SDME를 이용한 n/m Job-Shop 스케쥴링에 대한 발견적 방법 - A Heuristic Method for n/m Job-Shop Scheduling Using the SDME -

김 제 홍*
Kim, Je-Hong
조 남 호**
Cho, Nam-Ho

Abstract

To make a scheduling for minimizing the tardiness of delivery, we present an efficient heuristic algorithm for job-shop scheduling problems with due dates. The SDME(Slack Degree Modified Exchange) algorithm, which is the proposed heuristic algorithm, carries out the operation in two phases. In the phase 1, a dispatching rule is employed to generate an active or non-delay initial schedule. In the phase 2, tasks selected from a predetermined set of promising target operations in the initial schedule are tested to ascertain whether, by left-shifting their start times and rearranging some subset of the remaining operations, one can reduce the total tardiness. In the numerical example, the results indicate that the proposed SDME algorithm is capable of yielding notable reductions in the total tardiness for practical size problems.

1. 서론

스케쥴링에 대한 기존연구를 보면, 초기에는 SPT(Shortest Processing Time) 등 기본적인 우선순위규칙을 주로 고려한 단순한 법칙을 이용하여 적용 되어왔다. 그러나 오늘날에는 주문자의 다양한 요구로 인한 제품의 다품종화와 납기의 단기화로 인하여 일반적인 우선순위규칙이나 알고리즘에 의해서 주문자의 납기요구를 충족할 수가 없다. 이에 따라 다양한 제품을 요구되는 납기에 맞추는 스케쥴링문제가 공정관리의 가장 중요한 과제가 된다.

따라서 생산자는 주문자의 단납기에 대응할 수 있는 스케쥴링의 적절한 수행도평가척도를 설정하는 문제가 중요하게 된다. 그러나 어떤 한 가지 규칙이 모든 경우에 가장 좋은 결과를 가져오는 것은 아니며, 수행척도로서 특정한 하나의 우선순위규칙이 상대적 유효성은 낮아진다.

또한 다품종 소량 주문생산에서는 제품에 대한 시방 및 납기가 주문자에 의하여 결정되는 경우가 대부분이며, 생산자는 합리적인 공정관리를 통하여 주문시방에 적합한 제품을 납품요구일까지 생산할 수 있도록 계획을 수립하여야 한다.

* 경민전문대학 사무자동화과
** 건국대학교 산업공학과

기존의 스케쥴링을 위한 우선순위규칙은 부품의 이동/운반에 대하여는 고려하지 않고 있다[4]. 이는 비현실적인 발상이므로 부품의 이동/운반시간을 부품 가공시간과 함께 고려하고, 고객의 주문단위를 단수로 가정하지 않고도 해결할 수 있도록 하는 현실적인 우선순위규칙이 필요하다.

그러므로 본 연구에서는 부품의 이동/운반시간과 고객의 주문단위를 고려하며, 급변하는 고객의 수요변화에 합리적으로 대응하기 위하여 초기계획을 수정한 납기우선의 스케쥴링에 대한 다경로 발견적 절차를 제시한다. 이것의 초기 알고리즘[6]은 n/m Job-Shop 환경에서 총작업소요시간을 최소화하는 것에 목표를 둔다. 그러므로 중요한 방법적 수정은 납기내에 납품하는 것을 주요목표로 하는 스케쥴링문제를 수용하기 위하여 필요하다. 이를 통하여 현실적이고 합리적인 생산목표를 달성할 수 있는 보다 효율적인 스케쥴링방법을 개발하는 데 본 연구의 목적을 둔다.

2. 기존 스케쥴링방법의 문제점

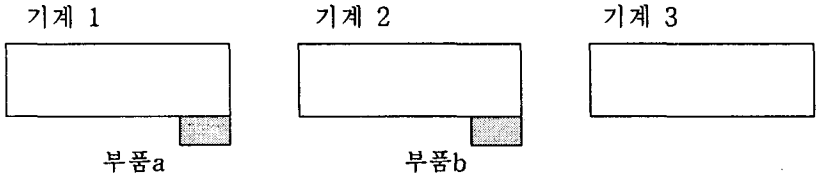
n/m Job-Shop 스케쥴링문제의 기존의 우선순위규칙은 작업시간에 부품의 이동 및 운반시간을 고려하지 않았으며, 주문단위에 대한 단수의 경우만을 다루어 현실적으로 적용에 문제점을 안고 있다.

2.1 작업시간에 기초한 기존 우선순위규칙의 문제점

Job-Shop 스케쥴링문제에 사용되는 기존의 우선순위규칙은 부품의 이동 및 운반 작업에 대하여 전혀 고려하지 않는다. 기존의 작업시간을 기반으로 하는 우선순위 규칙은 이동이나 운반을 작업으로 생각하고, 이동이나 운반에 소요되는 시간을 작업시간으로 간주하여야 하나 이를 고려하지 않는다는 비현실적인 문제점을 안고 있다[3]. 따라서 이동 및 운반시간을 작업시간에 포함시켜 더욱 현실적인 운용을 꾀하여야 한다.

2.1.1 기존의 SPT규칙의 경우

기계 3에서 부품 a의 작업시간 : 9분
기계 3에서 부품 b의 작업시간 : 10분

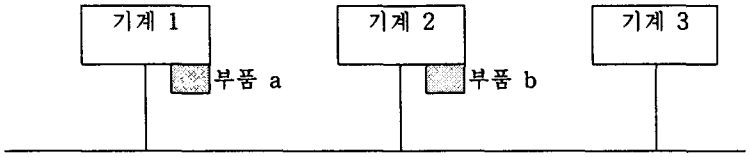


(그림 1) SPT 규칙의 상황

(그림 1)과 같은 경우의 기존 Baker[1] 방법의 SPT 규칙을 적용하면 기계 3은 부품 a를 선택할 것이다. 그러나 다음과 같이 이동 및 운반시간을 고려해 보기로 한다.

