공정 교체 시간을 고려한 배치작업의 일정계획

김주일* · 이영훈**

Batch Scheduling of Incompatible Job Families with Sequence Independent Setup Times

Jooil Kim* · Young Hoon Lee**

■ Abstract ■

The problem of minimizing total tardiness on a batch processing machine with incompatible job families when there are sequence independent setup times between families is studied where all jobs of the same family have identical processing times and jobs of different families cannot be processed together. A batch processing machine can process a number of jobs, within a maximal batch size, simultaneously as a batch. The processing time required of each batch is equal to the one of jobs. A dynamic programming algorithm which gives the optimal solution, and several heuristics are presented. Performance of simple dispatching rules based on due dates are compared, and the best of them is used as an initial solution for the decomposition algorithm, which is shown to give good schedules in relatively short computational time.

Keyword : Batch Scheduling, Job Family, Setup Time, Dynamic Programming

1. 서 론

배치 일정계획은 제조공장의 설비가 자동화 및 고도화되면서 동시에 여러 개의 작업을 진행할 수 있는 환경에서 배치의 구성과정을 일정계획에 포함시켜 고려하는 경우를 말한다. 배치로 구성될 수 있는 작업은 동일한 성격을 가지고 있는 경우가 대부분이며 때로는 모든 작업이 동일한 특성을 가지

논문접수일 : 2000년 11월 30일 논문게재확정일 : 2001년 4월 31일
* 서울시 서대문구 신촌동 134 연세대학교 컴퓨터과학 · 산업시스템공학과
** 서울시 서대문구 신촌동 134 연세대학교 기계전자공학부 정보산업공학전공

고 있어 일정 수로 구성된 하나의 배치로 구성될 수 있는 경우도 있다. 대부분의 생산환경에서 배치의 최대크기는 정해져 있어 그 이하의 수로 구성된 작업들이 하나의 배치를 구성하여 함께 작업이 진행된다. 작업에 소요되는 작업시간은 개별 작업의 소요시간의 합보다 작기 때문에 배치작업의 이점이 있다.

그룹의 구분이 없이 모든 작업이 배치로 구성될 수 있는 경우(compatible) 구성된 작업 중 작업시간이 가장 긴 작업의 소요시간이 배치작업 소요시간이 되는 경우가 일반적이다. 본 논문은 공정의 특성으로 인해 작업이 동일한 속성을 가지는 그룹으로 구분되어지고 동일한 그룹내의 작업끼리만 배치로 구성할 수 있는 경우(incompatible)의 일정계획에 관한 연구이다. 동일 그룹내의 작업은 작업시간이 동일하고 배치로 구성된 작업이 진행될 때 수에 관계없이 하나의 작업에 소요되는 시간과 동일한 작업시간이 소요된다. 하나의 배치 작업이 끝나고 그룹이 다른 작업들로 구성된 배치의 작업이 진행되면 교체시간이 필요하며 이 시간은 그룹의 종류에 관계없이 일정하다고 가정한다. 일반적으로 그룹의 종류가 서로 다른 배치의 작업이 연속될 때 필요한 교체시간은 앞뒤의 배치가 속한 그룹의 종류에 따라 다른 것이 보통이나 교체시간의 차이가 심하지 않아 무시될 수 있거나 또는 교체과정 자체가 일련의 프로세스로 진행되어 일정할 때 적용할 수 있는 모델에 대한 연구이다. 교체시간 자체는 작업시간 대비 무시할 수 없을 정도로 일정시간 이상이고 얼마나 자주 교체시간을 발생시켜 주어진 목적함수를 최적화할 것인가가 주요 문제인 경우에 해당된다. 각 작업에 필요한 시간은 그룹별로 일정하지만 납기는 작업별로 주어지며 목적함수는 각 작업의 종료시간과 납기의 차이로 정의되어지는 지연시간의 총합을 최소화하는 것이다.

이와 같은 형태의 작업환경은 반도체 제조공정에 찾을 수 있다. 반도체 제조공정은 제조환경에 따른 복잡한 공정으로 인하여 정밀한 일정관리가 필요하다. 특히 웨이퍼 Fabrication공정은 수백가지 이상의 공정으로 구성되어 반도체 제조공정 중에서 가장 복잡하게 구성되어 있으며 공정마다 각기 다른 작업시간을 갖고 있다. 공정의 종류가 변할 때는 설비세척과 공구의 교체로 인하여 일정시간의 교체시간이 발생한다. 웨이퍼 Fabrication공정 중 확산공정은 일정 수의 웨이퍼가 실린더 리액터와 함께 들어가 밀폐되어져 캐리어가스와 함께 가열된다. 웨이퍼는 자재특성에 따른 표준 로트 크기가 주어져 있으며 일반적으로 6개에서 12개의 로트로 구성된다. 공정의 화학적인 특성으로 하나의 배치를 구성하는 작업들은 동일한 작업시간을 가지며 이들 작업들은 동일한 작업시간을 이루는 작업들이 하나의 그룹을 형성한다. 이 공정의 효율적인 일정계획이 필요한 이유는 이들의 작업시간이 일반적으로 다른 공정에 비하여 긴 특성 때문이다. 또한 화학적인 공정의 특성상 종류가 다른 그룹의 작업이 진행될 때는 설비의 세척 및 작업사양의 조정 등으로 교체시간이 필요하고 이로 인한 작업가동률의 손실을 최소화할 필요가 있다. 반도체 산업은 대체적으로 적은 재공으로 짧은 시간 내에 납기를 맞추는 형태의 제조환경이 요구된다. 생산라인에 재공이 많으면 연속하여 동일 그룹에 속하는 작업들로 구성된 배치로 작업이 이루어지도록 일정관리를 운영하여 준비교체의 손실을 줄일 수 있으나 재공이 적으면 잦은 준비교체가 불가피하게 되고 이를 최소화하는 것이 생산라인의 생산성과 직결된다. 특히 반도체 생산에 있어서 버메모리 계열의 제품생산이 증가하고 있어 제품별 납기가 있고 이에 대한 지연도가 관리되고 있을 때 준비교체시간을 무시하고 일정계획을 작성할 경우 제품별 납기만을 고려하여 빈번한 준비교체를 유발하게 되어 이로 인한 손실이 커지게 된다. 반면 설비의 가동률을 높이고 준비교체시간만을 줄이고자 하는 일정계획은 제품별 납기관리에 문제점을 발생시킬 수 있다. 본 논문은 전체 납기지연시간을 최소화하기위한 일정계획을 작성함으로 납기와 준비교체시간의 적절한 통합관리에 대한 해법을 제시하는데 목적이 있다.

배치 공정의 일정계획은 작업을 어떻게 배치로 구성할 것인가와 구성되어진 배치들을 어떻게 나열할 것인가의 두 가지의 의사결정을 포함하며 서로 밀접하게 연결되어 있다. 정규 평가지수(regular measure of performances)를 사용하는 문제에서 그룹이 나누어지는 작업들로 구성된 배치 일정계획의 문제는 위의 두 가지 의사결정이 단계적으로 이루어질 수 있다(Uzsoy[17]). 본 논문에서는 단일 배치 공정기계에서 고정된 배치크기와 고정된 작업 그룹을 가지고 있으며 배치간 교체시간이 있는 문제에 대해 최적해를 계산할 수 있는 동적계획법을 제시하였다. 본 논문에서 제시된 해법은 작업의 수에 대해 다항식 시간(polynomial time)내에 해를 구할 수 있으나 그룹의 수가 증가할수록 계산에 소요되는 시간이 급격히 증가하므로 실제 작업현장에 적용하기에는 현실적이지 못하여 동적인 작업환경에서도 적용할 수 있는 발견적 규칙을 제시하였고 각각의 성능비교를 실시하였다.

전체 가중지연시간을 최소화하는 문제는 여러 연구자에 의해 광범위하게 이루어졌다. Lawler[7]는 이 문제에 관해 유사다항식 시간의 동적계획법을 제시하였고 Schrage and Baker[16]와 Potts and Wassenhove[14]는 동적계획법과 분지한계법을 이용한 최적화해법을 개발하였으나 30개 이상의 작업들에 대하여서는 해를 구하지 못하였다. Du and Leung[5]는 일반적인 작업환경에서 이 문제는 NP-hard문제임을 증명하였다. 따라서 계산시간이 문제의 크기가 커짐에 따라 비약적으로 증가함으로 인해 분지한계법과 동적계획법은 문제의 크기가 적은 경우에 이용되었다.

