Journal of the Society of Korea Industrial and Systems Engineering
Vol, 28, No. 1, pp.92－96, March 2005.

동적 도착의 총 납기 지연 최소화 문제

오세호* · 이근부* · 양희중*

*청주대학교 산업정보시스템공학과

A Batching Problem to minimize the total Tardiness with Dynamic Arrivals

Se-Ho Oh* · Keun-Boo Lee* · Hee-Joong Yang

*Department of Industrial & Information Systems Engineering, Chongju University

This paper deals with a batch processor model in which the batch processing times depend on the jobs assigned to the batch. Each job has a distinct processing time which is determined as not the exact value but the range from the lower limit to the upper, which makes it possible to group several jobs into the same batch. In point of this flexibility our model can be referred to as the generalization of the burn-in model in which the upper limit of each job is unbounded. The jobs to be scheduled may be available nonsimultaneously. Therefore they have different ready times. We develop the model to describe the problem situation and the heuristic methods to minimize the total tardiness. And our batching rule is compared with other dispatching ones.

Keywords : range, batching, nonsimultaneous, dispatching

1. 서 론

일괄 처리는 여러 개의 단위 작업들을 동시에 가공하는 것을 뜻한다. 이러한 처리 과정을 포함하는 일정계획 모형들은 제빵, 도자기 열처리, PCB(printed circuit board) 검사 공정 등에서 찾아 볼 수 있다.[2,4] 기계에서 일괄처리될 수 있는 작업 능력을 자재 취급에서의 단위적재처럼 해석한다면 일괄처리 문제는 적재 문제와 마찬가지로 NP-hard의 복잡도를 갖는다고 할 수 있다.[3] 제빵, 도자기 열처리 공정들은 일괄처리 시간이 미리 정해진 경우가 대부분이지만 그렇지 않은 경우도 있다. 일괄처리 공정시간이 일괄처리 묶음을 구성하고 있는 단위 작업들의 공정시간에 따라 결정되는 좀 더 복잡한 모형을 그 예라고 할 수 있다. 이러한 모형은 burn-in 모형이라 일컬어지고 있는데 반도체 제조 공정을 분석하는 과정에서 제시되었다.[5,6,7]

본 연구에서는 burn-in 모형의 일반화된 형태로서 단위 작업들의 가공 시간이 최소 일정한 값 이상이 되어야 하는 조건과 일정한 값을 넘어서도 안 되는 조건을 동시에 만족시켜야 하는 문제를 다루고 있다. 즉, 각각의 단위 작업들이 하한값과 상한값 범위 안에서 처리되어야 하는 점에서 일반적인 burn-in 모형들과 구별된다. 일정 시간을 넘기게 되면 과숙성의 문제점이 발생하는 유기물 숙성 과정에 적용시켜 볼 수 있을 것이다.

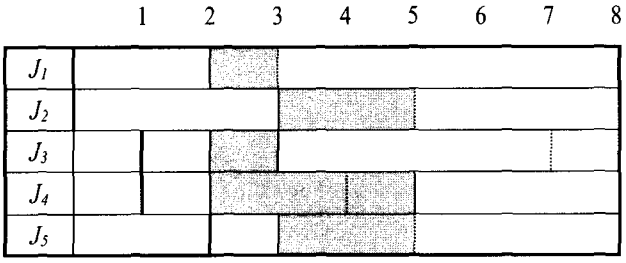
이미 제시된 가공 가능 시점이 동일한 모형[1]에서 나아가 준비 시간이 단위 작업마다 다른 동적 도착 모형을 다루고자 한다. 이러한 문제 상황을 모형화 하고 목적함수로서 총 납기지연 최소화를 설정하여 발견적 해법을 제시하고자 한다. 해법에서 사용된 일괄 작업 선정 기준과 주요한 휴리스틱 규칙들과 비교를 하였다.

