병렬라인 검사공정의 작업배분을 위한 휴리스틱 알고리즘의 성능 개선

박 승 헌* · 이 석 환*
*인하대학교 산업공학과

Performance improvement of heuristic algorithm to assign job in parallel line inspection process

Seung-Hun Park* · Seog-Hwan Lee*
*Inha University Industrial Engineering

Abstract

In this paper, we raised the performance of heuristic algorithm to assign job to workers in parallel line inspection process without sequence. In previous research, we developed the heuristic algorithm. But the heuristic algorithm can't find optimal solution perfectly. In order to solve this problem, we proposed new method to make initial solution called FN(First Next) method and combined the new FN method and old FE method using previous heuristic algorithm. Experiments of assigning job are performed to evaluate performance of this FE+FN heuristic algorithm. The result shows that the FE+FN heuristic algorithm can find the optimal solution to assign job to workers evenly in many type of cases. Especially, in case there are optimal solutions, this heuristic algorithm can find the optimal solution perfectly.

Keyword : Inspection process, Assign job, Heuristic algorithm

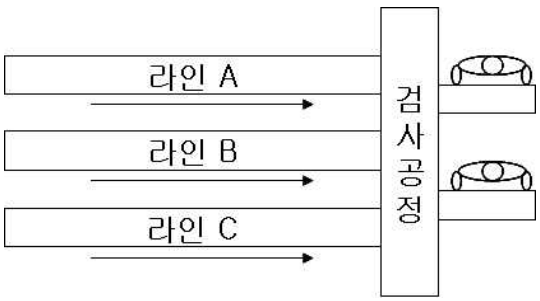
1. 서 론

많은 제조업체들은 고객의 다양한 제품수요를 충족시키기 위해 병렬라인 시스템을 도입하고 있다. 병렬라인 시스템은 다양한 품목을 대량으로 생산할 수 있다. 이러한 병렬라인 시스템의 검사공정은 전기·전자 분야 등에 도입되고 있으며 그 효율을 높이기 위해서 하나로 통합하여 운영한다. 병렬라인의 검사공정은 [그림 1]과 같다. 병렬라인 검사공정에는 복수의 작업자가 완성된 제품의 종류를 가리지 않고 검사를 수행한다. 이때 각 작업자들에게 어느 제품의 검사 작업들을 배분해야 하는 문제가 발생한다. 작업자들에 대한 작업의 배분은 공평하게 이루어져야 하는데 이와 관련된 연구는 많이 다루어지지 않고 있다. 그 중 병렬라인 검사공정에서 작업들 간에 순서가 없는 경우 작업자에게 작업량의 공평한 작업배분 문제를 다룬 기존의 연구가 있다[5].

본 연구의 목적은 기존 연구에서 개발한 휴리스틱 알고리즘에 새로운 방법을 추가하여 알고리즘의 성능을 개선하는 것이다. 검사공정의 작업배분을 위한 휴리스틱 알고리즘의 목적함수는 복수의 작업자에게 작업(시간)을 공평하게 배분하여 평활지수를 최소로 하는 것이다.

† 이 논문은 인하대학교 교내 연구비 지원에 의해 연구되었음.
† 교신저자: 이석환, 인천시 남구 용현동 253, 인하대학교 산업공학과 2북 670A
M · P: 010-5514-6242, E-mail: shleekun@inha.ac.kr
2012년 1월 20일 접수; 2012년 3월 13일 수정본 접수; 2012년 3월 14일 게재확정

기존 연구의 휴리스틱 알고리즘은 작업배분의 초기해를 생성하고 그 초기해를 이용하여 최적해를 찾는다. 작업배분의 초기해 생성과정은 다음과 같다. 먼저 작업들을 작업시간에 대한 내림차순으로 정렬한다. 다음으로 정렬된 작업의 맨 첫 작업 즉, 작업시간이 가장 큰 작업에 정렬된 작업의 맨 마지막 작업 즉, 작업시간이 가장 작은 작업부터 하나씩 배분한다. 이 과정은 작업배분의 결과가 일인 작업자의 이론적 배분시간인 산술평균시간에 근접할 때까지 반복된다(식 2 참조). 이 방법은 정렬된 작업의 첫 작업에 마지막 작업을 배분하므로 FE(First End)방법이라고 한다([그림 3] 참조). 기존의 FE방법으로는 최적해가 존재함에도 불구하고 최적해를 찾지 못하는 경우가 있었다. 따라서 본 연구에서는 이를 보완하기 위해 FN(First Next)방법을 제시한다. FN방법은 작업들을 작업시간의 내림차순으로 정렬한 후 정렬된 작업의 맨 첫 작업 즉, 작업시간이 가장 큰 작업에 그 다음으로 작업시간이 큰 작업들을 배분하면서 작업배분의 초기해를 생성한다([그림 4] 참조). 이 방법은 정렬된 작업의 첫 작업에 그 다음으로 큰 작업을 배분하므로 FN(First Next)방법이라고 한다. 본 연구에서 제시한 FN방법을 기존 연구의 FE방법과 혼용할 경우 FE방법의 단점을 보완할 수 있기 때문에 작업배분을 위한 휴리스틱 알고리즘의 성능은 크게 개선될 수 있다.



[그림 1] 검사공정

2. 검사공정의 작업배분 및 휴리스틱 알고리즘

본 연구는 검사공정에서 공평한 작업배분을 위한 휴리스틱 알고리즘의 성능을 개선하는 것이다. 2.1절에서는 검사공정에서 공평한 작업배분을 위한 휴리스틱 알고리즘의 목적함수에 대해 설명한다. 2.2절에서는 작업자에게 작업을 배분하는 방법인 FE방법과 FN방법에 대해 설명한다.

2.1 검사공정의 작업배분

병렬라인의 검사공정은 여러 라인에서 생산한 부품을 검사하는 공정으로 일반적인 형태는 <그림 1>과 같다. 검사공정에서 각 작업자에 대해 공평한 작업배분이 어려울 경우 다른 작업자보다 더 많은 시간의 작업을 하는 작업자가 발생한다. 이 경우 더 많은 시간의 작업을 할당받은 작업자는 육체적 피로의 증가와 정신적 스트레스로 인해 검사작업의 질이 저하되며 검사 시스템의 효율도 낮아진다. 따라서 공평한 작업배분은 검사작업의 질을 향상시키고 검사 시스템의 효율을 높이기 위해서 중요하다[2][10].

