# 유전알고리듬을 이용한 복수 U라인의 라인밸런싱
## -Line Balancing in the Multiple U-Type Lines Using Genetic Algorithms-

김동묵
Dong-Mook Kim
김용주
Yong-Ju Kim

**Abstract**

Multiple U-typed production lines are increasingly accepted in modern manufacturing system for the flexibility to adjust to changes in demand. This paper considers multiple U line balancing with the objective of minimizing cycle time considering of moving time of workforce given the number of workstation. Like the traditional line balancing problem this problem is NP-hard. In this paper, we show how genetic algorithm can be used to solve multiple U line balancing. For this, an encoding and a decoding method suitable to the problem are presented. Proper genetic operators are also employed. Extensive computational experiments are carried out to show the performance of the proposed algorithm. The computational results show that the algorithm is promising in solution quality.

## 1. 서론

JIT(Just-In-Time) 생산시스템에서는 생산라인의 형태로 U자형의 라인(이하 U라인이라 함)이 자주 사용된다. U라인에서는 흔히 입·출구 작업이 동일한 숙련작업자에 의해 이루어지면서 제품 한 단위가 출구에서 떠나면 한 단위의 재료가 공정의 입구에 투입된다. 따라서 라인내의 재공품을 항상 일정하게 유지할 수 있으며 작업자간의 작업 불균형을 쉽게 발견할 수 있어 공정개선이 촉구될 수 있다[11]. 또한 U라인에서는 일반적으로 컨베이어를 사용하지 않고 작업자가 보행을 하면서 여러 공정을 담당함으로써 작업자는 좌식작업시 신체의 특정부위를 집중적으로 사용함에 따라 발생하는 작업피로를 줄일 수 있다.

하나의 U라인은 약 4-6명의 작업자가 작업할 수 있도록 흔히 설계된다. 비교적 적은 수의 작업자와 짧은 라인길이는 관리가 용이하여, 자재 및 작업불량, 그리고 재공재고 등의 문제점을 쉽게 파악하여 이를 개선함으로써 품질 및 생산성 향상을 기할 수

동신대학교 산업공학과

있다. 따라서 생산부품의 수가 많거나 공정이 많은 경우 하나의 긴 U라인보다는 여러 개의 짧은 U라인을 결합한 형태가 사용될 수 있다[11].

공장내에서 복수개의 U라인이 있는 경우, 각 라인을 독립적으로 운영하면 특정 작업자 또는 전체 작업자에게 비교적 많은 여유시간이 발생할 수 있어 좋지 않은 균형효율과 과잉생산을 초래할 수 있다. 그러나 이를 결합하여 운영하면 이웃하는 라인들의 작업을 함께 한 작업자에게 할당할 수 있어 균형효율을 더 높일 수 있고 생산율의 변화에 따른 작업 재편성이 용이하게 된다[11].

본 연구에서는 복수개의 U라인의 밸런싱(Multiple U-Line Balancing: MULB) 문제를 다룬다. 라인밸런싱(line balancing) 문제는 생산라인 또는 대상 제품에 부과된 여러 제약들을 어기지 않고, 하나 또는 그 이상의 목적들이 최적이 되도록 작업자에게 작업들을 할당하는 문제이다. 지금까지 라인밸런싱에 관한 대부분의 연구는 직선형태의 컨베이어 조립라인을 대상으로 하여 선행제약과 사이클타임제약을 만족하면서 균형효율이 최대가 되는 작업 할당문제를 다루었다. 직선라인의 밸런싱은 첫 작업장부터 모든 선행작업이 할당된 작업들을 대상으로 하여 적절이 작업할당을 함으로써 이루어진다. 그러나 직선라인과는 다르게 단일 U라인의 작업할당에서는 U자형의 라인 배치이므로 모든 선행작업이 할당된 작업들과 모든 후행작업이 할당된 작업들이 함께 하나의 작업자에게 할당될 수 있다. 이는 직선라인 보다 U라인에서 작업배분의 경우 수가 훨씬 많다는 것을 의미한다. 그러므로 U라인에서는 작업할당에 대한 유연성이 높아 직선라인에 비해 좋은 작업편성이 가능하다[10].

단일 U라인밸런싱 문제에 관한 연구는 Miltenburg와 Wijngaard[10]에 의해 처음 시작된 이후로 아직까지 많은 연구가 이루어지지 않았다[1,4,13]. MULB 문제에 관한 연구는 1998년의 Miltenburg의 연구[9]가 있다. 직선라인밸런싱 문제에 대한 유전알고리듬의 적용은 여러 연구[2,5,6,7]가 있으나, MULB문제에 유전알고리듬을 적용한 연구는 아직 이루어지지 않았다. 따라서 본 연구에서는 MULB 문제에서 작업자의 이동시간을 고려한 유전알고리듬에 관하여 다룬다. 이때 라인밸런싱의 목적은 작업장 수가 주어진 경우에 있어서 사이클타임(cycle time: CT)의 최소화로 둔다. 이를 위하여 MULB 문제에 대한 적절한 개체표현, 해석 그리고 평가방법을 제시한다. 또한 실험을 통하여 제시한 유전알고리듬의 성능을 분석한다.

## 2. 복수U라인

본 연구에서 다루는 복수 U라인은 각각의 독립적인 제품을 생산하는 U라인들을 결합하는 형태로 운영된다. 이들 복수 U라인의 순서와 형태는 주어진 것으로 본다. 또한 작업자의 작업간 이동거리를 계산하기 위하여 각 라인의 작업이 배치되는 위치의 좌표는 주어진 것으로 가정한다[9]. U라인은 일반적으로 수요변동에 의한 생산율의 변화에 작업장의 작업편성을 용이하게 변경할 수 있기 때문에 그 변화에 적절하게 대처할 수 있다. 즉 복수U라인의 형태는 변경시키지 않고 작업자에게 작업을 적절하게 재 편성함으로써 생산율의 변화에 맞춰 필요한 작업자의 수를 가감할 수 있다[11].

