Journal of the Korean Institute of Industrial Engineers
Vol. 25, No. 1, pp. 47-55, March 1999.

정비작업 인력 수준 최소화를 위한 발견적 기법

장수영 · 홍유신 · 김중회 · 김새래

포항공과대학교 산업공학과

A Heuristic Algorithm for Minimizing Maintenance Workforce Level

Soo Y. Chang · Yu Shin Hong

This paper develops an efficient heuristic algorithm for scheduling workforce level that can accommodate all the requested maintenance jobs. Each job has its own release and due dates as well as man-day requirement, and must be scheduled in a non-interrupted time interval, namely, without preemption. Duration of each job is not fixed, but to be determined within given specific range. The objective is to minimize workforce level to complete all the requested maintenance jobs. We show that the problem can be seen as a variant of the two-dimensional bin-packing problem with some additional constraints. A non-linear mixed integer programming model for the problem is developed, and an efficient heuristic algorithm based on bin-packing algorithms is proposed. In order to evaluate goodness of the solution obtained from the proposed algorithm, a scheme for getting a good lower bound for the optimum solution is presented and analyzed. The computational experiment shows that the proposed algorithm performs quite satisfactorily.

1. 서 론

반도체, 중화학 공업과 같은 대형 장치 산업에서는 설비의 가동률이 생산성에 미치는 영향이 지대함에 따라 정비를 위하여 일시적 또는 전면적인 설비의 가동을 멈추는 계획은 보다 신중하게 수립되고 시행되어야 한다. 이러한 이유로, 많은 정비 작업들은 미리 정해진 예방 정비 기간에 집중되게 되어 이러한 "정비 수요 집중기간" 에는 정비 인력의 소요가 급증하게 되는 반면에 그 이외의 기간에서는 상대적으로 정비 인력의 소요가 줄어들게 된다. 따라서 대형 장치 산업에서는 설비의 가동률을 높이기 위하여 높은 수준의 상근 정비 인력을 보유할 수밖에 없게 되어 많은 비용이 발생됨에 따라, 이러한 상근 정비 인력의 감축을 통한 정비 비용 절감의 필요성이 대두된다.

그러나 요청되는 모든 정비 작업들을 수행하기 위하여 필요한 최소의 정비 인력의 수준을 정하고 이에 따른 정비 작업의 최적 일정 계획을 수립하는 것은 적지 않은 어려움이 있다. 일반적으로, 대형 장치 산업에서는 요청되는 정비 작업의 수가 상당히 많게 된다. K 제철소의 경우에도 매일 수백 건의 정비 작업이 요청됨에 따라 짧은 시간 내에 모든 정비 작업에 인력을 배정해야 한다는 제약이 따르게 된다. 또한 요청되는 정비 작업은 중단(Preemption) 없이 지속적으로 수행되어야 하며, 작업 수행에 필요한 인력의 소요(Man Days: MD), 작업 허용 기간 등이 미리 정해짐에 따라 그러한 기한 내에 얼마만한 정비 인력을 투입하여 어느 기간 동안에 작업을 수행하여야 하는가를 함께 결정하여야 한다. 예를 들어, 6 MD가 필요한 작업은 한 명이 6일간, 2명이 3일간 또는 1.5명이 4일간 수행할 수 있는 등 여러 방안이 가능하다. 여기서 1명 단위 이하의 Fractional한 인력의 배치는 한 사람이 하루에 여러 작업을 나누어 수행함을 의미한다. 또한 한 작업의 총 소요 MD값이 주어졌을 때, 작업기간이 정해지게 되면 이러한 작업에 할당된 인력 수준이 일자별로 일정하게 유지되어야 한다. 따라서, 우리

가 다루는 정비 작업의 배정에서는, 작업의 배정과 함께 작업 기간도 동시에 결정되어야 함에 따라 문제의 복잡도(Complexity)가 증가하게 되어 최적의 배정 계획의 수립을 더욱 어렵게 한다.

본 논문에서는 정비 작업에 있어서 작업들에 대한 최적 인력 할당 계획을 수립하는 문제의 해결 기법을 제시하고, 계산 실험을 통하여 제안된 기법의 효율성을 검증하게 된다. 먼저 2절에서는 본 논문에서 다루는 정비 작업 문제에서의 가정을 설명하고 이를 수리 모형화하여 제안된 문제가 2차원 Bin-Packing 문제[5]의 변형임을 설명하고, 제안된 문제가 NP-Complete 임을 고려하여 3절에서는 만족할 만한 해를 도출할 수 있는 발견적 기법(Heuristic)을 제시한다. 4절에서는 3절에서의 발견적 기법을 이용한 해의 적정성(Goodness)을 평가하기 위하여 최적해에 대한 "좋은" 하한값(Lower Bound)을 도출하는 기법을 제안하고, 마지막 5절에서는 무작위로 생성된 문제에 대하여 본 논문에서 제안된 발견적 기법이 얼마나 좋은 해를 제공하는 가를 계산 실험을 통하여 보이기로 한다.

