Society of Korea Industrial and Systems Engineering Vol. 37, No. 3 : 113-121, September 2014 ISSN : 2005-0461(print)
http://dx.doi.org/10.11627/jkise.2014.37.3.113 ISSN : 2287-7975(online)

A Two-Machine Flowshop Scheduling with Outsourcing Strategy Allowed

Ik Sun Lee†

School of Business, Dong-A University

아웃소싱 전략을 활용하는 두 단계 흐름생산라인의 일정계획

이 익 선†

동아대학교 경영학과

This paper considers a scheduling problem in a two-machine flowshop with outsourcing strategy incorporated. The jobs can be either processed in the first machine or outsourced to outside subcontractors. This paper wants to determine which jobs to be processed in-house and which jobs to be outsourced. If any job is decided to be outsourced, then an additional outsourcing cost is charged The objective of this paper is to minimize the sum of scheduling cost and outsourcing cost under a budget constraint. At first this paper characterizes some solution properties, and then it derives solution procedure including DP (Dynamic Programming) and B&B (Branch-and-Bound) algorithms and a greedy-type heuristic. Finally the performance of the algorithms are evaluated with some numerical tests.

Keywords : Scheduling, Outsourcing, Two-Machine Flowshop, Branch-and-Bound, Heuristic

1. 서 론

아웃소싱(Outsourcing)이란 기업의 업무나 기능을 보다 전문적인 외부업체에 의뢰하는 것을 뜻하며, 기업은 아웃소싱 전략을 도입함으로써 비용을 줄이고 업무의 리드타임을 감소시켜 시간과 비용 등의 측면에서 경쟁력 강화를 꾀할 수 있다.

아웃소싱은 근래에 보다 포괄적으로 업무의 프로세스와 기능을 위탁하는 방향으로 그 영역을 확대시켜나가고 있으며, 또한 기업의 핵심적인 기능들도 경우에 따라서 아웃소싱되기도 한다. 아웃소싱 전략은 기업의 역량향상을 저해하고 회사의 역할이 외부의 아웃소싱 업체에 맡겨지는 과정에서 여러 위험이 존재하기도 하지만, 비용의 절감으로 인한 재무적인 부담을 줄일 수 있으며, 결국 타이밍이 기업의 경쟁역량이 되는 상황에서 필수적인 옵션이라고 말할 수 있다.

본 논문은 생산설비의 아웃소싱 전략을 활용할 수 있는 환경에서 일정계획 문제를 논의하고자 한다. 즉 작업의 가공이 자체적인 설비에서 직접 처리되거나, 아웃소싱으로 외부에 위탁되어 구매되어 조달되는 선택이 공존하는 경우에 먼저 작업들 중에서 아웃소싱에 해당되는 작업들을 선택(job selection)하고 자체적으로 처리되는 작업들의 순서를 결정(job scheduling)하는 문제를 다룬다. 생산설비는 두 설비를 가지는 2 Machine Flowshop을 고려하며 작업이 아웃소싱된다면 추가적으로 아웃소싱 비용을 지불해야 하며, 그 작업은 즉시로 조달된다. 본 논문은 이러한 아웃소싱 전략하에서 총 아웃소싱 비용이 예산제약(budget constraint)의 범위 내에서 아웃소싱 작업의 선별과 일정계획을 동시에 결정된다.

아웃소싱에 관한 기존의 연구들은 다음과 같다. 아웃

Received 21 August 2014; Accepted 12 September 2014
† Corresponding Author : lis1007@dau.ac.kr

소싱의 전략적인 성과를 평가한 연구들이 존재하는데, 즉 시스템의 운영 측면에서, 아웃소싱이 얼마나 비용적인 측면에서 이득인지를 분석한 연구들로서 Nicholson et al.[16], Kaipia and Tanskanen[9], Cachon and Harker[5] 등이 있다.

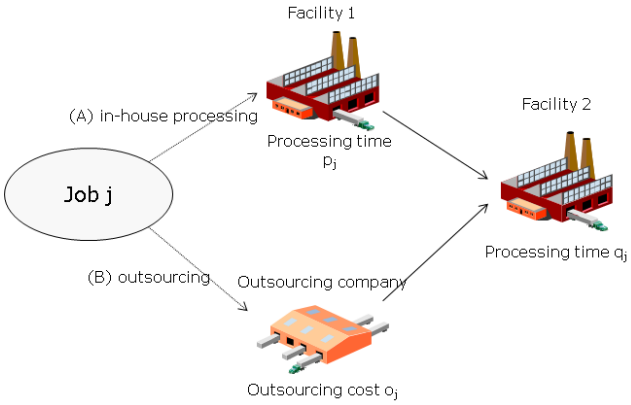
아웃소싱 전략을 보다 원활히 운영하기 위해 Framework을 제안하는 논문들이 있는데, Abdel-Malek[2]은 다층 공급사슬(multi-layered supply chains)의 상황에서 아웃소싱의 성능을 평가하였다. Wu[21], Kouvelis and Milner[12], Ngwenyama and Bryson[15]는 기업의 아웃소싱을 관리하기 위한 구조를 제안하였으며, Offodile and Abdel-Malek [17], Dekkers[7]는 경영관리기법들과 아웃소싱을 연계하는 구조를 제안했다.

아웃소싱의 효율을 향상시키기 위한 연구들이 있는데, Kim[11]은 아웃소싱 공급자 선정과 효율화를 연구하였다. Yang[22]은 아웃소싱 전략의 활용 하에서 재고 운용을 연구했으며, Huiskonen and Pirttila[8]는 아웃소싱 위탁회사와 본사간의 조정(coordination) 등의 문제들을 다루었다. Tarakci[19, 20]는 유지보수(maintenance activities)의 기능에 대한 아웃소싱 전략을 논의하였다.

