韓國OR學會誌
第6卷 第2號
1981年 10月

—研究論文—

N×3 Flow-shop 문제에 對한 修正된 發見的技法 分析과 기존技法과의 比較研究

(The Corrective Heuristic Algorithm Analysis of the N×3 Flow-shop Problem and Comparative Study with Multi-model)

姜 錫 昊*
鞠 光 鎬**

Abstract

This paper developed 3 flow-shop sequencing heuristic methods: modified RA method, modified RACS method and modified RAES method. These methods modified RA method, RACS method and RAES method developed by D.G. Dannenbring. These methods can easily determine desirable sequence of orders and can improve nx3 flow-shop's productivity and efficiency.

The maximum flow-time criterion is selected as the evaluation criterion of flow-shop's efficiency. We evaluated these 6 heuristic methods' performance. By the evaluation of the result, we can see that the modified methods produce a shorter maximum flow-time than the original methods.

Ⅰ. 序 論

최근에 flow-shop의 seququencing을 위해 많은 發見的(heuristic) 技法들이 개발되어 왔다. 그리고 D. G. Damnenbring [4]은 3가지 발견적 기법인 RA, RACS, RAES 기법을 개발하고, 이를 기존의 발견적 기법인 slope order index method [10], Campell, Dudek and Smith's algorithm [2], Merging algorithm, Individual exchange algorithm, Group exchange algorithm [8], [9] Random Sampling algorithm 들과 그 효율성을 비교하였다. 이 논문은 그 분석 결과 가장 좋다고 판정된 RA, RACS, RAES 기법과 이들을 개선하여 새로이 개발된 MRA, MRACS MRAES 기법들의 효율성을 비교 분석하고자 한다. 일반적으로 flow-shop의 효율성은 그 주문들의 처리순서에 따라 달라지게 되며 이 효율성을 최대화하는 주문들의 처리순서를 구하고자 하게 된다.

우리는 이 논문에서 주문의 수가 n개이고 공정이 3개인 $n\times3$ flow-shop을 고려하며 효율성의 평가기준으로 최대흐름시간(maximum flow time/makespan)을 선택하여 이를 최소화하고자 한다.

즉, Conway et al, [3]이 분류한 용어로는 이 문제는 $n/3/F/F\max$가 된다.

Ⅱ. 기존의 發見的 技法

기존의 발견적 기법으로는 D.G. Dannenbring이 개발한 RA, RACS, RAES 기법을 선택하였는데 이들을 설명하면 다음과 같다.

(1) Rapid access procedures [4] (RA)

Rapid access procedures는 다음과 같이 가중치를 주어서 n개의 작업공정을 갖는 경우를 2개의 작업공정을 갖는 경우로 변화시킨 후 Johnson의 two-machine

*서울大學校 產業工學科
**國土開發硏究院

algorithm을 적용하여 해를 구한다.

이때 가중치는 t_{ij}를 i번째 작업이 j번째 공정에서 처리되는데 걸리는 시간이라 하고 p_{ij}를 two-machine의 경우로 변화된 후에 i번째 작업이 j번째 공정에서 걸리는 시간이라고 하면 다음과 같이 된다.

$$Pi_1=\sum_{j=1}^{m}(m-j+1)t_{ij},\quad Pi_2=\sum_{j=1}^{m}(j)t_{ij}$$

(2) Rapid Access with Close order Search [4] (RACS) RACS는 RA algorithm으로 구한 작업순서를 기초로 하여 다음과 같이 해를 구한다.

(a) n개의 작업으로 이루어진 경우 RA algorithm에 의해 구한 해를 인접하는 두 개의 작업을 교환하여 $(n-1)$개의 Sub-problem을 만든다.

(b) 이 $(n-1)$개의 Sub-problem의 maximum flow-time을 구해 그중 가장 적은 시간을 갖는 Sequence를 최종해로 선택한다.

(3) Rapid access with extensive search [4] (RAES) RAES는 RACS에 의해 구해진 해에 계속 RACS 기법을 적용시키는 방법으로 더 이상 해가 좋아지지 않으면 이를 최종해로 선택한다.

Ⅲ. 改善된 發見的 技法

앞서 D.G. Dannenbring이 개발한 RA, RACS, RAES기법들을 개선하여 3개의 새로운 휴리스틱인 MRA, MRACS, MRAES을 개발하였는데 기본개념 및 절차는 다음과 같다.