복수 U라인에서 작업자의 이동거리는 유크리디언 거리(Euclidean distance)로 한다.

단 작업자가 이동하는데 제약이 있을 경우는 직각거리(rectangular distance)로 한다. 여기서 작업간 거리는 모든 라인에서 동일하게 1m로 보며, 단위길이 1m를 이동하는데 걸리는 시간은 1초로 한다. 한편 직선라인에서는 한 작업자의 이동거리는 첫 작업에서 마지막 작업까지의 거리의 두 배이다.

본 연구에서 다루는 MULB에서 작업장의 형태는 3가지 유형으로 나타난다. 즉 동일한 라인의 작업들이 직선라인 형태로 편성된 직선형태의 작업장(regular station), 동일한 라인에서 라인의 양쪽(전면과 후면)을 횡단하면서 작업을 하는 횡단작업장(crossover station), 한 작업장에 이웃하는 두 라인의 작업들이 편성된 복수라인 작업장(multiline station) 등으로 구분된다[9]. MULB 문제에서 직선형태의 작업장은, 작업길이가 길어짐으로 인해 다른 두 종류의 작업장 형태보다 상대적으로 작업자의 이동시간이 많이 걸린다. 또한 이와 같은 경우에는 작업자간 의사소통이 어렵고, 라인에 문제가 발생시 문제 해결을 더디게 하는 단점을 갖는다[9]. MULB 문제에서 횡단작업장이 될 수 있으면 많이 편성되도록 하는 것은 작업자의 이동거리를 줄이는 효과를 가져옴으로써 밸런싱의 효율을 더 높일 수 있다. 본 연구에서 라인밸런싱의 목적은 주어진 작업장 수 아래서 위에서 언급한 작업자의 이동시간을 고려한 사이클타임의 최소화로 둔다.

## 3. 복수 U라인 밸런싱을 위한 유전알고리듬

유전알고리듬은 해를 표현하는 개체로 구성된 모집단을 변화시키면서 최적해를 탐색해 간다. 모집단의 각 개체를 평가함수에 의해 평가하여, 이를 기초로 적합한 개체를 선별함으로써 새로운 모집단을 형성한다. 새로 형성된 모집단의 개체를 교차 또는 돌연변이를 시켜 새로운 개체를 재생산한다. 이 과정을 종료조건이 만족할 때까지 반복하게 된다

본 연구에서는 MULB 문제에 적합한 개체표현과 유전연산자, 개체해석 및 평가방법을 제안한다. 먼저 본 연구에서 사용되는 기호를 아래와 같이 정의한다.

| | |
|---|---|
| $N$ | : 작업장 수. |
| $NU$ | : 고려중인 라인의 수. |
| $NT_q$ | : 라인 $q$의 작업수, $q$=1,2,.,$NU$. |
| $T$ | : 전체 라인의 총작업시간. |
| $\overline{T}$ | : 전체 라인의 총작업시간 $T$ 의 평균. |
| $d$ | : 단위거리를 이동하는데 걸리는 시간. |
| $ST_j(i)$ | : 작업장 $j$에 작업 $i$를 할당했을 때 이동시간을 고려한 작업장 $j$의 작업시간. |
| $j_e$ | : 작업이 할당되지 않은 작업장 중 가장 이른 작업장. |
| $j_l$ | : 작업이 할당되지 않은 작업장 중 가장 늦은 작업장. |

### 3.1 표현

유전알고리듬을 주어진 문제에 적용하기 위해서는 먼저 문제의 특성에 적합한 개체 표현이 요구된다. 유전알고리듬에서 개체는 자연스럽고 명확하게 잠재해를 나타내야 하고, 가능한 잠재해가 중복 표현되지 않아야 한다. 중복 표현될 경우 해공간의 효율적인 탐색이 이루어질 수 없다. 또한 표현방법은 유전연산자와 밀접하게 관련되어 있다. 개체가 갖는 중요한 정보가 유전연산자에 의해 추출되고 자손에게 잘 전파될 수 있도록 개체를 표현해야 한다.

본 연구에서는 작업자의 작업간 이동시간을 고려해야 하기 때문에, 그룹번호표현을 사용하되 다루는 문제의 특성에 맞게 수정하여 사용하였다. 개체에서 인자위치는 각 라인의 작업을 나타내며, 인자값은 해당작업이 편성된 작업장의 번호와 작업이 배치되는 위치번호의 쌍으로 나타낸다. 예를 들어 아래와 같은 개체가 있다고 하자.

▼라인구분

| 1 | 1 | 4 | 3 | 1 | 2 | 2 | 4 | 3 | 3 | 5 | 2 | 5 | 5 | 5 | 5 | 5 | 4 | ←작업장 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 5 | 8 | 10 | 3 | 4 | 6 | 7 | 9 | 2 | 1 | 3 | 6 | 4 | 5 | 7 | 8 | ←위치 |

이 개체표현에서 인자의 위치는 작업을 나타내며, 개체의 첫번째 행은 작업장 번호를, 두번째 행은 작업이 할당되는 위치를 나타낸다. 개체에서 작업장 2에 편성된 작업과 그 작업이 할당된 위치를 보면, 라인 1의 작업 6, 7은 라인1의 위치 3과 4에, 라인 2의 작업 2는 라인 2의 위치 1에 할당되어 있다. 이 표현은 개체 해석이 용이할 뿐 아니라 중복표현이 없고, 각각의 U라인에서 그 작업의 위치와 편성된 작업장을 쉽게 알 수 있다는 장점을 갖는다. 또한 개체에서 이웃하는 두 라인에 속하는 작업들로 편성된 작업장(복수라인 작업장)을 쉽게 나타낼 수 있다.

