# 유전알고리듬에 기반을 둔 혼합제품 유연조립라인 밸런싱*

송원섭** · 김형수** · 김여근**

# Mixed-product flexible assembly line balancing based on a genetic algorithm*

Won Seop Song** · Hyeong Su Kim** · Yeo Keun Kim**

Abstract

A flexible assembly line (FAL) is a production system that assembles various parts in unidirectional flow line with many constraints and manufacturing flexibilities. In this research we deal with a FAL balancing problem with the objective of minimizing the maximum workload allocated to the stations. However, almost all the existing researches do not appropriately consider various constraints due to the problem complexity. Therefore, this study addresses a balancing problem of FAL with many constraints and manufacturing flexibilities, unlike the previous researches.

We use a genetic algorithm (GA) to solve this problem. To apply GA to FAL, we suggest a genetic representation suitable for FAL balancing and devise evaluation method for individual's fitness and genetic operators specific to the problem, including efficient repair method for preserving solution feasibility. After we obtain a solution using the proposed GA, we use a heuristic method for reassigning some tasks of each product to one or more stations. This method can improve workload smoothness and raise work efficiency of each station. The proposed algorithm is compared and analyzed in terms of solution quality through computational experiments.

Keyword : Flexible Assembly Line, Line Balancing, Flexibility, Genetic Algorithm

## 1. 서 론

유연조립시스템(flexible assembly system : FAS)은 조립기능을 수행하는 로봇, 조립품의 운반기능과 저장기능을 수행하는 자동자재 처리설비, 그리고 이들 설비를 제어하는 컨트롤 시스템으로 구성

논문접수일 : 2004년 8월 24일 논문게재확정일 : 2004년 12월 1일
* 이 논문은 2003년도 한국학술진흥재단 지원에 의하여 연구되었음(KRF-2003-002-D00362).
** 전남대학교 산업공학과

되어 여러 종류의 제품을 자동으로 조립할 수 있는 시스템을 말한다[9, 11]. 유연조립라인(flexible assembly line : FAL)은 FAS의 한 형태로 제품 조립이 단 방향의 흐름만을 허용하는 작업장에 의하여 이루어지는 생산시스템을 말한다[10, 13]. 본 연구에서는 FAL에서 밸런싱(balancing)문제를 다룬다. 밸런싱문제는 조립라인에 부과된 여러 제약들을 어기지 않고, 고려되는 목적이 최적이 되도록 라인 내의 작업장에 작업을 할당하는 문제이다. 이 문제는 생산성과 시스템 효율성의 향상을 위해 중요하게 다루어져야 한다[12].

FAL에서 작업할당 문제는 NP-hard문제이고, 흔히 많은 제약들을 포함한다[7, 14]. 이러한 이유로, 혼합제품을 생산하는 유연조립라인 밸런싱(flexible assembly line balancing : FALB)에 관한 대부분의 기존 연구에서는 각 제품별로 몇 개의 조립 순서가 주어지고 그 중 좋은 것을 선택하거나[12], 조립 순서를 고려하지 않고, 작업장 공간만을 고려하여 작업할당 문제를 다루었다[7]. 또한, Graves and Redfield[5]와 Ammons *et al.*,[3]은 FAL의 작업장이 갖는 공간 크기를 제약으로 고려하지 않고 있다.

본 연구에서는 FAL의 각 작업장에서 나타날 수 있는 현실적인 제약과 각 제품이 가질 수 있는 여러 조립순서를 고려한다. FAL의 각 작업장에서는 로봇에 의해 조립이 행하여지므로 조립 가능한 작업과 그렇지 못한 작업이 있고, 각 작업에 필요한 설비 배치로 인해 요구되는 작업장 공간 크기 때문에 할당 가능한 작업수가 제한될 수 있다. 이러한 작업장 특성으로 인하여 특정 작업이 가능한 작업장이 제한될 뿐더러, 작업의 선행관계에 의해 더 축소될 수 있다. 또한 제품의 조립 순서유연성을 고려함으로써 유연조립라인의 설비의 효율화를 얻고자 한다.

본 연구에서는 FAL이 갖는 작업장 제약과 조립 순서 유연성을 고려한 효율적인 작업할당 기법을 제시하는데 있다. 이를 위한 방법론으로는 유전알고리듬(genetic algorithm)을 사용한다. 유전알고리듬은 메타휴리스틱(metaheuristic)으로, 자연의 진화 과정을 모방한 일종의 탐색기법이다[6]. 유전알고리듬을 FALB 문제에 채용하기 위하여, 본 연구에서는 적절한 개체표현과 적응도 평가함수를 제안하고, 유전 연산자를 개발한다. 그리고 제안한 알고리듬의 성능 분석을 위하여, 새로운 발견적기법을 제안하고 이들 기법의 성능을 비교한다.

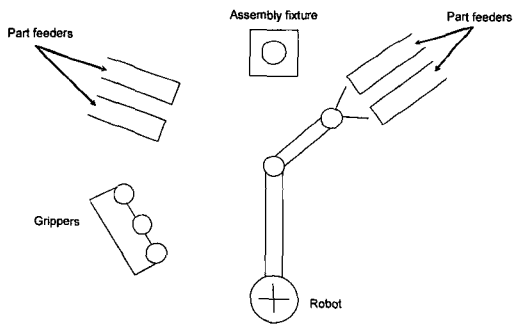
본 연구의 구성은 다음과 같다. 먼저, 제2장에서는 FAL에 관한 설명과 본 연구에 대하여 구체적으로 살펴본다. 제3장에서는 FALB를 해결하기 위해 개발된 유전알고리듬의 구성 요소를 설명하고, 제4장에서는 발견적 기법을 이용하여 한 제품의 작업을 여러 작업장에 할당하는 경우로 확장하는 과정을 설명한다. 제5장에서는 실험설계 및 결과를 제시하고 분석한다. 마지막으로 제6장은 결론으로 구성되어 있다.

## 2. 문제 정의

문제 정의에 앞서, 혼합제품을 생산하는 FAL과 전통적인 직선조립라인(straight assembly line : SAL)의 차이를 비교하여 보자. 첫째, 제품조립이 FAL에서는 자동조립기계(예로, 산업용 로봇)와 자동자재처리설비(부품공급장치, 자재처리 로봇 등)에 의해 이루어지고, SAL에서는 작업자와 수동(또는 반자동) 조립설비에 의해 이루어 진다. 이로 인해, 둘째, FAL에서는 흔히 각 작업의 할당 가능 작업장이 몇 개로 제한되어 있는 반면, SAL에서는 모든 작업이 각 작업장에 할당될 수 있다고 본다. 단지 예외적으로, 설비제약이 있는 작업은 특정 작업장에 할당되어야 한다는 제약을 갖는다. 셋째, FAL의 작업장에서는 작업을 위한 피딩 메커니즘(feeding mechanism)의 필요에 의해 이를 위한 설치공간을 필요로 한다. 이로 인해 작업장에 할당될 수 있는 작업의 수는 유한하다. 이것을 흔히 FAL의 유연성 용량(flexibility capacity)또는 작업장 용량(staging capacity)이라 부른다[9, 10]. 본 연구에서는 이로 인한 작업할당 제약을 작업장의 공간제약이라 부르기로 한다. FAL의 작업장 구조를 간단히 나타내면 [그림 1]과 같다.

