Journal of The Korea Society of Computer and Information
Vol. 21 No. 4, pp. 47-54, April 2016
www.ksci.re.kr
http://dx.doi.org/10.9708/jksci.2016.21.4.047

Machine Layout Decision Algorithm for Cellular Formation Problem

Sang-Un Lee *

Abstract

Cellular formation and layout problem has been known as a NP-hard problem. Because of the algorithm that can be solved exact solution within polynomial time has been unknown yet. This paper suggests a systematic method to be obtain of 2-degree partial directed path from the frequency of consecutive forward order. We apply the modified Kruskal algorithm of minimum spanning tree to be obtain the partial directed path. the proposed reverse constructive algorithm can be solved for this problem with $O(mn)$ time complexity. This algorithm performs same as best known result of heuristic and metaheuristic methods for 4 experimental data.

▸ Keywords : Cellular Formation, Layout, consecutive forward order, 2-degree partial directed path

• First Author: Sang-Un Lee , Corresponding Author: Sang-Un Lee
*Sang-Un Lee (sulee@gwnu.ac.kr), Dept. of Multimedia Engineering, Gangneung-Wonju National University
• Received: 2016. 01. 11, Revised: 2016. 01. 26, Accepted: 2016. 03. 02.

I. Introduction

셀 생산방식(cellular manufacturing system, CMS)은 생산 분야의 혁신을 위해 낭비 없는 최적 생산시스템을 구축할 목적으로 작업장을 설계하는 모델로 셀 형성 문제(cell formation problem, CFP)라고도 한다. 이는 군 기술(group technology, GT) 원리에 기반하여 제품(또는 부품)간 유사성에 따라 유사한 제품들을 동일 제품군(families)으로 그룹화하고 해당 제품군을 생산하기 위한 기계들을 그룹화한 제조 셀들을 형성하여 생산하는 방식이다. 이 방식의 주요 장점은 자재 흐름이 상당히 향상되며, 자재 이동거리와 더불어 제조 기간(lead time)을 상당히 감소시킬 수 있다. 따라서 이러한 낭비를 최소화시키도록 셀 단위의 군을 형성하여 자원의 최대 효율성을 달성하는 것이 목표이다[1-4].

셀 형성 문제는 m대의 기계와 n개의 제품으로 구성된 생산체계에서, $a_{ij}=0,1$로 특정 제품이 특정 기계를 사용하는지 여부를 명시하는 경우(이를 일반적으로 CFP라 한다.), $a_{ij}=1,2,\cdots,m$으로 기계사용 순서를 명시한 경우, $a_{ij}=t$로 특정 기계로 작업을 수행하는 시간을 명시하는 경우 셀 능력을 만족시키도록 형성하는 CaCFP(Capacitated CFP)로 분류된다. 기계사용 순서를 명시한 $a_{ij}=1,2,\cdots,m$ 문제는 다시 제품 이동 비용(cost)을 최소화시키도록 셀(군)을 형성하는 CoCFP(Cost CFP)와 장비 배치 순서 LCFP(layout CFP)를 결정하는 문제로 분류된다. LCFP와 CoCFP의 a_{ij}는 작업순서(sequence)로 값이 주어진다. 이 문제에 대해 셀 내부의 기계 배치순서는 고려하지 않고, 단지 셀 내부의 불연속적인 기계간과 다른 셀들로의 제품 이동비용만을 최소화시키도록 셀을 형성하는 문제를 CoCFP라 하며, 단순히 셀 내부의 기계 배치순서만을 고려하는 경우를 LCFP라 한다. 본 논문에서는 LCFP에 초점을 맞춘다.

셀 형성 문제를 풀기위해 많은 연구가 진행되었다. 대표적인 방법으로는 군집분석(cluster analysis), 그래프 분할법(graph partitioning approaches)과 메타휴리스틱 방법(metaheuristic approaches)이 있다[1]. 셀 형성 문제는 NP-난제(NP-Hard)인 조합 최적화 문제로, 휴리스틱의 다항시간 알고리즘이 존재하지 않고 있어 부득이 메타휴리스틱 방법을 적용하고 있다[2-4]. 대표적인 메타 휴리스틱 방법으로는 유전자 알고리즘(genetic algorithm, GA), 진화 알고리즘(evolutionary algorithm, EA), 군집최적화(particle swarm optimization, PSO), GRASP(greedy randomized adaptive search procedure) 등을 적용하고 있음에도 불구하고 아직까지 다양한 모든 문제들에 대해 최적 해를 구하는 알고리즘이 존재하지 않고 있다.

CoCFP는 Teymourian et al.[5]이 있으며, LCFP에 대해서는 Nair와 Narendran[6]의 군집분석 기법의 CASE, Mutingi와 Onwubolu[7]의 군 유전자 알고리즘(group genetic algorithm, GGA), Mahadivi와 Mahadevan[8]의 CLASS, Maddavi et al.[9]의 흐름행렬 기반 휴리스틱 알고리즘 등이 있다. a_{ij}는 기계 부하인 경우 George et al[10]의 군집 알고리즘(clustering algorithm)이 있다.

Mahdavi와 Mahadevan[8]의 CLASS 알고리즘과 Mahadavi et al.[9]의 휴리스틱 알고리즘은 모두 기계 i에서 j로의 연속적인 순방향 제품 이동 횟수를 계산한 흐름 행렬(flow matrix)에 기반을 두고 있으며, 각각 9단계와 15단계의 과정을 수행한다. 이들 방법은 셀을 형성하는데 있어 체계적인 방법 대신 휴리스틱 방법을 적용하여 기계 대수가 많아질수록 셀을 형성하는데 어려움이 있다.

본 논문에서는 LCFP에 대해 셀 형성과 셀 내부의 장비 배치순서를 체계적인 방법으로 결정하는 알고리즘을 제안한다. 2장에서는 LCFP의 효율성을 평가하는 척도를 고찰해본다. 3장에서는 LCFP의 셀 형성과 장비 배치순서를 결정하는 다항시간의 체계적인 방법을 제안한다. 4장에서는 실험 데이터에 대해 제안된 알고리즘을 적용하고 기존의 연구 결과와 비교하여 제안된 알고리즘의 적합성을 평가해 본다.

