Journal of the Society of Korea Industrial and Systems Engineering
Vol, 27, No. 3, pp.106-112, September 2004.

네트워크 분할 기법을 이용한 기계 그룹 형성 알고리즘

최 성 훈

상명대학교 공과대학 산업정보시스템공학전공

A Machine Cell Formation Algorithm Using Network Partition

Seong-Hoon Choi

Depart. of Industrial and Information Systems Engineering, Sangmyung University

This paper presents a new heuristic algorithm for the machine cell(MC) formation problem. MC formation problem is represented as an unbalanced k-way network partition and the proposed algorithm uses four stage-approach to solve the problem. Four stages are natural sub-network formation, determination of intial vertexes for each sub-network, determination of initial partition, and improvement of initial partition. Results of experiments show that the suggested algorithm provides near optimal solutions within very short computational time.

Keywords : group technology, machine cell formation, network, k-way partition

1. 서 론

기계 그룹(MC, machine cell) 형성 문제는 그룹 테크놀러지(GT, group technology)를 적용하기 위해 해결해야 하는 핵심 문제로 그동안 많은 연구가 진행되어 왔다[1,2].

기계 그룹 형성 문제는 분할 문제의 일종이다. 분할은 GT 분야 이외에 회로의 배치, 전체의 회로를 패키징 가능한 부분으로 나누는 문제, 병렬 논리 시뮬레이션을 위해서 각 부분간의 통신을 줄이면서 회로를 분할하는 문제 등 VLSI 설계 분야에도 다양하게 적용되고 있다[3].

Kumar 외 2인[4]이 지적한 바와 같이 MC 형성 문제는 완전 NP 문제이므로 규모가 큰 문제를 적절한 시간 내에 해결하기 위해서는 효율적인 발견적 기법에 기초한 해법을 적용하는 것이 현실적이라 할 수 있다.

본 연구에서는 제품별 작업 순서와 생산량을 네트워크로 표현하고 이 네트워크를 분할하여 기계 그룹 형성 문제의 해를 구하는 발견적 알고리즘을 제시한다. 제안 알고리즘은 Kernighan과 Lin의 $2m$ 개의 교점(vertex)을 갖는 네트워크를 m 개의 교점을 갖는 두 부분 네트워크으로 나누는 균형 이분할 알고리즘에 기반을 두고 있다[3,5,6].

그런데 본 연구에서 다루는 문제는 각 부분 네트워크의 교점 개수가 균등하지 않으며 그룹 개수가 여러 개인 불균형 다분할 문제이다. 먼저 기존 연구들은 불균형 문제를 해결하기 위해 교점 사이의 이동량이 0인 가상의(dummy) 교점을 도입하여 불균형 문제를 균형 문제로 변형하는 방법을 제시하고 있다[5,6]. 그러나 본 연구에서는 가상 교점을 도입하지 않고 불균형 문제를 다루는 알고리즘을 제안한다.

그리고 다분할 문제를 다루기 위해 Kernighan과 Lin[5]은 우선 이분할 알고리즘을 적용하여 $C*m$ 개의 교점을 2개의 부분 네트워크로 분할하고, 다시 각각에 대해 이분할 알고리즘을 적용하여 C개의 부분 네트워크로 분할하는 알고리즘을 제시하였다. 그러나 C개의 부분 네트워크를 결정하기 위해 내부 이동량이 가능한 한 최대가 되도록 이분할을 적용한 결과는 바로 다음 단계의 이분할에 좋지 않은 영향을 끼칠 수 있으므로 좋은 C분할 네트워크를 제시해준다는 보장이 없다. 본 연구에서는 C 분할 네트워크에 대한 초기해를 구하고 개선해 나가는

본 논문은 상명대학교 2004학년도 교내연구비 지원에 의하여 연구되었음.

방법을 적용하기로 한다.

본 논문의 전체 구성은 다음과 같다. 먼저, 2장에서 기계 그룹 형성 문제에 대한 네트워크 표현 방법에 대해 알아본다. 3장에서는 네트워크 분할을 이용한 기계 그룹 형성 알고리즘을 제시하고, 4장에서 제안 알고리즘의 성능을 평가하기 위해 몇 가지 문제에 대한 계산 결과를 제시한다. 마지막으로 5장에서는 결론을 제시한다.