2. 문제의 정의

대부분의 일괄 공정은 사전에 정해진 시간 안에 처리된다. 그러나 경우에 따라서는 일괄 처리되기 위해 묶여진 단위 작업들의 공정 시간에 의해 결정되기도 한다. 이러한 모형은 burn-in 모형으로 지칭되기도 하는데 Lee et al[7] 이 반도체 제조 공정을 분석하고자 제시하였다. 각기 처리 시간이 상이한 웨이퍼들을 일괄처리 되는데 일정 시간 이상 처리되어야 하는 상황을 모형화 하였다. 본 연구에서는 이러한 조건에 일정 시간을 넘지 말아야 하는 조건이 추가되는 공정들을 고려해보았다. 즉 각 단위 작업들의 공정 시간이 상한과 하한 값 범위 안에서만 지켜지면 가공이 이루어지는 것으로 본다. 또한 각 작업들의 도착 시점이 각기 다르다. 따라서 작업가능 시간이 다르게 된다. 본 연구에서 제시하고자 하는 모형을 위해 몇 가지 가정을 설정하고 예제를 통해 구체적으로 살펴보면 다음과 같다.

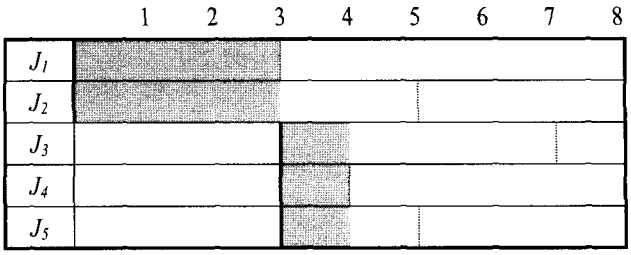
가정 1 : 일괄처리기의 능력(capacity) 범위 안에서 단위 작업들이 투입된다.
가정 2 : 일단 일괄처리공정이 시작되면 새로운 작업이 추가될 수 없고 제거될 수도 없다.
가정 3 : 각 작업들의 공정 시간은 최대, 최소값의 가능 범위로 주어진다.
가정 4 : 단위 작업들의 도착 시점과 납기 시점이 주어진다.

아래 그림은 5개의 단위 작업들의 도착 시점(실선), 처리 가능 범위(음영), 납기(점선)을 도시한 것이다.



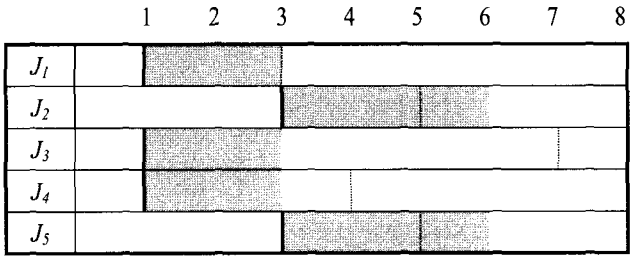
<그림 1> 5개의 단위 작업들의 예

위의 예제에서 일괄 처리기의 능력이 4라면 시작 시점에 (J_1, J_2)를 한 묶음으로 처리한 다음, T=3에 (J_3, J_4, J_5)를 묶음으로 처리하는 방안과 J_1 과 J_2 를 1단위 시간 지연시켜 (J_1, J_3, J_4)를 T=1에 일괄 처리한 다음 T=3에 (J_2, J_5)를 처리하는 방안을 고려해 볼 수 있다. 전자의 경우(<그림 2>)에는 납기 지연과 총 작업 소요시간이 각각 0, 4로, 후자의 경우(<그림 3>)에는 2, 6으로 계산된다.



굵은실선 : 시작시점, 점선 : 납기시점, 음영 : 공정시간

<그림 2> (J_1, J_2), (J_3, J_4, J_5) 묶음의 경우



굵은실선 : 시작시점, 점선 : 납기시점, 음영 : 공정시간

<그림 3>(J_1, J_3, J_4), (J_2, J_5) 묶음의 경우

3. 해법의 개발

납기지연은 작업의 완료시간과 비례하지 않기 때문에 단일기계 기본 모형에서조차도 납기지연 최소화 해법을 개발하기가 쉽지 않다.