전체 가중시간을 최소화 하는 문제에서 크기가 큰 문제를 풀기 위하여 발견적 기법에 관한 연구가 많이 이루어져왔다. Morton and Rachamadugu[12]은 작업들을 작업시간과 납기를 같이 고려한 우선순위를 정하여 실행하는 발견적 기법을 제시하였다. Vepsalainen and Morton[18]은 빠른 시간 내에 효율적인 실행가능해를 구할 수 있도록 우선순위를 정하여 실행순서를 결정하는 발견적 기법인 ATC(Apparent Tardiness Cost) 규칙을 소개하였다. Chambers et al.[2]는 문제의 크기가 큰 문제를 작은 크기로 분할한 후 최적화 기법을 적용하여 최적해에 가깝게 문제를 푸는 분할기법(DH : Decomposition Heuristic)을 개발하였고 WSPT와 EDD 규칙에 배경을 둔 발견적 기법과 성능비교를 하여 분할기법이 적정한 시간 내에 최적 스케줄에 근접한 해를 산출한다는 것을 보여주었다.

전체 가중지연시간을 최소화하는 문제에서 준비교체시간을 고려한 문제에 관한 연구는 비교적 최근에 이루어졌다. Lee et al.[9]는 이전 공정에 따라 변화하는 교체시간을 가지는 단일 공정 작업환경에서 전체 가중지연시간을 최소화하는 문제에 관해 ATC 규칙을 적용하여 효율적인 스케줄을 찾는 발견적 기법을 제시하였다. Park et al.[13]은 위의 문제에서 신경망 기법을 적용하여 보다 효율적인 스케줄을 찾을 수 있는 발견적 기법을 제시하였다. 배치 작업이 이루어지는 환경 가운데 그룹의 구분이 없는 경우 즉, 어느 작업도 서로 배치로 작업이 가능한 경우는 단일공정과는 달리 배치의 공정시간이 배치 내의 가장 긴 공정시간을 가지는 작업의 공정시간과 동일한 경우에 대한 연구는 Lee et al.[8], Chandru et al.[3, 4]등에서 찾을 수 있다. 본 논문에서 다루고자 하는 문제는 단일 배치공정 즉, 그룹이 나누어지고 그룹 내의 공정은 동일한 공정시간을 가지며 동일 그룹에 속하는 공정으로만 배치가 구성될 수 있다는 가정하의 일정계획에 대한 것이다. Uzsoy[17]는 고정된 그룹 종류와 고정된 배치크기를 가진 상황에 관하여 최대지연시간과 가중완료시간의 합을 최소화시키는 상황에 효율적인 알고리즘을 제시하였다. Mehta and Uzsoy[11]는 위의 동일한 환경하에서 전체지연시간의 합을 최소화하는 문제에 대해 다항식 계산시간 안에 실행될 수 있는 동적계획법을 제시하였고 ATC기법과 분할기법의 두 가지 발견적 규칙을 제시하여 비교하였다. 반도체 제조 환경에서 납기에 관한 문제에서 Kim et al.[6]은 기존 발견적 규칙과 반도체 제조의 특성을 고려한 새로운 발견적 규칙을 제시

하여 비교 실험하여 반도체의 제조특성을 고려한 규칙이 보다 좋은 스케줄을 구할 수 있다는 것을 보여주고 있으며 박종관 외[1]는 주문변화를 고려한 반도체 일정계획에서 여러 가지 발견적 기법의 성능을 시뮬레이션을 이용하여 비교, 평가하였다. 본 논문은 2장에서 문제에 대한 정의와 동적계획법을 이용한 해법과 발견적 기법을 소개하고 3장에서 실험 및 결과, 4장에서 결론을 정리하는 형태로 구성되어 있다.

2. 문제 정의 및 해법

다루고자 하는 문제는 모두 n 개의 작업을 대상으로 한다. 각 작업들은 m 개의 그룹으로 나누어져 있고 각각의 그룹 j 는 n_j 개의 작업이 존재하며 이들의 합 즉, 작업 총수는 n 이다. 각각의 공정은 최대 B 개의 배치단위로 동시에 작업이 진행되어진다. 각각의 작업은 모두 납기를 가지고 있으며 j 그룹의 i 번째 작업의 납기는 d_{ij} 로 표시한다. 그룹 j 의 k 번째 배치 B_{kj} 에는 최대 B 개의 작업을 할당할 수 있다. j 그룹의 i 번째 작업의 작업완료시간은 C_{ij} 로, 지연시간은 T_{ij} 로 표시하며 목적함수인 전체지연시간은 그룹 j 의 총지연시간 T_j 의 합으로 구할 수 있다. 목적함수를 수식으로 표현하면 아래와 같다.

$$T_{ij} = \max(C_{ij} - d_{ij}, 0)$$
$$j = 1, \cdots, m,\ i = 1, \cdots, n_j.$$
$$\sum T_j = \sum_{j=1}^{m} \sum_{i=1}^{n_j} T_{ij}$$

Mehta and Uzsoy[11]는 그룹의 종류 m 과 배치의 크기 B가 고정되어 있지 않고 준비교체가 없는 경우 단일 공정 배치 작업의 스케줄링에서 전체지연시간을 최소화하는 문제는 *NP-hard* 임을 증명하였다. 본 논문에서 다루는 준비교체가 있는 문제는 그룹의 종류와 배치의 크기가 결정되어 있다고 가정한다. 실제로 생산 현장에서는 그룹의 종류는 일정한 범위 내에 있고 배치의 크기는 설비구입당시에 결정되어 새로운 설비로 교체되지 않는 한 고정되어 있는 것이 상례이다. 또한 준비교체시간은 배치가 속한 그룹의 종류에 따라 다르지만 실제로 반도체 확산공정의 경우 시간의 차이가 크지않아 무시할 수 있는 수준이어서 항상 일정하다고 가정한다. 다음의 정리는 Mehta and Uzsoy[11]가 준비교체시간이 없는 경우에 성립함을 증명한 것으로 배치의 종류가 바뀔 경우에 준비교체시간이 발생하더라도 동일하게 성립함을 증명할 수 있다.

(정리 1) 배치를 구성할 때 마지막 배치를 제외한 모든 배치는 배치 최대크기만큼의 작업들로 구성된 최적 스케줄이 존재한다.

정리 1은 특별한 증명이 필요치 않은 직관적 결과이다. 즉 마지막 배치가 아닌 배치에서 최대배치크기보다 작은 작업수로 구성된 배치가 존재할 경우 후에 오는 동일 그룹의 배치에 속한 작업을 현재의 배치로 옮김으로써 모든 작업의 종료시간은 같거나 작아지기 때문에 중간에 배치최대크기 만큼의 작업들로 구성하지 않을 이유가 없다. 이 정리는 단순하지만 동적계획법의 구성을 가능케 하는 중요한 정리이다.

(정리 2) 동일 그룹에 속하는 작업들간에는 EDD (Earliest Due Date)의 순서대로 작업이 이루어지는 최적 스케줄이 존재한다. 즉, 동일배치에 속해있을 때 뿐 아니라 서로 다른 배치에 속해있어도 아래와 같은 수식이 성립한다.

$$\max_{ij \in B_{kj}} \{d_{ij}\} \le \min_{ij \in B_{(k+1),j}} \{d_{ij}\} \quad \forall i, j, k$$

정리 2는 Mehta and Uzsoy[11]에서 준비교체시간이 없는 경우에 증명한 과정을 동일하게 사용할 수 있는데 이는 동일 그룹 내에 속한 작업의 배치간 이동으로 준비교체가 추가로 발생하지 않기 때문이다. 위의 두 가지 결과를 통하여 최적 스케줄은 마지막 배치를 제외하고는 배치의 크기를 모두 채우고 동일한 그룹내의 작업들은 납기에 따라 순서적으로 이루어진 배치들로 구성된다는 것을 알 수 있

으며 이를 이용하여 최적 스케줄을 구할 수 있는 동적계획법모형을 구할 수 있게 된다. 따라서 그룹 j의 배치의 총수 k_j는 n_j를 B로 나눈 값의 윗 정수로 정할 수 있다. 즉, $\lceil A \rceil$를 A보다 크거나 같은 최소 정수라 할 때 $k_j = \lceil \frac{n_j}{B} \rceil$로 나타낼 수 있다.