2. 기계 그룹 형성 문제의 네트워크 표현

이분할 네트워크(bipartite network)는 교점들이 상호 배반적인 두 개의 집합으로 나누어지고 각 집합 내의 교점들 사이에는 호(edge)가 없는 형태이다[9]. 이분할 네트워크는 기계와 제품 사이의 관계를 표현하기 위해 널리 사용되어 왔다[4,7,8]. 그러나 이분할 네트워크는 작업 순서 정보를 표현할 수 없기 때문에 이분할 네트워크에 기반한 알고리즘은 기계 그룹간 이동량(inter-cellular moves)을 정확히 계산할 수 없다[2,9].

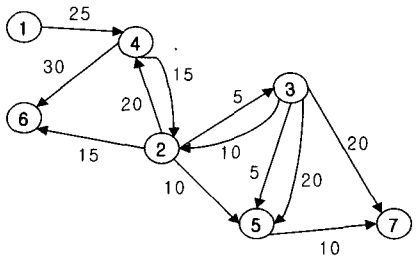
본 연구에서는 최[9]가 제안한 기계 사이의 이동량을 정확히 나타낼 수 있는 네트워크 표현 방법을 이용하기로 한다. <표 1>은 제품별 작업 순서와 생산량에 대한 간단한 예이다. <표 1>의 기계와 제품 사이의 흐름은 <그림 1>과 같이 방향이 있는 네트워크로 표현될 수 있다.

그런데 기계 그룹 형성 문제의 목적은 기계 그룹들 사이의 이동량을 최소화하는 기계 그룹들을 결정하는 것이므로 흐름의 방향은 본 연구의 관심사가 아니다. 따라서 본 연구의 목적을 위해서 <그림 1>은 <그림 2>와 같이 무방향 네트워크로 단순화될 수 있고 기계 그룹 형성 문제는 <그림 2>와 같이 표현된 무방향 네트워크의 분할 문제로 단순화된다.

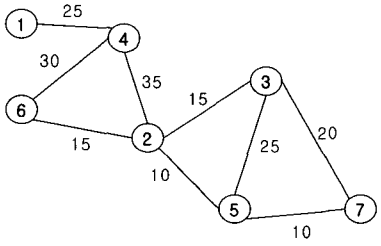
<표 1> 제품별 작업 순서와 생산량

제품 번호	작업 순서 (기계 번호)	생 산 량
1	5 - 3 - 7	20
2	1 - 4- 2 - 6	15
3	2 - 4 - 6	20
4	3 - 2 - 5 - 7	10
5	1 - 4 - 6	10
6	2 - 3 - 5	5

이제 연구 대상 무방향 네트워크를 $G=(V,E)$라 하자. V는 네트워크 G에 있는 교점(vertex)의 집합을 E는 네트워크 G에 있는 호(edge)의 집합을 나타낸다. 그러면 상호배반적인 기계 그룹 형성이 본 연구의 목적이므로, 연구 대상 문제는 $\bigcup_{k=1}^{C} V_k = V$ (단, C는 기계 그룹의 수)이고, i와 $j(i \neq j)$에 대하여 $V_i \cap V_j = \varnothing$ 이고 $L \leq |V_k| \leq U$(단, L은 각 기계 그룹에 속하는 기계 대수의 하한치이고 U는 각 기계 그룹에 속하는 기계 대수의 상한치, $L \leq U$)를 만족하는 $V_k(k=1,2,...,C)$를 구하는 것이다.



<그림 1> 제품별 작업 순서와 생산량에 대한 네트워크 표현



<그림 2> 단순화된 무방향 네트워크 표현

3. 네트워크 분할 기반 기계 그룹 형성 알고리즘

본 절에서는 먼저 C개의 부분 네트워크로 구성되는 초기해를 결정하고, 이 해를 개선시켜 나가는 알고리즘을 제안한다. 전체 알고리즘은 자연 부분 네트워크 구성 단계, 부분 네트워크의 초기 교점 선정 단계, 초기해 결정 단계, 초기해 개선 단계의 총 4단계로 구성된다.

각 단계에서 사용되는 용어에 대한 정의는 부록에 제시되어 있다.