3.1 개선된 발견적 기법의 기본개념

새로운 기법들은 two-macine의 경우에 적용되는 다음과 같은 성질에 착안하여 개발되었다. 즉, two-machine flow-shop sequencing problem에 Johnson's algorithm을 적용하여 구한 해의 작업쌍 사이에는 다음과 같은 성질이 존재한다.

성질 : 한쌍의 작업이 있을 때 그 순서를 바꾸어보아 작은 흐름 시간을 갖는 작업이 먼저 배치된다.

이 성질을 例를 들어 설명하면 다음과 같다.[그림 1] 앞에서 Johnson's algorithm에 의해 이 예의 최적해를 구하면 2-1-3-4가 됨을 밝혔다. 이 예에 위의 성질을 적용하면 다음과 같이 된다.

작업순서	흐름시간	작업순서	흐름시간	최적 작업순서
(1-2)	12	(2-1)	11	(2-1)
(1-3)	10	(3-1)	13	(1-3)
(1-4)	9	(4-1)	12	(1-4)
(2-3)	9	(3-2)	11	(2-3)
(2-4)	7	(4-2)	10	(2-4)
(3-4)	10	(4-3)	11	(3-4)

즉, 위 선행관계에 의해서도 Optimal Sequence 2-1-3-4가 얻어진다.

이 two-machine의 경우에 적용되는 성질을 three-machine의 경우로 확장하고자 하는 것이 이 논문의 기본 개념이다.

그러나 위와 같이 해를 구하려면 먼저 $_nC_2$의 작업쌍에 대해 작업 흐름시간을 구하고 이를 비교해야 하며, 또 이를 전체적으로 종합하는 작업이 필요하다. 따라서 개선된 기법은 이 작업을 좀 더 간단히 하기 위해서 RA algorithm에 의한 해를 초기해로 삼고 이 해의 인접하는 작업쌍들에 위의 성질을 적용시켜 해를 개선해 나가고자 한다.

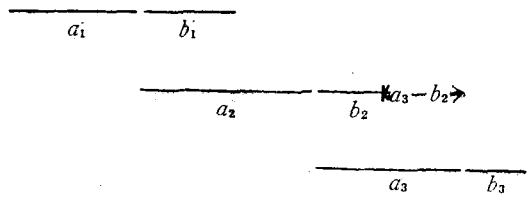
3.2 두 작업의 비교

두 작업 a, b를 비교하는 것은 다음과 같이 three-machine의 경우에도 쉽게 구해진다.

우선 작업 a와 b가 첫째, 둘째, 세째 공정에서 처리되는 시간이 각각 a_1b_1, a_2b_2, a_3b_3라 할 때 이를 다음

	Pi_1	Pi_2
1	3	5
2	2	4
3	5	2
4	4	1

〈그림 1〉



〈그림 2〉

과 같이 표현한다.

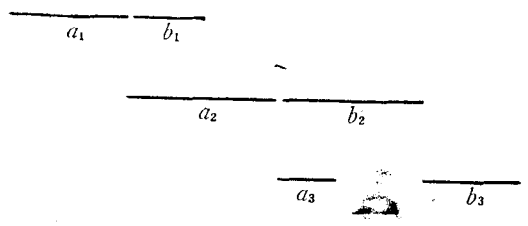
$a=(a_1, a_2, a_3)$, $b=(b_1, b_2, b_3)$

그러면 공정시간의 성질에 따라 다음과 같이 분류될 수 있다.

경우 1. $a_2 \geq b_1, a_3 \geq b_2 \Rightarrow b_1-a_2 \leq 0, a_3-b_2 \leq 0$〔그림 2〕