본 논문의 위의 문헌연구와 비교해볼 때, 마지막 경우에 해당되는데, 아웃소싱 수단의 효율적인 활용을 목적으로 한다. 기존의 일정계획 논문들은 거의 작업들을 자체적으로 처리하는 상황을 고려하지만, 본 논문은 아웃소싱 전략을 활용하여 시스템 자체의 비용과 리드타임을 줄이는 일정계획을 수립코자 한다. 본 논문과 유사한 연구가 최근에 Lee and Sung[13, 14], Lee and Yoon[1]에 의해서 진행되었다. Lee and Sung[13, 14]는 단일설비의 상황에서 아웃소싱 전략을 고려하였으며, Lee and Yoon[1]은 병렬설비의 상황에서 아웃소싱 전략을 연구하였다.

Choi and Chung[4], Choi and Chung[6], Lee and Chung [10], Qi[18]은 는 두 단계의 흐름생산라인에서 아웃소싱을 고려하였는데, 본 논문과 유사하게 아웃소싱 비용과 작업완료시간의 총합을 최소화하는 일정계획 수립연구를 다루었다. 기존의 연구들은 작업을 아웃소싱하게 되면 두 단계의 흐름생산라인에서의 모든 처리공정을 위탁하지만, 본 논문은 첫 번째 설비에서의 가공을 외부의 전문업체에 위탁할지, 혹은 직접적으로 처리할지를 의사결정하는 상황을 고려한다. 본 논문에서 고려하는 문제의 작업흐름 구조는 다음의 <Figure 1>을 참조할 수 있다.

본 논문에서 고려하는 문제가 NP-complete임을 고려하여 비교적 작은 크기의 문제에 대해 최적해를 구해줄 수 있는 동적계획 알고리즘과, 상대적으로 큰 문제에 대해 최적해를 구해주는 분지한계 알고리즘, 동시에 근사해를 제공해주는 휴리스틱 알고리즘을 제안코자 한다.



<Figure 1> The Job-Process Work-Flow in this Paper

2. 문제정의와 분석

본 논문은 처리할 n개의 작업을 고려하며, 두 설비가 연속적으로 배열되어 있는 flowshop을 고려한다. 첫 번째 설비에서의 처리가 자체적으로 이행되거나 외부업체에 아웃소싱될 수 있다. 두 번째 설비에서는 반드시 자체적으로 처리함을 가정한다. 작업 $i(i=1, 2, \cdots, n)$가 첫 번째 설비에서 자체적으로 가공되는 경우에는 p_i 시간이 소요되고, 만일 작업 i가 아웃소싱된다면 아웃소싱 비용 o_i가 발생한다. 외부에 아웃소싱되면 즉시에 중간가공품을 구매해오는 것으로 상정할 수 있으며, 이는 즉시에 이루어질 수 있음을 의미한다. $\tilde{\pi}$는 자체 처리되는 작업들의 집합이고, π가 아웃소싱되는 작업들이라고 볼 때, 아웃소싱의 총비용 $OC(\pi)$는 $\sum_{i\in\pi} o_i$이다. 두 번째 설비에서의 처리 소요시간은 q_i으로 가정하는데, 두 번째 설비에서는 아웃소싱 전략을 고려하지 않는다. 즉 첫 번째 설비에서 아웃소싱 되었어도 두 번째 설비에서는 모든 작업들이 직접 처리됨을 원칙으로 한다. 본 논문은 "아웃소싱 예산제약"을 고려하는데, 아웃소싱 총 비용 $OC(\pi)$가 예산 K를 넘지 못한다는 뜻으로, $OC(\pi)=\sum_{i\in\pi} o_i \le K$으로 표현할 수 있다.

본 논문의 목적함수는 작업완료시간과 아웃소싱 소요비용의 합이다. 자체적으로 가공하는 작업들의 수가 증가하면 총 작업완료시간이 커지고, 아웃소싱 작업들의 수가 늘어나면 아웃소싱 비용이 늘어난다. 따라서, 운영하는 측면에서는 두 비용간의 trade-off 관계를 감안하여 총 비용을 최소화하는 의사결정을 수립해야 한다. 본 논문이 고려하는 총 비용 TC는 다음과 같다.

$$TC = \delta\sum_{i\in\pi} o_i + (1-\delta)C_{\max} \qquad (1)$$

$C_{\max}$는 작업완료시간을 뜻하고, δ는 아웃소싱 비용과 작업완료시간과 간의 중요도를 설정하기 위해 감안하기 위해 0과 1사이의 실수 값으로 가정하며, 이는 운영자가 설정할 수 있는 값으로 상정한다. 본 논문은 아웃소싱 예산 $OC(\pi) \leq K$ 범위 내에서, 총 비용을 최소화하는 일정계획을 수립하는 것인데, 구체적으로는 아웃소싱되는 작업들 π를 정하고, 나머지 작업들 $\tilde{\pi}$의 두 설비 Flowshop에서의 생산일정계획을 수립한다. 본 논문에서 고려하는 일정계획문제를 편의상 FSO(Flowshop scheduling with outsourcing)문제라고 명명하기로 상정한다. 그러면 FSO 문제의 수리모델은 다음처럼 정리될 수 있다.