자연 부분 네트워크 구성 단계

강제로 분할을 하지 않은 초기의 무방향 네트워크 $G=(V,E)$에 자연스럽게 구성되어 있는 부분 네트워크가 있는 지를 파악하는 절차이다. G의 교점 집합 V에 대해 서로 연결되어 있는 교점들의 부분 집합들로 구분하면 자연 부분 네트워크를 알아낼 수 있다.

본 연구에서는 상향식 나무 구조 형성법[10]을 이용하여

자연 교점 집합을 구성한다. 구체적인 절차는 다음과 같다.

[단계 1] 자연 부분 네트워크 구성

절차 1. 네트워크 G에 대하여 상향식 나무 구조 형성법을 적용한다. 구성된 자연 교점 집합의 개수를 C_0로 놓는다.

절차 2. 만일 $C_0 = C$ 이면, 절차 3으로 간다. 그렇지 않으면, [단계 2]로 간다.

절차 3. 만일 모든 i(i=1,2,3,...,C)에 대하여 $L \leq V_i \leq U$ 조건을 만족하면, 최적해가 구해졌으므로 종료한다. 그렇지 않으면, [단계 2]로 간다.

부분 네트워크의 초기 교점 선정 단계

각 부분 네트워크의 교점 집합에 대한 초기 교점을 선정하기 위해 자연 부분 네트워크 구성 결과를 이용하고, 각 부분 네트워크의 초기 교점들이 네트워크 G 전체에 골고루 퍼져 있도록 하기 위해 서로 거리가 멀리 떨어져 있는 교점들을 선정하는 방법을 적용한다.

부분 네트워크 초기 교점 선정 단계의 구체적인 절차는 다음과 같다.

[단계 2] 부분 네트워크의 초기 교점 선정

절차 1. 만일 $C_0 \geq C$ 이면, 절차 1.1로 간다. 만일 $1 \leq C_0 < C$ 이면, 절차 1.2로 간다. 만일 $C_0 = 0$ 이면, 절차 1.3으로 간다.

1.1 임의로 C 개의 자연 교점 집합을 선택하고, 각 집합에서 임의로 교점을 한 개씩 선택하여 초기 교점 집합 V_0에 추가한다. 즉, 각 교점 집합의 초기 교점으로 놓는다. [단계 3]으로 간다.

1.2 각 자연 교점 집합에서 임의로 교점을 한 개씩 선정하여 V_0에 추가한다. 즉, 처음 C_0 개 교점 집합의 초기 교점으로 놓는다. 절차 2로 간다.

1.3 $i^* = Argmin[i \mid i \in V, \{\sum_{j=1}^{M} h_{ij}\}]$.

1.4 i^*를 V_0에 추가한다(즉, i^*를 첫 번째 교점 집합의 초기 교점으로 놓는다). 절차 2로 간다.

절차 2. 아래 식을 이용하여 i^*를 구한다.

$$i^* = Argmax[i \mid i \in (V - V_0), [j \in V_0, min\{p_{ij}\}]]$$

(단, p_{ij}는 Dijkstra의 최단 경로 알고리즘을 적용하여 구한다[11].)

절차 3. i^*를 V_0에 추가한다.

절차 4. 만일 $|V_0| = C$이면, [단계 3]으로 간다. 그렇지 않으면 절차 2로 간다.

초기해 결정 단계

본 연구에서는 V_i에 속하지 않으면서 V_i의 교점(들)과 직접 연결되어 있는 교점(부분 네트워크 i의 경계 교점이라 부름)들의 이동량 합이 큰 부분 네트워크 i^*에 새로운 교점의 추가를 시도하는 방법을 적용한다.

마찬가지로 $B_{i^*}^X$ 의 원소(교점) 중에서 이동량이 많은 원소를 우선 선택하는 방법을 적용한다. 구체적인 절차는 다음과 같다.

[단계 3] 초기해 결정

절차 1. VL, VU, B_i^X, b_{ik}, 그리고 f_i^X를 구한다.

절차 2. i^*를 구한다. 만일 $VL \neq \varnothing$ 이면, $Argmax[i \mid i \in VL\{f_i^X > 0\}]$, 그렇지 않으면 $Agrmax[i \mid i \in VU, \{f_i^X > 0\}]$를 구한다.