전체 흐름시간 : $a_1+[a_2+b_2]+(a_3-b_2)+b_3$

경우 2. $a_2>b_1, a_3<b_2 \Rightarrow b_1-a_2<0$ $a_3-b_2<0$

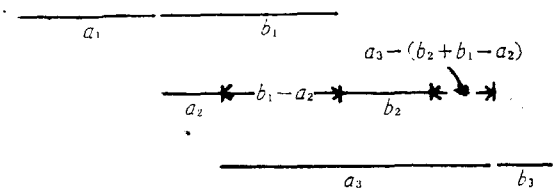


〈그림 3〉

전체 흐름시간 : $a_1+[a_2+b_2]+b_3$

경우 3. $a_2<b_1$, $a_3>(b_2+b_1-a_2) \Rightarrow a_3-b_2>b_1-a_2$

$a_3-b_2>0$, $b_1-a_2>0$



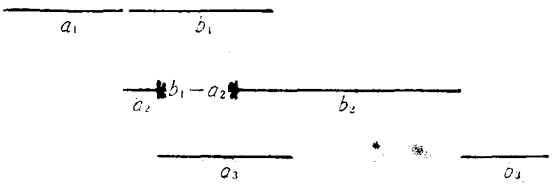
〈그림 4〉

전체 흐름시간 : $a_1+a_2+b_1-a_2+b_2+a_3-(b_2+b_1+a_2)$

$+b_3=a_1+[a_2+b_2]+(a_3-b_2)+b_3$

경우 4. $a_2<b_1$, $a_3<(b_2+b_1-a_2) \Rightarrow a_3-b_2<b_1-a_2$

$b_1-a_2>0$



〈그림 5〉

전체 흐름시간 : $a_1+[a_2+b_2]+(b_1-a_2)+b_3$

위의 결과를 자세히 살펴보면 각 경우의 전체흐름시간에는 a_2와 b_2의 값이 항상 들어감을 알 수 있다. 그리고 이들은 순서가 바뀌어도 합은 같으므로 이들을 빼고 위 결과를 요약하면 다음과 같이 된다.

(1) $\delta=\max[b_1-a_2, a_3-b_2]$를 구한다.

(2) $\delta>0$이면 $(a-b)$의 흐름시간을 $a_1+b_3+\delta$로 하고 $\delta<0$이면 $(a-b)$의 흐름시간을 $a+b_3$로 한다.

위 (1), (2)의 과정을 작업순서 $(a-b)$와 $(b-a)$에 적용하고 이 중 작은 흐름시간을 갖는 작업순서를 택한다.

3.3 Modified RA algrithm (MRA)

개선된 기법은 다음의 절차를 거쳐서 근사해를 구한다. 먼저 앞에서 설명한 RA algorithm에 의해서 초기해를 구한 후 이 해의 인접하는 작업쌍에 다음의 과정을 적용한다.

(1) i번째와 $i+1$번째의 작업을 각각 a와 b라 할 때 $(a-b)$의 작업순서의 흐름시간보다 $(b-a)$의 작업순서의 흐름시간이 작으면 i번째에 b를 놓고 $i+1$번째에 a를 놓아 처음 RA Sequence를 변화시킨다. 그렇지 않은 경우에는 원래의 RA sequence를 그대로 둔다.

(2) RA sequence가 변화하지 않으면 다음 작업쌍, 즉 $i+1$번째와 $i+2$번째의 작업쌍에 위 (1)의 과정을 적용하고 변화하는 경우에는 앞의 작업, 즉 $i-1$번째 작업과 현재 i번째 작업에 위 (1)의 과정을 적용한다. 모든 작업을 비교한 경우에는 다음 (3)을 적용한다.

(3) 위 (1), (2)의 과정을 적용하여 구해진 sequence의 흐름시간과 RA sequence의 흐름시간을 비교하여 적은 값을 갖는 sequence를 최종작업순서로 선택한다.

3.4 Modified RA algorithm with Close order Search (MRACS)

앞서 RAES가 RA sequence를 기초로 하여 구한 과정을 MRACS는 MRA에 의한 sequence에 적용하여 해를 구한다.

3.5 Modified RA algorithm Extensive Search (MRAES)

앞서 RACS가 RA sequence를 기초로 하여 구한 과정을 MRAES는 MRA에 의한 sequence에 적용하여 해를 구한다.

Ⅳ. 比較基準

4.1 비교된 문제들

위 각 기법들의 효율성을 비교하기 위해서 200개의 flow-shop sequenceing 문제를 컴퓨터를 이용하여 생성하였다. 각 문제에서 각 작업들의 각 공정에서 처리되는데 걸리는 시간은 컴퓨터를 이용하여 random number를 발생시켜 구했는데 이 값의 범위는 0-9의 정수값을 갖게 했다.

4.2 Performance의 평가기준

(1) Relative error : (R)

Relative error는 다음과 같이 구해진다.

$R=100[1-(MS/MS^*)]$

(2) Consistency : (C)

$C=R^2$

(3) Error potential ratio : (EPR)

$EPR=100(MS-MS^*/HS_*-MS^*)$

(4) Percentage of solutions equalling the optimum : (P)

$P=100(V/R)$

(5) Weighted rank order : (W)

$$W=\sum_{j=1}^{m}(j)b_j$$

위에서 각 기호들은 다음과 같다.

