韓國經營科學會誌
第33권 第3號
2008年 9月

총 납기지연시간 최소화를 위한 두 단계 조립시스템에서의 일정계획에 관한 연구

하귀룡* · 이익선** · 윤상흠***†

A Scheduling Problem to Minimize Total Tardiness in the Two-stage Assembly-type Flowshop

Gui Ryong Ha* · Ik Sun Lee** · Sang Hum Yoon***

■ Abstract ■

This paper considers a scheduling problem to minimize the total tardiness in the two-stage assembly-type flowshop. The system is composed of multiple fabrication machines in the first stage and a final-assembly machine in the second stage. Each job consists of multiple tasks, each task is performed on the fabrication machine specified in advance. After all the tasks of a job are finished, the assembly task can be started on the final-assembly machine. The completion time of a job is the time that the assembly task for the job is completed. The objective of this paper is to find the optimal schedule minimizing the total tardiness of a group of jobs. In the problem analysis, we first derive three solution properties to determine the sequence between two consecutive jobs. Moreover, two lower objective bounds are derived and tested along with the derived properties within a branch-and-bound scheme. Two efficient heuristic algorithms are also developed. The overall performances of the proposed properties, branch-and-bound and heuristic algorithms are evaluated through numerical experiments.

Keyword : Two-stage Assembly-type Flowshop, Scheduling, Total Tardiness, Branch-and-Bound, Heuristic

논문접수일 : 2008년 03월 14일 논문게재확정일 : 2008년 05월 31일
* 경북대학교 경영학부
** 동아대학교 경영학부
*** 영남대학교 경영학부
† 교신저자

1. 서 론

본 연구에서는 2단계 (주문형)조립생산시스템에서의 일정계획 문제를 고려하고 있다. 일반적으로 2단계 조립생산시스템은 부품생산단계에서 각 부품 또는 구성품이 서로 다른 설비와 경로를 통해 독립적으로 생산된 후 조립단계로 이동하게 되고, 그곳에서 최종 조립공정을 통해 제품으로 완성된다. 이때 최종 조립은 해당하는 제품에 속한 모든 부품이 준비된 이후에 시작될 수 있다. 2단계 조립시스템은 반제품이 생산라인을 따라 이동하면서 작은 구성품이 결합되어 점차 완제품화되는 직렬 흐름형 생산시스템(flowshop)과 달리 고정된 장소로 주요 부품(구성품)이 이동한 후 그곳에서 모든 조립이 이루어지게 되므로 고정형 조립(fixed assembly)이라고 불리기도 한다. 2단계 조립시스템에서 개별 부품생산설비와 조립설비 간에는 물리적으로 2단계 직렬구조를 이루지만 부품 생산설비들 간에는 독립적인 병렬구조를 가짐으로써 전체적으로는 직렬과 병렬이 혼재된 형태를 이루게 된다. 본 연구에서는 이와 같이 부품생산과 조립으로 이어지는 2단계 조립시스템을 2단계 AFS(two-stage assembly-type flowshop system)라고 부르기로 한다.

AFS에서 부품생산단계의 병렬설비는 부품생산의 전용성 여부에 따라, 지정설비구조(dedicated facility structure)와 유연설비구조(flexible facility structure)의 두 종류로 구분될 수 있다. 먼저, 지정설비구조에서는 개별 부품별로 미리 지정된 전용설비를 통해서 부품생산이 이루어지는 경우이고, 유연설비구조는 각 설비가 모든 종류의 부품을 생산할 수 있으며, 한 종류에서 다른 종류로 생산이 전환될 때는 추가적인 준비시간(setup)이 소요되는 경우이다. 따라서, 유연설비구조의 부품생산단계를 포함하는 2단계 AFS는 2단계 혼합 흐름라인(hybrid flowshop)과도 동일한 구조를 가진다고 할 수 있다. 본 연구는 첫 번째 형태인 지정설비구조를 가지는 2단계 AFS에서의 일정계획을 대상으로 하고 있다.

본 연구와 같이 지정설비구조를 가지는 2단계 AFS에서의 일정계획에 관한 연구는 Lee et al.[7]부터 시작된다. Lee et al.[7]은 부품생산단계에 2대의 전용설비가 존재하고 조립단계에 한 대의 조립설비가 있는 2단계 AFS를 최초로 소개하고 최종작업완료시간(makespan)을 최소화하기 위한 일정계획문제를 다루었다. 그들은 대상 문제가 NP-complete임을 증명하고 Johnson's rule을 활용한 3가지 휴리스틱을 제안하였으며 이들의 오차한계(error bound)를 도출하였다. Potts et al.[10]은 Lee et al.[7]의 결과를 부품생산단계의 설비가 m대인 경우로 확장하고 역시 Johnson's rule을 활용한 휴리스틱들의 오차한계를 계산하였다. 또한 Hariri and Potts[4]는 Potts et al.[10]과 동일한 문제에 대해 최적해를 도출할 수 있는 효과적인 분지한계(branch-and-bound) 알고리즘을 제안한 바 있으며, Sun et al.[13]은 14가지의 새로운 휴리스틱들을 제시하고, 이들의 성능을 난수데이터를 활용한 수치실험으로 비교하였다. 또한 Yoon et al.[15]은 2단계 AFS에서 최종 작업완료시간을 최소화하는 문제는 시스템의 흐름을 반대로 하여 분리단계와 병렬처리단계로 이어지는 2단계 분리시스템(disassembly system)에서의 일정계획과 동일한 문제라는 것을 밝힌바 있다.

이상의 기존연구에서는 모두 최종작업완료시간을 일정계획의 목적함수로 고려하고 있는데 반해 Tozkapan et al.[14]은 2단계 AFS에서 총 가중완료시간(total weighted completion time)을 최소화하는 문제를 제안하고 이를 해결하기 위한 분지한계 알고리즘을 제시하였다. 윤상흠 등[1]은 Tozkapan et al.[14]과 동일한 문제에 대하여 향상된 하한값(lower bound)을 제안함으로써 Tozkapan et al.[14]에 비해 우수한 성능의 분지한계 알고리즘을 제시하였다.

앞서 살펴본 바와 같이 2단계 AFS의 일정계획과 관련한 기존연구의 대부분은 최종완료시간이나 (가중)완료시간의 합을 최소화하는 문제에 국한되어 있다. 일정계획모델에서 이러한 완료시간에 대

한 목적함수는 생산율(throughput)향상이나 재공품재고(WIP) 비용을 줄이기 위한 생산자 위주의 목적함수에 해당하며, 고객 주문처리에 따른 고객만족도를 고려하기 위해서는 각 주문에 대한 고객의 개별적인 요구납기(due date)를 고려하는 것이 바람직하다. 또한, AFS가 주문형 생산방식에 적합한 모형이고, 개별 주문에 대한 납기준수와 이를 통한 고객만족도를 시스템의 성과목표에 반영하기 위해서는 지연시간(tardiness)을 최소화 할 수 있는 일정계획의 도출이 중요하다고 할 수 있다. 따라서, 본 연구에서는 부품생산단계에 m대의 전용설비가 존재하고 조립단계에 한 대의 조립설비가 존재하는 일반적인 2단계 AFS에서 각 고객주문에 대한 납기를 고려하여 총 납기지연시간(total tardiness)을 최소화하는 주문간 처리순서를 결정하는 일정계획문제를 고려한다.

현재까지 AFS에서 이와 같이 납기를 고려한 일정계획에 관한 연구로는 Allabverdi and Al-Anzi [3]의 연구가 유일하다. 그들은 최대 납기지연시간(maximum lateness)을 최소화하는 문제를 제안하고, PSO(particle swarm optimization)와 타부서치(tabu search)를 활용한 메타휴리스틱을 제안한 바 있다. 일반적으로 일정계획에서 납기지연(latcness)과 순수납기지연(tardiness)은 분석방법과 문제의 복잡도 측면에서 서로 다른 목적함수일 뿐 아니라, 본 연구에서는 최대 납기지연시간이 아닌 총 납기지연시간을 고려하고 있다는 측면에서 Allabverdi and Al-Anzi[3]의 연구와는 차별화 된다.

2단계 AFS는 전체적으로는 m개의 2단계 흐름라인(flowshop)이 병렬구조로 혼합된 모습이다. 따라서, 2단계 흐름라인 일정계획문제는 AFS에서 부품설비가 한 대($m=1$)인 특수한 경우(special case)에 해당된다. 이런 점을 감안할 때, AFS에서의 일정계획에 대한 연구는 잘 알려진 2단계 흐름라인에 대한 기존연구에 대한 고찰로부터 출발하는 것이 바람직하다고 할 수 있다.