본 연구에서는 아래와 같은 가정을 둔다.

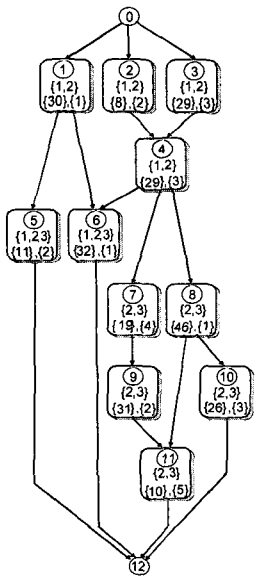
1) FAL은 고정된 수의 작업장을 가지며, 각 작업장에는 단지 하나의 조립기계가 있다.
2) 부품 조립을 위해 지나온 작업장 재방문(re-visiting)은 불가하나 건너 뛰기(bypass)의 전진은 가능하다.
3) 각 작업은 여러 작업장에서 작업될 수 있다.
4) 모든 제품의 작업선행관계를 표현한 결합선행공정도를 사용한다. 이는 [그림 2]와 같이 나타내어지고, 작업들의 선행관계, 작업 가능한 작업장집합, 작업소요시간, 작업에 필요한 공간크기의 정보를 담고 있다.
5) 각 제품에 대한 수요는 주어졌다.
6) 각 작업의 조립시간은 준비시간, 공구교환시간, 장착/탈착 시간을 포함한다.



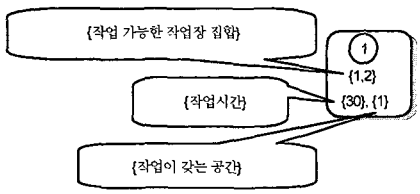

[그림 1] 작업장 구성요소

본 연구에서는 위에서 언급한 가정아래서, 작업선행제약과 작업장 공간제약 등의 작업할당 제약을 만족하면서, 작업장부하가 가장 큰 작업부하를 최소로 하는 작업할당을 하고자 한다.

본 연구에서 다루는 FALB의 작업할당은 다음과 같은 특징을 갖는다. 첫째, 같은 작업이라도 제품에 따라 할당 작업장이 다를 수 있다. 전통적인 혼합제품 SAL 밸런싱 [2, 8, 15]에서는 부품들의 배치에 따른 혼잡과 작업자 기능의 한계로 인하여 모든 제품에서 같은 작업은 하나의 작업장에 할당되는 것으로 보았다. 더 나아가, 둘째, 생산계획기간 동안 조립 제품의 수요량(계획생산량)이 복수 개일 때, 동일 모델 제품의 작업이라도 여러 작업장에 할당할 수 있다. 예로, 일부 제품은 작업장 A 에서 나머지 제품은 작업장 B 에 작업을 나누어 할당할 수 있다. 이는 작업설비나 작업 공간은 더 요구되나 작업부하를 더 균형있게 하여 생산율을 높일 수 있다. 반면 기존 연구[4, 15]의 SAL 밸런싱에서는 작업장 유연성의 한계로 인한 제약에 의해 이를 허용하지 않고 있다. 본 연구에서 첫째의 작업 할당은 유전알고리듬에 의하여 구한다. 이를 위한 유전알고리듬은 3장에서 다룬다. 4장에서는 두번째 언급한 작업할당을 위하여, 본 연구에서 새로운 발견적 기법을 제안한다. 이 기법은 3장에서 유전알고리듬에 의해 구한 해를 초기해로 사용한다.




(a) 결합 선행공정도




(b) 한 작업의 세부정보

[그림 2] FALB를 위한 네트워크 표현

본 연구에서 사용되는 기호를 먼저 정의하여 두자.

$I$ : 작업의 집합, $i \in I = \{1, 2, \cdots, m\}$.

$J$ : 작업장의 집합, $j \in J = \{1, 2, \cdots, n\}$.

$K$ : 제품의 집합, $k \in K = \{1, 2, \cdots, p\}$.

$d_k$ : 제품 $k$의 수요.

$B_j$ : 작업장 $j$의 공간 크기 .

$a_i$ : 작업 $i$를 수행하는 요구되는 작업장 공간 크기.

$b_j$ : 작업장$j$에 이미 할당된 작업들의 소요작업공간(만약, 모든 $a_i = 1$이라면 작업장에 할당된 작업의수).

$t_{ki}$ : 제품 $k$의 작업 $i$에 대한 조립시간.

$\hat{t}_{ki} = d_k t_{ki}$, $i = 1, 2, \cdots, m$, $k = 1, 2, \cdots p$, (제품 $k$의 수요를 고려한 작업 $i$의 조립시간).

$T$ : 총 작업 시간, $\sum_k \sum_i \hat{t}_{ki}$

$T_j$ : 작업장 $j$에 할당된 총 작업시간.

$\overline{T}$ : 작업장에 할당된 작업량의 평균(평균 작업량), 즉, $\overline{T} = T/n$.

$J(i)$ : 작업 $i$가 할당 가능한 작업장 집합.

$I(j)$ : 작업장 $j$에 할당된 작업의 집합

$IP(i)$ : 작업 $i$의 직선행작업 집합.

$S(i)$ : 작업 $i$의 후행작업집합.

# 3. 유연조립라인 밸런싱을 위한 유전알고리듬

이 장에서는 2장에서 언급한 유연조립라인 밸런싱을 위한 유전알고리듬을 제안한다. 유전알고리듬에서는 문제의 가능해를 개체로 표현하고, 이들 개체로 구성된 모집단을 운영한다. 본 연구에서는 엘리티즘(elitism)을 사용한 전통적인 유전알고리듬을 사용한다. 엘리티즘은 지금까지 얻은 가장 좋은 해를 보관하는 전략이다. 알고리듬의 기본 절차는 아래와 같다.

단계 1(초기모집단 생성) : 초기 모집단을 구성한다.

단계 2(개체 평가) : 모든 개체들의 적응도를 평가한다. 가장 높은 적응도를 $f_{best}$로 둔다.

단계 3(종료조건) : 종료조건이 만족되면 끝낸다.

단계 4(선택) : 새로운 세대의 모집단을 토너먼트선택으로 구성한다.

단계 5(재생산) : 교차와 돌연변이 유전연산자를 적용하여 자손을 생산한다.

단계 6(평가와 최선해 갱신) : 모집단에 있는 모든 개체의 적응도를 평가한다. 현재의 모집단에서 가장 높은 적응도를 $f_{best}$와 비교하여, 더 높은 값을 갖는 개체를 최선해로 둔다. 그리고 단계 3으로 간다.

