Journal of the Society of Korea Industrial and Systems Engineering
Vol, 29, No. 4, pp.34－42, December 2006.

부품외주를 고려한 조립형 Flowshop 일정계획문제 연구

전 재 호

충주대학교 행정학부 행정정보시스템학전공

An Assembly-Type Flowshop Scheduling Problem with Outsourcing Allowed

Jae-Ho Juhn

Division of Public Management Information System, Chungju National University

This paper considers an assembly-type flowshop scheduling problem in which each job is assembled with two types of components. One type of the components is outsourced with positive lead time but the other type is fabricated in-house at the first stage. The two types of the components should be prepared at the first stage before starting the assembly operation for each job at the second stage. The objective is to schedule the jobs so that the makespan is minimized. Some solution properties and lower bounds are derived and incorporated into a branch and bound algorithm. Also, an efficient heuristic is developed. The performances of the proposed branch and bound algorithm and heuristic are evaluated through computational experiments.

Keywords : Assembly-Type Flowshop, Scheduling, Heuristic, Branch-and-Bound

1. 서 론

조립형 flowshop 일정계획(assembly-type flowshop scheduling; ATF)문제는 부품생산단계(단계 1)와 부품조립단계(단계 2; 완제품생산 단계)의 두 단계가 직렬 흐름라인을 구성한 형태의 문제를 말한다[8, 9]. 부품생산단계는 완제품 조립에 필요한 여러 유형의 부품들을 유형별로 독립적으로 생산하는 다수의 병렬기계들(dedicated parallel machines)로 구성되어 있고, 부품조립단계에는 이전 단계에서 생산(준비)완료된 부품들을 조립하여 완제품을 생산하는 한 대의 조립기계가 존재한다. 본 논문의 목적은 이러한 ATF 라인에서 외주부품 조달시간 제약을 고려하면서 최대작업완료시간(maximum completion time; makespan)을 최소화하는 작업일정계획을 찾는 것이다.

현실 생산공정에서 ATF의 사례는 소방차 등의 특수목적차량 조립공정[8], FMC(flexible manufacturing cell, [10]), 컴퓨터 조립공정[9] 등 다양하게 찾을 수 있다.

일정계획이론분야에서 ATF라인을 대상으로 확정적(deterministic) 일정계획문제를 다룬 연구는 Lee et al.[8], Potts et al.[9], Hariri and Potts[3], Koulamas and Kyparisis [6], Kovalyov et al.[7], Tozkpan et al.[11], Juhn et al.[5] 등이 있다. Lee et al.[8]은 두 대의 부품생산기계와 한 대의 조립기계로 연결된 상황에서 최대작업완료시간을 최소화하는 ATF문제를 최초로 소개하였다. Sun et al. [10]은 동 문제에 대해 다양한 발견적 해법(heuristic)들을 제시하였다. Potts et al.[9]은 이 문제를 부품생산기계가 여러대인 경우로 일반화 하였고, 이렇게 일반화된 문제는 이후 Hariri and Potts[3], Koulamas and Kyparisis [6], Kovalyov et al.[7], Tozkpan et al.[11] 등에서 다루어졌다. Hariri and Potts[3]는 분지한계(branch-and-bound) 알고리즘을 제시하였으며, Koulamas and Kyparisis[6]는 이 문제를 3단계 조립일정계획 문제로 확장하였다. 또, Kovalyov et al.[7]은 동일 문제상황에서 배칭(batching) 의사결정을 고

려하였다. 이상의 문제들이 모두 ATF에서 최대작업완료시간을 최소화하는 것을 목적함수로 고려한 반면에 Tozkapan et al.[11]은 ATF에서 총가중흐름시간(total weighted flowtime)을 최소화하는 분지한계 알고리즘을 제시한 바 있다.

이상에서 언급한 연구들에서는 완제품 조립에 사용되는 부품들을 모두 자체적으로 생산한다고 가정하거나, 외주(outsourcing)로 조달하는 경우에도 외주조달시간이 일정계획상의 제약으로 작용하지 않는다라고 묵시적으로 가정하고 있다. 예를 들어, Lee et al.[8]의 연구에서 고려한 소방차 조립의 사례에서는 엔진, 차체(body), 차대(chassis)의 세가지 부품이 소요되는데, 이들중 차체와 차대는 자체생산하고 엔진은 외주를 통해 조달하는 상황을 고려하였다. 그런데, 엔진의 외주조달은 적시(just-in-time)에 이루어진다는 가정하에 외주조달시간은 고려대상에서 제외한 후, 자체생산부품의 생산시간과 조립시간만으로 구성된 문제를 연구대상으로 하였다. 그러나, 완제품의 특성이나 소요기술, 산업구조 및 시장환경에 따라 외주조달이 적시에 이루어지지 않는 환경이 존재하므로 외주조달 시간제약을 함께 고려한 ATF 연구가 필요하다. 특히, 외주부품 생산업체에 대한 조립업체의 지배력이 상대적으로 약하거나, 외주부품 자체가 주문설계생산(ETO; engineer-to-order)에 속한 경우이면 적시조달의 가능성은 낮아진다[1]. 이러한 상황을 고려하여 Juhn et al.[5]은 외주부품 조달시간 제약을 고려한 ATF문제를 최초로 제시하고, 최대작업완료시간을 최소화하는 상황에서 이 문제가 NP-complete 문제임을 규명하였다.

외주부품 조달시간 제약을 고려한 ATF상황에서 최대작업완료시간을 최소화하는 일정계획문제의 난이도가 NP-complete 임이 Juhn et al.[5]의 연구에서 규명되었으므로, 본 연구에서는 이 문제의 근사해를 효율적으로 제공할 수 있는 발견적 해법 및 동 문제의 작업갯수가 작은 경우 최적해를 도출하는 분지한계 알고리즘을 제시하였다. 또, 임의로 생성된 수치자료를 사용해서 두 알고리즘의 성능평가를 수행하였다.