2. 수리 모형

본 연구의 일정 계획 문제에서 주어진 각각의 정비 작업은 작업량, 작업 시작 가능일, 작업 완료 요구일, 최소 소요 작업 기간 및 최대 허용 작업 기간 등이 주어지게 된다. 또한, 모든 작업은 작업 기간중 항상 일정 수준의 인력이 투입되어야 하며, 한번 시작된 작업은 중단 없이 지속적으로 수행되어야 한다. 여기에 관련된 제반 모수들을 정리하면 아래와 같다.

m : 일정 계획 기간 (Planning Horizon)
n : 대상 작업의 수
p_j : 작업 j의 작업량 (MD)
f_j : 작업 j의 시작 가능일 (Ready Date)
t_j : 작업 j의 완료 요구일 (Due Date)
$\overline{d}_j$($\underline{d}_j$) : 작업 j의 최대(최소) 허용 작업 기간
$Interval_j$: 작업 j의 작업 허용 기간, $[f_j\ t_j]$

위의 정의에서 $\overline{d}_j$는 당연히 $Interval_j$의 폭인 ($t_j - f_j + 1$) 보다는 작거나 같아야 하고, 만약 $\underline{d}_j$ 와 $\overline{d}_j$ 이 같으면 작업 j의 작업 기간은 확정된 경우가 되며, $\overline{d}_j$, $\underline{d}_j$ 그리고 ($t_j - f_j + 1$)가 모두 같은 경우에는 작업 j는 반드시 정해진 기간에 배정되어야 함을 의미한다.

이어서, 문제의 수리 모형을 제안하기 위하여 필요한 변수들을 정의하면,

x_{ij} : 작업 j를 수행하기 위하여 i번째 일자에 배정된 인력의 수준
a_{ij} : 작업 j를 수행하기 위하여 i번째 일자에 인력이 배정되었는가의 여부를 결정하는 변수 (작업 j를 위해 i번째 일자에 인력이 배정되면 1, 아니면 0)
$b^{+}_{ij}0$: 작업 j를 수행하기 위하여 $(i+1)$번째 일자에 배정된 인력이 i번째 일자에 배정된 인력보다 증가(또는 감소)하면 1, 아니면 0

위에서 정의된 모수 및 변수들을 이용하여 주어진 정비 작업의 수행을 위하여 필요한 정비 인력의 최소 수준을 결정하는 문제는 아래와 같은 비선형 혼합 정수 계획 모형(Mixed Integer Nonlinear Programming Model: MINLP)으로 표현될 수 있는데, 이 모형의 목적 함수(2-0)는 주어진 모든 작업들을 수행할 수 있는 정비 인력의 수준을 의미한다. 제약 조건 (2-1)은 모든 작업에서 주어진 작업량을 반드시 수행하여야 함을 나타내며, 제약 조건 (2-2)는 정비 인력의 수준은 어느 일자에도 배정된 모든 작업량을 수행할 수 있어야 함을 표현한다. 제약 조건 (2-3)은 각 작업이 주어진 작업 기간() 내에 배정됨을 보장하고, 제약 조건 (2-4)와 (2-5)는 모든 작업이 작업 허용 기간의 범위 내에서 수행되어야 함을 의미한다. 제약 조건 (2-6)은 (2-7)과 함께 모든 작업은 중단 없이 지속적으로 수행되어야 함을 나타낸다. 또한 (2-7)에서는 모든 작업에 대하여 제한을 줌으로써 (2-8)과 함께 한 작업에 한번 할당된 인력의 수가 그 작업의 종료일까지 변화하지 못하게 하는 역할을 하게 된다.

MINLP:

$$Minimize \quad C_{\max} \tag{2-0}$$

$$Subject\ to \quad \sum_{i \in Interval_j} x_{ji} = p_j \quad \text{for } j = 1, \ldots, n \tag{2-1}$$

$$C_{\max} - \sum_{j=1}^{n} x_{ij} \geq 0 \quad \text{for } i = 1, \ldots, m \tag{2-2}$$

$$p_j\ a_{ij} - x_{ij} \geq 0 \quad \text{for all } \begin{matrix} i \in Interval_j, \\ j = i, \ldots, n \end{matrix} \tag{2-3}$$

$$\sum_{i \in Interval_j} a_{ij} \geq \overline{d}_j \quad \text{for } j = 1, \ldots, n \tag{2-4}$$

$$\sum_{i \in Interval_j} a_{ij} \le \bar{d}_j \quad \text{for } j=1,\ldots,n \qquad (2\text{-}5)$$

$$b_{ij}^{+} - b_{j}^{-} = a_{(i+1)j} - a_{ij} \quad \text{for all } \begin{matrix} i \in Interval_j, \\ j = i,\ldots,n \end{matrix} \qquad (2\text{-}6)$$

$$\sum_{i \in Interval_j} (b_{ij}^{+} + b_{ij}^{-}) \le 2 \quad \text{for } j=1,\ldots,n \qquad (2\text{-}7)$$

$$x_{ij} = \frac{p_j a_{ij}}{\sum_{i = Interval_j} a_{ij}} \quad \text{for all } \begin{matrix} i \in Interval_j, \\ j = i,\ldots,n \end{matrix} \qquad (2\text{-}8)$$

$$x_{ij} \ge 0,\ a_{ij},\ b_{ij}^{+}, b_{ij}^{-} \in \{0,1\} \quad \text{for } \begin{matrix} i=1,\ldots,m, \\ j=1,\ldots,n \end{matrix}$$

위의 수리 모형의 해법을 논하기 앞서, 제안된 문제의 몇 가지 특수한 경우들에 대하여 $\underline{d}_j = \bar{d}_j = 1$, $f_j = 1$, $t_j = m$ 고찰해 보자.

