並列處理機械에서 最大納期遲延 時間의 最小化에 관한 硏究

서울대학교 경영학과 안 상형

대구대학교 경영학과 이 송근

한국국방연구원 심 인섭

Abstract

The purpose of this study is to develope efficient approximate and exact algorithms for the problem of scheduling n independent jobs on m unequal parallel processors to minimize maximum lateness.

We incorporate opportunity cost concept to develop a new efficient heuristic. For an exact algorithm we use the result of new heuristic as an upper bound and the result of Lagrangean dual as an lower bound in branch and bound method.

With the heuristic and exact algorithm we have following computational

experiences.

First, by combining new heuristic with De and Morton's results we improve the quality of heuristic algorithms very well.

Second, this study gets tight lower bound which are almost similar to optimal solution in many cases.

Third, with this algorithm we can enhance the scale of the problem which is optimized.

1. 序 言

日程計劃(scheduling) 분야에서 주요한 문제중의 하나는 서로 다른 納期를 가지는 일련의 독립된 작업군(a set of independent jobs)을 일련의 竝列機械(a set of parallel processors)상에서 처리할 때 最大納期遲延時間(the maximum lateness)을 最小化하는 作業割當과 處理順序를 구하는 問題이다. 여기서 병렬처리기계문제라함은 1회 가공(single operation)을 요하는 다수의 작업들이 이 중 어느 기계를 통해서도 1회 가공으로 완료될 수 있는 상황에서, 일정한 작업배정을 통하여 특정 成果尺度(measure of performance)를 최소화하거나 최대화하는 경우의 문제를 의미한다. 이때 적용되는 성과척도로는 작업장의 성격과 목적에 따라서 평균체류시간(mean flow time), 평균납기지연시간(mean tardiness), 납기지연작업수(maximum lateness) 등의 여러 기준이 사용되고 있다. 병렬처리기계에서는 일반적으로 각 작업의 納期가 정해져 있지 않은 경우에는 總作業完了時間(makespan)의 最小化를 성과척도로써 많이 이용하고 있으며, 납기가 정해져 있는 경우에는 最大納期遲延時間의 最小化基準이 많이 적용된다.

일반적으로 납기가 주어진 문제에서의 목표는 가급적 납기이내에 작업을 마치는 것이다. 만약 납기이내에 작업을 마치지 못하여 納期遲延(lateness)이 일어나면 必然的으로 이에 따른 費用(penalty cost)을 지불하기 마련이다. 즉 契約違反에 따른 벌과금이나 반품 등의 비용은 납기지연이 허용한도의 범위내에서 발생하도록 통제할 필요가 있으므로 최대납기지연의 최소화는 이러한 경우에 적절한 목적함수가 되는 것이다.

본 연구에서는 최대납기지연의 최소화의 문제를 해결하기 위한 새로운 휴리스틱 解法과 最適解法에 초점을 맞춘다. 機會費用概念을 導入하여 기존의

휴리스틱 해법보다 계산복잡도가 높지않으면서 최적해와의 格差를 줄일 수 있는 보다 改善된 휴리스틱解法을 開發하고자 한다. 그리고 새로운 휴리스틱해법과 기존의 휴리스틱해법의 성능을 비교하고 최적해와의 평균적인 격차를 파악하기 위해 실용적 규모 문제의 효율적인 최적해법을 구하고자 한다. 최적해법으로 원문제에 대한 라그랑지안 弛緩問題를 구성하고 서브그래디언트技法을 適用하여 最善의 쌍대최적해를 구하고 이것을 原問題에 대한 엄밀한 下限價(tight lower bound)로 삼아 새로운 휴리스틱이 제공하는 엄밀한 上限價(tight upper bound)와 比較하여 上限價 = 下限價인 경우에 最適解를 구하는 방법을 사용한다.

2. 問題의 定義 및 模型의 構成

2.1. 問題의 槪要

竝列處理機械에서 最大納期遲延時間의 最小化라는 目的은 同一하더라도 作業이나 機械의 特性에 따라 다음과 같은 基準으로 問題의 模型을 區分할 수 있다.

1) 작업의 先行構造(precedence structure)의 存在與否

2) 等屬機械(equal machine)와 異速機械(unequal machine)의 區分

3) 單位作業의 分割加工의 可能性 與否

4) 作業着手時點에 모든 작업의 시스템 到着與否

5) 作業準備時點(set-up time)의 存在與否

본 연구에서는 위의 類型 중에서 다음과 같은 問題로 그 範圍를 限定시키기로 한다.

기계의 유형	이속기계
성과척도	최대납기지연시간의 최소화
작업의 선행구조 여부	독립작업
분할가공의 가능여부	분할 불가능
작업의 시스템 도착	0(제로)시점에 작업도착 완료
작업의 준비시간	가공시간에 포함
납기	0시점에 미리 정해져 있음

2.2. 數理模型

작업과 기계들의 집합을 다음과 같이 정의하자.

$i = \{1, 2, \cdots, n\}$ 작업(job)들의 집합

$j = \{1, 2, \cdots, m\}$ 기계(processor)들의 집합

그리고 변수와 모수들을 아래와 같이 정의하면 본 연구에서 다룰 문제는 整數計劃問題로 정립된다.

$$x_{ij} = \left\{ \begin{array}{l} 1 \text{ 만약 작업 } i\text{가 기계 } j\text{에서 가공된다면} \\ 0 \text{ 그렇지 않으면} \end{array} \right\} \begin{array}{l} i = 1, 2 \ldots m \\ j = 1, 2, \ldots n \end{array}$$

L = 최대납기지연시간

t_{ij} = 작업 i 의 가공시간

d_i = 작업 i 의 납기

$$(P) \quad Min \quad L \qquad (1)$$

$$s.\ t. \qquad \sum_{i=1}^{I} t_{ij} x_{ij} - d_I \leq L \qquad \text{for } I = 1,2,\cdots,n \qquad (2)$$

$$j = 1,2,\cdots,m$$

$$\sum_{j=1}^{m} x_{ij} = 1 \qquad \text{for } i = 1,2,\cdots,n \qquad (3)$$

$$x_{ij} = 0 \text{ or } 1 \qquad (4)$$

模型의 目的函數 (1)는 각 作業의 處理完了時間의 納期를 超過한 納期遲延時間 중에서 최대치를 최소화하는 것을 나타낸다. 제약조건식 (2)은 모든 작업의 처리완료시간과 납기와 차이가 최대납기지연시간(L)보다는 적거나 같다는 것을 나타낸다. 제약조건식 (3), (4)은 하나의 個別作業은 반드시 어느 한 대의 기계를 통해서만 加工이 되어야 함을 의미한다. 이 模型에서 모든 작업은 납기가 적은 순서로 整列이 되어 있다고 假定한다. (즉 $i < i' \Rightarrow d_i < d_{i'}$)

만약 마지막 $x_{Ij} = 1$ 이면 제약조건식 (2)의 왼쪽 항목은 작업 I 의 납기지연시간을 나타낸다. 만약 $x_{Ij} = 0$ 이고 $x_{I'j}$ 가 마지막 非陰항이면 ($I' < I$), $\sum_{i=1}^{I} t_{ij} x_{ij} - d_{I'}$ 는 작업 I'의 납기지연시간을 나타낸다. 그러나 $d_{I'} \leq d_I$로 가정했으므로 중복제약조건식(redundant)이 될 뿐이다.