2.1.2 기존의 SPT 규칙에 이동 및 운반시간을 포함할 경우

기계 3에서 부품 a의 작업시간 : 9분
기계 3에서 부품 b의 작업시간 : 10분
기계 1과 기계 3간의 이동시간 : 3.5분
기계 2와 기계 3간의 이동시간 : 2분



(그림 2) SPT 규칙에 따른 이동/운반시간을 포함시킨 경우

(그림 2)에서 알 수 있듯이 이동/운반시간까지 고려한 경우 기계 3은 부품 b를 선택하게 된다. 여기서 나타나는 바와같이 기계에서 가공시간만을 고려하는 것과 이동/운반시간을 함께 고려하는 것과는 차이가 있다. 그러므로 현실적인 스케쥴링문제에서는 이동/운반시간을 함께 고려하는 것이 더욱 합리적이라는 것을 알 수 있다.

2.2 납기시간에 기초한 기존 우선순위규칙의 문제점

현재 사용되고 있는 납기를 기반으로하는 우선순위규칙이 가지고 있는 문제점은 우선순위를 결정하기 위하여 작업시간과 납기만을 고려한다는 점이다. 이는 주문의 단위를 단수로 간주하기 때문에 발생되며, 현실적으로 합리적이 못된다[2]. 주문의 크기는 실제로 십단위에서 백단위이상이 되는 경우도 있다. 이러한 점에서 CR은 주문전체의 지연(Delay)을 고려하지 못하고 있다.

실질적으로 CR을 적용했을 때는 우선순위가 높지만 주문단위와 여유시간(납기시간까지 남은시간)을 비교하여 보면 긴급한 주문이 아닐 수 있다. 이와같은 경우에 CR에서는 주문단위와 여유시간의 관계를 고려하지 않기 때문이다.

<표 1> 각 주문에 대한 납기시간과 작업시간 및 이동/운반시간

주문품목	주문단위	납기시간-현재시간	작업시간	이동/운반시간
A	2	4	1	1
B	4	6	1	1

* 단, 이동/운반시간은 주문단위에 관계없이 1회의 소요시간으로 동시에 수행한다.

CR을 이용하여 주문품 A와 B의 우선순위를 비교하면 다음과 같다.

주문 A의 긴급률 : $\frac{4}{(1+1)} = 2$

주문 B의 긴급률 : $\frac{6}{(1+1)} = 3$

따라서 주문품 A의 긴급률이 더 작으므로 우선순위가 더 높다는 것을 알 수 있다. 그러나 주문품 A와 B의 총작업시간 대 여유시간을 비교하면 다음과 같다.

주문 A : 총작업시간 = 주문단위 × 작업시간 + 이동/운반시간 = 2 × 1 + 1 = 3시간
여유시간 = 납기시간 - 현재시간 = 4시간

주문 B : 총작업시간 = 4 × 1 + 1 = 5시간
여유시간 = 납기시간 - 현재시간 = 6시간

주문품 A의 여유도 : $\frac{4}{3} = 1.33$

주문품 B의 여유도 : $\frac{6}{5} = 1.20$

주문품 A는 총작업시간이 3시간이고, 여유시간이 4시간이며, 주문품 B는 총작업시간이 5시간이고, 여유시간이 6시간이다. 주문품 A는 작업시간에 비하여 여유시간은 적지만 여유도가 많아 여유있게 생산하여도 되며, 주문품 B는 상대적으로 여유시간은 많으나 여유도가 적으므로 주문품 A보다 먼저 생산하여야 한다. 즉, 여유도가 적은 주문품 B를 먼저 생산하여야 한다.

그러나 CR에서는 반대로 주문품 A에 더 많은 우선순위를 주고 있다. 따라서 주문단위가 복수일 경우에 CR을 사용하여 우선순위를 부여하면 문제가 발생한다.

3. 납기를 고려한 새로운 SDME 발견적 방법

생산스케줄링에서 지연을 최소화하기 위하여 납기시간이 정해 진 Job-Shop 스케줄링문제에 대하여 효과적인 SDME(Slack Degree Modified Exchange)의 발견적 방법을 제시한다. 이 SDME방법은 다음과 같은 2 가지 측면에서 수행되는 데 첫째, 우선순위규칙은 초기스케줄이 활동(Active) 또는 지연없이(Non-Delay) 발생하도록 한다. 둘째, 초기스케줄에서 목표작업으로 정하여진 집합으로부터 선택된 작업이 좌이동(Left-Shifting)에 의하여 시작시간과 가능한 전체지연을 줄일 수 있는 잔여작업에 대하여 몇 가지 부분집합을 재배열하는 것을 시도한다.

3.1 기호정의

수정교환 발견적 방법에 대한 기호의 정의는 다음과 같다.