유전알고리듬의 탐색 성능은 진화 전략등과 함께 진화의 구성요소들 즉, 개체의 표현, 유전연산, 개체의 평가, 선택(selection)과 유전파라미터(genetic parameters)에 의해 영향을 받는다. 기본적인 파라미터에는 모집단의 크기, 교차율, 돌연변이율 등이 있다.

## 3.1 표현 및 초기 모집단

### 3.1.1 표현

유전알고리듬에서 개체는 주어진 문제의 잠재해를 가능한 자연스럽고 단순하게 표현되는 것이 바람직하다. 또한, 하나의 잠재해가 중복 표현되지 않아야 한다. 개체는 행이 제품을, 열이 작업을 나타내는 (p×m) 행렬로 표현한다. [그림 3]은 이를 보여주는 데, 인자 값, $c_{ki}$는 제품 $k$의 작업 $i$가 행하여지는 작업장번호(단, 제품 $k$에서 작업 $i$가 수행되지 않으면 $c_{ki} = 0$)를 나타낸다.

| $k$ \ $i$ | 1 | 2 | 3 | ⋯ | $M$ |
|---|---|---|---|---|---|
| 1 | $c_{11}$ | $c_{12}$ | $c_{13}$ | ⋯ | $c_{1m}$ |
| 2 | $c_{21}$ | $c_{22}$ | $c_{23}$ | ⋯ | $c_{2m}$ |
| ⋮ | ⋮ | ⋮ | ⋮ | | ⋮ |
| $P$ | $c_{p1}$ | $c_{p2}$ | $c_{p3}$ | ⋯ | $c_{pm}$ |

[그림 3] 개체표현

이러한 표현은 일종의 그룹번호 표현으로 작업장 정보(작업장에 할당된 작업)가 자연스럽고 명확히 표현되어 개체의 해석과 정보의 이용이 용이하다는 장점을 갖는다.

### 3.1.2 초기모집단

초기모집단은 해공간의 효율적 탐색을 위하여 다양한 개체로 구성되도록 하여야 한다. 그리고 본 연구에서 초기 개체를 생산할 때 작업의 할당가능 작업장제약, 선행제약, 작업장 공간제약, 그리고 작업장 부하를 고려한다. 개체의 생성 절차는 아래와 같다.

단계 1 : 각 제품 $k$에서 작업 $i$가 수행되지 않으면 $c_{ki}=0$으로 둔다. 그리고 $b_j=0$, $T_j=0$, $j=1, 2, \cdots, n$ 과 $V=I$로 두고 $\overline{T}$를 계산한다.

단계 2 : $V$에서 선행 작업이 모두 할당된 작업 중에서 임의로 작업 $r$을 선택한다.

단계 3 : 작업 $r$의 직 선행작업이 할당된 작업장 중에서 가장 늦은 작업장 번호를 $s$로 둔다. 직 선행작업이 없으면 $s=0$로 둔다.

단계 4 : $J(r)$에서 $s$보다 작은 작업장을 제거하고 작업장번호를 오름차순으로 정렬한 작업장 집합을 $RJ(r)$로 둔다.

단계 5 : $u=|RJ(r)|$, $q=1$로 둔다. 그리고 $j_q$는 집합 $RJ(r)$의 $q$번째 원소 즉, $q$번째 작업장을 가리킨다.

단계 5.1 : $b_{j_q}+a_r \le B_{j_q}$이고 $T_{j_q}+\sum_k \hat{t}_{kr} \le \overline{T}$ 이면, $j^*=j_q$로 두고 단계 5.3으로 간다.

단계 5.2 : $q<u$이면, $q=q+1$로 두고 단계 5.1로 간다. 그렇지 않으면 $j^*=j_1$로 둔다.

단계 5.3 : 모든 제품의 작업 $r$을 작업장 $j^*$에 할당한다. 즉, 개체표현에서 $r$번째 열의 원소 중 0이 아닌 모든 원소를 $j^*$로 둔다.

단계 6 : $b_{j^*} \leftarrow b_{j^*}+a_r$, $V \leftarrow V \setminus \{r\}$, $T_{j^*} \leftarrow T_{j^*}+\sum_k \hat{t}_{kr}$ 로 갱신한다. $V=\emptyset$이면 종료하고, 그렇지 않으면 단계 2로 간다.

위 절차를 모집단의 크기만큼 반복하여 초기 모집단을 형성한다. 단계 2, 3, 4에서 작업의 선행제약과 할당가능 작업장 제약이 항상 만족되도록 한다. 그리고 단계 2에서 선행제약을 만족하는 작업들 중에서 임의로 선택함으로써 다양한 해를 생산하도록 유도한다. 단계5.1에서는 작업장 공간제약과 작업부하를 고려하고 있다. 단계5.2는 이들 두 제약을 만족하는 작업장이 없는 경우, 작업 $r$은 조립가능 작업장중에서 가장 앞에 위치한 작업장에 할당한다.

앞에서 언급했듯이, 본 연구에서 다루는 FAL에서 각 작업은 제품에 따라 다른 작업장에 할당될 수 있다. 예로, 제품1의 작업6은 작업장1에, 제품2의 작업6은 작업장 2에 할당될 수 있다. 그러나 위의 초기 개체의 생성에 있어서는 동일 작업은 제품에 상관 없이 동일 작업장에 할당되는 것으로 두고 해를 생성하였다. 이는 초기모집단에 문제가 갖는 모든 제약을 만족하는 개체를 가능한 많이 생산하기 위해서이다. 그리고 이들 해를 기반으로 한 유전 연산에 의해 더 다양하고 좋은 해로 진화하도록 유도하기 위함이다.

| $k$ \ $i$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 2 | 0 | 1 | 2 | 2 | 0 | 3 | 3 |
| 2 | 1 | 1 | 0 | 2 | 3 | 1 | 2 | 2 | 3 | 0 | 3 |
| 3 | 1 | 0 | 1 | 2 | 3 | 1 | 2 | 2 | 3 | 3 | 0 |

[그림 4] 초기모집단

[그림 4]는 [그림 2]의 선행 관계를 갖고 제품의 모델 수가 3 개인 경우의 개체 표현의 예를 보여주고 있다.

## 3.2 평가 함수 및 선택

개체의 적응도(fitness)는 개체의 생존능력으로 정의된다. 평가 함수는 개체의 적응도를 평가하는 함수로서 일반적으로 최적화하려는 문제의 목적함수를 사용한다.

본 연구에서 다루는 FALB는 작업의 선행제약, 작업의 할당가능작업장 제약, 작업장의 공간제약

등이 있다. 유전알고리듬에서 제약을 다루는 방법은 크게 모집단의 모든 개체가 항상 가능해가 되도록 재생산하는 방법과 모집단에 비가능해를 허용하면서 이들 해에 벌금을 부과하는 방법으로 나눌 수 있다[1]. 일반적으로 복잡하고 여러 제약을 갖는 문제를 다루는 유전알고리듬에서 모집단의 모든 개체가 가능해가 되도록 유지 또는 생산하기가 쉽지 않다. 모든 제약을 고려한 표현과 해석 그리고 유전연산이 쉽지 않기 때문이다. 본 연구에서 사용한 평가함수는 식 (1)과 같다. 첫째 항은 최대부하량을 갖는 작업장의 작업부하이고, 둘째 항은 벌금을 나타낸다. 적응도는 평가함수 $Eval$의 값이 낮을수록 높아진다.

$$Eval = \max_j T_j + (c \times ws^{\alpha}) \qquad (1)$$

여기서, $ws = \sum_{j=1}^{n} \max(0, b_j - B_j)$이고, $c$와 $\alpha$는 파라미터이다.