절차 3. 만일 $i^* \neq null$ 이면 k^*를 구하고, (단, $[k \mid k \in B_{i^*}^X, \{b_{i^*k}\}]$) 절차 5로 간다. 그렇지 않으면 절차 3.1~3.2를 수행한다.

3.1 만일 $VL \neq \varnothing$ 이면 VL에서 임의의 원소를 선택하여 i^*로 놓는다. 그렇지 않으면 UL에서 임의의 원소를 선택하여 i^*로 놓는다.

3.2 IN에서 임의의 원소를 선택하여 k^* 로 놓는다. 절차 4으로 간다.

절차 4. 아래의 내용을 수행한다.

4.1 $V_{i^*} = V_{i^*} \cup \{k^*\}$, $IN = IN - \{k^*\}$

4.2 $TCM = TCM + b_{i^*k^*}$

4.3 $B_{i^*}^X = B_{i^*}^X \cup NB_{i^*k^*}^X - \{k^*\}$

4.4 $b_{i^*k} = \sum_{j \in V_{i^*}} a_{jk}$ for $k \in B_{i^*}^X$

4.5 $f_{i^*}^X = \sum_{k \in B_{i^*}^X} b_{i^*k}$

절차 5. 만일 $IN = \varnothing$ 이면 [단계 4]로 간다.

절차 6. 만일 $|V_{i^*}| \geq L$이고 $V_{i^*} \in VL$이면 VL에서 i^*를 삭제하고 VU에 넣는다. 만일 $|V_{i^*}| \geq U$ 이고 $V_{i^*} \in VU$이면 VU에서 i^*를 삭제한다. 절차 2로 간다.

초기해 개선 단계

개선 단계는 Kernighan과 Lin[5]의 균형 이분할 알고

리즘에 기초하고 있으므로, 먼저 그들이 제시하는 방법에 대해 알아보기로 한다. 이분할 알고리즘의 기본 아이디어는 다음과 같다. 네트워크를 임의의 두 부분 네트워크로 분할하고, 각 그룹에 속해 있는 교점을 한 개씩 선택하여 교환할 경우 개선이 이루어지면, 즉 그룹 사이의 이동량이 줄어들면 교환을 수행한다. 해를 개선시키는 두 교점 쌍이 없을 때까지 반복한다. 이 방법은 매우 단순하지만, Kernighan과 Lin[5]이 경험적으로 보여준 것처럼 최적해를 구할 가능성이 매우 높다. 따라서 본 연구에서는 두 교점 또는 세 교점을 동시에 교환하는 방법은 고려하지 않고 한 번에 한 교점을 교환해 나가는 하는 단순한 방법을 적용한다.

Kernighan과 Lin[5]은 교환 대상 교점을 선택하기 위해 E_x, I_x, D_x를 정의하고 아래의 [Lemma 1]을 증명하였다. E_x, I_x, D_x는 부록의 E_x^{ij}, I_x^i, D_x^{ij}에서 i와 j가 1과 2로 고정되어 이분할만을 다루는 특수한 경우이다.

[Lemma 1] 임의의 $x \in V_1$와 $y \in V_2$를 고려하자. 만일 x와 y가 교환될 경우, 이득은 정확히 $D_x + D_y - 2a_{xy}$이다.

[Lemma 1]을 이용하면 모든 (x, y) 쌍에 대한 이득 중에서 양의 최대값을 갖는 x와 y를 교환해가는 방법을 적용할 수 있다. 그러나 다음의 [Property 1]을 적용하면 모든 쌍에 대해 이득을 계산할 필요는 없다. 이득을 계산하기 위해 많은 연산이 필요하므로 계산 대상을 줄이게 되면, 알고리즘의 실질적인 계산 시간을 줄일 수 있다.

[Property 1] 특정 두 부분 네트워크에서 만일 이득 $D_x + D_y - 2a_{xy}$이 양의 최대이면, $(x, y) \in CBxy$이다. 즉, 교점 x와 y 둘 중에서 최소한 한 개는 경계 교점이다.