그 동안 2단계 흐름라인의 일정계획과 관련해서는 많은 연구결과가 있어 왔으나 본 연구와 같이 목적함수가 총 납기지연시간인 경우는 높은 문제복잡도로 인해 다른 목적함수에 비해서 그리 많은 편은 아니다. 먼저, Koulamas[6]는 2단계 흐름라인에서 모든 주문의 납기가 0인 경우에는 총 납기지연시간이 작업완료시간의 합과 같아지게 되고 납기지연시간을 최소화하는 일정계획문제가 strongly NP- complete임을 증명하였다. 따라서 전자에 언급하였듯이, 2단계 흐름라인은 AFS의 특수한 형태이므로, 본 연구에서 고려하고 있는 2단계 AFS에서의 총 납기지연시간 최소화 문제도 역시 strongly NP-complete임을 쉽게 알 수 있다. Sen et al.[12]은 2단계 흐름라인에서 총 납기지연시간을 최소화할 수 있는 최초의 분지한계 알고리즘을 제안하였다. Kim[5]은 Sen et al.[12]에 비해 적용가능성이 더 큰 우월성질(dominance property)을 개발하고 분지한계트리에서 아직 처리순서가 정해지지 않은 작업집단에 대한 하한값을 계산함으로써 Sen et al.[12]보다 우수한 성능의 분지한계 알고리즘을 제안하였다. 또한, 이상의 연구들이 분지노드에 해당하는 부분순열(partial sequence)이 후진(back to front)방식으로 확장되는 것을 고려한 반면, Pan et al.[8]은 전진(front to back)방식으로 확장되는 부분순열을 고려한 분지한계 알고리즘을 제안하였다. 마지막으로 최근에 Schaller[11]는 Pan et al.[8]의 연구를 확장하여 새로운 우월성질과 하한값을 제안함으로써 분지한계 알고리즘의 성능을 보다 향상시켰다.

본 연구에서는 이러한 2단계 흐름라인에서의 기존 연구결과를 확장하여 2단계 AFS에 적용가능한 최초의 분지한계 알고리즘을 제안하고자 한다. 이를 위해 2개의 연속된 작업간 우선순위를 규명할 수 있는 3가지 우월성질을 도출하였으며, 전진방식의 분지한계 알고리즘에 적용가능한 2가지 새로운 하한값을 제안한다. 또한 분지한계 트리의 상한값(upper bound)을 제공하거나, 보다 큰 사이즈의 문제에 대해서도 효율적인 해를 제공할 수 있는 2개의 휴리스틱 알고리즘을 제안한다. 제안된 분지한계 알고리즘과 휴리스틱 알고리즘들이 다양한 형

태의 납기 데이터에 대해서 효과적으로 대처하고 있음을 수치실험을 통해 검증한다.

본 연구의 구성은 다음과 같다. 제 2장에서는 본 연구에서 고려하고 있는 납기를 고려한 2단계 AFS에 대한 엄격한 정의가 제공된다. 또한, 분석에서 두 개의 연속된 작업 간의 우선순위관계를 규명할 수 있는 효과적인 우월성질들이 제시된다. 제 3장에서는 목적함수에 대한 하한값을 활용한 분지한계 알고리즘과 효율적인 해를 제공할 수 있는 휴리스틱 알고리즘들이 제안된다. 제 4장에서는 다양한 수치실험을 통해 제안된 우월성질, 분지한계 알고리즘, 휴리스틱들의 성능을 평가한다. 마지막으로 제 5장에서는 연구의 결론과 추후연구과제에 대한 논의가 이루어진다.

2. 2단계 AFS문제의 정의와 기초분석

설명의 편의를 위해 본 연구의 대상문제인 '총 납기지연시간을 최소화하기 위한 2단계 조립시스템에서의 일정계획문제(two-stage assembly-type flowshop scheduling for tardiness)'를 AFS_T라고 부르기로 한다. AFS_T에서는 총 n개의 처리해야 할 작업이 존재하고 각 작업은 $m+1$개의 부분작업으로 구성된다. 그 중에서 m개는 서로 다른 부품을 생산하는 부품작업(fabrication)이고 나머지 1개는 생산된 부품을 조립하는 조립작업이다.

각 부품작업들은 m개의 전용설비에서 수행되며 부품작업에 대한 설비할당은 미리 정해져 있다. 즉, 각 작업의 k번째 부품작업($k=1, 2, \cdots, m$)은 부품설비 k에서 수행되며, 작업 j의 k번째 부품작업을 위해서는 $p_{j,k}$($j=1, \cdots, n, k=1, \cdots, m$)의 가공시간이 소요된다. 또한 조립단계에서 작업 j의 최종조립작업은 작업 j에 속한 모든 부품작업들이 완료되는 시점 이후에 시작될 수 있으며, $p_{j,A}$($j=1, \cdots, n$)의 조립작업시간이 필요하다.

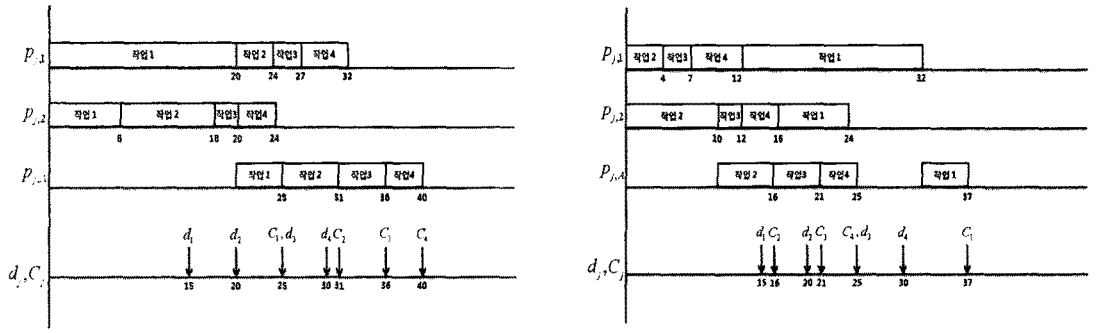
모든 부품설비 및 조립설비는 두 개 이상의 작업을 동시에 진행할 수 없으며, 일단 하나의 부품작업 또는 조립작업이 시작되면 중단없이(no preemption) 완료될 때까지 계속된다고 가정한다. 각 작업의 완료시간은 마지막 조립설비에서 조립이 완료되는 시점으로 모델링되며 작업 j의 완료시간을 C_j로 표시한다. 각 작업에 대한 정해진 납기(due date)를 d_j라 할 때, 작업 j의 납기지연시간은 $T_j=\max\{C_j-d_j, 0\}$로 표현된다. 따라서, 전체 n개 작업의 총 납기지연시간(total tardiness)은 $\sum_{j=1}^{n} T_j=\sum_{j=1}^{n}\max\{C_j-d_j,0\}$로 표현될 수 있으며 AFS_T의 목적은 이러한 총 납기지연시간을 최소화하는 각 설비에서의 최적 작업처리순서를 도출하는 것이다.

문제에 대한 정확한 이해를 돕기 위해 다음의 예제를 고려해 보자.

[예제 1] 처리해야 하는 총 4개의 작업이 있고 각 작업은 2개의 부품작업과 1개의 조립작업으로 구성된다. 각 부분작업의 가공시간과 납기가 <표 1>에 나타나 있다. 만약 모든 설비에서 납기우선순위(earliest due date, 이후 EDD)에 따라 작업을 처리할 경우에는 작업 1 → 작업 2 → 작업 3 → 작업 4의 순서로 작업이 진행되고 [그림 1-a]에 표시된 바와 각 작업의 납기지연시간은 $T_1=10$, $T_2=11$, $T_3=11$, $T_4=10$이고 총 납기지연시간은 42가 된다. 이때, 작업처리순서를 바꾸어 작업 2 → 작업 3 → 작업 4 → 작업 1의 순서로 작업이 진행된다면, [그림 1-b]에서와 같이 각 작업의 납기지연시간은 $T_1=22$, $T_2=0$, $T_3=0$, $T_4=0$이 되고 총 납기지연시간은 22가 되어 EDD에 비해 지연시간이 줄게 됨을 알 수 있다. 본 예제를 통해 AFS_T가 단순한 EDD에 의해 최적화될 수는 없음을 알 수 있다. □

〈표 1〉 AFS_T의 예제 데이터

	작업 1	작업 2	작업 3	작업 4
p_{j1}	20	4	3	5
p_{j2}	8	10	2	4
p_{jA}	5	6	5	4
d_j	15	20	25	30



(a) EDD에 대한 간트차트 (b) 작업 2→작업 3→작업 4→작업 1에 대한 간트차트

[그림 1] AFS_T의 예제에서 두 개의 일정계획해에 대한 간트차트

다음에 소개되는 정리 1은 '최적일정계획은 순열일정계획해(permutation schedule)들 중에서 존재한다'는 사실을 밝힌 것으로, 고려해야 할 해(solution)의 공간을 축소시키는 역할을 한다. 여기서, 순열일정계획해란 모든 부품설비에서의 생산순서와 조립설비에서의 생산순서가 동일한 일정계획해를 의미한다.