먼저, 모든 작업들에 대하여, 의 경우, 본 문제는 일반적인 일차원 Bin-Packing 문제 [1,2,3]가 됨을 볼 수 있으며, $\underline{d}_j = \bar{d}_j = 1$이고, $f_j \neq 1$ 또는 $t_j \neq m$의 경우의 본 문제는 기계들의 자격 조건(Machine Eligibility)이 주어진 경우의 병렬 기계들의 일정 계획을 수립하는 특수한 형태의 Bin-Packing 문제[4]가 됨을 알 수 있다. 또한, 본 문제에서 $\underline{d}_j = \bar{d}_j$, $f_j = 1$, $t_j = m$의 경우에는 작업 기간이 정해진 모든 작업들을 주어진 계획 기간내에 배정하되 모든 작업들을 수행하는데 필요한 소요 인력의 수준을 최소화하는 이차원 Bin-Packing 문제[2]와 동일하게 됨을 알 수 있다. Bin-Packing 문제는 잘 알려진 바와 같이, NP-Complete 문제이며, 또한 제안된 문제는 일반적인 Bin-Packing 문제보다 더욱 복잡한 구조를 가지고 있어 최적해를 구할 수 없음에 따라 본 논문에서는 빠른 시간내에 만족할 만한 해를 제공하는 발견적 기법을 제안하고자 한다.

3. 발견적 기법

일반적인 Bin-Packing 문제의 해를 구하기 위한 가장 대표적인 기법으로는 LPT (Longest Processing Time First)를 들 수 있다. 또한 본 문제에서는 각각의 작업들에 대하여 작업 허용 기간이 주어짐을 고려하여 자격 조건(Machine Eligibility)이 주어진 경우의 병렬 기계의 일정 계획을 위한 기법으로 알려진 LFJ (Least Flexible Job First) 원칙도 함께 이용하여 해를 구하고자 한다.

$$flexibility_j = \sum_{k=\underline{d}_j}^{\bar{d}_j} \{ (t_j - f_j + 1) - k + 1 \} \qquad (3\text{-}1)$$

먼저, 모든 작업들에 대하여 식 (3-1)로 정의되는 $flexibility_j$를 구한 후 모든 작업들을 $flexibility_j$의 올림차순으로 정리한다. 이는 본 문제의 특성 중에 작업 허용 기간이 주어지는 것은 자격 조건을 갖는 병렬 기계의 일정 계획 문제와 유사한 성격을 가진다는 점을 고려하여 LFJ 원칙을 먼저 적용한 것이다. $flexibility_j$는 작업 j가 제한된 조건을 만족하는 범위내에서 배정될 수 있는 방법의 수를 의미하게 된다. 만약, 동일한 $flexibility_j$를 갖는 작업들이 존재하는 경우에는 이러한 작업들을 대상으로 작업 크기의 내림차순으로 정리한 후, 아래의 유사 코드로 설명된 절차에 따라 차례로 배정하게 된다. 작업 크기의 내림차순으로 정리하는 이유는 일단 $flexibility_j$에 의하여 순서가 정해진 다음 같은 $flexibility_j$값을 갖는 경우에는 LPT 원칙에 따라 배정하는 것이 일반적인 병렬 기계 문제에서 좋은 효율을 나타낸다는 점을 감안한 것이다.

```
PROCEDURE PROPOSED_HEURISTIC
set the partial schedule to be empty;
FOR j = 1, ....., n DO;
    Length = t_j − f_j + 1; Height
    Height_ min = sufficiently large number;
    d_min = d_j ;
    k_min = 1 ;
    FOR d = d_j, ..., d̄_j DO ;
        FOR k = 1, ..., (Length − d + 1) DO ;
            Height = the level of workforce required
                     to schedule job j to start on the
                     k-th day in Intervalj while its
                     duration being set as d given the
                     current partially fixed schedule;
            IF (Height is less than Height_min)
                Height_min = Height ;
                d_min = d ;
                k_min = k ;
    END_DO
END_DO
```

Update the partial schedule by scheduling job j to start on the kmin-th day in $Interval_j$ while its duration being set as d_{min};

END_DO

<그림 1>은 여기서 제안한 기법에 의하여 작업이 배정되는 과정을 보여 준다.

<그림 1>에서 상단 왼쪽에 나타난 모양이 현재까지 배정된 결과라 할 때 $f_j=2$, $t_j=5$, $\underline{d}_j=1$, $\overline{d}_j=3$을 모수로 하는 다음 작업이 어떠한 절차를 거쳐 배정되는 가를 보여 주고 있다. 배정 대상 작업은 $\underline{d}_j=1$, $\overline{d}_j=3$ 그리고 $(t_j-f_j+1)=4$가 되어 이 작업의 작업 기간은 1일, 2일 또는 3일로 정해질 수 있으며, 이에 따른 소요 인력은 각각 4명, 2명 또는 4/3명으로 정해진다. <그림 1>에서 (a)와 (b)는 작업 기간을 3일로 할 때의 허용되는 모든 배정 방안을, (c), (d) 와 (e)는 작업 기간을 2일로 할 때의 허용 배정 방안을, 그리고 (f), (g), (h) 와 (i)는 작업 기간을 1일로 할 때의 모든 허용 배정 경우를 보여 주고 있다. 본 예에서는 (e)의 경우가 현재의 상태에서 필요한 소요 인력의 수준을 최소화하므로 (e)의 방안이 채택되게 된다.