기존 연구에서는 검사공정에서 작업자에게 작업을 공평하게 배분하기 위해 휴리스틱 알고리즘을 개발하였다[5]. 본 연구의 목적은 기존 알고리즘의 성능을 개선하는 것이다. 알고리즘의 개선여부 즉, 작업이 공평하게 배분되었는지는 기존연구와 동일하게 평활지수(smooth index)로 측정한다. 평활지수는 수치들의 편차가 높고 낮음을 나타내는 척도로써 수치들의 편차가 적어지면 평활지수는 낮아진다.

식 (1)은 평활지수로 본 연구의 휴리스틱 알고리즘에 대한 목적함수는 평활지수 최소이다.

$$\sqrt{\sum_{i=1}^{n}(Max[T_i]-T_i)^2} \qquad \cdots\cdots\cdots(1)$$

작업자 : $i=1,2,...,n$

n : 작업자의 수

T_i : 각 작업자의 작업시간 합

2.2 FE(First End)방법과 FN(First Next) 방법에 의한 초기해 생성

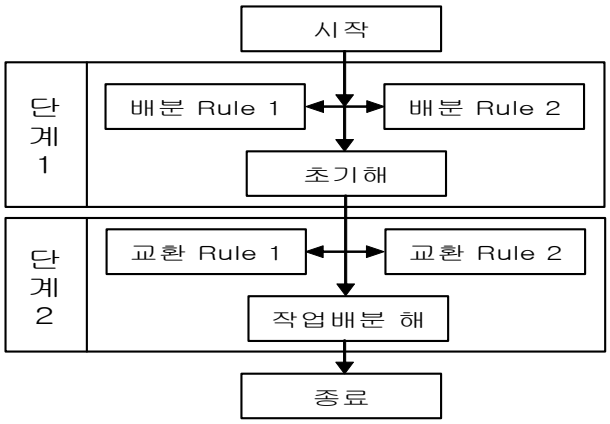
기존 연구의 휴리스틱 알고리즘 전개과정은 [그림 2]와 같다. 단계 1에서는 작업배분의 초기해를 생성하고 단계 2에서는 초기해를 이용하여 작업들을 교환하면서 해를 찾는다. 알고리즘에서 작업배분의 기준으로 사용하는 이론적 배분시간 T^*는 식 (2)와 같다.

$$\text{이론적 배분시간 } T^*=\frac{\sum_{i=1}^{k}J_i}{N} \qquad \cdots\cdots\cdots(2)$$

총 작업시간 $\sum_{i=1}^{k}J_i$

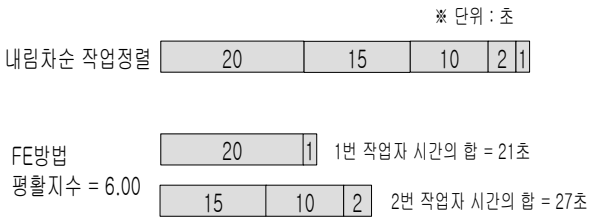
요소작업시간 $J_i,\ i=1,2,...,k$

작업자 $N_i,\ i=1,2,...,n$



[그림 2] FE방법(기존연구)의 작업배분과정

기존 연구의 단계 1에서 작업들을 배분하는데 사용한 초기해 생성 방법은 다음과 같다. 먼저 작업들을 작업시간의 내림차순으로 정렬한다. 다음으로 정렬된 작업의 맨 첫 작업 즉, 작업시간이 가장 큰 작업에 정렬된 작업의 맨 마지막 작업 즉, 작업시간이 가장 작은 작업부터 하나씩 배분한다. 이 과정은 작업배분의 결과가 일인 작업자의 이론적 배분시간에 근접할 때까지 반복된다. 이 방법은 첫 작업에 마지막 작업을 배분하기 때문에 FE(First End)방법라고 한다. FE 방법은 일인 작업자에게 작업시간이 가장 큰 작업과 가장 작은 작업을 균형 있게 배분할 수 있기 때문에 평활지수를 최소로 할 수 있다. [그림 3]은 FE방법을 사용하여 초기해를 생성하는 과정으로 5개의 작업을 2명의 작업자에게 할당하는 예이다. 총 작업시간은 48초이며 작업자가 두 명 이므로 일인 작업자의 이론적 배분시간은 24초이다.