### 3.2 초기 모집단

초기 모집단은 복수 U라인의 작업할당제약을 만족하는 임의의 가능해들로 구성한다. 초기해를 생성할 때 작업장 수가 주어진 경우, 작업장에 작업할당을 위한 편성기준은 일반적으로 총작업시간의 평균 $\overline{T}$를 사용할 수 있다. 그러나 본 연구에서는 작업자의 작업간 이동시간을 고려하기 때문에 $\overline{T}$를 편성기준으로 사용할 수 없다. 따라서 본 연구에서는 초기해를 생성할 때의 편성기준은 $\hat{T}$로 하며 다음과 같은 구간에서 초기해를 구할 때마다 임의로 정한다.

$$(\overline{T}+2+d\sum_{q=1}^{N}NT_q/N,\ \overline{T}+2d\sum_{q=1}^{N}NT_q/N)$$

위 구간에서 하한은 모든 작업장에 대하여, 이상적으로 작업들이 라인의 전면과 후면에 고루 할당되어 횡단작업장을 구성했을 때의 작업시간과 이동시간의 평균이다. 상

한은 모든 작업장의 작업편성이 직선라인 형태일 때의 작업시간과 이동시간의 평균이다. 이와 같은 편성기준으로 초기해를 생성함으로써 초기 모집단은 다양한 잠재해로 구성될 수 있다. 이는 편성기준을 하나로 정했을 때 보다 상대적으로 다양하게 해공간을 탐색할 수 있어, 좋은 해를 빨리 찾을 수 있는 가능성을 높일 수 있다. 여기서 2장에서 언급한 바와 같이 모든 라인에 있어서 작업간 거리는 동일하게 1m로 보며, 또한 단위길이를 이동하는데 걸리는시간 $d$=1로 둔다. 본 연구에서 제시한 초기해 생성절차는 다음과 같다.

단계 1. $j=j_e$로 둔다.

단계 2. 미할당 작업이 있는 가장 이른 라인을 $k$로 둔다. 할당가능 작업집합 $AT$를 다음과 같이 구한다.

a) 라인 $k$의 접점위치가 모두 할당되었거나 미할당이면,
미할당 선행작업이 없거나 미할당 후행작업이 없는 라인 $k$의 미할당 작업들을 $AT$로 둔다. 단계 3으로 간다.

b) 라인 $k$의 두 개의 접점위치 중 1개만 미할당인 경우

i) 미할당 접점위치가 순방향이면, 미할당 선행작업이 없는 라인 $k$의 미할당 작업들을 $AT$로 둔다.

ii) 미할당 접점위치가 역방향이면, 미할당 후행작업이 없는 라인 $k$의 미할당 작업들을 $AT$로 둔다.

iii) $k < NU$이고 라인 $k$의 기할당 접점위치가 작업장 $j$에 할당되어 있으면, 미할당 선행 또는 후행작업이 없는 라인 $k$+1의 미할당 작업들을 $AT$에 추가한다.

단계 3. 다음 조건을 만족하는 미할당 작업 집합 $AT'$를 구한다.

a) $j<j_l$이면 (마지막 작업장이 아니면), $AT' = \{i \mid ST_j(i) < \hat{T}, i \in AT\}$로 둔다.
$AT' = \phi$이면 $j=j+1$로 두고 단계 2로 간다.

b) $j=j_l$이면 $AT' = AT$.

단계 4. $AT'$중 임의의 작업을 선택하여 $i^*$로 둔다. 다음과 같은 규칙에 의해 작업 $i^*$의 할당 위치를 정한다.

a) 작업 $i^*$가 선행제약을 만족하는 작업이면 가장 이른 미할당 위치를 선택한다.

b) 작업 $i^*$가 후행제약을 만족하는 작업이면 가장 늦은 미할당 위치를 선택한다.

작업 $i^*$를 작업장 $j$에 할당한다. 미할당 작업이 없으면 종료하고, 그렇지 않으면 단계 2로 간다.

단계 1에서는 모든 개체를 빈 개체로 시작하기 때문에 작업장 $j=j_e$=1로 시작한다.

단계2는 미할당 작업이 있는 가장 이른 라인부터 할당가능 작업집합을 구하는 과정이다. 할당가능 작업집합은 접점위치에 작업의 할당여부에 따라 그 구성이 달라진다. 여기서 접점위치란 현재 편성중인 라인 $k$의 위치 중 다음 라인 $k$+1의 부품의 투입 및 출구와 이웃한 위치로 한다. 본 연구에서는 Miltenburg의 연구[9]와 같이 접점위치는 최대 2로 둔다. 따라서 현재 편성 중인 라인 $k$의 접점위치가 모두 미할당이면, 라인 $k$의 선행공정도에서 미할당 선행 작업 또는 후행작업들이 할당가능 작업집합 $AT$가 된다. 그러나 접점위치 중 하나의 접점 즉 순방향(역방향) 접점위치에 작업이 할당되지 않았으면, 미할당 선행작업(후행작업)이 없는 작업집합이 할당가능 작업집합이다. 또한 이미 작업이 할당된 접점이 속한 작업장의 작업편성이 완료되지 않는 경우에는 이웃하는 라인의 미할당 선행 또는 후행작업들이 없는 작업집합이 할당가능 작업집합에 추가된다. 복수U라인에서 마지막 라인은 접점이 없기 때문에 이 라인에서는 접점이 모두 할당된 것으로 본다. 선행공정도에서 순방향(forward)이란 첫 작업부터 이후의 작업으로 탐색하는 방향을, 역방향(backward)이란 마지막 작업부터 시작하여 첫 작업쪽으로 탐색하는 방향을 말한다. 라인의 위치제약에서는 순방향이란 부품의 투입구부터 이후의 위치로 탐색하는 방향을, 역방향이란 부품의 출구부터 시작하여 투입구쪽으로 라인을 따라 탐색하는 방향을 나타낸다.

단계 3에서는 할당가능 작업집합 $AT$중에서 해당 작업이 작업자에 할당되었을 경우, 작업자의 이동시간을 고려한 작업장 $j$의 작업시간이 작업편성기준 $\hat{T}$ 를 넘지않는 미할당 작업집합 $AT'$를 구한다.

단계 4는 미할당 작업집합 $AT'$중에서 임의로 한 작업 $i^*$를 선택한 후 이 작업 $i^*$의 선후행제약에 따라 배치할 위치를 정하고 작업 $i^*$를 작업장 $j$에 할당하는 과정이다.