II. LCFP Evaluation Metrics

특정 제품이 특정 기계로 작업하는 순서가 결정된 셀형성 문제는 $n\times m(i=1,2,\cdots,n, j=1,2,\cdots,m)$의 제품 P_i와 기계 M_j 행렬에 대해 s_{ij}는 제품 i가 기계 j로 작업을 수행하는 순서로 주어진다. LCFP는 셀을 형성함과 더불어 셀 내부의 장비 배치순서도 결정하는 문제이다. 이 문제에 대해 다음과 같이 정의한다.

- 제품 $P_i, i=1,2,\cdots,n$, n: 제품 수
- 기계 $M_j, j=1,2,\cdots,m$, m: 기계 수
- 셀 $C_k, k=1,2,\cdots,c, (c<m)$
- P_k: k번째 셀에 속한 제품 수
- M_k: 셀 k에 속한 기계 수
- N_{sk}: 셀 k의 크기 (size) $=P_k\times M_k$
- s_{ij}: i번째 제품이 j번째 기계에서 작업 (operation)이 수행되는 순서
- f_{ij}: i번째 기계에서 j번째 기계로 연속적인 순방향 이동 수 (number of consecutive forward movement)
- v_k: 셀 k의 빈칸 (void) 수
- N_{mk}: 셀 k내부 제품들의 연속적인 순방향 이동 수
- N_{tk}: 셀 k내부 제품들의 총 이동 수$=(n_k\times m_k)-v_k-n_k$
- N_{opk}: 셀 k내부의 작업 (operation) 수
- N_{ops}: 총 작업 수 $=\sum_{i=1}^{n}\max_j s_{ij}$
- N_{mov}: 총 작업 이동 수 $=N_{ops}-n$

LCFP의 기계 배치 효율성은 Mahdavi와 Mahadevan[8]이 제안한 식 (1)의 ACMI(average cell movement index)와 식 (2)의 OMI(overall movement index)로 평가한다. Mahdavi et al.[9]은 CMI_k를 계산하는데 있어 N_{tk}를 셀 내부의 순방향 이동 수(number of forward movement)로 잘못 적용하였다. 특정 제품이 4대의 기계

www.kci.go.kr

로 구성된 k번째 셀에서의 작업순서가 1→x→2→3인 경우, N_{mk}는 1→2, 2→3=2가 아닌 2→3=1로 계산하자. 즉, 불연속적인 순방향 이동수는 고려하지 않는다.

$$ACMI=\frac{\sum_{k=1}^{c} P_k \times CMI_k}{n}, \text{ where } CMI_k=\frac{N_{mk}}{N_{tk}} \quad (1)$$

$$OMI=\frac{\sum_{k=1}^{c} N_{mk}}{N_{mov}} \quad (2)$$

$ACMI$와 OMI는 모두 연속적인 순방향 작업 순서 효율성에만 초점을 맞추고 있다. 만약, N_{mk}를 극대화하도록 1개의 셀로 구성한다면 $ACMI$와 OMI를 증가시킬 수 있다. 그러나 척도만을 강조하면 셀 내부의 기계 활용도 측면은 저하된다. 이러한 단점을 보완하고자, Mahdavi et al.[9]은 셀 내부의 기계 활용도에 대한 셀 형성 효율성 평가 척도로 식 (3)의 ACUI(average cell utilization index)를 제안하였다.

$$ACUI=\frac{\sum_{k=1}^{C} CUI_k}{C}, \text{ where } CUI_k=\frac{N_{opk}}{N_{sk}} \quad (3)$$

참고로, Mahdavi et al.[9]의 휴리스틱 알고리즘은 최대 흐름 수 장비 (j,j')를 하나의 셀로 선택하고, 이 셀을 확장시키는 방법으로 다음과 같이 수행된다. 이를 셀 구성 알고리즘(constructive cell algorithm, CCA)이라 하자.

Step 1. $S=[s_{ij}]$로부터 $F=[f_{ij}]$ 흐름 행렬 작성.

Step 2. F 행렬에서 최대값 $\max f(j,j')$ 선택, 만약, 동일 값 존재 시 임의의 (j,j') 선택, 새로운 셀 생성

Step 3. j' 행 (j열)에 동일한 값 (j,j''') or (j'',j)이 있는지 확인. 만약 존재하면 $j'''(j'')$를 현재 셀에 배정, $j'(j)$ 기계는 삭제하고, 다른 셀로 생성하도록 유보시킴
j''' 행 (j'' 열)이 현재의 값과 동일한 값이 존재하는지 확인, 만약, 존재하면 동일한 방법으로 처리

Step 4. 형성된 셀의 마지막 기계(첫 번째 기계)에 대응하는 행 (열) 기계 선택, Step 3 수행

Step 5. 만약 모든 값을 확인하였으면 다음 최대값에 대해 Step 2 수행.

Step 6. 제품에 대해서는 가장 많이 작업될 셀로 배정, 만약, 동일 작업 횟수이면 CUI를 향상시킬 수 있는 셀로 배정.

III. Reverse Constructive Algorithm

본 장에서는 LCFP에 대해 셀 형성과 장비 배치 순서를 결정하는 체계적인 방법을 제안한다. 제안된 방법은 Mahdavi와 Mahadevan[8], Mahdavi et al.[9]와 동일하게 기계별 연속적인 순방향 이동 빈도수를 계산한 흐름행렬(flow matrix)에 기반한다. 제안된 알고리즘이 기존 방법과의 차이점은 다음과 같다. 흐름행렬의 빈도수를 내림차순으로 정렬시키고, 2-차수 부분 방향신장 경로(2-degree constrained partial directed spanning path, 2DCPDSP) 방식으로 셀을 형성하고, 기계와 제품에 대해 보다 효율성을 증대시키는 셀로 이동시키는 최적화를 수행한다. 여기서는 최소신장트리(minimum spanning tree, MST)를 구하는 Kruskal 알고리즘 기법을 응용하였다. Kruskal 알고리즘은 차수 (가지 수)는 고려하지 않고, 단지 가중치 합이 최소가 되도록 사이클이 없는 하나의 트리를 형성하는 문제이다. 반면에, 셀 형성의 장비 배치 순서 문제는 셀 내의 장비 배치 순서를 구하기 위해 최대 빈도수인 2-차수 부분 방향신장 경로를 찾는 것이 목적이다. 제안된 알고리즘을 역-구성 알고리즘(reverse constructive algorithm, RCA)이라 하자.