본 연구의 구성은 다음과 같다. 2장에서는 연구대상 문제를 명확히 정의하고, 몇 개의 문제성질(solution property)들을 제시하였다. 3장에서는 분지한계 알고리즘의 한계규칙 및 발견적 해법의 성능평가에 활용될 세 개의 하한값(lower bound)들을 정리하였고, 4장에서는 분지한계 알고리즘과 발견적 해법을 제시하였다. 5장에서는 제시된 발견적 해법과 분지한계 알고리즘의 성능평가를 위한 다양한 수치실험 결과를 정리하였고, 마지막 6장에서는 결론 및 추후 연구방향에 대한 논의를 하였다.

2. 문제정의와 분석

처리할 n개의 작업(제품)이 있다. 각 작업은 모두 2단계의 처리(processing)를 통해 완결된다. 제 1단계에서는 두가지 유형의 부품(p유형 부품, r유형 부품)을 자체생산과 외주조달을 통해 준비하고, 제 2단계에서는 단계1에서 준비된 두가지 부품을 조립하여 해당 작업을 완성한다. 제 1단계에서 p유형 부품은 자체기계 M_p를 사용해 생산하고, r유형 부품은 외주를 통해 조달한다. 또, 제 2단계에서의 조립작업은 자체기계 M_q를 사용해 이루어진다. 제 2단계에서 작업 $i\,(i=1,\cdots,n)$의 조립을 시작하기 위해서는, 반드시 단계1에서 작업 i의 조립에 소요되는 두가지 유형의 부품이 준비되어 있어야만 한다. 작업 i에 대한 기계 M_p와 M_q에서의 처리시간은 각각 p_i와 q_i이고, 외주부품(r유형 부품)의 조달소요시간은 a_i이다. 각 작업의 완료시간은 조립기계 M_q에서의 조립완결시점으로 측정된다. 본 문제에서는 각 작업의 완료시간중에서 최대인 완료시간을 최소화하는 작업순서와, 그 최대작업완료시간을 구하는 것을 목적으로 한다.

본 문제에서는 다음과 같은 가정을 전제로 하고 있다.

- 모든 작업들은 시점 0에서 처리를 시작할 수 있다.
- 각 기계에서의 처리시간과 외부조달시간은 미리 알고 있으며, 확정적인 양의 정수이다.
- 각 기계에서 일단 처리가 시작된 작업은 중간에 중단될 수 없다.
- 각 기계는 한번에 한 개의 작업만 처리가능하다.
- 작업준비시간(setup time)은 고려하지 않는다.
- 부품준비단계인 단계 1에서 조립단계인 단계 2로 부품을 이동시키는 시간은 고려하지 않는다.

분석에 사용될 추가적인 기호들은 다음과 같다.

- i, j : 각각의 작업을 표시하는 첨자(subscript)들.
- $[i]$: 순서가 결정된 작업순열에서 i번째 처리되는 작업을 표시하는 순서표시첨자. 여기서, $i\in\{1,\cdots,n\}$.
- S : 임의의 일정계획해.
- S^* : 최적 일정계획해.
- $C_{\max}^S$: 임의의 일정계획해 S의 최대작업완료시간.

다음의 정리는 '최적일정계획해는 순열일정계획해(permutation schedule)들중에 존재한다'는 사실을 밝힌 것으로, 고려해야 할 해공간을 축소시키는 역할을 한다. 여기서, 순열일정계획해란 두 기계상에서 모든 작업들이

같은 순서로 처리되는 일정계획해를 말한다..

정리 1. 순열일정계획해의 집합이 우월집합(dominant set)을 구성한다.

증명. 모든 비순열일정계획해(non-permutation schedule)를 작업간 교환과정(pairwise interchange)을 통해 우월하거나 동일한 최대작업완료시간을 갖는 순열일정계획해로 항상 변환할 수 있음을 보임으로써 증명이 이루어진다. 기계 M_p와 M_q에서 작업순서가 동일하지 않은 임의의 비순열일정계획해 S를 고려하자. 즉, S를 구성하는 임의의 두 작업 J_i와 J_j가 M_p에서는 J_i가 J_j 바로 앞에서 처리되고, M_q에서는 반대순서(이 경우는 J_j가 J_i보다 앞서 처리되는 것을 의미하며, 반드시 바로 앞에서 처리될 필요는 없다)로 처리된다고 하자. 이때, M_p상에서 J_i와 J_j의 처리순서를 서로 맞교환하면 교환 후 최대작업완료시간이 교환전보다 악화되지는 않음을 쉽게 알 수 있다. 이와 동일하게, 두 기계에서 처리순서가 상이한 작업들에 대해 M_p상에서의 처리순서를 맞교환함으로써, 최대작업완료시간이 개선되거나 혹은 동일하게 유지되는 순열일정계획해로 변환시킬 수 있다.

다음의 정리는 '어떤 기계 앞에 처리해야 할 작업이 대기중일때는 해당기계가 유휴시간(삽입된 유휴시간; inserted idle time)을 갖지 않는 일정계획해가 우월함'을 말하고 있다.

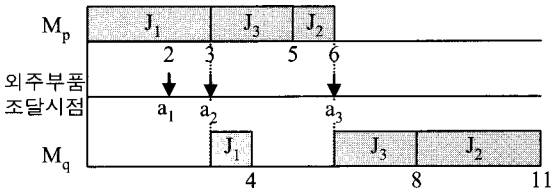
정리 2. 삽입된 유휴시간이 없는 일정계획해 집합이 우월집합을 구성한다.