(i, j) : 작업 i에 대한 가공 j
$t_s(i, j)$: 작업(i, j)의 시작시간
$t_f(i, j)$: 작업(i, j)의 종료시간
L : 남아있는 지연작업사이의 가장 큰 지연시간과 관련된 이동가능작업의 집합
LHB : 작업(l, k-1)이 목표작업(l, k)의 선행작업일 때 $t_s(l, k-1)$. 만일 목표 작업이 선행작업을 갖지 않으면 LHB(Left Hand Boundary)=0.
B : 목표작업(l, k)을 제외한 LHB 이후 시작한 작업의 집합
D_t : 작업 i의 납기시간
F : 목표작업의 후속작업 집합
ITER : 반복 수
TDi : 작업 i의 지연
TT : 총지연

3.2 여유도를 이용한 스케줄링

3.2.1 여유도

여유도(Slack Degree: SD)는 기존의 긴급률에 의한 방법에 주문단위와 이동/운반시간을 고려한 동태적인 우선순위규칙으로 개발하였다.

$$\text{여유도}(SD) = \frac{\text{작업가능시간(납기시간)}}{(\text{주문단위} \times \text{가공시간}) + \text{이동/운반시간}} \quad (1)$$

여유도는 수주 작업이 어느 정도의 여유를 가지고 있는가를 나타내는 우선순위지수로서 여유도가 적을수록 그 작업은 여유가 없으므로 먼저 수행되어야 함을 나타낸다.

또한 본 연구에서 작성된 스케줄링작업의 선행작업이 연속작업이 아닐경우 이동/운반시간은 그 사이에 수행하는 것으로 조치한다. 다음에 여유도는 우선순위지수라고도 한다.

3.2.2 SD를 이용한 스케줄링

식 (1)의 납기, 이동/운반시간, 주문단위 등을 고려하여 새로운 방법으로 제시한 SD를 사용한 알고리즘은 다음과 같다.

절차 1

t = 0 및 PS = Null Partial Schedule에서 시작한다. 최초에는 St는 선행작업이 없는 모든 작업을 포함한다.

절차 2

$\sigma^* = \text{Min}\{\sigma j\}\ \forall\ j \in St$와 σ^*를 얻을 수 있는 기계 m*을 결정한다..

절차 3

기계 m*을 필요로 하는 각 작업 $j \in St$와 $\sigma j < \varphi^*$을 위하여 특정한 우선순위규칙에 의한 우선순위지수(Priority Index)를 계산한다. 최소의 우선순위지수를 갖는 작업을 찾아서 가장 먼저 PSt에 더하여 단 하나의 부분스케줄 PSt+1을 작성한다.

절차 4

절차 3에서 작성한 새로운 각 부분스케줄 PS_t를 다음과 같이 수정한다.

(a) S_t로부터 작업 j를 없앤다.

(b) 작업 j의 직후속작업을 S_t에 더하여 S_{t+1}을 만든다.

(c) t를 하나씩 증가시킨다.

절차 5

절차 3과 4에서 작성된 PS_{t+1}를 위하여 완전한 스케줄이 작성될 때까지 절차 2로 되돌아간다.

3.3 수정교환 발견적 스케줄링

수정교환 발견적 방법은 2 가지 기본적인 방법을 상용한다. Yang and Ignizio[5]는 이것을 BMO(Backward Move Operation)와 FMO (Forward Move Operation)로 각각 표현하였다. 납기시간 문제에서 RSMO는 목표작업에 대한 새로운 시작시간을 결정하는 방법과 B집합에서 작업을 다루는 방법에서 BMO와는 다르다. 그러나 여기서 사용하는 LSMO는 거의 FMO와 같다. RSMO와 LSMO에서는 목표작업과 B집합이 나타난다는 것을 가정한다. 그리고 목표작업은 일시적으로 스케줄링에서 제거된다는 것도 가정한다.

3.3.1 RSMO(Right-Shift Move Operation)

RSMO라 불리는 각 시간에서의 이 단계는 현재 정의된 단일 목표작업에 대하여 수행된 것이다. RSMO는 지연작업의 증가없이 가능한 한 그것의 후속(Successor)작업에 대하여 마지막 시간에 근접하도록 이동시킨다. 이에 따른 RSMO의 발견적 방법에 대한 알고리즘의 절차는 다음과 같다.

절차 1

: B에 남아있는 작업중에서 가장 마지막 시간을 가진 작업(i,j)를 선택한다.
만약 B가 공집합이면 RSMO를 끝낸다. 그렇지 않으면 절차 2로 간다.

절차 2
: 어떤 다른 작업의 이동없이 그리고 지연이 계속되거나 발생되는 일이 없이 가능한한 오른쪽으로 작업을 이동시킨다. 새로운 $t_f(i,j)$는 $t_s(i,j+1)$보다 크지 않다는 제한이 있다. 만약 작업(i,j)가 마지막 작업이고, 새로운 $t_f(i,j)$가 D_i보다 크지않은 곳에서 작업이 지연되지 않으면 이동이 가능하다. 만약 이동이 가능하지 않으면 절차 1로 되돌아가고 그렇지 않으면 절차 3으로 간다.

절차 3
: (a) 만약 선택된 작업이 목표작업과 같은 기계를 필요로 한다면, 그리고 목표 작업이 성공적으로 재스케줄링될 수 있다면 그 때 절차 3(b)로 간다. 그렇지 않으면 절차 1로 되돌아간다.
(b) 목표작업이 새로운 시작시간을 갖는 곳에서 수정된 스케줄링을 취하고, 이 스케줄링에서부터 다시 목표작업을 이동한다. 그 때 같은 목표작업을 갖는 절차 1로 되돌아간다.