정의 : 작업 J_i, J_j 에 대하여 $m_i < m_j$이 $n_i \leq n_j$을 의미한다면 두 모수 집합들은 agreeable하다고 한다.

위의 정의에 의해 모든 단위 작업 쌍의 공정 시간과 납기가 agreeable이면 총납기지연이 SPT 혹은 EDD 순서에 의해 최소화된다. 따라서 일괄처리기의 처리 용량이 1이고 모든 단위 작업 쌍에 대해서 L_j와 d_j가 agreeable인 특별한 경우에만 총납기지연을 SPT 순서 혹은 EDD 순서에 의해 최소화 시킬 수 있을 것이다. 만약 이러한 상황에서 일괄처리가 가능하다면 낱개로 처리하는 것보다 가능한 한 묶음으로 처리하는 것이 후속되는 작업들의 납기지연을 줄일 가능성을 높여 줄 것이다. 즉 일괄처리가 낱개로 처리하는 대안을 지배한다. 그러나 이와 같은 지배성은 모든 단위 작업 쌍에 대해서 L_j와 d_j가 agreeable일 때만 성립되기 때문에 본 모형에 적용시키기에는 제한적일 수밖에 없다.

지금까지 살펴본 SPT 혹은 EDD 규칙보다는 약한(weak) 규칙으로서 진행 시점 T에 의해 순서가 결정되는 규칙을 고려해 보자.

정리 1. 시점 T에서 단위 작업 J_i와 J_j의 순서를 결정하려 할 때, 다음 식이 성립하는 경우를 제외하고는 납기가 이른 작업을 선행 시킨다.

$$T + \max\{p_i, p_j\} > \max\{d_i, d_j\} \quad \cdots\cdots (1)$$

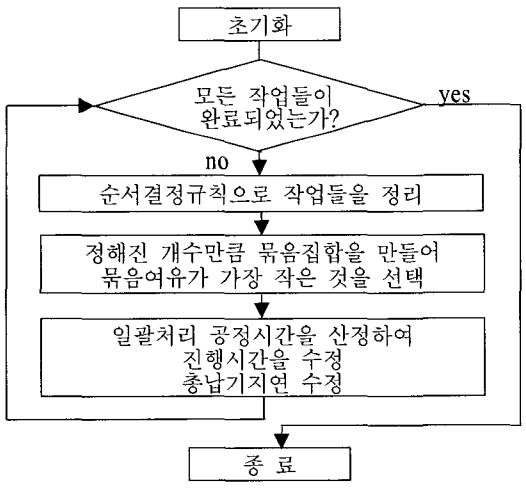
위의 정리를 통해서 두 작업간의 순서는 어느 진행 시점에서 결정되느냐에 따라 달라진다. 다른 각도에서 살펴보면 위의 규칙은 최적조건의 충분조건이 아닌 필요조건이라고 할 수 있다. 그러나 이와 같이 최적 선행관계를 보장할 수 없지만 진행 시점 T에서 고려되는 작업들(candidates)간의 선행관계를 파악할 수 있기 때문에 일괄 집합을 생성하는데 이용될 수 있다.

앞으로의 논리 전개에 필요한 표기와 용어 정의는 다음과 같다.

표기와 용어 정의

- T_k : k번째 일괄처리 후의 시점
- J_i (i = 1, 2,..., n) : 단위 작업(job)
- L_i, U_i : J_i 의 공정시간의 하한, 상한 값
- d_i : J_i 의 납기
- C : 처리기의 일괄처리 능력
- TD_t : 시점 t까지의 총납기지연시간
- BP_q : 묶음 q의 일괄처리공정시간
- B^j_t (j = 1, 2,...,q) : 시점 t에서의 일괄처리가 가능한 작업들의 집합
- N : 모든 작업들의 집합
- FN : 처리된 작업들의 집합
- R_t : T_k에서 묶음 대상이 되는 작업들의 집합
- BS_j : 묶음 여유(batch slack)

해법의 절차를 흐름도로 나타내면 다음과 같다.