기계 $M=\{1,2,\cdots,m\}$에 대한 연속적인 순방향(consecutive forward) 장비 사용 순서를 계산한 흐름행렬 $F=[f_{ij}]$에 대해, 현재 k개의 셀 $C_i,(i=1,2,\cdots,k)$이 형성되었고, 이들 중 임의의 2개 셀을 $C_1=(x_1,\cdots,y_1), C_2=(x_2,\cdots,y_2)$라 하자. 여기서, x는 최 좌측(most left, 첫 번째), y는 최 우측(most right, 마지막)에 위치한 기계를 의미한다. RCA는 다음과 같이 수행된다.

Step 1. 흐름행렬 작성
$S=[s_{ij}]$로부터 $j=1,2,\cdots,m$의 각 기계별 연속적인 순방향 작업 수행순서 빈도수 f_{ij}인 $m\times m$ 흐름행렬 $F=[f_{ij}]$을 구한다. 이는 각 기계에 대한 $s_{ij}>1$인 제품 i 행의 $s_{ij}+1$을 갖는 j열 기계의 대수로 구한다.

Step 2. 흐름행렬에서 $f(w,z)$ 내림차순으로 정렬시키고 다음을 수행하여 단순화 시킨다.
(1) 동일 빈도수 $f_1=f_2$와 $f_3>(f_1=f_2)\geq f_3$에 대해
- $f_1(x,y)=f_2(y,x)$인 경우 : $f_3(x,z)$ or $f_3(y,z)$ 존재시 $f_2(y,x)$를, $f_3(x,z)$ or $f_3(w,y)$ 존재시 $f_1(x,y)$를 삭제한다.
- $f_1(x,y)=f_2(x,z)$인 경우 : $f_3(y,w)$ 가 존재시 $f_2(x,z)$를, $f_3(z,w)$ 이면 $f_1(x,y)$를 삭제한다.
- $f_1(x,z)=f_2(y,z)$인 경우 : $f_3(w,x)$ 가 존재시 $f_2(y,z)$를, $f_3(w,y)$ 이면 $f_1(x,z)$를 삭제한다.

(2) $f\geq 2$에 대해 $f_1(x,y)>f_2(y,x)$ 존재시 $f_2(y,x)$를 삭제한다.

Step 3. 첫 번째 $f(w,z)$에 대해 모든 $f(w,z)$을 수행할 때까지 다음 규칙을 적용하면서 기계에 대한 셀을 형성한다. 여기서 현재 생성된 셀 개수를 k, 임의의 2개 셀을 $C_1=(x_1,\cdots,y_1), C_2=(x_2,\cdots,y_2)$라 하자.
(1) $f\geq 2$인 경우
(추가) $(w\in M, z\notin M)$ or $(w\notin M, z\in M)$인 경우
$z=x_1$이면 $C_1=(w,x_1,\cdots,y_1)$
$w=y_1$이면 $C_1=(x_1,\cdots,y_1,z)$
(병합) $(w\notin M, z\notin M)$ 인 경우

www.kci.go.kr

$w=y_1, z=x_2$이면 $C_1=(x_1,\cdots,y_1,x_2,\cdots,y_2)$

(생성) $(w\in M, z\in M)$ 인 경우

$C_{k+1}=(w,z)$

(2) $f=1$, $M\neq\{\phi\}$인 경우 해당 기계에 대해서만 수행

(생성) $(w\in M, z\in M)$ 인 경우

$C_{k+1}=(w,z)$

(추가) $(w\in M, z\notin M)$ or $(w\notin M, z\in M)$인 경우

$z=x_1$이면 $C_1=(w,x_1,\cdots,y_1)$

$w=y_1$이면 $C_1=(x_1,\cdots,y_1,z)$

$f(w,z)$ 삭제, M에서 w,z 삭제.

Step 4. 기계에 대한 셀이 형성되면 각 제품을 가장 많이 포함하고 있는 셀에 배정한다. 이 때 동일한 개수이면 셀의 크기 (기계 대수)가 최소인 셀에 배정한다. 제품을 셀 별로 배치하고, 셀 내부의 제품은 순서대로 나열한다. 만약, 선택되지 않은 셀이 존재하면 해당 셀의 장비를 다른 셀로 병합한다.

Step 5. 기계와 제품에 대해 보다 크기가 작은 셀로 이동시켜 최적화를 수행한다.

IV. Applications and Evaluation

본 장에서는 Tam[11]의 12×19, Nair와 Narendran[6]의 8×20과 25×40, Harhalakis et al.[12]의 20×20 문제에 대해 제안된 RCA를 적용하여 본다. 본 문제들은 표 1에 제시되어 있다. 이 데이터들이 실험 데이터로 선정된 이유는 기존 알고리즘으로 얻은 해가 알려져 있기 때문에 기존 알고리즘과 본 논문에서 제안되는 알고리즘의 결과를 비교하여 제안된 알고리즘의 적합성을 검증할 수 있는 벤치마킹 데이터로 적합하기 때문이다.