4. 최적해의 하한값 (Lower Bound)

3절에서 설명한 발견적 기법의 성능을 평가하기 위하여서는 도출된 해가 최적해에 얼마나 근사한가를 살펴 볼 필요가 있다. 그러나 제안된 문제는 앞에서 설명한 바와 같이 NP-Complete 문제로서 최적해를 구할 수 없음에 따라, 본 절에서는 최적해에 대한 좋은 하한값(Lower Bound)을 구한 후 이러한 하한값과의 비교를 통하여 제안된 기법에 대한 평가를 하고자 한다.

먼저, 수리 모형, MINLP에서 (2-8)의 비선형 제약 조건과 함께 모형의 모든 정수 변수를 제거하게 되면, (2-3)부터 (2-8)까

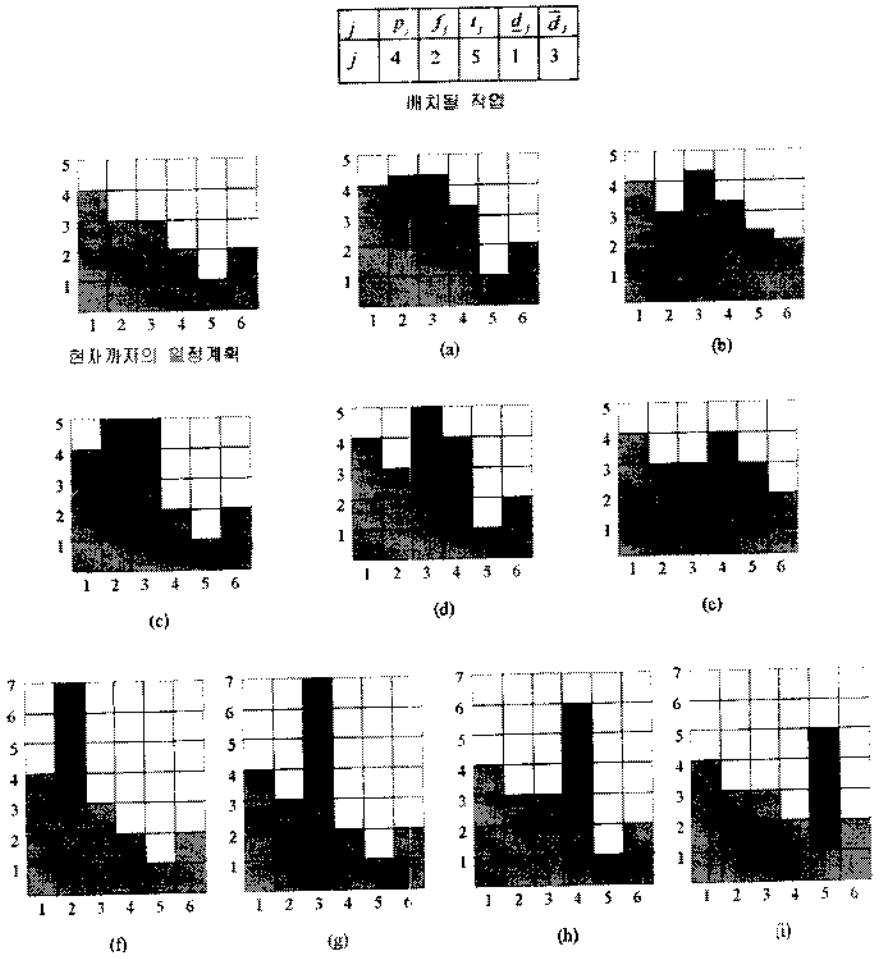


그림 1. 제안된 기법에 의하여 하나의 작업이 배정되는 예.

지의 제약 조건이 없어짐에 따라 아래와 같은 이완 선형 계획(LP Relaxation) 문제를 얻게 된다.

LP Relaxation:

$$\textit{Minimize} \quad C_{\max}$$

$$\textit{Subject to} \quad \sum_{i \in interval_j} x_{ij} = p_j \qquad \text{for } j = 1, \ldots, n$$

$$C_{\max} - \sum_{j=1}^{n} x_{ij} \geq 0 \qquad \text{for } i = 1, \ldots, m$$

$$x_{ij} \geq 0 \qquad \text{for all } i \in Interval_j,\ j = 1, \ldots, n$$

위의 LP Relaxation의 최적해가 3절의 MINLP의 최적해의 하한값이 됨은 자명하다. 그러나 LP Relaxation의 최적해를 구하기 위해서는 제법 큰 규모의 선형계획 문제를 풀어야 되기 때문에 보다 빠른 계산 시간내에 LP Relaxation의 최적해보다 더 좋은 하한값을 구하는 방안을 제안하고자 한다.