유전알고리듬에서는 자연선택(natural selection)을 모방하여 적응력이 높은 개체들에게 다음 세대의 개체 생산에 참여할 확률을 높게 준다. 본 연구에서는 토너먼트(tournament) 선택 방법을 사용한다[1]. 토너먼트 선택방법은 모집단의 개체를 임의로 나열하고, 앞에서부터 토너먼트 크기 $l$개씩 차례로 그룹으로 나누어, 각 그룹에서 적응도 값이 가장 좋은 개체를 선택한다. 그리고 모집단의 개체를 다시 임의로 나열하여, 다시 반복한다. 이 과정을 모집단의 크기의 개체 수가 얻어 질 때까지 반복한다. 토너먼트 선택 방법은 최소화문제에 적용이 용이하고, 토너먼트 크기 $l$에 의해 모집단의 다양성과 선택압력을 적절히 조정할 수 있다는 장점이 있다.

### 3.3 유전연산자

유전연산자에는 교차(crossover)와 돌연변이(mutation)가 있다. 교차는 두 부모의 염색체를 교차하여 자손을 생산하는 유전연산자이고, 돌연변이는 부모로부터 유전된 특정 인자를 임의로 변화시키는 유전연산자이다. 교차는 부모가 갖는 형질을 자손에 상속시킴으로써, 결과적으로 좋은 해의 이용(exploitation)이라는 역할을 하게 되고, 돌연변이는 인자에 변화를 주어, 다양한 해공간을 탐색(exploration)하는 역할을 한다

본 연구에서는 변형된 구조교차(structural crossover)를 사용한다. 사용된 구조교차는 다음과 같다. 범위 [1, $n$]에 있는 정수 중에서 임의로 정수 $r$을 선택한다. 한 부모(P1)로부터 인자 값이 1부터 $r$인 인자를 자손(O1)에게 상속한다. 다른 부모(P2)에서 1부터 $r$까지의 값을 갖는 인자들을 모두 지운다. 자손(O1)에서 비어있는 인자들을 부모(P2)에서 복사한다. 자손(O1)에서 인자 값이 결정되지 않은 인자는 일종의 발견적 기법인 재할당 기법에 의해 보수한다. 다른 자손(O2)은 두 부모의 역할을 바꾸어 생산한다.

[그림 5]는 $r$ = 2일 때, 교차의 예를 보이고 있다. 자손 (O1)에서 밑줄이 있는 인자 값은 부모1(P1)로부터 윗줄이 있는 것은 부모2(P2)로부터 상속 받은 것이다. 그리고 는 부모로부터 인자값을 상속받지 못하고 비어 있는 인자를 의미한다. 이 인자 * 들은 다음에 제시하는 재할당절차에 의해서 인자 값을 부여 받는다.

| | $k$ \ $i$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| P1 | 1 | 1 | 1 | 1 | 2 | 0 | 1 | 3 | 3 | 0 | 3 | 3 |
| | 2 | 1 | 1 | 0 | 2 | 3 | 2 | 2 | 2 | 2 | 0 | 3 |
| | 3 | 1 | 0 | 1 | 2 | 3 | 1 | 2 | 2 | 3 | 3 | 0 |
| | | ↓ | ↓ | ↓ | ↓ | ↓ | ↓ | ↓ | ↓ | ↓ | | |
| O1 | 1 | 1 | 1 | 1 | 2 | 0 | 1 | * | * | 0 | 3 | 3 |
| | 2 | 1 | 1 | 0 | 2 | * | 2 | 2 | 2 | 2 | 0 | 3 |
| | 3 | 1 | 0 | 1 | 2 | * | 1 | 2 | 2 | 3 | 3 | 0 |
| | | | | | | | | | | ↑ | ↑ | ↑ |
| P2 | 1 | 1 | 1 | 2 | 1 | 0 | 1 | 2 | 2 | 0 | 3 | 3 |
| | 2 | 1 | 1 | 0 | 2 | 2 | 1 | 3 | 2 | 3 | 0 | 3 |
| | 3 | 1 | 0 | 2 | 1 | 2 | 1 | 3 | 2 | 3 | 3 | 0 |

[그림 5] 교차

돌연변이는 개체 돌연변이율에 의해 개체들을 선택하고, 선택된 개체에 대하여 인자 돌연변이율에 의해 인자를 선택한다. 이들 인자들은 재할당 절차에 의해 작업장번호(인자 값)가 부여된다.

본 연구에서 제안하는 재할당 절차에서는 각 작업장에 이미 할당된 작업의 부하와 소요공간 크기의 정보를 이용한다. 이 절차에서는 재할당 작업이 선후행 관계를 만족하는 할당 가능한 작업장들 중에서, 공간제약을 만족하면 작업 부하량이 가장 작은 작업장에 할당하고, 그렇지 않으면 초과로 요구되는 공간크기가 가장 작은 작업장에 할당되게 한다. 이러한 재할당 과정과 이에 필요한 기호는 다음과 같다.

$R(k)$ : 제품 $k$, $k=1, 2, \cdots, p$에서 작업장이 미할당된 작업의 집합.

$AR(k)$ : $IP(i)$, $i \in R(k)$의 원소가 모두 개체에 할당된 작업집합. 즉, $AR(k) = \{i \mid \{R(k) \cap IP(i)\} = \emptyset, i \in R(k)\}$.

$E_i$ : 개체에서 $IP(i)$의 작업들에 해당하는 인자 값 중 가장 큰 값. 만약 $IP(i) = \emptyset$이면, $E_i = 1$.

$L_i$ : 개체에서 $S(i)$의 작업들에 해당하는 인자 값 중 0이 아닌 가장 작은 값. 만약 $S(i)-R = \emptyset$이면, $L(i) = n$.

$AS(i)=\{j \mid E_i \le j \le L_i, j \in J(i)\}$, $J(i)$에서 $E_i$보다 이른 작업장과 $L_i$보다 늦은 작업장을 제거한 작업장 집합.

$$\overline{a_{ij}} = \begin{cases} a_i, & if \quad i \notin I(j) \\ 0, & \text{그밖에} \end{cases}$$

단계 1 : $\overline{T}$와 $T_j$ 그리고 $b_j$, $j=1, 2, \cdots, n$를 계산하고, $k = 1$로 둔다.

단계 2 : $R(k)$와 $AR(k)$를 구한다. $R(k) = \emptyset$이면 단계 8로 간다.