실제로 제약조건식 (2)의 $n \times m$ 가 제약조건식 중에서 n개의 식을 제외한 나머지 $n \times (m-1)$ 개의 제약조건식은 重複制約式(redundant constraints)이 된다[1].

1) 문제를 명확히 하기 위한 例題問題의 모형은 부록을 참조하기바람.

2.3. 最大納期遅延時間의 最小化問題의 計算複雜度

앞절에서 제시한 최대납기지연시간의 최소화문제에 대한 정수계획문제는 만약 모든 작업의 납기가 0 ($d_i = 0$, for all i)이게 되면 아래 整數計劃問題와 같이 最大納期遅延時間의 最小化(minimizing makespan) 問題로 축소될 수 있다.

(P) $Min \quad M$ [2] (1)

$$s.\ t. \qquad \sum_{i=1}^{I} t_{ij} x_{ij} \leq M \qquad \text{for } j = 1,2,\cdots,m \tag{2}$$

$$\sum_{j=1}^{m} x_{ij} = 1 \qquad \text{for } i = 1,2,\cdots,n \tag{3}$$

$$x_{ij} = 0 \ or \ 1 \tag{4}$$

여기서 첨자집합(index set)을 다음과 같이 정의한다.

$i = \{1,2,\cdots,n\}$ 작업(job)들의 집합

$j = \{1,2,\cdots,m\}$ 기계(processor)들의 집합

그리고 각 변수와 모수들은 다음과 같은 의미를 갖고 있다.

$$x_{ij} = \left\{ \begin{array}{l} 1 \ \text{만약 작업 } i\text{가 기계 } j\text{에서 가공된다면} \\ 0 \ \text{그렇지 않으면} \end{array} \right\} \begin{array}{l} i = 1,2 \ldots m \\ j = 1,2 \ldots n \end{array}$$

M = 최대납기지연시간

t_{ij} = 작업 i 의 가공시간

d_i = 작업 i 의 납기

2) 안상형, 이송근, " 병렬처리기계상에서 총작업완료시간의 최소화해법에 관한 연구, 한국경영과학회지, 제16권 제2호, 1991. pp. 17-18.

따라서 最大納期遲延時間의 最小化問題는 安과 李[1991]의 연구에서 증명되었던 總作業完了時間의 最小化問題와 마찬가지로 가장 간단한 형태인 2대의 等屬機械問題에서 조차 難解性 組合的 最適化問題(NP-complete problem)가 된다. 그러므로 異速器械에서는 더욱 어려운 문제가 되어 최적해를 구하는 일반적인 多項時間解法(polynomial time algorithm)을 개발하는 것은 난망해진다.

3. 最大納期遲延時間의 最小化問題의 휴리스틱 解法

3.1. EDD 解法

일반적으로 單一機械問題(single processor)문제에 最大納期遲延時間問題는 納期가 적은 順으로 優先해서 作業을 配定하는 것이다. 이것을 EDD(the Earliest Due Date) 順序라고 한다.

병렬처리기계문제에 이 순서를 適用하기 위해서 우선 모형에서 가정한대로 작업을 납기가 적은 순으로 정렬시킨 다음 하나씩 기계간 負荷를 均等化(load equalization)시키는 방향으로 작업을 배정하는 것이다. 이것을 휴리스틱 EDD라 하며 그절차는 다음과 같다.

S_j 를 기계 j 에 대한 현재까지 누적된 작업시간, 즉 部分作業完了時間(partial makespan)을 나타낸다면

(1 단계) 모든 $j = 1, 2, \cdots, m$에 대하여 $S_j \leftarrow 0$으로 초기화시킨다.

(2 단계) 각 작업의 납기 d_i 를 오름차순으로 정돈하여 납기가 빠른 작업

부터 1, 2, …, n 으로 순번을 부여한다.

(3 단계) I = 1부터 n 까지 다음과정을 반복한다.

(a) min $(S_j + t_{ij})$ 에 해당하는 기계 j 를 찾아 해당기계를 j(i) 로 規定한다.

만약 同率이 발생하면 j 의 작은 값을 취한다.

(b) 작업 I를 기계 j(i) 에 배정하고 부분작업완료시간 $S_{i(j)}$를 다음과 같이 갱신한다.

$$S_{j(i)} \leftarrow S_{j(i)} + t_{ij(i)}$$

부록의 예제문제에 EDD 해법을 적용하면 그 결과는 다음과 같다.

시 간	0	5	10	15(日)
기계 1	작업1		(-2)	
기계 2	작업2(-10)	작업5	(-11)	
기계 3	작업3(-11)	작업4 (-9)		

( )는 납기지연시간

납기지연시간(lateness)은 陰數로도 나타날 수 있으며 이것은 納期以前에 작업을 완료할 수 있음을 나타낸다. 위 문제의 경우는 모든 작업을 납기 내에 완료할 수 있으며, EDD 해법에 의한 최대납기지연시간은 -2 이다.

3.2 LPT 해법

EDD 順으로 작업을 배정할 경우 相對的으로 작업시간이 긴 작업이 너무 늦게 配定되어 이 작업 때문에 최대납기지연시간이 크게 될 가능성이 있다. 이것을 피하기 위해서 최장작업시간의 작업(Longest Processing Time Job)부

터 우선적으로 배정할 수도 있다.

이때 等屬機械(equal processor)에서는 LPT를 결정하기 쉬우나 異速機械(unequal processor)에서는 기계간 작업계산이 틀리기 때문에 평균작업시간 즉 $\overline{t_i} = \frac{1}{m}\sum_{j=1}^{m} t_{ij}$를 이용하기로 한다.