<그림 4> 해법의 흐름도

k번째 일괄처리가 이루어진 직후인 T_k 시점에서 단위 작업들은 다음의 규칙들에 의해서 배열된다.

순서 결정 규칙

1) SPT
2) EDD
3) 수정 납기(modified due date)
4) 탄력성(flexibility of processing time)

SPT 규칙을 적용할 경우, 고려되는 각 작업들의 공정시간을 L_i과 U_i사이의 어떤 값으로 고정시킬 것인지 결정하여야 한다. 본 연구에서는 L_i를 채택하였는데 납기가 늦게 잡혀있다면 탄력성 있게 좀더 크게 잡는 방법 등에 대한 분석이 요구된다. 두 번째 EDD 규칙은 기존의 방법과 동일하다. 세 번째로 수정납기 규칙(MDD)은 다음 식에 의해 구해진 납기의 올림차순으로 정리한다.

$$md_i = \max\{T_k + L_i , d_i\}, \ \forall i \in R_t \quad \cdots\cdots (2)$$

이 방법은 최적 선행관계를 보장할 수 없지만 진행시점 T_k에서 고려되는 작업들(R_t)간의 선행관계를 파악할 수 있기 때문에 일괄 집합을 생성시키는데 이용하였다. 마지막 네 번째 탄력성 규칙은 단일기계일정계획에서 사용되는 긴급률 규칙과 흡사하다. 단위 작업들에 탄력성을 부여하여 가중치처럼 사용한다. 탄력성(FPT)은 공정시간의 상한 값과 하한 값의 차의 역수로 정의하여 긴급률규칙의 가중치로 계산하였다. 따라서 식 (3)의 값이 작은 순으로 우선순위를 정한다.

$$(L_i - T_k)(U_i - L_i) \quad \cdots\cdots (3)$$

이상의 방법들을 통해 나열된 작업들은 몇 개의 묶음 집합들을 생성시킨다. 정해진 개수(q)만큼 B^j_t 를 만들어 각 묶음 집합들의 묶음여유를 계산한다.

$$BSi = \sum_{i \in Bjt} \{di - BPj\}$$
$$\text{where } BPj = \max\{Li , i \in Bjt\} \quad \cdots\cdots (4)$$

묶음여유는 묶음에 소속된 작업들이 t시점에 가공을 시작한다고 가정했을 때 각각의 작업들이 납기에 접근한 정도의 합을 나타낸다. 따라서 묶음여유가 작은 묶음 집합을 선택하여 묶음공정시간을 산정하고 마지막 단계로 진행한다. 이 때 선택된 묶음집합이 B^r_t이라면 공정시간은 다음과 같이 계산된다.

$BP_r = \max\{L_i, i \in B^r_t\}$ ..... (5)

끝으로 해법의 절차를 약술하면 다음과 같다.

해법의 절차

단계 0 : $FN = \varnothing$. $T_0 = 0$. $k = 0$

단계 1 : kard(FN) = n 이면 종료.
집합 N-FN을 대상으로 순서결정규칙에 의해 정리

단계 2 : R_t를 대상으로 묶음 B^j_t (j = 1, 2,...,q)를 생성하여 묶음여유가 가장 작은 것(B^r_t)을 선택.