먼저, 본 문제에서 만약 모든 작업에 대하여 $f_j = 1$이고 $t_j = m$인 경우에는 모든 작업의 작업량의 합을 계획 기간의 일수로 나눔으로써 간단히 하한값을 계산할 수 있으나, 이 하한값은 당연히 앞의 LP Relaxation의 최적해와 같게 된다. 그러나, $f_j \neq 1$ 또는 $t_j \neq m$ 이면, 기존의 이차원 Bin Packing 문제보다 엄격한 제약 조건이 있으므로, 보다 좋은 하한값을 계산할 수 있을 것으로 예상됨에 따라 기존의 이차원 Bin Packing 문제에서 사용되는 하한값 계산 방식을 다소 수정하여 제안된 문제에 적용하여 보고자 한다. 앞에서 설명한 바와 같이, 만약 $f_j = 1$, $t_j = m$ 이고 $\underline{d}_j = \bar{d}_j$이면, 본 문제는 기존의 이차원 Bin Packing 문제와 동일하게 된다. 이 때, 모든 작업의 작업 허용 기간을 계획 기간 $m(= t_j - f_j + 1)$으로 정하고 작업들을 배정하게 되면 소요 인력의 수준이 최소화되는 것은 명백하다. 그러나 $f_j \neq 1$ 또는 $t_j \neq m$ 일 때, 이러한 방법을 적용하게 되면, <그림 2>에서 보는 바와 같이 부적절한 하한값을 얻게 된다.

<그림 2>는 3개의 정비 작업이 6일간의 계획 기간에 배정되는 예를 보여 준다. 만약 이차원 Bin Packing 문제의 하한값 계산 방식에 의하면, 각각의 작업의 작업 허용 기간을 $(t_j - f_j + 1)$로 정하고 구간 $[f_j, t_j]$에 배정하게 되면, 인력 수준은 <그림 2>의 좌편에서 보는 바와 같이 5라는 하한값을 주게 된다. 그러나 <그림 2>의 우편의 경우와 같이 최소 인력 수준은 3이 되어 5는 하한값이 될 수 없게 된다.

새로운 하한값을 구하기 위하여, m일의 계획 기간에서 가능한 모든 작업 기간에 대하여 각각의 하한값을 구한 후 그 중에서 최대값을 하한값으로 정하는 방안을 제안하고자 한다. 먼저, 주어진 m일의 계획 기간에 포함되는 모든 작업 기간, 즉, $mC_2 = m(m-1)/2$개의 작업 기간들을 $V_1, V_2, \ldots, V_k, V_{m(m-1)/2}$라 정의하고, 모든 V_k에 대하여 아래와 같이 집합 O_k와 집합 P_k를 정의한다.

j	p_j	f_j	t_j	$\underline{d}_j$	$\bar{d}_j$
1	6	1	6	2	6
2	3	3	5	1	3
3	6	3	4	1	2

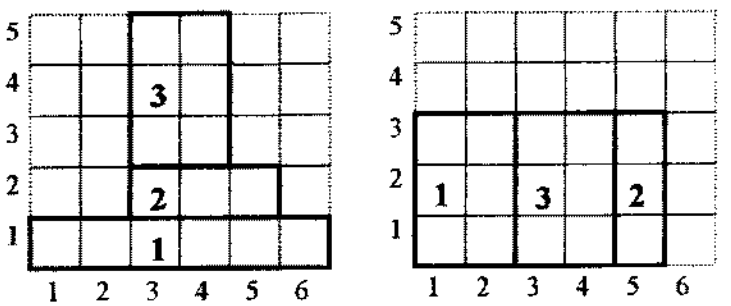


그림 2. 이차원 Bin Packing 문제의 하한값이 부적절한 예.

$$O_k = \{ j \mid Interval_j \subseteq V_k \}$$
$$P_k = \{ j \mid j \notin O_k \}$$
$$\text{for } k = 1, 2, \ldots, m(m-1)/2$$

여기서, 집합 O_k에 포함되는 작업들의 작업량의 합을 $\sigma_k^1 (= \sum_{j \in O_k} p_j)$라 하고, 집합 P_k에 포함된 모든 작업들에 대해서는 반드시 V_k에 배정되어야 하는 작업량들의 합을 구하여 σ_k^2로 정하면 V_k에 대한 하한값 $Bound_k$는 다음과 같이 계산된다.

$$Bound_k = (\sigma_k^1 + \sigma_k^2) / Length_k \quad (4\text{-}1)$$

식 (4-1)에서 $Length_k$는 V_k의 길이를 의미한다. 여기서, σ_k^2가 계산되는 과정은 σ_k^1이 계산되는 과정보다 다소 복잡한데, 이에 대한 이해를 돕기 위하여 <그림 3>에는 구간 $[f_i, t_j]$에 배정하게 되도록 요청된 작업량 중에서 반드시 구간 V_k에 배정되어야 하는 작업량이 얼마나 되는가를 계산하는 과정을 보여 주고 있다. 구간 V_k가 구간 $[f_j, t_j]$를 완전히 포함하지 않을 때, $[f_j, t_j]$와 V_k구간과의 관계는 4가지 가능한 경우가 있으며, 이들 각각의 경우들이 <그림 3>에 나타나 있다. <그림 3>에서 V_k는 $[b, e]$로 표기되었다.