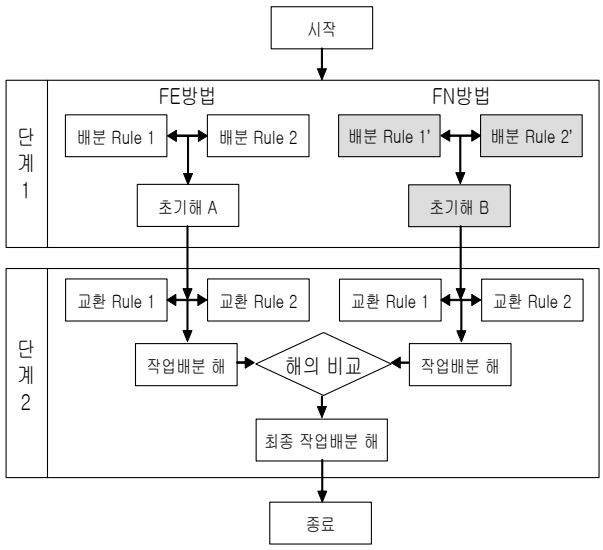


[그림 3] FE방법의 초기해 생성

FE방법은 작업을 균형 있게 배분할 수 있어서 평활지수를 최소로 할 수 있다. 그러나 일인 작업자에게 작업시간이 큰 작업들로만 배분하여도 평활지수는 최소가 될 수 있다. 이 방법을 설명하면 다음과 같다. 먼저 작업들을 작업시간의 내림차순으로 정렬한다. 다음으로 정렬된 작업의 맨 첫 작업 즉, 작업시간이 가장 큰 작업에 그 다음으로 작업시간이 큰 작업부터 하나씩 배분한다. 이 방법은 첫 작업에 그 다음 작업을 배분하기 때문에 FN(First Next)방법이라고 한다. [그림 4]는 그 예를 보여주는 것으로 5개의 작업을 2명의 작업자에게 할당하는 예이다. 총 작업시간은 60초이며 작업자가 두 명 이므로 일인 작업자의 이론적 배분시간은 30초이다. [그림 4]의 예에서 FE방법을 사용하여 작업을 배분하면 평활지수가 14가 된다. 그러나 FN방법을 사용할 경우 작업시간이 가장 큰 15초와 그 다음으로 작업시간이 큰 14초 두 개가 한 작업자에 배분되므로 평활지수는 2가 된다. [그림 4]의 예와 같이 초기해 생성과정인 단계 1에서 FE와 FN방법을 혼용할 경우 FE방법에서 찾을 수 없는 좋은 해를 찾을 수 있기 때문에 알고리즘의 성능은 개선될 수 있다.

[그림 4] FE와 FN방법의 초기해 생성 비교

[그림 5]는 본 연구에서 새롭게 제시한 FN방법을 추가하여 FE+FN방법을 혼용하는 알고리즘 전개과정이다. 음영 부분은 본 연구에서 새롭게 제시한 FN방법이다.



<그림 5> FE+FN 방법 혼용 작업배분과정

본 연구에서 추가한 FN방법과 기존 연구의 FE방법과의 차이는 <그림 6>과 같다.

FE방법	FN방법
배분 Rule 1 : $k-1$개의 작업 중 작은 순서대로 하나씩 넣어 T^*를 넘지 않으면 가장 작은 작업을 배분한다. (단, T^*와 동일한 작업이 있으면 그 작업을 배분한다)	배분 Rule 1' : $k-1$개의 작업 중 큰 순서대로 하나씩 넣어 T^*를 넘지 않으면 가장 큰 작업을 배분한다. (단, T^*와 동일한 작업이 있으면 그 작업을 배분한다)
배분 Rule 2 : $k-1$개의 작업 중 작은 순서대로 하나씩 넣어 T^*를 넘는 작업이 있으면, 첫 번째로 T^*를 넘는 작업의 초과시간과 T^*를 넘기 직전 작업의 미달시간의 차이를 계산하여 차이가 적은 작업을 배분한다.	배분 Rule 2' : $k-1$개의 작업 중 큰 순서대로 하나씩 넣어 T^*를 넘는 작업이 있으면, 첫 번째로 T^*를 넘는 작업의 초과시간과 T^*를 넘기 직전 작업의 미달시간의 차이를 계산하여 차이가 적은 작업을 배분한다.

[그림 6] FE와 FN방법의 작업배분 비교

단계 1에 추가한 FN방법을 상세하게 설명하면 다음과 같다.

단계 1 (FN방법)

(1) 총 작업시간을 작업자의 수로 나누어 일인 작업자의 이론적 평균시간 T^*를 구한다.
(2) 작업들을 작업시간의 내림차순으로 정렬한다.
(3) 한 명의 작업자에게 가장 작업시간이 긴 작업 $\max(J_i)$을 배분한다.
(4) $k-1$개의 작업 중 큰 순서대로 한 번씩 $\max(J_i)$에 붙여서 총 작업시간이 T^*를 넘는 작업이 있으면 (5)로 이동하고, 그렇지 않으면 가장 큰 작업을 배분하고 나머지 $k-2$개로 (4)를 재 적용한다. (단, T^*와 동일한 작업이 있으면 그 작업을 배분한다)
(5) T^*를 넘는 작업이 있으면, 첫 번째로 T^*를 넘는 작업의 초과시간과 T^*를 넘기 직전 작업의 미달시간의 차이를 계산하여 차이가 적은 작업을 배분하고 작업자 한 명에 대한 작업배분을 종료한다. 다음 작업자로의 배분은 (3)으로 이동하여 같은 순서를 밟는다.

단계 2는 단계 1에서 생성한 작업배분의 초기해를 이용하여 작업들을 교환하면서 해를 찾는 과정이다. 단계 2에서는 최적해를 발견하기 위해 작업교환을 사용한다. 작업교환은 교환 Rule 1과 교환 Rule 2로 구성한다. 교환 Rule 1, 2는 두 작업자의 작업을 교환하는 것으로 각 작업자의 총 작업시간을 균등하게 하는 것을 목적으로 한다. 단계 2의 자세한 과정은 기존 연구와 동일하므로 기존 연구를 참조하기 바란다[5]. [그림 7]은 본 연구의 FE+FN방법 혼용 알고리즘을 C언어로 나타낸 것이다.

작업배분 1단계 FE방법

```
for (i=1; i <= jobNum; i++)
{
        for (j=jobNum; j > i; j--)
        {
                workerSum = workerTime[i] +
                workerTime[j]
                if all ( T* <= workerSum)
                          //작업배분 1단계 Rule 1
                {
                        작업자에게 맨 끝작업 배분
                }
                else        //작업배분 1단계 Rule 2
                {
                if abs( T* - workerTime[i]
                        + workerTime[j])
                  < abs( T* - workerTime[i]
                        + workerTime[j+1])
                   j 작업 배분
                else
                   j+1 작업 배분
                }
        }
}
```