### 3.3 평가함수와 선별

유전알고리듬에서는 개체의 생존에 대한 적응척도로서 평가함수가 사용된다. 개체의 적응도를 평가하는 함수로는 흔히 대상문제의 목적함수를 사용한다. 본 연구에서는 다루는 문제의 목적함수인 작업장 수($N$)가 주어진 경우에 있어서 사이클타임($CT$)을 평가함수로 사용한다.

선별(selection)은 각 개체의 평가함수 값을 기초로 다음 세대에 생존할 개체를 선택하는 과정이다. 선별방법에는 확률바퀴 선별(roulette wheel selection), 순위 선별(ranking selection), 토너먼트 선별(tournament selection) 등이 있다[3,8]. 본 연구에서는 토너먼트 선별을 사용한다. 본 연구에서는 토너먼트 크기 $l$=2로 두었다.

유전알고리듬에서 선별된 후보개체들은 유전연산을 하게 된다. 유전연산을 한 후, 다음 세대를 최종적으로 구성하기 전에 우수개체 보존전략(elitism)을 사용하였다. 이 전략은 가장 좋은 개체가 유전연산을 통해 손실되는 것을 방지하기 위하여 현재까지 생산된 개체 중에서 가장 좋은 개체를 모집단에 보존시키는 전략이다.

### 3.4 유전연산자

유전연산자는 교차와 돌연변이가 있다. 교차는 두 부모가 갖는 인자를 결합하여 자손을 생산하는 과정이다. 교차는 선별과정에서 선택된 우수한 개체들의 형질들을 모집단내에 전파하는 역할을 한다. 이를 위해서 교차는 부모의 좋은 형질이 가능한 파괴되지 않고 자손에 상속될 수 있어야 한다. 돌연변이는 개체에 새로운 형질이 생성되는 것으로, 한 개체에서 몇 개의 인자를 임의로 변화시키는 과정이다.

본 연구에서 교차는 일점교차를 다루는 문제에 적합하게 수정하여 사용하였다. 그 절차는 아래와 같다. 교차할 부모를 $p_1$, $p_2$로 두자.

단계1. 양의 정수 구간 [1~(*N*-1)] 중 임의의 정수 *R*1을 선택한다.

단계2. 부모 $p_1$으로부터 작업장 1부터 작업장 *R*1의 정보(작업과 위치의 쌍)를 자손 $o_1$에 복사한다.

단계3. 단계 2에서 복사된 작업과 동일한 부모 $p_2$의 작업이 속한 작업장 번호 중 가장 큰 것을 *R*2로 둔다.

단계4. 부모 $p_2$로부터 *R*2 이후의 작업장들의 정보를 $o_1$에 복사한다.

단계5. $o_1$에서 할당되지 않은 작업과 위치를 재할당한다. 재할당방법은 3.5절에서 다룬다.

$p_1$과 $p_2$ 의 역할을 바꾸어 자손 $o_2$를 생산한다.

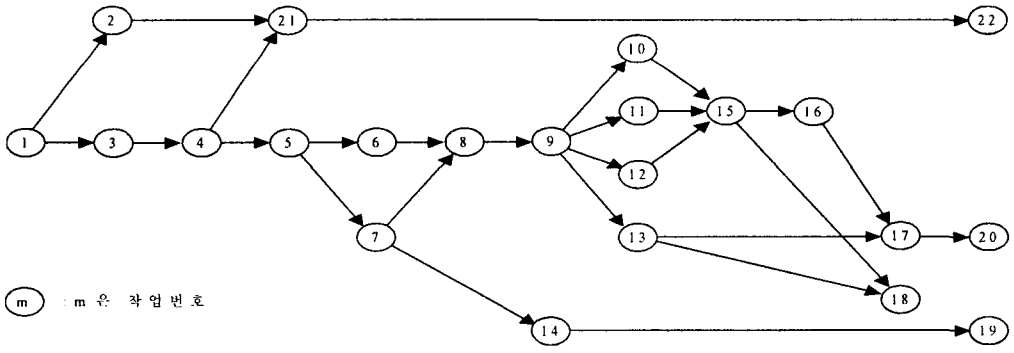
단계 2는 단계 1에서 임의로 선택된 절단점 *R*1까지의 부모 $p_1$의 작업장 정보를 자손에 상속하는 과정이다. 단계 3는 부모 $p_1$으로부터 자손에 상속한 작업과 동일한 $p_2$의 작업이 속한 작업장 번호 중 가장 큰 것을 확인한다. 이것은 자손에게 상속된 $p_1$의 정보 즉, 작업과 위치가 중복되는 것은 $p_2$에서 지울 뿐만 아니라 중복된 작업 또는 위치가 속한 작업장의 나머지 작업과 위치들을 삭제하는 것을 의미한다. 단계 4는 단계 3에 의해 손상되지 않은 부모 $p_2$의 작업장 정보를 자손에게 상속하는 과정이다. 단계 5는 자손에게 상속되지 않은 작업과 위치를 미할당으로 두고 재할당하는 과정이다. 이 절차는 3.5절에서 설명한다.

유전알고리듬에서 돌연변이는 다양한 해공간의 탐색을 가능하게 하여, 부분최적에 조기수렴하는 것을 방지하는 역할을 한다. 본 연구에서 제안한 돌연변이는 돌연변이율에 의해 선택된 개체에 대해 다음과 같이 행한다. 먼저 양의 정수 구간 [1~(*N*-1)] 중 임의의 정수를 선택하여 *R*1으로 두고, [(*R*1+1)~*N*] 중 임의의 정수를 선택하여 *R*2로 둔다. 그리고 개체에서 R1부터 R2까지의 작업장에 속하는 작업과 위치를 미할당으로 두고 재할당한다. 이때 재할당 방법은 제안한 교차에서와 같은 방법을 사용한다. 돌연변이는 개체 단위로 한다.