(증명) 만일 x와 y 두 교점 모두 경계 교점이 아니면, $a_{xy} = 0$이고 $E_x = E_y = 0$이다. $I_x \geq 0$이므로, $D_x = E_x - I_x \leq 0$. 마찬가지로 $D_y = E_y - I_y \leq 0$. 따라서 이득은 양수가 될 수 없다.

앞에서 지적한 바와 같이 기계 그룹 형성 문제는 불균형 다분할 네트워크이다. 따라서 균형 이분할 네트워크 알고리즘을 적용하기 위해서는 불균형 문제에 대한 접근법이 제시되어야 한다. 기존 연구들[5,6]은 교점간 이동량이 0인 가상 교점을 도입하여 불균형 문제를 균형 문제로 변형하는 방법을 제시하고 있다. 그러나 본 연구에서는 가상 교점을 도입하지 않고 불균형 부분 네트워크 사이에 단순히 이동을 시도하는 방법을 제안한다.

$x \in V_i$이고 $|V_i| > L$과 $|V_j| < U$를 만족하는 x, i, j에 대해 $D_x^{ij} > 0$이면 교점 x를 부분 네트워크 i에서 j로 이동하면 이득이 발생하므로 이동하는 것이 바람직하다. 따라서 아래의 식을 적용하여 이동 대상 교점 x를 선정하고 이동을 실시하면 번거롭게 가상 교점을 도입할 필요가 없다.

$$(x^*, i^*, j^*) = Argmax[(x,i,j) \mid x \in V, \; |V_i| > L, |V_j| < U, \{D_x^{ij} > 0\}]$$

본 연구에서 초기 해의 개선을 위해 적용하는 개선 단계에 대해 정리하면 다음과 같다. 먼저 특정 그룹 내의 한 교점을 다른 그룹으로 이동시킬 경우, 해의 개선이 이루어지면 그 교점을 이동시키는 방법을 적용한다. 더 이상 이동에 의한 개선이 없으면, 이동량이 많은 두 부분 네트워크을 선정하여 교점의 교환을 통해 해의 개선을 시도한다. 더 이상의 개선이 없으면 다시 이동을 시도한다. 구체적인 절차는 다음과 같다.

[단계 4] 초기해 개선

절차 1. SA_x, E_x^{ij}, I_x^i, D_x^{ij}를 구한다.

절차 2. 만일 $max\{D_x^{ij}\} < 0$이면 중지한다.

절차 3. 이동

3.1 아래의 값들을 구한다.

$$(x^*, i^*, j^*) = Argmax[(x,i,j) \mid x \in V_i, \; |V_i| > L, |V_j| < U, D_x^{ij} > 0\}]$$

그리고, $g3 = D_{x^*}^{i^*j^*}$.

3.2 만일 $x^* = null$이면 절차 4로 간다. 그렇지 않으면 x^*를 V_{j^*}에 추가하고 V_{i^*}에서 삭제한다.

3.3 $TCM = TCM + g$

3.4 $z \in \{x^*\} \cup VA_{x^*}$에 대하여 SA_x, E_x^{ij}, I_x^i, D_x^{ij}를 갱신하고 절차 3.1로 간다.

절차 4. 교환

4.1 VE와 E^{ij}를 구하고 $TCM_0 = TCM$로 놓는다.

4.2 아래의 식으로 (i^*, j^*)를 구한다.

$$(i^*, j^*) = Argmax\,[(i,j) \mid (i,j) \in VE, E^{ij}\}]$$

4.3 만일 $(i^*, j^*) \neq null$ 이면 아래의 값들을 구하고 절차 4.4로 간다. 그렇지 않으면, $TCM > TCM_0$이면 절차 3으로 가고, 그렇지 않으면 중지한다.

$$(x^*, y^*) = Argmax[(x,y) \mid (x,y) \in CBxy, \; \{D_x^{i^*j^*} + D_y^{i^*j^*} - 2a_{xy}\}]$$

그리고, $g = D_{x^*}^{i^*j^*} + D_{y^*}^{i^*j^*} - 2a_{x^*y^*}$

4.4 만일 $g \leq 0$이면 VE에서 (i^*, j^*)를 제거하고 절차 4.2로 간다. 그렇지 않으면 $TCM = TCM + g$로 놓고, x^*를 V_{i^*}에서 V_{j^*}로, y^*를 V_{j^*}에서 V_{i^*}로 이동한다.