3.3.2 LSMO(Left-Shift Move Operation)

LSMO는 RSMO에 의한 작업에 초점을 맞춘다. 그것의 의도는 전체지연을 줄이기 위한 스케줄링에서 초기위치에 지연작업을 이동하는 것이다. 작업은 시작시간의 순서에 따라 LSMO에 의하여 선택된다. 이에 따른 LSMO 발견적 방법에 대한 알고리즘의 절차는 다음과 같다.

절차 1
: 집합 B의 작업중에서 가장 빠른 시작시간을 갖는 작업(i,j)를 선택한다. 만약 그런 작업이 없으면 (즉 B가 공집합이면) LSMO를 끝낸다. 그렇지 않으면 절차 2로 간다.

절차 2
: 만약 이동이 자원의 제한에 위반되지 않으면 다른 작업의 이동없이 가능한 한 왼쪽으로 작업 (i,j)을 이동한다. 작업(i,j)에 대한 이동의 제약은 새로운 $t_s(i,j)$는 $t_f(i,j-1)$ 보다 더 빠르지 않다. 그리고 절차 1로 되돌아간다.

3.4 SDME 발견적 스케줄링

여유도의 사용과 수정교환을 적용한 새로운 SDME 발견적 방법의 절차는 다음과 같다.

절차 1
t = 0 및 PS = Null Partial Schedule에서 시작한다. 최초의 St는 선행작업이 없는 모든 작업을 포함한다.

절차 2
σ^* = Min{σj} $\forall$ j$\in$St와 σ^*를 얻을 수 있는 기계 m*을 결정한다..

절차 3
기계 m*을 필요로 하는 각 작업 j $\in$ St와 $\sigma j < \varphi^*$을 위하여 특정한 우선순위규칙에 의한 우선순위지수를 계산한다. 최소의 우선순위지수(여유도:SD)가 되는 작업을 찾아서 가장 먼저 PSt에 더하여 단 하나의 부분 스케줄 PSt+1을 작성한다.

절차 4

절차 3에서 작성한 새로운 각 부분스케쥴 PSt를 다음과 같이 수정한다.

(a) St로부터 작업 j를 없앤다.

(b) 작업 j의 직후속작업을 St에 더하여 St-1을 만든다.

(c) t를 하나씩 증가시킨다.

절차 5

절차 3과 4에서 작성된 PSt+1를 위하여 완전한 스케줄이 작성될 때까지 절차 2로 되돌아간다.

절차 6

SD에 의한 방법의 초기화

(a) S_0을 초기 해로 놓는다.

(b) ITER=0, N=0, V=ϕ로 놓는다.

절차 7

이동이 가능한 작업을 모으고, 집합 T_0라 놓는다.

만약 T_0가 공집합이면 멈춘다.

절차 8

(a) $T = T_0 - V$라 하고, 만약 T가 공집합이면 멈춘다.

그렇지 않으면 ITER =ITER + 1로 놓고 절차 8의 (b)로 간다.

(b) 다음과 같은 방식으로 T로부터 목표작업(l,k)을 선택한다.

$$T\,D_t = \underset{i\in I}{Max}\ \{TD_i\} \quad \text{and} \quad t_f(l,k) = \underset{j\in L}{Max}[\,t_f(l,j)]$$

(c) 다음과 같은 연구영역을 정의한다. 만약 목표작업(l,k)의 선행작업이 없으면 LHB=0으로 하고, 그렇지 않으면 LHB= $t_t(l,k-1)$로 놓는다. 여기서 (l, k-1)은 목표작업(l,k)의 선행작업이다.

절차 9

(a) 현 스케쥴링(즉 자원사용을 0이라고 놓은 것)으로부터 목표작업을 이동하고, 집합 B와 F로 구성된 작업을 모은다.

(b) 만약 목표작업이 작업중에 마지막 작업이면 절차 4로 간다. 그러나 만일 목표작업이 작업중에 마지막 작업이 아니면 B는 공집합이 아니다. F가 공집합이면 이때 V 집합에 목표(l,k)를 더한다. 스케쥴링에 목표작업을 재배치하고, 절차 8로 간다.

절차 10

집합 B의 작업에 대한 RSMO를 수행한다. 즉 가능한한 LHB에 근접하게 목표작업을 이동한다. 만약 목표작업에 대한 새로운 작업이 발견되지 않으면 V에 목표작업(l,k)를 더하고, RSMO가 수행되기전에 스케쥴링을 유지시킨다. 그 때 절차 8로 되돌아가고, 그렇지 않으면 절차 11로 간다.

절차 11

집합 B의 작업에 대한 LSMO를 수행한다.

(a) 만약 집합 F의 작업이 좌이동되지 않거나 전체지연에 대해서 감소 되지 않으면 V에 목표(l,k)를 더하고, RSMO를 수행하기 전에 스케줄링을 유지한다. 이때 절차 8로 되돌아간다.

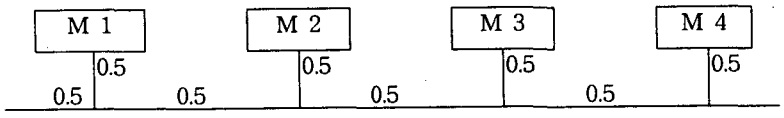
(b) 만약 전체지연이 감소되면 그 때 N=N+1로 놓고, 스케줄링을 새롭게 한다. 즉 $S = S_N$이라 놓고, 절차 7로 되돌아간다.

4. 실증연구

<표 2>에서 <표 6>과 같이 6/4 Job-Shop 스케줄링이 주어진 경우 여유도와 수정교환을 사용한 SDME 발견적 방법에 의하여 스케줄을 작성하면 다음과 같다.