LPT 순으로 배정하는 경우에 어느 하나의 기계에서 보면 납기가 늦은 작업이 납기가 빠른 작업보다 먼저 배정될 수도 있기 때문에 최종배정이 끝난 뒤 각 기계별로 EDD 순서로 다시 정돈해야 한다.

(1 단계) 모든 $j = 1, 2, \cdots, m$에 대해서 $S_j \leftarrow 0$ 으로 초기화시킨다.

(2 단계) $\overline{t_i} = \frac{1}{m}\sum_{j=1}^{m} t_{ij}$ 를 계산하면 $\overline{t_i}$ 는 작업 i 의 각 기계에서의 작업시간의 평균치가 된다. 이 $\overline{t_i}$ 를 내림차순으로 정돈하여 각 작업에 $1, 2, \cdots, n$ 의 순번을 부여한다.

(3 단계) i = 1부터 n 까지 다음과정을 반복한다.

(a) $\min (S_j + t_{ij})$ 에 해당하는 기계 j 를 찾아 해당기계를 j(i)로 규정한다. 만약 동률이 발생하면 j 의 작은 값을 취한다.

(b) 작업 I를 기계 j(i) 에 배정한다.

부분작업완료시간 $S_{j(i)}$를 다음과 같이 갱신한다.

$S_{j(i)} \leftarrow S_{j(i)} + t_{ij(i)}$

(4 단계) 각 기계별로 EDD 순으로 재정돈한다.

3.3 De and Morton 의 解法

竝列處理機械의 問題에 대한 旣存의 硏究에는 準最適解(near-optimal solution)를 구하는 휴리스틱解法의 開發과 이 解法의 成果分析을 통해서 계산해 質을 높이고자 한 De와 Morton(1978) 연구가 있다. De와 Morton은 A, B, C 의 3가지 휴리스틱을 제시하고 각 휴리스틱의 성과를 比較 分析하였다.

EDD와 LPT는 작업시간과 납기의 성격에 따라 그 결과가 다르게 나타나기 때문에 우열을 비교하기가 힘들다. 그러므로 양자를 보다 조화롭게 절충할 필요가 있다.

일반적으로 최대납기지연시간의 발생을 피하기 위해서는 다음 3 가지 노력이 필요하다.

첫째, 각 기계에서 작업을 EDD 순으로 配列한다.

둘째, 납기가 빠른 작업부터 각 기계에 均等하게 配定한다.

셋째, 작업시간이 긴 작업이 너무 늦게 배정되는 것을 피한다.

첫 번째 목표는 쉽게 달성될 수 있으나 두 번째의 EDD 순서와 세 번째의 LPT 순서는 서로 상치될 가능성이 있다. 그러므로 납기와 작업시간의 상대적인 중요성을 반영하는 EDD와 LPT의 최적조합(optimal combination)을 구할 수 있다면 좋을 것이다. 最適組合을 찾기 위하여 De와 Morton은 각 작업에 대해서 다음 수식을 계산하였다.

$$P_i = rd_i - (1-r)\overline{t_i}$$

여기서 $\overline{t_i} = \frac{1}{m}\sum_{j=1}^{m} t_{ij}$ 이며, r 은 $0 \leq r \leq 1$의 임의의 수를 이용하여 작업의 배정순서를 납기와 평균작업시간을 절충한 P_i 의 오름차순으로 한다는 것으로, 서로 다른 r 값을 시도하여 최선의 결과를 취한다. 이때 r = 1

이면 P_i 의 순서는 EDD 순서와 일치하고 r = 0 이면 LPT 순서와 같다. P_i 의 계산에 기초하여 De와 Morton은 휴리스틱 A 와 휴리스틱 B를 제시하였다.

휴리스틱 A는 P_i 의 순서에 따라서 기계간 부하를 균등히 한다는 것으로 EDD 와 LPT 의 절충이라고 볼 수 있다.

(1) 휴리스틱 A

(0 단계) 서로 다른 r 값을 적용하면서 1 단계에서 4 단계까지를 반복 계산하여 최상의 해를 구한다.

(1 단계) 모든 기계 $j = 1, 2, ,\cdots, m$ 에 대하여 $S_j \leftarrow 0$으로 초기화시킨다.

(2 단계) $P_i = rd_i - (1-r)\overline{t_i}$ 의 오름차순으로 작업순서를 재정돈하고, 재정돈 된 작업에 $1, 2, ,\cdots, n$ 의 일련번호를 부여한다.

(3 단계) EDD의 3단계와 동일함

(4 단계) LPT의 4단계와 동일함

만약 K 개의 r 값을 적용했다면 이 휴리스틱의 計算複雜度(time complexity)는 $O(K(n \log n + nm))$이다.

휴리스틱 B 는 機械間 負荷의 均等化는 무시하고 전적으로 納期遲延(lateness)의 最小化에만 重點을 둔다. 즉 P_i 의 순서에 따라 작업을 배정하되 매 단계마다 최대납기지연시간을 최소화시키는 기계를 찾아 작업을 배정하는 방법이다.

(2) 휴리스틱 B

L_j 를 j 기계에 대한 현재까지의 최대납기지연시간 이라고 하면

(0 단계) 서로 다른 r 값을 적용하면서 1 단계에서 4 단계까지를 반복계산하여 최상의 해를 구한다.

(1 단계) 모든 기계 $j = 1, 2, ,\cdots, m$에 대해서 $L_j \leftarrow 0$으로 초기화시킨다.

(2 단계) $P_i = rd_i - (1-r)\overline{t_i}$ 의 오름차순으로 작업순서를 재정돈하고, 재정돈 된 작업에 $1, 2, ,\cdots, n$ 의 일련번호를 부여한다.

(3 단계) i = 1 부터 n 까지의 다음과정을 반복한다.

(a) j = 1 부터 m 까지의 다음과정을 반복한다.

(i) EDD 순서를 유지하면서 작업 i 가 기계 j 에 배정될 적절한 위치(proper position를 찾는다. 이 위치를 K(j)라 한다.
납기가 同率인 작업에 대해서는 K의 적은 값을 선택한다.

(ii) 잠정적으로 작업 i를 기계 j 상의 K(j)번째 위치에 배정한다. 기계 j 상의 새로운 최대납기지연시간을 계산하여 이를 L'_j 라 한다.

(b) $\min_j \{L'_j\}$에 해당하는 기계 j를 구한다. 이 기계를 j(i)라 한다.
만약 同率이 발생하면 j가 적은 값을 취한다.