이러한 방법으로 $m(m-1)/2$개의 모든 기간에 대하여 $Bound_k$를 정하고,

$$Bound = Max(Bound_k, Bound_k, \ldots, Bound_k, \ldots, Bound_{m(m-1)/2}) \quad (4\text{-}2)$$

식 (4-2)와 같이 모든 $Bound_k$ 중에서 가장 큰 값을 본 문제의 하한값으로 정하게 된다. 이러한 과정을 유사 코드로 정리하게 되면 아래와 같다.

```
PROCEDURE LOWER_BOUND;
Bound = 0;
FOR b = 1, ..., m DO;
    FOR e = b, ... m DO;
        Bound_k = 0;
    FOR j = 1, ..., m DO;
        IF ( b ≤ f_j AND t_j ≤ e )
            THEN
                Bound_k =
                    bound_k + p_j/(e-b+1);
            ELSE
                CASE 1 ( e < f_j OR t_j < b )
                    Overlap = 0;
                CASE 2 ( f_k < b ≤ t_j ≤ e )
                    Overlap = ( p_j / d_j )*
                    MAX f_j + ( d_j - b, 0);
                CASE 3 ( b ≤ f_j ≤ e < t_j)
                    Overlap = ( p_j / d_j )*
                    MAX (e - t_j + d_j - 0, );
                CASE 4 ( f_j < b AND e < t_j)
                    Thin_L = ( p_j / d_j )*
                      MAX( e - b + 1,
                      MAX( f_j + d_j - b, 0));
                    Thin_L = ( p_j / d_j )*
                      MAX( e - b + 1,
                      MAX( f_j + d_j - b, 0));
                    Thin_R = ( p_j / d_j )*
                      MAX( e - b + 1,
                      MAX( e - t_j + d_j, 0 ));
                    Thin_R = ( p_j / d_j )*
                      MAX( e - b + 1,
                      MAX( f_j + d_j - b, 0 ));
                      Overlap = MIN(Thin_L,
                      Think_L, thin_R, Thick_R);
                Bound_k = Bound_k
                    + Overlap/(e - b + 1);
            END_IF;
        END_DO;
    IF Bound < Bound_k
        Then Bound = Bound_k;
    END_DO;
END_DO;
RETURN Bound;
```

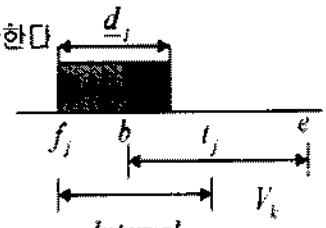
위에서 설명된 방안의 계산량은 $O(m^2 n)$이 되며 이는 *LP Relaxation* 문제의 해를 구하는 것보다 훨씬 적은 계산량을 가지게 된다.

(CASE 1)

$e < f_j$ 거나 $t_j < b$ 이면 반드시 배정되어야 하는 작업량이 없다.

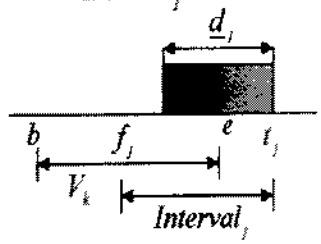
(CASE 2)

$f_j < b \le t_j \le e$ 이면, $(P_j / \underline{d}_j) * \mathrm{MAX}(f_j + \underline{d}_j - b, 0)$를 계산한다.



(CASE 3)

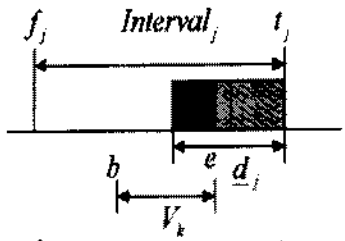
$b \le f_j \le e < t_j$ 이면 $(P_j / \underline{d}_j) * \mathrm{MAX}(e - t_j + \underline{d}_j, 0)$을 계산한다.



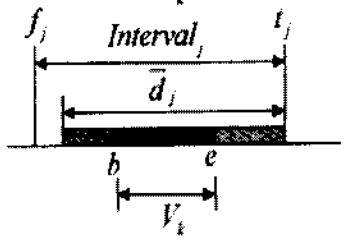
(CASE 4)

$f_j < b$ AND $e < t_j$ 이면,

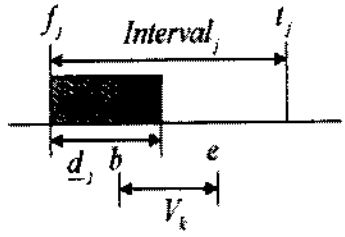
$(P_j / \underline{d}_j) * \mathrm{MIN}(e - b + 1, \mathrm{MAX}(e - t_j + \underline{d}_j, 0))$



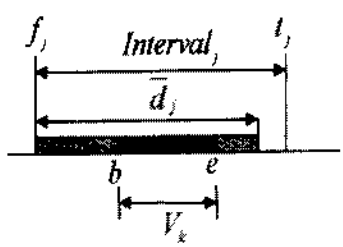
$(P_j / \overline{d}_j) * \mathrm{MIN}(e - b + 1, \mathrm{MAX}(e - t_j + \overline{d}_j, 0))$



$(P_j / \underline{d}_j) * \mathrm{MIN}(e - b + 1, \mathrm{MAX}(f_j + \underline{d}_j - b, 0))$



$(P_j / \overline{d}_j) * \mathrm{MIN}(e - b + 1, \mathrm{MAX}(f_j + \overline{d}_j - b, 0))$



의 네 가능한 값 중 가장 작은 값을 선택한다.