2.1 동적계획법 모형

각 그룹의 작업들은 납기가 작은 작업부터 순서대로 최대크기에 맞게 채워서 배치를 구성할 수 있고 그룹 j에는 총 k_j개의 배치가 존재하게 된다. 작업에 대한 순서와 배치를 결정하는 과정에서 s는 공정교체시간이며 T는 현재까지 이루어진 그룹 교체 회수이고 q는 직전에 할당되어진 작업의 그룹을 나타낸다고 가정할 때 $f(w_1, w_2, \cdots, w_m, T, q)$를 각각의 그룹에 대해 그룹 j의 첫번째부터 w_j개의 배치까지 스케줄링이 이루어졌을 때 납기 지연시간의 합의 최소값이라고 정의한다. 우리가 최종적으로 구하고자 하는 값은 $f(k_1, k_2, \cdots, k_m, T, q)$이며 T와 q는 고정되어있지 않고 취할 수 있는 모든 범위에 대해 최소값을 가지는 경우에 해당한다. 동적계획법을 이용한 모형은 다음과 같다.

기저조건 : $f(0, 0, \cdots, 0, 0, *) = 0$,

$f(w_1, w_2, \cdots, w_m, T, q) = \infty$,

$w_j > k_j$ 또는 $w_j < 0$, $T < 0$, $T > \sum k_j - 1$ 일 경우

순환방정식 : $f(w_1, \cdots, w_2, \cdots, w_m, T, q) =$

$$\min_q \{f(w_1, \cdots, w_q - 1, \cdots, w_m, T^*, q^*) + \sum_{iq \in B_{u,g}} \max(0, C_{iq} - d_{iq})\},$$

여기서 $q = q^*$ 이면 $T^* = T$, 그렇지 않으면 $T^* = T - 1$.

작업완료시간 C_{iq}는 아래와 같이 계산한다.

$$C_{iq} = \sum_{k=1}^{m} w_k p_k + s \cdot T$$

위 문제의 복잡도(Complexity)는 다음과 같이 계산할 수 있다. 각 그룹마다 k_j개의 배치가 가능하므로 상태변수 0을 포함하여 총 $(1+k_j)$개의 상태변수가 있고 그룹의 교체는 최대 $(\sum k_j - 1)$이며, 직전 배치의 종류를 나타내는 q는 m개 만큼의 상태변수를 갖게 된다. 따라서 전체의 복잡도는

$$m^2 (\sum_{j=1}^{m} k_j - 1) \prod_{j=1}^{m} (k_j + 1)$$

$$= m^2 (\sum_{j=1}^{m} \lceil \frac{n_j}{B} \rceil - 1) \prod_{j=1}^{m} (\lceil \frac{n_j}{B} \rceil + 1)$$

$$\leq m^2 (\frac{n}{B} + m - 1)(2 + \frac{n}{mB})^m$$

로서 그룹의 수 m과 배치최대크기 B가 정해져 있을 때 다항식 시간을 가지는 알고리즘이다. 그러나 실제로 그룹의 수가 5가 넘는 경우 위 동적계획법 모형을 통해 해를 구하여 현장에 적용하기에는 계산시간상의 한계가 있다. 현장 적용성을 높이기 위해 빠른 시간 내에 준 최적해를 구할 수 있는 발견적 기법을 다음과 같이 제시한다.

2.2 우선순위 규칙

납기 지연을 최소화하는 문제에서 일반적으로 많이 사용되는 방법은 각 배치의 우선순위를 정하여 작업을 진행하는 것이다. 우선순위를 이용하여 스케줄을 결정하는 방법은 빠른 시간에 효율적인 스케줄을 제공할 수 있으나 전체적인 스케줄에서 최적 스케줄과 큰 차이가 발생할 수 있는 가능성이 있다. 납기 위주로 고안된 우선순위 규칙 3가지와 교체시간을 고려한 복합적인 우선순위 규칙 3가지를 다음과 같이 제시한다.

납기 지연 문제에서 가장 많이 사용되는 우선순위는 EDD 규칙으로서 최소 납기를 가진 작업을 가장 먼저 작업하는 방법이다. 배치가 형성되는 경우는 배치 내 여러 작업의 납기가 다르기 때문에 이러한 상황을 고려하여 배치 내의 평균 납기가 가장 적은 배치를 우선적으로 선택하는 EADD (Earliest Average Due Date first) 규칙과 가장 급한 납기를 가진 배치를 우선적으로 선택하는

EMDD(Earliest Minimum Due Date first) 규칙으로 변형하여 사용할 수 있다. EADD와 EMDD의 할당하고자 하는 그룹 j의 k번째 배치의 우선순위 인덱스인 $A_{kj}(t)$와 $B_{kj}(t)$는 다음과 같이 구한다.

$$A_{kj}(t) = \overline{d}_{kj} - t - s^*,\ B_{kj}(t) = \min(d_{kj}) - t - s^*$$

여기서 $\overline{d}_{kj}$는 배치 내 평균 납기를 의미하며 $\min(d_{kj})$는 배치 내 작업들의 납기 중에서 최소 납기를 의미한다. t는 스케줄링 하고자 하는 현 시점을 의미하며 s^*는 직전에 할당된 작업과 동일한 배치는 0의 값을 가지고 이전 배치와 다른 배치에는 교체시간 s의 값을 가지게 된다.

납기와 공정 시간을 동시에 고려하여 우선 순위를 할당하는 것이 MSLACK(Minimum Slack) 규칙으로서 배치의 평균 납기에 대한 배치의 여유시간을 계산하여 최소의 여유시간을 가지는 배치를 선택하여 할당하며 그룹 j의 k번째 배치에 대한 우선순위 인덱스 $S_{kj}(t)$는 다음과 같이 구해진다.

$$S_{kj}(t) = \overline{d}_{kj} - s^* - p_j - t.$$

위의 3가지 우선순위 규칙은 기본적으로 납기 위주의 규칙으로서 납기가 작은 배치일수록 긴급하다는 가정을 반영한 것이며 MSLACK의 경우 여유시간에 대한 비용이 선형적으로 증가한다고 보고 긴급한 배치 우선 할당의 형식을 취하고 있다.

GH(Greedy Heuristic) 규칙은 스케줄링이 가능한 배치들에 대해 스케줄링 되었을 때의 납기 지연시간을 계산하여 납기 지연시간이 가장 큰 배치를 선택하는 방법이다. 즉 현재 스케줄링 되었을 때 납기 지연시간이 가장 큰 배치는 후에 스케줄링 되었을 때 납기 지연시간이 공정시간만큼 후에 오는 모든 작업에 대해 증가할 가능성이 높아 우선적으로 할당되도록 하는 근시안적인 우선순위 규칙이다.

ATC(Apparent Tardiness Cost) 규칙은 공정이 가지고 있는 여유시간에 대해 긴급도의 비용이 지수적으로 증가한다고 가정하고 이를 반영한 우선순위 규칙이다. 즉, 여유시간의 값이 공정시간에 비해 클 때에는 인덱스의 증가가 완만하지만 납기에 가까워져서 여유시간이 작아지면 인덱스의 값이 급속히 증가하도록 하여 기존 MSLACK 규칙에서의 선형적 증가와 차별을 두고 있다. 이 방법은 Vepsalainen and Morton[18]에 의해 전체 지연시간을 최소화하는 문제에 탁월한 성능을 보이고 있음이 발표되었고 Mehta and Uzsoy[11]는 이를 배치공정의 일정계획에 적용하였다. Raman et al.[15]은 교체시간이 발생하는 경우에 대해 교체시간을 공정시간의 일부로 생각하여 반영한 규칙을 제시하였고, Lee et al.[9]와 Lee and Pinedo[10]는 교체시간과 여유시간에 대해 각각 지수함수를 적용한 ATCS(Apparent Tardiness Cost with Setup) 규칙을 개발, 가중지연시간의 합을 최소화하는 문제에 대해 성능의 우수함을 보였다. 본 논문에서는 Raman et al.[15]의 규칙과 Lee et al.[9]의 규칙을 배치상황에 맞게 변형한 BATCS1과 BATCS2를 아래와 같이 제시하고 스케줄링이 가능한 그룹 j의 l번째 배치 B_{lj}에 대하여 다음의 인덱스를 계산한다. 여기에서 p_j는 j그룹의 작업시간이며 $\overline{p}$는 모든 그룹의 작업시간의 평균이다.