정리 1 : AFS_T에서 최적일정계획은 순열일정계획해에서 존재한다.

증명 : AFS_T에서 순열일정계획해중에 최적해가 존재한다는 사실은 최대완료시간이나 완료시간의 합의 목적함수인 경우에는 Lee et al.[7]과 Tozkapan et al.[14]에서 이미 증명된 바 있다. 납기지연시간을 목적함수로 할 경우에도 원칙적으로 별도의 증명이 필요하나 기존연구와 마찬가지로 간단한 작업교환(job interchange)에 의해 쉽게 증명될 수 있으므로 여기서는 구체적인 내용은 생략한다. □

정리 1에 의해 설비별로 작업들의 순서를 따로 결정할 필요가 없고 n개의 작업에 대한 최적해가 $n!$개의 순열일정계획 중에 존재함을 알 수 있다. 다음으로 작업들 간의 우선순위관계를 규명할 수 있는 3가지 우월성질들을 제시한다. 아래의 우월성질들은 다음 장에서 소개되는 분지한계 알고리즘에서 한계규칙(bounding rule)과 함께 분지한계트리의 노드를 제거하는 역할을 하게 된다.

성질 1 : 아래의 조건식들을 만족하는 두 작업 i와 j가 가공순서상 연속해서 처리되는 경우에는 작업 j를 i보다 앞서서 처리하는 것이 그 반대의 경우보다 우월하다.

$$C_i(S') - d_i \le 0 \qquad \text{(C-1)}$$

$$C_i(S') \le C_j(S) \qquad \text{(C-2a)}$$

또는

$$C_i(S') \le \min_{l \in \sigma}\left[\max_{1 \le k \le m}\left\{\sum_{l \in \sigma} p_{l,k} + p_{i,k}\right\}\right] \qquad \text{(C-2b)}$$

여기서 S는 작업 i가 작업 j직전에 처리되는 임의의 순열(sequence)이고 S'는 반대로 작업 j가 i보다 앞서서 처리되고 i와 j를 제외한 나머지 작업의 처리순서는 S와 동일한 순열이다. 또한 σ는 S에서 작업 j와 작업 j 이전에 처리되는 작업들을 포함하는 부분순열을 의미한다.

증명 : 증명은 조건 조합 {(C-1)과 (C-2a)} 또는 {(C-1)과 (C-2b)}가 만족할 경우 S'에서의 작업 i와 j의 납기지연시간의 합이 S의 것보다 작거나 같고 작업 i와 j 이후 작업들의 완료시간도 S'의 경우가 S보다 크지 않음으로써 전체적으로 S'이 S에 비해 우월함을 밝히면 된다.

먼저 S'에서 작업 j의 완료시간은 조건에 관계없이 S에서의 완료시간에 비해 작으므로 지연시간도 $T_j(S')$

$\leq T_j(S)$의 관계가 성립하며, 더욱이 조건 (C-1)은 $T_i(S')=0$을 의미하므로 $T_i(S')+T_j(S')=T_j(S') \leq T_i(S)+T_j(S)$임을 알 수 있다.

또한 조건 (C-2a)가 성립하면 S'에서 작업 i와 j이후에 처리되는 작업들의 조립시작시점은 S에서의 조립시작시점보다 작거나 같으므로 조립 완료시간도 작거나 같다. 다음으로 조건 (C-2b)에서 부등호 오른쪽 항은 작업 i와 j이후의 첫 번째 처리작업의 부품생산단계에서의 완료시점(또는, 조립단계에서의 처리시작시점)의 최소값을 의미하므로 조건 (C-2b)가 만족할 경우 조립단계에서는 유휴시간(idle time)이 발생하게 되고 작업 i와 j이후 작업들의 처리완료시점은 S와 S'에서 동일하게 된다. 따라서, S'이 S에 비해 총 납기지연시간 측면에서 우월하다. □

성질 2 : 아래의 조건식들을 만족하는 두 작업 i와 j가 가공순서상 연속해서 처리되는 경우에는 작업 j를 i보다 앞서서 처리하는 것이 그 반대의 경우보다 우월하다.

$$C_j(S')-d_j \geq 0 \qquad \text{(C-3)}$$

$$C_i(S) \geq C_j(S') \qquad \text{(C-4)}$$

$$t_v+p_{j,v} \leq \min[t_u+p_{i,u}, t_x+p_{j,A}] \qquad \text{(C-5)}$$

여기서 t_k는 작업 i와 j보다 이전에 처리된 작업들의 부품설비 k에서의 최종완료시간을 나타내고, 기호 x, u, v는 각각 $t_x+p_{i,x}+p_{j,x}=\max\{t_k+p_{i,k}+p_{j,k} | k=1, \cdots, m\}$, $t_u+p_{i,u}=\max\{t_k+p_{i,k} | k=1, \cdots, m\}$, $t_v+p_{j,v}=\max\{t_k+p_{j,k} | k=1, \cdots, m\}$의 식을 만족하는 부품설비를 지칭한다.

증명 : 먼저 $C_j(S)$와 $C_i(S')$을 표현하면 다음과 같다.

$$C_j(S)=t_A+\max[0, t_u+p_{i,u}-t_A, t_x+p_{i,x}+p_{i,x}-(t_A+p_{i,A})]+p_{i,A}+p_{j,A}$$

$$C_i(S')=t_A+\max[0, t_v+p_{j,v}-t_A, t_x+p_{j,x}+p_{i,x}-(t_A+p_{j,A})]+p_{j,A}+p_{i,A}$$

여기서 t_A는 조립설비 A에서 작업 i와 j보다 이전에 처리된 작업들의 최종완료시간을 의미한다. 조건 (C-5)에 의해서 $t_u+p_{i,u}-t_A \geq t_v+p_{j,v}-t_A$의 관계가 성립하고 다음의 전개를 통해 $(t_u+p_{i,u}-t_A) \geq (t_x+p_{j,x}+p_{i,x})-(t_A+p_{j,A})$의 관계식이 성립하므로 결국 $C_j(S) \geq C_i(S')$임을 알 수 있다.

$$\begin{aligned}(t_u+p_{i,u}-t_A)-\{(t_x+p_{j,x}+p_{i,x})-(t_A+p_{j,A})\} &= t_u+p_{i,u}-\{(t_x+p_{j,x}+p_{i,x})-p_{j,A}\} \\ &\geq t_x+p_{i,x}+p_{j,A}-(t_x+p_{j,x}+p_{i,x}) \\ &\quad \text{(인덱스 } u\text{의 정의에 의해)} \\ &= t_x+p_{j,A}-(t_x+p_{j,x}) \text{(조건 (C-5)와 인덱스 } v\text{의 정의에 의해)} \\ &\geq 0\end{aligned}$$

따라서 조건 (C-5)를 만족하면 성질 1의 (C-2a)를 만족하게 되므로 작업 j를 i보다 앞서서 처리할 경우(즉, S'의 경우), 작업 j와 i 이후에 처리되는 작업들의 납기지연시간이 반대의 경우(즉, S의 경우)에 비해 최소한 증가하지 않는다.

다음으로 작업 i와 j의 지연시간의 합이 S'의 경우가 S의 경우보다 작거나 같다는 것을 보이면 증명은 끝난다. 이는 $C_i(S)-d_i \geq 0$인 경우와 $C_i(S)-d_i<0$의 경우로 나뉘어 증명된다.

Case 1 : $C_i(S)-d_i \geq 0$인 경우.

이 경우는 조건 (C-3)에 의해 작업 i와 j가 S와 S'에서 모두 납기지연됨을 알 수 있다. 따라서

$$T_i(S)+T_j(S)=C_i(S)+C_j(S)-d_i-d_j$$

$$T_j(S')+T_i(S')=C_j(S')+C_i(S')-d_i-d_j$$

이 성립하고 조건 (C-4)와 앞에서 구한 $C_j(S) \geq C_i(S')$의 조건에 의해 $T_i(S)+T_j(S) \geq T_j(S')+T_i(S')$의 관계가 성립한다.

Case 2 : $C_i(S)-d_i<0$인 경우.