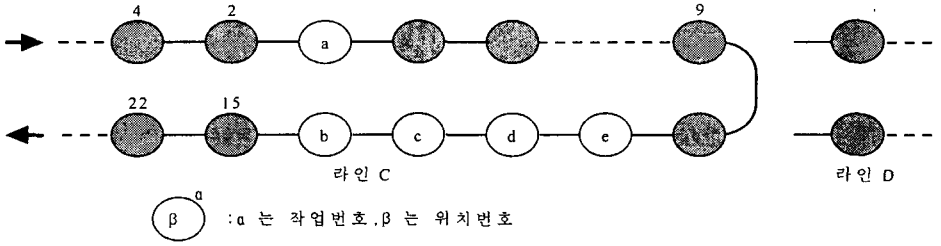
### 3.5 재할당 방법

제안한 유전연산에서 인자값이 결정되지 않는 인자(미할당 작업과 위치)들의 인자값을 부여하는 과정이 필요하다. 즉, 유전연산 과정에서 삭제된 작업과 위치의 재할당이 요구된다. 이때 가능해이면서 좋은 해을 생산하기 위하여 문제가 갖는 특정정보를 이용한다.

유전연산에 의해 특정 라인의 미할당 위치들은 연속적이지 않을 수 있다. <그림 1>의 (b)에서 유전연산에 의해 위치 a, b, c, d, e가 미할당 위치라고 하자. 이와 같은 경우는 미할당 위치들이 연속적이지 않다. <그림 1>에서 음영으로 표시된 위치는 작업이 할당되어 있다고 보자. 이러한 미할당 위치에 할당될 수 있는 작업들은 한정적이어서 해의 변화를 제한하는 결과를 초래할 수 있다. 예로, <그림 1>의 선행공정도에서 작업 10~13과 21을 미할당 위치 a~e에 재할당하는 경우를 보자. 작업 21의 선행작업 2가 순방향 탐색으로 라인의 전면에 할당되었고, 후행작업 22는 역방향 탐색으로 라인의 후면에 할당되었다. 그러므로 작업 21은 미할당 위치 어느 곳이나 할당가능하다. 그러나 미할당 작업 10~13의 선행작업 9가 미할당 위치 a 뒤에 할당되어 있기 때문에 이들 작업은 반드시 미할당 위치 b~e에 할당되어야 한다. 이와 같은 경우에는 재할당에 의한 해의 변화가 제한적일 수 있다. 따라서, 본 연구에서는 유전연산에 의해 발생되는 특정라인의 미할당 위치들 사이에 존재하는 기할당 위치와 이 위치에 할당된 작업들을 미할당 상태로 만들어 미할당 위치가 연속될 수 있도록 하였다. 즉, 재할당 대상이 되는 작업과 위치는 유전연산에 의해 발생되는 미할당 작업과 위치 이외에 다음과 같은 조건에 해당하는 작업과 위치를 포함한다.




(a) C부품의 선행공정도




(b) 라인 C의 재할당위치

<그림 1> 재할당을 위한 작업과 위치의 예

조건 1. 미할당 위치가 존재하는 라인에 대해 가장 이른 미할당 위치와 가장 늦은 미할당 위치 사이에 이미 작업이 할당된 위치가 존재하면, 그 위치와 그 곳에 할당된 작업들을 미할당 위치와 작업으로 둔다.

조건 2. 미할당 위치가 있는 가장 마지막 라인의 접점 위치가 조건 1에 의해 미할당 위치가 되면, 이 위치가 할당되었던 작업장에 할당된 다음 라인의 작업들을 미할당 작업으로 둔다. 단, 이 작업들은 위치가 고정된 작업으로 두어, 재할당 절차에서 이 작업들의 할당 위치는 변하지 않도록 한다.

조건 2에서 보면 조건 1에 의해 접점위치가 미할당 위치로 되었을 때, 이 접접위치가 속했던 작업장에 다음 라인의 작업들이 편성된 경우는 이들 작업이 할당된 위치는 고정하되 작업장 편성만 새롭게 한다. 그 이유는 유전연산에 의해 부모로부터 상속된 정보의 손실을 적게 하기 위해서다.

재할당 시 새롭게 생성되는 작업장은 유전알고리듬이 진행되는 과정에서 찾은 최선해의 사이클타임 이하로 작업과 위치들이 할당되어야만 최선해 보다 좋은 해가 생산될 수 있다. 따라서, 본 연구에서는 재할당 시 작업할당 기준은 다음과 같이 두었다.

$\hat{T}$ = 최선해의 $CT$

이러한 기준은 마지막 미할당 작업장의 완료시간을 늦출 수 있지만, 마지막 작업장을 제외한 새롭게 생성된 작업장의 정보는 모집단에 전파되어 좋은 해를 생성하는 정보로 활용될 수 있다.

재할당은 미할당 위치가 존재하는 가장 이른 라인부터 작업을 재할당한다. 제안된 재할당 방법은 앞에서 설명된 초기해 생성방법과 유사하다. 제안한 재할당 절차의 할당기준은 초기해 생성에서 사용하였던 $\hat{T}$ 대신에 위에서 제시한 최선해의 사이클타임 $\hat{T}$를 사용한다. 그러나 할당가능 작업집합 $AT$를 구하는 절차는 동일하다. 제안된 재할당 절차 중 초기해 생성절차와 다른 부분을 정리하면 다음과 같다. 여기서 $D_j(i)$를 작업장 $j$에 작업 $i$를 할당했을 때의 작업자 이동거리의 합으로 둔다.

단계 3. 다음 조건을 만족하는 미할당 작업 집합 $AT'$를 구한다.

a) $j < j_l$이면 (마지막 작업장이 아니면) $AT' = \{i | ST_j(i) < \hat{T},\ i \in AT\}$로 둔다.

$AT' = \phi$이면, $j = j+1$로 두고 단계 2로 간다.

b) $j = j_l$이면 $AT' = AT$.