$$BATCS1(B_{lj}, Q) = \frac{1}{p_j}\exp\left\{-\frac{\sum_{lj\in B\in ij}\max(d_{ij} - s^* - p_j - t, 0)}{k(\overline{p} + s)}\right\},$$

$$BATCS2(B_{lj}, Q) = \frac{1}{p_j}\exp\left\{-\frac{\sum_{lj\in B\in ij}\max(d_{ij} - p_j - t, 0)}{k_1\overline{p}}\right\}\exp\left(\frac{s^*}{k_2 s}\right).$$

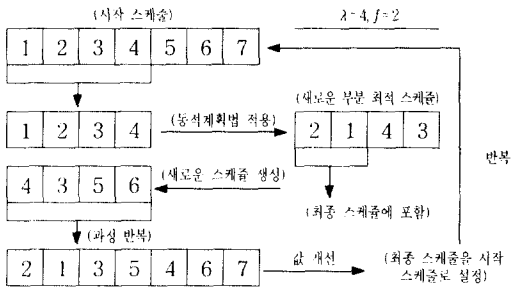
BATCS1은 모수로서 k 하나를 가지며 BATCS2는 k_1, k_2 두 가지의 모수를 사용한다. s는 교체시간이며 s^*는 직전에 할당되어진 작업의 그룹 Q와 현재 할당할 대상인 작업의 그룹 j가 동일하다면 0의 값을 가지고 다르다면 교체시간 s의 값을 가진다. BATCS1과 BATCS2 각각에 대해 인덱스의 값을 구하고 최대의 값을 가지는 배치를 할당하여 우선순위 규칙을 적용하였다. 모수값인 k, k_1, k_2는

0.2에서 2까지의 값을 0.2단위로 구분하여 이 모든 경우의 총지연시간을 구한 후 최소 납기지연시간을 선택한 k, k_1, k_2를 BEST BATCS로 사용하였다.

2.3 분할기법(Decomposition Heuristic)

분할기법은 전체 문제를 부분적으로 나누어 각각의 부분적인 최적값을 여러 번 구하여 이를 조정함으로써 분할된 부분문제의 최적해를 연결하여 한정된 시간 안에 최적해에 가까운 값을 구하고자 하는 방법이다. 스케줄링 문제 중에서는 가중지연시간에 관한 문제에서 빠른 시간 내에 최적해에 가까운 값을 구하고자 하는 문제에 사용되었다. [2, 11, 17] 전체문제에서 λ개 만큼의 배치만으로 부분최적값을 구한 후 이 중 이 최적해의 스케줄 중에 f개의 배치를 전체 스케줄에 할당을 한 후 다시 남은 배치들로 모든 작업의 할당이 끝날 때까지 반복하는 과정을 실시한다. 분할기법을 적용하여 구해진 스케줄이 시작 스케줄보다 좋은 결과값을 가지고 있으면 현재의 스케줄을 기준으로 분할기법을 다시 반복적용하고 그렇지 않으면 최종스케줄로 확정한다.

분할기법의 전개과정의 예를 순서에 따라 그림으로 나타내면 [그림 1]과 같다. 전체 작업의 배치들이 7개로 이루어져 있고 시작 스케줄의 작업순서가 번호순서와 동일하게 이루어져 있다고 가정하고 $\lambda=4, f=2$의 모수를 가지고 분할기법을 적용하면 먼저 시작 스케줄에서 처음부터 λ개의 작업들, 즉 1,2,3,4의 작업을 대상으로 동적계획법을 적용하여 부분 최적 스케줄을 구한다. 부분 최적 스케줄이 2,1,4,3의 작업순서로 구해졌다면 f개의 작업, 즉 2,1번 작업을 최종 스케줄로서 고정하고 4,3,5,6의 4개의 부분 스케줄에 대해서 동적계획법을 적용하여 부분 최적 스케줄을 구한다. 이러한 과정은 7개의 작업들이 모두 할당되어질 때까지 반복되어진다. 모든 작업들이 할당된 최종 스케줄의 지연도값을 시작 스케줄의 지연도값과 비교하여 시작 스케줄보다 값이 개선되었다면 최종 스케줄을 다시 시작 스케줄로 만들어 다시 분할기법을 적용한다. 그러나 값이 개선되지 않았다면 시작 스케줄을 문제의 최종 스케줄로 결정하고 분할기법을 마친다.



[그림 1] 분할기법 전개과정

분할기법 알고리즘 진행과정

Step1 : BEST BATCS를 이용하여 시작 스케줄로 결정하며 모수 λ, f를 설정한다.

Step2 : $k=\min\{N, \lambda\}$, $j=0$, 여기서 N은 스케줄이 이루어질 배치의 전체수를 말하며 최종 스케줄은 공집합이다.

Step3 : $B_{[i]}$는 시작 스케줄의 i번째 배치라고 할 때 $\{B_{[1]},\cdots, B_{[k]}\}$로 이루어진 부분문제 P를 구성한다. 부분문제P를 동적계획법을 이용하여 부분 최적 스케줄을 구한다. $\beta_{[i]}$는 부분 최적 스케줄의 i번째 배치를 가리킨다.

Step4 : $\beta_{[1]}$부터 $\beta_{[x]}$ 까지를 최종 스케줄의 $j+1$부터 $j+x$까지 할당한다. ($x=\min\{|P|, f\}$) 그리고 $j=j+x$, $P=P/\{\beta_{[1]},\cdots, \beta_{[x]}\} \cup \{\beta_{[k+1]}, \cdots, \beta_{[q]}\}$로 설정한다. ($q=\min\{k+f, N\}$, $k=q$)

Step5 : j가 N 보다 작다면 Step 3부터 다시 반복하며 그렇지 않다면 Step 6를 수행한다.

Step6 : 최종 스케줄의 전체지연시간이 시작 스케줄보다 작다면 시작 스케줄을 최종 스케줄로 대치하고 Step 2 부터 반복한다. 그렇지 않다면 시작 스케줄을 문제의 최종 스케줄로 결정한다.

동적계획법은 문제의 크기가 커질수록 상태변수의 수가 급격히 늘어나 계산시간이 크게 증가하는데 비해 위 규칙은 한계가 정해진 상태변수를 사용함으로써 문제의 크기가 커질수록 동적계획법보다 빠른 시간 내에 해를 구할 수 있다. 또한 이 규칙에서 계산의 정확성과 계산시간은 각각의 부분 동적계획법을 적용하는 λ와 f값의 크기에 따라 달라진다. 즉 λ를 크게 하면 최적해에 가까운 값을 산출하지만 계산시간이 증가하게 되고 f값을 크게 하면 작은 분할회수로 빠르게 값을 구할 수 있지만 최적해와의 차이는 커지게 된다.

일반적인 분할기법은 λ, f를 고정하여 사용한다. 공정 교체시간이 존재하고 그룹이 나누어져 있는 문제에서는 동일한 그룹으로 이루어진 여러 개의 단위의 스케줄 단위가 존재하게 된다. 또한 동일한 그룹의 작업들은 같이 작업이 이루어지려는 성질이 있으며 이러한 성질을 이용하여 λ, f를 고정시키지 않는 유연 분할기법(FDH : Flexible Decomposition Heuristic)을 고려할 수 있다. 여러 가지 스케줄 단위 중에서 처음의 스케줄 단위와 다음의 동일한 그룹을 가진 스케줄 단위 사이의 모든 배치를 λ로 설정한다. 이 λ를 이용하여 동적 계획법을 적용하여 부분 최적해를 구하고 f값으로는 이 부분 최적해의 최초 스케줄 단위를 설정한다. 이 방법은 λ와 f가 정해져 있지 않고 스케줄이 이루어질 때마다 계속 변화하는 특성을 가지고 있다.

