韓 軍事運營分析 學會誌
10卷, 第1號, 1984.6.

0 - 1 多段階背囊技法
An algorithm for 0 - 1 Multiperiod Knapsack Problem)

權 致 明*
鄭 聖 進**

Abstract

The 0-1 multi-period Knapsack problem (MPKP) has a horizon of m periods, each having a number of types of projects with values and weights.

Subject to the requirement, the cummulative capacity of the problem in each period i cannot be exceeded by the total weight of the projects selected in period 1, 2, ..., i. It is a problem of selecting the projects in such a way that the total value in the knapsack through the horizon of m periods is maximized.

A search algorithm is developed and tested in this paper. Search rules that avoid the search of redundant partial solutions are used in the algorithm.

Using the property of MPKP, a surrogate constraint concerned with the most available requirement is used in the bounding technique.

東亞大學校 應用統計學科
서울工大 産業工學科

1. 序　論

0-1 배낭(knapsack)문제는 보통 하나의 制限된 資源의 制約下에서 과제(item,project)를 선택하여 배낭을 最適化시키는 문제이다.

0-1 多段階 背襄問題(multi-period knapsack problem: MPKP)는 m개의 기간 혹은 단계(period)가 주어지고 각 기간에 대한 制限된 資源의 量이 주어질 때 전체 기간 內에서 배낭을 最適化 시키는 과제를 선택하고자 한다. 다단계 배낭 문제는 一般的인 0-1 整數計劃 문제와 같은 形態를 취하고 있으나 制限式이 각각의 기간내에서 과제 선택에 따른 使用 資源의 量은 그대로 유지되고 增加된 기간의 決定變數의 數와 資源의 量이 增加하는 여러개의 0-1 배낭 형태를 가지고 있다.

여기에서는 일반적인 0-1 整數計劃 技法을 이용하여 배낭 문제의 성질을 添加한 효율적인 技法을 제시하고자 한다.

2. 數 式 模 型

앞으로 사용되는 기호는 다음과 같다.

- m : 기간수
- n : 선택 과제의 수
- I(i) : 기간 i의 선택 과제의 集合
- b_i : 기간 i까지의 制限 자원의 量($b_i \geq 0$)
- N : 선택 과제의 index 集合 $N=\{1,2,\cdots,n\}$
- M : 기간 i의 index 集合 $M=\{1,2,\cdots,m\}$
- X_{ij} : 기간 i에서의 決定變數 $i \in M,\ j \in I(i)$
- C_{ij} : 目的函數에서 X_{ij}의 계수 ($C_{ij} > 0$).
- a_{ij} : 制限式에서 X_{ij}의 계수 ($a_{ij} > 0$).
- $\lambda_{ij} : c_{ij} / a_{ij}$

m개의 기간과 각 기간의 과제의 수가 I(i)인 다단계 배낭 문제의 模型은 다음과 같이 쓸 수 있다.

$$\max Z = \sum_{i=j}^{m} \sum_{j \in I(i)} c_{ij} x_{ij} \cdots\cdots (1.1)$$

sub. to

$$(P) \quad \sum_{j \in I(1)} a_{1j} x_{1j} \leqq b_1$$

$$\sum_{j \in I(1)} a_{1j} x_{1j} + \sum_{j \in I(2)} a_{2j} x_{2j} \leqq b_2$$

$$\vdots \qquad \vdots$$

$$\sum_{j \in I(1)} a_{1j} x_{1j} + \sum_{j \in I(2)} a_{2j} x_{2j} + \cdots + \sum_{j \in I(m)} a_{mj} x_{mj} \leqq b_m$$

$$x_{ij} = 0 \text{ or } 1 \quad i \in M \quad j \in N \cdots\cdots (1,3)$$

여기서 c_{ij}, a_{ij}, b_i는 양의 계수이며 $0 \leqq b_1 \leqq b_2 \leq \cdots \leqq b_m$ 이다.