단계 3 : $AR(k)$에서 작업소요공간이 가장 큰 작업을 $i^*$로 선택한다. 여러 작업이 존재하면 그 중에서 작업시간이 가장 큰 작업을 선택한다.

단계 4 : $E_{i^*}$와 $L_{i^*}$를 구하고, 이로부터 $AS(i^*)$를 구한다.

단계 5 : $RS = \{j \mid b_j + \overline{a}_{i^*j} \le B_j, j \in AS(i^*)\}$로 둔다.

단계 5.1 : $RS \neq \emptyset$이면, $j^* = \arg\min_{j \in AS(i^*)} (T_j + t_{ki^*})$로 두고, 단계 6으로 간다.

단계 5.2 : 그렇지 않으면, $j^* = \arg\min_{j \in AS(i^*)} (b_j + \overline{a}_{i^*j})$로 둔다.

단계 6 : 제품 $k$의 작업 $i^*$를 작업장 $j^*$에 할당하고, 집합 $R(k)$에서 $i^*$를 제거한다.

단계 7 : $R(k) \neq \emptyset$이면, $AR(k)$, $T_{j^*}$와 $b_{j^*}$를 갱신한 후 단계 3으로 간다.

단계 8 : $k = k + 1$로 둔다.

단계 9 : $k > p$이면 종료하고, 그렇지 않으면 $T_{j^*}$, $b_{j^*}$, $j = 1, 2, \cdots, n$를 갱신하고 단계 2로 간다.

위 재할당 절차에서 작업 소요 공간과 그에 따른 작업장 공간 정보는 단계 3과 단계 5.2에서, 작업의 할당 가능한 작업장 정보는 단계 4에서, 작업장부하 정보는 단계 5.1에서 각각 사용되고 있다.

## 4. 할당제약 완화 : 한 제품의 동일 작업을 여러 작업장에 분산 할당

생산계획기간 동안 제품이 복수 개 생산된다. 이 때 어떤 제품의 특정 작업이 할당된 작업장의 부하가 많으면 이 제품의 일부는 이 작업을 다른 작업장에서 행할 수 있다. 그런데, 제 3장에서는 한 제품에서 같은 작업은 하나의 동일 작업장에서 조립된다고 보았다. 이 장에서는 이러한 제약을 완화하여, 즉 한 제품의 동일 작업을 복수 작업장에 할당하는 것을 허용한다.

혼합제품조립라인 밸런싱 문제에서 유전표현으로는 흔히 순열표현과 그룹번호표현을 사용해 왔다. 이러한 표현 방법은 각 제품의 한 작업을 하나의 인자로 표현하고 인자 값은 작업할당 정보를 나타낸 것이다. 마찬가지로 동일 작업을 여러 작업장에 분산 할당하는 유전 알고리듬을 위해 각 제품에서 한 작업이 2개 이상의 작업장에 중복 할당되는, 제품, 작업, 할당가능작업장의 3차원 염색체의 개체 표현을 검토하였다. 하지만 이 표현은 개체 크기가 아주 크고, 좋은 스키마의 이용과 탐색을 효율적으로 하여 좋은 해를 생산할 수 있는 유전연산자의 개발에 어려움이 있었다. 따라서 제 3장에서 제안된

유전 알고리듬에 의해 구한 해를 초기해로 사용하여 이를 개선시키는 발견적 기법을 제안한다.

제안하는 발견적 기법에서는 가장 큰 작업부하를 갖는 작업장에서 작업시간이 가장 많이 요구되는 제품의 작업을 선택하고, 이 제품의 생산량을 다른 작업장에 적절히 분할하여 최대 작업장 부하를 줄인다. 이러한 과정을 이 부하가 감소되지 않을 때까지 반복된다. 구체적인 절차는 아래와 같다.

단계 1 : $T_j$와 $b_j$, $j = 1, \cdots, n$를 계산하고, $jj = 0$로 둔다.

단계 2 : $j^* = \arg\max_{j \in J} T_j$로 두고 $I(j^*)$를 구한다. $j^* = jj$이면 끝낸다.

단계 3 : $I(j^*)$중 가장 많은 작업시간을 갖는 제품 $k$의 $i^*$를 선택한다.

단계 4 : $RAS(i^*) = \{j \mid b_j + \bar{a}_{i^*j} \le B_j, j \in AS(i^*)\}$로 둔다. $RAS(i^*) = \emptyset$이면 단계 7로 간다.

단계 5 : $RS = \{j \mid T_{j^*} - t_{ki^*} \ge T_j + t_{ki^*}, j \in RAS(i^*), j \neq j^*\}$로 둔다. 그리고 $RS = \emptyset$이면, 단계 7로 간다.

단계 6 : $w = \arg\min_{j \in RS} T_j$로 둔다.

단계 6.1 : 제품$k$의 작업 $i^*$를 작업장 $j^*$에서 작업장 $w$로 한 단위(제품)씩 옮긴다. 그리고 $T_w$와 $T_{j^*}$를 수정한다.

단계 6.2 : 작업장 부하가 $T_{j^*} < T_w$이 될 때까지 단계 6.1을 반복한다.

단계 6.3 : $T_{j^*} < T_w$이면, 작업장 $j^*$와 작업장 $w$에서 각각 행하여지는 제품$k$의 작업 $i^*$의 단위 수를 계산한다. 그리고 $b_{j^*}$와 $b_j$를 수정한다.

단계 7 : $I(j^*)$에서 $i^*$를 제거하고 $I(j^*) = \emptyset$이면, $jj = j^*$로 두고 단계 2로 가고, 그렇지 않으면 단계 3로 간다.

위 기법에서는 작업부하가 가장 큰 작업장에서 (단계2) 가장 많은 작업시간을 갖는 작업을 선택하여(단계 3), 이 작업을 행할 수 있는 작업장 중에서 공간제약을 만족하는 작업장 집합을 구한다(단계 4). 이 작업장 집합 중에서 최대 작업 부하를 감소시키는 작업장으로 선택된 작업을 이동시킨다(단계 5와 6). 이와 같은 과정을 최대 작업 부하가 감소되지 않을 때까지 반복한다.

# 5. 실험 및 분석

## 5.1 비교 알고리듬

본 연구에서는 하나의 유연조립라인 밸런싱을 위한 새로운 발견적 기법(heuristic for FALB : HFALB)을 제안하고, 이를 3장에서 제안한 유전 알고리듬(genetic algorithm)과 4장에서 제안한 발견적 방법(genetic algorithm + heuristic method)의 성능 비교 분석에 사용한다. 제 3장과 4장에서 제안한 기법은 각각 GA, GA + HM로 표기한다.

HFALB의 기본 개념은 주어진 cycle time(CT)을 넘지 않도록 문제의 제약을 만족하는 작업을 할당 규칙에 의해 할당하고, 만일 마지막 작업장에서의 작업 할당이 CT를 넘게 되면 CT를 증가시켜 모든 작업장에서 CT를 만족하는 작업할당이 될 때까지 이 과정을 반복하는 발견적 기법이다.