4.5 $z \in \{x^*, y^*\} \cup VA_{x^*} \cup VA_{y^*}$에 대하여 E_x^{ij}, I_x^i, D_x^{ij}, B_i^j, VE, 그리고 E^{ij}를 갱신한다. 절차 4.2로 간다.

4. 제안 알고리즘의 성능 평가 실험

본 절에서는 제안 알고리즘의 성능을 평가하기 위해 C 언어로 개발한 프로그램을 2GHz 속도의 팬티엄 IV PC에서 실행하였다. 비교 대상은 최[9]가 제안한 기계 그룹 형성 문제의 0-1 정수계획 모형으로 구한 최적해이다. 2개의 문제 집합에 대해 목적함수 값과 실행 시간을 비교하였다. 참고로 정수계획 모형을 풀기 위해 미국 LINDO 사의 LINGO 소프트웨어 패키지를 적용하였다.

첫 번째 문제 집합은 최[9]의 논문에서 적용한 15 문제들로 구성된다. 이 문제들은Groover[12]의 저서에 소개되어 있는 기계 23대, 제품 20개의 기계-제품 행렬에 기초하여 각 제품의 생산량은 모두 1단위로 가정하고, 작업 순서는 무작위로 발생시켜서 작성되었다. (L,U,C) = (8,15,2)로 정했을 때, 15개 문제 각각 대한 최적해의 목적함수 값(즉, 기계 그룹 내의 이동량 합)과 실행 소요 시간에 대한 결과가 <표 2>에 제시되어 있다.

해의 차이에 대한 99% 신뢰구간은 [-0.38%, 4.30%]로 유의수준 1%(즉, $\alpha = 0.01$) 하에서 두 방법의 차이가 없음을 알 수 있다. <표 2>에서 알 수 있듯이 최적해를 구하기 위해 5초~35초 정도의 시간이 소요되었다. 반면에 본 연구에서 제안한 알고리즘을 적용할 경우 모든 문제에 대해 1초 미만의 시간 내에 해를 도출할 수 있었다. 특히 15개 문제 중에서 9개에 대해서는 최적해를 구할 수 있었다.

두 번째로 적용한 문제 집합은 King[13]과 Gupta[14]의 논문에 제시되어 있는 43개의 제품과 16대의 기계 문제를 이용하여 준비된 15개 문제이다. (L,U,C) = $(L,U,5)$로 정했을 때(단, L과 U는 <표 3>의 하한과 상한 값임.), 15개 문제 각각 대한 최적해의 목적함수 값과 실행 소요 시간에 대한 결과가 <표 3>에 정리되어 있다.

문제 집합 2의 해의 차이에 대한 99% 신뢰구간은 [0.24%, 1.79%]로 유의수준 1% 하에서 두 방법의 차이가 있으나, 해의 차이에 대한 평균은 1.01%로 매우 작음을 알 수 있다. 이 문제 집합의 경우, 15개 문제 중에서 8개에 대해 제안 알고리즘을 적용하여 최적해을 구할 수 있었다. 실행 시간을 비교하여 보면, 최적해의 경우 54초 ~ 155 초가 소요된 반면에 제안 알고리즘의 경우 모두 1초 미만의 시간이 소요되었다.

위의 두 문제 집합을 이용하여 최적해와 제안 알고리즘의 해를 비교한 결과, 제안 알고리즘이 매우 짧은 시간 내에 우수한 해를 제시하고 있음을 알 수 있다.

5. 결 론

본 연구에서는 기계 그룹 형성 문제를 네트워크의 불균형 다분할 문제로 표현하고 현실적인 해법으로 균형 네트워크 이분할 기법에 기초한 발견적 알고리즘을 제안하였다. 불균형 네트워크 문제를 풀기 위해 기존 연구와 달리 이동량 0의 가상 교점을 도입하지 않았으며, 그룹간 교환 대상 교점에 대한 성질을 유도하여 비교 대상 교점의 수를 줄일 수 있었다.

제안 알고리즘에 대한 성능 평가를 통하여 매우 짧은 시간 이내에 우수한 해를 얻을 수 있음을 알 수 있었다.