<표 2> 6/4 Job-Shop 스케줄링문제의 가공시간

작업 \ 순서	1	2	3	4
1	4.0	6.5	7.0	6.5
2	3.0	4.0	3.0	2.0
3	7.0	9.0	3.5	4.0
4	4.5	6.0	5.5	6.0
5	5.0	4.0	2.0	2.0
6	6.5	7.5	6.0	4.5



(그림 3) 4 대의 기계에 대한 이동/운반시간

<표 3> 6/4 Job-Shop 스케줄링문제의 이동/운반시간

작업 \ 순서	1	2	3	4
1	1.0	1.5	2.0	1.5
2	1.0	2.0	1.5	2.0
3	2.0	2.0	1.5	2.0
4	2.5	2.0	1.5	2.0
5	1.5	2.0	2.5	2.0
6	2.5	1.5	2.0	1.5

<표 4> 6/4 Job-Shop 스케줄링문제의 가공기계

작업 \ 순서	1	2	3	4
1	1	2	4	3
2	1	3	4	2
3	3	1	2	4
4	4	2	3	1
5	2	4	1	3
6	4	3	1	2

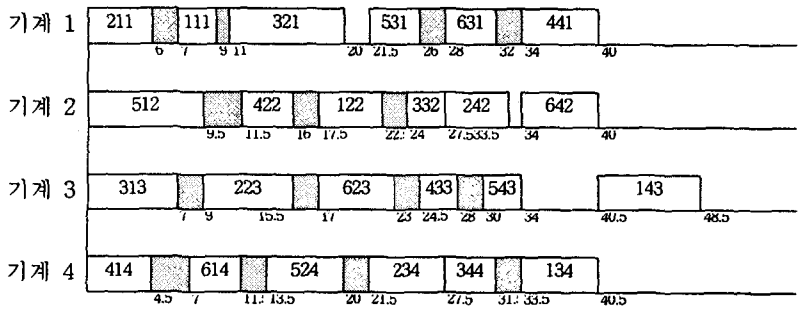
<표 5> 6/4 Job-Shop 스케줄링문제의 주문단위 및 납기시간

작 업	1	2	3	4	5	6
주문단위	1	2	1	1	2	1
납기시간	43	44	44	42	53	55

<표 6> 6/4 Job-Shop 스케줄링문제의 총작업완료시간

작업 \ 순서	1	2	3	4	합계
1	5.0	8.0	9.0	8.0	30.0
2	7.0	10.0	7.5	6.0	30.5
3	9.0	11.0	5.0	6.0	31.0
4	7.0	8.0	7.0	8.0	30.0
5	11.5	10.0	6.5	6.0	34.0
6	9.0	9.0	8.0	6.0	32.0

먼저 SD를 사용하여 초기스케줄링을 작성하면 연산결과는 <표 7>과 같고, 칸트도표는 (그림 4)와 같다.



(그림 4) 6/4 Job-Shop 문제의 SD를 사용한 스케줄링 칸트도표

< 표 7> 6/4 Job-Shop 문제의 SD를 사용한 스케줄링 연산결과

절 차	(f_1, f_2, f_3, f_4)	$j \in S_{i-1}$	t_j	σ_j	SD_i	min SD_i
1	(0, 0, 0, 0)	1 1 1	5	0 *	8.6	
		2 1 1	7	0 *	6.3	
		3 1 3	9	0 *	4.9	
		4 1 4	7	0 *	6.0	
		5 1 2	11.5	0 *	4.6	**
		6 1 4	9	0 *	6.1	
2	(0, 11.5, 0, 0)	1 1 1	5	0 *	8.6	
		2 1 1	7	0 *	6.3	
		3 1 3	9	0 *	4.9	**
		4 1 4	7	0 *	6.0	
		5 2 4	10	11.5		
		6 1 4	9	0 *	6.1	
3	(0, 11.5, 9, 0)	1 1 1	5	0 *	8.6	
		2 1 1	7	0 *	6.3	
		3 2 1	11	9		
		4 1 4	7	0 *	6.0	**
		5 2 4	10	11.5		
		6 1 4	9	0	6.1	
4	(0, 11.5, 9, 7)	1 1 1	5	0 *	8.6	
		2 1 1	7	0 *	6.3	**
		3 2 1	11	9		
		4 2 2	8	11.5		
		5 2 4	10	11.5		
		6 1 4	9	7		
5	(7, 11.5, 9, 7)	1 1 1	5	7 *	7.2	
		2 2 3	10	9		
		3 2 1	11	9		
		4 2 2	8	11.5		
		5 2 4	10	11.5		
		6 1 4	9	7 *	5.3	**
6	(7, 11.5, 9, 13.5)	1 1 1	5	7	*	
		2 2 3	10	9		
		3 2 1	11	9		
		4 2 2	8	11.5		
		5 2 4	10	13.5		
		6 2 3	9	13.5		
7	(11.0, 11.5, 9, 13.5)	1 2 2	8	11.5		
		2 2 3	10	9	*	
		3 2 1	11	11.0		
		4 2 2	8	11.5		
		5 2 4	10	13.5		
		6 2 3	9	13.5		
8	(11.0, 11.5, 17, 13.5)	1 2 2	8	11.5		
		2 3 4	7.5	17		
		3 2 1	11	11.0	*	
		4 2 2	8	11.5		
		5 2 4	10	13.5		
		6 2 3	9	17		

(계속)