Table 1. Cellular Formation and Layout Problem

(a) 8×20 problem

Machine	Product																			
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1		1	2					1	1		3		1	1		1	3		1	
2				1	1		1	4					2					2		2
3		2							2	3		2		2	3		2	1		2
4				5	2		2	2			2							1		1
5	2					2	5				3		1			1		2		
6	1					1				2	1		3			2				3
7				3	3		3	3				1	2						4		4
8				4	4		4	1											3		5

(b) 12×19 problem

Product	Machine											
	1	2	3	4	5	6	7	8	9	10	11	12
1	1			2				3				
2	1	2		4		3	6	5				
3	1	2		3			4	5	6			
4	1			2			3		4			
5	1					2	4		5	3		
6						1	3	4	5	2		
7				2		1		3	4			
8		3	1	5	2	4		6	7			
9			1	4	2	3		5	6			
10			1	3		2		4				
11						1						2
12						1	2			3		
13							2			1		
14							2				1	3
15							3			2	1	4
16							2			3	1	
17										2	1	
18											1	2
19							2				1	3

(c) 20×20 problem

Product	Machine																			
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	2							3			1							4		5
2		2	3							1										
3								1											3	2
4		3	1						4	2										
5				1		3	4							2						
6					5					1			2		3	4				
7					1										2	3				
8				5			3		4			2		1						
9	4								2		3	5						1		
10								3											1	2
11			3					1					2							
12	5				3				1			4						2		
13						1	2							3		4				
14	3	4						1		2										
15													1	2		3	4			
16						3	2								1				4	
17	2								1			3								
18								1		4									2	3
19		2	1		4						3									
20	3								2			4						1		

(d) 25×40 problem

Product	Machine																								
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
1				5			3			1						4		2				6			
2	2	3															4								1
3			2								3									1					
4												1												2	
5				3								2						1							
6												3				2								1	
7				3			2			5						4		1							
8				1												3			2						
9			3							4										1					2
10								2	1																3
11								2					3								1				
12	1		4														3							2	5
13			3								2									1					
14			4		1						2									3					
15		4			3				5							1			2						
16				1			3									2		4							
17							1			3								2							
18														3	2							1			
19								1	3	2															
20																2								1	
21								1	3	2															
22			3					4	2								1								
23				2												3			1						
24				1												2									
25					1									3							2				
26			2								3			4										1	
27											1										3	2			
28								2	1	3															
29				3	2																1				
30			4			2										3		1							
31				2													1		3						
32													2	1	3							4			
33											1									3					2
34												2												1	3
35						2									4						1	3			
36	2	3								4							1								
37						3						2												1	
38							2	3														1			
39						2						1													
40					2										3						1				

8×20 문제와 25×40 문제에 대해 제안된 RCA를 수행한 결과는 각각 표 2와 표 3에 제시되어 있다. 표 3에서 RCA 수행 방법은 표 4에 별도로 제시하였다. 12×19와 20×20 문제에 대해서는 결과만을 표 5에 제시하였다.

Table 2. Reverse-Constructive Algorithm for 8×20 Problem

(a) Step 1

	1	2	3	4	5	6	7	8
1		**1**	**5**			**1**	**1**	
2	**1**		**1**	**2**		**1**		**1**
3	**1**				**1**			
4		**2**			**1**		**3**	
5	**1**					**1**	**1**	
6			**1**	**1**	**2**		**1**	
7		**1**	**1**			**1**		**4**
8				**2**	**1**		**1**	

(b) Step 2 & Step 3

Sort	Simplify	Cell formation	Cell	M
5(1,3)	5(1,3)	5(1,3)	5(1,3)	{2,4,5,6,7,8}
4(7,8)	4(7,8)	4(7,8)	5,(1,3), 4(7,8)	{2,4,5,6}
3(4,7)	3(4,7)	3(4,7)+4(7,8)=7(4,7,8)	5,(1,3), 7(4,7,8)	{2,5,6}
2(2,4)	**2(2,4)**	2(2,4)+7(4,7,8)=9(2,4,7,8)	5,(1,3), 9(2,4,7,8)	{5,6}
2(4,2)	–	–	–	{5,6}
2(6,5)	2(6,5)	2(6,5)	5,(1,3), 9(2,4,7,8), 2(6,5)	$\{\phi\}$
2(8,4)	2(8,4)	–	–	–
$C_1=(1,3), C_2=(2,4,7,8), C_3=(6,5)$				

(c) Step 4

www.kci.go.kr

Machine	Product																			
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1		1	2					1	1		3		1	1		1	3		1	
3		2						2	3		2		2	3		2	1		2	
2			1	1		1	4							2				2		2
4			5	2		2	2			2								1		1
7			3	3		3	3				1	2						4		4
8			4	4		4	1											3		5
6	1				1				2	1		3			2					3
5	2				2	5				3		1			1		2			

(d) Step 5

Machine	Product																			
	2	8	9	11	13	14	16	17	19	3	4	6	7	18	20	1	5	10	12	15
1	1	1	1	3	1	1	1	3	1	2										
3	2	2	3	2	2	3	2	1	2											
2						2				1	1	1	4	2	2					
4										5	2	2	2	1	1			2		
7					1					3	3	3	3	4	4				2	
8										4	4	4	1	3	5					
6			2												3	1	1	1	3	2
5								2				5				2	2	3	1	1

Table 3. Reverse-Constructive Algorithm for 25×40 Problem

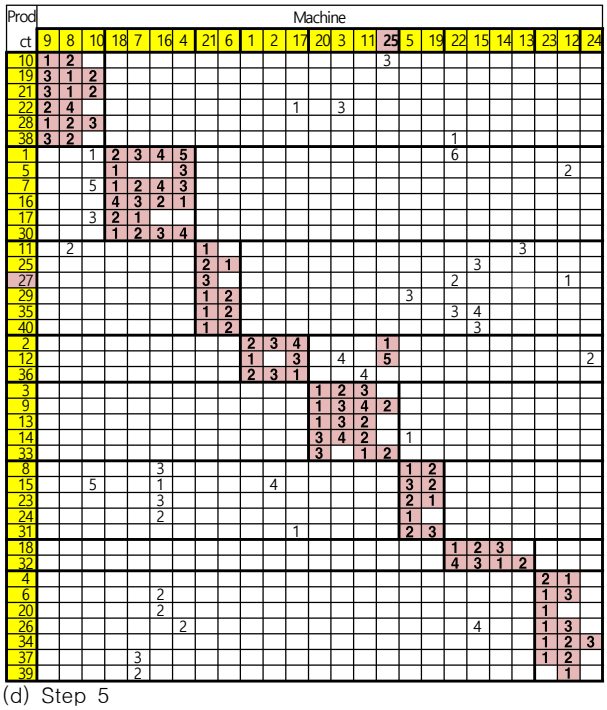
(a) Step 1

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
1		2																					1		
2										1	1						1								
3								1			2														1
4												1				2						1			
5		1								1						2			2						
6					1										1						1	1			
7				1												2		2							
8									1	3			1												1
9			1					2																	
10									2										1						
11			1																	1					1
12				1			2								1							1	1	1	
13															1										
14													1												
15														1								1			
16				2			1			1		1							1						
17	1		1		1				1																
18							3			1		1													
19					2												1								
20			2							1															1
21						3		1								1									
22								1								2						1			
23				1							2						2								
24																		1							
25	1		1																	1					