증명. 위 정리 1과 유사하게, 삽입된 유휴시간을 갖는 일정계획해는 그 유휴시간을 제거함으로써 개선된 일정계획해를 항상 찾을 수 있음을 알 수 있다.

정리 1과 정리 2에 의해 고려해야 할 해공간의 크기가 $n!$이 된다. 또, 임의의 일정계획해 S의 최대작업완료시간은 다음과 같이 표현된다.

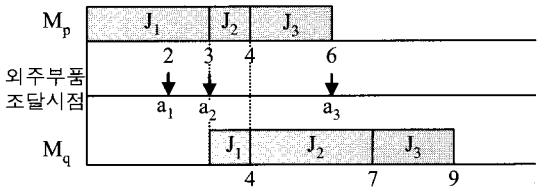
$$C^S_{\max} = \max_{u\in\{1,\cdots,n\}}\left[\max\left\{\sum_{i=1}^{u} p_{[i]}, a_{[u]}\right\} + \sum_{i=u}^{n} q_{[i]}\right] \cdots\cdots (1)$$

여기서, 식 (1)의 값을 동일하게 최대화하는 $u(u\in 1, \cdots, n)$가 여러개 존재하는 경우도 있는데, 이 경우에는 편의상 가장 작은 u값을 선택하여 사용하는 것으로 한다. 예를 들어, 세 개의 작업 $J_1=(3,1,2)$, $J_2=(1,3,3)$, $J_3=(2,2,6)$으로 주어진 수치예제를 고려하자. 여기서, $J_i=(p_i, q_i, a_i)$는 작업 i의 자체처리시간들과 외주조달시간을 나타낸다. 이때, 임의의 일정계획해 $S_1=(J_{[1]}, J_{[2]}, J_{[3]})=(J_1, J_3, J_2)$과 $S_2=(J_{[1]}, J_{[2]}, J_{[3]})=(J_1, J_2, J_3)$를 고려하자. 일정계획 S_1에 대해 식 (1)의 값을 결정해주는 값은 $a_{[2]}$ $(a_{[2]} > \sum_{i=1}^{2} p_{[i]})$이므로 u는 2로 단일하다(<그림 1-a> 참조). 반면, 일정계획 S_2에 대해 식 (1)을 결정하는 u값으로는 1과 2가 존재한다($p_{[1]} > a_{[1]}$, $\sum_{i=1}^{2} p_{[i]} > a_{[2]}$). 이런 경우에는 편의상 작은값인 1을 u값으로 사용한다는 것이다(<그림 1-b> 참조).



$$C^{S_1}_{\max} = a_{[2]} + \sum_{i=2}^{3} q_{[i]} = 6+(2+3) = 11$$

(a) S_1 : 외주조달시간(a_3)이 식 (1)을 결정하는 경우



$u=1$인 경우 : $C^{S_2}_{\max} = p_{[1]} + \sum_{i=1}^{3} q_{[i]} = 3+(1+3+2) = 9$

$u=2$인 경우 : $C^{S_2}_{\max} = \sum_{i=1}^{2} p_{[i]} + \sum_{i=2}^{3} q_{[i]} = (3+1)+(3+2) = 9$

(b) S_2 : 식 (1)을 결정하는 u가 다수 존재하는 경우

<그림 1> 두 일정계획해(S_1, S_2)의 Gantt Chart

해의 도출을 위한 본격적인 분석에 들어가기에 앞서, 본 문제가 NP-complete임에 주목해야 한다. Juhn et al. [5]의 연구에서 본 문제가 NP-complete임을 3-분할문제[2]를 활용해 증명한 바 있다. 이 사실은 본 문제의 경우 작업갯수가 커지면 가능한 알고리즘의 복잡도가 지수적으로 증가하는 난제이어서, 작업수가 큰 경우에는 주어진 시간제약내에 최적해를 구하는 것이 매우 어렵다는 것을 의미한다. 따라서, 본 연구에서는 작업수가 큰 경우에 효율적인 해를 제공할 수 있는 발견적 해법의 제안과, 작업수가 크지 않은 문제에 대해서는 최적해를 제공해주는 분지한계 알고리즘을 동시에 제안하고 실험을 통해 성능을 평가하였다.

다음 정리는 분지한계 알고리즘의 분지노드를 제거하는데 유용하게 사용되는 우월성질(dominance property)로

써, 역시 고려해야 할 해공간을 줄여주는 역할을 수행한다.

정리 3. 총 n개의 작업중 m개 $(m \leq n)$의 작업에 대해 이미 순서가 결정된 부분순열(partial permutation)을 σ라고 하고, 이 부분순열 σ를 구성하고 있는 작업들중에서 맨 뒤 두 개 작업만 서로 맞교환한 부분순열을 σ'이라고 했을 때, 다음 조건이 성립한다면 부분순열 σ'으로 시작되지 않는 최적일정계획(최적순열)이 존재한다.

$$C^{\sigma}_{\max} \leq C^{\sigma'}_{\max}. \quad \cdots\cdots (2)$$

증명. 부분순열 σ'으로 시작하는 최적일정계획이 존재한다고 가정하고 이를 $S^* = \sigma'\sigma''$이라고 하자. 여기서, σ''은 전체 작업중에서 σ'에 속하지 않는 작업들로 이루어지고, σ'이후에 위치한 부분순열을 표시한다. 다시 새로운 일정계획 $S = \sigma\sigma''$을 고려하자(여기서, σ는 본 정리의 전제조건에 의해, 맨 뒤 두 개 작업의 순서만 σ'과 서로 반대이고 나머지 작업들은 모두 동일한 부분순열이다.). 두 일정계획 S^*와 S 각각의 σ''에 속한 각 작업들의 기계 M_p에서의 처리완결시점은 동일하다. 그런데, 식 (2)에 의해 σ''에 속한 개별 작업들이 기계 M_q에서 완결되는 시점은 S에 속한 경우가 S^*에 속한 경우보다 빠르거나 같음을 알 수 있다. 따라서, 부분순열 σ'로 시작하지 않는 최적일정계획의 존재가 증명된다.