<표 2> 문제 집합 1의 계산 결과 비교

문제 번호	최 적 해		근 사 해		해의 차이 (%)
	목적 함수 값	실행 시간(초)	목적 함수 값	실행 시간 (초)	
1	78	5	78	< 1	0.00
2	74	18	73	< 1	1.35
3	75	26	70	< 1	6.67
4	77	4	76	< 1	1.30
5	77	8	77	< 1	0.00
6	77	19	77	< 1	0.00
7	80	5	80	< 1	0.00
8	78	14	78	< 1	0.00
9	79	10	77	< 1	2.53
10	74	35	65	< 1	12.16
11	74	25	74	< 1	0.00
12	76	12	76	< 1	0.00
13	81	8	81	< 1	0.00
14	79	6	79	< 1	0.00
15	75	7	71	< 1	5.33

<표 3> 문제 집합 2의 계산 결과 비교

문제			최적해		근사해		해의 차이 (%)
번호	하한	상한	목적 함수 값	실행 시간 (초)	목적 함수 값	실행 시간 (초)	
1	0	7	88,919	54	88,919	< 1	0.00
2	1	7	88,479	55	88,479	< 1	0.00
3	2	7	85,394	34	85,394	< 1	0.00
4	3	7	66,837	152	65,695	< 1	1.71
5	0	6	84,542	77	84,542	< 1	0.00
6	1	6	84,542	55	84,542	< 1	0.00
7	2	6	81,127	60	78,892	< 1	2.75
8	3	6	66,837	150	65,695	< 1	1.71
9	0	5	77,297	92	77,297	< 1	0.00
10	1	5	77,297	70	77,297	< 1	0.00
11	2	5	76,754	81	76,754	< 1	0.00
12	3	5	66,837	128	65,695	< 1	1.71
13	0	4	70,057	155	68,345	< 1	2.44
14	1	4	70,057	128	68,345	< 1	2.44
15	2	4	70,057	90	68,345	< 1	2.44

부 록

M 기계 총수, 즉 집합 V의 원소 개수

a_{ij} 기계 i와 j 사이의 이동량

$|A|$ 집합 A의 원소의 개수

H $M \times M$ 행렬, 만일 $a_{ij} > 0$ 이면 $h_{ij} = 1$, 그렇지 않으면 $h_{ij} = 0$.

p_{ij} 행렬 H를 인접 행렬(adjacency matrix)로 했을 경우의 교점 i에서 교점 j까지의 최단 거리

IN 아직 할당되지 않은 교점(들)의 집합,
$IN = V - \bigcup_{i=1}^{C} V_i$.

B_i 부분 네트워크 i의 경계 교점(V_i에 속하지 않으면서 V_i의 교점(들)과 연결되어 있는 교점) 집합.

TCM Total Intra-cellular Moves의 약자로 그룹 내 이동량의 총합

B_i^X B_i에 속하면서 (즉 그룹 i의 경계 교점이면서) 아직까지 어느 그룹에도 할당되지 않은 교점의 집합, $B_i^X = B_i \cap IN$.

NB_{ik}^X k가 V_i에 추가될 때, 새로이 B_i^X에 추가되는 교점(들)의 집합. 즉,
$NB_{ik}^X = \{j; a_{kj} > 0, j \in IN\}$

VL $\{i; |V_i| < L, i = 1, 2, \ldots, C\}$.

VU $\{i; |V_i| < U, i = 1, 2, \ldots, C\}$.

b_{ik} 부분 네트워크 i와 B_i^X 소속 교점 $k(k \in B_i^X)$ 사이의 이동량. 즉, $b_{ik} = \sum_{j \in V_i} a_{jk}$.

f_i^X 부분 네트워크 i와 B_i^X의 모든 교점 사이의 이동량 합, $f_i^X = \sum_{k \in B_i^X} b_{ik}$.

VA_x 교점 $x(x \in V)$와 인접하고 있는 교점들의 집합,

SA_x 교점 $x(x \in V)$와 인접하고 있는 부분 네트워크들의 인덱스 집합.