[FSO 수리모형]

$$\text{Min} \quad \delta\sum_i o_i y_i + (1-\delta) C_{\max} \tag{2}$$

$$\text{s.t.} \quad C_{\max} \geq C_i \text{ for } \forall i \tag{3}$$

$$C_i \geq x_{ik}\sum_{l=1}^{k}\sum_{j=1}^{n} p_j x_{jl} + q_i \text{ for } \forall i,k \tag{4}$$

$$C_i \geq x_{ik}\sum_j x_{jk-1} C_j \text{ for } \forall i,k \tag{5}$$

$$\sum_i o_i y_i \leq K \tag{6}$$

$$\sum_k x_{ik} + y_i = 1, \text{ for } \forall i \tag{7}$$

$$x_{ij},\ y_i \in \{0,1\}.$$

수리모델에서의 변수 y_i는 작업 i가 아웃소싱 된다면 1의 값을 가지고, 그렇지 않으면 0 값을 가진다. 변수 x_{ik}는 작업 i가 첫 번째 설비에서 k번째 위치에서 직접 가공된다면 1의 값을 가지고, 그렇지 않으면 0 값을 가진다. C_i는 두 번째 설비에서 작업 i의 처리완료시간을 뜻한다. 식 (2)는 목적함수로서 아웃소싱 비용과 작업완료시간의 합을 최소화함을 뜻한다. 식 (3)~식 (5)는 일정계획문제에서 작업완료시간을 계산하기 위한 제약들이다. 식 (6)은 예산제약을 의미하고, 식 (7)은 작업 i가 아웃소싱되든지 k번째 위치에서 직접 생산되든지 결정된다는 뜻이다.

위의 수리모델의 제약식 (4)와 식 (5)에서 변수들의 곱으로 표현되어 있으므로, 비선형계획(Nonlinear Programming)이라고 판단하며, 이러한 형태의 수리모델에서 효과적인 해법을 개발하는 것은 간단한 일이 아니다.

본 논문의 FSO 문제가 경영과학 분야에서의 NP-complete 문제임을 감안하길 바란다. Lee and Sung[14]의 논문에서처럼 설비가 단일인 경우에도 NP-complete임을 참조할 수 있다. 본 논문은 시간복잡도 pseudo-polynomial의 동적계획 알고리즘과 분지한계 알고리즘을 제안하고, 효과적인 해를 빠른 시간에 구하는 휴리스틱 알고리즘을 개발코자 한다. 이러한 알고리즘들을 도출하기 위해서 다음과 같은 성질들을 활용할 수 있다.

성질 1 : 자체적으로 처리하는 작업들의 작업순서 순열일정(permutation schedule)을 따르는 것이 이득이다.

증명 : 인접작업교환(pairwise job interchange)으로 간단히 확인할 수 있으므로, 세부적인 증명을 생략한다. □

본 논문이 다루는 FSO 문제는 직접 처리하는 작업들과 아웃소싱 작업들을 구분하는 문제와, 자체적으로 처리하는 작업들의 가공순서를 결정하는 두 가지 의사결정이 내포되어 있다. 만일 아웃소싱할 작업들이 이미 결정되어 있다면 처리순서를 결정하는 두 번째 의사결정문제는 다음의 성질들을 활용하여 해결할 수 있다.

성질 2 : 아웃소싱하는 작업들은 설비 2에서 최대한 우선적으로 처리하는 것이 이득이며, 아웃소싱 작업들 간의 처리순서는 임의로 결정해도 손해가 없다.

증명 : 인접작업교환(pairwise job interchange)으로 간단히 확인할 수 있으므로, 세부적인 증명을 생략한다. □

성질 2의 내용을 활용한다면, 아웃소싱하는 작업들이 결정된 상황이라면, 설비 2에서는 이러한 아웃소싱 작업들을 최우선적으로 처리하는 것이 이득이며, 또한 이러한 작업들 간의 처리순서는 어떤 식으로 처리해도 무관함을 알 수 있다. 다음의 성질 3은 자체적으로 처리하는 작업들에 대한 처리순서에 관한 분석내용이다.

성질 3 : 자체적으로 생산하는 작업들은 다음의 Johnson's rule에 의해서 최적의 처리순서를 결정할 수 있다.

증명 : 인접작업교환(pairwise job interchange)으로 간단히 확인할 수 있으므로, 세부적인 증명을 생략한다. □

<Johnson's Rule>

단계 0 : 작업들을 집합들 U와 V으로 구분한다. $U = \{j | p_j < q_j\}$, $V = \{j | p_j \geq q_j\}$.

단계 1 : U에 속한 작업들을 p_j의 오름차순으로 정렬하고, V에 속한 작업들을 q_j의 내림차순으로 정렬한다.

단계 2 : U에 속한 작업들을 먼저 처리하고, V에 속한 작업들을 이후에 순서대로 처리한다.

자체적으로 가공되는 작업들의 처리순서는 위의 Johnson's Rule으로부터 결정되어 질 수 있으며, 아웃소싱되는 작업들은 설비 1의 단계는 생략되며 외부로부터 즉시에 구매하며, 설비 2에서는 우선적으로 처리되는 것이 이득임을

성질 2로부터 확인할 수 있다. 그러므로 성질들 2와 3으로부터 기본적으로 작업들의 Indexing은 Johnson's Rule을 통해서 결정될 수 있음을 확인할 수 있다. 즉 기본적으로 Johnson's Rule을 활용해서 처리순서를 결정해놓고, 만일 아웃소싱으로 결정된다면 설비 2에서만 최우선적으로 처리할 수 있는 것이다.