이 경우는 조건 (C-3)와 (C-4)에 의해 다음의 관계식이 성립한다.

$$d_i > C_i(S) \geq C_j(S') \geq d_j \qquad \text{(C-6)}$$

따라서 S와 S'에서 작업 i와 j에 의해 발생하는 납기지연시간의 합은 다음과 같다.

$$T_i(S)+T_j(S)=\max[0, C_j(S)-d_j]$$

$$T_j(S')+T_i(S')=C_j(S')-d_j+\max[0, C_i(S')-d_i]$$

$= \max\{C_j(S') - d_j,\ C_i(S') + C_j(S') - d_j - d_i\}$

이때 조건 (C-6)과 $C_j(S) \geq C_j(S')$에 의해 $T_i(S) + T_j(S) \geq T_j(S') + T_i(S')$임을 쉽게 알 수 있다. □

마지막으로 다음의 3번째 우월성질은 Schaller[11]에서 제시된 2단계 흐름라인에 대한 우월성질을 2단계 AFS에서도 적용될 수 있도록 확장한 것이다.

성질 3 : 아래의 조건식들을 만족하는 두 작업 i와 j가 가공순서상 연속해서 처리되는 경우에는 j를 i보다 앞서서 처리하는 것이 그 반대의 경우보다 우월하다.

$$T_j(S') + T_i(S') \leq T_i(S) + T_j(S) \qquad \text{(C-7)}$$

$$\{T_i(S) + T_j(S)\} - \{T_j(S') + T_i(S')\} \geq \{C_i(S') - LC\}(n - n_\sigma) \qquad \text{(C-8)}$$

여기서 $LC = \max\left[\min_{1 \leq l \leq n}\left\{\max_{1 \leq k \leq m}\left(\sum_{l' \in \sigma} p_{l',k} + \sum_{l' \in \sigma}(p_{l',k})\right)\right\}, C_j(S)\right]$이며, n은 전체 작업의 수이고, n_σ는 σ에 속하는 작업의 수를 나타낸다.

증명 : 조건 (C-7)을 만족하면, S'에서 작업 j와 i에 의해 발생하는 납기지연의 합은 S의 경우보다 작게 된다. 또한 조건 (C-8)에서 우변의 $(C_i(S') - LC)(n - n_\sigma)$는 S'에서 작업 j와 i 이후의 작업들의 납기지연시간 증가분의 상한값을 나타낸다. 따라서 이 값이 S에서 작업 i와 j의 납기지연과 S'의 작업 j와 i의 납기지연의 차보다 작다는 것은 S'에서 발생할 수 있는 총 납기지연이 S에서 발생하는 총 납기지연보다 작다는 것을 의미한다. □

3. 분지한계와 휴리스틱 알고리즘의 개발

3.1 분지한계 알고리즘

본 연구에서 제안하는 분지한계 알고리즘에서는 분지규칙으로 깊이우선탐색(depth-first search)을 사용하게 된다. 이는 탐색트리에서 다음으로 분지할 노드를 선택할 때 현재 트리상에서 가장 아래쪽의 노드(즉, 가장 큰 부분순열을 가진 노드)를 선택하는 것을 의미한다. 복수개의 후보가 발생할 경우에는 그 중에서 하한 값(lower bound)이 가장 작은 노드가 선택된다. Kim[5]은 2단계 흐름라인에서 총 납기지연시간을 최소화하기 위한 분지한계 알고리즘의 분지규칙은 일반적으로 깊이우선탐색이 넓이우선탐색(breath-first search)에 비해 우수하다는 것을 지적한 바 있다. 따라서 본 연구에서도 AFS가 2단계 흐름라인의 확장모델임을 감안하여 깊이우선탐색을 사용하였다.

한계규칙(bounding rule)은 각 노드의 상한과 하한을 이용하여 현재 탐색중인 노드의 제거(fathoming) 여부를 결정하고 해의 수렴을 촉진시킨다. 각 노드에서의 하한값으로는 Kim[5], Pan et al.[9], Schaller[11]가 제안한 2단계 흐름라인에 대한 아이디어를 AFS에 맞게 확장하였다.

이에 대한 설명을 위해 다음과 같은 기호를 사용한다.

- σ : 이전 노드에서 이미 처리순서가 결정된 작업들의 부분순열,
- σ' : σ에 포함되지 않은 나머지 작업들로 구성된 임의의 부분 순열,
- $n_{\sigma'}$: σ'에 속한 작업의 수,
- $C_{[z]}(\sigma')$: σ'에서 z번째 처리작업의 완료시간, $z = 1, 2, \cdots, n_{\sigma'}$,
- $p_{[l],A}(\sigma')$: σ'에서 l번째 처리작업의 조립기계 A에서의 가공시간, $l = 1, 2, \cdots, n_{\sigma'}$,
- $d_{EDD[z]}(\sigma')$: σ'에 속한 작업들을 납기우선순위(EDD)에 의해 재정렬할 경우, z번째에 해당하는 작업의 납기일, $z = 1, 2, \cdots, n_{\sigma'}$,
- $p_{SPT[l],k}(\sigma')$: σ'에 속한 작업들을 설비 k에서의 가공시간($p_{j,k}$)에 대한 오름차순(SPT)으로 재정렬할 경우 l번째 작업의 조립가공시간, $l = 1, 2, \cdots, n_{\sigma'}$, $k = 1, 2, \cdots, m$, A.

우선 전체 순열 $S = \sigma \cup \sigma'$에 대한 총 납기지연시간은 다음과 같이 표현될 수 있다.

$$\sum_{j\in S} T_j = \sum_{j\in\sigma} T_j + \sum_{j\in\sigma'} T_j \quad (1)$$

이때 $\sum_{j\in\sigma'} T_j$은 다음의 관계가 성립된다.

$$\sum_{j\in\sigma'} T_j = \sum_{j\in\sigma'} \max\{C_j - d_j,\ 0\}$$
$$\geq \sum_{z=1}^{n_{\sigma'}} \max\{C_{[z]}(\sigma') - d_{EDD[z]}(\sigma'),\ 0\} \quad (2)$$

위의 식 (2)는 Kim[5]의 연구에서 2단계 흐름라인에 대해 성립함을 제시한 바 있으며, 본 연구의 환경인 2단계 AFS에서도 적용가능하다는 것은 별도의 증명 없이 쉽게 알 수 있다.

또한 식 (2)는 다음의 관계가 성립한다.

$$\text{식 (2)} \geq \sum_{z=1}^{n_{\sigma'}} \max\left[\begin{array}{l}\max\{C_{\sigma,A},\ \min_{j\in\sigma'}\{\max_{1\leq k\leq m}(C_{\sigma,k}+p_{j,k})\}\} \\ +\sum_{l=1}^{z} p_{[l],A}(\sigma') - d_{EDD[z]}(\sigma'),\ 0\end{array}\right] \quad (3)$$

대괄호 안의 첫 번째 항인 $C_{\sigma,A}$는 σ에 속한 마지막 작업의 조립설비에서의 작업완료시점을 의미한다. 또한 두 번째 항에서 $C_{\sigma,k}$는 σ에 속한 마지막 작업의 부품설비 k에서의 완료시점이므로 $\delta = \min_{j\in\sigma'}\{\max_{1\leq k\leq m}(C_{\sigma,k}+p_{j,k})\}$은 σ'에 속한 한 개의 작업을 σ뒤에 추가했을 경우의 부품생산단계에서의 완료시점의 최소값을 의미한다. 따라서, 이 값이 $C_{\sigma,A}$보다 클 경우에는 조립기계에 유휴시간이 발생하게 되고 반대의 경우에는 유휴시간이 발생하지 않는다. 결국, 위의 관계식에서 $\max\{C_{\sigma,A},\ \delta\}$의 의미는 σ이후 첫 번째 작업의 조립기계에서의 가공시작시간의 최소값을 의미하게 된다. 또한 대괄호 안의 세 번째 항은 σ'에 속한 작업들의 조립설비에서 순수가공시간의 합을 나타내고, 이는 작업시간 $p_{j,A}$에 대해 SPT 순으로 정렬할 때 최소화 될 수 있다. 즉, $\sum_{l=1}^{z} p_{[l],A}(\sigma') \geq \sum_{l=1}^{z} p_{SPT'[l],A}(\sigma')$의 관계가 성립한다. 따라서 본 연구의 분지한계 알고리즘에서 사용하는 첫 번째 하한값 LB_1은 식 (1)과 식 (3)을 활용하여 다음과 같이 정의된다.

$$LB_1 = \sum_{j\in\sigma} T_j + \sum_{z=1}^{n_{\sigma'}} \max\left[\begin{array}{l}\max\{C_{\sigma,A},\ \min_{j\in\sigma'}\{\max_{1\leq k\leq m}(C_{\sigma,k}-p_{j,k})\}\} \\ +\sum_{l=1}^{z} p_{SPT'[l],A}(\sigma') - d_{EDD'[z]}(\sigma'),\ 0\end{array}\right]$$