(c) 작업 i를 기계 j(i)의 K(j(i))번째 위치에 배정한다.
$L_{j(1)} \leftarrow L'_{j(i)}$ 로 갱신한다.

휴리스틱 B 에서는 각 기계 상에서 EDD 순서가 항상 유지되므로 마지막에 별도의 순서 재배열이 필요 없다. K 개의 r 값을 적용할 때 휴리스틱 B의 계산복잡도는 $O(K(n^2 + nm))$ 이다.

(3) 휴리스틱 C

일단 양질의 휴리스틱해가 구해진 뒤에도 적정규모이내의 문제에 대해서는 機械間 作業交換(interchanging jobs between processor) 을 통해서 해의 改善을 도모할 수 있다. 휴리스틱에 의한 最大納期遲延時間이 $Lmax$라고 하고 이것이 $j_{\max}$ 기계상의 $i_{\max}$ 위치에서 발생한다고 가정하자.

($1 \leq i_{\max} \leq n_{j_{\max}}$, 여기서 n_j 는 j 기계에 旣配定된 작업의 개수를 의미한다.) 그러면 $j_{\max}$ 기계상의 $i_{\max}$ 위치 앞의 어떤 작업을 제거한다던가 다른 작업과 위치를 相互交換(replacement)함으로서 $L_{\max}$ 값의 개선을 도모할 수 있다.

P 를 $j_{\max}$ 기계의 i 번째 위치에 있는 작업이라고 가정한다. ($i \leq i_{\max}$) P를 $j_{\max}$ 기계로부터 어떤 다른 기계 j 로 옮긴다고 생각해 보자.

EDD 순서를 유지하기 위해서 P는 기계 j 상의 r-1 과 r번째 위치 사이에 놓여지게 된다. (즉 기계 j 에 할당된다면 P의 위치는 r 이 된다.) 그러면 기계 j상에서 r보다 크거나 같은 위치의 모든 작업들은 우측으로 이동하게 되며 따라서 L_j의 증가를 가져온다. 마찬가지로 $j_{\max}$ 기계상의 i 보다 우측위치에 있는 모든 작업들은 좌측으로 이동하게 되어 $L_{j_{\max}}$의 감소를 가져올 것이다.

이것은 $j_{\max}$ 기계에서 작업 P를 다른 작업과 상호교환 없이 다른 기계를 옮기는 것이다. 그러나 다른 작업과의 교환도 물론 가능하다. j 기계상의 K 번째 위치의 작업을 q라고 하고, 이 작업과 P를 상호교환한다고 가정하자. EDD 순서를 유지하기 위해서 q는 $j_{\max}$기계상의 S 번째 위치에 삽입된다. 이 경우에도 두 기계 상에서 기존 작업들의 위치가 변동하게 된다.

이 원리에 기초해서 어떤 주어진 $j_{\max}$에 있어서 모든 작업 $i \leq i_{\max}$에 대

해서 모든 $j \neq j_{max}$ 기계상의 작업과 상호교환을 시도해 본다, 한 번의 교환이 한 번의 반복계산(iteration)으로 수행되며, 전체 반복계산과정에서 이루어지지 않으면, 휴리스틱해의 개선은 없다고 본다.

개선이 있는 경우에는 새로운 j_{max} 기계와 i_{max} 작업을 가지고 새로운 반복계산이 수행된다.

이 과정은 한 번에 두 개씩의 작업교환이 이루어지기 때문에 , 해의 최적성을 보장해 주지는 못하지만 상당한 改善效果가 있을 것으로 推定된다.

n_j, L_{max}, i_{max} j_{max}가 위에서 설명한 것을 의미하는 기호라고 한다.

(1 단계) $j \leftarrow 0$

(2 단계) $j \leftarrow j+1$, 만약 $j = j_{max}$이면 $j \leftarrow j+1$, 만약 $j > m$이면 중지한다.

(3 단계) $i \leftarrow 0$

(4 단계) $i \leftarrow i+1$, 만약 $i = i_{max}$이면 2 단계로 돌아간다.

(5 단계) P가 j_{max} 기계상의 i 위치에 있는 작업이라면 잠정적으로 P를 j_{max} 기계 상에서 제거한다. $n'_{j_{max}} \leftarrow n_{j_{max}} - 1$

j_{max} 기계상의 작업들에 대한 새로운 최대납기지연시간 $L'_{j_{max}}$를 계산한다.

(6 단계) $K \leftarrow 0$ 10 단계로 간다.

(7 단계) $K \leftarrow K+1$, 만약 $K > n_j$ 이면 4단계로 간다.

(8 단계) q를 기계 j의 K 번째 위치에 있는 작업이라고 하면 잠정적으로 q를 기계 j 상에서 제거한다.

(9 단계) (a) EDD 순서에 의거해서 j_{max} 기계 상에 q를 삽입할 위치 I를

찾는다.

(b) 점증적으로 작업 q를 j_{max} 기계상의 I 번째 위치에 할당한다.

j_{max}기계상의 새로운 최대납기지연시간을 계산 하여 이를 $L'_{j_{max}}$라 한다.

(10 단계) (a) EDD 순서에 의거해서 기계 j 상에 작업 P를 삽입할 위치를 찾는다. 이 위치를 r이라 한다.

(b) 잠정적으로 P를 기계 j 상의 r 번째에 할당한다.

기계 j상의 새로운 최대납기지연시간을 계산하여 이를 L'_j라 한다.

(11 단계) $L' \leftarrow \max\{L'_j, L'_{j_{max}}\}$ 만약 $L' \geq L_{max}$ 이면 7단계로 간다.

(12 단계) 작업 P를 기계 j상의 r 번째 위치에 할당한다.

$L_j \leftarrow L'_j$; $n_j \leftarrow n_j+1$로 갱신한다.

(13 단계) 만약 K = 0이면 $n_{j_{max}} \leftarrow n'_{j_{max}}$ 로 갱신하고 15 단계로 간다.

(14 단계) 작업 q를 j_{max} 기계상의 S번째 위치에 할당한다.

$L_{j_{max}} \leftarrow L'_{j_{max}}$; $n_{j_{max}} \leftarrow n_{j_{max}}+1$로 갱신한다.

(15 단계) (a) $L \leftarrow \max\{L_j\}$ 라면 해당 기계 j를 j''라고 하고 기계 j'' 상에서 최대납기지연이 발생하는 위치를 i''라 한다.

(b) $L_{max} \leftarrow L$; $j_{max} \leftarrow j''$; $i_{max} \leftarrow i''$로 갱신한다.

1 단계로 되돌아간다.