3. 0 − 1 다단계배낭기법

0-1 整數計劃 技法에서 많이 사용되는 探索技法(search algorithm)에서는 모든 가능한 0-1 整數解의 集合을 探索하지 않은 暗示的(implicit) 기준을 어떻게 選定하는가에 따라서 기법의 효율성이 많은 영향을 받으며, 또한 探索過程에서 sub-problem의 0-1 整數解중에서 分技點에서의 分技變數를 선택하는 기준에 따라서 많은 변수를 探索하는 時間을 절약할 수 있으며 最適解를 얻을 수 있는 段階를 단축시킬 수 있다. 0-1 multi-period knapsack 문제의

m period의 制約式을 利用하여 初期解를 빠른 段階에 구하고 얻은 解의 實行可能性(feasibilty)檢査를 통하여 非實行可能性이 가장 큰 기간을 택하여 단위 자원당 目的函數의 값을 가장 크게 증가시키는 變數를 分枝變數로 선택한다. 이와같은 段階를 거쳐서 可能解를 얻으면 다른 分枝를 통해서 解를 探索해 나아가는 기준(implicit enumeration criteria)을 정하고 그 기준에 따른 可能한 分枝點의 集合中에서 새로운 분지점으로 探索하는 戰略을 세운 후에 각 分枝의 노드(node)에서 目的函數의 값과 각기간의 制限式의 可用 資源 量을 고려하여 해를 증진시킬 수 있는 變數를 선택하여 단순 배낭(single knapsack) 形態로 변환된 문제의 限界檢査(bounding test)를 행하여 探索여부를 決定하고 더 나은 可能解를 얻을 수 있는 노드로 探索해 最適解를 얻는다.

(1) 初期解의 接近 方法

문제 (P)의 制約式 (1-2)에서 m기간만을 고려하면 0-1 단순 배낭 문제가 된다.

$$\max z = \sum_{i=1} \sum_{j \in I(i)} c_{ij} x_{ij}$$

$$\text{sub.to} \sum_{j \in I(1)} a_{1j} x_{1j} + \sum_{j \in I(2)} a_{2j} x_{2j} + \cdots + \sum_{i \in I(m)} a_{mj} x_{mj} \leq b_m$$

(PS) $\quad x_{ij} = 0 \text{ or } 1$

a_{ij}, c_{ij}, b_m은 양의 계수

$\lambda_{ij} = c_{ij} / a_{ij}$을 구하여 기간 i에 관계없이 $c_1/a_1 \geq c_2/a_2 \geq \cdots \geq c_n/a_n$ 순으로 배열하여 다음과 같은 방법으로 문제(PS)의 最適解에 가까운 解를 구한다.

$$S_1(\lambda) = \sum_{j \mid \lambda_j > \lambda} a_j$$

$$S_2(\lambda) = \sum_{j \mid \lambda_j = \lambda} a_j + s_1(\lambda)$$

$S_1(\lambda) \leq b_m < S_2(\lambda)$을 만족하는 λ^*를 구하며

$$N_1 = \{ j \mid \lambda_j > \lambda^* \}$$

$$N_0 = \{ j \mid \lambda_j < \lambda^* \}$$

$$N_2 = \{ j \in N_0 \mid b_m - S_1(\lambda) \geq \sum_{j \in N_1} a_j \}$$

$$X_{ij} = \begin{cases} 1 & j \in N_1 \cup N_2 \\ 0 & j \in N_0 - N_2 \end{cases}$$

$i \in M, j \in I(i)$

위에서 구한 해가 기간 1,2,…,m-1까지 가능해 이면 이것을 初期可能解로 使用하고 만일 非可能한 기간이 發生하면 非可能性이 가장 큰 기간을 택하여 $\lambda_{ij} = c_{ij} / a_{ij}$가 작은 순으로 $x_{ij} = 1$을 0으로 완화시켜 非可能性을 改善하여 初期可能解로 使用한다.

(2) 分枝變數의 選擇과 pivot 段階

一般的인 0-1 整數計劃 문제에 있어서는 각 列 벡터의 계수가 일정하지 않으므로 分枝의 적절한 루울을 정하기가 용이하지 않다.

MPKP에서는 각 기간의 列 벡터의 계수가 0이거나 같으므로 可用資源이 가장 큰 기간을 선택하여 단위 자원당 目的函數값을 增加시킬 수 있는 비율 즉 λ_{ij}를 구하여 λ_{ij}가 가장 큰 變數를 택하여 分枝變數로 선택한다.