두 번째 하한을 구하기 위해 식 (2)를 다음과 같이 다시 표현한다.

$$\text{식 (2)} \geq \sum_{z=1}^{n_{\sigma'}} \max\left[\max_{1\leq k\leq m}\left\{C_{\sigma,k}+\left(\sum_{l=1}^{z} p_{SPT[l],k}(\sigma')\right)\right\} + p_{SPT[y],A}(\sigma') - d_{EDD[z]}(\sigma'),\ 0\right] \quad (4)$$

여기에서 $C_{\sigma,k}+\left(\sum_{l=1}^{z} p_{SPT[l],k}(\sigma')\right)$은 σ'에 속한 z번째 처리작업의 부품설비 k에서의 완료시간의 하한값을 나타낸다. 식 (4)의 대괄호내의 세 번째 항인 $p_{SPT[y],A}(\sigma')$은 σ'에 속한 작업들의 조립설비에서의 가공시간을 SPT순으로 정렬했을 경우 y번째 해당되는 작업을 나타낸다. 이 식에서 y의 값은 반드시 1부터 시작되며, 만약 임의의 z단계에서 납기지연이 되는 작업이 발생하면, z단계에서는 $SPT[y]$위치에 해당하는 작업의 가공시간 $p_{SPT[y],A}(\sigma')$을 더하게 되고, $z+1$번째 단계에서는 $SPT[y]$에 해당되는 작업의 가동시간 대신에 $SPT[y+1]$에 해당되는 작업의 가공시간 $p_{SPT[y+1],A}(\sigma')$을 적용하게 된다. 이는 조립설비에서의 가장 작은 가공시간$(\min_{j\in\sigma'}\{p_{j,A}\})$을 더하는 것보다는 향상된 하한값을 제공하게 된다.

따라서 식 (1)과 식 (4)로부터 두 번째 하한값인 LB_2는 다음과 같이 도출된다.

$$LB_2 = \sum_{j\in\sigma} T_j + \sum_{z=1}^{n_{\sigma'}} \max\left[\max_{1\leq k\leq m}\left\{C_{\sigma,k}+\left(\sum_{l=1}^{z} p_{SPT[l],k}(\sigma')\right)\right\} + p_{SPT[y],A}(\sigma') - d_{EDD[z]}(\sigma'),\ 0\right] \quad (5)$$

기술된 LB_2에 대한 계산과정의 정확한 이해를 위해 다음과 같은 예제를 제시한다.

[예제 2] 부품설비의 수가 2대이고, 현재 고려하고 있는 분지한계 노드의 σ'에 속한 작업이 5개이며,

각 작업의 부품설비에서의 가공시간, 조립설비에서의 가공시간, 납기는 아래의 <표 2>와 같다. 부품설비 1의 σ의 완료시점은 3이고 부품설비 2에서의 σ의 완료시점은 5이다. 또한 σ를 처리하는데 현재까지 발생한 총 납기지연시간은 1이다.

〈표 2〉 예제 데이터

	작업 1	작업 2	작업 3	작업 4	작업 5
$p_{j,1}$	4	6	8	2	9
$p_{j,2}$	6	7	5	3	2
$p_{j,A}$	4	6	3	5	7
d_j	10	15	20	25	30

개별설비에서의 가공시간과 납기를 SPT와 EDD 순서로 다시 정렬하면 다음과 같다.

		작업 1	작업 2	작업 3	작업 4	작업 5
$p_{j,1}$	SPT 순	2	4	6	8	9
$p_{j,2}$	SPT 순	2	3	5	6	7
$p_{j,A}$	SPT 순	3	4	5	6	7
d_j	EDD 순	10	15	20	25	30

이때 하한값 LB_2는 식 (5)로부터 다음과 같이 구할 수 있다.

$$LB_2 = 1 + \left\lceil \begin{array}{l} \max\{\max(5,\ 7)+3-10,\ 0\} \\ +\max\{\max(9,\ 10)+3-15,0\} \\ +\max\{\max(15,\ 15)+3-20,\ 0\} \\ +\max\{\max(23,\ 21)+3-25,\ 0\} \\ +\max\{\max(32,\ 28)+4-30,\ 0\} \end{array} \right\rceil = 8$$

이 경우 세 번째 해당되는 작업까지는($z=1,\ 2,\ 3$) 납기지연이 발생하지 않으므로, $p_{SPT[y],\ A}$는 $y=1$에 해당되는 3이 된다. 그러나 네 번째 작업($z=4$)에서 납기지연이 발생하게 되고 네 번째 작업의 조립설비의 가공시간은 $p_{SPT[1],\ A}$에 해당되는 값인 3이 되지만, 다섯 번째 작업의 조립설비 가공시간은 $y-2$에 해당되는 $p_{SPT[2],\ A}$인 4를 더하게 된다. □

최종적으로 본 연구의 분지한계 알고리즘에서는 도출된 두 가지 하한 값 LB_1, LB_2 중에서 우수한 것을 각 노드별 최종 하한으로 사용한다.

3.2 휴리스틱 알고리즘

본 연구에서 제안하는 휴리스틱들은 크기가 큰 문제에 대한 효과적이고 효율적인 근사해(near-optimal)를 제공해 줄 뿐 아니라 분지한계 알고리즘의 초기에 루트노드의 상한값을 제공함으로써 분지트리상의 노드제거를 촉진시키는 역할을 하게 된다.

휴리스틱 1은 그리디(greedy) 방식의 발견적 해법으로 이익선과 윤상흠[2]이 단일단계 주문일정계획연구에서 제안한 아이디어를 2단계 AFS에 맞게 변형한 것이다. 휴리스틱 1은 매 단계(stage)마다 아직 할당되지 않은 나머지 작업들의 집합 σ'으로부터 한 개의 작업을 선택하여 이미 처리순서가 정해진(할당된) 부분순열 σ의 다음 위치에 할당하게 된다. 이때 σ에 속한 작업들의 최종완료시간을 $C_{\sigma,A}$라 할 때 아직 할당되지 않은 σ'에 속한 작업 j에 대해 $r_j = d_j - C_{\sigma,A}$를 계산하면 r_j는 작업 j의 납기까지의 잔여시간을 의미하게 되고 이 값이 작을수록 지연이 발생할 가능성이 크므로 우선적으로 선택되는 것이 유리할 것이다. 또한 σ의 다음 위치에 작업 j를 할당할 경우 작업 j의 완료시간은 $C_j = \max\left[\max_{1 \le k \le m}\{C_{\sigma,k}+p_{j,k}\},\ C_{\sigma,A}\right]+p_{j,A}$로 계산될 수 있다. $C_{\sigma,k}$는 σ에 속한 모든 작업들의 부품설비 k에서의 완료시점이며 $C_{\sigma,k} = \sum_{j \in \sigma} p_{j,k}$로 계산된다. 이때 작업 j에 대해 계산되는 $\hat{p}_j = C_j - C_{\sigma,A}$의 값은 작업 j를 σ다음에 할당함으로써 발생되는 최종완료시간의 증분치에 해당하므로 만약 $\hat{p}_j$의 값이 잔여시간 r_j를 초과한다면 지연이 발생하게 되고, 이러한 작업들에 대해서는 r_j를 기준으로 작업을 할

당하기 보다는 $\hat{p_j}$의 값이 작은 작업을 우선적으로 선택하는 것이 타당하다.

이러한 개념을 바탕으로 휴리스틱 1에서는 현재 단계에서 할당할 작업을 선택할 때 납기까지의 잔여시간 r_j와 완료시간에 대한 증분치 $\hat{p_j}$를 동시에 고려한 기준치 α_j를 아직 할당되지 않은 σ'에 속한 작업들을 대상으로 계산하여 이 값이 가장 작은 작업을 우선적으로 선택한다. α_j에 대한 구체적인 표현은 다음과 같다.

$$\alpha_j = \max\{C_j - C_{\sigma,A},\ d_j - C_{\sigma,A}\} \tag{6}$$

휴리스틱 1을 단계별로 자세히 표현하면 다음과 같다.

<휴리스틱 1>

단계 0 : $\sigma' = \{1, 2, \cdots, n\}$, $C_{\sigma,A} = 0$.

단계 1 : σ'에 속하는 작업들에 대해 식 (6)의 α_j를 계산하고 이 값이 가장 작은 작업 x를 선택한다.

단계 2 : 작업 x를 집합 σ'에서 제거하고, σ의 마지막 위치에 삽입한다. 또한, 시점 $C_{\sigma,A}$을 $\max[\max_{1 \le k \le m}\{C_{\sigma,k} + p_{x,k}\},\ C_{\sigma,A}] + p_{x,A}$의 값으로 업데이트 한다.