HFALB의 절차는 아래와 같다.

단계 1 : $CT = \overline{T}$로 둔다.

단계 2 : $T_j = 0$, $b_j = 0$, $j = 1, \cdots, n$, 그리고 $j = 1$로 두고, 작업 $i$를 행하는 모든 제품의 집합 $K(i)$와 $t_i = \sum_k t_{ki}$, $i = 1, \cdots, m$를 구한다.

단계 3 : 모든 미할당 작업들 중 선행제약과 공간제약을 만족하고, 작업장 $j$를 할당가능 작업장으로 갖는 작업집합 $F(j)$를 구한다. 만약, $F(j) = \emptyset$이면, 단계 7로 간다.

단계 4 : 집합 $F(j)$에서 할당규칙 중에서 임의로 하나를 적용하여 작업 $i^*$를 선택한다.

단계 5 : $T_j + t_{i^*} < CT$이면, 미할당된 모든 제품의 작업 $i^*$를 작업장 $j$에 할당하고, $F(j) \leftarrow F(j) - \{j\}$, $b_j \leftarrow b_j + \bar{a}_{i^*j}$, $T_j \leftarrow T_j + t_{i^*}$로 갱신한 후 단계 3으로 간다.

단계 6 : $u = |K(i^*)|$, $q = 1$로 둔다.

단계 6.1 : 집합 $K(i^*)$에서 $t_{ki^*}$가 $q$번째로 큰 제품 $k^*$를 선택한다.

단계 6.2 : $T_j + t_{k^*i^*} < CT$이면, 제품 $k^*$의 작업 $i^*$를 작업장 $j$에 할당한다. 그리고 $t_{i^*} = t_{i^*} - t_{k^*i^*}$로 두고, $K(i^*) \leftarrow K(i^*)$, $b_j \leftarrow b_j + \bar{a}_{i^*j}$, $T_j \leftarrow T_j + t_{ki^*}$로 갱신한다.

단계 6.3 : $q < u$이면, $q = q + 1$로 두고 단계 6.1로 간다.

단계 7 : $j = j + 1$로 두고, $j \leq n$이면 단계 3으로 간다.

단계 8 : 모든 작업이 할당되었으면 종료하고 그렇지 않으면, $CT = CT + a$로 두고 단계 2로 간다.

제안한 HFALB에서는 선행제약, 공간제약, 할당가능 작업장제약을 만족하는 작업을 구한다(단계 3). 그리고 단계 4에서는 다양한 작업 순서를 유도하기 위하여 이들 작업 중에서 임의 할당 규칙을 사용하여 선택한다. 여기서 사용한 할당 규칙은 아래와 같다.

1. 최대작업시간 규칙 : 작업시간이 가장 큰 작업을 선택.
2. 최대순위위치 규칙 : 자신의 작업시간과 모든 후행작업의 작업시간의 합이 가장 큰 작업을 선택.
3. 최소작업시간 규칙 : 작업시간이 가장 작은 작업을 선택.
4. 최소할당가능작업장 수 규칙 : 할당 가능한 작업장의 수가 가장 작은 작업을 선택.
5. 임의 할당 : 임의로 작업을 선택.

선택된 작업 $i^*$를 행하는 모든 제품의 작업시간 합 $t_{i^*}$를 현 작업부하에 더하여 $CT$를 넘지 않으면 작업 $i^*$를 작업장 $j$에 할당하고 (단계 5), 그렇지 않으면, 작업 $i^*$를 행하는 일부 제품만을 주어진 $CT$를 넘지 않도록 작업장 $j$에 할당 된다(단계 6). 그리고 작업장을 하나 더 증가시킨다 (단계 7). 만약 마지막 작업장에 작업을 할당할 때 주어진 $CT$를 넘게 되면 $CT$를 일정수준 증가하여 처음부터 다시 작업을 할당하게 된다(단계 8). 그리고 이와 같은 절차를 일정 수만큼 반복하여 가장 좋은 값을 HFALB의 해로 둔다.

## 5.2 실험설계 및 파라미터 설정

실험은 19개의 작업을 갖는 Tomopoulos문제[15], 111개 작업을 가진 Arcus문제[4]를 대상으로 하였다. 이들 두 문제 이외에 산학연구[2]에서 얻은 작업의 수가 61개인 문제를 사용하였다. 다양한 문제에 대한 실험을 위하여 <표 1>과 같이 제품 수와 작업장 수를 변화시켜 실험문제로 사용하였다. Arcus문제의 경우 작업장수가 많은 경우를 실험하기 위하여 작업 95의 작업시간을 33491에서 6615로 변경하였다. 그리고, 제품 모델 수가 6개인 Kim문제와 제품 모델 수가 7개와 10개인 Arcus문제에서, 작업시간은 각 문제의 작업시간 중 가장 작은 값과 가장 큰 값 사이에서 임의로 발생하여 사용하였다. 이들 실험 문제의 자료는 [16]을 참고할 수 있다. 그리고 각 실험 문제에서 작업장 공간의 크기는 <표 1>과 같이 두었다. 각 문제에서 모든 작업장의 공간 크기는 동일하게 두었으며, 이 크기는 비교 알고리듬들이 특징이 작동할 수 있도록 설정하였다. 이에 대해서는 5.3절에서 좀더 다루기로 한다. 본 연구에서 사용된 실험문제의 최적해는 알려져 있지 않다. 따라서 이론적인 작업장 최소부하량, $\overline{T}$를 사용하여 제안한 알고리듬의 성능을 비교하고자 한다. 이를 위해 <표 1>의 마지막 열에 $\overline{T}$를 나타내었다. 실험문제 중 prob07~prob15까지는 1000으로 나누어 반올림한 값이다.

제안한 알고리듬은 C++프로그램 언어로 구현되었으며, 2.66GHz Pentium CPU를 장착한 IBM-PC에서 수행되었다. 유전파라미터는 많은 연구에도 불구하고 아직까지 기술(art)적인 문제로 남아있다[1]. 본 연구에서 각 파라미터는 실험을 통하여 결정하였다. 예로써 교차율은 0.2~0.8까지 0.1의 간격으로 10회 반복실험을 통하여 결정하였다. 종료조

건은 각 실험 문제별로 해의 개선이 상당 세대 동안 거의 일어나지 않는 세대로 두었다. 이러한 방법을 통해 얻은 유전파라미터는 다음과 같다. 첫째, 모집단 크기는 200으로, 그리고 토너먼트 선택의 크기는 2로 두었다. 둘째, 교차율은 0.4로, 개체 돌연변이율은 0.015, 그리고 인자 돌연변이율은 0.3을 사용하였다. 마지막으로, 알고리듬의 종료조건은 세대수로 두었다. 작업 수와 제품 수에 따라 해 공간에 차이가 있으므로, Tomopoulos, Kim, Arcus 문제는 각각 100, 400, 750세대에서 종료하였다.