이러한 결과로부터 작업들의 번호를 Johnson's rule에 따라서 새롭게 매긴다면, 위의 [FSO 수리모형]은 다음과 같이 [New_FSO 수리모형]으로 새롭게 표현될 수 있다. 새로운 수리모형을 활용한다면 보다 효율적으로 해법을 탐색할 가능성을 높일 수 있을 것이다.

[New_FSO 수리모형]

$$\text{Min} \quad \delta\sum_i o_i y_i + (1-\delta) C_{\max}$$

$$\text{s.t.} \quad C_{\max} \geq C_i \text{ for } \forall i$$

$$C_i \geq \sum_{j=1}^{i} p_j x_j + q_i \text{ for } \forall i$$

$$C_i \geq C_{i-1} \text{ for } \forall i$$

$$\sum_i o_i y_i \leq K$$

$$x_i + y_i = 1, \text{ for } \forall i$$

$$x_i, y_i \in \{0,1\}.$$

변수 x_i는 작업 i가 첫 번째 설비에서 자체적으로 가공된다면 1의 값을 가지고, 그렇지 않으면 0 값을 가진다. 이제까지는 아웃소싱 작업들과 자체 처리작업들이 구분된 경우의 가공순서를 결정하는 방안에 대해서 고려하였다. 여기서는 아웃소싱의 고려대상에서 배제가능한 작업들을 구분할 수 있는 방안을 제시코자 한다. 다음의 성질 4의 조건을 만족한다면 해당 작업은 아웃소싱의 대상에서 원천적으로 배제가 가능하다.

성질 4 : 작업 j가 다음의 조건을 만족한다면, 아웃소싱 대상에서 제외되는 것이 이득이다.

$$\delta o_j > (1-\delta)(p_j + q_j) \tag{8}$$

증명 : 작업 j가 아웃소싱 된다면, 발생되는 목적식 증가분의 최소값은 δo_j이다. 반대로 작업 j를 직접 가공한다면 발생하는 목적식의 증가분의 최대값은$(1-\delta)(p_j+q_j)$이다. 따라서 식 (8)이 성립한다면 작업 j를 아웃소싱하는 비용 자체가 상대적으로 너무 크다는 것을 뜻하므로, 아웃소싱의 가능성을 배제하는 편이 보다 이득이다. □

3. FSO 문제 해법

$$f_j(x_1, y_1, y_2) =$$

$$\min \begin{cases} \infty, \text{ if } y_1 > y_2 - q_j & \text{(i)} \\ \min_{0 \leq k \leq (y_2 - q_j - x_1)} \{ f_{j-1}(x_1 - p_j, \\ \quad y_1, y_2 - q_j - k) \}, \text{ if } x_1 \leq y_2 - q_j & \text{(ii)} \\ f_{j-1}(x_1, y_1 + q_j, y_2) + (1-\delta) q_j + \delta o_j & \text{(iii)} \end{cases}$$

본 장에서는 FSO 문제를 위한 해법으로서 동적계획법, 분지한계법, 발견적 휴리스틱 등을 제시한다.

3.1 동적계획법

FSO 문제가 이미 NP-complete 문제라는 점을 언급하였다. 다항식 시간복잡도를 가지는 최적해법을 제안할 수 없음을 뜻한다. 그래서 이번 절에서는 FSO 문제 자체가 아니라 다양한 환경에 따라서 어떠한 해법이 존재하겠는지를 논의코자 한다.

먼저 FSO 문제가 NP-complete 문제이지만, 아웃소싱 예산이 충분한 경우에 대해서는 Johnson's rule에 기반한 아래와 같은 동적계획법을 활용하여 최적해를 다항식 시간복잡도 내에서 구할 수 있다.

<DP 알고리즘 X>

- 정렬(indexing) : Johnson's rule을 활용하여 작업들을 정렬한다.
- 가치함수(value function), $f_j(x_1, y_1, y_2)$: 작업 1에서부터 j까지의 작업들이 설비 1에서 $[0, x_1]$ 동안에 처리되고, 설비 2에서 $[y_1, y_2]$ 시간 동안에 처리되는 상황에서의 최소 목적함수 값이다.
- 초기조건 : $f_0(0, 0, 0) = 0$
- 최적해 : $\min_{\{(x_1,y_1,y_2) \mid 0 \leq x_1 \leq A, 0 \leq y_1 \leq y_2 \leq B\}} f_n(x_1, y_1, y_2)$,

$$\text{where } A = \sum_{j=1}^{n} p_j, \; B = \sum_{j=1}^{n} q_j$$

- 순환식(recursive equation) : 순환식에서 식 (i)은 작업 j를 설비 2에서 처리할 여유가 없음을 뜻한다. 따라서 무한대의 상태값을 가진다. 식 (ii)는 작업 j를 직접적으로 가공하는 경우를 뜻한다. 식 (iii)는 작업 j를 아웃소싱함을 뜻한다. 위의 DP 알고리즘 X의 상태(state)의 수는 최대 AB^2이므로 $O(AB^2)$의 시간복잡도를 가진다.

<DP 알고리즘 Y>

- 정렬(indexing) : Johnson's rule을 활용하여 모든 작업들을 순서대로 나열한다.

• 가치함수(value function), $f_j(x_1, y_1, y_2; b)$: 작업 1에서부터 j까지 작업들을 설비 1에는 [0, x_1]시간 동안에 처리되고, 설비 2에서는 [y_1, y_2]시간 동안에 처리하며, 아웃소싱 소요비용은 b를 활용하는 목적함수의 최소값을 나타낸다.