3. 하한값(Lower bounds)

본 장에서는 분지한계 알고리즘의 한계규칙 및 발견적 해법의 성능분석에 사용할 하한값들을 제시하였다. 다음의 정리는 본 연구에서 고려하는 문제에서 외주부품 조달시간제약을 완화시키면 전통적인 2-machine flowshop 문제와 일치하므로 이 사실을 이용하여 하한값을 제공할 수 있음을 말하고 있다.

정리 4. 본 연구에서 고려하고 있는 원래문제의 최적일정계획을 S^*라고 하고, 원래문제를 두 기계 M_p와 M_q로만 구성된 2-machine flowshop 문제로 변환(완화)한 후, Johnson의 알고리즘[4]을 적용하여 구한 일정계획을 S^*_J라고 하면 다음식이 항상 성립한다.

$$C^{S^*_J}_{\max} \leq C^{S^*}_{\max}.$$

증명. 원래문제의 최적일정계획을 S^*, 그리고 원래문제에서 외주부품 조달시간제약을 제거(완화)하여 2-machine flowshop문제로 변환한 문제에 Johnson의 알고리즘을 사용해서 구한 일정계획을 S^*_J로 각각 표시하기로 하자. 여기서, 원래문제와 변환된 문제(2-machine flowshop 문제) 모두 최대작업완료시간을 최소화하는 문제이고, Johnson의 알고리즘은 변환된 문제(제약이 완화된 문제)의 최적일정계획을 제공한다는 사실[4]을 함께 고려하면 $C^{S^*_J}_{\max} \leq C^{S^*}_{\max}$가 항상 성립함을 알 수 있다.

정리 4로부터 첫 번째 하한값(LB_1)을 다음과 같이 표기할 수 있다.

$$LB_1 = \max_{u \in \{1,\cdots,n\}} \left\{ \sum_{i=1}^{u} p_{[i]} + \sum_{i=u}^{n} q_{[i]} \right\}.$$

여기서, $p_{[i]}$와 $q_{[i]}$는 각각 원문제에서 외주부품 조달시간을 고려하지 않은 변환된 문제에 Johnson의 알고리즘을 적용하여 구한 일정계획해에서 i번째에 위치한 작업에 대한 기계 M_p와 M_q에서의 작업시간을 각각 의미한다. LB_1은 자체처리 작업시간에 중심을 둔 하한값이다. 다음 정리는 조립기계 M_q에서 최초로 작업을 시작할 수 있는 가장 빠른 시간은 제 1단계에서 부품이 준비될 수 있는 시간중 최소값보다는 작지 않다는 사실을 이용하고 있다.

정리 5. 본 연구에서 고려하는 문제의 최적일정계획을 S^*라고 할 때, 다음 식이 항상 성립한다.

$$C^{S^*}_{\max} \geq \min_{i \in \{1,\cdots,n\}} [\max\{p_i, a_i\}] + \sum_{j=1}^{n} q_j.$$

증명. 본 연구 문제의 최적일정계획을 S^*라고 하고, S^*의 i번째 작업에 대한 기계 M_p와 M_q에서의 작업시간을 각각 $p_{[i]}$와 $q_{[i]}$라고 하면 식 (1)로부터 다음식이 성립한다.

$$C^{S^*}_{\max} = \max_{u \in \{1,\cdots,n\}} \left[\max\left\{ \sum_{i=1}^{u} p_{[i]}, a_{[u]} \right\} + \sum_{i=u}^{n} q_{[i]} \right].$$

여기서, 위 식의 우변을 최대화하는 미지수 u를 대신하여 정수 1을 사용하면 다음의 첫 번째 부등식이 성립한다.

$$\begin{aligned} C^{S^*}_{\max} &\geq \max\{p_{[1]}, a_{[1]}\} + \sum_{i=1}^{n} q_{[i]} \\ &\geq \min_{i \in \{1,\cdots,n\}} [\max\{p_i, a_i\}] + \sum_{i=1}^{n} q_i. \end{aligned}$$

위 두 번째 부등식은 다음의 두 관계식
$\max\{p_{[1]}, a_{[1]}\} \geq \min_{i \in \{1,\cdots,n\}} [\max\{p_i, a_i\}]$와
$\sum_{i=1}^{n} q_{[i]} = \sum_{i=1}^{n} q_i$이 성립함에 따라 도출된 것이고 이로부터

본 정리의 부등식이 증명된다.

정리 5로부터 두 번째 하한값(LB_2)를 다음과 같이 나타낼 수 있다.

$$LB_2 = \min_{i\in\{1,\cdots,n\}}[\max\{p_i, a_i\}] + \sum_{j=1}^{n} q_j.$$

두 번째 하한값 LB_2는 조립기계 M_q의 처리시간에 중심을 두고 있다.