작업배분 1단계 FN방법

```
for (i=1; i <= jobNum; i++)
{
        for (j=i+1; j <= jobNum; j++)
        {
                workerSum = workerTime[i] +
                workerTime[j]
                if all ( T* <= workerSum)
                          //작업배분 1단계 Rule 1'
                {
                        작업자에게 다음 작업 배분
                }
                else        //작업배분 1단계 Rule 2'
                {
                if abs( T* - workerTime[i]
                        + workerTime[j])
                  < abs( T* - workerTime[i]
                        + workerTime[j+1])
                   j 작업 배분
                else
                   j+1 작업 배분
                }
        }
}
```

```
작업배분 2단계 Rule 1

Tmax 오름차순정렬
for (i=1; i <= TmaxJobNum; i++)
if Tmin의 작은 작업 < Tmax(i) then 작업교환
        if (Rule 1의 Tmax - Tmin)
          < (Rule 2의 Tmax - Tmin)
        Rule 1 작업배분 적용
else
        Rule 2 작업배분 적용
```

```
작업배분 2단계 Rule 2

Tmax 오름차순정렬
Tmax = Tmax - Tmax(1)
Tmin = Tmin + Tmax(1)
```

```
메인 프로그램

작업을 내림차순으로 정렬
작업배분 1단계 실행 (초기해 A, B생성)
do
{
        작업배분 2단계 Rule 2 실행
        작업배분 2단계 Rule 1 실행
} while (이전 평활지수 > 이후 평활지수)

if (초기해 A의 평활지수 < 초기해 B의 평활지수)
        초기해 A 선택
else
        초기해 B 선택
```

[그림 7] FE+FN 방법 혼용 휴리스틱 알고리즘 상세

3. 적용결과 및 분석

본 연구의 휴리스틱 알고리즘은 C++언어로 구현하였다. FE+FN 방법 혼용 알고리즘이 최적배분의 해를 어느 정도 찾을 수 있는지 평가하기 위해 두 가지 실험을 수행하였다.

첫 번째 실험은 기존 연구와 동일한 데이터를 사용하여 FE+FN 혼용 알고리즘이 최적배분의 해를 어느 정도 찾을 수 있는지 평가하는 것이다. 이 경우는 3.1절에서 수행하였다. 두 번째 실험은 새로운 데이터로 작업개수를 증가시켜서 FE+FN 혼용 알고리즘이 평활지수 0에 근사한 작업배분을 찾을 수 있는지 평가하는 것이다. 이 경우는 3.2절에서 수행하였다. 마지막으로 3.3절에서는 작업시간의 분산이 평활지수에 어떤 영향을 주는지 평가하였다.

3.1 기존 연구의 데이터 적용결과

첫번째 실험은 기존 연구의 데이터를 사용하여 FE+FN 혼용 알고리즘이 기존 알고리즘에 비해서 최적배분의 해를 어느 정도 찾을 수 있는지 평가한다. <표 1>은 기존 알고리즘과 FE+FN 혼용 알고리즘에 따른 작업배분의 평활지수를 비교한 것이다. 기존 알고리즘은 FE방법만을 사용한 것이고 FE+FN 혼용 알고리즘은 FE와 FN방법을 혼용한 것이다. 음영부분은 평활지수가 가장 좋은 알고리즘을 표시한 것이다. <표 1>의 결과와 같이 총 26회의 실험에서 FE+FN 혼용 알고리즘은 기존의 알고리즘으로 찾을 수 없는 좋은 해를 2개 더 찾을 수 있었다. 한 개의 최적해가 존재하는 경우 작업개수 12개를 3명의 작업자에게 할당하는 예에서 FE+FN 혼용 알고리즘은 평활지수를 0으로 하는 최적해를 발견하였다. 최적해가 없는 경우 작업개수 15개를 4명의 작업자에게 할당하는 예에서 FE+FN 혼용 알고리즘은 기존 알고리즘의 평활지수를 6.16에서 1.41로 77.1% 향상시켰다. <표 2>와 <표 3>는 <표 1>에서 FE+FN 혼용 알고리즘이 기존 FE방법보다 더 나은 해를 찾아낸 작업배분 결과를 나타낸 것이다. <표 2>는 최적해가 한 개 있는 경우 12개 작업을 3명의 작업자에게 배분한 결과를 나타낸 것이다. <표 3>은 최적해가 없는 경우 15개 작업을 4명의 작업자에게 할당한 결과를 나타낸 것이다. 이 결과는 FE방법에 FN방법을 혼용할 경우 더욱 좋은 해를 발견할 수 있다는 것을 보여준다.

<표 1> FE방법과 FE+FN방법 혼용 알고리즘의 평활지수 비교(기존 연구의 데이터)

	작업 개수	작업자 수	FE	FE+FN
한 개 최적해	9	3	0.00	0.00
		4	0.00	0.00
		6	0.00	0.00
	12	3	2.24	0.00
		4	0.00	0.00
		6	0.00	0.00
	15	3	2.24	2.24
		4	2.45	2.45
		6	0.00	0.00
여러 개 최적 해	9	3	0.00	0.00
		4	0.00	0.00
		6	-	-
	12	3	0.00	0.00
		4	0.00	0.00
		6	0.00	0.00
	15	3	0.00	0.00
		4	0.00	0.00
		6	0.00	0.00
최적 해가 없는 경우	9	3	1.41	1.41
		4	2.00	2.00
		6	4.00	4.00
	12	3	1.41	1.41
		4	4.90	4.90
		6	4.90	4.90
	15	3	2.00	2.00
		4	6.16	1.41
		6	2.83	2.83

<표 2> 작업개수 12개에서 FE방법과 FE+FN방법의 작업배분 결과(한 개의 최적해가 존재하는 경우)

(단위 : 초)

작업번호	1	2	3	4	5	6	7	8	9	10	11	12	합
작업시간	12	8	7	13	6	11	9	14	4	14	10	12	120

알고리즘 / 작업자 수	FE	FE+ FN
3명	8 - 9 - 5 - 3 - 7 => 1번 작업자 작업시간의 합 = 40 2 - 11 - 6 - 1 => 2번 작업자 작업시간의 합 = 41 10 - 12 - 4 => 3번 작업자 작업시간의 합 = 39 평활지수 = 2.24	8 - 10 - 12 => 1번 작업자 작업시간의 합 = 40 4 - 1 - 6 - 9 => 2번 작업자 작업시간의 합 = 40 11 - 7 - 2 - 3 - 5 => 3번 작업자 작업시간의 합 = 40 평활지수 = 0.00

※ 작업배분 결과는 작업번호를 표시한 것임

<표 3> 작업개수 15개에서 FE방법과 FE+FN방법의 작업배분 결과(최적해가 없는 경우)

