기계-부품군 형성문제의 사례를 통한 유전 알고리즘의 최적화 문제에의 응용†

한용호* · 류광렬**

Genetic Alogorithms for Optimization : A Case study of Machine-Part Group Formaiton Problems†

Yong Ho Han* · Kwang Ryel Ryu**

ABSTRACT

This paper solves different machine-part group formation (MPGF) problems using genetic algorithms to demonstrate that it can be a new robust alternative to the conventional heuristic approaches for optimization problems. We first give an overview of genetic algorithms: Its principle, various considerations required for its implementation, and the method for setting up parameter values are explained. Then, we describe the MPGF problem which are critical to the successful operation of cellular manufacturing or flexible manufacturing systems. We concentrate on three models of the MPGF problems whose forms of the objective function and/or constraints are quite different from each other. Finally, numerical examples of each of the models described above are solved by using genetic algorithms. The result shows that the solutions derived by genetic algorithms are comparable to those obtained through problem-specific heuristic methods.

† 이 논문은 1994년도 한국학술진흥재단의 지방대육성과제 연구비에 의하여 연구되었음

* 부산외국어대학교 경영정보학과 부교수
** 부산대학교 컴퓨터공학과 조교수

1. 서 론

이제까지 상당수의 최적화 문제가 NP-complete으로 알려져 있다. 이러한 유형의 현실적인 대규모 문제를 해결하기 위해서는 최적해를 보장하는 수리적 기법은 더 이상 사용할 수 없고, 각 문제의 고유한 특징을 나름대로 이용하는 휴리스틱 기법에 의존할 수 밖에 없다. 하지만 휴리스틱 해법의 적용은 특정한 문제에 한정되고, 그 해법의 성능에 대해 아무런 보장도 할 수 없다는 문제점을 가지고 있다. 최근들어 전문가 시스템은 이러한 휴리스틱 기법들을 보완할 수 있는 하나의 효과적인 수단으로 평가받고 있으나 이 또한 적용 분야에 따라 어려운 지식획득의 과정을 매번 거쳐야 한다는 문제점을 가지고 있다. 따라서 최적화 문제의 성격에 관계없이 일관되게 적용할 수 있는 기법이 있다면 이는 매우 유익한 도구가 될 것이다.

인공지능의 한 기법인 유전 알고리즘 (Genetic Algorithm: 이하 GA로 약칭함)은 생태계의 적자생존 및 유전법칙에 그 원리의 바탕을 둔, 최적해에 대한 adaptive 탐색 알고리즘으로, 휴리스틱 기법에 비해 다음과 같은 상대적 특성을 지니고 있다. 첫째, 휴리스틱 기법은 문제의 유형에 따른 고유한 해법을 일일이 개발해야만 하나, GA는 거의 모든 유형의 문제에 대하여 일관성있게 적용할 수 있다. 둘째, 휴리스틱 기법에서는 주어진 문제의 상황이나 가정이 조금만 달라져도 그 전의 해법이 더 이상 적용될 수 없는 경우가 허다하나, GA에서는 변경내용에 따라 목적함수 및 제약조건의 코딩 내용만 수정하면 계속적인 적용이 가능하다. 세째, unimodal 문제로 부터 multimodal 문제, 그리고 combinatorial 문제에 이르기까지 광범위한 문제영역에 걸쳐, 비록 최적해의 여부는 알 수 없어도 좋은 해를 제공하는 robustness를 지니고 있다. 전통적인 최적화 기법들(해석적 기법, 열거법, 무작위 탐색법 등)과 큰 차이를 보이고 있는 GA의 이러한 특성들로 인하여 GA는 대규모 NP-complete 문제의 최적해에 대한 탐색해법으로서의 가치를 지니고 있다.

외국의 경우 GA는 생물학, 컴퓨터공학, 이미지 처리 및 패턴인식, 엔지니어링, OR, 사회과학 등 광범위한 분야에서 성공적으로 응용되고 있다. 아직 국내에서는 경영과학, 산업공학 혹은 MIS 분야에서의 응용사례 발표가 매우 적은 실정이다. 따라서 최적화 문제에 대한 일반적 해법으로서의 가치를 평가해 보기 위하여 GA의 적용사례연구를 수행할 필요성이 제기된다.

본 연구에서는 GA의 적용대상으로 NP-complete로 알려진 기계-부품군 형성(Machine-Part Group Formation: 이하 MPGF로 약칭함) 문제를 선정한다. MPGF 문제란 group technology의 개념을 이용하는 셀 제조방식(cellular manufacturing) 혹은 FMS에서 발생하는 문제로서, 군(group)간 상호의존성이 최소화되도록 전체 부품 및 전체 기계를 각각 몇 개의 부품군(part family) 및 기계군(machine cell)으로 할당하는 것이다.

MPGF문제에 대하여 지난 20년간 다양한 해법들이 제시되었다. [7, 21, 29] 그 중 주요한 해법들은 정식화 방법 및 해법에 따라 [표 1]과 같이 분류될 수 있다.

이들 각 연구에서는 공정순서(routing), 사용공구(tooling) 혹은 설계속성 면에서의 유사성이나, 작업, 자재취급, 투자비와 같은 비용을 반영한 각기 다른 여러가지 목적함수를 고려하였다. 제약조건으로는 부품의 처리시간, 기계의 가동률, 기계의 처리능력, 복수 경로, 부품의 작업순서, 생산량, 로트 크기, 군의 크기 및 군의 위치 등을 고려하였다.

〈표 1〉 MPGF 문제의 정식화 및 해법의 분류

정식화 방법	해 법
행 렬	Similarity coefficient methods [10, 22] Sorting-based algorithms [5, 6] Bond energy algorithm [25] Cluster identification algorithm [30]
수 리 계 획 법	[18, 27, 32]
그 래 프 이 론	[19, 30]
새 로 운 접 근 방 법	전문가 시스템 [20] Fuzzy 이론 [8] 신경망 이론 [16] 유전 알고리즘 [1, 3]

최근들어 MPGF 문제에 대한 새로운 접근방법으로서 GA를 적용한 사례들이 발표되었다. Venugopal & Narendran[31]은 목적함수가 2개이고 군의 크기에 제약이 있는 MPGF 문제에 GA를 적용하여 비교적 좋은 해를 구하였다. 선지웅[1]은 휴리스틱과 혼합된 GA기법을 제시하였다. 이들 두 연구에서는 MPGF 문제의 특정한 한가지 모형만을 다룸으로써, GA를 최적화 문제에 대한 하나의 일반적인 접근방법으로서가 아니라, 특정한 문제에 적용되는 구체적인 하나의 해법으로만 소개하고 있다. GA의 적용방법 면에서 볼 때, Venugopal & Narendran[31]은 세대변천을 함에 따라 illegal string의 발생을 허용하는 문제점을 가지고 있다. 선지웅[1]은 이 문제를 해결하기 위하여 PMX 연산자[14]를 이용하는 GA와 휴리스틱의 혼합 기법을 제시하였다. 이에 반해 본 연구에서는 GA의 robust함을 보이기 위해 의도적으로 휴리스틱을 배제한 순수한 GA기법을 사용한다.

본 연구의 목적은 MPGF 문제의 여러가지 모형들에 대하여 GA의 적용사례를 연구함으로써 일반적인 대형 최적화 문제에 대하여 GA의 광범위한 응용가능성을 제시하는 데 있다. 구체적으로 말하자면, 첫째, 기존의 휴리스틱 해와의 비교를 통하여 GA의 robust한 정도를 보이고, 둘째, GA 기법이 여러 모형들에 일관되게 적용될 수 있음을 보이고자 한다.

제2장에서는 GA의 일반적인 방법론적 특성을 설명한다. 제3장에서는 MPGF 문제와 이 문제의 세가지 모형을 설명한다. 제4장에서는 이들 모형의 각 예제에 대하여 GA의 실행방법, 각종 파라미터 값의 설정 및 실행결과를 설명한다. 제5장 결론에서는 연구내용을 요약하고 향후 연구분야를 제시한다.

2. Genetic Algorithm의 개관

이 장에서는 GA의 원리, 최적화 문제에 GA를 적용하는 데 필요한 일반적 연구내용, 그리고 GA의 실행에 필요한 통제 파라미터의 내용을 기술한다.

2.1 Genetic Algorithm의 원리

GA는 생태계의 적자생존 및 유전법칙에 바탕을 둔 일종의 휴리스틱 탐색 알고리즘이다. 즉, 최적화할 목적함수 $f(X_1, X_2,\cdots,X_m)$에 대한 임의의 후보해(candidate solution)의 집단(population)을 가지고 출발하되, 유전법칙을 적용하여 이를 진화, 발전시켜 가는 반복적 과정을 거치는 알고리즘이다. GA의 일반적인 구조는 [그림 1]과 같다. 후보해의 집단은 매 세대(generation)마다 evaluation, selection, recombination (crossover와 mutation으로 구성)의 세 단계를 거친다. 이하에서는 GA의 원리를 각 단계별로 나누어 살펴 본다.

```
procedure GA
begin
  t=0 :
  initialize P(t);
        --P(t) is the population at time t
  evaluate structures in p(t);
  while (termination condition not satisfied) do
  begin
    t=t+1;
    select P(t) from P(t-1);
    recombine structures in P(t);
    eveluate structures in P(t);
  end
end.
```

[그림 1] Genetic Algorithm의 구조

2.1.1 Initialization

최적화할 대상 목적함수의 한 후보해는 하나의 bit string으로 부호화되어 표현되며, 다수의 bit string들이 모여 집단을 형성하게 된다. 주어진 집단이 존재하는 시점을 그 집단의 세대(generation)라 부른다.