만일 기간 i의 j번째 變數가 分枝變數로 決定되면,

$$X_{ij} = \begin{cases} 1 & j \in J_s \\ 0 & j \in N - J_s \end{cases}$$

$$\pi_i = b_i - \sum_{j \in J_s} c_{ij}$$

(단 J_s는 1로 고정된 변수의 집합)

을 구한다.

(3) 限界檢査

문제 (P)는 다음과 같이 표시할 수 있다.

$$\text{(P)}\quad \max\ \mathbf{CX}\quad \text{sub.to}\ \mathbf{AX} \leqq \mathbf{b},\quad X_{ij} = 0 \text{ or } 1 \quad i \in M,\ j \in I(i)$$

위의 문제에서 $X_{ij} = 1$로 고정된 $\mathbf{C}$, $\mathbf{X}$의 벡터를 $\mathbf{C}^\alpha$, $\mathbf{X}^\alpha$라 하고 制約式의 行列$\mathbf{A}$에 해당하는 submatrix를 $\mathbf{A}^\alpha$라 두면 임의의 노드 α에서 풀어야 될 문제는 다음과 같다.

$$\max\ \mathbf{C}^\beta \mathbf{X}^\beta$$

$$\text{sub.to}\ \mathbf{A}^\beta \mathbf{X}^\beta \leqq \mathbf{b}' \quad \cdots\cdots (3.1)$$

$$(p^\alpha)\quad \text{여기서}\ \begin{cases} \mathbf{X}^\beta = \mathbf{0} \text{ or } \mathbf{1} \\ \mathbf{X}^\alpha = \{1, 1, \cdots, 1\} \\ \mathbf{b}' = \mathbf{b} - \mathbf{A}^\alpha \mathbf{X}^\alpha \text{이다.} \end{cases}$$

문제 (p^α)의 식 (3,1)에 적어도 하나의 要素가 0이 아니고 非陰인 m次元의 行벡터 ω를 곱하여 非陰인 組合 (combination)을 구하면 다음과 같다.

$$\max\ \mathbf{C}^\beta \mathbf{X}^\beta$$

$$(p^{\alpha\prime})\ \text{sub.to}\ \mathbf{aX} \leqq \mathbf{b}_0 \quad \cdots\cdots (3,2)$$

$$\text{여기서}\ \begin{cases} \mathbf{X}^\beta = \mathbf{0} \text{ or } \mathbf{1} \\ \mathbf{a} = \omega \cdot \mathbf{A}^\beta \\ \mathbf{b}_0 = \omega \cdot \mathbf{b}' \text{이다.} \end{cases}$$

이와같이 하나의 制約式을 가지는 문제 $(p^{\alpha\prime})$로 변환하면 문제 (p^α)와 문제 $(p^{\alpha\prime})$사이에는 다음과 같은 關係가 成立한다.

① $\mathbf{X}$가 문제 (p^α)에 대해 可能解이면 $\mathbf{X}$는 문제 $(p^{\alpha\prime})$에 대해서도 可能解이다.

② $\mathbf{X}$가 문제 $(p^{\alpha\prime})$에 대해 最適解이고 문제 (p^α)에 대해 可能解이면, $\mathbf{X}$가 문제 (p^α)에 대해서도 最適解이다.

③ $\bar{\mathbf{X}}$가 문제 (p^α)에 대해 最適解이고 $\mathbf{X}^*$가 문제 $(p^{\alpha\prime})$에 대해 最適이면 $\mathbf{CX}^* \geqq \mathbf{C}\bar{\mathbf{X}}$ 이다.

정리 1. 행렬 $\mathbf{A}$가 列벡터 $\mathbf{A}_1, \mathbf{A}_2$로 구성되어 있고 벡터 $\mathbf{b} = (b_1, b_2)^T$, $\omega = (\omega_1, \omega_2) > \mathbf{0}$ 일때, $\omega \mathbf{AX} \leqq \omega \mathbf{b}$이고 $\mathbf{A}_2\mathbf{X} \geq b_2$라 가정하면 다음 관계가 성립한다.