(단위 : 초)

작업번호	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	합
작업시간	21	9	11	12	8	7	14	13	6	18	8	29	10	21	23	166

알고리즘 / 작업자 수	FE	FE+ FN
4명	12 - 9 - 10 => 1번 작업자 작업시간의 합 = 53 11 - 1 - 15 => 2번 작업자 작업시간의 합 = 52 5 - 2 - 13 - 14 - 6 => 3번 작업자 작업시간의 합 = 55 7 - 3 - 4 - 8 => 4번 작업자 작업시간의 합 = 50 평활지수 = 6.16	12 - 15 => 1번 작업자 작업시간의 합 = 52 14 - 1 - 3 => 2번 작업자 작업시간의 합 = 53 10 - 7 - 8 - 11 => 3번 작업자 작업시간의 합 = 53 4 - 13 - 2 - 5 - 6 - 9 => 4번 작업자 작업시간의 합 = 52 평활지수 = 1.41

※ 작업배분 결과는 작업번호를 표시한 것임

3.2 작업의 개수를 증가시킨 실험결과

3.1절의 실험결과와 같이 FE+FN 혼용 알고리즘은 기존의 알고리즘에서 발견할 수 없는 해를 발견할 수 있었다. 두 번째 실험은 새로운 데이터를 이용한 실험이다. 이 실험은 작업의 개수가 많아질 경우 FE+FN 혼용 휴리스틱 알고리즘이 기존 알고리즘에 비해서 최적배분의 해를 어느 정도 찾을 수 있는지 평가한다. 실험 데이터는 최적해가 한 개 있는 경우, 여러 개 있는 경우, 최적 해가 없는 경우의 세 가지로 하고 작업개수는 18, 21, 24에 작업자 수는 3, 4, 6으로 하였다.

<표 4>는 기존 알고리즘과 FE+FN 혼용 알고리즘에 따른 작업배분의 평활지수를 비교한 표이다. 음영부분은 평활지수가 가장 좋은 알고리즘을 표시한 것이다.

<표 4>의 결과와 같이 총 27회의 실험에서 FE+FN 혼용 알고리즘은 기존의 알고리즘으로 찾을 수 없는 좋은 해를 9개 더 찾을 수 있었다. 한 개의 최적해가 존재하는 경우 18개 작업을 4명의 작업자에게 할당하는 예에서 FE+FN 혼용 알고리즘은 최적해를 발견하였다.

<표 4> FE방법과 FE+FN방법의 평활지수 비교
(작업의 개수를 증가시킨 데이터)

	작업 개수	작업자 수	FE	FE+ FN
한 개 최적 해	18	3	0.00	0.00
		4	2.45	0.00
		6	0.00	0.00
	21	3	0.00	0.00
		4	0.00	0.00
		6	0.00	0.00
	24	3	0.00	0.00
		4	0.00	0.00
		6	0.00	0.00
여러 개 최적 해	18	3	4.47	0.00
		4	7.35	0.00
		6	0.00	0.00
	21	3	4.47	0.00
		4	0.00	0.00
		6	0.00	0.00
	24	3	0.00	0.00
		4	0.00	0.00
		6	0.00	0.00
최적 해가 없는 경우	18	3	7.62	3.16
		4	2.24	1.73
		6	1.00	1.00
	21	3	1.00	1.00
		4	1.00	1.00
		6	3.61	1.00
	24	3	3.16	1.00
		4	1.73	1.73
		6	6.86	6.40

여러 개의 최적해가 존재하는 경우 18개 작업을 3, 4명의 작업자에게 할당하는 예에서 FE+FN 혼용 알고리즘은 최적해를 발견하였다. 21개 작업을 3명의 작업자에게 할당하는 예에서 FE+FN 혼용 알고리즘은 최적해를 발견하였다. 최적해가 없는 경우 18개 작업을 3, 4명의 작업자에게 할당하는 예에서 FE+FN 혼용 알고리즘은 기존 알고리즘의 평활지수를 7.62에서 3.16, 2.24에서 1.73으로 각각 58.5%, 22.8% 향상시켰다. 21개 작업을 6명의 작업자에게 할당하는 예에서 FE+FN 혼용 알고리즘은 기존 알고리즘의 평활지수를 3.61에서 1.00으로 72.3% 향상시켰다. 작업개수 24개를 3, 6명의 작업자에게 할당하는 예에서 FE+FN 혼용 알고리즘은 기존 알고리즘의 평활지수를 3.16에서 1.00, 6.86에서 6.40으로 각각 68.4%, 6.7% 향상시켰다. 이 결과는 3.1절의 실험(기존 연구의 데이터를 사용)과 동일하게 FE과 FN방법을 혼용할 경우 더욱 좋은 해를 발견할 수 있다는 것을 보여주는 것이다.

<표 5>~<표 10>는 <표 4>에서 기존 알고리즘과 FE+FN 혼용 알고리즘이 찾아낸 작업배분 결과를 나타낸 것이다.

<표 5> 작업개수 18개에서 FE 방법과 FE+FN방법의 작업배분 결과(한 개의 최적해가 존재하는 경우)

(단위 : 초)

작업번호	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	합
작업시간	21	19	17	15	16	14	13	12	11	10	9	8	7	6	5	4	3	2	192

작업자 수 \ 알고리즘	FE	FE+ FN
4명	17 - 16 - 2 - 1 => 1번 작업자 작업시간의 합 = 47 3 - 15 - 14 - 13 - 7 => 2번 작업자 작업시간의 합 = 48 5 - 12 - 11 - 4 => 3번 작업자 작업시간의 합 = 48 10 - 9 - 8 - 6 - 18 => 4번 작업자 작업시간의 합 = 49 평활지수 = 2.45	1 - 2 - 12 => 1번 작업자 작업시간의 합 = 48 3 - 5 - 4 => 2번 작업자 작업시간의 합 = 48 6 - 7 - 8 - 11 => 3번 작업자 작업시간의 합 = 48 9 - 10 - 13 - 14 - 15 - 16 - 17 - 18 => 4번 작업자 작업시간의 합 = 48 평활지수 = 0.00