정리 **6.** 본 연구문제의 최적일정계획을 S^*라고 하고, $k = \arg\max_{i\in\{1,\cdots,n\}}\{a_i\}$가 n개의 외주조달시간(a_i, $i=1, \cdots, n$) 중에서 최대값을 갖는 작업을 의미한다고 할 때, 다음 식이 항상 성립한다.

$$C_{\max}^{S^*} \geq a_k + q_k. \quad \cdots\cdots (3)$$

증명. $k = \arg\max_{i\in\{1,\cdots,n\}}\{a_i\}$를 만족하는 작업 J_k가 최적일정계획 S^*에서 m번째 처리된다고 가정하자. 즉, $q_{[m]} = q_k$, $a_{[m]} = a_k$가 된다. 또, 2장에서 제시한 식 (1)로부터 최적일정계획 S^*의 최대작업완료시간은 다음과 같이 표현된다.

$$C_{\max}^{S^*} = \max_{u\in\{1,\cdots,n\}}\left[\max\left\{\sum_{i=1}^{u} p_{[i]}, a_{[u]}\right\} + \sum_{i=u}^{n} q_{[i]}\right].$$

여기서, 위 식을 결정하는 u대신 J_k의 S^*내에서의 작업위치 $m(m\in\{1,\cdots,n\})$을 사용하면 다음의 첫 번째 부등식이 성립한다(max함수 성질). 아래에서 두 번째와 세번째의 부등식은 각각 'max함수 성질' 및 '각 작업의 조립시간은 양의 정수라는 가정'에 기반한 것이다.

$$\begin{aligned} C_{\max}^{S^*} &\geq \max\left\{\sum_{i=1}^{m} p_{[i]}, a_{[m]}\right\} + \sum_{i=m}^{n} q_{[i]} \\ &\geq a_{[m]} + \sum_{i=m}^{n} q_{[i]} \geq a_{[m]} + q_{[m]} = a_k + q_k. \end{aligned}$$

위 식으로부터 식 (3)이 성립함을 알 수 있다.

정리 6으로부터 특정 부품의 외주조달 시간제약이 매우 큰 경우를 고려한 하한값 LB_3를 다음과 같이 표기할 수 있다.

$$LB_3 = \max_{i\in\{1,\cdots,n\}}\{a_i\} + q_{\arg\max_{i\in\{1,\cdots,n\}}\{a_i\}}.$$

이후에 논의되는 분지한계 알고리즘의 한계규칙과, 발견적 해법에 의해 도출된 근사해의 성능실험에서는 이상의 하한값들중 가장 큰값을 이용하는 최대 하한값 $LB = \max\{LB_1, LB_2, LB_3\}$를 사용한다. 특히, 분지한계 알고리즘의 한계규칙에서 하한값을 이용할 때에는 각 노드가 작업들의 부분순열(전체작업중 일부 작업의 순서가 결정된 순열)을 나타낸다는 사실을 감안하여 하한값을 계산하여야 한다.

4. 분지한계 알고리즘

4.1 분지규칙(branching rule)

정리 1과 정리 2에 의해, 본 문제는 $n!$개의 순열일정계획중에서 최적순열을 찾는 문제로 축소된 바 있다. 본 연구에서는 $n!$개의 순열일정계획을 생성해내기 위해 순방향순서결정 분지규칙(forward sequencing branching rule)을 사용한다. 즉, 탐색트리의 l번째 계위 ($l \leq n$)에 존재하는 노드(node)들에서는 순열의 l번째에 위치할 작업을 결정한다. 따라서, l번째 계위에 존재하는 각 노드는 첫 번째부터 l번째 처리될 작업까지만 순서가 결정된 (부분)순열을 나타낸다.

분지단계에서는 이렇게 생성된 노드들중에서 분지할 하나의 노드(부분순열)를 선택한다. 분지할 노드의 선택에는 일차적으로 깊이우선탐색(depth-first search)방법을 사용하는데, 이는 탐색트리에서 가장 큰 부분순열을 가진 노드를 분지노드로 선택함을 의미한다. 또, 동일한 크기의 부분순열이 다수 존재하는 경우에는 하한값이 가장 작은 노드가 선택된다(best-first search).

4.2 한계규칙(bounding rule)

한계규칙은 정리 3에서 제시한 우월성질 및 각 노드의 상한값(upper bound)과 하한값을 이용하여, 현재노드의 제거(fathoming)여부를 결정함으로써, 최적해로의 수렴을 촉진시키는 역할을 수행한다. 본 연구의 분지한계 알고리즘에서는 초기 상한값(initial upper bound)으로 다음의 4.3절에서 제시하는 발견적 해법의 목적함수값을 사용한다. 이후의 탐색에서 보다 우월한 가능해가 얻어지면 상한값은 갱신된다. 또, 하한값으로는 3장에서 제시한 하한값들을 적용하여, 각 노드의 부분순열을 고려한 형태로 계산한 후 최대 하한값을 해당 노드의 하한값으로 사용한다. 이렇게 계산된 하한값이 상한값 이상의 값을 갖는 노드는 제거된다. 또, 하한값이 상한값 미만인 노드에 대해서는 정리 3을 적용하여 제거여부를 결정한 후, 이 과정에서도 제거되지 않은 노드들에 대해서는 다시 분지를 수행한다.

4.3 발견적 해법

본 연구에서 고려하는 문제의 난이도가 NP-complete 임은 전술한 바 있다. 따라서, 작업수 n이 큰 문제에 대해서 효율적인 해를 산출하고, 동시에 분지한계 알고리즘의 초기 상한값을 제공한다는 두가지 측면에서 발견적 해법의 개발은 필수적이다. 발견적 해법이 제공하는 가능해가 최적해에 근사할수록 분지한계 알고리즘의 성능이 높아지므로, 우수한 발견적 해법의 개발은 매우 중요하다.