단계 3 : σ'가 공집합이면 종료하고, 그렇지 않으면, 단계 1로 간다.

두 번째 휴리스틱에서는 미할당 작업집합 σ'에 속한 작업들에 대하여, 납기 d_j와 현재까지의 완료시간 $C_{\sigma,A}$를 비교하여 이미 납기지연이 확정된 작업들이 발생되지 않으면, EDD 순위상의 첫 번째 작업을 σ이후의 첫 번째 작업으로 할당한다. 그러나 현재 단계에서 σ'에 속한 작업들 중 이미 납기지연이 확실한 작업들이 발생하면, 그러한 작업들 중에서 해당 작업을 σ의 다음 위치에 할당할 경우 발생되는 완료시간의 증분치 $\hat{p_j} = C_j - C_{\sigma,A}$를 계산하여 이 값이 작은 작업을 우선적으로 할당한다.

휴리스틱 2를 단계별로 자세히 표현하면 다음과 같다.

<휴리스틱 2>

단계 0 : $\sigma' = \{1, 2, \cdots, n\}$, $C_{\sigma,A} = 0$.

단계 1 : σ'에 속하는 작업들에 대해 현재시점에서 이미 납기지연이 확정된 작업들의 집합 $\Omega = \{j;\ d_j \le C_{\sigma,A}\}$를 구성한다.

단계 2 : 만약 집합 Ω가 공집합이면, σ'에 속한 작업 중에 가장 작은 d_j를 가지는 작업 x를 선택하고 단계 4로 간다. 그렇지 않으면 단계 3으로 간다.

단계 3 : Ω에 속하는 작업들에 대해 완료시간 증분치 $\hat{p_j} = C_j - C_{\sigma,A}$를 계산하고 이 값이 가장 작은 작업 x를 선택하고 단계 4로 간다.

단계 4, 5 : 휴리스틱 1의 단계 2, 3과 동일하다.

본 연구에서 제안된 2개의 휴리스틱의 목적함수 값을 각각 UB_1, UB_2라고 할 때, 분지한계 알고리즘에서는 초기상한값으로 이 중에서 값이 작은 것을 사용하게 된다.

4. 수치실험을 통한 알고리즘 성능평가

본 장에서는 앞에서 제안한 우월성질, 분지한계 알고리즘, 그리고 휴리스틱의 성능을 다양한 수치실험을 통해 분석한다. 알고리즘은 C언어로 프로그래밍 되었고, Pentium Ⅳ 3.0 GHz PC에서 실행되었다. 먼저 평가에 사용된 실험 데이터들의 생성방법은 다음과 같다.

1) 부품설비에서의 가공시간 $p_{j,k}(j = 1, \cdots, n,\ k = 1, \cdots, m)$와 조립설비에서의 가공시간 $p_{j,A}(j = 1, \cdots, n)$는 모두 $U(1,\ 10)$으로 부터 생성된다. 여기서, $U(a,\ b)$는 a와 b를 모수로 가지는 균등분포를 의미한다.

2) 납기 $d_j(j = 1, \cdots, n)$는 $U\left[P \times \left(1 - TF - \frac{RDD}{2}\right)\right.$,

$P\times\left(1-TF+\frac{RDD}{2}\right)\Big]$로 부터 생성된다. 여기서, P는 최종작업완료시간(makespan)의 하한값을 의미하며

$$P=\max\left[\max_{1\le k\le m}\left(\sum_{j=1}^{n}p_{j,k}\right)+\min_{1\le j\le n}(p_{j,A}),\ \sum_{j=1}^{n}p_{j,A}+\min_{1\le j\le n}\{\max_{1\le k\le m}(p_{j,k})\}\right]$$ 의

값을 사용한다.

전체 실험은 작업 수(n)를 6종류(8, 10, 12, 14, 16, 18), 부품설비 수(m)는 3종류(2, 5, 8)에 대해 실시되었다. 납기데이터는 문제의 난이도 조절을 위해 납기지연시간을 목적함수로 가지는 일정계획 연구에서 많이 활용되는 납기 범위계수(RDD : relative range of due date)와 납기 tightness(TF : tardiness factor)를 고려하여 생성하였다. 본 실험에서 이용한 RDD와 TF계수는 Kim[5]의 2단계 흐름라인에서 이용한 계수값을 사용하였으며, RDD는 5종류(0.8, 1.0, 1.2, 1.4, 1.6), TF를 5종류(0.1, 0.2, 0.3, 0.4, 0.5)로 설정하였다. 작업수, 부품설비수, RDD, TF의 총 450문제($6\times3\times5\times5$)의 문제조합에 대해 각각 20문제씩, 총 9,000개의 문제를 임의로 생성하였다.

납기의 경우 RDD의 값이 증가하면 각 작업의 납기 데이터의 편차(분산)가 커지게 되고, 반대로 감소하면 편차가 작게 된다. 또한 TF의 값이 증가하면 각 작업의 납기 값이 전체적으로 작게 되어 지연되는 작업이 늘어나게 되고, 반대로 감소하면 납기 값이 작업의 완료시간에 비해 느슨해지는 결과가 된다.

지금까지 설명한 실험환경 하에서 먼저 제안된

〈표 3〉 분지한계 알고리즘 성능평가

작업수	부품 설비수	계산시간			탐색 노드 수		
		Average	Median	Max.	Average	Median	Max.
8	2	<< 0.001	<< 0.001	0.02	89	54	797
	5	<< 0.001	<< 0.001	0.02	73	50	620
	8	<< 0.001	<< 0.001	0.02	77	50	623
10	2	0.01	0.02	0.16	583	176	10,477
	5	0.01	0.02	0.22	464	138	13,016
	8	0.01	0.02	0.11	353	123	4,718
12	2	0.14	0.02	7.09	6,275	593	480,284
	5	0.14	0.02	3.42	4,256	365	129,880
	8	0.14	0.03	3.11	3,199	488	92,275
14	2	2.49	0.06	228.58	93,188	1,409	10513,512
	5	1.89	0.08	196.14	53,393	1,326	7884,625
	8	1.92	0.09	103.95	43,310	1,163	3589,342
16	2	19.09	0.15	912.11	502,790	2,685	24,713,767
	5	22.15	0.27	993.75	435,278	3,218	16,267,847
	8	18.12	0.14	1,021.83	292,724	1,260	18,477,828
18	2	(*)78.73	(*)0.19	(*)	(*)1,483,231	(*)2,395	(*)
	5	(*)81.83	(*)0.39	(*)	(*)1,073,640	(*)3,375	(*)
	8	(*)61.32	(*)0.20	(*)	(*)610,437	(*)1,564	(*)

주) (*) 1200초(CPU)안에 풀 수 없는 문제가 존재하여 통계량을 얻을 수 없는 경우임.

〈표 4〉 우월성질들을 사용하지 않을 경우의 분지한계 결과

작업수	부품 설비수	계산시간			탐색 노드 수		
		Average	Median	Max.	Average	Median	Max.
8	2	<< 0.001	<< 0.001	0.03	139	68	1864
	5	<< 0.001	<< 0.001	0.03	113	67	1365
	8	<< 0.001	<< 0.001	0.03	123	69	1875
10	2	0.02	0.02	0.47	1,143	271	33,991
	5	0.02	0.02	0.42	877	211	24,786
	8	0.03	0.02	0.81	1035	205	60,475
12	2	0.42	0.03	65.19	22,466	889	4,768,126
	5	0.39	0.03	25.73	12,632	609	1,188,243
	8	0.38	0.09	20.19	10,311	2,134	960,798
14	2	5.87	0.13	546.28	250,760	2,137	26,890,889
	5	(*)	(*)	(*)	(*)	(*)	(*)
	8	(*)	(*)	(*)	(*)	(*)	(*)