MS : 각 heuristic 기법들의 최대흐름시간(makespan)

MS* : 최적해의 최대흐름시간

MS_* : 발견적 기법들로 구한 해중 가장 나쁜 최대흐름시간

V : R개의 문제중 각 heuristic algorithm들이 최적해를 낮는 횟수

b_j : j번째 좋은 해를 갖는 회수($j=1$은 최적해를 나타낸다.)

4.3 最適解의 결정

최적해는 Ignall and Schrage [5]의 방법을 적용하여 구하였다. 즉 각 발견적 기법들의 평가기준으로 사용하기 위해 구하였다. 그러나 限界分岐法(branch and bound method)을 사용하는 것은 시간이 많이 걸리는 단점이 있다.

Ⅴ. 結果의 比較 및 分析

5.1 發見的 技法들의 효율성 비교결과

200개의 문제에 각 heuristic algorithm들을 적용하여 해를 구한 후 이들의 효율성을 비교 분석한 것이 다음 표-1에 나와 있다.

〈표 1〉 발견적 기법들의 효율성 비교 결과

	R	C	EPR	P	W	WR*
	%	%	%	%		%
MRAES	0.957	9.913	12.511	80.5	1.20	29.545
RAES	1.200	13.232	16.269	74.5	1.28	36.364
MRACS	1.306	13.138	16.258	75.0	1.29	29.545
RACS	1.833	18.863	23.475	65.0	1.47	36.364
MRA	2.247	21.478	35.647	58.0	1.61	29.545
RA	3.565	35.269	62.500	37.5	2.03	36.364

WR* : 200문제중 worst case의 상대오차(상대오차중의 최대값)을 나타낸다.

그리고 각 작업들의 특성에 따라 다음과 같이 6가지 경우로 나누어 각 경우에서 발견적 기법들이 갖는 효율성을 비교하였다. 이 200문제중 각 경우들이 발생한 횟수는 다음과 같다.

(1) $A\geq B\geq C$……20문제

(2) $A\geq C\geq B$……25

(3) $B\geq A\geq C$……41

(4) $B\geq C\geq A$……46

(5) $C\geq A\geq B$……30

(6) $C\geq B\geq A$……38

이때 A, B, C는 다음을 나타낸다.

A : 첫번째 공정에서 처리되는 작업들의 첫번째 공정시간을 모두 더한 값 $A=\sum_{i=1}^{n}A_i$

〈표 2〉 $A\geq B\geq C$인 경우의 발견적 기법들의 효율성 비교 결과

	R	C	EPR	P	W	WR*
	%	%	%	%		%
MRAES	0.0	0.0	0.0	100.0	0.0	0.0
RAES	0.0	0.0	0.0	100.0	0.0	0.0
MRACS	0.0	0.0	0.0	100.0	0.0	0.0
RACS	0.0	0.0	0.0	100.0	0.0	0.0
MRA	0.384	1.087	15.0	85.0	1.15	3.704
RA	1.534	5.546	50.0	50.0	1.5	5.0

WR* : $A\geq B\geq C$인 경우의 worst case의 상대오차를 나타낸다.

B : 모든 작업들의 두번째 공정시간을 더한 값

$$B=\sum_{i=1}^{n}B_i$$

C : 모든 작업들의 세번째 공정시간을 더한 값

$$C=\sum_{i=1}^{n}C_i$$

위 6가지 경우하에서의 각 발견적 기법들의 결과를 계산한 결과가 표-2부터 표-7까지에 나타나 있다.

〈표 3〉 $A\geq C\geq B$인 경우의 발견적 기법들의 효율성 비교 결과

	R	C	EPR	P	W	WR*
	%	%	%	%		%
MRAES	0.075	0.142	4.000	96.0	1.04	1.887
RAES	0.153	0.231	5.333	92.0	1.08	1.887
MRACS	0.075	0.142	4.000	96.0	1.04	1.887
RACS	0.135	0.231	5.333	92.0	1.08	1.887
MRA	0.456	3.770	8.000	92.0	1.08	9.524
RA	1.674	9.942	40.000	60.0	1.44	9.524

WR* : $A\geq C\geq B$인 경우의 worst case의 상대오차를 나타낸다.