본 연구에서 고려하는 문제에서 외주부품 조달시간제약을 완화하면 전통적인 2-machine flowshop문제와 동일하고, 이 문제의 경우 Johnson의 알고리즘으로 최적해가 구해짐은 잘 알려진 사실이다[4]. 따라서, 본 연구에서는 Johnson의 알고리즘에 외주부품 조달시간제약을 결합시킨 발견적 해법을 제안한다. 즉, 일부작업의 순서결정이 완결된 상황에서 아직 작업순서가 정해지지 않은 작업들중 하나를 선택한다고 하자. 이 경우에 외주부품 조달시간제약(a_i, $i\in\sigma$, 여기서, σ는 순서미결정 작업들의 집합을 표시)이 없거나 혹은 이 제약이 영향을 미치지 못한다고 가정하면 기존의 Johnson 법칙에 맞는 작업을 선택하는 것이 최선이다. 따라서, 가능하면 현재 상황에서 순서미결정 작업들의 a_i가 작업순서결정에 영향을 미치지 못하는 작업들만을 선별하고, 이들에 대해서 Johnson의 법칙을 적용하여 직후에 처리할 작업을 선택하는 방법을 사용한다. 전체적인 발견적 해법의 과정은 다음과 같이 표현된다. 발견적 해법에서 사용된 C_p와 C_q는 현재시점에서 순서가 결정된 작업들중 최종작업이 기계 M_p와 M_q에서 종료되는 시점을 각각 나타내고, σ는 현재시점까지 순서가 결정되지 않은 작업들의 집합을 의미한다.

<Johnson 법칙기반 휴리스틱>

단계 **0**. (초기화)

- $u=0$, $C_p=0$, $C_q=0$, $\sigma=\{1,\cdots,n\}$.

단계 **1**. (할당 작업 찾기)

- 기준시점 R을 계산한다. 여기서, $R=\max\left[C_q,\min_{j\in\sigma}\{C_p+p_j\},\min_{j\in\sigma}\{a_j\}\right]$.
- 순서미결정 작업들중 기준시점 R이하의 a_i값을 갖는 작업들을 선별한다(다음에서, 이 작업들에 대해 Johnson의 법칙을 적용하여 현 위치에 할당할 작업을 찾는다). 즉, 작업집합 $\Omega=\{j\mid j\in\sigma, a_j\le R, p_j\le q_j\}$와 $\tilde{\Omega}=\{j\mid j\in\sigma, a_j\le R, p_j>q_j\}$를 구한다.
- 여기서, $\Omega\neq\varnothing$이면 Ω의 원소들중 가장 큰 q_j값을 갖는 작업을 선택하고, $\Omega=\varnothing$이면 $\tilde{\Omega}$의 원소들 중 가장 큰 q_j값을 갖는 작업을 선택한다(선택된 작업을 편의상 작업 x로 표기한다.).

단계 **2**. (선택작업 할당과 중요값 수정)

- $u=u+1$로 갱신한 후, u번째 위치에 작업 x를 할당한다.
- $C_p=C_p+p_x$, $C_q=\max\{C_p, C_q, a_x\}+q_x$ 그리고 $\sigma=\sigma-x$로 수정한다.

단계 **3**. (종료조건 검사)

- $\sigma=\varnothing$이면 종료하고, 그렇지 않으면 단계 1로 간다.

5. 계산실험 및 성능평가

본 절에서는 분지한계 알고리즘과 Johnson법칙기반 휴리스틱의 성능을 다양한 수치실험을 통해 분석한다. 알고리즘들은 C언어로 프로그래밍 되었고, Pentium IV 3GHz PC에서 실행되었다. 실험에 사용된 수치의 생성방법은 다음과 같다.

(1) 각 작업의 부품가공시간(p_i)과 조립시간(q_i)

- p_i와 q_i의 변이가 상대적으로 큰 문제유형(Type 1 문제) : $U[1,50]$으로부터 생성, 여기서 $U[a,b]$는 a와 b를 모수로 갖는 이산형 균등분포(discrete uniform distribution)을 의미.
- p_i와 q_i의 변이가 상대적으로 작은 문제유형(Type 2 문제) : $U[\rho+1,\rho+10]$으로부터 생성, 여기서 ρ값은 $U[1,50]$으로부터 생성.

(2) 각 작업의 외주부품 조달시간 (a_i) : $U[1,\alpha P]$로부터 생성, 여기서 $P=\sum_{i=1}^{n}p_i$, $\alpha\in\{0.4, 0.6, 0.8, 1.0\}$.

- α는 외주부품 조달시점의 산포를 조절하기 위한 계수, α값이 클수록 외주부품의 조달시간 편차가 커짐

이하에서는 2단계로 대별해서 계산실험이 이루어졌다. 즉, 일차적으로는 적은 수의 작업을 갖는 문제들에 대해 분지한계 알고리즘을 이용한 성능실험과, 분지한계 알고리즘이 도출한 최적해에 대비한 발견적해의 상대오차 분석을 통해 발견적 해법의 성능을 검토한다. 다음 단계로는, 상대적으로 작업수가 큰 문제들에 대해 발견적 해법을 사용해 얻은 근사해와 3장에서 제시한 하한값과의 상대오차를 검토함으로써 발견적 해법의 효과성을 살펴본다.

제 1단계 실험에서는 작업수(n)을 5종류(10, 20, 30,

40, 50)로 선택하여, 위에서 논의한 8가지 경우(Type 1, Type 2 문제 각각에 대해 4종류의 α값 사용)에 대해 각각 고려함으로써 총 40가지 문제조합을 생성하였다. 각각의 문제조합에 대해서 30개씩의 수치집합을 임의로 생성함으로써 1단계 실험에 총 1,200개의 임의생성 문제가 사용되었다. 분지한계 알고리즘의 성능에 대한 실험결과는 <표 1>에 요약되어 있다. 표에서 "no"는 30개의 문제중 60초(CPU시간)내에 최적해를 찾은 문제수를 의미한다.