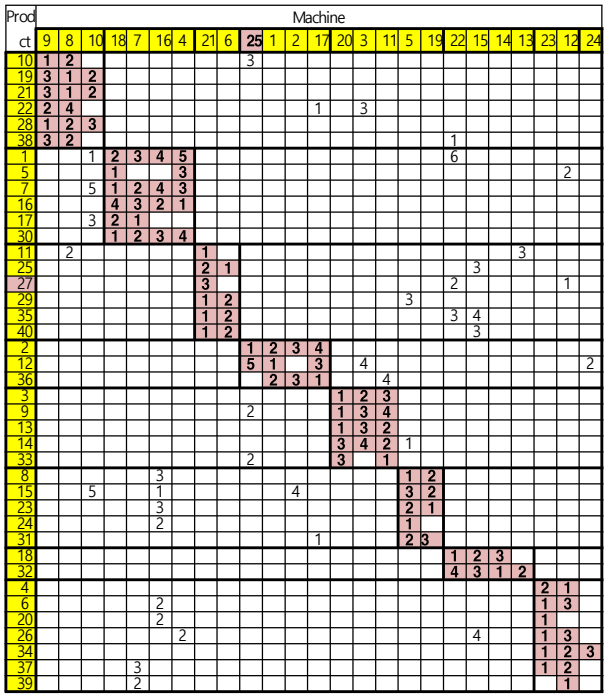
(b) Step 2 & Step 3

Prod ct	Machine																								
	9	8	10	18	7	16	4	21	6	1	2	17	20	3	11	25	5	19	22	15	14	13	23	12	24
1			1	2	3	4	5												6						
2										2	3	4				1									
3													1	2	3										
5				1			3																	2	
7			5	1	2	4	3																		
9													1	3	4	2									
10	1	2														3									
12										1		3		4		5									2
13													1	3	2										
14													3	4	2		1								
16				4	3	2	1																		
17			3	2	1																				
19	3	1	2																						
21	3	1	2																						
22	2	4										1		3											
28	1	2	3																						
30				1	2	3	4																		
33													3		1	2									
36										2	3	1			4										
38	3	2																1							
8						3											1	2							
11		2						1														3			
15			5			1					4						3	2							
23						3											2	1							
24						2											1								
25								2	1										3						
29								1	2								3								
31												1					2	3							
35								1	2										3	4					
40								1	2											3					
4																							2	1	
6						2																	1	3	
18																			1	2	3				
20						2																1			
26							2												4				1	3	
27								3											2					1	
32																			4	3	1	2			
34																							1	2	3
37					3																		1	2	
39					2																			1	

(c) Step 4



(d) Step 5



8×20 문제와 25×40 문제에 대한 RCA 결과를 CASE, GGA, CCA, CLASS와 비교한 결과는 각각 표 6과 표 7에 제시되어 있다.

제안된 RCA는 CCA와 동일한 결과를 얻었으며, CCA의 휴리스틱 방법으로 셀을 형성하는 방법 대신 보다 체계적인 방법을 적용한 특징을 갖고 있다.

4개의 실험 데이터에 RCA와 다른 알고리즘 결과를 비교한 결과는 표 8에 제시되어 있다. 여기서, RCA는 2가지 결과를 보여주고 있다. 각 실험데이터의 상단 결과는 3장에서 제안된

www.kci.go.kr

Table 4. Cell Formation Process of Reverse-Constructive Algorithm for 25×40 Problem