〈표 5〉 RDD와 TF값 변화에 따른 분지한계 알고리즘 성능평가

RDD	TF	계산시간			탐색 노드 수		
		Average	Median	Max.	Average	Median	Max.
0.8	0.1	<< 0.001	<< 0.001	0.02	5	1	97
	0.2	37.00	0.02	699.98	1,114,014	105	21,155,291
	0.3	57.62	1.94	392.06	2,116,910	32,598	15,730,649
	0.4	87.10	50.47	379.02	2,265,934	1,556,460	9,062,740
	0.5	44.05	29.06	199.70	982,851	636,618	4,546,172
1.0	0.1	0.96	<< 0.001	11.63	22,189	1	252,787
	0.2	9.78	<< 0.001	181.14	218,648	1	4,091,648
	0.3	5.18	0.20	72.59	108,087	4,003	1,538,936
	0.4	26.89	7.69	128.00	712,122	168,833	3,938,515
	0.5	45.08	16.08	293.67	1,132,545	357,288	7,885,685
1.2	0.1	0.95	<< 0.001	18.94	22,569	1	451,389
	0.2	3.63	0.02	39.77	75,973	1	824,594
	0.3	2.78	0.05	23.25	62,744	643	531,235
	0.4	8.91	0.46	73.06	215,002	7,607	2,027,922
	0.5	13.74	1.34	192.03	381,867	28,074	5,700,134
1.4	0.1	0.08	<< 0.001	0.92	1,397	1	17,340
	0.2	0.32	0.01	3.36	6,147	16	67,876
	0.3	1.80	0.49	12.50	44,249	8,966	343,980
	0.4	34.06	0.19	311.56	866,073	3,228	8,081,221
	0.5	12.70	1.46	129.16	319,289	33,278	3,272,484
1.6	0.1	0.15	<< 0.001	1.56	2,341	31	23,382
	0.2	0.72	0.03	5.27	13,640	389	104,026
	0.3	4.65	0.05	66.56	105,871	832	1,559,034
	0.4	9.95	0.14	73.56	227,486	2,128	1,707,396
	0.5	22.58	2.12	217.66	537,049	39,353	5,620,950

분지한계 알고리즘의 성능을 평가한다. 부품설비수와 개별 작업수 변화에 따른 1,500개씩의 문제에 대한 실험을 수행한 결과가 <표 3>에 요약되어 있다. 표에서 "탐색노드 수"는 분지한계 트리에서 생성된 전체 노드의 수를 의미하고, "계산시간"은 컴퓨터의 CPU 소요시간을 나타낸다. 표에 나타난 바와 같이 1,200초(CPU)를 한계시간으로 설정할 때 약 18개 전후의 작업까지를 풀 수 있음을 알 수 있다. 18개 미만의 작업에서는 모든 실험문제에서 한계시간 이내에 최적해를 도출하였으며, 18개 작업인 경우에 500개 중에 최적해를 찾은 문제는 부품설비가 2대인 경우에 469개이며, 5대인 경우에는 474개, 8대인 경우에서는 461개이다. 표에서 작업수 18인 경우는 한계시간 이내에 최적해를 찾은 경우들에 대한 통계량을 제시한 것이다.

표에서 부품설비의 수가 증가함에도 불구하고 분지한계 트리의 평균 노드수가 크게 증가하지 않음을 알 수 있다. 특히 부품설비수가 8대로 늘어난 경우에는 오히려 노드수가 감소하고 있어 제안된 분지한계 알고리즘의 하한값이 부품설비가 다중인 경우에 대해 효과적으로 설계되었음을 알 수 있다. 여기서, 각 부품설비수의 변화에 따라 노드수가 감소하고 있음에도 계산시간이 늘어나는 경우가 발생하는 것은 각 노드에서 우월성질의 적용여부를 체크하거나, 하한값을 계산하는 복잡도가 늘어나기 때문으로 풀이된다.

다음으로 제 2장에서 제시한 우월성질들이 분지한계 알고리즘의 성능향상에 얼마나 도움을 주었는지 평가하기 위한 실험을 실시하였다. <표 4>는 제안된 우월성질들을 사용하지 않았을 경우의 분지한계 알고리즘의 성능결과를 보여주고 있다. 예를 들어 작업수가 12개이고 부품설비의 수가 2대인 경우를 <표 3>과 비교해 보면, 우월성질을 사용한 경우의 평균탐색 노드 수가 6,275인 반면, 그렇지 않은 경우에는 22,466개로 크게 증가함을 알 수 있다.

또한 RDD와 TF값의 변화에 따른 성능변화를 살펴보기 위한 실험을 수행하였다. 이 실험은 부품설비 수를 2대, 작업수를 16개로 고정하여 수행한 것으로, 결과는 <표 5>에 요약되어 있다. 납기데이터의 산포가 커지거나(RDD가 증가), 상대적인 납기의 크기가 작아지는 경우(TF가 증가)에도 분지한계가 효과적으로 대응하고 있음을 알 수 있다. 또한 RDD값이 감소할수록(0.8에 가까울수록), TF값은 증가할수록(0.5에 가까울수록) 알고리즘의 성능이 저하되는 것을 알 수 있다.

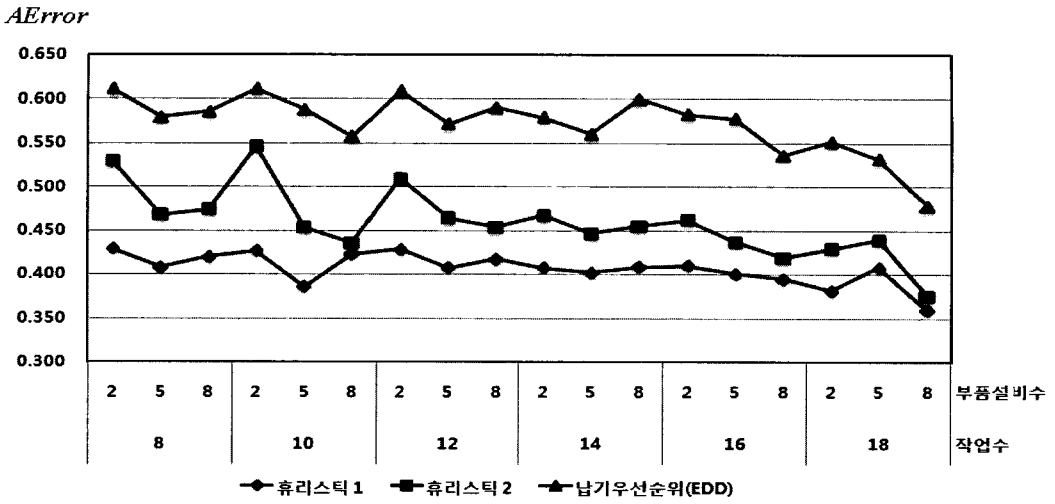
다음으로 휴리스틱에 대한 성능평가를 수행하였으며 결과는 <표 6>과 <표 7>에 요약되어 있다. 표에서 $AError$는 최적해 대비 휴리스틱의 평균오차(average error)를 나타내기 위한 척도로써 각 문제에 대해 다음과 같이 계산된 개별적인 $Error$에 대한 평균값을 의미한다.

$$Error = \frac{T(S_H) - T(S^*)}{T(S_{Worst}) - T(S^*)}$$

여기서, $T(S_H)$는 해당 휴리스틱을 통해 구해진

〈표 6〉 휴리스틱의 성능평가

작업수	부품설비수	$AError$		
		휴리스틱 1	휴리스틱 2	EDD
8	2	0.429	0.529	0.611
	5	0.408	0.468	0.579
	8	0.420	0.474	0.585
10	2	0.427	0.546	0.611
	5	0.385	0.453	0.588
	8	0.423	0.436	0.557
12	2	0.429	0.510	0.609
	5	0.407	0.464	0.571
	8	0.418	0.454	0.590
14	2	0.407	0.467	0.579
	5	0.402	0.447	0.560
	8	0.409	0.455	0.600
16	2	0.410	0.462	0.582
	5	0.401	0.437	0.578
	8	0.395	0.419	0.536
18	2	0.381	0.429	0.552
	5	0.408	0.439	0.532
	8	0.359	0.376	0.478



[그림 2] 휴리스틱들간의 최적해 대비 평균오차

해의 총 납기지연시간을 나타내고, $T(S^*)$는 분지한계 알고리즘을 통해 구한 최적해의 총 납기지연시간을 의미한다. 또한 $T(S_{Worst})$는 3가지 휴리스틱(납기우선순위규칙(EDD), 휴리스틱 1과 2)을 통해 구해진 각각의 해들의 총 납기지연시간 중에 가장 큰 값을 뜻한다. 따라서 *Error*는 각 문제에서 최적해 대비 최대발생편차에 대한 해당 휴리스틱 편차의 비율값을 의미한다. 이 값이 작을수록 3가지 휴리스틱 중에 상대적으로 성능이 좋음을 의미하고 동시에 최적해와의 편차도 적음을 나타낸다.