B_i^j 부분 네트워크 i의 경계 교점들 중에서 부분 네트워크 j에 속하는 교점들의 집합,
$B_i^j = \{y; y \in B_i \cap V_j\}$.

B_{ij} 부분 네트워크 i와 부분 네트워크 j의 사이의 경계 교점들의 합집합, $B_{ij} = B_i^j \cup B_j^i$.

$CBxy$ $\{(x, y); x \in B_j^i \text{ and } y \in V_j \text{ or } x \in V_i \text{ and } y \in B_i^j\}$

VE 경제 교점(들)이 존재하는 부분 네트워크들의 인덱스 쌍의 집합. 즉,
$VE = \{(i, j); B_j^i \neq \varnothing, i < j, \text{ and } i, j = 1, 2, \ldots, C\}$

E_x^{ij} 교점 x의 외부 이동량(부분 네트워크 i의 교점 x와 부분 네트워크 j 사이의 이동량 합),
$E_x^{ij} = \sum_{v \in (V_j \cap VA_x)} a_{xv}$.

I_x^i 교점 x의 내부 이동량(부분 네트워크 i의 교점 x와 내부의 교점(들) 사이의 이동량 합),
$I_x^i = \sum_{v \in (V_i \cap VA_x)} a_{xv}$.

D_x^{ij} 교점 x의 외부 이동량과 내부 이동량의 차이,
$D_x^{ij} = E_x^{ij} - I_x^i$.

E^{ij} i와 j $((i, j) \in VE)$ 사이의 외부 이동량 합,
$E^{ij} = \sum_{x \in B_j} E_x^{ij} = \sum_{y \in B_i} E_y^{ji}$.

참고문헌

[1] H.M. Selim, R.G. Askin, and A.J. Vakharia, "Cell Formation in Group Technology: Review, Evaluation and directions for Future Research", *Computers and Industrial Engineering*,Vol. 34, pp. 3-20, 1998.

[2] Y.K. Won and K.C. Lee, "Group Technology Cell Formation Considering Operation Sequences and Production Volumes", *International Journal of Production Research*, Vol. 39, pp. 2755-2768, 2001.

[3] 조진윤과 조준동, "전력량 감축을 위한 회로 분할", *정보과학회논문지(A)*, 제25권11호, pp. 1327-1338, 1998.

[4] K. Ravi Kumar, Andrew Kusiak, and Anthony Vannelli, "Grouping of Parts and Components in Flexible Manufacturing Systems", European Journal of Operational Research, Vol. 24, pp. 387-397, 1986.

[5] B.W. Kernighan and S. Lin, "An Efficient Heuristic Procedure for Partitioning Graphs", T*he Bell System Technical Journal*, Vol. 49, pp.291-307, 1970.

[6] C. Park, K. Park, and M. Kim, "An Efficient Algori-

thm for K-Way Partitioning of Graph", *IEEE, TENCON* '87, Seoul, pp. 463-466, 1986.

[7] M.P. Chandrasekharan and R. Rajagopalan, "An Ideal Seed Non-Hierachical Clustering Algorithm for Cellular Manufacturing", *international Journal of Production Research*, Vol. 24, pp. 451-464, 1986.

[8] J.R. King and V. Narkornchai, "Machine-Component Group Formation in Group Technology", *international Journal of Production Research*, Vol. 20, pp. 117-133, 1982.

[9] 최성훈, "기계 그룹 형성 문제의 최적해", *한국산업경영시스템학회*, 2004. (논문 심사중)

[10] 이재규, C로 배우는 알고리즘, 도서출판 세화, 1998.

[11] 강맹규, 네트워크와 알고리즘, 박영사, 1995

[12] M.P. Groover, Automation, Production Systems and Computer Aided Manufacturing, Prentice-Hall, Englewood Cliffs, NJ, 1980.

[13] J.R. King, "Machine-Component Grouping in Production Flow Analysis: an Approach Using a Rank Order Clustering Algorithm", *international Journal of Production Research*, Vol. 18, pp. 213-232, 1980.

[14] T. Gupta and H. Seifoddini, "Production Data Based Similarity Coefficient for Machine-Part grouping Decisions in the Design of a Cellular Manufacturing System", *International Journal of Production Research*, Vol. 28, pp. 1247-1269, 1990.