9	(20, 11.5, 17, 13.5)	1 2 2 2 3 4 3 3 2 4 2 2 5 2 4 6 2 3	8 7.5 5 8 10 9	11.5 * 17 20 11.5 * 13.5 17	3.94 3.81 	 **
10	(20, 17.5, 17, 13.5)	1 2 2 2 3 4 3 3 2 4 3 3 5 2 4 6 2 3	8 7.5 5 7 10 9	17.5 17 20 17.5 13.5 17	 * 	
11	(20, 17.5, 19, 21.5)	1 2 2 2 3 4 3 3 2 4 3 3 5 3 1 6 2 3	8 7.5 5 7 6.5 9	17.5 21.5 20 17.5 21.5 17	 *	
12	(20, 17.5, 24.5, 21.5)	1 2 2 2 3 4 3 3 2 4 3 3 5 3 1 6 3 1	8 7.5 5 7 6.5 8	17.5 21.5 20 24.5 21.5 24.5	* 	
13	(20, 24, 24.5, 21.5)	1 3 4 2 3 4 3 3 2 4 3 3 5 3 1 6 3 1	9 7.5 5 7 6.5 8	24 21.5 * 24 24.5 21.5 * 24.5	 3.0 4.8 	 **
14	(20, 24, 24.5, 27.5)	1 3 4 2 4 2 3 3 2 4 3 3 5 3 1 6 3 1	9 6 5 7 6.5 8	27.5 27.5 24 24.5 21.5 24.5	 * 	
15	(28, 24.5, 24.5, 27.5)	1 3 4 2 4 2 3 3 2 4 3 3 5 4 3 6 3 1	9 6 5 7 6 8	27.5 27.5 24 24.5 28 28	 * 	
16	(28, 27.5, 24.5, 27.5)	1 3 4 2 4 2 3 4 4 4 3 3 5 4 3 6 3 1	9 6 6 7 6 8	27.5 27.5 27.5 24.5 28 28	 * 	

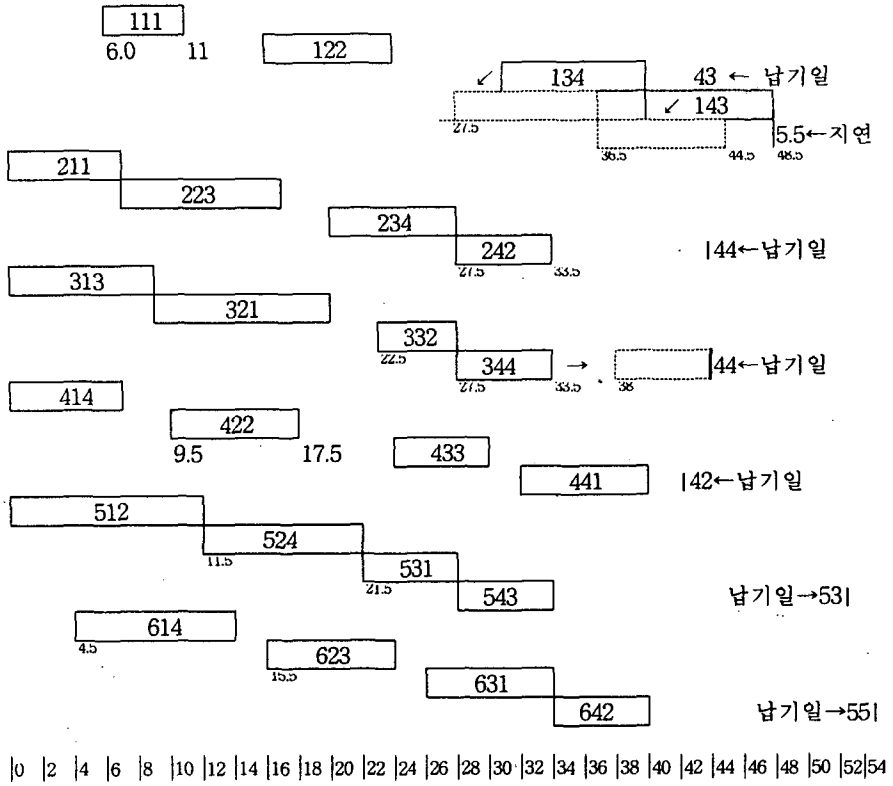
(계속)

17	(28, 27.5, 30, 27.5)	1 3 4 2 4 2 3 4 4 4 4 1 5 4 3 6 3 1	9 6 6 8 6 8	28 27.5 * 27.5 * 30 30 28	 2.7 2.7	 **
18	(28, 27.5, 30, 33.5)	1 3 4 2 4 2 4 4 1 5 4 3 6 3 1	9 6 8 6 8	33.5 27.5 30 30 28	 * 	
19	(28, 33.5, 30, 33.5)	1 3 4 4 4 1 5 4 3 6 3 1	9 8 6 8	33.5 30 30 28	 *	
20	(34, 33.5, 30, 33.5)	1 3 4 4 4 1 5 4 3 6 4 2	9 8 6 6	33.5 34 30 34	 *	
21	(34, 33.5, 34, 33.5)	1 3 4 4 4 1 6 4 2	9 8 6	33.5 34 34	*	
22	(34, 33.5, 34, 40.5)	1 4 3 4 4 1 6 4 2	8 8 6	40.5 34 * 34 *	 1.0 3.5	 **
23	(40, 33.5, 34, 40.5)	1 4 3 6 4 2	8 6	40.5 34	 *	
24	(40, 40, 34, 40.5)	1 4 3	8	40.5	*	

* 절차 17에서는 (2 4 2)와 (3 4 4)가 같은 값을 갖기 때문에 임의로 선택하였음.