2.4 발견적 기법의 예제

본 연구에서 제시한 발견적 기법에 대한 예제를 다음과 같이 구성하여 풀이과정을 예시하고자 한다. 모든 작업은 3개의 그룹으로 구성되어 있으며 작업은 각 그룹 당 10개씩 모두 30개의 작업이 존재한다. 배치는 최대 4개의 작업으로 구성되어진다. 첫 번째 그룹의 작업시간은 2, 두 번째 그룹의 작업시간은 4, 마지막 그룹의 작업시간은 3이며 교체시간은 5이다. 각 작업의 납기와 납기를 기준으로 한 배치의 구성은 다음과 같다. 여기서 각 배치에 표시된 수는 각 작업의 납기를 의미한다.

그룹 1 : B_{11} = [2,4,6,7], B_{21} = [9,11,12,13], B_{31} = [17,18]
그룹 2 : B_{12} = [7,8,9,10], B_{22} = [10,12,12,14],
B_{32} = [15,16]
그룹 3 : B_{13} = [4,6,7,7], B_{23} = [9,10,16,16], B_{33} = [18,19]

분할기법을 적용하기 위해서는 먼저 BATCS를 이용하여 초기 스케줄을 구하여야 한다. 본 예제에서는 BATCS1를 사용하여 할당될 수 있는 배치들의 인덱스값을 결정하고 이중 값이 가장 큰 인덱스를 가진 배치를 스케줄에 포함한다. 마지막 배치가 할당되어질 때까지 순차적으로 이루어지며 인덱스값과 이에 의하여 선택된 배치 스케줄의 구성은 <표 1>과 같으며 사용되어진 모수 k의 값은 1.0이다. ( )안의 값은 현재 선택되어진 그룹별 배치의 총 수이다.

BATCS1를 이용하여 선택되어진 배치는 B_{11},

〈표 1〉 BATCS1의 적용과정

적용순서	그룹 1	그룹 2	그룹 3	최대 인덱스값	선택된 그룹 및 배치
1	0.0128(0)	0.0006(0)	0.0061(0)	0.0128	1 : B_{11}
2	0.0294(1)	0.2116(0)	0.3333(0)	0.3333	3 : B_{13}
3	0.5000(1)	0.2500(0)	0.1711(1)	0.5000	1 : B_{21}
4	0.5000(2)	0.2500(0)	0.3333(1)	0.5000	1 : B_{31}
5	(3)	0.2500(0)	0.3333(1)	0.3333	3 : B_{23}
6	(3)	0.2500(0)	0.3333(2)	0.3333	3 : B_{33}
7	(3)	0.2500(0)	(3)	0.2500	2 : B_{12}
8	(3)	0.2500(1)	(3)	0.2500	2 : B_{22}
9	(3)	0.2500(2)	(3)	0.2500	2 : B_{32}

〈표 2〉 부분최적해의 동적계획법 적용과정

Stage1	$f(1,0,0,0,1) = f(0,0,0,0,*) + 0 = 0$,	$f(0,1,0,0,3) = f(0,0,0,0,*) + 0 = 0$
Stage2	$f(2,0,0,0,1) = f(1,0,0,0,1) + 0 = 0$,	$f(1,0,1,1,3) = f(1,0,0,0,1) + 1 = 16$
	$f(1,0,1,1,1) = f(0,1,0,0,3) + 21 = 21$,	$f(0,0,2,0,3) = f(0,1,0,0,3) + 0 = 0$
Stage3	$f(3,0,0,0,1) = f(2,0,0,0,1) + 0 = 0$,	$f(2,0,1,1,3) = f(2,0,0,0,1) + 12 = 12$
	$f(2,0,1,2,1) = f(1,0,1,1,3) + 23 = 39$,	$f(1,0,2,1,3) = f(1,0,1,1,3) + 7 = 23$
	$f(2,0,1,1,1) = f(1,0,1,1,1) + 4 = 25$,	$f(1,0,2,2,3) = f(1,0,1,1,1) + 21 = 42$
	$f(1,0,2,1,1) = f(0,0,2,0,3) + 33 = 33$,	$f(0,0,3,0,3) = f(0,0,2,0,3) + 0 = 0$
Stage 4	$f(3,0,1,1,3) = f(3,0,0,0,1) + 32 = 32$,	$f(1,0,3,1,1) = f(0,0,3,0,3) + 13 = 13$
	$f(3,0,1,2,1) = \min(f(2,0,1,1,3) + 3 = 15, f(2,0,1,2,1) + 0 = 39) = 15$	
	$f(2,0,2,1,3) = f(2,0,1,1,3) + 11 = 23$,	$f(2,0,2,3,3) = f(2,0,1,2,1) + 49 = 88$
	$f(2,0,2,2,1) = \min(f(1,0,2,1,3) + 35 = 55, f(1,0,2,1,1) + 15 = 48) = 48$	
	$f(1,0,3,1,3) = f(1,0,2,1,3) + 0 = 23$,	$f(3,0,1,1,1) = f(2,0,1,1,1) + 0 = 25$
	$f(2,0,2,2,3) = f(2,0,1,1,1) + 1 = 26$,	$f(2,0,2,3,1) = f(1,0,2,2,3) + 55 = 97$
	$f(1,0,3,2,3) = \min(f(1,0,2,2,3) + 5 = 47, f(1,0,2,1,1) + 5 = 38) = 38$	
Stage5	$f(3,0,2,1,3) = f(3,0,1,1,3) + 17 = 49$,	$f(3,0,2,3,3) = f(3,0,1,2,1) + 57 = 72$
	$f(3,0,2,2,1) = \min(f(2,0,2,1,3) + 9 = 32, f(2,0,2,2,1) + 9 = 57) = 32$	
	$f(2,0,3,1,3) = f(2,0,2,1,3) + 27 = 50$,	$f(3,0,2,4,1) = f(2,0,2,3,3) + 59 = 147$
	$f(3,0,2,2,1) = f(2,0,2,2,1) + 9 = 57$	
	$f(2,0,3,3,3) = \min(f(2,0,2,2,1) + 19 = 67, f(2,0,2,3,3) + 19 = 107) = 67$	
	$f(2,0,3,2,3) = f(1,0,3,1,3) + 9 = 32$,	$f(3,0,2,2,3) = f(3,0,1,1,1) + 37 = 62$
	$f(3,0,2,3,1) = \min(f(2,0,2,2,3) + 57 = 83, f(2,0,2,3,1) + 49 = 146) = 83$	
	$f(2,0,3,2,3) = f(2,0,2,2,3) + 41 = 67$,	$f(2,0,3,4,3) = f(2,0,2,3,1) = 29 = 126$
	$f(2,0,3,3,1) = f(1,0,3,2,3) + 67 = 105$,	$f(2,0,3,1,1) = f(1,0,3,1,1) + 27 = 40$
Stage6	$f(3,0,3,1,3) = f(3,0,2,1,3) + 3 = 52$,	$f(3,0,3,3,3) = f(3,0,2,3,3) + 23 = 95$
	$f(3,0,3,3,3) = f(3,0,2,2,1) + 23 = 55$,	$f(3,0,3,2,1) = f(2,0,3,1,3) + 15 = 65$
	$f(3,0,3,5,3) = f(3,0,2,4,1) + 43 = 190$,	$f(3,0,3,4,1) = f(2,0,3,3,3) + 35 = 102$
	$f(3,0,3,3,1) = \min(f(2,0,3,2,3) + 25 = 55, f(2,0,3,3,1) + 25 = 130) = 55$	
	$f(3,0,3,2,3) = f(3,0,2,2,3) + 13 = 75$,	$f(3,0,3,4,3) = f(3,0,2,3,1) + 33 = 116$
	$f(3,0,3,5,1) = f(2,0,3,4,3) + 45 = 171$,	$f(3,0,3,1,1) = f(2,0,3,1,1) + 5 = 45$

B_{13}, B_{21}, B_{31}, B_{23}, B_{33}, B_{12}, B_{22}, B_{32}의 순서이며 DH(6,3)을 이용하여 부분최적화의 해를 구한다. 초기 스케줄에서 6개의 배치인 B_{11}, B_{13}, B_{21}, B_{31}, B_{23}, B_{33}의 배치들을 동적계획법을 이용하여 부분최적 스케줄를 구하고 이 부분최적 스케줄에서 3개의 배치를 최종 스케줄로 선택한다. 최초 6개의 부분 스케줄에 대한 동적계획법 적용과정은 <표 2>와 같다.