<표 1> 분지한계 알고리즘을 사용한 계산실험 결과

n	α	Type 1 문제			Type 2 문제		
		no	평균 계산시간	평균 탐색노드수	no	평균 계산시간	평균 탐색노드수
10	0.4	30	≪0.01(*)	4.9	30	≪0.01	1.3
	0.6	30	≪0.01	5.3	30	≪0.01	1.6
	0.8	30	≪0.01	8.6	30	≪0.01	3.6
	1.0	30	≪0.01	36.7	30	≪0.01	9.7
20	0.4	30	≪0.01	11.1	30	≪0.01	2.6
	0.6	30	≪0.01	14.4	30	≪0.01	6.5
	0.8	30	0.04	557.1	30	≪0.01	17.6
	1.0	30	1.80	57,396.4	30	0.89	22,810.8
30	0.4	30	≪0.01	4.9	30	≪0.01	6.9
	0.6	30	≪0.01	12.6	30	≪0.01	11.8
	0.8	30	0.02	100.5	30	0.01	65.8
	1.0	21	N/A(**)	N/A	28	N/A	N/A
40	0.4	30	≪0.01	11.7	30	≪0.01	6.6
	0.6	30	0.01	24.1	30	0.01	11.9
	0.8	28	8.24	20,188.2	30	0.36	915.4
	1.0	17	N/A	N/A	21	N/A	N/A
50	0.4	30	0.01	14.8	30	0.01	15.3
	0.6	30	0.17	193.5	30	0.01	15.2
	0.8	27	N/A	N/A	30	1.84	2,849.1
	1.0	14	N/A	N/A	16	N/A	N/A

주) n : 작업수
α : 외주부품 조달시점 산포조절 계수
no : 30개의 실험문제중에서 60초내에 최적해를 찾은 문제수
(*) : 평균계산시간이 0.01초보다 매우 작음을 의미
(**) : 30개의 문제중에서 7,200초를 초과하여도 최적해를 찾지 못하는 문제가 일부 존재하여 통계치를 제공하지 못하였음.

<표 1>의 실험결과에서 나타나는 전반적인 경향은 다음과 같다. 첫째, 작업수가 20개 이하인 경우에는 제안된 분지한계 알고리즘이 60초내에 최적해를 모두 찾아주는 반면, 그 이상인 경우에는 α(외주부품 조달시점 산포조절 계수)가 커질 수록 최적해를 찾는 문제수가 감소한다. 두 번째로, 근소한 차이이지만 Type 1 문제보다 Type 2 문제에 대해 최적해를 상대적으로 잘 찾아주고 있음을 볼 수 있다.

다음으로, 앞에서 제안한 발견적 해법의 평균적인 성능평가를 위한 실험을 수행하였다. 실험에 사용된 수치데이터는 분지한계 알고리즘 계산실험에서 사용한 것들과 동일하다. 성능평가의 척도로는 평균상대오차, 최대상대오차, 그리고 최적해산출횟수(no; number of problems solved optimally)를 사용하였다. 상대오차(re; relative error)는 $re(\%) = 100 \times (Heu - Opt)/Opt$를 사용해 계산되었다. 여기서, Heu는 발견적 해이고, Opt는 최적해를 각각 의미한다. 이러한 척도를 사용한 발견적 해법의 성능평가 결과가 <표 2>에 요약되어 있다. 전체적으로 발견적 해법이 상당히 우수한 성능을 보이고 있음을 알 수 있다. 또, 근소한 차이지만 Type 1 문제보다 Type 2 문제가 전반적으로 오차가 적게 나타나고 있음을 볼 수 있다.

<표 2> 발견적 해의 최적해 대비 상대오차

(단위 : %)

n	α	Type 1 문제			Type 2 문제		
		no	평균 상대오차	최대 상대오차	no	평균 상대오차	최대 상대오차
10	0.4	30	1.66	9.54	30	0.14	3.16
	0.6	30	1.28	4.96	30	0.20	3.33
	0.8	30	1.42	7.36	30	0.33	4.88
	1.0	30	0.81	9.37	30	0.00	0.00
20	0.4	30	1.35	5.48	30	0.08	2.01
	0.6	30	0.98	8.67	30	0.34	4.61
	0.8	30	0.98	4.94	30	0.03	0.39
	1.0	30	0.83	6.76	30	0.07	1.70
30	0.4	30	0.27	2.95	30	0.12	0.96
	0.6	30	0.44	3.73	30	0.09	1.11
	0.8	30	0.63	5.08	30	0.03	0.75
	1.0	21	N/A(*)	N/A	28	N/A	N/A
40	0.4	30	0.36	2.10	30	0.06	0.70
	0.6	30	0.35	3.21	30	0.06	1.03
	0.8	28	0.84	7.07	30	0.07	0.88
	1.0	17	N/A	N/A	21	N/A	N/A
50	0.4	30	0.30	2.36	30	0.06	0.88
	0.6	30	0.28	2.39	30	0.10	1.00
	0.8	27	N/A	N/A	30	0.04	0.32
	1.0	14	N/A	N/A	16	N/A	N/A

주) (*) : 30개의 문제 중에서 7,200초를 초과하여도 최적해를 찾지 못하는 문제가 일부 존재하여 통계치를 제공하지 못하였음.

작업의 수가 커지는 경우에는, 분지한계법을 사용해 제한된 시간내에 최적해를 구하지 못하는 경우가 빈번히 발생하므로, 3장에서 제시한 하한값과의 비교를 통해 발견적 해법의 성능평가를 수행하였다. 따라서, 이 경우의 척도는 하한값(LB; lower bound value)과 대비한 평균상대오차와 최대상대오차를 사용하였다.