Sort	Simplify	Cell formation	Cells	M
3(8,10)	3(8,10)	3(8,10)	3(8,10)	{1,2,3,4,5,6,7,9,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25}
3(18,7)	3(18,7)	3(18,7)	3(8,10), 3(18,7)	{1,2,3,4,5,6,9,11,12,13,14,15,16,17,19,20,21,22,23,24,25}
3(21,6)	3(21,6)	3(21,6)	3(8,10), 3(18,7), 3(21,6)	{1,2,3,4,5,9,11,12,13,14,15,16,17,19,20,22,23,24,25}
2(1,2)	2(1,2)	2(1,2)	3(8,10), 3(18,7), 3(21,6), 2(1,2)	{3,4,5,9,11,12,13,14,15,16,17,19,20,22,23,24,25}
2(3,11)	2(3,11)	2(3,11)	3(8,10), 3(18,7), 3(21,6), 2(1,2), 2(3,11)	{4,5,9,12,13,14,15,16,17,19,20,22,23,24,25}
2(4,16),2(16,4)	2(4,16)	2(4,16)	3(8,10), 3(18,7), 3(21,6), 2(1,2), 2(3,11), 2(4,16)	{5,9,12,13,14,15,17,19,20,22,23,24,25}
2(5,16)	2(5,16)	-	3(8,10), 3(18,7), 3(21,6), 2(1,2), 2(3,11), 2(4,16)	{5,9,12,13,14,15,17,19,20,22,23,24,25}
2(5,19),2(19,5)	2(5,19)	2(5,19)	3(8,10), 3(18,7), 3(21,6), 2(1,2), 2(3,11), 2(4,16), 2(5,19)	{9,12,13,14,15,17,20,22,23,24,25}
2(7,16)	2(7,16)	3(18,7)+2(7,16)+2(16,4)=7(18,7,16,4)	3(8,10), 7(18,7,16,4), 3(21,6), 2(1,2), 2(3,11), 2(5,19)	{9,12,13,14,15,17,20,22,23,24,25}
2(7,18)		-	-	-
2(9,8)	2(9,8)	2(9,8)+3(8,10)=5(9,8,10)	5(9,8,10), 7(18,7,16,4), 3(21,6), 2(1,2), 2(3,11), 2(5,19)	{12,13,14,15,17,20,22,23,24,25}
2(10,8)		-	-	-
2(12,7)	2(12,7)	-	5(9,8,10), 7(18,7,16,4), 3(21,6), 2(1,2), 2(3,11), 2(5,19)	{12,13,14,15,17,20,22,23,24,25}
2(16,4)		-	-	-
2(19,5)		-	-	-
2(20,3)	2(20,3)	2(20,3)+2(3,11)=4(20,3,11)	5(9,8,10), 7(18,7,16,4), 3(21,6), 2(1,2), 4(20,3,11), 2(5,19)	{12,13,14,15,17,22,23,24,25}
2(22,15)	2(22,15)	2(22,15)	5(9,8,10), 7(18,7,16,4), 3(21,6), 2(1,2), 4(20,3,11), 2(5,19), 2(22,15)	{12,13,14,17,23,24,25}
2(23,12)	2(23,12)	2(23,12)	5(9,8,10), 7(18,7,16,4), 3(21,6), 2(1,2), 4(20,3,11), 2(5,19), 2(22,15), 2(23,12)	{13,14,17,24,25}
2(23,16)	2(23,16)	-	5(9,8,10), 7(18,7,16,4), 3(21,6), 2(1,2), 4(20,3,11), 2(5,19), 2(22,15), 2(23,12)	{13,14,17,24,25}
1(1,24)	1(1,24)	-	5(9,8,10), 7(18,7,16,4), 3(21,6), 2(1,2), 4(20,3,11), 2(5,19), 2(22,15), 2(23,12)	{13,14,17,24,25}
1(2,17)	1(2,17)	2(1,2)+1(2,17)=3(1,2,17)	5(9,8,10), 7(18,7,16,4), 3(21,6), 3(1,2,17), 4(20,3,11), 2(5,19), 2(22,15), 2(23,12)	{13,14,24,25}
1(3,25)	1(3,25)	-	5(9,8,10), 7(18,7,16,4), 3(21,6), 3(1,2,17), 4(20,3,11), 2(5,19), 2(22,15), 2(23,12)	{13,14,24,25}
1(8,25)	1(8,25)	-	5(9,8,10), 7(18,7,16,4), 3(21,6), 3(1,2,17), 4(20,3,11), 2(5,19), 2(22,15), 2(23,12)	{13,14,24,25}
1(11,25)	1(11,25)	4(20,3,11)+1(11,25)=5(20,3,11,25)	5(9,8,10), 7(18,7,16,4), 3(21,6), 3(1,2,17), 5(20,3,11,25), 2(5,19), 2(22,15), 2(23,12)	{13,14,24}
1(12,24)	1(12,24)	2(23,12)+1(12,24)=3(23,12,24)	5(9,8,10), 7(18,7,16,4), 3(21,6), 3(1,2,17), 5(20,3,11,25), 2(5,19), 2(22,15), 3(23,12,24)	{13,14}
1(13,15)	1(13,15)	-	5(9,8,10), 7(18,7,16,4), 3(21,6), 3(1,2,17), 5(20,3,11,25), 2(5,19), 2(22,15), 3(23,12,24)	{13,14}
1(14,13)	1(14,13)	1(14,13)	5(9,8,10), 7(18,7,16,4), 3(21,6), 3(1,2,17), 5(20,3,11,25), 2(5,19), 2(22,15), 3(23,12,24), 1(14,13)	{ }
1(15,14)	1(15,14)	2(22,15)+1(15,14)+1(14,13)=4(22,15,14,13)	5(9,8,10), 7(18,7,16,4), 3(21,6), 3(1,2,17), 5(20,3,11,25), 2(5,19), 4(22,15,14,13), 3(23,12,24)	{ }

Table 5. Result of Reverse-Constructive Algorithm for 12×19 and 20×20 Problem

(a) 12×19

Product	Machine											
	1	2	3	5	6	4	8	9	11	10	7	12
1	1					2	3					
2	1	2			3	4	5				6	
3	1	2				3	5	6			4	
4	1					2		4			3	
7					1	2	3	4				
8		3	1	2	4	5	6	7				
9			1	2	3	4	5	6				
10			1		2	3	4					
5	1				2			5		3	4	
6					1		4	5		2	3	
11					1							2
12					1					3	2	
13										1	2	
14									1		2	3
15									1	2	3	4
16									1	3	2	
17									1	2		
18									1			2
19									1		2	3

(b) 20×20

Product	Machine																			
	10	1	12	13	14	16	17	5	4	15	6	7	9	18	3	11	2	8	19	20
9		4	5										2	1		3				
14	2	3															4	1		
17		2	3										1							
20	2	3	4											1						
6					2	3	4	5								1				
7						2	3	1												
15				1	2	3	4													
5									1	2	3	4								
8				2					5	1		3	4							
13							4			3	1	2								
16										1	3	2							4	
1		2	1										3	4						5
12		5	4					3					1	2						
2															3	1	2			
4	4														1	2	3			
11					2										3			1		
19								4							1	3	2			
3																		1	3	2
10																		3	1	2
18	4																	1	2	3

Table 6. Compare of Algorithm Performance for 8×20 Problem

(a) CASE

Product	Machine																			
	2	8	9	11	13	14	16	17	19	3	4	6	7	18	20	1	5	10	12	15
3	**2**	**2**	**3**	**2**	**2**	**3**	**2**	**1**	**2**											
1	**1**	**1**	**1**	**3**	**1**	**1**	**1**	**3**	**1**	2										
4										**5**	**2**	**2**	**2**	**1**	**1**			2		
7				1						**3**	**3**	**3**	**3**	**4**	**4**				2	
8										**4**	**4**	**4**	**1**	**3**	**5**					
2						2				**1**	**1**	**1**	**4**	**2**	**2**					
5								2				5				**2**	**2**	**3**	**1**	**1**
6			2											3		**1**	**1**	**1**	**3**	**2**

Cell No.	Machine	Product	P_k	N_{mk}	N_{tk}	$CMI(\%)$
1	(3,1)	{2,8,9,11,13,14,16,17,19}	9	1	9	11.1
2	(4,7,8,2)	{3,4,6,7,18,20}	6	7	18	38.9
3	(5,6)	{1,5,10,12,15}	5	1	5	20.0
$ACMI(\%)$	4.33/20=21.7%					
$OMI(\%)$	9/41=22.0%					
$ACUI(\%)$	100.0%					