〈표 4〉 $B\geq A\geq C$인 경우의 발견적 기법들의 효율성 비교 결과

	R	C	EPR	P	W	WR*
	%	%	%	%		%
MRAES	0.723	3.842	15.447	82.9	1.171	6.667
RAES	1.019	5.317	19.340	75.6	1.268	6.667
MRACS	1.147	5.826	21.219	70.7	1.293	7.273
RACS	1.751	8.385	31.870	51.2	1.561	7.273
MRA	2.793	14.422	65.122	24.4	1.902	10.909
RA	3.613	20.810	82.927	17.1	2.244	10.909

WR* : $B\geq A\geq C$인 경우의 worst case의 상대오차를 나타낸다.

〈표 5〉 $B\geq C\geq A$인 경우의 발견적 기법들의 효율성 비교 결과

	R	C	EPR	P	W	WR*
	%	%	%	%		%
MRAES	2.204	27.511	25.854	56.5	1.435	29,545
RAES	2.474	37.615	29.413	52.2	1.522	36.364
MRACS	3.059	36.309	34.556	47.8	1.652	29.545
RACS	3.535	48.225	40.543	43.5	1.804	36.364
MRA	4.502	54,428	59.955	34.8	2.174	29.545
RA	5.615	74.078	78.261	21.7	2.478	36.364

WR* : $B\geq C\geq A$인 경우의 worst case의 상대오차를 나타낸다.

〈표 6〉 $C\geq A\geq B$인 경우의 발견적 기법들의 효율성 비교 결과

	R	C	EPR	P	W	WR*
	%	%	%	%		%
MRAES	0.303	1.209	6.032	90.0	1.10	3.846
RAES	0.608	2.617	10.794	83.3	1.20	6.154
MRACS	0.488	1.578	8.115	83.3	1.167	3.846
RACS	1.213	7.157	15.877	73.3	1.333	10.204
MRA	1.353	9.353	18.198	73.3	1.367	12.245
RA	2.714	24.946	43.333	56.7	1.80	16.327

WR* : $C\geq A\geq B$인 경우의 worst case의 상대오차를 나타낸다.

〈표 7〉 $C\geq B\geq A$인 경우의 발견적 기법들의 효율성 비교 결과

	R	C	EPR	P	W	WR*
	%	%	%	%		%
MRAES	1.299	13.821	10.490	78.9	1.237	15.517
RAES	1.651	16.154	17.069	68.4	1.342	15.517
MRACS	1.497	17.568	11.806	78.9	1.263	15.517
RACS	2.433	26.053	24.044	63.2	1.579	15.517
MRA	1.793	21.156	17.251	73.7	1.395	16.071
RA	4.019	44.348	57.895	42.1	2.105	16.667

WR* : $C\geq B\geq A$인 경우의 worst case의 상대오차를 나타낸다.

5.2 발견적 기법들의 계산시간 비교

200개의 문제를 푸는데 각 발견적 기법들이 소요한 컴퓨터 계산시간은 다음 〈표 8〉에서 보는 바와 같다.

〈표 8〉 발견적 기법들의 계산시간 비교

	CPU Time	Process Time
RA	51S 058MS	4M 11S 520MS
MRA	58S 204MS	4M 30S 813MS
RACS	1M 03S 438MS	4M 30S 935MS
MRACS	1M 12S 501MS	4M 52S 660MS
RAES	1M 20S 148MS	4M 46S 675MS
MRAES	1M 23S 859MS	5M 00S 281MS

〈표 1〉부터 〈표 8〉까지에서 각 발견적 기법들의 결과가 다음과 같이 됨을 알 수 있다.

(1) 모든 문제 :

RA<MRA<RACS<MRACS<RAES<MRAES

(2) $A\geq B\geq C$:

RA<MRA<RACS=MRACS=RAES<MRAES

(3) $A\geq C\geq B$:

RA<MRA<RACS=RAES<MRACS=MRAES

(4) $B \geq A \geq C$:

RA<MRA<RACS<MRACS<RAES<MRAES

(5) $B \geq C \geq A$:

RA<MRA<RACS<MRACS<RAES<MRAES

(6) $C \geq A \geq B$:

RA<MRA<RACS<MRACS<RAES<MRAES

(7) $C \geq B \geq A$:

RA>RACS<MRA<RAES<MRACS<MRAES

그리고 각 경우의 MRAES와 relative error와 worst case의 relative error를 조사하면 다음과 같다.