K를 반복계산회수의 상한치라면 휴리스틱C의 계산복잡도는

$O(K \cdot \frac{n^2}{m^2}(n+m))$ 이 된다.

4. 最適解法의 構成

휴리스틱해법의 性能은 計算時間, 最適解와의 平均格差, 最惡限界 등의 觀點에서 評價되어야 하나 이중 가장 중요한 요소는 最適解와의 平均格差를 평가하기 위해서 일반적으로 다음과 같은 成果尺度가 이용된다.

$$Z = \frac{Z_1 - Z_2}{Z_2} \times 100 \qquad (Z_1 : \text{휴리스틱해의 값}, \quad Z_2 : \text{최적해의 값})$$

여기서 Z 는 특정 휴리스틱해의 最適解로부터 比率隔差(percentage deviation)를 나타낸다. 따라서 휴리스틱해의 性能을 評價하기 위해서는 필연적으로 最適解를 구해야 하나 앞에서 언급한 바와 같이 병렬처리기계상에서 최대납기지연시간의 최소화문제는 강력한 난해성조합최적화문제(NP-hard)에 속하기 때문에 최적해를 구하기가 쉽지 않다.

De 와 Morton은 單純列擧法(enumeration method)에 기초한 分段探索法(branch and bound)을 이용하여 最適解探索을 試圖하였으나 90 문제 중 27문제의 최적해를 發見하였고, 그것도 대부분 10개의 작업과 2대 또는 3대의 기계문제가 대부분이었다. 20개 작업에 대해서 최적해를 발견한 것은 단지 2개의 문제에서였다.

보다 실용적인 문제규모(기계 10대, 작업 50대) 까지 最適解 探索을 試圖하기 위해서는 보다 강력한 下限價(lower bound)를 구할 필요가 있는데 이를 위해서 라그랑지안이완(Lagrangean Relaxation)을 적용해 보기로 한다. 라그

랑지안 이완법은 어떤 整數計劃問題가 일부 難解한 制約條件式 때문에 풀이가 어려운 경우에 일부 제약조건식을 이완시켜 목적함수에 포함시킴으로써 비교적 풀이가 容易한 문제로 재구성하는 기법이다.

2 장에서 제시한 原問題(P)의 라그랑지안 이완식은 어느 제약조건식을 이완시키느냐에 따라서 두 가지 방법을 고려해 볼 수 있다. 즉 제약조건식(2)을 이완시킨 경우와 제약조건식(3)을 이완시킨 경우로 나눌 수 있는데, 전자를 (LR I)이라고 하고 후자를(LR II)라고 하자.

우선 (LR II)의 경우는 제약조건식이 다수제약식 배낭문제(multiple constrains knapsack problem)가 되며 이 문제 자체가 또 하나의 NP-complete 문제가 되므로 해를 구하기가 사실상 不可能하다. 또한 제약조건식의 구조가 총작업완료시간의 최소화문제처럼 단일제약식 배낭문제(single constraint knapsack)로 分解될 수 있는 성질도 아니기 때문에 (LR II)를 통한 下限價 導出은 抛棄하고 (LR I)에 초점을 맞추어 下限價를 구해보기로 한다.

U_{ij} 를 라그랑지안 승수(Lagrangean multiplier)라면 (LR I)은 다음과 같다.

(LR I)

$$Z_{LR} = Min\left[L + \sum_{j=1}^{m}\sum_{I=1}^{n} U_{ij}\left(\sum_{i=1}^{I} t_{ij}\, x_{ij} - d_I - L\right)\right]$$

$$= Min\left[L + \sum_{j=1}^{m}\sum_{I=1}^{n} U_{ij}\sum_{i=1}^{I} t_{ij}\, x_{ij} - \sum_{j=1}^{m}\sum_{I=1}^{n} U_{Ij}\, d_I - \sum_{j=1}^{m}\sum_{I=1}^{n} U_{Ij}\, L\right]$$

$$= Min\left[\sum_{j=1}^{m}\sum_{I=1}^{n} U_{ij}\sum_{i=1}^{I} t_{ij}\, x_{ij} - \sum_{j=1}^{m}\sum_{I=1}^{n} U_{Ij}\, d_I + L\left(1 - \sum_{j=1}^{m}\sum_{I=1}^{n} U_{Ij}\right)\right]$$

s. t. $\sum_{j=1}^{m} x_{ij} = 1$ for $i = 1, 2, \cdots, n$

$x_{ij} = 0$ *or* 1, L 은 자유변수 (free variable)

(LR I)의 풀이를 용이하게 하기 위하여 납기지연시간 L에 대한 상한가와 하한가를 구하여 제약조건식에 추가시켜본다. 휴리스틱에서 구한 최선의 값을 L 에 대한 上限價(upper bound) 로 간주할 수 있다. 또한 L 에 대한 하한가는 주어진 문제의 구조에게 구할 수 있는데, 임의의 작업이 각 기계에 배정된다고 가정했을 때 기존에 배정된 작업이 없다고 간주하면 각 기계별 (작업시간 - 납기)의 최소치가 L의 하한가가 될 수 있다.

즉 부록의 예제문제에서 작업 1 은 1 기계에서 13 -15 = -2, 기계 2 에서 18 - 15 = 3, 기계 3 에서 15 - 15 = 0 임으로 다른 작업의 배정여부에 관계없이 L 은 -2 이하로 내려갈 수는 없다. 이 값을 전 작업별로 계산해서 가장 큰 값이 L 의 하한가가 된다. 이 값을 構造的 下限價(structural lower bound) 라고 하자.

그러면 (LR I)의 제약조건식에는 L 에 대한 상한가 $\overline{L}$ 와 하한가 $\underline{L}$ 가 추가되어 다음과 같이 된다.

$$Z_{LR} = Min\left[\sum_{j=1}^{m}\sum_{I=1}^{n} U_{ij}\sum_{i=1}^{I} t_{ij}x_{ij} - \sum_{j=1}^{m}\sum_{I=1}^{n} U_{Ij}d_I + L\left(1 - \sum_{j=1}^{m}\sum_{I=1}^{n} U_{Ij}\right)\right]$$

s. t. $\sum_{j=1}^{m} x_{ij} = 1$ for i= 1, 2, $\cdots$, n

$\underline{L} \le L \le \overline{L}$

$x_{ij} = 0$ *or* 1, L 은 자유변수(free variable)

이 문제에 대한 최적해는 다음과 같이 구해진다.

$$L = \begin{cases} \underline{L} & ; \quad (1-\sum_{j=1}^{m}\sum_{I=1}^{n} U_{Ij}) > 0 \text{ 일때} \\ \overline{L} & ; \quad (1-\sum_{j=1}^{m}\sum_{I=1}^{n} U_{Ij}) < 0 \text{ 일때} \\ \underline{L} \text{ 과 } \overline{L} \text{ 사이의 임의의 값}; & (1-\sum_{j=1}^{m}\sum_{I=1}^{n} U_{Ij}) = 0 \text{ 일때} \end{cases}$$

$$x_{ij} = \begin{cases} 1 ; & \text{만약 } C'_{ij} = Min\{C_{ij}\} \text{ 이면, for } i=1,2,\ldots,n \\ 0 ; & \text{아니면} \end{cases}$$

이때 $C_{ij} = t_{ij} \sum_{I=1}^{n} U_{ij}$ 로 정의한다.