위의 결과에서 B_{13}, B_{23}, B_{33}, B_{11}, B_{21}, B_{31}의 부분 최적해가 나온다. 여기서 B_{13}, B_{23}, B_{33}의 3개의 배치의 순열은 고정시키고 나머지 B_{11}, B_{21}, B_{31}, B_{12}, B_{22}, B_{32}의 여섯 가지 배치로 다시 부분최적해를 구한다. 모든 배치가 나올 때까지 분할기법을 수행하며 결과값으로 나온 해는 B_{13}, B_{23}, B_{33}, B_{11}, B_{21}, B_{31}, B_{12}, B_{22}, B_{32}의 스케줄이 산출되었으며 전체지연시간은 261로 BATCS의 결과의 전체지연시간인 431에 비하여 더 좋은 결과이다. 이 문제에서는 이 결과값이 또한 전체 최적 스케줄 값과 동일하다.

3. 실험 및 결과

3.1 데이터 생성

발견적 규칙의 결과값으로 성능을 비교하기 위하여 각각의 실험에 사용되는 예제는 데이터를 <표 3>의 규칙 내에서 무작위로 생성하여 비교하였다.

사용된 모델의 구성요소는 그룹의 수(m)로 3, 4, 5, 6을 사용하였고 그룹 당 작업의 수(n_j)는 30, 40, 50, 60개의 종류에 대하여서 사용하였다. 그러나 동적계획법을 통한 최적 스케줄의 구현은 계산시간의 급격한 증가로 인하여 그룹의 수(m)로 3, 4를

〈표 3〉 실험 Data 설계

	실험 경우의 수
그룹의 작업 수(n_j)	30, 40, 50, 60
그룹의 수(m)	3, 4, 5, 6
배치크기(B)	4
그룹 작업시간(p_j)	(5, 15) 사이에서 균일하게 난수 생성
교체시간(s)	$s = \bar{p} * w$, w = 0.5, 1.0, 1.5
작업별 납기(d_{ij})	d_{ij} : τ의 확률로 ($\bar{d} - R\bar{d}$, $\bar{d}$) 범위 내에서, (1-τ)의 확률로 ($\bar{d}$, $\bar{d}$ + ($C_{max} - \bar{d}$)*R) 범위내에서 균일하게 생성 τ = 0.6, 0.7, 0.8, 0.9, R = 0.2, 0.4, 0.6, 0.8

사용하였고 그룹 당 작업의 수(n_j)는 30, 40, 50으로 제한하였다. 각 그룹의 공정시간(p_j)은 5와 15사이의 정수값 중에서 무작위로 생성하여 사용하였으며 교체시간(s)은 평균 공정시간의 0.5, 1, 1.5배(w)의 3가지 종류로 실험하였다. 배치의 크기(B)는 4로 고정하였다. 각 작업의 납기(d_{ij})의 결정은 예상되는 전체 작업의 종료시간과 연관되어 생성해야 한다. 전체 종료시간 이후에 생성되는 납기는 의미가 없으며 납기의 분포는 실험 예제 생성에 영향을 주고 공정한 실험을 위해 다양한 예제 생성이 이루어져야 한다. 준비교체시간이 존재하므로 실제로 스케줄이 생성되기 전에는 작업 종료시간을 알 수 없으나 다음과 같이 추정할 수 있다. 작업 종료시간을 C_{max}라고 할 때 이는 공정작업으로 소요되는 시간 부분과 준비교체시간으로 소요되는 부분의 합으로 표현될 수 있는데 이에 대한 추정치 $\widehat{C}_{max}$는 준비교체가 평균적으로 배치 수의 1/2만큼 발생한다고 가정한 가운데 다음과 같이 나타낼 수 있다.

$$\widehat{C}_{max} = \lceil \sum_{j=1}^{m} \frac{n_j}{B} \rceil \cdot \bar{p} + s \cdot \lceil \sum_{j=1}^{m} \frac{n_j}{2B} \rceil$$

추정 종료시간 내에 납기의 분포는 긴급도 지수와 납기범위지수 R에 의해 정의하였는데 τ와 R에 의한 납기의 생성은 다양한 납기의 경우를 대표하고 있다. 여기서 긴급도 지수 τ는 1 - ($\bar{d}/C_{max}$)로 정의하고 납기범위지수 $R = (d_{max} - d_{min})/C_{max}$로 정의하여 납기의 분포에 대한 특성을 수치화한 것으로 납기지연도 문제생성에 자주 사용된다[9, 10, 13]. τ의 값은 0과 1사이의 값을 취하고 이 값이 크면 납기의 대체적인 값이 전체 작업 소요시간에 비해 작은 경우에 해당하는 것이다. τ의 값이 1인 예제에서는 모든 작업의 납기가 0이며 τ의 값이 0인 경우는 모든 작업의 납기가 작업 종료시간과 동일한 상황을 의미한다. R의 값은 납기의 분포 범위를 나타내는 것으로 모든 작업의 납기가 동일하면 0, 납기의 최대값과 최소값의 차이가 전체 작업 종료시간과 같으면 1로서 납기에 관하여 다양한 경우의 예제를 생성하기 위하여 τ와 R의 지수가 사용된다. 본 논문에서는 τ의 값을 납기의 긴급도가 비교적 큰 (0.6~0.9) 범위에서 0.1 단위로 4가지 경우와 R의 값은 (0.2~0.8) 범위에서 0.2 단위로 4가지 경우의 데이터를 생성하기 위해 사용되었다. <표 3>과 같이 납기의 데이터를 τ의 확률로 ($\bar{d} - R\bar{d}$, $\bar{d}$) 범위 내에서, (1 - τ)의 확률로 ($\bar{d}$, $\bar{d}$ + ($C_{max} - \bar{d}$) · R) 범위내에서 균일하게 생성하면 생성된 납기는 C_{max} 내에 존재하고 τ와 R의 값을 만족하게 되어 의미 있는 데이터에 대한 실험을 할 수 있다. 이 실험은 모두 C언어를 이용하여 Pentium Ⅲ 500MHz PC에서 실험하였다.

3.2 결과분석

결과분석은 크게 두 가지로 구성되어진다. 먼저 우선순위 규칙들간의 비교를 하였으며 결과에서 가장 좋은 결과를 주는 우선순위 규칙은 분할기법의 초기 스케줄을 구성하는데 사용되어진다. 결과값의 비교에 있어서 교체시간이 있는 경우의 배치 스케줄에 대한 기존 연구를 찾을 수 없어 제안된 우선순위 규칙간의 상대비교를 통하여 가장 효율적인 규칙을 찾고자 한다. 우선순위 규칙에서의 평가지수는 생성된 데이터에 대해 지연도의 값이 가장 큰 규칙의 지연도 대비 각 규칙으로 생성된 스

〈표 4〉 우선순위 규칙들의 성능비교 (상대적 평가지수)

그룹수	작업수	EADD	EMDD	MSLACK	GH	BATCS1	BATCS2
3	30	0.9825	0.9340	0.9751	0.7212	0.5579	0.5727
3	40	0.9782	0.9433	0.9756	0.6826	0.5057	0.5230
3	50	0.9836	0.9357	0.9696	0.6780	0.5060	0.5263
3	60	0.9797	0.9465	0.9738	0.6561	0.4828	0.4937
평 균		0.9810	0.9399	0.9735	0.6845	0.5131	0.5289
4	30	0.9825	0.9698	0.9812	0.7412	0.5733	0.6006
4	40	0.9842	0.9577	0.9930	0.7173	0.5211	0.5493
4	50	0.9827	0.9706	0.9884	0.6971	0.5237	0.5377
4	60	0.9865	0.9649	0.9886	0.6870	0.5069	0.5266
평 균		0.9839	0.9658	0.9878	0.7106	0.5313	0.5536
5	30	0.9866	0.9677	0.9920	0.7345	0.5851	0.6100
5	40	0.9871	0.9734	0.9957	0.6976	0.5438	0.5642
5	50	0.9917	0.9795	0.9913	0.6935	0.5457	0.5649
5	60	0.9876	0.9764	0.9948	0.6650	0.5096	0.5255
평 균		0.9882	0.9743	0.9934	0.6976	0.5460	0.5662
6	30	0.9863	0.9799	0.9947	0.7348	0.5997	0.6169
6	40	0.9872	0.9767	0.9975	0.6938	0.5417	0.5616
6	50	0.9870	0.9789	0.9966	0.6600	0.5601	0.5767
6	60	0.9927	0.9782	0.9969	0.6222	0.5348	0.5490
평 균		0.9883	0.9785	0.9964	0.6777	0.5590	0.5760
전 체 평 균		0.9854	0.9646	0.9878	0.6926	0.5374	0.5562

케줄의 지연도의 비율을 구하였다.

