for general product structures [J]. European Journal of Operations Research, 2003, 148(1): 211-228

[3] Harris F. How many parts to make at once [J]. The Magazine of Management, 1913, 10(2): 135-136

[4] Wagner H, Whitin T. Dynamic version of the economic lot size model [J]. Management Science, 1958, 5(1): 89-96

[5] Coleman B. A further analysis of variable demand lot-sizing techniques [J]. Production and Inventory Management, 1992, 33(3): 19-24

[6] Wagelmans A, Van Hoesel S, Kolen A. Economic lotsizing: an O($n\log n$) algorithm that runs in linear time in the Wagner-Whitin case [J]. Operations Research, 1992, 40(S1): 145-155

[7] Aggarwal A, Park J. Improved algorithms for economic lot-size problem [J]. Operations Research, 1993, 41(3): 549-571

[8] Federgruen A, Tzur M. A simple forward algorithm to solve general dynamic lot sizing models with n periods in O($n\log n$) or O(n) time [J]. Management Science, 1991, 37(8): 909-925

[9] 唐立新，杨自厚，王梦光. 单级无能力约束批量大小问题的遗传搜索算法 [J]. 东北大学学报：自然科学版，1997，18(3)：312-315

[10] Schussel G. Job-shop lot release sizes [J]. Management Science, 1968, 14(8): B449-B472

[11] 韩毅. 离散制造业中生产批量计划问题的求解算法研究[D]. 沈阳：东北大学，2009

(上接第 216 页)

集 $B-\{b\}$ 下可以发生合并，其中 $x_i\in E_i, x_j\in E_j$ 满足 $\delta_X(x_i)\ \delta_X(x_j)=-1$。故 $\rho(x_i,b)\neq\rho(x_j,b)$ 且 $\forall b'\in(B-\{b\})$，均有 $\rho(x_i,b')=\rho(x_j,b')$。依据式(9)，有 $b\in c_{ij}$ 且 $\forall b'\in(B-\{b\})$，均有 $b'\notin c_{ij}$，故 $(B-\{b\})\cap c_{ij}=\varnothing$。从而说明 $\forall b\in B$，$\exists i,j,c_{ij}\neq\varnothing$，使得 $(B-\{b\})\cap c_{ij}=\varnothing$。

定理 3 说明，对于二值决策问题，基于全局相对增益函数和基于期望绝对增益函数构建的 Bayesian 粗糙集属性约简模型可通过统一的形式予以描述，其实质是将各决策类关于条件属性集 C 的 $*$-正域与 $*$-负域中的对象进行分辨，即保持各决策类关于条件属性集 C 的 $*$-正域与 $*$-负域在属性约简过程中均不减小，从而使得各决策类关于 U/C 各原子集的后验概率在属性约简前后保持不变。

结束语 Bayesian 粗糙集模型综合了粗糙集理论与 Bayesian 推理的优点，对现实问题处理所得结论的可解释性更强。在分析目前二值决策问题中两种 Bayesian 粗糙集属性约简模型的基础上，证明了基于全局增益函数及期望绝对增益函数建立的二值决策问题属性约简模型等价。作为重要的知识表示形式，分辨矩阵在粗糙集理论知识约简研究中具有重要地位。对于二值决策问题 Bayesian 粗糙集属性约简模型给出了相应的分辨矩阵表示，从而 Pawlak 经典粗糙集模型中已有基于分辨矩阵的知识约简策略及算法可直接平移应用于 Bayesian 粗糙集模型。现实应用中，人们往往面对多值决策问题，如何在二值决策 Bayesian 粗糙集理论基础上，构建多值决策问题下 Bayesian 粗糙集近似度量指标及属性约简模型将是下一步的主要研究工作。

参 考 文 献

[1] Pawlak Z. New look on Bayes' theorem-The rough set outlook [C]// Proc. of RSTGC'2001. Matsue Shimane, Japan, 2001: 1-8

[2] Pawlak Z. A rough set view on Bayes' theorem[J]. International Journal of Intelligent Systems, 2003, 18(5): 487-498

[3] Slezak D, Ziarko W. Bayesian rough set model[C]// Proc. of the International Workshop on Foundation of Data Mining (FDM'2002). Maebashi, Japan, 2002: 131-135

[4] Ziarko W. Variable precision rough set model [J]. Journal of Computer and System Sciences, 1993, 46(1): 44-54

[5] Slezak D. Attribute reduction in the Bayesian version of variable precision rough set model[J]. Electronic Notes in Theoretical Computer Science, 2003, 82(4): 1-11

[6] Nishina T, Nagamachi M, Tanaka H. Variable precision Bayesian rough set model and its application to Kansei engineering [J]. *Transactions on Rough Sets V*, LNCS, Springer-Verlag, 2006, 4100: 190-206

[7] Widz S, Revett K, Slezak D. A hybrid approach to MR imaging segmentation using unsupervised clustering and approximate reducts[C]// Proc. of RSFDGrC'2005, LNAI. Springer-Verlag, 2005, 3642: 372-382

[8] Slezak D, Ziarko W. The investigation of the Bayesian rough set model[J]. International Journal of Approximation Reasoning, 2005, 40(1/2): 81-91

[9] Ziarko W. Probabilistic approach to rough sets[J]. International Journal of Approximate Reasoning, 2008, 49(2): 272-284

[10] 王虹，张文修. 关于贝叶斯粗糙集模型的知识约简[J]. 计算机科学，2005，32(11)：150-151

[11] 蔡娜，张雪峰. 基于贝叶斯粗糙集模型的属性约简[J]. 计算机工程，2007，33(24)：46-48

[12] 韩敏，张俊杰，彭飞，等. 一种基于多决策类的贝叶斯粗糙集模型[J]. 控制与决策，2009，24(11)：1615-1619

[13] Pawlak Z, Skowron A. Rudiments of rough sets[J]. Information Sciences, 2007, 177(1): 3-27

[14] 王国胤，姚一豫，于洪. 粗糙集理论与应用研究综述[J]. 计算机学报，2009，32(7)：1229-1246

[15] Ziarko W. Stochastic approach to rough set theory[C]// Proc. of RSCTC 2006, LNAI. Springer-Verlag, 2006, 4259: 38-48

[16] Skowron A, Rauszer C. The discernibility matrices and functions in information systems[M]. Slowinski R, et al., eds. Intelligent Decision Support Handbook of Applications and Advances of the Rough Sets Theory, Dordrecht: Kluwer Academic Publishers, 1991: 331-362