라그랑지안 이완문제 (LR I)의 최적해를 라그랑지안 乘數 U_{ij} 의 調整을 통해서 最上의 最適解 $Max\ U\{Z_{LR}\}$ 를 구하기 위해서 서브그래디언트 技法을 적용한다. 여기서 구해진 Z_{LR} 은 原問題 (P) 에 대한 下限價 (lower bound) 가 된다. 이 하한가가 어느 정도 정밀할 것이냐는 문제의 성격에 따라 달라지므로 실제 실험을 통해서 確認하는 수밖에 없다.

휴리스틱에서 구한 최상의 上限價와 라그랑지안 이완문제에서 구한 최상의 하한가 사이에 격차가 존재하면 이 사이에서 最適解를 구하기 위한 분단탐색법을 구성하여 하고 일반적으로 막대한 계산시간이 소요된다. 만약 위에서 구한 상한가와 하한가가 최적해에 대단히 근접하여 대부분의 문제에서 상한가=하한가로 나타난다면 분단탐색법에 의존하지 않고도 최적해를 확인할 수 있다. 일단 後者의 可能性을 實驗을 통해서 打診해 보고자 한다.

5. 새로운 휴리스틱

5.1. 휴리스틱 S

여기서는 앞에서 설명한 De 와 Morton의 휴리스틱과는 별도로 기계간 작업부하의 균등화에 초점을 두는 새로운 휴리스틱을 제시한다. 모든 납기를 일단 제로라고 가정하면 최대납기지연시간의 최소화문제는 총작업완료시간의 최소화문제가 된다는 것은 II장에서 이미 밝힌 바 있다.

安 과 李의 先行硏究(1991)에서 總作業完了時間의 最小化에 가장 强力한 휴리스틱으로서 휴리스틱 S 를 提示한 바 있다. 휴리스틱 S 는 기계간의 負荷를 最大한 均等化시킴으로써 총작업완료시간의 최소화를 도모한다. 이 개념을 도입하여 일단 납기를 무시하고 총작업완료시간을 최소화시키는 작업배정(또는 기계간 부하를 최대한 균등화시키는 작업배정) 을 한 다음, 각 기계별로 EDD 순서로 재배열하면 기계 수에 비해서 작업수가 많은 경우에 효과가 클 것으로 추정된다. 휴리스틱 S 는 機會費用(opportunity cost)의 槪念에 基礎하여 작업처리의 우선 순위를 결정한다. 이때 기회비용은 어떤 미배정 작업 i 에 대해서 현재까지의 각 기계별 부분작업완료시간 S_j와의 합 즉 $S_j + t_{ij}$ 가 가장 적은 기계(最善의 機械)와 그 다음 적은 기계(次善의 機械)와의 차이를 의미하며, 이 차이가 큰 작업부터 최선의 기계에 우선 배정시켜 나감으로서 기회비용을 최소화하는 것이 機械負荷의 均等化를 가져오고 나아가서 총작업완료시간의 최소화에 기여하게 한다. 그리고 이 해법의 후반부에는 또한 LPT 에 의한 작업 배분도 도모한다.

(휴리스틱 S)

(0 단계) 여러 가지 다른 β 값에 대해서 제 1 단계부터 제 5 단계

까지 반복 적용하여 최상의 결과를 선택한다. ($o < \beta \leq 1$)

(1 단계) 모든 기계 j 에 대해서 部分作業完了時間 $S_j \leftarrow 0$ 으로 초기화시킨다.

(2 단계) 모든 未配定作業 i, $i \in T$ 에 대해서

$$K1_{ij}^{*} = \min_{j} \{s_j + t_{ij}\}$$

$$K2_{ij}^{**} = \min_{j \neq j^{*}} \{s_j + t_{ij}\}$$

$K_i = K2_{ij}^{**} - K1_{ij}^{*}$를 구한다.

각 작업 i 에 대해서 해당하는 j^{*} 를 j(i) 로 규정함.

(3 단계) (a) $K = \max_{i \in T} \{K_i\}$를 발견한다.

해당 i 를 $\bar{i}$ 로 정의한다.

만약 max K_i가 둘 이상의 작업에서 同率이 발생하면 $K1_{ij}^{*}$가 작은 작업 i 부터 배정한다.

(b) 작업 $\bar{i}$ 를 기계 $j_{(\bar{i})}$ 에 배정한다.

$S_{j(\bar{i})} \leftarrow S_{j(\bar{i})} + t_{ij(\bar{i})}$, $T \leftarrow T - \{\bar{i}\}$ 로 갱신한다.

(c) 만약 $|T| > \beta \cdot n$ 이면 제 2단계로 돌아간다.

아니면 제 4단계로 간다.

(4 단계) 미배정된 모든 작업 i, $i \in T$ 에 대해서 $K_i = \frac{1}{m} \cdot \sum_{j=1}^{m} t_{ij}$ 를 계산하여, K_i 를 내림차순으로 정렬하여, 각4 작업에 새로운 순위를 부여한다.

(5 단계) 모든 未配定作業 $i \in T$ 에 대해서

(a) $\min_j \{S_j + T_{ij}\}$를 발견하여 해당기계 j 를 $j_{(i)}$ 으로 규정한다.

(b) 작업 i 를 기계 $j_{(i)}$ 에 배정한다.

$S_{j(i)} \leftarrow S_{j(i)} + t_{ij(i)}$ 로 갱신한다.

(c) 미배정작업집합 T 가 공집합이 되면 ($T = \emptyset$) 해법진행 이 종료됨.

아니면 제 5단계로 되돌아감.

(6 단계) 각 기계별로 EDD 순으로 재정돈한다.

휴리스틱 S 의 계산복잡도는 다음과 같다.