(b) CLASS, GGA, CCA, RCA,

Machine	Product																			
	2	8	9	11	13	14	16	17	19	3	4	6	7	18	20	1	5	10	12	15
1	**1**	**1**	**1**	**3**	**1**	**1**	**1**	**3**	**1**	2										
3	**2**	**2**	**3**	**2**	**2**	**3**	**2**	**1**	**2**											
2						2				**1**	**1**	**1**	**4**	**2**	**2**					
4										**5**	**2**	**2**	**2**	**1**	**1**			2		
7				1						**3**	**3**	**3**	**3**	**4**	**4**				2	
8										**4**	**4**	**4**	**1**	**3**	**5**					
6			2												3	**1**	**1**	**1**	**3**	**2**
5								2					5			**2**	**2**	**3**	**1**	**1**

Cell No.	Machine	Product	P_k	N_{mk}	N_{tk}	$CMI(\%)$
1	(1,3)	{2,8,9,11,12,14,16,17,19}	9	5	9	55.6
2	(2,4,7,8)	{3,4,6,7,18,20}	6	9	18	50.0
3	(6,5)	{1,5,10,12,15}	5	2	5	40.0
$ACMI(\%)$	**10.0/20=50.0%**					
$OMI(\%)$	**16/41=39.0%**					
$ACUI(\%)$	100.0%					

www.kci.go.kr

Table 7. Comparison of RCA with Other Algorithms for 25×40 Problem

Cell No.	CASE		CLASS		CCA		RCA	
	Machine	Product	Machine	Product	Machine	Product	Machine	Product
1	14,13,15,22	18,32	9,8,10,18,7,4,16,1,2,17	1,2,5,7,10,15,16,17,19,21,22,28,30,36,38	8,10,9	10,19,21,22,28,38	8,10,9	10,19,21,22,28,38
2	18,4,7,16	1,5,7,16,17,30	24,20,3,11,25	3,9,12,13,14,33	18,7,16,4	1,5,7,16,17,30	18,7,16,4	1,5,7,16,17,30
3	19,5	8,15,23,24,31	21,6,5,19	8,11,23,24,25,29,31,35,40	21,6	11,25,27,29,35,40	21,6	11,25,27,29,35,40
4	20,3,11,25	3,9,13,14,33	23,12,22,15,14,13	4,6,18,20,26,27,32,34,37,39	25,1,2,17	2,12,36	25,1,2,17	2,12,36
5	21,6	11,25,27,29,35,40	-	-	20,3,11	3,9,13,14,33	20,3,11	3,9,13,14,33
6	23,12	4,6,20,26,34,37,39	-	-	5,19	8,15,23,24,31	5,19	8,15,23,24,31
7	1,2,17,24	2,12,36	-	-	22,15,14,13	18,32	22,15,14,13	18,32
8	9,8,10	10,19,21,22,28,38	-	-	23,12,24	4,6,20,26,34,37,39	23,12,24	4,6,20,26,34,37,39
$ACMI$(%)	46.5%		48.8%		**52.7%**		**52.7%**	
OMI(%)	26.3%		30.5%		**32.6%**		**32.6%**	
$ACUI$(%)	**83.5%**		22.4%		83.3%		83.3%	

Table 8. Comparison of Algorithms

문제	CASE				CLASS				CCA				GGA				RCA			
	No. of cells	ACMI	OMI	ACUI	No. of cells	ACMI	OMI	ACUI	No. of cells	ACMI	OMI	ACUI	No. of cells	ACMI	OMI	ACUI	No. of cells	ACMI	OMI	ACUI
8×20	3	21%	22%	100.0%	3	**50.0%**	**39.0%**	**100.0%**	3	**50.0%**	**39.0%**	**100.0%**	3	**50.0%**	**39.0%**	**100.0%**	3	**50.0%**	**39.0%**	**100.0%**
																	2	**50.0%**	**41.5%**	**75.0%**
12×19	-	-	-	-	2	**65%**	50%	-	-	-	-	-	2	**65%**	50%	-	2	**65.2%**	**52.7%**	**57.1%**
																	2	**65.2%**	**52.7%**	**57.1%**
20×20	-	-	-	-	4	**65%**	**41%**	-	-	-	-	-	-	-	-	-	6	**69.5%**	**40.7%**	**85.5%**
																	4	**64.9%**	**44.1%**	**63.6%**
25×40	8	46.5%	26.3%	83.5%	4	48.8%	30.5%	22.4%	8	**52.7%**	**32.6%**	**83.3%**	-	-	-	-	8	**52.7%**	**32.6%**	**83.3%**
																	4	**54.8%**	**36.8%**	**43.9%**

RCA를 수행하는 방법을, 하단은 $f=1$의 모든 흐름 행렬 데이터에 대해 셀 병합, 추가를 수행한 경우이다. 전자는 OMI와 $ACUI$의 균형점을 찾는 방법이며, 후자는 OMI를 극대화시키기 위해 최대 N_{mk}를 찾는 방법이다. OMI를 향상시키면 $ACMI$와 $ACUI$가 감소하는 경향이 있다. GGA는 $ACUI$를 적용하지 않아 $ACMI$와 OMI만으로 알고리즘 성능을 단순 비교하는 것은 의미가 없다. 일예로, 8×20에 대해 제안된 알고리즘을 $f=1$까지 모두 수행하면 2개의 셀을 얻으며, 각 셀의 기계 순서를 방향성을 가진 순서쌍인 (x,y)로, 제품은 무방향 집합인 $\{x,y\}$로 표기하면 $(2,4,7,8),\{3,4,6,7,18,20\}$과 $(6,5,1,3),\{1,2,5,8,9,10,11,12,13,14,15,16,17,18\}$를 얻는다. 이 경우, $ACMI=50.0\%$, $OMI=41.5\%$, $ACUI=75.0\%$로 OMI를 2.5% 증가시키기 위해 $ACUI$를 25% 희생시키는 결과를 얻는다.