• 초기조건 : $f_0(0, 0, 0; 0) = 0$

• 최적해 : $\min_{\{(x_1,y_1,y_2,b)\mid 0 \le x_1 \le A, 0 \le y_1 \le y_2 \le B, 0 \le b \le K\}} f_n(x_1, y_1, y_2; b)$

$$\text{where } A = \sum_{j=1}^{n} p_j,\ B = \sum_{j=1}^{n} q_j$$

• 순환식(recursive equation) :

$f_j(x_1,y_1,y_2;b) =$

$$\min \begin{cases} \infty, \text{ if } y_1 > y_2 - q_j & \text{(i)} \\ \infty, \text{ if } x_1 > y_2 - q_j \text{ and } o_j > b & \text{(ii)} \\ \min_{0 \le k \le (y_2 - q_j - x_1)} \{f_{j-1}(x_1 - p_j, y_1, y_2 - q_j - k; b) + (1-\delta)(q_j + k)\}, \\ \quad \text{if } x_1 \le y_2 - q_j & \text{(iii)} \\ f_{j-1}(x_1, y_1 + q_j, y_2; b - o_j) + (1-\delta)q_i + \delta o_i, \text{ if } o_i \le b & \text{(iv)} \end{cases}$$

순환식에서 식 (i)은 작업 j를 설비 2에서 처리할 여유가 없음을 뜻한다. 따라서 무한대의 상태값을 가진다. 식 (ii)는 작업 j를 아웃소싱할 비용적인 여유가 없으며 설비 1에서 직접 처리할 시간적 여유도 없음을 뜻한다. 따라서 무한대의 상태값을 가진다. 식 (iii)는 작업 j를 직접적으로 가공하는 경우를 뜻한다. 식 (iv)는 작업 j를 아웃소싱함을 뜻한다. 위의 DP 알고리즘 Y의 상태(state)의 수는 최대 AB^2K이므로 $O(AB^2K)$의 시간복잡도를 가진다.

3.2 휴리스틱 알고리즘

여기서부터는 짧은 시간 안에 효율적인 해결책을 얻을 수 있는 휴리스틱 알고리즘을 제안한다. 제안되는 휴리스틱은 Greedy 유형이다. 먼저 작업별로 $\frac{(1-\delta)(p_j+q_j)}{\delta o_j + (1-\delta)q_j}$을 구하고, 관련되는 값이 클수록 아웃소싱에 따른 이익이 클 것이라는 아이디어에 착안하였다. 그러한 작업들에게 우선적인 아웃소싱의 기회를 제공하는 바, 이름을 "휴리스틱 Greedy"으로 명명하였으며, 세부적인 절차는 다음과 같이 정의될 수 있다.

<휴리스틱 Greedy>

단계 0 : 작업들을 $\frac{(1-\delta)(p_{[1]}+q_{[1]})}{\delta o_{[1]} + (1-\delta)q_{[1]}} \ge \frac{(1-\delta)(p_{[2]}+q_{[2]})}{\delta o_{[2]} + (1-\delta)q_{[2]}} \ge \cdots \ge \frac{(1-\delta)(p_{[n]}+q_{[n]})}{\delta o_{[n]} + (1-\delta)q_{[n]}}$이 성립하도록 순서를 정한다. $q=1$, $B=K$ 초기화한다.

단계 1 : $q > n$의 조건을 만족한다면, 자체적으로 처리하는 작업들을 Johnson's rule으로 처리순서를 정하고 종료.

단계 2 : 성질 4의 조건에 해당되거나, 조건 $o_{[q]} \ge B$이 충족한다면, 작업 $[q]$는 아웃소싱되지 않고, $q = q+1$으로 업데이트를 한 후에, 단계 1로 이동한다. 그렇지 않으면, 단계 3으로 이동한다.

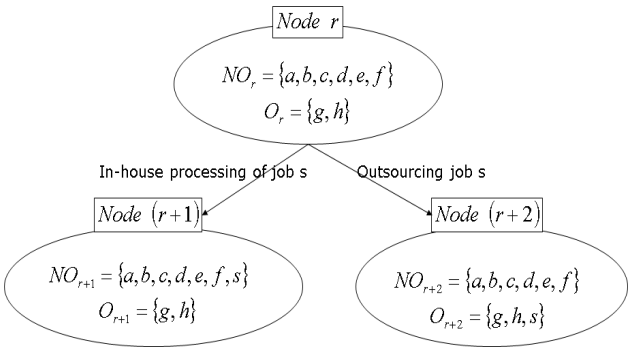
단계 3 : 작업 $[q]$가 아웃소싱 된다. $q=q+1$, $B = B - o_{[q]}$업데이트하고, 단계 1로 이동한다.

4. 분지한계 알고리즘

4.1 분지규칙

본 논문은 최적해를 구하기 위한 분지한계(Branch-and-bound) 알고리즘을 제안한다. 앞서 제안한 DP 알고리즘 Y를 활용하면 최적해를 구할 수 있다. 제 5장에서 DP 알고리즘의 성능을 확인하겠지만, DP 알고리즘의 효율성은 그다지 높지 않음이 일반적인 견해이다. 따라서 보다 효율적으로 최적해를 구하기 위한 분지한계 알고리즘을 제시코자 한다.