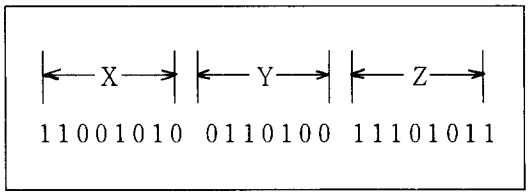
세대 t 에서 N 개의 bit string으로 구성된 집단 $P(t)$를 다음과 같이 표시할 수 있다.

$$P(t) = \langle S_1(t),\ S_2(t),\cdots,S_N(t)\rangle$$

여기서 집단 P(t)를 구성하고 있는 각 원소 $S_i(t)$는 일정한 길이의 bit string을 나타내며, 목적함수 $f(X_1, X_2,\cdots,X_m)$의 각 후보해(독립변수의 값)에 대응한다.

초기집단 $P(0)$는 보통 무작위로 선택되어 지나, 휴리스틱하게 선택된 bit string들을 포함할 수도 있다. 어느 경우이든 초기집단 $P(0)$는 다양한 bit string들을 포함해야 한다.

간단한 예로서, 삼변수함수 $f(X,Y,Z)$의 최대화 문제의 경우, 후보해를 bit string으로 부호화 하는 한가지 방법을 [그림 2]에 나타내었다. 변수 X, Y, Z가 각각 취할 수 있는 값의 영역을 8 bit의 이진수가 취하는 값의 범위로 mapping 하는 것이 곧 부호화에 해당한다. [그림 2]의 string은 결국 X, Y, Z가 각각 어떤 특정값을 가짐을 표시하는 것으로, 함수 f의 최대치를 찾기위한 하나의 후보해이다. GA는 이런 후보해들을 대개 수십내지 수백개 무작위적으로 생성시켜 만든 초기집단을 가지고 출발한다.



[그림 2] 부호화된 후보해의 bit string

여기서 개별적인 bit를 *feature* (생물학에서의 gene에 대응)라고 하며, 이것들이 1 혹은 0의 구체적인 값을 취했을 때, 이를 *feature value* (생물학에서의 allele에 대응)하고 부른다. 이들 bit들이 모여 X, Y, Z와 같이 하나의 의미있는 값

을 나타낼 때 이것을 string (생물학에서의 chromosome에 대응)이라 부른다. (X, Y, Z)에 대응하는 string들을 모아서 이를 structure (생물학에서의 genotype에 대응)라고 부른다. structure를 decoding하여 (X, Y, Z)와 같이 하나의 후보해로 나타낸 것을 solution (생물학에서의 phenotype에 대응)이라고 부른다. 이들 용어는 〈표 2〉와 같이 요약될 수 있다.

[표 2] 생물학 용어 및 GA 용어의 대비

Natural system	Genetic algorithm
gene	feature
allele	feature value
chromosome	string
genotype	structure
phenotype	solution

2.1.2 Evaluation

이 단계에서는 집단 $P(t)$에 있는 각 structure들을 평가(Evaluation)한다. 집단내의 각 structure $S_1, \cdots, S_N$에 대한 목적함수 또는 적합함수 (objective function, fitness function) f의 값 $f(S_1), \cdots f(S_N)$을 그 structure의 적합도(fitness)라고 한다. 평가란 각 structure에 대하여 바로 이 적합도를 구하는 것이다. [그림 2]와 관련한 예의 경우, 적합도는 string이 표시하는 X, Y, Z의 값에 대응하여 함수 f가 취하는 값이 된다.

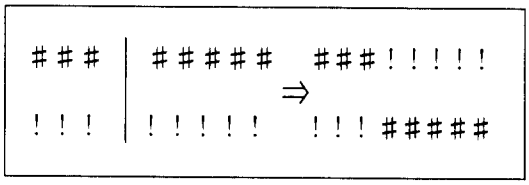
이와 같이 구해진 각 structure의 적합도는 그 structure의 상대적 우열정도에 대한 정보를 지니고 있다. 이 값들은 차세대에서 offstring을 차별적으로 생성시키는 데 사용된다.

2.1.3 Genetic Operators

집단의 세대변천은 "유전법칙"을 적용하여 새로운 세대로 집단을 계속 진화시켜 감으로써 이루어 지며, 이를 통하여 최적해를 탐색한다. 유전법칙은 기본적으로 다음의 세가지 유전 연산자 (genetic operation)에 의해 적용된다.

(1) Reproduction (Selection) : 매 세대마다 집단내의 각 structure에 대하여, 적합도가 높을수록 많은 수의 자기 복제를 허용함으로써, 차세대에 그 세력이 확장되게 한다.

(2) Crossover : reproduction이후 변화된 세력판도 하에서 집단내의 structure들이 무작위적으로 모두 쌍을 짓게 한후, 쌍들이 서로 일부 bit string을 교환하게 함으로써, 새로운 쌍의 structure를 만들어 낸다. [그림 3]은 crossover 연산을 구체적으로 보인 것이다. 실제로는 모든 string 쌍들에 대해 공히 crossover를 적용하는 것은 아니다. 실험적으로 약 60%의 쌍들에 대해서만 적용한 경우 GA의 성능이 좋은 것으로 알려져 있다[14]. Crossover를 위한 string 절단 지점은 보통 무작위적으로 선택된다.



[그림 3] Crossover Operation의 예

(3) Mutation : 집단에 무작위성을 부여하기 위해 string내의 임의의 한 bit를 0에서 1로 혹은 1에서 0으로 바꾸어 준다. 탐색의 방향이 지나치게 무작위적으로 바뀌지 않아야 하므로, mutation은 조심스럽게 매우 낮은 확률로 적용된다.

이상의 세가지 유전 연산을 기본으로, 보다 복잡하고 고도화된 연산들이 개발되어 활용되고 있으나[14], 이들이 궁극적으로 추구하는 것은 적합도(fitness)가 높은 structure들 간의 유사성을 파악하고 선택적으로 잘 조합함으로써 최적의 structure, 즉 최적해를 찾고자 하는 것이다.

2.1.4 Termination Condition

집단이 세대변천을 계속함에 따라 그 집단을 구성하고 있는 structure 들 사이에는 그것들이 취하는 bit string 의 형태가 서로 유사해지는 경향이 있다. 이에 따라 집단의 세대변천을 끝내는 데에는 두가지 방법이 있다. 첫째는 전 세대에 걸쳐 평가할 structure의 총 수를 사전에 지정하는 방법이며, 둘째는 집단내 모든 structure 들 사이에 존재하는 bit string 형태상의 유사 정도를 나타내는 척도를 미리 정의하고 이 척도의 값이 사전에 설정한 값을 초과할 때에는 population이 converge한 것으로 간주하여 실행을 끝내는 방법이다. 일반적으로, 이 두가지 방법을 병행 사용하여, 두가지 방법에 의한 조건중 어느 하나가 먼저 만족될 때 실행을 마치도록 한다.

2.1.5 GA사용의 잇점

(1) 목적함수 형태의 무제약성 : 전통적인 분석적 최적화 기법들의 경우 문제에 주어진 결정변수들을 직접 사용해야 하므로, 대상 함수의 연속성 또는 미분가능성 등의 제약이 따르지만, GA의 경우 결정변수들을 일단 부호화하여 bit string들로 이루어진 공간에서 탐색을 하므로 그러한 제약을 받지 않는다.

(2) parallel한 탐색 : branch-and-bound search와 같은 enumerative한 탐색기법들 및 기타 해석적 최적화 기법들은 모두 하나의 point에서 출발하여 다음의 point를 찾아가는 point-to-point 탐색방법들로서 local optima에 빠지기 쉬운 결점이 있다. GA는 집단(population)을 가지고 출발하여 적정 수준의 다양성(diversity)을 유지해 가며, parallel하게 우수 후보해들간에 정보를 교환해 가면서 탐색을 하므로 local optima를 피할 가능성이 높은 대단히 robust한 방법이다.

(3) 문제에 관한 지식 불필요 : 대개의 최적화 방법들은 효율적인 탐색을 위해 대상문제 특유의 지식 (problem-specific knowledge)을 필요로 한다. 휴리스틱 기법이 바로 그러한 경우의 단적인 예이다. 그러나, 매번 문제가 주어질때마다 필요한 지식을 이끌어 낸다는 것이 항상 용이 하지만은 않으며 경우에 따라서는 불가능하기 조차 하다. GA는 각 후보해 bit string에 대해 목적함수의 값만을 필요로 하고, 일체의 다른 정보를 요구하지 않으므로, 매우 광범위한 적용이 가능하다. 사실 고난도의 문제일수록 문제의 특성파악이 어렵고, 따라서 필요지식 혹은 휴리스틱을 이끌어 내기가 어렵다. GA는 특히 이런 경우 다른 여느 최적화 방법보다 강력한 효능을 발휘할 수 있다.