단계 4. $AT'$ 중 $D_j(i)$, $i \in AT'$이 최소인 작업을 선택하고 복수개 존재하면 이들 중 임의로 선택하여 $i^*$로 둔다. 선택된 작업 $i^*$의 할당 위치가 미리 정해져 있으면, 정해진 위치에 할당한다. 그렇지 않으면, 다음과 같은 규칙에 의해 작업 $i^*$

의 할당 위치를 정한다.

a) 작업 $i^*$가 선행제약을 만족하는 작업이면 가장 이른 미할당 위치를 선택한다.

b) 작업 $i^*$가 후행제약을 만족하는 작업이면 가장 늦은 미할당 위치를 선택한다.

작업 $i^*$를 작업장 $j$에 할당한다. 미할당 작업이 없으면 종료하고, 그렇지 않으면 단계 2로 간다.

초기해 생성절차와 다르게 단계 3의 a)에서 $ST_j(i)$가 $\hat{T}$보다 작은 작업들을 $AT'$로 둔다. 그리고 초기해 생성의 단계 4에서는 $AT'$중 임의의 작업을 선택하여 $i^*$로 하였지만, 재할당 절차에서는 $AT'$중 작업장 $j$의 총이동거리 $D_j(i)$를 최소로 하는 작업을 선택하고, 만약 복수 개 존재하면 임의로 선택하여 $i^*$로 한다. 그러나 선택된 작업 $i^*$의 할당위치를 정하는 규칙은 초기해 생성절차와 동일하다.

## 4. 실험 및 분석

제안한 유전알고리듬의 성능 분석을 위하여 기존의 발견적 기법과 비교하였다. 비교한 발견적 기법은 할당가능한 작업중에서 작업시간이 가장 큰 것을 할당하는 최대 작업시간 규칙(MAXD-0)[12]과 임의로 할당하는 임의 할당방법(RND-0)을 사용하였다. 또한 본 연구에서는 작업자의 이동시간을 고려하기 때문에 작업 $i$의 작업시간과, 작업장에 작업 $i$가 할당될 때 작업자의 이동거리( $D_j(i)$)의 합의 비율( 작업 $i$의 작업시간/ $D_j(i)$)이 최대인 작업을 할당하는 할당방법(MAXTD)를 추가하여 제안한 알고리듬과 비교하였다. 이 방법은 작업자의 이동거리가 적으면서 작업시간이 큰 작업이 작업장에 할당되도록 한다. 실험 대상문제는 Miltenburg[9]가 실험하기 위하여 만든 5개의 문제가 사용되었다. 즉, Monden[11]이 제시한 예를 문제로 구성한 문제(ORIP), 이 ORIP문제에서 각 U라인에서 몇 개의 작업에 대해 이동제약을 고려한 문제(SOTM), ORIP문제에서 각 U라인의 위치에 작업이 배정되었다고 보고 작업장 편성만을 고려한 문제(NOTM), 라인밸런싱의 벤치마크 문제를 각각의 U라인으로 본 문제(SSBP), 라인밸런싱의 벤치마크 문제들을 각각 두개씩 묶어서 U라인으로 하여 보다 큰 복수 U라인으로 구성한 문제(LSBP) 등이 사용되었다.

적절한 유전파라미터의 결정을 위해 예비실험을 실시하였다. 예비실험 결과 모집단의 크기는 50으로 두었다. 모집단의 크기가 작은 경우는 모집단의 개체들이 조기 수렴하는 경향이 있으며, 너무 큰 경우에는 모집단의 운용에 많은 계산시간과 자원이 소요된다. 예비실험에서 모집단의 크기를 50보다 크게 하는 경우, 모집단의 다양성은 증가되지만 한정된 수의 개체를 재생산하는 상황아래서 해의 탐색효율은 오히려 저하되는 결과를 가져왔다.

예비실험에서 교차율과 돌연변이율은 각각 (0.5~0.8), (0.1~0.3)의 범위내로 설정한

경우가 좋은 결과를 나타내었다. 따라서 본 연구에서는 문제의 크기가 비교적 작은 문제(ORIP, SOTM, NOTM)는 교차율과 돌연변이율을 각각 0.7과 0.2로 두었다. 반면에 문제의 크기가 큰 문제(SSBP, LSBP)는 교차율은 0.8로, 돌연변이율은 0.3으로 하였다. 알고리듬은 생성된 개체의 수가 20,000개에 이르면 종료하였다. 제안한 유전알고리듬과 발견적 기법들은 C++언어로 구현하였으며, 300MHz Pentium CPU를 장착한 IBM-PC에서 수행되었다.

### 4.1 기존의 발견적 기법의 수정

비교실험을 위하여 기존의 발견적 기법을 MULB 문제에 맞게 수정하여 적용하였다. 수정된 발견적 기법의 적용절차는 초기해 생성 절차와 유사하다. 차이점은 작업장에 작업할당의 편성기준으로 $\hat{T}$ 대신 이론적인 사이클타임 $\tilde{c}(=T/N)$로부터 시작한다. 그리고 초기해 생성절차의 단계 4와 같이 $AT'$중 임의의 작업을 선택하지 않고 정해진 특정 발견적 할당규칙에 따라 선택하는 것이 다르다. 이와 같이 초기해 생성절차를 수정하여, 작업편성 기준인 이론적인 사이클타임 $\tilde{c}$부터 일정률씩 증가시키면서 구해진 작업장 수가 $N$과 같아질 때까지 반복 수행하여 해를 구하였다. 수정된 발견적 기법의 적용절차는 다음과 같다.

단계 1. $j=j_e=1$, $\tilde{c}=T/N$로 둔다.

단계 2. 초기해 생성 절차 단계 2와 동일.

단계 3. 다음 조건을 만족하는 미할당 작업 집합 $AT'$를 구한다.

$AT'=\{i|ST_j(i)<\tilde{c},\ i\in AT\}$로 둔다.

$AT'=\phi$이면 $j$ = $j$+1로 두고 단계 2으로 간다.