※ 작업배분 결과는 작업번호를 표시한 것임

<표 6> 작업개수 18개에서 FE방법과 FE+FN방법의 작업배분 결과(여러 개의 최적해가 존재하는 경우)

(단위 : 초)

작업번호	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	합
작업시간	21	15	9	15	7	20	18	10	5	20	14	12	16	8	21	19	10	6	246

작업자 수 \ 알고리즘	FE	FE+ FN
3명	1 - 9 - 18 - 5 - 14 - 3 - 17 - 13 => 1번 작업자 작업시간의 합 = 82 8 - 12 - 11 - 4 - 2 - 7 => 2번 작업자 작업시간의 합 = 84 15 - 16 - 10 - 6 => 3번 작업자 작업시간의 합 = 80 평활지수 = 4.47	1 - 15 - 10 - 6 => 1번 작업자 작업시간의 합 = 82 16 - 7 - 13 - 2 - 11 => 2번 작업자 작업시간의 합 = 82 4 - 12 - 8 - 17 - 3 - 14 - 5 - 18 - 9 => 3번 작업자 작업시간의 합 = 82 평활지수 = 0.00

작업번호	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	합
작업시간	21	17	9	15	7	20	18	10	5	20	14	12	16	8	21	19	10	6	248

4명	1 - 9 - 18 - 5 - 14 - 4 => 1번 작업자 작업시간의 합 = 62 15 - 3 - 17 - 8 - 12 => 2번 작업자 작업시간의 합 = 62 11 - 13 - 2 - 7 => 3번 작업자 작업시간의 합 = 65 10 - 16 - 6 => 4번 작업자 작업시간의 합 = 59 평활지수 = 7.35	1 - 15 - 6 => 1번 작업자 작업시간의 합 = 62 10 - 16 - 7 - 9 => 2번 작업자 작업시간의 합 = 62 2 - 13 - 4 - 11 => 3번 작업자 작업시간의 합 = 62 12 - 8 - 17 - 3 - 14 - 5 - 18 => 4번 작업자 작업시간의 합 = 62 평활지수 = 0.00

※ 작업배분 결과는 작업번호를 표시한 것임

<표 7> 작업개수 21개에서 FE방법과 FE+FN방법의 작업배분 결과(여러 개의 최적해가 존재하는 경우)

(단위 : 초)

작업번호	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	합
작업시간	30	9	13	10	11	4	24	23	9	14	18	22	31	10	11	11	12	5	25	24	7	323

작업자 수 \ 알고리즘	FE	FE+ FN
6명	13 - 6 - 18 - 21 - 9 - 2 - 4 - 14 - 8 => 1번 작업자 작업시간의 합 = 108 1 - 16 - 15 - 5 - 17 - 3 - 12 => 2번 작업자 작업시간의 합 = 110 7 - 19 - 20 - 11 - 10 => 3번 작업자 작업시간의 합 = 106 평활지수 = 4.47	13 - 1 - 7 - 12 => 1번 작업자 작업시간의 합 = 108 19 - 20 - 8 - 11 - 10 - 6 => 2번 작업자 작업시간의 합 = 108 3 - 17 - 5 - 15 - 16 - 14 - 4 - 2 - 9 - 21 - 18 => 3번 작업자 작업시간의 합 = 108 평활지수 = 0.00

※ 작업배분 결과는 작업번호를 표시한 것임

<표 8> 작업개수 18개에서 FE방법과 FE+FN방법의 작업배분 결과(최적해가 없는 경우)

(단위 : 초)

작업번호	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	합
작업시간	19	13	11	15	7	20	18	10	5	20	14	12	16	8	21	19	11	6	245

작업자 수 \ 알고리즘	FE	FE+ FN
3명	15 - 9 - 18 - 5 - 14 - 8 - 17 - 11 => 1번 작업자 작업시간의 합 = 82 3 - 12 - 2 - 4 - 13 - 7 => 2번 작업자 작업시간의 합 = 85 10 - 16 - 1 - 6 => 3번 작업자 작업시간의 합 = 78 평활지수 = 7.62	15 - 10 - 6 - 1 => 1번 작업자 작업시간의 합 = 80 16 - 7 - 13 - 4 - 11 => 2번 작업자 작업시간의 합 = 82 2 - 12 - 3 - 17 - 8 - 14 - 5 - 18 - 9 => 3번 작업자 작업시간의 합 = 83 평활지수 = 3.16
4명	9 - 18 - 5 - 14 - 4 - 1 => 1번 작업자 작업시간의 합 = 60 8 - 17 - 3 - 12 - 7 => 2번 작업자 작업시간의 합 = 62 2 - 11 - 13 - 16 => 3번 작업자 작업시간의 합 = 62 15 - 6 - 10 => 4번 작업자 작업시간의 합 = 61 평활지수 = 2.24	15 - 10 - 6 => 1번 작업자 작업시간의 합 = 61 9 - 7 - 1 - 16 => 2번 작업자 작업시간의 합 = 61 18 - 8 - 11 - 4 - 13 => 3번 작업자 작업시간의 합 = 61 5 - 14 - 2 - 17 - 3 - 12 => 4번 작업자 작업시간의 합 = 62 평활지수 = 1.73

※ 작업배분 결과는 작업번호를 표시한 것임

<표 9> 작업개수 21개에서 FE방법과 FE+FN방법의 작업배분 결과(최적해가 없는 경우)

(단위 : 초)

작업번호	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	합
작업시간	30	9	13	10	11	4	24	23	9	14	18	22	31	10	11	11	12	5	25	24	7	323