2.2 GA에 의한 최적화 연구의 내용

가스파이프라인 시스템의 제어[13], Job Shop Scheduling[11], 다목적최적화[12, 28], 통신망 링크규모의 최적화[9], 다중차량경로 선정문제[3], 집합분할문제 [24], 최대흐름문제[26] 등의 최적화문제에 대하여 GA의 적용사례가 발표된 바 있다.

이와 같이 GA는 적용영역이 광범위한 대신, 어떤 특정문제의 해결을 위해 전용으로 특별히 고안된 기법과 비교할 때, 그 문제에 관한 한 전

용기법을 능가하는 성능을 보이지 못하는 경우가 있을 것이다. 따라서 앞서 2. 1. 3절에서 설명한 유전 연산(genetic operation)을 기본으로, 주어진 문제의 특성에 따라 적절한 변형을 가하여 보다 효율적인 유전 연산자를 개발하는 것이 중요한 연구 내용이 된다. 문제의 특성에 적합한 유전 연산자의 개발이라 함은 문제 특유의 지식이나 휴리스틱에 의존하는 전용기법의 개발과는 분명히 구별된다. GA를 문제 특성에 맞게 조율한다는 것은 어디까지나 문제의 유형적 특성(예를 들어, unimodal 문제인가 혹은 multimodal 문제인가 등)의 파악에 기본을 둔 것이지 문제의 내용적 특성까지 고려해야 하는 것은 아니기 때문이다. 문제를 부호화하는 방식이 어떠냐에 따라 recombination 연산자가 문제 특유의 형태를 취할 수는 있다.

이하에서는 최적화 문제를 풀기위해 GA를 문제의 유형적 특성에 적합화시키는데 필요한 연구 내용과 방법을 조목별로 설명한다.

2.2.1 제약조건식의 처리

GA는 기본적으로 제약조건식이 없는 목적함수에 대한 최적해 탐색기법이다. 따라서 제약조건식이 있는 최적화 문제는 제약조건식의 위배 정도에 따라 〈표 2〉과 같이 목적함수의 값을 조정하는 Penalty Method[14]를 사용함으로써, 제약조건식이 없는 최적화 문제로 전환시킬 수 있다.

〈표 3〉 Penalty method의 내용

제약조건식이 있는 최적화 모형	제약조건식이 없는 최적화 모형
Minimze f(X) s.t. $g_i(X) \geq 0$, i=1, 2,…,m X : vector	Minimize $f(X)+r\sum_{i=1}^{m} \Phi [g_i(X)]$ Φ : penalty 함수 r : penalty 상수

2.2.2 Illegal String의 처리

후보해를 부호화한 bit string은 유전 연산의 적용에 의해 불법(illegal) string으로 변하는 수가 종종 있다. 예를 들어, 0에서 5까지의 정수값을 취할 수 있는 변수 X를 3 bit의 string으로 표시할 경우, 1 1 0 (6에 해당)이라는 불법 string이 crossover에 의해 생성될 수도 있는 것이다. 이런 문제의 해결책으로는, 단순히 penalty를 부여하여 도태시키는 방법과 합법 string으로의 강제 변환 방법 (forcing technique) 등이 주로 사용되고 있다.

2.2.3 적정수준의 다양성 유지전략 개발

탐색 초기단계에 집단내에 상대적으로 특출한 string이 reproduction 과정에서 세력확대를 과도하게 하여 local optimum으로 수렴하는 현상을 막기위해 적절한 대응전략이 필요하다. 타당한 후보 전략으로는 첫째, static하게 혹은 generation 변천 추이에 따라 dynamic하게 string의 적합도를 재분배하는 fitness rescaling 방법, 둘째, 매 generation마다 다양성(diversity)의 정도를 측정하여 crossover의 확률을 dynamic하게 변동시키는 방안 등을 들 수 있다.

2.2.4 Crossover의 변형 방안 연구

Crossover의 기본취지는 두 string간의 부분적 정보 교환에 있는데, 애초에 후보해를 어떤 방식으로 부호화했느냐에 따라 crossover의 구체적 방법이 달라진다. 후보해의 부호화 자체가 주어진 문제의 특성에는 부합이 되지만 crossover에 의해 illegal string이 생성되고 forcing technique만으로는 그것을 legal string으로 변환시킬 수 없을 경우에는 새로운 crossover 연산자의 사용을 고려해야 한다. 우선 앞의 [그림 3]의 예에서와 같이 가장 기본적인 crossover 연산자의 사용을 검토할 수 있으며, 그렇지 못한 경우 PMX, oder crossover(OX), cycle crossover(CX) 등 [14] 이미 개발된 다른 crossover 연산자를 사용해야 할 경우가 있다. 심지어는 새로운 recombination 연산자를 고안할 필요성이 생기는 경우도 있다.

2.3 GA실행을 위한 통제 파라미터 값의 설정

GA를 실행하기 위해서는 여러가지 통제 파라미터의 값을 사전에 설정해야 한다. 적절한 파라미터의 값의 선정여부는 실행시간 및 최종해의 성능에 큰 영향을 미친다. [4] GA package로 널리 쓰이고 있는 GENESIS[15]를 중심으로 한 통제 파라미터의 내용은 다음과 같다.

(1) Experiment Number : 주어진 최적화 문제에 대한 독립 실행 횟수

(2) Total Trial : 한 번의 독립 실행에서 평가할 총 structure 수의 상한선

(3) Population Size : 한 집단을 구성하는 structure의 수

(4) Structure Length : 하나의 structure를 구성하는 bit의 수

(5) Crossover Rate : bit당 crossing point 수의 기대치

(6) Mutation Rate : string의 한 bit가 mutation을 겪을 평균확률

(7) Generation Gap : 매세대마다 교체되는 집단내 structure의 비율 ($0<G\leq 1$)

(8) 'elitist' 선택전략 : 현 세대에서 적합도가 가장 높은 structure는 차세대에서도 항상 살아남도록 하는 전략 (이는 crossover나 mutation으로 인해 가장 좋은 structure가 사라질 가능성에 대비하고자 하는 것이다.)

다음 (9)-(11)의 세 파라미터는 termination condition에 관련된다.

(9) Maximum Bias : 하나의 독립실행을 끝내기 위한 기준으로 사용하기 위하여 사전에 설정한 string position들의 평균 bias 값의 최소치. 여기서

A_i=한 집단내 string position i 에서 이진값 0을 취하는 structure의 갯수,

B_i=한 집단내 string position i 에서 이진값 1을 취하는 structure의 갯수,

N=한 집단내 structure 의 수,

K=structure내 bit string 의 수를 나타낼 때,

string position i의 bias

$$=\max(A_i, B_i)/N \quad (i=1,2,\cdots,K)$$

한 집단의 평균 bias

$$=\sum_{i=1}^{K}\max(A_i, B_i)/NK$$

으로 정의된다.

세대변천중 어느 집단의 평균 Bias가 Maximum bias 를 초과할 때 실행을 끝내게 된다.

(10) Convergence Threshold : 각 string position에 대하여 하나의 allele(0 혹은 1)이 converge했다고 판정하기 위하여 사전에 설정한 최소 bias 값. 각 string position의 평균 bias가 이 값을 초과할 때 하나의 allele이 converge했다고 판정한다.

(11) Maximum Convergence : 하나의 독립실행을 마치는 데 필요한 converge한 allele의 최소 갯수. 세대 변천중 어느 집단의 converge한 allele의 수가 maximum convergence 를 초과할 때 실행을 끝내 게 된다.

적정수준의 다양성을 유지하기 위한 전략으로 적합도(fitness)가 일정수준 이하인 후보해에 대하여 0의 적합도 값을 부여함으로써 이를 차세대의 선택대상에서 완전히 제외시켜 버리는 Sigma Scaling 방법[14,15]이 있다. 이 방법의 사용을 위해서는 (12)의 파라미터가 필요하다.

(12) Sigma Scaling Factor : 집단내 적합도의 표준편차에 대한 적절한 승수값 (보통 1과 3 사이의 값을 취한다.)

이러한 파라미터 값이 함수의 최적화에 미치는 영향을 파악하기 위하여 De Jong이 5개 유형의 함수를 대상으로 광범위하게 수행한 실험연구 결과가 [14]에 정리되어 있다. 이 연구결과는 지금까지 GA 실행시 파라미터 값의 설정에 대한 이정표의 구실을 하고 있다.

3. MPGF 문제

이 장에서는 GA의 성능(performance)을 사례를 들어 구체적으로 평가하는 데 있어 그 적용대상이 되는 기계-부품그룹 형성(Machine-Part Group Formation: MPGF) 문제 및 이 문제의 세가지 모형을 설명한다.