단계 3 : 일괄공정시간 산정과 처리된 작업 수정.
$BP_r = \max\{L_i, i \in B^r_t\}$)
$k = k + 1$
$FN = FN - B^r_t$
$T_k = T_k + BP_r$

4. 수치 예제와 순서 결정 규칙

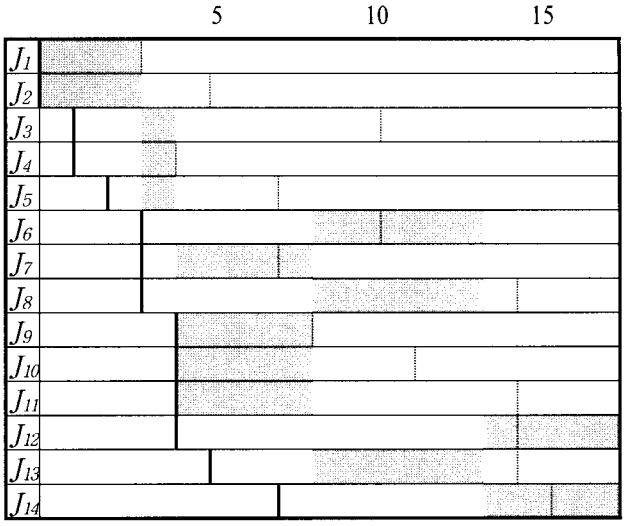
14개의 단위 작업을 일괄처리하는 문제(<표 1>)의 해를 앞 절의 방법으로 구해보면 <그림 5>와 같다.

<표 1> 수치예제

	도착시각	납기	Li	Ui	
1	0	3	2	3	
2	0	5	3	5	
3	1	10	1	2	
4	1	4	1	4	
5	2	7	1	3	
6	3	10	4	6	
7	3	7	4	8	
8	3	14	5	8	C=4
9	4	8	3	5	
10	4	11	2	4	
11	4	14	3	5	
12	4	14	4	10	
13	5	14	3	5	
14	7	15	4	5	

순서결정규칙으로 수정납기방법을 채택하여 구한 결과로 <그림 5>를 얻는다. 작업 총완료시간은 17이고 총납기지연은 9이다.

앞 절에서 제시한 순서결정규칙들을 비교하기 위해 단위 작업들의 개수가 각각 15, 20, 25, 30으로 구성된 문제를 20개씩 만든 다음 각각의 규칙들을 적용시켜 총납기지연시간을 구하였다. 매회 묶음집합을 만들기 위해 고려되는 작업들(R_t)의 개수를 크게 잡으면 계산 량이 증가한다. 그리고 단위작업들의 크기가 작은 경우에는 순서결정규칙의 상대적 비교가 확연하게 들어나지 안을 수도 있다. 따라서 R_t를 C로 하여 순서결정규칙을 적용하였을 때의 총납기지연시간을 계산하여 상대적 효율성 비교를 하였다. 각 문제에 각각의 규칙으로 적용시켜 총납기지연과 총납기지연의 평균을 구한 다음 총납기지연을 평균으로 재산정한 값들을 <표 2>와 같이 얻었다.



굵은실선 : 도착시점, 점선 : 납기시점, 음영 : 공정시간

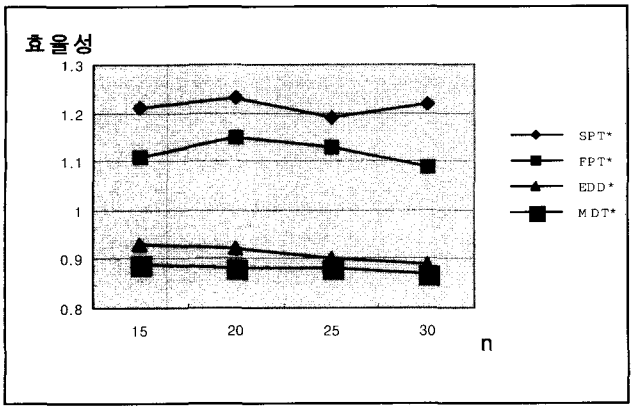
<그림 5> 수치예제의 결과

<표 2> 효율성 산정치

n	SPT*	FPT*	EDD*	MDD*
15	1.21	1.11	0.93	0.89
20	1.23	1.15	0.92	0.88
25	1.19	1.13	0.90	0.88
30	1.22	1.09	0.89	0.87

MDD*=4MDT/(MDD+EDD+FPT+SPT)