본 논문의 분지한계 알고리즘의 노드는 작업들의 부분집합이며, 일부작업들은 직접 처리하고, 나머지 작업들은 아웃소싱으로 구분된다. 직접 처리하는 작업들은 Johnson's rule에 의해 순서가 결정된다. 분지한계 알고리즘은 하나의 노드를 선택하고, 아직 결정되지 않은 하나의 작업을 선택하여, 해당 작업을 직접 처리하거나, 아웃소싱하는 경우들을 상정하여 두 개의 새로운 노드를 만들어서 접목시킨다. 본 논문의 분지한계 알고리즘의 트리구조는 다음 <Figure 2>와 같다.



<Figure 2> The Node Structure in the Branching Tree

<Figure 2>에서 기호 NO_r은 해당 노드 r에서 직접적으로 처리하는 작업들 집합이고, O_r은 해당 노드 r에 아웃소싱되는 작업들을 뜻한다. $O_r = \{g,h\}$는 작업들 g와 h가 아웃소싱으로 처리됨을 뜻한다. 노드 r에서 결정되지 않은 작업들 중에서 임의의 작업 s를 선별하고 두 개의 새로운 노드들$(r+1)$과 $(r+2)$을 생성한다. 노드 $(r+1)$는 작업 s가 직접 처리되는 경우이며, Johnson's rule에 기반하여 처리 순서를 결정한다. 노드 $(r+2)$은 작업 s를 아웃소싱으로 처리하는 경우이다.

본 논문은 깊이우선 검색(depth-first search)을 노드 선택을 위한 방법론으로 활용한다. 즉 분지한계 트리에서 가장 많은 수의 작업들을 포함하는 노드를 우선적으로 선택함을 뜻한다.

4.2 한계규칙(Bounding Rule)

분지한계 알고리즘에서 한계규칙은 분지한계 탐색트리의 노드들을 검색하여, 조건에 만족하는 노드들의 제거 여부를 결정함으로써 보다 효과적인 해의 탐색을 가능토록 만든다. 먼저 본 논문에서는 초기상한 값으로 제3.2절에서 제시된 휴리스틱 Greedy의 해 값을 적용한다. 또한 노드 제거를 위해서 구하는 하한값을 구하는데, 구체적으로 노드 r의 하한값 LB_r은 다음의 계산 값을 활용한다.

$$LB_r = TC(\pi_r) + (1-\delta)\sum_{j\in\pi_r} q_j$$

4.3 노드제거규칙(Fathoming Rules)

본 논문의 분지한계 알고리즘은 두 가지 노드제거 규칙들을 활용하는데, 첫 번째는 성질 4를 활용한다. 두 번째는 다음의 성질 5에 기반한다.

성질 5 : 작업들 i, j가 $p_i < p_j$, $q_i < q_j$, $o_i \geq o_j$을 만족한다면, 만일 i가 아웃소싱 된다면, j도 반드시 아웃소싱되는 것이 이득이다.

증명 : 인접작업교환(pairwise job interchange)으로 간단히 확인할 수 있으므로, 세부적인 증명을 생략한다. □

5. 성능 평가

이제는 제 3장에서 유도된 DP 알고리즘과 휴리스틱 Greedy와 제 4장에서 제시된 분지한계 알고리즘의 성능을 분석한다. 성능의 측정을 위해 활용되는 파라메터들은 다음과 같이 셋팅될 수 있다.

1) 설비 1에서 작업들의 처리시간 p_i와 설비 2에서의 처리시간 q_i는 $U(1, 10)$으로부터 생성시킨다. $U(a, b)$는 이산형 균등분포이다.
2) 아웃소싱에 필요한 비용 o_i은 $U(1, 30)$으로 생성시킨다.
3) 예산 K와 δ는 각각 $U(50, 150)$와 $U(0.3, 0.7)$으로 생성시킨다.

작업의 수는 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60, 65, 70 75, 80의 총 16가지를 고려하였다. 각 작업에 대하여 20개 인스턴스들을 랜덤으로 생성하여 여러 알고리즘들을 평가하였다. 결과는 <Table 1>에 정리되어 있는데, 분지한계 알고리즘에 대해서는 최적해를 구하기까지의 평균 노드수, 최대노드 수, 평균시간, 최대시간을 구하였다. DP 알고리즘 Y가 최적해를 구하기까지의 평균 states 수, 최대 states 수, 평균시간, 최대시간을 구하였다. 휴리스틱 Greedy에 대해선 최적해와의 Gap을 계산하였다. OPT가 최적해이고 $Heuristic$이 휴리스틱 Greedy를 통해 얻은 값이라면, <Table 1>에 나타난 Gap(%)은 다음과 같다.

$$Gap(\%) = \frac{(Heuristic - OPT)}{OPT} \times 100$$

Table의 알고리즘들의 성능평가를 확인해보면 분지한계 알고리즘의 경우 비교적 큰 수의 작업에 대해서도 최적해를 도출하고 있다. DP 알고리즘 Y의 경우는 상대적으로 그다지 크지 않은 수의 작업들에 대해서 최적해를 구하는 것을 확인할 수 있다. 제안된 휴리스틱 Greedy의 경우도 작업의 수가 증가함에 따라서 Gap(%)이 크게 떨어지지 않고 오히려 안정적인 성능을 나타냄을 확인할 수 있다.

아웃소싱을 위한 예산제약의 변동에 따른 알고리즘들의 성능을 평가하기 위한 실험을 진행하였다. 예산 K를 $U(50, 100)$, $U(100, 150)$, $U(150, 200)$, $U(200, 250)$ 4가지로 설정하여 테스트하였으며, 결과를 <Table 2>에 정리하였다. 예산의 변동에 따라서 분지한계 알고리즘의 성능은 크게 변동이 없었으나, 휴리스틱 Greedy의 성능은 다소 저하되었으나, 변동폭은 그다지 크지 않았다.