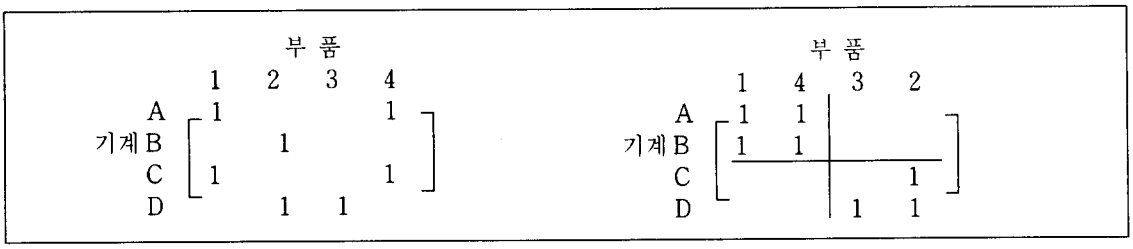
3.1 MPGF 문제의 설명

Group technology 개념을 이용하는 셀 제조방식(Cellular manufacturing)은 제조시스템의 생산성을 향상시키기 위한 주요 요소로서 인식되고 있다. 이 시스템은 가공대상이 되는 전체 부품을 유사 부품군으로, 전체 기계를 여러 기계군으로 그룹화함으로써 그룹내 가공수량을 크게하고, 공정의 흐름을 단순화시켜 궁극적으로 개별생산의 형태에서 대량생산의 효과를 기대하고 있다. 또한 FMS 운영시 그 구성요소인 FMC(flexible manufacturing cell)간 부품의 이동빈도가 최소화되도록 기계 및 자재운반수단(Robot, AGV 등)을 각 군으로 할당하는 개념적 설비배치계획이 필요하다.

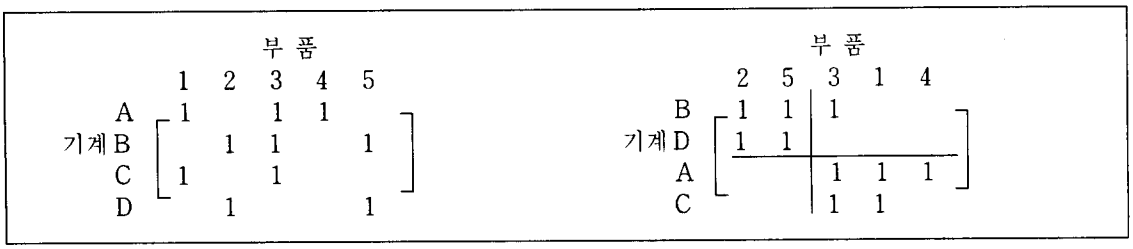
MPGF문제는 이와 같은 셀 제조방식이나 FMS의 운영시에 발생하는 문제로서, 여러가지 변형된 형태의 문제가 있으나 그 중 가장 기본적인 문제의 내용은 다음과 같다. 먼저 각 부품의 공정경로 정보를 나타내는 0-1 기계-부품 빈도행렬(machine-part incidence matrix) $[a_{ij}]$이 입력데이터로서 주어진다. 이 빈도행렬에서 1(0)의 값을 가지는 요소 a_{ij}는 기계 i가 부품 j를 처리하기 위하여 사용됨(사용되지 않음)을 나타낸다. 이 때 주어진 각종 제약조건들을 만족하면서, (가급적 모든 부품들이 할당된 군내에서만 처리되어) 군(group)간 상호의존성이 최소화되도록 전체 부품 및 전체 기계를 각각 몇 개의 부품군(Part Family) 및 기계군(Machine Cell)으로 할당하는 것이다. 이 문제는 NP-complete으로 알려져 있다. [18, 20]

일반적으로 최초의 빈도행렬이 주어졌을 때, 기계셀이나 부품군은 잘 식별되지 않는다. 이 문제를 푼다는 것은 하나의 군집화 해법(clustering algorithm)을 사용하여 주어진 초기의 빈도행렬을 가능한 한 블록대각(block diagonal) 구조형태로 전환시키는 것을 의미한다. 예를 들어, MPGF문제는 [그림 4]의 왼쪽 행렬이 입력정보로 주어 졌을 때, 각 기계 및 부품들을 2개의 군 중 하나에 할당해야 하고, 하나의 군은 각각 2-3개의 기계 및 부품으로 구성되어야 한다는 제약

조건하에, [그림 4]의 오른쪽 행렬을 찾는 것이다. 최적해에 대응하는 빈도행렬이 [그림 4]에서와 같이 완전한 블럭대각행렬을 취하는 경우도 있으나, [그림 5]와 같이 불완전한 블럭대각행렬을 얻는 경우가 일반적이다.



[그림 4] 완전한 블럭대각행렬을 구할 수 있는 MPGF문제



[그림 5] 완전한 블럭대각행렬을 구할 수 없는 MPGF문제

3.2 모형의 설정

MPGF문제에 대하여 빈도행렬의 형태, 군의 수의 사전결정여부, 목적함수 및 제약조건의 내용면에서 차이가 나는 여러가지 모형이 제시되었다. [7, 29] 본 연구에서는 그 중 GA의 실행대상으로 〈표 4〉와 같이 특성이 다른 세가지 모형을 설정함으로써 GA가 robust함을 보이고자 한다.

〈표 4〉 세가지 모형의 특성

특 성	모 형 1	모 형 2	모 형 3
빈도행렬의 형태	· 이진 혹은 실수값	· 이진값	· 실수값
그룹 수의 결정	· 사전에 주어짐	· 사전에 주어짐	· 사후에 도출됨
목적함수에 내용	· 부품의 군간이동 빈도의 최소화	· 예외부품 처리 비용의 최소화	· 부품의 군간이동 빈도의 최소화
제약조건의 종류	· 1가지(셀의 규모)	· 1가지(셀의 규모)	· 4가지 제약조건
모형의 원천	· Kumar 등[18] · 정성진 등[2]	· King[17] · Kuma & Vannelli[19] · Wei & Gaither[32]	· Kusiak[20]

3. 2. 1 모형 1

이 모형은 MPGF문제의 가장 기본적인 모형으로 다음 가정들을 포함하고 있다.

(1) 각 부품의 가중치(예를 들어, 생산량)가 동일할 수도 있고 상이할 수도 있다.

(2) 기계-부품 빈도행렬 $[a_{ij}]$과 목표로 하는 군의 수 k가 입력 데이터로서 사전에 주어진다.

(3) 한 군에 속하는 기계댓수와 부품갯수의 합은 지정된 상, 하한선 (l, u) 사이에 있어야 한다.

이 때, 부품의 군간 총 이동빈도를 최소화시키는 기계군 및 부품군을 형성하고자 한다. Kumar 등[18]은 이 문제를 일단 그래프의 최적 k-분할(decomposition) 문제로 정식화하였다. 그리고, 다음과 같은 기호를 정의하여,

$x_{ij}=$ { 0 node i가 subgraph j에 포함되지 않는 경우
1 node i 가 subgraph j 에 포함되는 경우

$l(u)$ = 각 subgraph에 포함이 가능한 node의 최소갯수(최대갯수)

n= 기계댓수(m)와 부품갯수(n)의 합

k = 목표로 하는 subgraph의 수

이 문제를 다음과 같이 0-1 2차형 정수계획 모형으로 정식화하였다.

$$\text{Max} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \sum_{p=1}^{k} a_{ij}\, x_{ip}\, x_{jp}$$

$$\text{s.t} \sum_{j=1}^{k} x_{ij}=1,\ \forall\ i=1,2,\cdots,n$$

$$1 \le \sum_{i=1}^{k} x_{ij} \le u\ \forall\ j=1,2,\cdots,k$$

$$x_{ij}=0 \text{ or } 1$$

여기서 첫번째 제약식은 각 node는 반드시 하나의 subgraph에만 속해야 함을 나타낸다. 두번째 제약식은 각 subgraph에 포함될 node수의 제약을 나타낸다.

이 모형에서 기계-부품 빈도행렬 $[a_{ij}]$을 0-1의 이진값대신 실수값으로 나타낼 경우 각 부품의 상이한 가중치(생산량)까지도 감안할 수 있다.[2]

1) 예제 1-1 : 본 모형의 첫번째 예로서 다음의 Kumar 등[18]의 문제를 들 수 있다. 23대의 기계와 20개의 부품(n = 43)에 대한 빈도행렬이 [그림 6]과 같이 주어지고 u = 29, l = 14일 때, 이 기계 및 부품들을 2개군(k = 2) 중 하나로 할당하는 최적 방법을 찾고자 한다.

2) 예제 1-2 : 본 모형의 두번째 예는 다음의 Kumar 등[18]의 문제이다. 9대의 기계와 15개의 부품(n = 11)에 대한 빈도행렬이 [그림 7]과 같이 주어지고 l = 5, u = 11 일 때, 이 기계 및 부품들을 3개군(k = 3)으로 할당하는 최적 방법을 찾고자 한다.

3) 예제 1-3 : 본 모형의 세번째 예는 다음의 정성진 등[2]의 문제이다. 5대의 기계와 6개의 부품(n = 11)에 대한 빈도행렬이 다음 [그림 8]과 같이 주어졌을 때, 이 기계 및 부품들을 2개군(k = 2)으로 할당하는 최적 방법을 찾고자 한다. 부품 1, 2, 3, 4, 5, 6에 대한 생산량의 비율은 1 : 2 : 1 : 1 : 3 : 2 이다. 구하고자 하는 두 군에 속할 기계 및 부품수의 균형을 유지하기 위해 l = 4, u = 9 로 설정한다.

3.2.2 모형 2

본 모형의 주요 가정은 다음과 같다.

(1) 각 부품의 가중치(예를 들어, 생산량)는 동일하다.

(2) 기계군간 이동이 불가피한 bottle neck 부품들은 외주로 처리된다.