	R	WR
(1) $A \geq B \geq C$	0.000%	0.000%
(2) $A \geq C \geq B$	0.075	1.887
(3) $B \geq A \geq C$	0.723	6.667
(4) $B \geq C \geq A$	2.204	29.545
(5) $C \geq A \geq B$	0.303	3.846
(6) $C \geq B \geq A$	1.299	15.517

즉 MRAES는 $A \geq B \geq C$, $A \geq C \geq B$, $C \geq A \geq B$의 경우, 즉 $A \geq B$인 경우에 Performance가 매우 좋음을 알 수 있다.

그런데 RACS나 MRACS, RAES, MRAES 등은 Computer의 도움이 없이는 해를 구하기 어려운 단점이 있다. 그러나 MRA는 computer의 도움이 없이도 해를 쉽게 구할 수 있으므로 MRA의 relative error와 worst case의 relative error를 조사하면 다음과 같다.

	R	WR
(1) $A \geq B \geq C$	0.384%	3.704%
(2) $A \geq C \geq B$	0.456	9.524
(3) $B \geq A \geq C$	2.793	10.909
(4) $B \geq C \geq A$	4.502	29.545
(5) $C \geq A \geq B$	1.353	12.245
(6) $C \geq B \geq A$	1.793	16.071

즉, MRA도 $A \geq B \geq C$, $A \geq C \geq B$, $C \geq A \geq B$의 경우 즉 $A \geq B$의 경우에 Performance가 매우 좋음을 알 수 있다.

그러므로 컴퓨터가 없는 소규모 flow-shop인 경우에는 MRA를 사용하여 sequencing함으로써 shop의 생산성이나 효율성을 높일 수 있음을 알 수 있다.

Ⅵ. 結 論

지금까지 이 논문은 $n \times 3$ flow-shop sequencing 문제를 풀 수 있는 modified algorithm을 개발하고 이를 기존의 algorithm의 performance와 비교 검토하였다. 그 결과 modified algorithm의 performance가 더 좋음이 판명되었고 거의 최적해에 가까운 해를 낳는다는 것을 알 수 있었다.

이 방법의 이용분야는 꼭 flow-shop의 경우에만 국한되지는 않는다. 보통의 service업에서도 flow-shop의 특성을 갖는 경우에는 이 방법을 적용하여 효율성을 높일 수 있을 것이다.

끝으로 이 논문은 $n \times 3$ flow-shop의 경우만 다루었는데 이 modified algorithm은 $n \times m$ flow-shop의 경우에도 확장될 수 있다.

참 고 문 헌

1. Burns, F., and J. Rooker, "Extensions and Comments Regarding Special Cases of the Three Machine Flow-Shop Problem", *Nav. Res. Log. Quart.* 22, 1975, pp. 811—817.
2. Campell, H.G., Dudek, R.A., and M.L. Smith, "A Heuristic Algorithm for the n Job, m Machine Sequencing Problem", *Management Science*, Vol. 16, pp. 214—221.
3. Conway, R.W., Maxwell, W.L., and L.W. Miller, *Theory of Scheduling*, Addison-Wesley, Reading, Massachusetts, 1967.
4. Dannenbring D.G., "An Evaluation of Flow-Shop Seqencing Heuristics", *Management Science, Vol.* 23, No. 11, July 1977, pp. 774—1182.
5. Ignall, E.J., and L.E. Schrage, "Application of the Branch and Bound Technique to Some Flow-Shop Scheduling Problems", *Operations Research*, 13 (3), 1965.
6. Johnson, L.A., and D.C. Montgomery, *Operations Research in Production Planning, Scheduling and Inventory Control*, John Wiley and Sons Inc. 1974.
7. Johnson S.M., "Optimal Two and Three Stage Production Schedules With Set-up Times Included," *Nav. Res. Log. Quart.* 1, 1954, pp. 61—68.
8. Page, E.S., "An Approach to Scheduling Jobs on Machines", *Journal of the Royal Statistical Society*, Series B, Vol. 23, pp. 484—492.
9. ———, "On the Scheduling of Jobs by Computer", *The Computer Journal*, Vol. 5, pp. 214—221.
10. Palmer, D.S., "Sequencng Jobs Through a

Multi-Stage Process in the Minimum Total Time-A Quich Method of Obtaining a Near Optimum", *Operational Research Quartrely*, Vol. 16, pp. 101—107.

11. Szwarc, W., "A Note on Mathematical Aspects of the 3xn Job-Shop Sequencing Problem", *Nav. Res. Log. Quart.* 21, 1974, pp. 725—726.
12. ——, "Mathematical Aspects of the 3xn Job-Shop Sequencing Problem", *Nav. Res. Log. Quart.* 1974, pp. 145—153.