아웃소싱에 소요되는 비용 o_j의 변동에 알고리즘들 성능을 분석하였다. o_j는 $U(10, 30)$, $U(20, 40)$, $U(30, 50)$, $U(40, 60)$을 고려했으며, 결과는 <Table 3>에 정리되어 있다. 비용이 커질수록 분지한계 알고리즘과 Greedy 휴리스틱의 성능이 향상되는 것을 확인할 수 있다.

<Table 1> The Test Summary of DP Algorithm Y, Branch-and-Bound and Greedy Heuristic

job	DP algorithm Y				No	B&B algorithm			Greedy heuristic Gap (%)		
	number of states		time(s)			number of states		time(s)			
	aver.	max.	aver.	max.		aver.	max.	aver.	max.	aver.	max.
5	59	62	0.00	0.00	20	12	35	0.00	0.00	35.40	114.0
10	1705	2046	0.00	0.00	20	37	67	0.00	0.00	35.50	90.6
15	38409	64806	0.03	0.05	20	89	223	0.00	0.00	29.07	59.5
20	837411	2029839	0.62	1.49	20	316	1412	0.00	0.00	37.78	66.4
25	10109285	43742832	7.62	32.69	20	441	3564	0.00	0.01	31.44	50.5
30	128367893	509200806	100.02	389.51	20	756	6445	0.00	0.01	29.98	68.0
35	(*)	(*)	(*)	(*)	18	3350	43175	0.00	0.05	28.98	52.9
40	(*)	(*)	(*)	(*)	18	1883	17699	0.00	0.02	26.65	48.3
45	(*)	(*)	(*)	(*)	18	2384	18053	0.00	0.02	26.59	57.3
50	(*)	(*)	(*)	(*)	19	2697	24572	0.00	0.03	23.93	45.3
55	(*)	(*)	(*)	(*)	16	45770	873480	0.05	0.97	22.23	35.6
60	(*)	(*)	(*)	(*)	17	48922	749421	0.06	0.85	23.11	43.5
65	(*)	(*)	(*)	(*)	5	132438	2504523	0.16	2.92	21.41	34.2
70	(*)	(*)	(*)	(*)	3	13584	139658	0.02	0.18	21.92	37.6
75	(*)	(*)	(*)	(*)	1	320998	5971369	0.40	7.45	20.16	36.8
80	(*)	(*)	(*)	(*)	1	68037	876343	0.09	1.19	20.24	37.7

(*) indicates the case that the optimal solutions are not found within the computation time.

<Table 2> The Test Summary for the Outsourcing Budget

K	n	Greedy heuristic Gap(%)		B&B algorithm			
		aver.	max.	number of states		time(s)	
				aver.	max.	aver.	max.
[50, 100]	10	31.30	65.42	53	216	0.00	0.00
	30	35.99	70.47	216	702	0.00	0.00
	50	22.61	50.51	1013	5570	0.00	0.01
	Aver.	29.97	62.13	427	2162	0.00	0.00
[100, 150]	10	35.64	97.30	47	176	0.00	0.00
	30	42.25	79.09	260	1005	0.00	0.00
	50	32.35	62.48	1716	27299	0.00	0.03
	Aver.	36.75	79.62	674	9493	0.00	0.01
[150, 200]	10	41.98	92.73	50	125	0.00	0.00
	30	44.61	98.35	230	1099	0.00	0.00
	50	42.09	88.92	422	850	0.00	0.00
	Aver.	42.89	93.33	234	691	0.00	0.00
[200, 250]	10	44.11	99.32	43	102	0.00	0.00
	30	41.55	86.74	270	1069	0.00	0.00
	50	47.59	73.51	736	2478	0.00	0.00
	Aver.	44.42	86.52	349	1216	0.00	0.00

<Table 3> The Test Summary for the Outsourcing Cost

K	n	Greedy heuristic Gap(%)		B&B algorithm			
		aver.	max.	number of states		time(s)	
				aver.	max.	aver.	max.
[10, 30]	10	19.38	56.90	23	73	0.00	0.00
	30	18.35	61.96	100	538	0.00	0.00
	50	14.76	58.13	415	5959	0.00	0.01
	Aver.	17.50	59.00	179	2190	0.00	0.00
[20, 40]	10	6.06	52.22	14	65	0.00	0.00
	30	4.51	22.95	30	111	0.00	0.00
	50	3.88	25.78	62	364	0.00	0.00
	Aver.	4.82	33.65	35	180	0.00	0.01
[30, 50]	10	0.41	8.18	10	17	0.00	0.00
	30	0.63	12.59	20	34	0.00	0.00
	50	0.19	3.80	30	47	0.00	0.00
	Aver.	0.41	8.19	20	32	0.00	0.00
[40, 60]	10	0.00	0.00	9	10	0.00	0.00
	30	0.00	0.00	19	20	0.00	0.01
	50	0.00	0.00	30	41	0.00	0.00
	Aver.	0.00	0.00	19	23	0.00	0.00

6. 결 론

본 논문은 두 설비에서 연속적으로 가공하여 제품을 완성하는 경우에, 아웃소싱을 위한 예산의 제약을 상정하고 설비 1에서의 작업의 가공을 자체적으로 처리하거나 외부설비에 아웃소싱하는 전략을 선택할 수 있을 때, 최적의 의사결정계획을 수립하는 문제를 고려하고 있다. 작업완료시간으로서의 일정계획 코스트와 아웃소싱 비용간의 합을 최소화하는 동적계획법, 분지한계 알고리즘, 휴리스틱 등을 제시하였다.