표 8로 부터 ACMI, OMI와 ACUI의 3가지 성능평가 척도를 살펴보면 CASE는 모든 성능평가 척도에 대해 2개 데이터 모두에서 최적 해를 얻지 못하였으며, CLASS는 4개 데이터 중에서 8×20 데이터에 대해서만 ACMI, OMI와 ACUI의 최적 해를 얻었고, 12×19 데이터에 대해서는 ACMI만, 20×20 데이터에 대해서는 ACMI와 OMI 성능만 최적 해를 얻었다. 또한, 25×40 데이터에 대해서는 최적 해를 얻는데 실패하였다. CCA는 2개 데이터에 대해 ACMI, OMI와 ACUI 성능의 최적 해를 얻었으며, GGA는 8×20 데이터에 대해서는 ACMI, OMI와 ACUI 성능의 최적 해를 얻은 반면에, 12×19 데이터에 대해서는 ACMI 성능만 최적 해를 얻었다. 이로부터 기존의 알고리즘들은 일부 데이터에만 적용되고 ACMI, OMI와 ACUI의 성능 척도들 중 일부 척도에서만 최적 해를 얻어 일반화된 알고리즘이라 할 수 없다. 반면에, 제안된 RCA는 4개 데이터 모두에서 ACMI, OMI와 ACUI의 3개 성능척도 모두에 대해 최적 해를 얻어 보다 일반화된 알고리즘이라 할 수 있다.

V. Conclusions

본 논문은 셀의 기계 배치 순서를 결정하는 문제에 대해 체계적인 방법의 휴리스틱 방법을 제안하였다. 제안된 알고리즘은 최대 빈도수를 가진 2-차수 부분 방향 경로를 구하기 위해 최소신장트리를 구하는 Kruskal 알고리즘을 변형시켜 적용하였다.

제안된 알고리즘을 4개의 실험 데이터에 적용하고, 기존의 휴리스틱과 메타휴리스틱 방법들과 비교한 결과 3개의 평가기준에서 기존의 최적 알고리즘들과 동일한 결과를 구하였다.

기존의 알고리즘들은 일부 데이터에만 적용되고 일부 성능평가 척도에 대해서만 최적 해를 얻어 일반화된 알고리즘이라 할 수 없다. 반면에, 본 논문에서 제안된 알고리즘은 4개 데이터 모두에 대해 ACMI, OMI와 ACUI의 3개 성능척도 모두에 대해 최적 해를 얻어 보다 일반화된 알고리즘이라 할 수 있는 장점을 갖고 있다.

REFERENCES

[1] J. F. Gonçalves and M. G. C. Resende, "An Evolutionary

www.kci.go.kr

Algorithm for Manufacturing Cell Formation," Computers & Industrial Engineering, Vol. 47, Issue. 2-3, pp. 247-273, Nov. 2004.

[2] J. A. Diaz, D. Luna, and R. Luna, "A GRASP Heuristic for the Manufacturing Cell Formation Problem," Trabajos de Investigación Operativa, Vol. 20, Issue. 3, pp. 679-706, Oct. 2012.

[3] H. Nouri, S. H. Tang, B. T. H. Tuah, M. K. A. Ariffin, and R. Samin, "Metaheuristic Techniques on Cell Formation in Cellular Manufacturing System," Journal of Automation and Control Engineering, Vol. 1, No. 1, pp. 49-54, Jan. 2013.

[4] M. Murugan and V. Selladurai, "Formation of Machine Cells/Part Families in Cellular Manufacturing Systems Using an ART-Modified Single Linkage Clustering Approach - A Comparative Study," Jordan Journal of Mechanical and Industrial Engineering, Vol. 5, No. 3, pp. 199-212, Jun. 2011.

[5] E. Teymourian, I. Mahdavi, and V. Kayvanfar, "A New Cell Formation Model Using Sequence Data and Handling Cost Factors," Proceedings of International Conference on Industrial Engineering and Operations Management, Kuala Lumpur, Malaysia, Jan, 22-24, 2011.

[6] G. J. Nair and T. T. Narendran, "CASE: A Clustering Algorithm for Cell Formation with Sequence Data," International Journal of Production Research, Vol. 36, No. 1, pp. 157-179, 1998.

[7] M. Mutingi and G. C. Onwubolu, "Manufacturing System, Chapter 10. Integrated Cellular Manufacturing System Design and Layout Using Group Genetic Algorithms," pp. 205-222, Interopen.com, May. 2012.

[8] I. Mahadavi and B. Mahadevan, "CLASS: An Algorithm for Cellular Manufacturing System and Layout Design Using Sequence Data," Robotics and Computer-Integrated Manufacturing, Vol. 24, Issue 3, pp. 488-497, Jun. 2008.

[9] I. Mahadavi, B. Shirazi, and M. M. Paydar, "A Flow Matrix-based Heuristic Algorithm for Cell Formation and Layout Design in Cellular Manufacturing System," International Journal of Advanced Manufacturing Technology, Vol. 39, Issue 9-10, pp. 943-953, Nov. 2008.

[10] A. P. George, C. Rajendran, and, S. Ghosh, "An Analytical-Iterative Clustering Algorithm for Cell Formation in Cellular Manufacturing Systems with Ordinal-level and Ratio-level Data," International Journal of Manufacturing Technology, Vol. 22, Issue. 1-2, pp. 125-133, Sep. 2003.

[11] K. Y. Tam, "An Operation Sequence Based Similarity Coefficient for Part Family Formations," Journal of Manufacturing Systems, Vol. 9, pp. 55-68, 1988.

[12] G. Harhalakis, R. Nagi, and J. M. Proth, "An Efficient Heuristic in Manufacturing Cell Formation for Group Technology Applications," International Journal of Production Research, Vol. 28, No. 1, pp. 185-198, Mar, 1990.

Author

Sang Un Lee received the B. Sc. degree in avionics from the Korea Aerospace University in 1997. He received the M. Sc. and Ph. D. degrees in Computer Science from Gyeongsang National University, Korea, in 1997 and 2001, respectively. He is currently Professor with the Department of Multimedia Science, Gangneung-Wonju National University, Korea. He is interested in software quality assurance and reliability modeling, software engineering, software project management, neural networks, and algorithm.

www.kci.go.kr