i) $\mathbf{X}$는 문제 $(p^{\alpha\prime})$에 대해 $0 < \omega_2/\omega_1 \leq \dfrac{b' - \mathbf{A}_1\mathbf{X}}{\mathbf{A}_2\mathbf{X} - b_2}$이면 $\mathbf{X}$는 可能解이다

ii) $\mathbf{X}$는 문제 $(p^{\alpha\prime})$에 대해 $\omega_2/\omega_1 \geq \dfrac{b_1 - \mathbf{A}_1\mathbf{X}}{\mathbf{A}_2\mathbf{X} - b_2}$이면 $\mathbf{X}$는 非可能解이다

문제 $(p^{\alpha\prime})$의 式 (3,2)에서 ω를 구하기 위해서 문제 (p^α)의 맨 마지막 制約式과 目的函數를 利用한 배낭. 문제의 最適解를 유도하는 λ^*를 구하여 문제 (p^α)의 制約式에서

$$F(\alpha) = \sum_{i=1}^{m} b_i' \geq \mathbf{A}_i^\beta \mathbf{X} \quad (b_i' - \mathbf{A}_i^\beta \mathbf{X})$$

$$IF(\alpha) = \sum_{i=1}^{m} b_i < \mathbf{A}_i^\beta \mathbf{X} \quad (\mathbf{A}i^\beta \mathbf{X} - b$$

를 구한다.

$$\text{여기서}\ \mathbf{X} = \begin{cases} \mathbf{1} : \lambda_j > \lambda^* \\ \mathbf{0} : \lambda_j < \lambda^* \\ (b_m - \sum_{\lambda_j > \lambda^*} a_j)/\lambda^* : \lambda_j = \end{cases}$$

$\omega_2/\omega_1 = F(\alpha)/IF(\alpha)$라 두고 ω_1을 1로 고정시키면 ω_2와 ω_1의 비율을 구할 수 있으며 위의 定理 1을 이용 하여 문제 (p^α)의 式 (3,1)에 ω_2를 곱하여 새로운 knapsack 形態로 制約式을 변형한다. 이와같이 변형된 문제의 最適解는 分枝水準 α에서의 sub-problem의 最適解보다 그 값이 크지만 LP를 풀어서 限界를 구하는 경우보다 쉽게 限界를 구할 수 있다는 장점이 있다.

분지수준 α에서의 sub-problem 의 目的函數의 값을 Z^α라 하고 incumbent solution의 目的函數 값을 Z^*라 하면 $Z^\alpha + \mathbf{A}^\alpha \mathbf{X}^\alpha > Z^*$를 만족시키는 分枝點

에서 backward, forward 段階로 더 나은 可能解를 探索해 나아간다.

(4) implicit enumeration 기준

MPKP의 探索段階 α에서의 sub-problem은 다음과 같다.

$$\max Z = \mathbf{C}^{\alpha}\mathbf{X}^{\alpha} + \mathbf{C}^{\beta}\mathbf{X}^{\beta}$$

$$\text{sub.to } \mathbf{A}^{\beta}\mathbf{X}^{\beta} \leq \mathbf{b}'$$

$$\mathbf{X}^{\beta} = \mathbf{0} \text{ or } \mathbf{1}\ \mathbf{X}^{\alpha} = \mathbf{1}$$

$$\mathbf{b}' = \mathbf{b} - \mathbf{A}^{\alpha}\mathbf{X}^{\alpha}$$

① 非可能性 檢査

위의 sub-problem에서 餘裕變數의 값을 π라 하면 $\pi_i = b_i' - \mathbf{A}_i^{\beta}\mathbf{X}^{\beta}$ 이다.

MPKP에서는 b_i, A_{ij}가 모두 양의 계수이므로 $\pi_i < 0\ (i \in M)$되는 기간 i의 과제선택 $(\mathbf{X}^{\alpha}, \mathbf{X}^{\beta})$는 非可能解이다.

② Ceiling Test

探索 段階 α에서 目的函數의 값을 Z^{α}, 1 또는 0으로 배정된 과제의 集合을 CS^{α}라 하고 D_{α}를 다음과 같이 定義하자.

$$D_{\alpha} = \{ j \mid j \in (N - CS^{\alpha}), Cj \geq Z^{*} - Z^{\alpha} \}$$

探索段階 α에서 目的函數의 값을 增進시킬 수 있는 基底 變數의 선택을 하려면 D_{α}의 集合中에서 과제를 選定하여야 한다.