(1 단계) : $O(m)$

(2, 3 단계) : $O(mn_1^2)$, 여기서 n_1은 $n_1 \leq \beta n$ 을 만족시키는 最大整數

(4, 5 단계) : $n_2 = n - n_1$이라면 $O(n_2 \log n_2 + n_2 m)$

따라서 1회 반복시의 계산의 복잡도는

$O(mn_1^2 + n_2 \log n_2 + n_2 m) \leq O(mn^2 + n\log n)$

(여기서 $n_1, \ n_2 < n$)

k 회 반복계산이 있었다면 $O(kmn^2 + n\log n)$이 휴리스틱 S의 계산복잡도가 된다.

5.2 새로운 휴리스틱의 效率性 評價

앞에서 제시한 휴리스틱기법들과 최적해법의 성능을 평가하기 위하여 30

개씩의 실험문제를 만들었다. 각 실험문제의 작업시간행렬 t_{ij} 는 $0 \leq t_{ij} \leq 50$ 의 整數를 離散的 一樣分布(discrete uniform distribution)에 의거하여 難數(random number)로 구성하였다. 또한 납기 d_i 는 $0 \leq d_i \leq 100$ 의 정수를 동일한 방법으로 구하였다. 문제의 규모는 기계대수를 2대에서 10대까지, 작업개수는 10개에서 50개까지 차례로 증가시켜 가면서 양자를 적당히 조합하여 구성하였다. 異速機械에서의 성과비교를 위해서 實驗結果를 (표 5-1)에 정리하였다.

(표 5-1)의 실험결과에 의하면 우선 휴리스틱 EDD, LPT, A, B, S 를 비교한 결과 5개의 휴리스틱 중에서 A > B > S > EDD > LPT 순으로 성과가 나타났다. 이중에서 A 와 B 는 서로 비슷하게 증감하는데 비해서 S 는 기계대수에 비해서 작업수가 많은 경우에 성과가 상당히 좋게 나타나는 경향이기 때문에 휴리스틱 A 와 결합한 휴리스틱 SA 를 구성해 본 결과 30 문제 중 28문제에서 최상의 결과를 실현하였다.

마지막으로 휴리스틱 S의 결과를 상한가(upper bound)로 하여 라그랑지안 이완 문제 (LR I)에 대한 서브그래디언트기법을 적용한 결과 30문제 중 모두 15개의 문제에서 상한가와 일치하는 강력한 하한가가 도출됨을 발견하였다. 따라서 약 50% 정도의 문제에서 분단탐색법에 의존하지 않고도 최적해가 발견되었다. 최적해가 발견되지 않은 문제에 있어서도 약 40% 정도의 문제에서 하한가와 상한가 차이가 매우 작게 나타남으로 엄밀한 하한가를 제공해 준다고 할 수 있다. 즉 약 70% 정도의 문제에서 최적해 또는 엄밀한 하한가를 제공해 준다고 할 수 있다.

이 결과는 De 와 Morton 이 기계대수 4 × 작업개수 10개 정도의 문제까지 최적해를 구할 수 있었던 것과 비교하면, 최적해를 발견할 수 있는 문제의 규모를 상당히 확장시켰다고 볼 수 있다. 이 실험에 이용한 컴퓨터는

486DX-80 PC 를 사용하였는 바 대형기종을 통해 실험한다면 보다 큰 규모의 문제도 해결할 수 있으리라고 추정된다.

아래 (표 5-2)는 (표 5-1) 을 이용하여 구한 각 휴리스틱해의 최적화의 비율격차(percentage deviation)를 나타낸다.

(표 5-2)를 볼 때 휴리스틱 S 와 휴리스틱 A 를 결합한 휴리스틱 SA의 평균이 3.05% 로 휴리스틱 A 의 6.86% 보다 훨씬 적었다. 이는 휴리스틱 SA 가 휴리스틱 A 에 비해 더욱더 최적해에 근접하는 휴리스틱 알고리즘이라는 것은 보여주는 것이다.

(표 5-1) 異速機械問題

실 험 문 제		EDD	LPT	A	B	I	IA	LB	최적해 확인여부
기계대수	작업개수								
2	10	55	65	49	49	64	49	48	△
	20	108	109	108	104	100	100	99	△
	30	230	257	219	223	226	219	218	△
	40	321	287	280	282	251	251	250	△
	50	419	358	359	349	334	334	334	○
3	10	12	17	12	12	17	12	12	○
	20	62	57	48	48	57	48	48	○
	30	83	87	71	74	84	71	62	×
	40	130	127	119	116	96	96	93	×
	50	189	162	154	151	153	153	140	×
4	10	-7	-7	-7	-7	-7	-7	-7	○
	20	13	26	13	13	26	13	13	○
	30	21	37	14	21	16	14	14	○
	40	37	68	31	37	46	31	25	×
	50	82	77	71	68	55	55	52	×
5	20	11	13	7	7	11	7	5.47	○
	30	17	29	15	15	29	15	13	×
	40	37	37	37	30	33	33	30.09	×
	50	33	59	28	29	25	25	25	○
6	20	5	5	5	5	5	5	5	○
	30	13	20	13	13	18	13	12.05	△
	40	8	8	8	8	8	8	8	○
	50	11	23	11	13	21	11	11	○
8	30	22	22	22	22	26	22	22	○
	40	-9	5	-9	-9	-5	-9	-9	○
	50	14	22	13	13	21	13	12.01	△
10	20	5	5	5	5	5	5	5	○
	30	14	14	14	14	14	14	14	○
	40	18	21	18	18	18	18	18	○
	50	13	9	9	9	9	9	7.98	×

(표 5-2) 휴리스틱해의 최적해와 비율격차평균

	EDD	LPT	A	B	I	IA
비율격차평균	17.57	38.08	6.86	10.28	21.54	3.05

비율격차 $Z = \frac{Z_1 - Z_2}{Z_2} \times 100$

(Z_1 : 휴리스틱해의 값, Z_2 ; 최적해의 값[3])

6. 結 論

본 연구는 병렬처리기계상에서 최대납기지연시간(maximum lateness)을 최소화하는 휴리스틱해법과 최적해법을 구하고자 하였다. 납기를 고려하지 않은 경우에 이 문제는 총작업완료시간(makespan) 최소화문제로 축소되며, 이 문제에 대해서는 이미 安 과 李 (1991)의 선행연구가 있었다. 따라서 최대납기지연최소화문제는 총작업완료시간문제보다 납기(due date)라는 요소가 추가되므로 보다 복잡한 문제가 되며, 분할가공이 허용되지 않는 경우 이 문제도 당연히 난해성 조합적 최적화문제(NP-complete)가 된다. 그러므로 多項時間이내에 수렴하는 최적해법을 구하기는 사실상 불가능하기 때문에 우선 最適解에 최대한 근접하는 準最適解를 산출하는 휴리스틱해법을 개발하였다. 다음으로 개발된 휴리스틱법의 성능을 평가하기 위하여 비교기준으로 필요한 최적해를 구하고자 라그랑지안 이완과 서브그래디언트법에 의한 효율적인 類似多項時間最適解法(pseudo-polynomial time algorithms)을 구성하여 실용적인 범위이내

3) 최적해의 값은 (표 -1)을 이용해서 구할 수 있는 값만을 포함시켰다.