본 논문의 내용은 생산시스템에서 경쟁이 치열하며, 시간적으로 압박이 크며, 아웃소싱 전략을 빈번히 고려하는 경쟁환경에서 활용될 수 있으며, 예산에 따른 제약이 크게 고려되는 환경에서 효율적으로 작용할 가능성이 있다.

기존 일정계획 논문들은 자체적인 설비에서 작업들을 모두 처리하는 경우만을 상정하고 있으나, 본 논문은 아웃소싱 전략을 일정계획 문제에 접목하는 새로운 의사결정문제를 다루고 있으며, 다변적인 환경과 제약을 고려함으로써 보다 광범위한 연구를 검토할 수 있다.

Acknowledgement

이 논문은 동아대학교 학술연구비 지원에 의하여 연구되었음.

References

[1] Lee, I.S. and Yoon, S.H., A Parallel Machine Scheduling Problem with Outsourcing Options. *Journal of the Society of Korea Industrial and Systems Engineering*, 2008, Vol. 31, No. 3, p 101-109.

[2] Abdel-Malek, L., Kullpattaranirun, T., and Nanthavanij, S., A Framework for Comparing Outsourcing Strategies in Multi-layered Supply Chains. *International Journal of Production Economics*, 2005, Vol. 97, p 318-328.

[3] Baker, K.R., Introduction to Sequencing and Scheduling, John Wiley and Sons, Inc., New York, 1974, p 118-119.

[4] Byung-Cheon, C. and Jibok, C., Two-machine flow shop scheduling problem with an outsourcing option. *European Journal of Operational Research*, 2011, Vol. 213, p 66-72.

[5] Cachon, G.P. and Harker, P.T., Competition and Outsourcing with Scale Economies. *Management Science*, 2002, Vol. 48, p 1314-1333.

[6] Dae-Young, C. and Byung-Cheon C., Outsourcing and scheduling for two-machine ordered flow shop scheduling problems. *European Journal of Operational Research*, 2013, Vol. 226 p 46-52.

[7] Dekkers, R., Decision Models for Outsourcing and Core Competencies in Manufacturing. *International Journal of Production Research*, 2000, Vol. 38, p 4085-4096.

[8] Huiskonen, J. and Pirttila, T., Lateral Coordination in a Logistics Outsourcing Relationship. *International Journal of Production Economics*, 2002, Vol. 78, p 177-185.

[9] Kaipia, R. and Tanskanen, K., Vendor Managed Category Management-an Outsourcing Solution in Retailing. *Journal of Purchasing and Supply Management*, 2003, Vol. 9, p 165-175.

[10] Kangbok, L. and Byung-Cheon, C., Two-stage production scheduling with an outsourcing option. *European Journal of Operational Research*, 2011, Vol. 213, p 489-497.

[11] Kim, B., Dynamic Outsourcing to Contract Manufacturers with Different Capabilities of Reducing the Supply Cost. *International Journal of Production Economics*, 2003, Vol. 86, p 63-80.

[12] Kouvelis, P. and Milner, J.M., Supply Chain Capacity and Outsourcing Decisions : the Dynamic Interplay of Demand and Supply Uncertainty. *IIE Transactions*, 2002, Vol. 34, p 717-728.

[13] Lee, I.S. and Sung, C.S., Single Machine Scheduling with Outsourcing Allowed. *International Journal of Production Economics*, 2008, Vol. 111, p. 623-634.

[14] Lee, I.S. and Sung, C.S., Minimizing Due Date Related Measures for a Single Machine Scheduling with Outsourcing Allowed. *European Journal of Operational Research*, 2008, Vol. 186, p 931-952.

[15] Ngwenyama, O.K. and Bryson, N., Making the Information Systems Outsourcing Decision : A Transaction Cost Approach to Analyzing Outsourcing Decision Problems. *European Journal of Operational Research*, 1999, Vol. 115, p 351-367.

[16] Nicholson, L., Vakharia, A.J., and Erenguc, S.S., Outsourcing Inventory Management Decisions in Healthcare : Models and Application. *European Journal of Operational Research*, 2004, Vol. 154, p 271-290.

[17] Offodile, O.F. and Abdel-Malek, L.L., The Virtual Manufacturing Paradigm : The Impact of IT/IS Outsourcing on Manufacturing Strategy. *International Journal of Production Economics*, 2002, Vol. 75, p 147-159.

[18] Xiangtong, Q., Two-stage production scheduling with

an option of outsourcing from a remote supplier. *Journal of System Science and System Engineering*, 2009, Vol. 18, p 1-15.

[19] Tarakci, H., Tang, K., Moskowitz, H., and Plante, R., Incentive Maintenance Outsourcing Contracts for Channel Coordination and Improvement. *IIE Transactions*, 2006, Vol. 38, p 671-684.

[20] Tarakci, H., Tang, K., Moskowitz, H., and Plante, R., Maintenance Outsourcing of a Multi-process Manufacturing System with Multiple Contractors. *IIE Transactions*, 2006, Vol. 38, p 67-78.

[21] Wu, F., Li, H.Z., Chu, L.K., and Sculli, D., An Outsourcing Decision Model for Sustaining Long-term Performance. *International Journal of Production Research*, 2005, Vol. 43, p 2513-2535.

[22] Yang, J., Qi, X., and Xia, Y., A Production-inventory System with Markovian Capacity and Outsourcing Option. *Operations Research*, 2005, Vol. 53, p 328-349.