(우선순위 규칙간 상대적 평가지수) =
(적용규칙에 의한 지연도의 합)/(최대 지연도의 합)

<표 4>의 결과를 통하여 살펴보면 그룹 수나 작업 수에 따라 각 우선순위 규칙의 성능은 큰 차이를 보이고 있지 않으며 남기 위주의 우선순위 규칙인 EADD, EMDD, MSLACK 규칙간에 서로 큰 차이를 보이고 있지 않다. 복합적인 우선순위 규칙인 GH, BATCS1, BATCS2의 3가지 규칙을 살펴보았을 때 BATCS 규칙이 GH보다 우수하며 BATCS 규칙 중에서 BATCS1이 약간 더 좋은 성능을 보이고 있다. 이로서 복합적인 우선순위 규칙 BATCS가 배치가 아닌 경우의 일반적인 스케줄에서와 같이 가장 좋은 성능을 보이고 있는 것을 발견할 수 있었다. 동적계획법과 발견적 기법들과의 비교에서는 우선 순위규칙에 따른 스케줄에서 가장 좋은 결과를 보여주는 BATCS1 규칙과 BATCS1 규칙의 결과를 초기 스케줄로 가지는 분할기법의 결과를 비교하였으며 <표 5>에 나타낸 평가지수는 실험 결과에서 발견적 기법의 결과와 최적 스케줄인 동적계획법과의 차이를 동적계획법의 결과값으로 나눈 값, 즉 최적값 대비 편차의 비율을 의미한다. 여기서 DH(6,3)은 λ가 6, f가 3임을 의미하며 DH(12,6), DH(24,12)와 FDH를 포함한 4가지 분할기법의 실험결과를 분석하였다. 참고로 DH(24,12)의 경우 그룹수 3, 작업수 30의 경우 배치 크기가 4이므로 그룹별로 최대 8개의 배치로 전체 24개의 배치가 형성되어 λ = 24인 분할기법을 적용하여도 최적해가 생성되므로 상대적 평가지수는 0이 된다.

(최적해대비 상대적 평가지수) =
(발견적 규칙의 지연도 - 최적해의 지연도)/
(최적해의 지연도)

〈표 5〉 동적계획법과 발견적 규칙들의 비교 (최적화 대비 상대적 평가지수)

그룹수	작업수	τ	BATCS1	DH(6,3)	DH(12,6)	DH(24,12)	FDH
3	30	0.6	1.846	0.851	0.415	0.000	0.802
3	30	0.7	0.898	0.358	0.130	0.000	0.094
3	30	0.8	0.364	0.132	0.106	0.000	0.018
3	30	0.9	0.197	0.087	0.054	0.000	0.002
	평균		0.826	0.357	0.177	0.000	0.229
3	40	0.6	2.847	1.232	0.818	0.388	0.829
3	40	0.7	1.008	0.510	0.233	0.066	0.116
3	40	0.8	0.371	0.236	0.087	0.027	0.031
3	40	0.9	0.152	0.052	0.041	0.034	0.024
	평균		1.094	0.507	0.295	0.129	0.250
3	50	0.6	3.276	1.438	0.826	0.547	1.435
3	50	0.7	1.164	0.477	0.284	0.118	0.120
3	50	0.8	0.423	0.217	0.120	0.028	0.037
3	50	0.9	0.206	0.085	0.029	0.019	0.011
	평균		1.267	0.554	0.315	0.178	0.401
4	30	0.6	2.206	1.113	0.500	0.353	0.505
4	30	0.7	1.242	0.633	0.240	0.118	0.475
4	30	0.8	0.550	0.257	0.140	0.038	0.014
4	30	0.9	0.220	0.119	0.074	0.047	0.010
	평균		1.054	0.531	0.239	0.139	0.251
4	40	0.6	3.264	1.601	0.969	0.569	1.102
4	40	0.7	0.970	0.537	0.374	0.234	0.068
4	40	0.8	0.381	0.295	0.175	0.071	0.070
4	40	0.9	0.209	0.158	0.084	0.039	0.020
	평균		1.206	0.648	0.400	0.228	0.315
4	50	0.6	2.849	1.051	0.903	0.547	0.782
4	50	0.7	1.129	0.627	0.533	0.233	0.245
4	50	0.8	0.417	0.240	0.128	0.098	0.114
4	50	0.9	0.152	0.107	0.070	0.017	0.022
	평균		1.137	0.506	0.409	0.224	0.291
	전체 평균		1.098	0.517	0.306	0.150	0.289

분할기법을 이용한 발견적 기법은 λ와 f의 값이 증가할수록 좋은 성능이 나타날 것으로 쉽게 예상할 수 있으며 실제의 결과값도 이를 나타내고 있다. FDH는 DH(12,6)과 비슷한 수준의 성능을 보이고 있는데 이는 그룹 당 배치의 수가 8~13개 수준을 가지는 실험 데이터에 대해 λ가 12인 경우와 비슷한 성향이 나타나기 때문이며 유연성을 첨가함으로 λ가 12로 정해진 경우보다 약간 우수한 성능을 보이고 있다고 말할 수 있다. <표 5>에 나타난 발견적 규칙의 성능은 BATCS의 경우 최적해 대비 110%의 편차를 보이며 분할기법을 통해 편차가 현저히 감소하고 있음을 알 수 있다. 교체시간이 있으며 배치조건이 없는 개별 작업의 스케줄 문제에 대해 ATCS의 규칙을 적용했을 때의 성능은 최적값 대비 10~50%의 편차를 가지고 있다고 보고된 것[10]과 비교하여 BATCS의 성능이 현저히

〈표 6〉 모델 생성 모수 차이에 의한 비교 (최적화 대비 상대적 평가지수)

		BATCS	DH(6,3)	DH(12,6)	DH(24,12)	FDH
τ	0.6	2.715	1.214	0.739	0.401	0.909
	0.7	1.069	0.524	0.299	0.128	0.186
	0.8	0.418	0.230	0.126	0.044	0.048
	0.9	0.189	0.101	0.059	0.026	0.015
R	0.2	0.864	0.434	0.261	0.089	0.252
	0.4	1.185	0.565	0.297	0.139	0.281
	0.6	1.170	0.559	0.320	0.172	0.299
	0.8	1.172	0.509	0.344	0.199	0.324
w	0.5	0.433	0.232	0.137	0.063	0.129
	1	1.045	0.498	0.287	0.140	0.275
	1.5	1.815	0.822	0.495	0.246	0.464

〈표 7〉 동적계획법과 발견적 규칙들의 계산시간 (단위 : 초)

그룹수	작업수	DP	BATCS	DH(6,3)	DH(12,6)	DH(24,12)	FDH
3	30	1.13	0.007	0.89	0.99	1.13	1.10
3	40	1.66	0.009	1.12	1.08	1.36	1.17
3	50	5.20	0.012	1.41	1.87	1.92	2.75
4	30	32.88	0.016	0.89	1.69	9.44	15.71
4	40	610.25	0.024	2.21	2.73	19.14	36.22
4	50	6486.39	0.026	2.42	2.96	20.28	85.46

떨어지는 것은 개별 작업마다 납기가 있으나 작업을 배치로 진행해서 납기를 만족시킬 수 있는 합리적 우선순위 선정이 그만큼 어렵고 복잡해지는 상황의 문제 특성 때문으로 판단된다. 그러나 분할기법을 적용했을 때의 성능의 개선이 현저하여 배치가 없는 스케줄 문제의 경우와 비슷한 수준으로 15~50% 수준의 편차 내에서 결과를 얻었고 특히 해의 도출에 필요한 시간은 최적해의 도출에 소요되는 시간과 비교하여 매우 작다. (<표 7> 참조)

<표 6>은 데이터 생성에 사용된 τ, R, w의 값에 따른 평가지수의 성향을 보여주고 있다. τ의 값이 증가할수록 즉 납기의 긴급도가 증가할수록 최적해에 대한 편차가 감소하여 우수한 성능을 보이고 있으며 반대로 납기가 느슨할수록 편차가 심한데 이는 느슨한 납기의 경우 목적함수인 지연도의 값이 작아 스케줄에 따라 상대적 비교를 할 때 평가지수의 변화가 심하기 때문이다. R값의 변화에 따른 평가지수의 차이는 심하지 않으나 w의 값에 따른 성능의 차이는 많이 나타난다. 즉 교체시간의 크기가 작업시간에 비해서 상대적으로 커지게 되면 작업시간이나 교체시간보다 납기 위주로 구성된 BATCS의 우선순위 규칙이 효율적이지 못하여 이를 시작 스케줄로 사용한 분할기법의 개선도에 한계가 있음을 알 수 있다.