의 문제에 대한 최적해의 모색을 시도하였다. 이 시도를 통해서 다음과 같은 연구성과를 거두었다.

첫째, 기존의 휴리스틱이 先納期優先(EDD), 최장시간작업우선(LPT)의 조화에 비중을 두는데 비해서, 새로운 휴리스틱해법은 機會費用(opportunity cost)의 개념에 입각한 기계간 負荷均等化(load equalization)를 최대한 실현한 다음, 이를 다시 EDD 순으로 재배열하는 방법을 개발하고(휴리스틱 S), 이를 기존의 휴리스틱 A 와 결합시켜 상호보완적으로 작용하게 함으로써 휴리스틱해의 질을 크게 개선하였다.

둘째, 原問題의 일부제약조건식을 목적함수로 이완시켜 라그랑지안 이완문제를 구성하고 서브그래디언트기법을 적용하여 이완문제의 쌍대최적해를 구해본 결과, 原問題의 最適解값에 거의 수렴하는 강력한 하한가를 구할 수 있었다.

셋째, 위에서 구한 휴리스틱해(상한가)와 라그랑지안 이완문제의 쌍대최적해(하한가)값이 많은 문제에서 原問題의 최적해와 일치하여 막대한 계산시간이 소요되는 분단탐색의 절차를 거치지 않고도 최적해를 확인할 수 있었다.

넷째, 이러한 해법의 개발로 인하여 개인 컴퓨터 (486DX-80 PC) 수준으로 최적해를 구할 수 있는 문제의 규모와 비율이 크게 향상되었다.

다섯째, 본 연구에서 개발한 새로운 휴리스틱해법이나 최적해법은 병렬처리 컴퓨터 상에서의 작업 스케줄링 문제나 생산형장의 실제 문제에 응용되어 시스템의 성과를 제고시키는 데 크게 기여할 수 있을 것이다.

그러나 De 와 Morton의 휴리스틱 알고리즘 중 기계간의 상호교환과정을 이용한 휴리스틱 C 를 추가시켜 분석해 보지는 못했다. 이에 대해서는 추가연구가 필요할 것으로 생각되어 진다.

부록: 예제문제와 모형

예제문제

기계 \ 작업	1	2	3	4	5
1	13	3	18	5	10
2	18	6	11	9	8
3	15	9	8	4	9
납 기	15	16	19	21	25

$Min \quad L$

$s. \quad t. \quad 13x_{11} - 15 \leq L$

$13x_{11} + 3x_{21} - 16 \leq L$

$13x_{11} + 3x_{21} + 18x_{31} - 19 \leq L$

$13x_{11} + 3x_{21} + 18x_{31} + 5x_{41} - 21 \leq L$

$13x_{11} + 3x_{21} + 18x_{31} + 5x_{41} + 10x_{51} - 25 \leq L$

$18x_{12} - 15 \leq L$

$18x_{12} + 6x_{22} - 16 \leq L$

$18x_{12} + 6x_{22} + 11x_{32} - 19 \leq L$

$18x_{12} + 6x_{22} + 11x_{32} + 9x_{41} - 21 \leq L$

$18x_{12} + 6x_{22} + 11x_{32} + 9x_{42} + 8x_{52} - 25 \leq L$

$15x_{13} - 15 \leq L$

$15x_{13} + 9x_{23} - 16 \leq L$

$$15x_{13} + 9x_{23} + 4x_{33} - 19 \leq L$$

$$15x_{13} + 9x_{23} + 8x_{33} + 4x_{43} - 21 \leq L$$

$$15x_{13} + 9x_{23} + 8x_{33} + 4x_{43} + 9x_{53} - 25 \leq L$$

$$x_{11} + x_{12} + x_{13} = 1$$

$$x_{21} + x_{22} + x_{23} = 1$$

$$x_{31} + x_{32} + x_{33} = 1$$

$$x_{41} + x_{42} + x_{43} = 1$$

$$x_{51} + x_{52} + x_{53} = 1$$

모든 $x_{ij} = 0$ *or* 1

參考 文獻

1. 論文

안상형, 이송근, "병렬처리기계상에서 총작업완료시간의 최소화해법에 관한 연구, 한국경영과학회지, 제16권 제2호, 1991.

Coffman, E., M.R. Garey and D.S. Johnson, *"An Application of Bin to Packing Multiprocessor Scheduling,"* SIAM J. Comput., 7, 1987, pp. 1-17.

P. De, T.E. Morton, *"Scheduling to Minimize Makespan on Unequal Parallel Processors"*, J. ACM, 26, 1979.

McNaughton, R., "Scheduling with Deadlines and Loss Functions," Management Science, Vol.6, No.1, 1959

Geoffrion, A., "Lagrangean Relaxation and its Uses in Integer Programming," Math programming Study, Vol. 2, 1974, pp. 82-114.

Ibarra, O. and C.E.Kim, "Heuristic Algorithms for Scheduling Independent Tasks on Nonidentical Processors." J. of ACM, V.24, 1979, pp. 280-289.

Gonzales, T. and S. Sahni, *"Preemptive Scheduling of Uniform Processor,"* J. of ACM, Vol.25, No.1, Jan., 1978, pp. 82-114.

2. 書籍

尹錫喆, 計量經營學, 經文社, 1991.

Baker, K., *Introduction to Sequencing and Scheduling*, NewYork, John Wiley and Sons, 1974.

C. H. Papadimitriou, K. steiglitz, Combinatorial Optimization, Prentice-Hall, 1982

Garey, M. and David S. Johnson, *Computers and Intractability*, W. H. Freeman and Company, San Francisco, 1979.

H. T. Lewis, C. H. Papadimitriou, Element of the theory of computation, Prentice-Hall, 1981.

T. C. Hu, Combinatorial Algorithm, Addison_Welsley, 1982.