그림 3. 하한값의 계산.

표 1. 계산 실험의 결과

	N=10	N=50	N=100	N=300	N=500	Average PI
5 Man-days	121%	116%	110%	105%	104%	111%
20 Man-days	115%	112%	108%	104%	103%	108%
40 Man-days	114%	110%	108%	104%	103%	108%
70 Man-days	114%	110%	107%	104%	103%	107%
100 Man-days	113%	109%	107%	104%	103%	107%
Average PI	115%	111%	108%	104%	103%	108%

다음은, $(x^*, C^*_{\max})$을 LP Relaxation 문제의 최적해라 할 때, $Bound \geq C^*_{\max}$가 성립함을 증명하고자 한다. 즉, 위의 방법에 의하여 도출된 하한값이 LP Relaxation 문제에서 구한 최적해(하한값)보다 좋음을 보이고자 한다.

LP Relaxation 문제의 최적해 $(x^*, C^*_{\max})$에 대하여 아래와 같이 두 개의 집합을 정의한다.

$$D^= = \{j \mid C_j(x^*) = C^*_{\max}\},$$
$$D^< = \{j \mid C_j(x^*) < C^*_{\max}\}$$

위의 집합에서 $C_j(x^*) = \sum_{j=1}^{n} x^*_{ij}$이다.

그러면, 집합 $D^=$에 속한 특정 기간 V_k에서 $C^*_{\max} = \sigma^1_k / Length_k$가 성립함을 보이면 $Bound \geq C^*_{\max}$가 성립함은 식 (4-2)로부터 자명하게 된다.

집합 $D^=$에 대하여 합 $J^= = \{j \mid x^*_{ij} > i \in D^=\}$를 정의하고, 구축(Constructive) 방법을 이용하여 식 (4-3)이 성립함을 보이고자 한다.

$$Interval_j \bigcap D^< = \emptyset, \quad \text{for all } j \in J^= \tag{4-3}$$

즉, LP Relaxation 문제의 최적해에 대하여 식 (4-3)이 성립한다면 증명이 필요하지 않고, 만약 식 (4-3)이 성립하지 않는다면 주어진 최적해로부터, 식 (4-3)을 만족하는 다른 최적해를 항상 구축할 수 있음을 보이고자 한다.

이를 위하여, 주어진 최적해 $(x^*, C^*_{\max})$이 식 (4-3)을 만족하지 못한다고 가정하면, 반드시 $J^=$에 포함되는 작업 $j(j \in J^=)$에 대하여 $i^0 \in Interval_j \bigcap D^<$를 만족하는 일자 j^0가 존재한다.

이때, $j \in J^=$임으로, 집합 $D^=$에는 반드시 $x^*_{i^*j} > 0$인 일자 i^*가 존재함에 따라 다음과 같은 절차를 거쳐 새로운 최적해를 구할 수 있다.

즉, $0 < \varepsilon < Min\{x^*_{i^*j},\ C^*_{\max} - C_{i^0}(x^*)\}$를 만족하는 ε를 택하여 $x^*_{i^*j} = x^*_{i^*j} - \varepsilon$, $x^*_{i^*j} = x^*_{i^*j} + \varepsilon$로 하면, $C_{i^*}(x^*) < C^*_{\max}$와 $C_{i^0}(x^*) < C^*_{\max}$가 성립함에 따라 새로운 해도 LP Relaxation 문제의 최적해임을 알 수 있다.

이와 같은 과정을 거쳐 얻어진 새로운 최적해에 대하여 집합 $D^=$와 $D^<$를 각각 $D^= = D^= - \{i^*\}$과 $D^< = D^< \bigcup \{i^*\}$로 수정하게 되면, 집합 $D^=$에 포함된 원소의 수는 줄어들게 된다.

이러한 과정을 반복적으로 수행하여 집합 $D^=$에 포함된 원소의 수를 최대한 줄일 수 있으나 결코 공집합은 될 수 없다.

이는 $D^= = \emptyset$ 이라면 $C^*_{\max}$이 LP Relaxation 문제의 최적해가 될 수 없기 때문이다. 따라서, 우리는 일반성을 잃어버리지 않고, 식 (4-3)이 성립함을 알 수 있다.

위의 과정을 거쳐 수정된 집합 $D^=$에서 연속적인 일자를 포함하는 하나의 부분 집합을 $\overline{D}^=$라 정의하고, 집합 $\overline{J}^= = \{j \mid x^*_{ij} > 0,\ i \in \overline{D}^=\}$를 정의하게 되면 식 (4-3)으로 부터 아래의 식 (4-4)가 성립하게 된다.

$$Interval_j \subseteq \overline{D}^=, \quad \text{for all } j \in \overline{J}^= \tag{4-4}$$

따라서, 집합 $\overline{D}^=$의 기간을 V_k라 하면

$C^*_{\max} = \sigma_k / Length_k$가 성립하게 되어 식 (4-2)로부터 $Bound \geq C^*_{\max}$를 만족시킴을 알 수 있어 위에서 설명한 방법을 이용하게 되면 LP Relaxation 문제의 최적해보다 항상 좋은 하한값을 도출하게 된다.