효율성 산정치들을 그림으로 나타내면 <그림 5>와 같다. 수치들이 작으면 작을수록 상대적으로 납기지연이 작은 해를 구할 수 있다는 것을 의미한다. SPT규칙과 탄력성 규칙의 성능은 수정납기에 비해 효율성이 떨어진다. 이들은 납기시점은 고려하지 않고 단위공정시간의 상한과 하한 값만 반영한 결정기준을 이용하기 때문이다. 특히 SPT규칙은 공정시간의 하한값만을 단순 비교했기 때문에 효율이 현저히 떨어진다고 볼 수 있다. 그러나 상한값까지 공정시간으로서 고정이 가능하므로 개선의 여지는 많다고 볼 수 있다. 그리고 탄력성규칙의 경우도 납기시점을 고려한 측정수식의 개발도 가능하다.



<그림 6> 순서결정규칙들 간의 효율성 비교

5. 결 론

본 연구에서 다룬 것은 구간공정시간을 갖는 단위 작업들을 일괄처리할 때 발생하는 납기 지연을 최소화시키는 문제이다. 단위 작업들의 공정시간이 서로 다르지만 일정 범위 안에서 탄력적으로 운용할 수 있기 때문에 일괄처리 접근이 가능하다.

제시된 해법의 전개 과정에서 묶음집합을 생성하고 선택하는 방안을 몇 가지 제안 하였다.

그리고 제한적이긴 하지만 각각의 방법에 대한 상대적인 비교를 시도 하였다. 이러한 방안들을 좀더 정교하게 개선해 나간다면 보다 효율적인 해법과 최적해를 제공할 수 있는 해법의 개발이 가능할 것이다. 특히 공정시간 구간탄력성과 납기가 함께 반영된 측정식을 개발하면 해법의 접근성을 높일 수 있을 것이다.

또한 SPT, 수정납기, 탄력성 규칙을 적용시킬 때 단위 작업들의 공정시간을 상한과 하한 값 사이에서 어떠한 값으로 고정시킬 것인지에 대한 연구가 필요하다. 이들 두 값의 차이가 클수록 묶음집합의 경우 수가 증가하고' 따라서 의사결정에 미치는 영향이 증대될 것으로 볼 수 있기 때문이다.

참고문헌

[1] 오세호, "구간 공정 시간을 갖는 작업들의 일괄처리 일정계획문제", 산업경영시스템학회지, 제27권 제1호, pp,47-50, 2004.

[2] Ahmed, I., and W.W. Fisher, "Due Date Assignment, Job Order Release and Sequencing," *Decision Sciences*, 23, pp.633-644, 1992.

[3] K. R. Baker, *Elements of Sequencing and Scheduling*, Amos Tuck School of Business Administration, Dartmouth College, Hanover., 1998.

[4] V. Chandru, C.Y. Lee and R. Uzsoy, "Minimizing Total Completion Time on Batch Processing Machine," *International Journal of Production Research* 31, pp.2097-2122, 1993.

[5] G. R. Dobson and R. S. Nambimadom, "The Batch Loading and Scheduling Problem," *Oper. Res.*, 49, pp.52-65, 2001.

[6] D. S. Hochbaum and D. Landy, "Scheduling semiconductor burn-in operations to minimize total flow-time," *Oper. Res.*, 45, pp.874-885, 1997.

[7] C. Y. Lee, R. Uzsoy and L. A. Martin-Vega, "Efficient algorithms for scheduling semiconductor burn-in operations," *Oper. Res.*, 40, pp.764-775, 1992.