〈표 1〉 실험문제

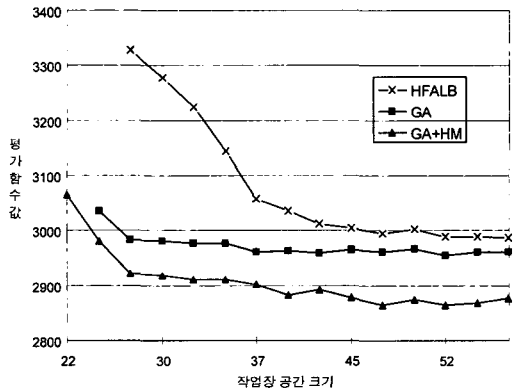
| | 문제 번호 | 작업 수 | 제품 모델 수 | 작업장 수 | 작업장 공간 | $\overline{T}$ |
|---|---|---|---|---|---|---|
| Tomopoulos | Prob01 | 19 | 3 | 3 | 20 | 254.0 |
| | Prob02 | 19 | 3 | 4 | 16 | 190.5 |
| Kim | Prob03 | 61 | 4 | 12 | 35 | 1531.0 |
| | Prob04 | 61 | 4 | 18 | 30 | 1020.7 |
| | Prob05 | 61 | 6 | 12 | 35 | 2858.7 |
| | Prob06 | 61 | 6 | 18 | 30 | 1905.8 |
| Arcus | Prob07 | 111 | 5 | 15 | 121 | 1450.4 |
| | Prob08 | 111 | 5 | 24 | 94 | 906.5 |
| | Prob09 | 111 | 5 | 30 | 77 | 725.2 |
| | Prob10 | 111 | 7 | 15 | 125 | 2730.2 |
| | Prob11 | 111 | 7 | 24 | 105 | 1706.4 |
| | Prob12 | 111 | 7 | 30 | 85 | 1365.1 |
| | Prob13 | 111 | 10 | 15 | 127 | 5328.4 |
| | Prob14 | 111 | 10 | 24 | 125 | 3330.3 |
| | Prob15 | 111 | 10 | 30 | 100 | 2664.2 |

개발한 HFALB의 종료조건은 반복횟수를 사용하였다. 본 연구에서는 Tomopoulos, Kim, Arcus 문제에 대하여 각각 8,000, 32,000, 60,000회 반복 실험하여 얻은 가장 좋은 값을 해로 하였다. 이러한 반복 수는 유전알고리듬에서 생산되는 개체 수와 거의 동일하게 둔 것이다. 그리고, $\alpha = 0.01 \times CT$ 두었다.

## 5.3 작업장 공간 크기

성능 분석에 앞서, 작업장 공간 크기의 변화에 따른 비교 알고리듬들의 특징을 분석해 보자. [그림 6]은 실험문제 Prob05에서 작업장 공간 크기의 변화에 따른 비교 알고리듬의 평가함수 값을 보여주고 있다. 실험값은 10회 반복하여 얻은 값의 평균을 사용하였다.

실험문제에서 작업장 공간 크기를 너무 작게 두면 가능해 영역이 아주 작거나 존재하지 않을 수 있고, 너무 크게 두면 제약으로 작동하지 않게 될 것이다. Prob05 실험문제에서 알고리듬 별로 가능해가 나타나는 최소 작업장 크기는 GA+HM, GA, HFALB에서 각각 평균 22, 25, 28 이었다. 이는 GA가 HFALB보다 탐색능력이 우수함을 보인 것이다. 그리고 또한 GA + HM가 GA에 의해 구한 해를 작업장 공간 제약을 고려하여(4장에서 단계4 참조) 작업부하를 평활화하기 때문이다. [그림 6]으로부터 비교 알고리듬들 간에 약간의 차이는 있으나 작업장 공간이 40 이상이면 해에 거의 영향을 주지 못함을 알 수 있다. 따라서 본 연구에서는 알고리듬간의 성능비교를 위해 각 실험문제 마다 예비 실험을 통해 GA에서 가능해가 존재하는 최소 작업장 크기의 약 1.2배를 작업장 크기로 두었다. 이는 <표 1>과 같다.




[그림 6] 작업장 공간크기에 따른 해의 변화 : Prob05

## 5.4 성능 비교

이 절에서는 앞에서 언급한 HFALB, GA, GA + HM의 성능을 비교 분석한다.

<표 2>에서 'Best', 'Mean', 'Std'는 각각 10회 반

〈표 2〉 알고리듬 성능 비교

| Problem | HFALB | | | GA | | | GA+HM | | | Improved rate1 (%) | Improved rate2 (%) | Difference (%) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Best | Mean | Std | Best | Mean | Std | Best | Mean | Std | | | |
| Prob01 | 255.0 | 256.2 | 0.8 | 254.0 | 254.4 | 0.7 | 254.0 | 254.0 | 0.0 | 0.70 | 0.16 | 0.00 |
| Prob02 | 195.0 | 197.4 | 1.9 | 192.0 | 192.4 | 0.7 | 190.6 | 190.8 | 0.2 | 2.53 | 0.85 | 0.16 |
| Prob03 | 1824.0 | 1907.0 | 59.5 | 1692.0 | 1756.4 | 49.1 | 1578.6 | 1654.4 | 57.4 | 7.90 | 5.81 | 8.06 |
| Prob04 | 1206.0 | 1266.2 | 48.7 | 1100.0 | 1131.6 | 25.5 | 1028.1 | 1061.3 | 22.2 | 10.63 | 6.21 | 3.98 |
| Prob05 | 3075.0 | 3143.7 | 65.6 | 2956.0 | 2976.8 | 28.4 | 2870.9 | 2907.3 | 24.5 | 5.31 | 2.34 | 1.70 |
| Prob06 | 2198.0 | 2316.0 | 88.5 | 2040.0 | 2096.4 | 79.0 | 1909.6 | 1955.9 | 44.5 | 9.48 | 6.70 | 2.63 |
| Prob07 | 1536.5 | 1591.0 | 27.8 | 1480.2 | 1492.6 | 6.1 | 1450.9 | 1454.9 | 6.2 | 6.18 | 2.53 | 0.31 |
| Prob08 | 976.8 | 983.4 | 4.6 | 958.2 | 965.3 | 4.6 | 907.5 | 914.7 | 9.5 | 1.84 | 5.24 | 0.90 |
| Prob09 | 824.5 | 831.5 | 3.6 | 785.7 | 798.7 | 6.8 | 725.7 | 730.8 | 9.9 | 3.95 | 8.50 | 0.77 |
| Prob10 | 2875.4 | 2982.4 | 91.8 | 2788.6 | 2806.4 | 11.2 | 2730.6 | 2733.2 | 4.6 | 5.90 | 2.61 | 0.11 |
| Prob11 | 1880.7 | 1894.7 | 6.6 | 1784.3 | 1801.7 | 11.1 | 1707.1 | 1712.1 | 5.3 | 4.90 | 4.98 | 0.33 |
| Prob12 | 1584.1 | 1610.4 | 17.4 | 1498.9 | 1516.4 | 20.9 | 1366.1 | 1379.5 | 23.4 | 5.83 | 9.03 | 1.05 |
| Prob13 | 5891.4 | 6007.6 | 88.5 | 5471.1 | 5521.6 | 48.0 | 5328.7 | 5362.4 | 51.5 | 8.09 | 2.88 | 0.64 |
| Prob14 | 3499.0 | 3513.7 | 14.4 | 3448.0 | 3471.5 | 17.1 | 3330.7 | 3335.6 | 5.9 | 1.20 | 3.91 | 0.16 |
| Prob15 | 2985.0 | 3001.2 | 15.2 | 2855.2 | 2895.7 | 27.8 | 2665.1 | 2669.3 | 5.7 | 3.51 | 7.82 | 0.19 |