(3) 기계-부품 빈도행렬 $[a_{ij}]$과 목표로 하는 군의 수 k의 값은 입력 데이터로서 사전에 주어진다.

부 품

기계	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43
1	1		1	1	1	1		1				1	1	1	1			1	1	
2			1			1			1	1		1								
3	1			1				1										1	1	1
4					1								1							
5		1							1		1					1			1	
6										1							1			
7		1					1		1	1	1					1	1			
8							1			1										
9							1			1										
10	1				1	1			1			1	1					1		
11	1		1		1							1								
12				1				1						1	1					1
13		1					1										1			
14		1														1				
15	1	1	1		1	1		1				1	1	1			1	1		1
16								1		1				1						1
17				1						1										
18			1						1			1	1			1	1	1		
19									1						1		1			
20												1	1			1				
21	1			1					1						1					
22	1		1	1		1		1	1	1				1	1					1
23	1			1					1						1					

[그림 6] 예제 1-1의 빈도행렬

부 품

기계	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1			1	1				1	1	1			1		1
2			1					1	1			1	1		
3	1						1								
4				1		1					1				
5		1		1				1	1	1			1		1
6						1								1	
7					1						1				
8					1						1				
9		1					1								

[그림 7] 예제 1-2의 빈도행렬

부 품

기계	1	2	3	4	5	6
1	1		1		3	
2		2		1		2
3	1			1	3	
4		2	1			
5		2		1		2

[그림 8] 예제 1-3의 빈도행렬

(4) 각 군에 할당되는 기계 수는 지정된 상한(u)을 초과할 수 없다.

이 때, 외주로 처리될 부품의 외주처리비용의 합을 최소화시키는 기계군 및 부품군을 형성하고자 한다. 다음과 같이 기호를 정의함으로써

입력 파라미터 :

I = 기계댓수

J = 부품갯수

K = 목표로 하는 군의 수

m = 하나의 군에 포함가능한 최대기계댓수

M = 임의의 큰 수

S_j = 부품 j를 제조하는 데 필요한 모든 기계들의 집합

P_j = S_j안에 있는 기계댓수의 합

T_i = 기계 i에 의해 처리되는 부품들의 집합

C_j = 부품 j의 외주처리비용

0-1 결정변수 :

X_{ik} = 1 기계 i가 군 k에 할당될 경우
0 그렇지 않은 경우

Y_{jk} = 1 부품 j가 군 k에 있는 기계에 의해 처리되는 경우
0 그렇지 않은 경우

B_j = 1 부품 j가 외주로 처리되는 경우
0 그렇지 않은 경우

이 문제는 다음과 같이 0-1 정수계획 모형으로 정식화될 수 있다.[32]

$$\text{Minumize} \sum_{j=1}^{J} C_j B_j$$

$$\text{s.t} \quad \sum_{k=1}^{K} X_{ik} = 1, \quad i=1,2,\cdots,I$$

$$1 \le \sum_{i=1}^{I} X_{ik} \le m, \quad k=1,2,\cdots,K,$$

$$Y_{jk} \le \sum_{i \in S_j} X_{ik} \le P_j Y_{jk}, \quad k=1,2,\cdots,K,$$

$$j=1,2,\cdots,J,$$

$$1+B_j \le \sum_{k=1}^{K} Y_{jk} \le 1 + MB_j, \quad j=1,2,\cdots,J,$$

$$B_j, X_{ik}, Y_{jk}=0 \text{ or } 1 \quad \text{for all } i, j, k.$$

첫번째 제약식은 각 기계가 단 하나의 군에 할당되어야 함을 나타낸다. 두번째 제약식은 하나의 군에 최대 m대의 기계까지 할당될 수 있음을 나타낸다. 세번째 제약식은 부품 j와 연관된 기계중 어느 하나가 군 k에 할당될 경우 $Y_{jk} = 1$ 이, 어느 기계도 군 k에 할당되지 않을 경우 $Y_{jk} = 0$ 이 되도록 한다. 네번째 제약식에서는 부품 j가 두가지 이상의 군에서 처리되어야 하는 bottle neck 부품이 되는 경우 (Y_{jk}의 값이 1이 될 경우) $B_j = 1$ 이 되도록 한다.

1) 예제 2-1 : 본 모형의 첫번째 예는 다음의 King[17]의 문제이다. 14대의 기계와 24개의 부품을 각각 2개, 3개, 4개의 그룹으로 분할하고자 한다. 하나의 군에는 각각 최대 10종류, 5종류, 4종류까지의 기계가 포함될 수 있다. 빈도행렬은 [그림 9]와 같다.

2) 예제 2-2 : 본 모형의 두번째 예는 다음의 Kumar & Vannelli[19]의 문제이다. 30대의 기계와 41개의 부품을 2개의 그룹으로 분할하고자 한다. 하나의 군에는 최대 20대까지의 기계가 포함될 수 있다. 빈도행렬은 [그림 10]과 같다.

3.2.3 모형 3

이 모형은 Kusiak[20]에 의해 제시된 것으로 다음과 같은 가정들을 취하고 있다.

(1) 각 부품의 가중치(예를 들어, 생산량)는 상이하다.

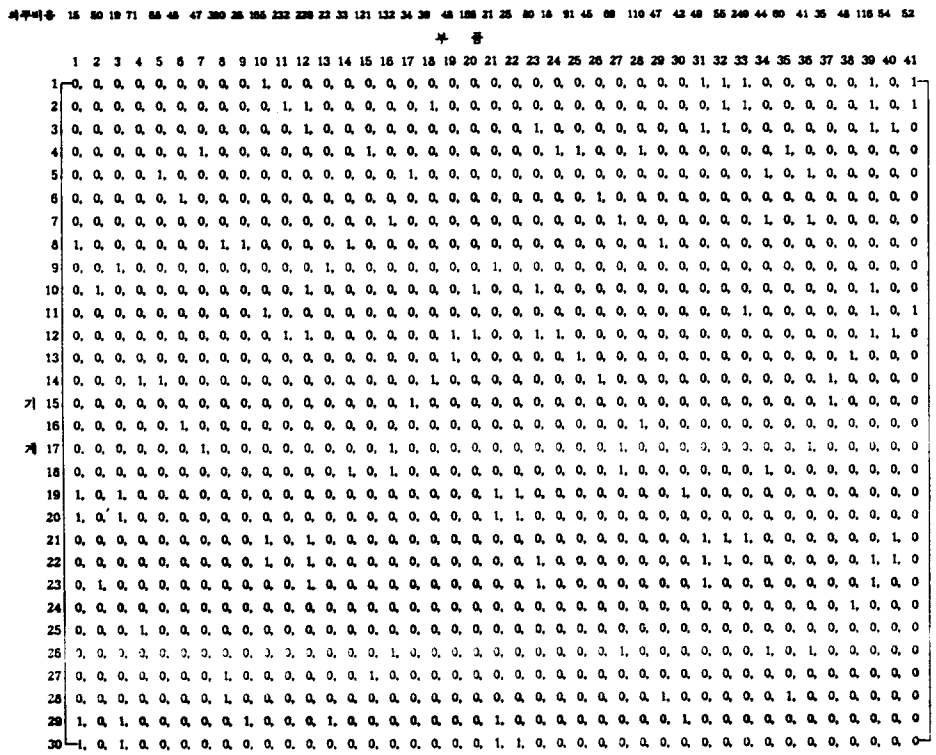
(2) 일부 부품들은 모든 제약조건을 충족시키지 못함으로 인하여 각 기계군에 대응하는 어떠한 부품군에도 포함되지 않을 수 있다.

(3) 다음의 4가지 제약조건을 만족해야 한다.

· 제약조건 1 : 각 기계군에 할당되는 기계댓

외주비용	7	8	15	5	23	6	12	8	20	22	13	19	13	24	6	17	3	21	16	8	15	7	5	28
	부 품																							
	1	2	3	4	5	6	7	8	9	0	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
4	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	0	0	1	0
5	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0
6	0	0	0	0	0	0	0	0	1	1	1	1	0	1	1	1	0	0	0	0	0	1	0	0
기 7	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
계 8	0	0	0	0	1	0	0	0	1	1	0	1	0	1	1	1	0	0	0	0	0	1	0	0
9	0	0	0	0	1	0	0	0	1	0	0	1	1	0	1	0	0	0	0	0	0	0	0	0
10	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
11	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1
12	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0
14	0	0	0	0	0	0	0	0	1	0	1	0	1	0	1	0	0	0	0	0	0	0	0	0

[그림 9] 예제 2-1의 빈도행렬



[그림 10] 예제 2-2의 빈도행렬

수는 지정된 상한을 초과할 수 없다.

· 제약조건 2 : 각 기계별 총 처리시간은 총 가용시간을 초과할 수 없다.

· 제약조건 3 : 각 기계군마다 두 종류의 부품 운반장비(즉, AGV 및 robot) 중 어느 한 장비를 사용해 야 한다. 따라서 어느 한 장비의 군내 총 사용빈도는 군내 총 가용빈도를 초과할 수 없다.

· 제약조건 4 : 기술적인 문제로 인하여 특정 기계들은 반드시 동일한 기계군에 할당되어야 한다. 이 때, 각 부품군에 할당되는 모든 부품의 가중치의 합이 최대화되도록, 첫째, 기계군의 수를 결정하고, 둘째, 각 기계군 및 부품군에 기계 및 부품들을 할당하고, 세째, 각 기계군에 적합한 부품운반장치를 선택해야 한다.