이때 사용된 상대오차 계산식은 $re(\%) = 100 \times (Heu - LB)/LB$이다. 비교적 작업수 n이 큰 값들(5종류; 100, 300, 500, 700, 900)을 갖는 40개 문제조합들에 대해 각각 30개씩 임의생성된 1,200개 수치문제들을 사용해서 계산실험을 수행한 결과가 <표 3>에 나타나 있다.

<표 3> 발견적 해의 하한값 대비 상대오차

(단위 : %)

n	α	Type 1 문제		Type 2 문제	
		평균 상대오차	최대 상대오차	평균 상대오차	최대 상대오차
100	0.4	0.26	2.26	0.04	0.49
	0.6	0.40	2.61	0.07	1.33
	0.8	0.84	10.04	0.42	2.64
	1.0	2.57	11.77	4.27	11.48
300	0.4	0.13	0.73	0.02	0.09
	0.6	0.11	0.79	0.01	0.13
	0.8	0.33	2.05	0.19	1.37
	1.0	2.33	8.72	2.00	7.74
500	0.4	0.04	0.34	0.00	0.05
	0.6	0.05	0.32	0.02	0.24
	0.8	0.13	1.38	0.09	0.9
	1.0	0.38	1.32	0.76	2.52
700	0.4	0.04	0.31	0.01	0.18
	0.6	0.03	0.20	0.01	0.12
	0.8	0.08	0.55	0.02	0.23
	1.0	0.63	3.02	0.67	3.91
900	0.4	0.04	0.35	0.00	0.01
	0.6	0.03	0.17	0.01	0.07
	0.8	0.05	0.28	0.02	0.18
	1.0	0.15	0.50	0.23	1.64

<표 3>에 나타난 바와 같이, 제안된 발견적 해법은 작업수가 큰 경우에도 매우 우수한 성능을 보이고 있다. 특히, 작업수가 커질 수록, α가 작을 수록 오차가 작게 나타나고 있다. 또, Type2문제의 오차가 근소하게 Type1 문제보다 작게 나타나는데, 이는 이전의 결과들과 일관되게 나타나는 현상이다.

6. 결 론

본 연구에서는 두 종류의 부품중 한 종류는 자체생산하고, 다른 한 종류는 외주를 통해 조달한 후, 이들이 하나의 주조립 공정을 통해 완제품으로 조립되는 2단계 조립형 흐름라인에서의 일정계획문제를 다루었다. 최대작업완료시간을 목적함수로 하는 문제에 대해 몇가지 해의 특성 및 하한값들이 규명되었고, Johnson규칙에 기반한 발견적 해법과 분지한계 해법을 제시한 후 계산실험을 통해 성능을 평가하였다.

계산실험 결과로부터 제안된 분지한계 알고리즘은 작업수가 대략 20개 정도까지로 구성된 문제에 대해서는 최적해를 제공할 수 있으며, 발견적 해법은 작업수에 무관하게 오차가 매우 적은 근사해를 제공함을 알 수 있었다. 또, 전반적으로 내부처리시간이나 외주부품조달시간의 변이가 적을수록 두 알고리즘의 성능이 향상되는 경향이 나타남을 볼 수 있었다.

본 연구결과는 자체생산 및 외주 부품의 수가 여러개인 경우로 확장될 수 있을 것으로 예상되며, 총작업완료시간이나 납기와 관련된 목적함수를 가지는 문제 등이 추후의 연구과제가 될 수 있을 것이다.

참고문헌

[1] Bertrand, J. W. M. and Muntslag, D. R.; "Production control in engineer-to-order firms," *International Journal of Production Economics*, 30-31 : 3-22, 1993.

[2] Garey, M. R. and Johnson, D. S.; *Computers and Intractability : A Guide to the Theory of NP-Completeness*, W.H. Freeman and Company, San Francisco, 1979.

[3] Hariri, A. M. A. and Potts, C. N.; "A branch and bound algorithm for the two-stage assembly scheduling problem," *European Journal of Operational Research*, 103 : 547-556, 1997.

[4] Johnson, S. M.; "Optimal two- and three-stage production schedules with setup times included," *Naval Research Logistics Quarterly*, 1 : 61-68, 1954.

[5] Juhn, J., Sung, C. S., and Yoon, S. H.; "Analysis of heuristics for a two-stage assembly scheduling problem with component available time constraint," Submitted to Operations Research Letters for publication, 2006.

[6] Koulamas, C. and Kyparisis, G. J.; "The three-stage assembly flowshop scheduling problem," *Computers & Operations Research*, 28 : 689-704, 2001.

[7] Kovalyov, M. Y., Potts, C. N., and Strusevich, V. A.; "Batching decisions for assembly production systems,"

European Journal of Operational Research, 157 : 620-642, 2004.

[8] Lee, C.-Y., Cheng, T. C. E., and Lin, B. M. T.; "Minimizing the makespan in the 3-machine assembly-type flowshop scheduling problem," *Management Science*, 39 : 616-625, 1993.

[9] Potts, C. N., Sevast'janov, S. V., Strusevich, V. A., Wassenhove, L. N. V., and Zwaneveld, C. M.; "The two-stage assembly scheduling problem : complexity and approximation," *Operations Research*, 43 : 346-355, 1995.

[10] Sun, X., Morizawa, K., and Nagasawa, H.; "Powerful heuristics to minimize makespan in fixed, 3-machine, assembly-type flowshop scheduling," *European Journal of Operational Research*, 146 : 498-516, 2003.

[11] Tozkapan, A., Kirca, O., and Chung, C.-S.; "A branch and bound algorithm to minimize the total weighted flowtime for the two-stage assembly scheduling problem," *Computers & Operations Research*, 30 : 309-320, 2003.