작업자 수 \ 알고리즘	FE	FE+ FN
6명	13 - 8 => 1번 작업자 작업시간의 합 = 54 1 - 20 => 2번 작업자 작업시간의 합 = 54 19 - 6 - 18 - 11 => 3번 작업자 작업시간의 합 = 52 7 - 21 - 9 - 10 => 4번 작업자 작업시간의 합 = 54 12 - 2 - 4 - 3 => 5번 작업자 작업시간의 합 = 54 17 - 5 - 15 - 16 - 14 => 6번 작업자 작업시간의 합 = 55 평활지수 = 3.61	13 - 8 => 1번 작업자 작업시간의 합 = 54 1 - 20 => 2번 작업자 작업시간의 합 = 54 19 - 7 - 18 => 3번 작업자 작업시간의 합 = 54 12 - 11 - 10 => 4번 작업자 작업시간의 합 = 54 3 - 17 - 5 - 15 - 21 => 5번 작업자 작업시간의 합 = 54 16 - 14 - 4 - 2 - 9 - 6 => 6번 작업자 작업시간의 합 = 53 평활지수 = 1.00

※ 작업배분 결과는 작업번호를 표시한 것임

<표 10> 작업개수 24개에서 FE방법과 FE+FN방법의 작업배분 결과(최적해가 없는 경우)

(단위 : 초)

작업번호	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	합
작업시간	21	9	11	12	8	16	14	13	6	18	8	29	10	21	23	22	10	12	13	9	8	15	14	7	329

작업자 수 \ 알고리즘	FE	FE+ FN
3명	12 - 9 - 24 - 5 - 11 - 21 - 20 - 2 - 17 - 6 => 1번 작업자 작업시간의 합 = 110 13 - 3 - 4 - 18 - 19 - 8 - 7 - 15 => 2번 작업자 작업시간의 합 = 108 10 - 23 - 22 - 14 - 1 - 16 => 3번 작업자 작업시간의 합 = 111 평활지수 = 3.16	12 - 15 - 16 - 1 - 22 => 1번 작업자 작업시간의 합 = 110 14 - 10 - 6 - 7 - 23 - 8 - 19 => 2번 작업자 작업시간의 합 = 109 18 - 4 - 3 - 13 - 17 - 2 - 20 - 21 - 11 - 5 - 24 - 9 => 3번 작업자 작업시간의 합 = 110 평활지수 = 1.00
6명	12 - 9 - 14 => 1번 작업자 작업시간의 합 = 56 15 - 24 - 5 - 10 => 2번 작업자 작업시간의 합 = 56 11 - 21 - 23 - 16 => 3번 작업자 작업시간의 합 = 52 6 - 20 - 2 - 17 - 3 => 4번 작업자 작업시간의 합 = 55 4 - 18 - 7 - 22 => 5번 작업자 작업시간의 합 = 53 13 - 19 - 8 - 1 => 6번 작업자 작업시간의 합 = 57 평활지수 = 6.86	12 - 15 => 1번 작업자 작업시간의 합 = 52 16 - 1 - 4 => 2번 작업자 작업시간의 합 = 55 14 - 10 - 6 => 3번 작업자 작업시간의 합 = 55 22 - 7 - 23 - 18 => 4번 작업자 작업시간의 합 = 55 8 - 19 - 3 - 13 - 5 => 5번 작업자 작업시간의 합 = 55 17 - 2 - 20 - 21 - 11 - 24 - 9 => 6번 작업자 작업시간의 합 = 57 평활지수 = 6.40

※ 작업배분 결과는 작업번호를 표시한 것임

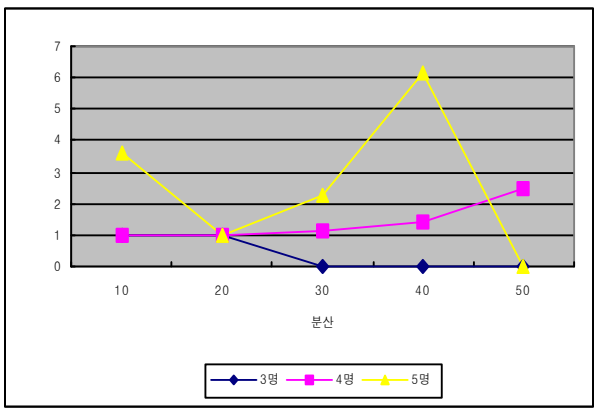
3.3 작업시간 분산과 평활지수 간의 관계

다음은 작업시간의 분산과 평활지수간의 관계를 확인하는 실험이다. 작업의 개수는 18, 21, 24개로 하였고 각 경우의 분산은 10, 20, 30, 40, 50으로 설정하였다. 작업자의 수는 앞의 실험과 동일하게 3, 4, 6명으로 하였다.

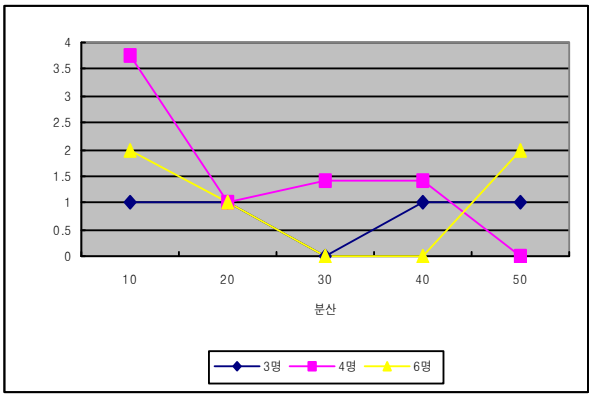
실험결과는 <표 11>과 같이 나타났다. 작업시간의 분산과 평활지수 간의 관계를 확인하기 위해 <표 11>을 [그림 8], [그림 9], [그림 10]의 그래프로 표현하였다. 기존 연구에서는 작업시간의 분산이 커질수록 평활지수가 증가하였다. 그러나 본 연구에서는 작업시간의 분산과 평활지수 간에 특별한 경향이 발견되지 않았다. 본 연구에서는 작업의 개수를 증가시켜서 실험을 하였다. 작업의 개수가 증가하면 일인 작업자에게 배분되는 작업이 증가한다. 따라서 각 작업자들에게 배분되는 작업시간 합의 차이가 감소하기 때문에 작업시간의 분산이 커지더라도 평활지수에 미치는 영향은 적다고 판단된다.