5. 계산 실험

본 논문에서 제안된 기법에 의하여 도출된 해의 적정성을 살

퍼보기 위하여 위에서 설명된 기법을 C 언어를 이용하여 구현한 후 Sun-Compatible Workstation에서 계산 실험을 수행하였다. 대상 문제들은 계획 기간은 30일($m = 30$)로 하고, 두개의 모수, 즉 작업 개수 N과 최대 작업 시간 P의 모수 짝 (N, P)에 대하여 아래의 과정을 거쳐 각각 9,000개씩의 무작위 문제들을 생성하였다. 두 개의 모수 N과 P는 각각 집합 {10, 50, 100, 300, 500}과 집합 {5, 20, 40, 70, 100}을 사용하였으며, 따라서 모든 25개의 모수 짝에 대하여 총 225,000개의 문제를 생성하여 해를 구하였다.

단계 1 : $Uniform(1, P)$ 분포에서 한 값을 생성하여 p_j라 정한다.

단계 2 : $Uniform(1, m)$분포에서 두 값을 생성, 작은 값을 f_j, 큰 값을 t_j라 정한다.

단계 3 : $Uniform(1, t_j - f_j + 1)$ 분포에서 두 값을 생성하여 작은 값을 a_j, 큰 값을 $\overline{d}_j$라 한다.

또한 도출된 해의 적정성을 평가하기 위해서는 본 문제의 최적해와 비교하여야 하나, 앞에서 설명한 바와 같이 본 문제는 NP-Complete 문제로서 최적해의 도출이 불가능함에 따라 앞에서 구한 하한값과 간접적으로 비교하기 위하여 아래와 같은 성능 지표(*Performance Index*, PI)를 설정하여 제안된 기법을 평가하여 보았다.

$$\Pi = \frac{Solution\ by\ Heuristic}{Lower\ Bound} \times 100(\%)$$

총 25개의 모수 짝에 대한 계산 실험의 결과는 <표 1>과 <그림 4>에 주어져 있다. <표 1>과 <그림 4>에 나타난 바와 같이, 작업의 수가 증가함에 따라, 그리고 각 작업의 작업량이 커짐에 따라, 제안된 기법은 더 좋은 성능을 보이고 있음을 볼 수 있다. 작업의 수가 증가함에 따라, 효율이 좋아지는 것은 총 계획 대상 일수가 30으로 묶여져 있기 때문인 것으로 보이며, 이는 일반적인 Bin packing 문제에서 관찰되는 현상임을 알 수 있다. 즉, Bin의 개수가 정해지면, Bin packing 기법들은 packing될 작업의 개수가 늘어남에 따라, 더 낮은 결과를 가져오는 경향이 있기 때문인 것으로 해석된다. 본 논문에서의 정비 인력의 할당 문제는 우리가 아는 바로는 다루어진 적이 없어 비교할 만한 기법이 없으나, 제안된 기법은 최적해와 비교할 때 평균 108% 이내의 인력 수준을 필요로 하는 해를 주고 있을 뿐만 아니라 모든 경우에 대하여 3초 이내에 해를 제공하고 있어 매우 유용한 기법임을 알 수 있다. K 제철소의 경우, 통상 300개 이상의 작업이 배정되어야 하는 점을 감안하면, 주어진 기법은 실용적인 면에서도 매우 가치가 있다고 판단되며, 위의 기법은 현재 K 제철소의 현장에서 정비 인력 배정에 사용되고 있고, 실제 문제의 경우, PI값이 105%를 넘는 경우가 없는 것으로 나타났다. 본 논문에서 제시한 기법의 적용에 따라 얻어진 해는 Neighborhood Search와 같은 기법을 추가 적용하게 되면 개선될 여지가 있으나, 단순한 교환이나, 작업 배정 순서 조정으로는 개선을 기대하기 어렵다고 예상된다. 따라서, 우리 문제에 적절한 Neighborhood Search 방법의 개발을 위한 연구도 현재 진행중에 있으며, 제안된 기법의 이론적인 Worst Case Bound Analysis에 대한 연구도 현재 진행 에 있다.

참고문헌

1. Coffman, Jr. E. G., Garey, M. R. and Johnson, D. S., "An application of bin-packing to multiprocessor scheduling," *SIAM Journal of Computing*, Vol. 7, No. 1, pp. 1-17, 1978.
2. Coffman, Jr. E. G., Garey, M. R. and Johnson, D. S., *Approximation Algorithms for Bin-Packing an Updated Survey*, Bell Laboratories, Murray Hill, N. J., pp. 49-106, 1983.
3. Coffman, Jr. E. G., Garey, M. R. and Johnson, D. S., *Approximation Algorithms for Bin-Packing: A Survey, Approximation Algorithms for NP-Hard Problem*, D. Hochbaum. PWS Publishing, Boston, pp. 46-93, 1996.
4. Pinedo, M., *Scheduling, Theory, Algorithm, and Systems*, Prentice Hall, 1997.
5. Garey, M. R. and Johnson, D. S., *Computers and Intractibility : A Guide to the Theory of NP-Completeness*, Freeman, 1979.

1998년 6월 접수, 1999년 1월 채택