단계 4. $AT'$중 정해진 특정 발견적 할당규칙에 따라 작업을 선택하여 $i^*$로 둔다. 다음과 같은 규칙에 의해 작업 $i^*$의 할당 위치를 정한다.

a) 작업 $i^*$가 선행제약을 만족하는 작업이면 가장 이른 미할당 위치를 선택한다.

b) 작업 $i^*$가 후행제약을 만족하는 작업이면 가장 늦은 미할당 위치를 선택한다.

작업 $i^*$를 작업장 $j$에 할당한다. 미할당 작업이 있으면 단계 2로 간다.

미할당 작업이 없는 경우에는

i) 구한 작업장 수가 $N$과 같으면 종료하고

ii) 그렇지 않으면 $\tilde{c}=\tilde{c}+0.01\tilde{c}$로 하고 단계 1로 간다.

단계 4에서 작업 $i^*$를 선택하는 규칙은 RND-0, MAXD-0, MAXTD 등이 사용되었다. 본 연구에서 제안한 재할당 절차에서 보면, 단계 4에서 $AT'$중 $D_j(i)$, $i\in AT'$이

최소인 작업을 선택하고 복수개 존재하면 이들 중 임의로 선택하였다. 이와 같은 할당규칙(RND-1)을 위 적용절차의 발견적 할당규칙에 추가하였다. 그러나 이러한 작업 선택기준(RND-1)을 따르지 않고, $AT'$중 $D_j(i)$, $i \in AT'$이 최소인 작업을 선택하되 복수개 존재하면 최대 작업기간을 갖는 작업을 선택하는 방법(MAXD-1)도 좋은 발견적 기법이 될 수 있다. 따라서 위 발견적 기법의 적용절차에서 할당규칙은 5개(RND-0, MAXD-0, MAXTD, RND-1, MAXD-1)로 하였다. 이들 발견적 기법들과 본 연구에서 제안한 알고리듬을 실험을 통하여 비교하였다.

## 4.2 비교 분석

제안한 유전알고리듬과 발견적 기법의 적용결과가 <표 1>에 제시되었다. 1열은 각 문제명과 U라인의 수를, 2열은 작업장 수를 나타낸다. 3열부터 7열까지는 전술한 발견적 기법을 각각 2,000번 반복 실험하여 구한 해중에서 가장 좋은 해를 나타내었다. 8열과 9열은 본 연구에서 제안한 유전알고리듬을 10회 수행하여 얻은 가장 좋은 해와 가장 나쁜 해이고, 10열과 11열은 10회 수행결과의 평균과 표준편차이다. 마지막으로 12열은 발견적 기법으로 구한 해중 가장 좋은 해($H_{best}$)에서 유전알고리듬의 평균($GA_{mean}$)을 뺀 값을 $H_{best}$로 나누어 100을 곱한 값이다. 즉, 개선율 =[ ($H_{best}$- $GA_{mean}$)/ $H_{best}$]×100이다.

실험결과, 본 연구에서 제시한 유전알고리듬이 각 문제의 모든 작업장 수에 대하여 발견적 기법보다 좋은 결과를 보여 주었다. 본 연구는 작업자의 이동시간을 고려하기 때문에 발견적 기법간의 해의 결과를 보면, 발견적 기법 중 거리를 고려한 기법들(MAXTD, RND-1, MAXD-1)이 더 좋은 결과를 보이고 있다. 특히 이들 중 거리와 작업시간을 동시에 고려한 MAXTD와 MAXD-1이 발견적 기법 중 그 결과가 우수하였다. 그러나 유전알고리듬으로 구한 해 중 가장 나쁜 해도 모든 발견적 기법보다 좋은 결과를 나타내었다. 그리고 구한 해들의 표준편차가 비교적 낮았다. 이는 제안한 기법이 안정적임을 알 수 있다.

## 5. 결론

본 연구에서는 유전알고리듬을 이용하여, 주어진 작업장 수에 대하여 작업자의 이동시간을 고려한 사이클타임의 최소화를 목적으로 하는 복수 U라인의 라인밸런싱 문제를 다루었다. 복수 U라인의 특성과 작업자의 이동거리를 반영한 효율적인 유전알고리듬의 개발을 위하여, 해를 자연스럽게 나타낼 수 있는 표현방법과 그에 따른 유전연산자를 개발하였다. 제안한 알고리듬의 성능을 평가하기 위하여, 여러 실험문제를 사용하여 발견적 기법들과 그 성능을 비교 분석하였다. 실험결과, 제안한 알고리듬은 비교된 기법들 보다 그 성능이 우수하였다.

유전알고리듬의 한 장점은 최적화 문제에서 목적함수와 제약식의 변화에 유연하다는 것이다. 따라서 제안한 알고리듬은 작업자의 이동시간을 고려한 복수 U라인에서 작업장의 작업량 평활화 문제에 쉽게 적용할 수 있다.