<표 11> 작업시간의 분산에 따른 FE+FN 방법 혼용 휴리스틱 알고리즘의 평활지수

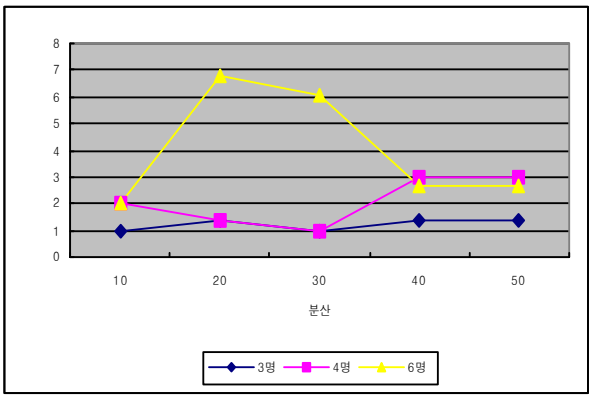
작업 / 작업자 / 분산	18			21			24		
	3	4	6	3	4	6	3	4	6
10	1.00	1.00	3.61	1.00	3.74	2.00	1.00	2.00	2.00
20	1.00	1.00	1.00	1.00	1.00	1.00	1.41	1.41	6.78
30	0.00	1.11	2.24	0.00	1.41	0.00	1.00	1.00	6.08
40	0.00	1.41	6.16	1.00	1.41	0.00	1.41	3.00	2.65
50	0.00	2.45	0.00	1.00	0.00	2.00	1.41	3.00	2.66



[그림 8] 작업시간 분산에 따른 FE+FN 방법 혼용 휴리스틱 알고리즘의 평활지수 변화 - 18개 작업



[그림 9] 작업시간 분산에 따른 FE+FN 방법 혼용 휴리스틱 알고리즘의 평활지수 변화 - 21개 작업



[그림 10] 작업시간 분산에 따른 FE+FN 방법 혼용 휴리스틱 알고리즘의 평활지수 변화 - 24개 작업

4. 결 론

본 연구에서는 병렬라인 검사공정에서 복수의 작업자에게 작업을 배분하는 기존 휴리스틱 알고리즘(FE방법)의 성능을 개선하였다. 기존 알고리즘은 FE방법만으로 초기해를 생성하였으나 본 연구에서는 새로운 FN 방법을 고안하여 FE와 FN 방법을 혼용함으로써 알고리즘의 성능을 개선하였다. FE+FN 방법 혼용 휴리스틱 알고리즘의 성능은 실험을 통하여 확인하였으며 그 효과 및 특성은 다음과 같다.

먼저 FE+FN 혼용 알고리즘의 개선효과는 다음과 같이 세 종류의 실험을 통해 확인되었다. 첫째, 평활지수 0인 최적배분의 해가 한 개 이상인 경우 이 혼용 알고리즘은 최적배분의 해를 대부분 찾을 수 있어 우수한 성능을 확인하였다(<표 4> 참조). 둘째, 평활지수 0인 최적배분의 해가 없는 경우 FE+FN 혼용 알고리즘은 FE방법 알고리즘의 평활지수를 평균 43.42% 개선한 해를 찾을 수 있었다(<표 4> 참조).

다음으로 FE+FN 혼용 알고리즘의 특성은 다음과 같다. 첫째, 평활지수 0인 최적배분의 해가 존재하는 경우 작업의개수가 많아질수록 FE+FN 혼용 알고리즘은 최적배분 해를 발견하는 확률이 높아진다. 둘째, 작업의 개수가 많아질 경우 작업시간의 분산이 평활지수에 주는 영향은 감소한다.

본 연구의 FE+FN 혼용 알고리즘은 FE방법에서 찾을 수 없는 좋은 해를 찾을 수 있기 때문에 휴리스틱 알고리즘의 성능을 개선할 수 있었다. 이 알고리즘을 사용하여 작업배분을 할 경우 모든 작업자에게 공평한 작업배분이 가능하기 때문에 불공평한 작업배분으로 발생하는 문제를 해결할 수 있다.

5. 참 고 문 헌

[1] 김여근, 김재윤, 김영호, "공진화 알고리듬을 이용한 혼합모델 조립라인에서의 작업할당과 투입순서결정", 한국경영과학회, 1999년 학술대회논문집 제1권, 1999, 34-35

[2] 문성민, 권근석, 최경현, "작업시간과 육체적인 작업부하를 고려한 혼합모델 조립공정의 라인밸런싱", IE interface, 제 17권 3호, 2004, 282-293

[3] 이명호, 유지수, "생산관리.", 박영사 (1999)

[4] 이상범, "현대생산운영관리.", 명경사 (2007)

[5] 이석환, 박승헌, "검사공정의 작업배분을 위한 휴리스틱 알고리즘 개발", 대한안전경영학회지, 제 10권 제 3호, 2008, 253-265

[6] Chaffin, D., "Ergonomics guide for the assessment of human static strength", American industrial hygiene association journal, 1975, 36, 505-510

[7] G.A. kohring, "Dynamic load balancing for parallelized particle simulations on MIMD computers", Parallel computing, Vol 21, 1995, 683-693

[8] Grandjean, E,. "Fitting the task to the man : Ergonomic approach", Taylor and Francis, London, 1985

[9] K. Devine, J. Flaherty, S. Wheat, and A. Maccabe, "A massively parallel adaptive finite element metho with dynamic load balancing", Supercomputing, 1993, 2-11

[10] Matanachai, S. and Yano, C. A., "Balancing mixed-model assembly lines to reduce work overload", IIE transactions, 2001, 29-42

저 자 소 개

박 승 헌

인하대학교 금속공학과에서 공학사, 일본 Keio대학 관리공학과에서 공학석사 및 공학박사 학위를 취득 하였다. 현재 인하대학교 산업공학과 교수로 재직 중이다. 주요 관심 분야는 FMS와 각종 생산시스템의 설계 및 운영, 인터넷 마케팅과 데이터 마이닝 등이다.

주소: 인천광역시 남구 용현동 253, 인하대학교 산업공학과

이 석 환

인하대학교 산업공학과에서 공학사, 공학석사 및 공학박사 학위를 취득하였다. 주요 관심 분야는 데이터 마이닝이다.

주소: 인천광역시 남구 용현동 253, 인하대학교 산업공학과