빗금친 부분은 작성된 스케줄링작업의 선행작업이 연속작업이 아닐 경우 이동/운반시간은 그 사이에 수행할 수 있는 처리 중복시간이다.

그리고 좌 · 우로 이동시킨 새로운 스케줄을 작성하면 (그림 5)와 같다.



(그림 5) SDME 방법에 의하여 개선된 스케줄

절차 1

: (그림 4-4)로부터 스케줄링은

S_0 = [{(1,1), 6.0}, {(1,2), 6}, {(1,3), 31.5}, {(1,4), 40.5}, {(2,1), 0}, {(2,2), 7}, {(2,3), 20}, {(2,4), 27.5}, {(3,1), 0}, {(3,2), 9}, {(3,3), 22.5}, {(3,4), 27.5}, {(4,1), 0}, {(4,2), 9.5}, {(4,3), 23}, {(4,4), 22}, {(5,1), 0}, {(5,2), 11.5}, {(5,3), 21.5}, {(5,4), 28}, {(6,1), 4.5}, {(6,2), 15.5}, {(6,3), 26}, {(6,4), 34}].

다음 변수를 ITER=0, N=0, V=ϕ로 초기화한다.

절차 2

: 지연 : $TD_1=5.5$ $TD_2=0$, $TD_3=0$, $TD_4=0$, $TD_5=0$, $TD_6=0$. TT = 5.5

이동이 가능한 작업을 모은다. 즉, T_0 : T_0={(1,1), (1,2), (1,3)}

절차 3

: ITER = 1, T = T_0-V = {(1,1), (1,2), (1,3)}

TD_1=Max{ TD_i}=Max{5.5}=5.5, L={(1,1), (1,2), (1,3)}

L로부터 목표작업을 선택한다. 작업(1,3)은 가장 늦은 마지막 시간을 갖는다. 그러므로 목표작업을 (1,3)이라 놓고, LHB= t_i(1,2)=24로 놓는다.

절차 4

: 집합 B와 F를 모은다. B={(1,3), (1,4), (2,4), (3,4), (4,4), (5,4), (6,3), (6,4)}
F= {(1,4)}

절차 5

: 집합 B에서 다음과 같이 대상작업에 대하여 RSMO를 수행한다.
(1,4) 변화되지 않는다.
(2,4) 38에서 시작하기 위해 이동
(3,4) 38에서 시작하기 위해 이동

작업 (3,4)는 목표작업(1,3)과 같은 기계를 필요로 한다. 그러므로 (1,3)에 대하여 가능한한 가장 빠른 시작시간을 찾는다. 이용가능한 새로운 시작시간 때문에 재스케줄링된 목표작업 (1,3)은 다음과 같다. {(1.3), 31.5} → {(1,3) 27.5}. 구해진 스케줄을 유지하고, 이 스케줄링으로부터 다시 목표작업을 이동하며, 다음과 같은 RSMO를 계속하여 수행한다.

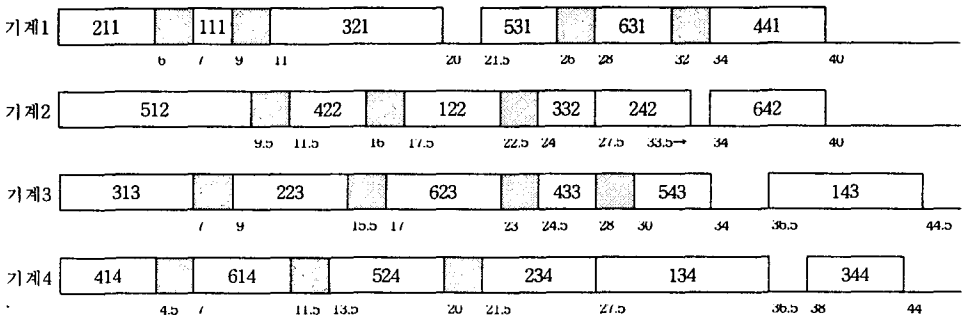
(4,4) 34에서 시작하기 위하여 이동, (5,4) 47에서 시작하기 위하여 이동
(6,4) 49에서 시작하기 위하여 이동, (6,3) 41에서 시작하기 위하여 이동
목표작업에 대한 새로운 시작시간이 발견되면, 절차 6으로 간다.

절차 6

: 집합 B와 F 작업에 대하여 절차 5에 의하여 구한 스케줄링에 LSMO를 수행한다.
(2,4) 27.5에서 시작하기 위하여 이동, (5,4) 28에서 시작하기 위하여 이동
(1,4) 36.5에서 시작하기 위하여 이동, (3,4) 변화되지 않는다.
(4,4) 32에서 시작하기 위하여 이동, (6,4) 34에서 시작하기 위하여 이동
(6,3) 26에서 시작하기 위하여 이동

납기에 기준하여 지연이 감소되었는가 체크한다. $TD_1 = 1.5$, $TD_2 = 0$, $TD_3 = 0$, $TD_4 = 0$, $TD_5 = 0$, $TD_6 = 0$ 그리고 TT=1.5 이다. 전체지연은 4만큼 줄었다.

상기 결과에 의한 새로운 스케줄은 (그림 6)과 같다.



(그림 6) SDME 방법에 의한 스케줄링의 칸트도표

여기서 나타난 빗금친 부분은 작성된 스케줄링작업의 선행작업이 연속작업이 아닐 경우 이동/운반시간은 그 사이에 수행할 수 있는 처리 중복시간이다.