Kusiak[20]은 본 모형을 수리적 해법과 전문가 시스템의 혼합 기법을 사용하여 해를 구하였다. GA기법을 사용하는 데에는 수리적 모형수립이 반드시 전제될 필요가 없기 때문에 이를 생략한다.

1) 예제 3 : 본 모형의 예로서 다음의 Kusiak [20]의 문제를 들 수 있다. 7대의 기계와 12종류의 부품에 대한 입력데이터는 [그림 11]과 같다. 부품운반장치로는 AGV나 로보트의 두 종류가 있고 이들 장치의 가용빈도의 상한치는 각각 40 및 100이다. 하나의 군에 할당되는 기계는 3대를 초과할 수 없다. 기계 2와 5는 기술적인 이유로 반드시 동일 기계군에 할당되어야 한다.

부 품	1	2	3	4	5	6	7	8	9	10	11	12	상한
부품j의 생산량	60	55	15	13	10	20	8	18	40	5	31	30	
부품j 처리에 필요한 AGV의 이동빈도	11	30	2.5	–	6	10	–	6	7	15	18	14	40
부품j 처리에 필요한 robot의 이동빈도	11	30	5	3	6	15	10	12	7	–	36	28	100

기계별 부품의 처리시간

기계	부품 1	2	3	4	5	6	7	8	9	10	11	12	기계별 가용시간
1		4		21				8					40
2	26		5			10							40
3			20		10					22		8	40
4		35					2	6					50
5	5					6			25				50
6		16		10			3				18		60
7					1					7		7	20

[그림 11] 예제 3의 입력 데이터

4. MPGF문제에 대한 GA의 적용

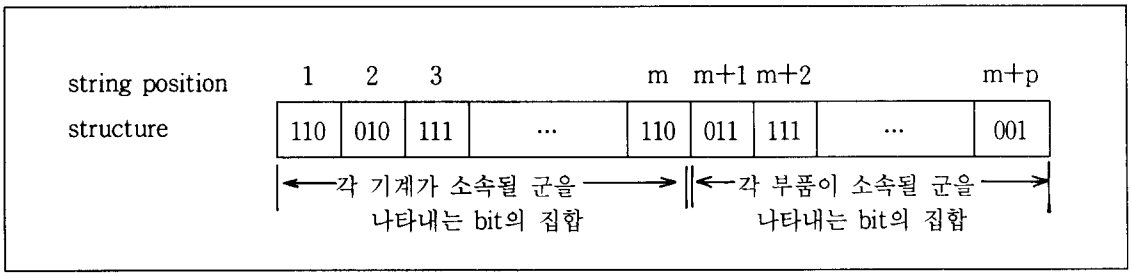
이 장에서는 3장에서 소개한 각 예제에 대하여 GA의 실행방법, 파라미터 값의 선정내용, 그리고 실행결과를 구체적으로 설명한다.

4.1 GA의 실행방법

각 예제별로 GA를 적용하기 위하여 다음과 같은 사항들을 사전에 해결해야 한다.

(1) 결정변수의 bit string화 : 각 예제별 결정변수로는 공히 각 기계 및 부품들이 할당될 군의 번호를 잡는다. 따라서 하나의 structure는 [그림 12]와 같이 이 결정변수들이 취할 수 있는 값들을 부호화시켜 일렬로 정렬시킨 것이다. 각 예제별로 한 feature 당 구성 bit수는 〈표 5〉와 같다. 예제 3 의 경우 군의 수가 사전에 결정되어 있지 않고, 3부터 6까지의 값 중 하나를 취할 수 있기 때문에 모든 경우에 대비하여 feature당 3 bit를 확보한다.

(3) 효과적인 유전 연산자의 개발 : 문제의 특성에 따라서는 PMX, OX, CX 등[14] 다양한 유전 연산자가 필요한 경우도 있으나, MPGF문제의 특성 및 앞서 채택한 string의 부호화 방식으로 인하여 2.1.3절에서 소개한 crossover 및 mutation의 두 연산자만을 사용하기로 한다. crossover 연산을 수행할 때 crossover point는 한 structure를 구성하는 bit string 내에서 무작위로 선택된다. 한 structure내에서 crossover point의 갯수는 crossover rate가 취하는 값에 따



[그림 12] 결정변수의 bit string화

〈표 5〉 예제별 feature의 구성 bit수

	문 제			feature 당 구성 bit수
	기계댓수	부품갯수	군의 수	
예제 1-1	23	20	2	1
예제 1-2	9	15	3	2
예제 1-3	5	6	2	1
예제 2-1-1	14	24	2	1
예제 2-1-2	14	24	3	2
예제 2-1-3	14	24	4	2
예제 2-2	30	41	2	1
예제 3	7	12	$3 \le k \le 6$	3

라 평균 1보다 클 수도 있고 (crossover rate> 1), 1 보다 작을 수도 있다 (crossover rate<1). 본 실험에서는 [14]에서 제안된 바에 따라 주로 crossover rate를 0.6으로 설정하였다.

(3) 제약조건식의 처리 : 본 예제들은 모두 제약조건식이 있는 최적화 문제이므로 2.2.1절에서 소개한 Penalty Method를 사용함으로써, 이 문제들을 제약조건식이 없는 최적화 문제로 전환시킨다. 그러기 위하여 $\Phi[g_i(X)]=\max(0,-g_i(X))$ 을 사용한다. penalty 상수 r에는 목적함수가 취할 수 있는 값보다 충분히 큰 수를 부과하여야 하므로 본 연구에서는 r = 1000 을 사용한다.

(4) illegal string의 처리: 예제 1 및 2에서는 군의 수가 사전에 지정되어 있다. 세대변천에 따라 crossover 연산을 수행한 후, 각 기계나 주문이 하나도 할당되지 않는 군이 생기는 경우 그 structure는 illegal string을 포함하게 된다. 그러나, structure의 전체 길이가 길어짐에 따라 그 확률은 0으로 수렴하게 된다. 따라서 이 모형들에 대해서는 illegal string에 대한 별도의 조치를 취하지 않기로 한다. 반면, 예제 3에서 군의 수는 각 structure마다 그 값을 달리 취하는 결정변수의 성격을 지니고 있을 뿐만 아니라 structure의 길이가 짧기 때문에 illegal string의 문제가 심각해진다. 각 기계 및 주문은 최소 세개 이상 여섯개 이하의 군으로 나뉘게 되어 있다. 따라서, structure로 부호화할 때 자기가 속할 군의 번호로서 1과 6사이의 숫자를 부여받는다. 그러나, 이러한 부호화 방식은 crossover 연산에 의해 illegal string을 생성시킬 수 있다. 예를 들어, 기계 및 주문을 똑같이 3개의 군으로 나누는 두 structure내에서 실제로 사용된 군의 번호가 한쪽은 1, 2, 3인 반면 다른쪽은 4, 5, 6이었다면, 이들이 단순히 crossover되어 만들어지는 structure에는 1 부터 6까지의 군의 번호가 모두 등장할 가능성이 높다. 사실 군의 번호가 무엇인지는 의미가 없고 군의 갯수가 몇 개인지가 중요하다. 따라서, 이 문제는 군 번호 4, 5, 6을 각각 1, 2, 3으로 변경 시킨 뒤 crossover함으로써 해결될 수 있다. 매 세대마다 각 structure를 검사하여 군의 갯수를 파 악한 다음 군의 번호를 1 부터 파악된 군의 갯수사이의 하나의 정수값으로 강제 변환시키는 forcing technique을 사용함으로써 crossover에 의한 illegal string 생성을 방지할 수 있다.

(5) 평가 루틴의 coding: 이미 개발된 GA package의 사용을 전제로 하여, 예제별로 위의 (1)-(4)항의 내용을 감안하여 집단내 각 structure들의 적합도를 구하는 평가(evaluation) 루틴만을 작성한다.

4.2 GA의 통제 파라미터의 선정

각 예제에 GA를 적용하기 위하여 GA package GENESIS[18]를 이용하기로 한다. GENESIS는 UNIX용 소프트웨어로 본 연구에서는 PC용 UNIX System V R.4 운영체제하에서 실행시킨다. 본 연구에서는 De Jong의 연구결과 [14]를 기초로 하여 2.3절에서 설명한 GENESIS의 통제 파라미터들의 값을 설정한다. 또한, 'elitist' 선택전략을 각 예제에서 공히 사용한다. 본 연구에서 설정한 예제별 입력 파라미터의 값은 <표 6>과 같다.

〈표 6〉 예제별 파라미터 값의 설정

파 라 미 터	예제 1-1	예제 1-2	예제 1-3	예제 2-1-1	예제 2-1-2	예제 2-1-3	예제 2-2	예제 3
Experiment No.	5	5	5	5	5	5	5	5
Total Trial	100000	100000	10000	100000	500000	200000	200000	100000
Population Size	400	400	100	400	400	500	500	200
Structure Length	43	48	11	38	76	71	71	57
Crossover Rate	0.6	0.6	0.6	0.6	0.6	3.0	3.0	0.6
Mutation Rate	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001
Genertion GAp	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
Max. Bias	0.95	0.95	0.90	0.90	0.99	0.95	0.95	0.90
Max. Convergence	43	48	11	38	76	71	71	57
Conver. Threshold	0.95	0.95	0.90	0.90	0.95	0.95	0.095	0.90
Sigma Scaling	2.0	2.0	2.0	2.0	1.0	1.0	1.0	2.0

4.3 GA의 실행결과

예제별 다섯 번의 독립 실행 결과 및 기존의 휴리스틱해는 〈표 7〉과 같다.