<표 7>은 최적화 방법과 각 발견적 기법의 해를 도출하는데 소요된 계산시간에 관한 비교이다. 계산시간의 결과에서 발견적 규칙은 그룹과 작업수에 따라 계산시간이 동적계획법에 비해 아주 작게 상승하는 것을 발견할 수 있다. 그러나 동적 계획법은 문제의 크기가 커질 때 계산 시간이 비약적으로 상승하여 그룹과 작업 수가 일정 수를 넘으면

한정된 시간 안에 최적 스케줄을 구하는 데에는 한계가 따른다. 또한 유연분할기법(FDH)의 시간 역시 그룹의 크기에 따라 계산 시간이 크게 커지는데 이러한 결과는 분할되어지는 구역의 크기가 커짐으로써 분할 문제의 동적 계획법 적용시간이 커지기 때문이라고 볼 수 있다. 분할기법을 사용하여 생성된 스케줄의 결과값은 문제의 크기에 따라 변화량에 큰 차이가 나타나지 않고 계산시간의 증가도 그리 크지않아 문제의 크기가 커짐에 따라 분할기법의 효용성은 더욱 크다고 할 수 있다.

4. 결 론

본 논문에서 해법을 제시한 문제는 배치공정에서 그룹이 바뀔 경우의 배치간 교체시간이 발생하는 경우로서 지연도 문제에 교체시간을 고려하는 것은 실제 작업 현장에서 쉽게 찾을 수 있는 현실적인 문제이다. 이러한 상황에서 전체 지연시간을 최소화하는 것을 목적함수로 한 문제에서 최적값을 구할 수 있는 동적계획법을 개발하였고 빠른 시간 내에 실행 가능한 스케줄을 만들 수 있는 우선순위 규칙과 이를 응용한 분할기법의 발견적 기법을 제시, 성능을 비교하였다. 또한 다양한 문제에 따라 발견적 기법의 성능을 알아보기 위하여 납기의 긴급함과 분포의 정도를 조정할 수 있도록 문제의 새로운 생성방법을 제시하였다. 동적계획법을 통해 최적값을 얻을 수 있으나 그룹의 수나 작업의 크기 등 문제 크기가 커지면 계산시간이 비약적으로 증가함을 알 수 있었으며 발견적 규칙은 비교적 빠른 시간에 적당한 실행 가능해를 찾을 수 있었다. 또한 납기가 급한 작업 환경일수록 발견적 기법의 효율이 증가하는 것을 확인할 수 있었다. 경쟁이 치열해지고 있는 반도체 산업 환경에 미루어 볼 때 제시한 발견적 기법들은 효율적으로 사용되어질 수 있다고 판단된다.

본 연구에서는 그룹 간의 교체에 따른 교체시간을 일정한 값으로 정의하여 본 연구의 모델이 된 반도체 확산공정의 상황에 근거한 해법을 제시하였다. 교체시간을 일반적으로 앞의 그룹과 뒤에 오는 그룹에 영향을 받는 것이 일반적이므로 s_{ij} (i = 교체 전 그룹, j = 교체 후 그룹)라고 정의할 수 있다. 이때 모든 작업의 소요시간이 0이라고 가정하고 최대 배치 크기가 주어져 있지않다면 각 그룹은 하나의 배치로 구성될 것이며 이때 목적함수가 작업종료시간의 최소화라면 본 문제는 전형적인 판매원 순환문제(Traveling Salesman Problem)가 된다. 따라서 더 이상 다항식 시간 내에 최적해를 구할 수 있는 문제가 아니며 배치별 최대크기가 주어지고 목적함수가 지연도의 최소화인 경우 문제의 복잡도는 급격히 증가한다. 교체 전후의 그룹에 의해 교체시간이 정의되는 일반적인 문제에 교체시간의 변화가 심할 경우에 적용할 수 있는 해법에 대한 연구는 유용할 것으로 사려된다.

참 고 문 헌

[1] 박종관, 이영훈, "주문변화를 반영한 스케줄링 규칙에 대한 시뮬레이션을 이용한 성능평가", 「한국시뮬레이션 학회지」, 제9권 제3호(2000), pp.13-26.

[2] Chambers, R.J., R.L. Carraway, T.J. Lowe, and T.L. Morin, "Dominance and decomposition heuristic for single machine scheduling," *Operations Research*, Vol.39.(1991), pp.639-647.

[3] Chandru, V., C.Y. Lee, and R. Uzsoy, "Minimizing total completion time on batch processing machines," *International Journal of Production Research*, Vol.31.(1993), pp. 2097-2121.

[4] Chandru, V., C.Y. Lee, and R. Uzsoy, "Minimizing total completion time on batch processing machine with job families," *Op-*

eration Research Letters, Vol.13.(1993), pp. 61-65.

[5] Du, J. and J.Y.T. Leung, "Minimizing total tardiness on one machine is NP-hard," *Mathematics of Operations Research*, Vol.15. (1990), pp.83-495.

[6] Kim, Y.D., J.U. Kim, S.K. Lim, and H.B. Jun, "Due date based scheduling and control policies in a multiproduct semiconductor wafer fabrication facility," *IEEE Transactions on semiconductor manufacturing*, Vol.11.(1998), pp.155-164.

[7] Lawler, E.L. "A pseudopolynomial algorithm for sequencing jobs to minimize total tardiness," *Annals of Discrete Mathematics*, Vol.1.(1977), pp.331-342.

[8] Lee, C.Y., R. Uzsoy, and L. A. Martin-Vega, "Efficient algorithms for scheduling batch processing machines," *Operations Research*, Vol.40.(1992), pp.764-775.

[9] Lee, Y.H., K. Bhaskaran and M. Pinedo, "A heuristic to minimize the total weighted tardiness with sequence-dependent setups," *IIE Transactions*, Vol.29.(1997), pp.45-52.

[10] Lee, Y.H. and M. Pinedo, "Scheduling jobs on parallel machines with sequence-dependent setup times," *European Journal of Operational Research*, Vol.100.(1997), pp. 464-474.

[11] Mehta, S.V. and R. Uzsoy, "Minimizing total tardiness on a batch processing machine with incompatible job families," *IIE Transactions*, Vol.30.(1998), pp.165-178.

[12] Morton, T.E., and R.M. U. Rachamadugu, "Myopic heuristics for the single machine weighted tardiness problem," *Working Paper* 30-82-83(1982), Graduate School of Industrial Administration, Carnegie Mellon University, Pittsburgh.

[13] Park, Y.S., S.Y. Kim and Y.H. Lee, "Scheduling jobs on parallel machines applying neural network and heuristic rules," *Computers & Industrial Engineering*, Vol.38. (2000), pp.189-202.

[14] Potts, C. N. and L. N. Van Wassenhove, "A branch and bound algorithm for the total weighted tardiness problem," *Operations Research*, Vol.33.(1985), pp. 363-377.

[15] Raman, N., R.V. Rachamadugu and F.B. Talbot, "Real-time scheduling of an automated manufacturing center," *European Journal of Operational Research*, Vol.40. (1989), pp.222-242.

[16] Schrage, L. and K.R. Baker, "Dynamic programming solution of sequencing problems with precedence constraints," *Operations Research*, Vol.18.(1978), pp.444- 449.

[17] Uzsoy, R. "Scheduling batch processing machines with incompatible job families," *International Journal of Production Research*, Vol.33.(1995), pp.2685-2708.

[18] Vepsalainen, A. and T.E. Morton, "Priority rules and lead time estimation for job shop scheduling with weighted tardiness costs," *Management Science*, Vol.33.(1987), pp.1036-1047.