스케줄링의 평가척도에 의하여 수행도면에서 CR 방법과 SDME 방법에 의한 스케줄링의 결과를 비교하여 보면 <표 8>과 같이 산출된다.

<표 8> 스케줄링평가척도에 대한 CR과 SDME의 결과치비교

구 분	M	$\overline{F}$	I	$\overline{T}$
(A) CR 방법	52.5	49.00	14.0	5.42
(B) SDME 방법	44.5	39.33	6.0	0.25
차이 (A)-(B)	8.0	9.67	8.0	5.17

그러므로 <표 8>에 의하면 CR 방법보다 SDME 방법이 모든 수행평가척도면에서 우수하다. 아울러 기존의 우선순위규칙의 CR 방법과 본 연구의 새로운 SDME 방법에 의하여 스케줄링한 후 각 작업의 납기를 작업별로 각각 비교하여 보면 <표 9>와 같다.

<표 9> 납기를 고려한 CR과 SDME 방법의 결과치비교

구분 \ 작업		1	2	3	4	5	6
(A) 납기시간		43	44	44	42	53	55
CR 방법	(B) 완료시간	46.0	52.5	48.5	48.5	52.0	46.5
	차이 (A)-(B)	-3.0	-8.5	-4.5	-6.5	1.0	8.5
SDME 방법	(C) 완료시간	44.5	33.5	36.5	40.0	34.0	40.0
	차이 (A)-(C)	-1.5	10.5	7.5	2.0	19.0	15.0
CR-SDME 차이 (B)-(C)		1.5	19.0	12.0	8.5	18.0	6.5

<표 9>에서와 같이 CR 방법에 의하여 스케줄링하면 작업 1, 2, 3, 4 즉, 4 개의 작업이 빗금친 부분과 같이 납기가 지연되고, 새로운 SDME 방법에 의하여 스케줄링하면 작업 1 만이 지연되며, 그 지연시간도 3에서 1.5로 50% 감소되며, 기존의 CR 방법과 새로운 SDME 방법의 차이시간에서 볼 수 있듯이 모든 작업의 완료시간이 향상되는 효과를 가져온다.

5. 결론

본 연구는 현실적 문제의 적용성을 위하여 단일품목위주의 수주 및 가공을 위주로하는 기존의 연구와는 달리 각 수주 및 가공단위가 단수 또는 복수의 모든 경우에 적용하도록 하는데 초점을 맞추었으며, 이러한 경우 수주단위를 고려한 작업순서의 결정에 따른 납기지연은 하나의 가공부품단위로서 결정되는 것이 아니라 주문단위 전체의 지연으로 결정된다. 다시 말하여 주문단위 모두를 납기내에 납품하지 않으면 손해를 입게 되므로 주문단위만큼을 납기내에 제조하는 것이 중요하다.

또한 본 연구에서는 기존연구가 부품의 가공시간에 모든 것을 포함시키고, 부품의 이동 및 운반시간을 고려하지 않고 있다는 한계점을 대상으로하여 가공시간과 이동/운반시간을 불리하여 처리하며, 납기지연작업이 나타나면 작업스케줄의 좌·우이동에 따른 수정교환을 통하여 지연작업과 지연시간을 최소화하는 새로운 SDME의 스케줄링방법을 개발하였다.

SDME 방법은 활동스케줄에서 시작하며, SDME 방법을 통하여 우선순위규칙과 주문단위, 이동 및 운반시간을 고려한 상황하에서 지연작업의 수를 감소하고, 전체 지연시간을 절감할 수 있다는 것을 보였다. 그러므로 새로운 SDME 방법은 납기가 정해진 일반 n/m Job-Shop

스케쥴링문제의 응용에 효과적으로 이용할 수 있다. 실질적으로 납기내 이행도를 향상시킬 수 있어 납기준수는 물론 공정내 정체, 혼잡도, 재공품재고, 물류비 등의 효율적인 관리에 활용도가 높다.

n/m Job-Shop 스케쥴링문제에 있어 평가척도로서 납기문제를 중점적으로 모색하였으나 공정관리에 있어 재공품의 정체 및 체류시간의 단축문제는 재공량의 감축문제와 관련하여 중요한 과제이므로 납기와 정체 및 체류시간의 단축문제를 동시에 고려한 연구가 앞으로의 중요한 연구과제라 생각된다.

참고문헌

1. Baker, K. R., *Introduction to Sequencing and Scheduling*, John Wiley & Sons, Inc., New York, pp.29~35, 65~70, 178~202, 1974.
2. Kim, Y. D., "A Comparison of Dispatching Rules for Job Shops with Multiple Identical Jobs and Alternative Routings" , *International Journal of Production Research*, Vol.28, No.5, pp.953~962, 1990.
3. Sabuncuoglu, I., et al., "Experimental Investigation of Machine and AGV Scheduling Rules Against the Mean Flow Time Criterion", *International Journal of Production Research*, Vol.30, No.7, pp.1617~1635, 1992.
4. Wu, S. D., et al., "An Application of Discrete Event Simulation to On Line Control and Scheduling in Flexible Manufacturing", *International Journal of Production Research*, Vol.27, No.9, pp.1603~1623, 1989.
5. Yang, T. and Ignizio, J. P., "An Algorithm for the Scheduling of Army Battalion Training Exercises", *Computers and Operations Research*, Vol.14, No.5, pp.479~491, 1987.
6. Yang, T., Ignizio, J. P. and Deal, D. E., "An Exchange Heuristic for Generalized Job Shop Scheduling", *Engineering Optimization*, Vol.15, No.1, pp.83~96, 1989.