<표 1> 유전알고리듬과 발견적 기법과의 해의 성능비교

| 문제 | 작업장 | 발견적 기법 | | | | | GA | | | | 개선율 (%) |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | 수 | RND-0 | MAXD-0 | MAXTD | RND-1 | MAXD-1 | Best | Worst | Mean | s.d. | |
| ORIP | 6 | 84.42 | 85.00 | 81.69 | 81.94 | 81.69 | 80.41 | 80.41 | 80.41 | 0.00 | 1.56 |
| 6 | 7 | 72.84 | 71.42 | 70.72 | 71.24 | 70.72 | 69.41 | 69.41 | 69.41 | 0.00 | 1.85 |
| | 8 | 63.76 | 63.12 | 63.75 | 62.00 | 61.88 | 60.00 | 60.00 | 60.00 | 0.00 | 3.04 |
| | 9 | 57.90 | 58.39 | 56.67 | 56.67 | 56.67 | 54.41 | 54.41 | 54.41 | 0.00 | 3.98 |
| | 10 | 53.01 | 52.55 | 50.50 | 51.89 | 50.50 | 49.16 | 50.41 | 49.54 | 0.49 | 1.90 |
| SOTM | 6 | 84.57 | 85.85 | 81.69 | 81.81 | 81.69 | 80.41 | 80.41 | 80.41 | 0.00 | 1.56 |
| 6 | 7 | 72.88 | 72.86 | 71.42 | 71.52 | 71.42 | 71.41 | 71.41 | 71.41 | 0.00 | 0.01 |
| | 8 | 63.82 | 63.12 | 63.75 | 61.68 | 61.88 | 60.00 | 60.00 | 60.00 | 0.00 | 2.72 |
| | 9 | 58.22 | 58.39 | 56.67 | 56.83 | 56.67 | 55.41 | 55.83 | 55.79 | 0.13 | 1.55 |
| | 10 | 53.12 | 52.03 | 50.50 | 50.83 | 50.50 | 50.41 | 50.41 | 50.41 | 0.00 | 0.17 |
| NOTM | 6 | 84.54 | 85.00 | 84.16 | 81.69 | 81.69 | 81.00 | 81.00 | 81.00 | 0.00 | 0.84 |
| 6 | 7 | 72.84 | 73.59 | 72.14 | 71.42 | 71.42 | 71.41 | 71.41 | 71.41 | 0.00 | 0.01 |
| | 8 | 62.96 | 61.88 | 61.27 | 60.42 | 61.27 | 60.00 | 60.00 | 60.00 | 0.00 | 0.70 |
| | 9 | 58.25 | 57.81 | 57.81 | 57.67 | 57.81 | 56.41 | 56.83 | 56.79 | 0.13 | 1.52 |
| | 10 | 52.93 | 52.03 | 52.55 | 51.97 | 50.50 | 50.41 | 50.41 | 50.41 | 0.00 | 0.18 |
| SSBP | 18 | 68.39 | 67.68 | 67.01 | 67.66 | 67.68 | 63.41 | 64.06 | 63.74 | 0.29 | 4.88 |
| 6 | 19 | 64.84 | 62.85 | 62.85 | 64.11 | 62.85 | 60.00 | 61.41 | 61.01 | 0.53 | 2.93 |
| | 20 | 62.20 | 59.12 | 60.31 | 61.90 | 60.31 | 57.41 | 59.12 | 58.61 | 0.52 | 0.86 |
| | 21 | 59.92 | 56.78 | 56.78 | 59.42 | 58.50 | 55.16 | 56.00 | 55.80 | 0.33 | 1.73 |
| | 22 | 57.33 | 55.11 | 54.03 | 56.74 | 56.23 | 53.24 | 53.41 | 53.33 | 0.09 | 1.29 |
| LSBP | 18 | 74.14 | 71.82 | 71.82 | 73.48 | 71.10 | 69.00 | 70.41 | 69.66 | 0.45 | 2.03 |
| 4 | 19 | 70.34 | 70.10 | 68.04 | 69.69 | 69.40 | 66.00 | 67.41 | 66.57 | 0.43 | 2.15 |
| | 20 | 67.38 | 67.26 | 65.28 | 66.69 | 64.63 | 62.41 | 63.41 | 63.20 | 0.33 | 2.21 |
| | 21 | 64.74 | 62.17 | 62.17 | 64.18 | 62.17 | 59.41 | 61.24 | 60.24 | 0.47 | 3.10 |
| | 22 | 62.30 | 59.35 | 60.54 | 61.66 | 59.35 | 57.41 | 58.65 | 58.05 | 0.32 | 2.18 |

## 참고문헌


[1] 김여근, 김재윤, 김동묵, 송원섭, "U라인 라인밸런싱을 위한 분지한계법," 한국경영과학회지, 23권 12호, (1998), pp. 83-101.

[2] Anderson, E. J. and M. C. Ferris, "Genetic Algorithms for Combinatorial Optimization: the Assembly Line Balancing Problem," *ORSAJ. Computing*, Vol. 6, (1994), pp. 161-173.

[3] Goldberg, D. E., *Genetic Algorithms in Search, Optimization & Machine Learning*, Addison-Wesly, Reading, (1989).

[4] Hwang, H., J.U. Sun, and T.H. Yoon, "U-line Line Balancing with Simulated Annealing," *Proceedings of the First ASIA-PACIFIC Decision Sciences Institute Conference*, Hong Kong, (1996), pp.101-108.

[5] Kim, Y. K., Y. J. Kim, and Y. K. Cho, "A Heuristic-Based Genetic Algorithm for Workload Smoothing in Assembly Lines," *Computers and Operations Research*, Vol.25, No.2,(1998), pp. 99-111.

[6] Kim, Y. K., Y. J. Kim, and Y. H. Kim, "Genetic Algorithms for Assembly Line Balancing with Various Objectives," *Computers and Industrial Engineering*," No.30,(1996), pp. 397-409.

[7] Leu, Y.Y., L.A. Matheson, and L.P. Rees, "Assembly Line balancing Using genetic Algorithms with Heuristic-Generated Initial Populations and Multiple Evaluation Criteria," *Decision Science*, Vol.25,(1994), pp. 581-606.

[8] Michalewicz, Z., *Genetic Algorithms + Data Structures = Evolutionary Programs* (2nd ed.) Springer-Verlag, Berlin, (1994).

[9] Miltenburg, G.J., "Balancing U-Lines in a Multiple U-Line Facility," *European Journal of Operation Research*, Vol.109, (1998), pp. 1-23.

[10] Miltenburg, G.J. and J. Wijngaard, "The U-line Line Balancing Problem," *Management Science*, Vol.40, (1994), pp.1378-1388.

[11] Monden, Y., *Toyota Production System* (2nd Ed.), Industrial Engineering and Management Press, Institute of Industrial Engineers, Norcross, GA, (1993).

[12] Moodie, C.L. and H.H. Young, "A Heuristic Method of Assembly Line Balancing for Assumptions of Constant or Variable Work Element Times," *Journal of Industrial Engineering*, Vol. 16, No. 1, (1965), pp. 23-29.

[13] Urban, T.L., "Note. Optimal Balancing of U-Shaped Assembly Lines," *Management Science*, Vol. 44, No. 5, (1998), pp. 738-741.