(1) 예제 1-1의 실행결과 : 다섯 번의 모든 실행에서 집단내 feature의 bias가 사전에 설정한 0.95를 초과함으로써 실행이 종료되었다. 모든 실행에서 Kumar 등[18]의 휴리스틱 해와 동일한 해를 구하였다.

(2) 예제 1-2의 실행결과 : 4번 실행을 제외한 나머지 네 번의 실행에서는 지정한 Maximum Bias 0.95에 도달하지 못함으로써 termination condition으로 지정한 100000개의 structure를 생성하였다. 모든 실행에서 Kumar 등[18]의 휴리스틱 해와 동일한 해를 구하였다.

(3) 예제 1-3의 실행결과 : 모든 실행에서 사전에 설정한 bias를 초과하여 실행이 종료되었다. 모든 실행에서 정성진 등[2]의 해와 동일한 해를 구하였다.

(4) 예제 2-1-1, 2-1-2, 2-1-3의 실행결과: 동일한 King[17]의 문제에 대하여 원하는 군의 수와 한 군내에 포함가능한 기계댓수의 상한만을 변화시켜 실행한 결과이다. 예제 2-1-1의 경우 모든 실행에서 지정한 bias에 도달하였고, King[17] 과 Wei & Gaither[32]의 해와 동일한 해를 구하였다. 예제 2-1-2의 경우 모든 실행에서 King[17]이나 Wei & Gaither[32]의 해보다 더 좋은 해를 구하였다. 예제 2-1-3의 경우 어느 실행에서도 지정한 bias를 얻지 못했지만 두 번의 실행에서 King[17]이나 Wei & Gaither[32]의 해와 같은 해를 구하였다. 예제 2-1-1과는 달리 예제 2-1-2와 2-1-3의 어느 실행에서도 지정한 bias를 얻지 못했다. 그 이유는 예제 2-1-2와 예제 2-1-3의 경우 feature value가 취할 수 있는 값의 범위가 예제 2-1-1에 비해 늘어짐에 따라 하나의 structure를 구성하는 bit 수도 예제 2-1-1의 71개에서 172개로 늘어나서 이 들 bit string이 취할 수 있는 값의 총 경우수가 훨씬 늘어 났기 때문으로 추측된다.

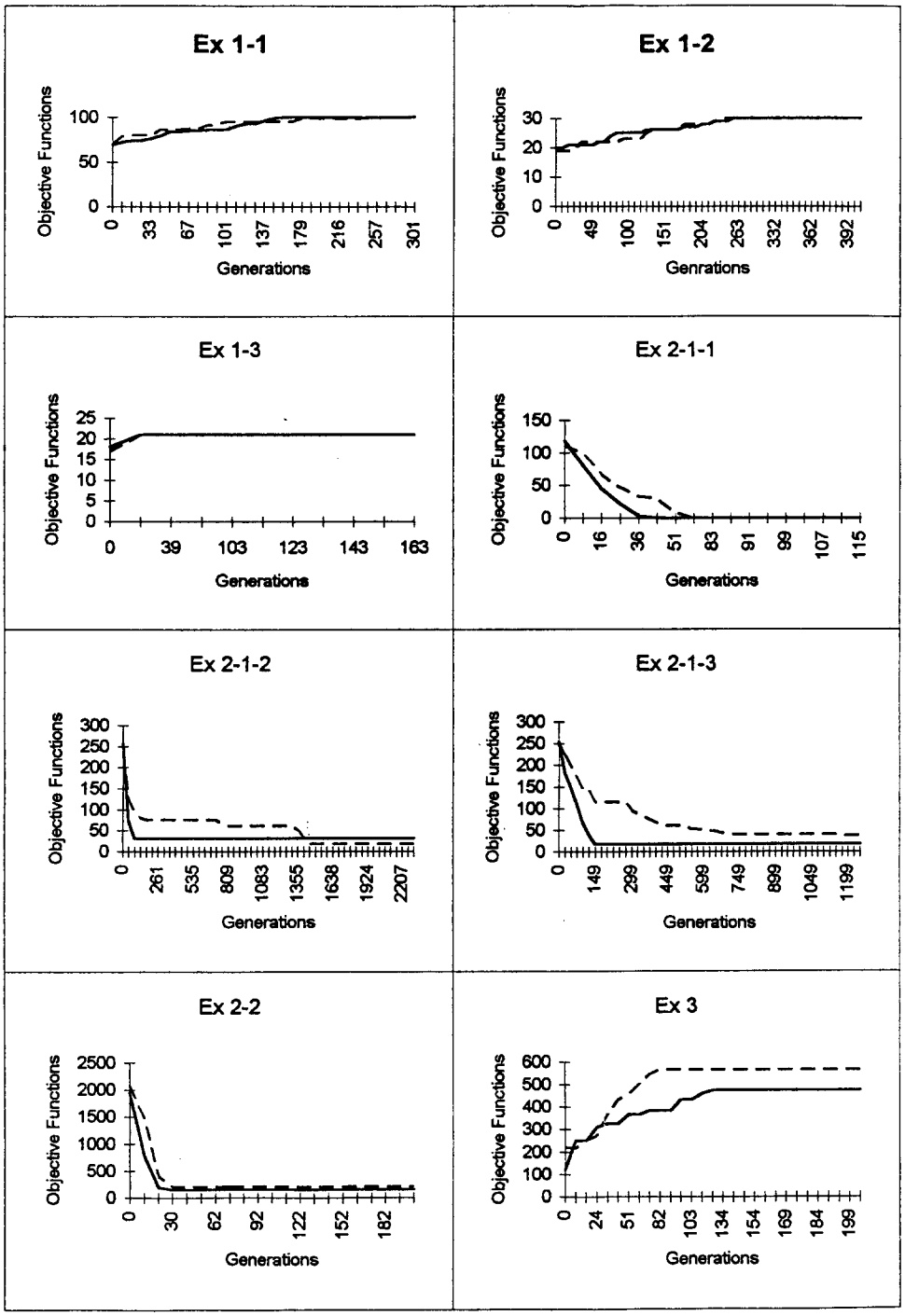
(5) 예제 2-2의 실행결과 : 모든 실행에서 지정한 bias 0.95에 도달했다. 그 중 두 번의 실행에서는 Wei & Gaither[32]의 해와 같은 해를 구하였으며 세 번의 실행에서는 이보다 열등한 해를 구하였다. 그러나, 5번의 실행에서 얻어진 해는 모두 Kumar & Vanelli[19]의 해보다는 우수하였다.

〈표 7〉 예제별 실행결과

예제	기계 갯수	부품 갯수	그룹수	GA 실행번호	Biased 여부	생 성 세대수	Trial 횟 수	목적함수값	
								GA	기존의 휴리스틱
1−1	23	20	2	1	y (0.95)	188	39996	100	100[18]
				2	y (0.95)	285	60058	100	
				3	y (0.95)	332	66018	100	
				4	y (0.95)	346	72021	100	
				5	y (0.95)	283	58060	100	
1−2	9	15	3	1	n (0.90)	480	100000	30	30[18]
				2	n (0.93)	503	100000	30	
				3	n (0.90)	475	100000	30	
				4	n (0.95)	327	70057	30	
				5	n (0.90)	493	100000	30	
1−3	5	6	2	1	y (0.90)	448	9000	21	21[2]
				2	y (0.90)	624	6003	21	
				3	y (0.90)	103	4000	21	
				4	y (0.90)	310	5002	21	
				5	y (0.90)	328	6000	21	
2-1-1	14	24	2	1	y (0.90)	62	13996	0	0[17], [32]
				2	y (0.90)	51	10066	0	
				3	y (0.90)	45	10086	0	
				4	y (0.90)	83	15995	0	
				5	y (0.90)	102	14006	0	
2-1-2	14	24	3	1	n (0.89)	2387	500000	31	52(*)-134 [32]
				2	n (0.90)	2377	500000	18	
				3	n (0.90)	2383	500000	18	
				4	n (0.89)	2345	500000	18	
				5	n (0.90)	2302	500000	18	
2-1-3	14	24	4	1	n (0.97)	1249	500000	17	17 [32]
				2	n (0.96)	1249	500000	26	
				3	n (0.96)	1249	500000	26	
				4	n (0.94)	1249	500000	37	
				5	n (0.97)	1249	500000	17	
2−2	30	41	2	1	y (0.95)	34	17234	148	146[32]
				2	y (0.95)	32	16062	146	217[19]
				3	y (0.95)	34	17245	205	
				4	y (0.95)	40	20149	205	
				5	y (0.95)	47	23340	146	
3	7	12	3≤k≤6	1	y (0.90)	205	22057	564	564[20]
				2	y (0.90)	144	16014	474	
				3	y (0.90)	93	10018	564	
				4	y (0.90)	276	30064	564	
				5	y (0.90)	177	19065	564	

예제 1−1, 1−2, 1−3, 3은 최대화 문제로, 그리고 예제 2−1−1, 2−1−2, 2−1−3, 3−2는 최소화문제로 정식화하여 실행시켰음.