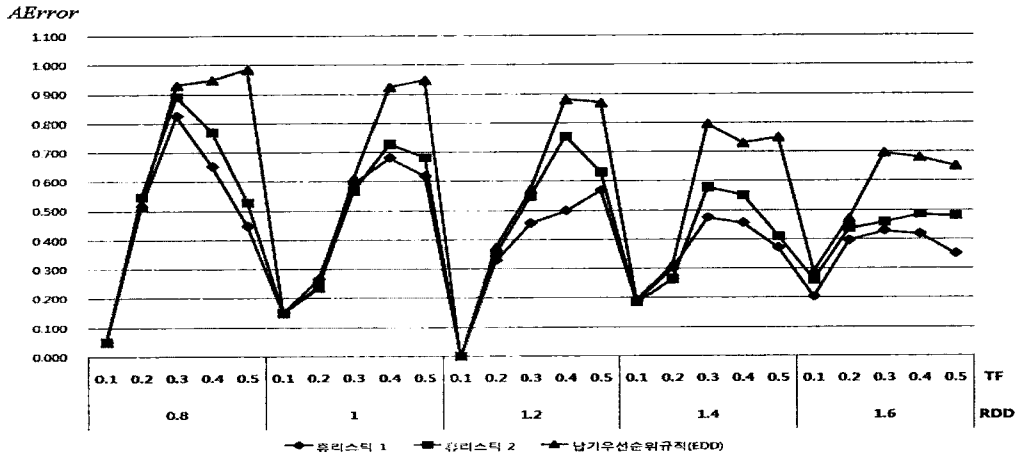
<표 6>과 [그림 2]는 부품설비수와 작업수의 조합에 대해 발생된 임의의 500문제에 대한 *Error*의 평균값과 그 추세를 보여주고 있다.

휴리스틱 1과 2는 모든 작업문제 조합에서 항상 납기우선순위에 비해 우수한 성능을 보여주고 있다. 또한, 작업수나 부품설비수의 증가에도 성능이 크게 나빠지지 않고 있음을 확인할 수 있다. 또한 평균의 관점에서는 휴리스틱 1이 가장 우수한 성능을 나타내고 있음을 [그림 2]를 통해 확인할 수 있다.

<표 7>은 RDD와 TF값의 변화에 따른 휴리스틱의 성능변화를 보여 주고 있다. 실험은 부품설비수를 2대, 작업수를 16개로 고정한 상태에서 진행하였으며, RDD와 TF값의 각 조합에 대해 20문제씩

〈표 7〉 RDD와 TF값 변화에 따른 휴리스틱들의 성능평가

RDD	TF	*Error*		
		휴리스틱 1	휴리스틱 2	EDD
0.8	0.1	0.050	0.050	0.050
	0.2	0.516	0.546	0.516
	0.3	0.824	0.890	0.930
	0.4	0.652	0.771	0.948
	0.5	0.448	0.529	0.985
1.0	0.1	0.150	0.150	0.150
	0.2	0.266	0.242	0.237
	0.3	0.592	0.568	0.611
	0.4	0.681	0.728	0.923
	0.5	0.618	0.682	0.947
1.2	0.1	0.000	0.000	0.000
	0.2	0.330	0.354	0.374
	0.3	0.455	0.547	0.571
	0.4	0.498	0.754	0.879
	0.5	0.568	0.629	0.868
1.4	0.1	0.188	0.186	0.193
	0.2	0.303	0.264	0.313
	0.3	0.474	0.578	0.796
	0.4	0.456	0.550	0.730
	0.5	0.371	0.411	0.750
1.6	0.1	0.204	0.261	0.291
	0.2	0.396	0.436	0.470
	0.3	0.429	0.458	0.697
	0.4	0.419	0.485	0.681
	0.5	0.351	0.479	0.651



[그림 3] RDD와 TF에 따른 휴리스틱들의 평균오차

을 실험한 결과를 요약한 것이다. [그림 3]는 RDD와 TF값의 변화에 따른 평균오차의 추세를 보여주고 있다. 특히 TF의 값이 0.5에 가까울수록 납기지연이 크게 발생하게 되고 문제가 어려워짐으로써 휴리스틱간의 성능차이가 커진다는 것을 확인할 수 있다.

5. 결론 및 추후연구

본 연구에서는 부품생산단계에 m대의 전용설비가 존재하는 일반적인 2단계 AFS에서 각 작업에 대한 납기와 관련된 일정계획문제를 제안하였다. 일정계획의 성과지표 중에서 총 납기지연시간을 고려하였으며, 본 연구는 이를 최소화하는 각 설비에서의 작업간 처리순서결정을 위한 방안을 제공하고 있다. 총 납기지연시간은 고객만족도를 대변하는 대표적인 성과지표이며 이를 최소화하는 것은 문제 복잡도 측면에서 가장 어려운 문제 중 하나이다. 본 연구에서는 최적해의 도출을 위한 분지한계 알고리즘을 제안하고, 알고리즘의 성능을 향상시키는데 효과적인 우월성질들과 하한값들을 개발하였다. 또한 분지한계 알고리즘의 상한값으로 활용될 수 있는 효율적인 휴리스틱들을 개발하였다.

AFS는 주문형 조립생산뿐만 아니라 분산데이터베이스(Allabverdi and Al-Anzi[3] 참조)나 분산처리(parallel processing)와 같은 동시성(concurrency)이 존재하는 많은 응용분야에 적용이 가능하며 추가적으로 다양한 일정계획의 성과지표와 예외상황을 고려할 경우, 향후 많은 연구과제가 존재한다고 할 수 있다. 하지만 AFS는 흐름라인(flowshop)의 변형인 만큼 잘 알려진 2단계 흐름라인(flowshop)에 대한 수많은 연구결과가 2단계 AFS에 대한 새로운 연구를 위한 출발점이자 한계점을 동시에 제공한다고 할 수 있다.

참 고 문 헌

[1] 윤상흠, 이익선, 이종협, "두 단계 조립시스템에서 총 가중완료시간을 최소화하는 일정계획문제", 「대한산업공학회지」, 제33권, 제2호(2007), pp.254-264.

[2] 이익선, 윤상흠, "가중납기지연시간을 고려한 최적 주문처리순서에 관한 연구", 「한국경영과학회지」, 제33권, 제2호(2008), pp.87-101.

[3] Allahverdi, A. and F.S. Al-Anzi, "A PSO and a Tabu Search Heuristics for the Assembly Scheduling Problem of the Two-stage Distributed Database Application," *Computers*

and Operations Research, Vol.33(2006), pp. 1056-1080.

[4] Hariri, A.M.A. and C.N. Potts, "A Branch-and-Bound Algorithm for the Two-Stage Assembly Scheduling Problem," *European Journal of Operational Research*, Vol.103 (1997),. pp.547-556.

[5] Kim, Y.D., "A New Branch and Bound Algorithm for Minimizing Mean Tardiness in Two-machine Flowshops," *Computers and Operations Research*, Vol.20(1993), pp.391-401.

[6] Koulamus, C., "The Total Tardiness Problem : Review and Extensions," *Operations Research*, Vol.42(1994), pp.1025-1041.

[7] Lee, C.Y., T.C.E. Cheng and B.M.T. Lin, "Minimizing the Makespan in the 3-Machine Assembly-type Flowshop Scheduling Problem," *Management Science*, Vol.39(1993), pp.616-625.

[8] Pan, J.C., J. Chen and C. Chao, "Minimizing Tardiness in a Two-machine Flow-shop," *Computers and Operations Research*, Vol.29 (2002), pp.869-885.

[9] Pan, J.C. and E.T. Fan, "Two-machine Flowshop Scheduling to Minimize Total Tardiness," *International Journal of Systems Science*, Vol.28(1997), pp.405-414.

[10] Potts, C.N., S.V. Sevast'janov, V.A. Strusevich, L.N. van Wassenhove and C.M. Zwaneveld, "The Two-stage Assembly Scheduling Problem : Complexity and Approximation," *Operations Research*, Vol.43(1995), pp.346-355.

[11] Schaller, J., "Note on Minimizing Total Tardiness in a Two-machine Flowshop," *Computers and Operations Research*, Vol.32(2005), pp.3273-3281.

[12] Sen, T., P. Dileepan and J.N.D. Gupta, "The Two-machine Flowshop Scheduling Problem with Total Tardiness," *Computers and Operations Research*, Vol.16(1989), pp.333-340.

[13] Sun, X., K. Morizawa and H. Nagasawa, "Powerful Heuristics to Minimize Makespan in Fixed, 3-Machine, Assembly-type Flowshop Scheduling," *European Journal of Operational Research*, Vol.146(2003), pp.499-517.

[14] Tozkapan, A., O. Kirca and C.S. Chung, "A Branch and Bound Algorithm to Minimize the Total Weighted Flowtime for the Two-stage Assembly Scheduling Problem," *Computers and Operations Research*, Vol.30 (2003), pp.309-320.

[15] Yoon, S.H., I.S. Lee and C.S. Sung, "A Note on the Reversibility of the Two Stage Assembly Scheduling Problem," *International Journal of Management Science*, Vol.13(2007), pp.25-34.