복실험 동안 HFALB와 GA, GA+HM이 보인 해중에서 가장 좋은 값, 평균값, 표준편차를 나타낸다. <표 2>의 Prob07~Prob15 문제에서 제시한 값은 결과 값을 1000으로 나눈 것이다. Improved rate은 각기 HFALB에 대한 GA의 개선율과 GA에 대한 GA + HM의 개선율이다. Improved rate1는 {(HFALB의 Mean GA의 Mean)/HFALB의 Mean}100(%)에 의해 계산되었으며, Improved rate2도 GA와 GA+HM에 대해 동일한 방법으로 계산되었다. Difference는 $\overline{T}$와 제안한 알고리듬에서 얻은 해와의 차이를 나타낸 것으로 {(GA+HM의 Mean - $\overline{T}$)/$\overline{T}$}*100(%)로 계산되었다.

<표 2>로 부터, GA는 HFALB보다 모든 실험문제에서 더 좋은 해를 생산함을 알 수 있다. 또한 대부분의 실험문제에서 낮은 표준편차를 갖는다. 이는 제안한 GA가 안정적으로 좋은 해를 유도할 수 있음을 의미한다. 그리고, GA+HM과 GA의 실험결과의 비교로부터, 복잡도가 높은 FAL에서 한 제품의 동일작업을 여러 작업장에 분산하여 할당하는 방법은 작업의 평활화를 더욱 높일 수 있음을 알 수 있다. 그리고 제안한 알고리듬의 탐색 성능의 우수성은 마지막 열의 Difference로부터 확인 할 수 있다. Tomopoulos, Kim, Arcus 각각의 문제별로 소요된 평균 계산시간은 HFALB가 각각 7초, 610초, 5700초이었고, GA는 2초, 43초, 442초이었다. 그리고 HM은 모든 문제에서 단지 1초 내의 계산시간이 소요되었다.

## 6. 결 론

본 연구에서는 FAL에서 작업을 작업장에 할당하는 문제를 해결하기 위한 방법론으로 유전알고리듬을 제안하였다. 또한, 한 제품의 작업을 여러 작업장에 할당하여 작업 부하를 평활화하는 발견적 기법을 제안하였으며, 이들 기법의 성능을 실험을 통해 비교 분석하였다.

FAL의 작업장에 배치된 조립기계 및 구성요소들은 신중하게 고려되어야 할 요소들이다. 조립작업의 특성 측면과 조립기계의 특성 측면 그리고 그에 따른 제약들은 FAL에 대한 연구를 어렵게 하는 요인이 되고 있다. 이에 관하여 본 연구에서는 제조유연성과 관련 제약들을 제시하였다.

본 연구에서는 다음의 관점에서 실험을 수행하였

다. 첫째, 작업장 크기의 변화에 따른 해를 분석하였다. 둘째, 제안된 발견적기법과 유전알고리듬을 해의 성능 측면에서 비교하였으며 셋째, 한 제품의 동일작업이 여러 작업장에 할당되는 것을 허용하는 경우, 그 효과를 분석하였다. 실험 결과, 제안한 알고리듬은 유망한 하나의 기법임을 알 수 있었다.

추후 연구 주제로, 본 연구에서는 작업할당문제만을 다루고 있으나, 생산의 관점에서 작업할당과 일정계획을 통합적으로 다룰 수 있는 알고리듬의 개발이 요구된다. 또한, 한 제품의 작업이 복수 작업장에 나누어 할당되는 경우, 이를 위하여 순서적으로 문제를 해결하고 있다. 이 문제 또한 통합적으로 동시에 다룰 수 있는 방법론의 개발이 요구된다.

## 참 고 문 헌


[1] 김여근, 윤복식, 이상복, 「메타휴리스틱」, 영지문화사, 1997.

[2] 김여근, 이수연, 김용주, "혼합모델조립라인에서 작업부하의 평활화를 위한 유전알고리듬", 「대한산업공학회지」, 제23권, 제3호(1997), pp.515-532.

[3] Ammons, J.C., C.B. Lofgren and L.F. McGinnis, "A large scale machine loading problem in flexible assembly," *Annals of Operations Research*, Vol.3(1985), pp.319-332.

[4] Arcus, A.L., "An analysis of a computer method of sequencing assembly line operations," Ph.D. dissertation, University of California, 1963.

[5] Graves, S.C. and C.H. Redfield, "Equipment selection and task assignment for multiproduct assembly system design," *The International Journal of Flexible Manufacturing Systems*, Vol.6(1988), pp.361-382.

[6] Goldberg, D.E., *Genetic algorithm in search, optimization & Machine Learning*, Addison-Wesley, Reading, MA., 1989.

[7] Kim, H. and S. Park, "A strong cutting plane algorithm for the robotic assembly line balancing problem," *International Journal of Production Research*, Vol.33(1995), pp.2311-2323.

[8] Kim, Y.K., J.Y. Kim and Y.H. Kim, "A coevolutionary algorithm for balancing and sequencing in mixed model assembly lines," *Applied Intelligence* Vol.13, No.3(2000), pp.247-258.

[9] Lee, H.F. and R.V. Johnson, "A line-balancing strategy for designing flexible assembly systems," *International Journal of Flexible Manufacturing Systems*, Vol.3(1991), pp.91-120.

[10] Lee, H.F. and K.E. Stecke, "An integrated design support method for flexible assembly system," *Journal of Manufacturing Systems*, Vol.15(1996), pp.13-32.

[11] Ram, R. and K.E. Stecke, "Classification and review of FMS scheduling procedures," *Production Planning & control*, Vol.5(1994), pp. 2-20.

[12] Sawik, T., "An interactive approach to bicriterion loading of a flexible assembly system," *Mathematical and Computer Modeling*, Vol.25(1997), pp.71-83.

[13] Sawik, T., "Mixed integer programming for scheduling surface mount technology lines," *International Journal of Production Research*, Vol.39(2001), pp.3219-3235.

[14] Sawik, T., *Production planning and scheduling in flexible assembly systems*, Springer-Verlag, Berlin, 1999.

[15] Thomopoulos, N.T., "Line balancing-sequencing for mixed-model assembly," *Management Science*, Vol.14(1967), pp.59-75.

[16] http://syslab.chonnam.ac.kr/links/data.htm.