(*) 52는 사용자의 사전지식을 이용하여 7종류의 기계를 특정한 cell에 사전에 배정함으로써 얻어진 값임.



〈그림 13〉 예제별 세대변천에 따른 목적함수값의 추이

(6) 예제 3의 실행결과 : 다섯 번의 실행 중 네 번의 실행에서 Kusiak[20]의 해와 동일한 해를 얻었다.

실행결과를 종합해 보면, 각 예제별로 GA기법을 5번씩 실행시킨 결과, 그 중 적어도 하나의 실행에서는 각 모형별 특성을 감안한 독특한 휴리스틱 기법으로 구한 해와 일치하거나 더 우수한 해(예제 2-1-2의 경우)를 도출하였다. 이러한 실행결과를 바탕으로 GA의 우수성을 주장하기에는 다소 미흡하지만, 다른 heuristic 접근법에서 처럼 문제마다 특유한 알고리즘을 개발해야 하는 어려움 없이, 여러 모형에 대하여 GA의 가장 기본적인 연산자만을 사용하더라도 GA의 성능이 비교적 robust하다는 사실을 확인할 수 있었다.

각 예제별로 가장 우수한 해와 가장 열등한 해가 생성된 두 실행에 대하여 세대변천에 따른 목적함수값의 추이는 [그림 13]과 같다. 각 예제를 5회 반복 실행하는 데 소요된 시간은 예제에 따라 1-35분 정도였다.

각 예제별로 문제의 특성을 감안하여 다양한 연산자를 개발하여 사용하거나, 해석적인 탐색알고리즘과 혼합 사용하거나, 또는 입력 파라미터의 값들을 잘 조율함으로써 더 우수한 해도 기대할 수 있을 것이다.[4] 또한 본 연구에서 다루지 않은 MPGF문제의 여타 모형에 대해서도 GA를 적용해 볼 경우 여전히 좋은 해를 기대할 수 있을 것이다.

5. 결 론

본 연구에서는 먼저 GA의 방법론적 특성을 소개하였다. 그리고 사례를 들어 GA의 성능을 구체적으로 평가하는 데 있어 그 적용대상이 되는 기계-부품그룹 형성 문제를 설명하고 이 문제의 세가지 모형을 선정하였다. 이 세가지 모형의 각 예제별로 GA의 구체적 실행방법 및 파라미터 값을 정하여 실행해 본 결과, 다른 heuristic 접근법에서 처럼 문제마다 특유한 알고리즘을 개발해야 하는 어려움 없이, 여러 모형에 대하여 GA의 가장 기본적인 연산자만을 사용하더라도 GA의 성능이 비교적 robust하다는 사실을 확인할 수 있었다.

향후 수리계획, 스케쥴링, 설계 최적화 및 Combinatorial 문제 등 NP-complete으로 인하여 최적해를 보장 할 수 없는 각종 대형 최적화 문제에 대하여 우수한 근사해의 도출을 위해 GA기법의 계속적인 적용이 기대된다. 그 과정에서 문제의 특성을 감안한 각종 유전 연산자의 개발, illegal string의 처리 등 GA의 효율적 적용을 위한 연구가 병행되어야 할 것이다.

참 고 문 헌

[1] 선지웅, [셀 생산시스템에서 셀 형성을 위한 혼합 유전자 알고리즘], 한국과학기술원 석사학위논문, KAIST, 1994.

[2] 정성진, 박진우, 김재윤, "FMS의 설계 및 운용을 위한 기계 부품 그룹 형성에 관한 연구", [경영과학], 제4권(1987), pp. 76-83.

[3] Blanton, J. L. and Wainwright, R. L., "Multiple Vehicle Routing with Time and Capacity Constraints Using Genetic Algorithms," in Proceedings of *ICGA Conference*, pp. 452-459, 1993.

[4] Buckles, B. P. and Petry, F. E., *Genetic Algorithms*, IEEE Computer Society Press, Los Alamitos, 1992.

[5] Chan, H. M., and Milner, D. A., "Direct Clustering Algorithm for Group Formation in Cellular Manufacture," *Journal of Ma-*

nufacturing Systems, Vol. 1(1982), pp. 65-74.

[6] Chandrasekharan, M. P. and Rajagopalan, R., "ZODIAC-An Algorithm for Concurrent Formation of Part-Families and Machine-Cells," *International Journal of Production Research*, Vol. 25(1987), pp. 835-850.

[7] Chu, C. H., "Cluster Analysis in Manufacturing Cellular Formation," *OMEGA*, Vol. 17(1989), pp. 289-295.

[8] Chu, C. H., and Hayya, J. C., "A Fuzzy Clustering Approach to Manufacturing Cell Formation," *International Journal of Production Research*, Vol. 29(1991), pp. 1475-1487.

[9] Davis, L. and Coombs, S., "Genetic Algorithms and Communication Link Speed Design: Theoretical Considerations," *in Proceedings of the Second International Conference on Genetic Algorithms*, pp. 252-256, 1987.

[10] De Witte, J., "The Use of Similarity Coefficients in Production Flow Analysis," *International Journal of Production Research*, Vol. 18(1980), pp. 503-514.

[11] Fang, H. L., Ross, p., and Corne, D., "A Promising Genetic Algorithm Approach to Job-Shop Scheduling, Re-Scheduling, and Open-Shop Scheduling Problems," *in Proceedings of ICGA Conference*, pp. 375-382, 1993.

[12] Fonseca, C. M. and Fleming P. J., "Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization," in Proceedings of *ICGA Conference*, pp. 416-423, 1993.

[13] Goldberg, D. E., *"Computer-aided Gas Pipeline Operation Using Genetic Algorithms and Rule Learning*. " Doctoral dissertation, University of Michigan, 1983.

[14] Goldberg, D. E., *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, 1989.

[15] Grefenstette, J. J., *A User's Guide to GENESIS 1. 2ucsd*, 1990.

[16] Karapathi, S., and Suresh, N. C., "Machine-Component Cell Formation in Group Technology: A Neural Network Approach," *International Journal of Production Research*, Vol. 30(1992), pp. 1353-1367.

[17] King, J. R., "Machine-Component Group Formulation in Group Technology," *OMEGA*, Vol. 8(1980), pp. 193-199.

[18] Kumar, K. R., Kusiak, A. and Vannelli, A., "Grouping of Parts and Components in Flexible Manufacturing Systems," *European Journal of Opreational Research*, Vol. 24(1986), pp. 387-397.

[19] Kumar K. R., and Vannelli, A., "Strategic Subcontracting for Efficient Dia-aggregated Mamufacturing," *International Journal of Production Research*, Vol. 25, No. 12(1987), pp. 1715-1728.

[20] Kusiak, A., "EXGT-S: A Knowledge Based System for Group Technology, *International Journal of Production Re-*

search," Vol. 26(1988), pp. 887-904.

[21] Kusiak, A., *Intelligent Manufacturing Systems*, Prentice Hall, 1990.

[22] Kusiak, A., and Cho, M., "Similarity Coefficient Algorithms for Solving the Group Technology Problems, " *International Journal of Production Research*," Vol. 30(1992), pp. 2633-2646.

[23] Kusiak A., and Chow, W. S., "Efficient Solving of the Group Technology Problem," *Journal of Manufacturing Systems*, Vol. 6(1987), pp. 117-124.

[24] Levine, D. M., "A Genetic Algorithm for the Set Partitioning Problem," in Proceedings of *ICGA Conference*, pp. 481-487, 1993.

[25] McCormick, W. T., Schweizer, P. J., and White, T. W., "Problem Decomposition and Data Reorganization by Cluster Technique," *Operations Research*, Vol. 20, No. 5(1982), pp. 993-1009.

[26] Munakata, T. and Hashier, D. J., "A Genetic Algorithm Applied to the Maximum Flow Problem," in Proceedings of *ICGA Conference*, pp. 488-493, 1993.

[27] Rajamani, D., Singh, N., and Aneja, Y. P., "A Model for Cell Formation in Manufacturing Systems with Sequence Dependence," *International Journal of Production Research*, Vol. 30(1992), pp. 1227-1235.

[28] Schaffer, J. D., "Multiple Objective Optimizations with Vector Evaluated Genetic Algorithms," *in Proceedings of an International Conference on Genetic Algorithms and Their Applications*, pp. 93-100, 1985.

[29] Singh, N., "Design of Cellular Manufacturing Systems: An invited review," *Europen Journal of Operational Research*," Vol. 69(1993), pp. 284-291.

[30] Vannelli, A., and Kumar, K. R., "A Method for Finding Minimal Bottle-neck Cellss for Groping Part-machine families," *International Journal of Production Research*, Vol.24(1986), pp. 387-400.

[31] Venugopal, V. and Narendran, T.T., "A Genetic Algorithm Approach to the Machine-Component Grouping Problem with Multipl4e Objectives," *Compkuters and Industrial Engineering*, Vol.22(1992), pp.469-480.

[32] Wei, J.C., and Gaither, N., "Cell Formation Decisions," *Decision Science*, Vol. 21(1990), pp.416-433.