③ 限界 檢査

(3)에서 언급한 方法을 利用하여 sub-problem이 만들어지는 探索水準 α에서의 探索여부를 決定한다.

(5) 技法의 흐름

MPKP의 技法은 아래와 같은 흐름을 갖고 있다. 여기에 나타나는 용어에 대하여 說明하면,

- NS : 選擇可能한 과제의 集合
- J_s : 選擇된 과제들의 集合
- NS_k : Backtracking 段階 k에서의 選擇可能한 과제들의 集合
- ZBAR : Backtracking 段階 k에서의 限界檢査로 얻어진 값
- Jpq : 기간 p의 q번째 과제로 分枝變數로 選擇된 과제

段階 1 初期解 段階

$$N_1 = \{ j \mid \lambda_j > \lambda^{*} \}$$

$$N_2 = \{ j \mid bm - S_1(\lambda) \geq \sum_{j \in N} a_j \}$$

$$Js = N_1 \cup N_2$$

$$\pi_i = b_i - \sum_{j \in Js} a_{ij} \quad i \in M$$

$$x_{ij} = \begin{cases} 1 & j \in J_s \\ 0 & j \in N - J_s \end{cases}$$

$$Z^{*} = 0 \quad Zs = \sum_{j \in Js} c_{ij}$$

段階 2 選擇可能한 과제를 NS에 포함시킨다.

$$NS = \{ P_{ij} \,/\, P_{ij} \in N - Js,\ a_{ij} \leq \pi_i,\ i \in M \}$$

① $NS = \phi$이면 단계 5로

② $NS \neq \phi$이면 단계 3으로

段階 3 NS에 포함된 과제중 Pqr를 選擇한다.

$$q = \max_{i \in M} \pi_i \quad (\pi_i > 0)$$

$$Pqr = \max_{r \in I(1), I(2) \cdots I(q)} \{ c_{qj} / a_{qj} \}$$

段階 4 pivot 段階

$$Js = Js + \{ Jpq \}$$

$$X_{ij} = \begin{cases} 0 & j \in N - Js \\ 1 & j \in Js \end{cases}$$

$$\pi_i = b_i - \sum_{j \in Js} a_{ij} \quad i \in M$$

$$Zs = \sum_{j \in Js} c_{ij}$$

단계 2로

段階 5 새로운 解를 구하는 段階

① $Zs > Z^{*}$

Set $Z^{*} = Zs$

$$X_{ij} = \begin{cases} 1 & j \in Js \\ 0 & j \in N - Js \end{cases}$$

$\pi_i = b_i - \sum_{j \in Js} a_{ij}$ 단계 6으로

② $Zs \leq Z^*$ 단계 6으로

段階 6 Backtracking 段階(a)

Backtracking 段階 k에서 NS_k에 選擇 可能한 과제를 포함시킨다.

$NS_k = \{P_{ij} \mid P_{ij} \in N - Js, a_{ij} \leq \pi_i, i \in M\} - \{P_{ij} \mid C_{ij} \leq Z^* - Zk\}$ $(k < s)$

① $NS_k = \phi$ 단계 8로

② $NS_k \neq \phi$ 단계 7로

段階 7 限界檢査 段階

① $ZBAR > Z^* - Zk$ 단계 2로

② $ZBAR \leq Z^* - Zk$ 단계 8로

段階 8 Backtracking 段階(b)

Backtracking 段階에서 $K_\alpha < K$ $(Jk_\alpha \subset Jk)$를 구하여,

① K_α가 存在하면 단계 6으로

② K_α가 없으면 stop

4. 技法의 適用

適用 對象 문제의 크기는 기간수와 變數의 수는 각각 10 × 20, 10 × 30, 10 × 50, 10 × 100으로 하고 난수를 이용하여 目的函數와 制約式의 계수를 만들었다.

使用컴퓨터의 기종은 MV-8000 이며 난수는 서브루틴인 GGUW를 이용하였다.

각 계수의 분포는 一樣(uniform) 분포를 따르며 a_{ij}, c_{ij}의 범위는 10과 100사이의 整數로 b_i는 $\sum_{j \in N} a_{ij} / 2$로 하였다.

각 문제의 크기에 따른 平均 CPU 時間은 아래와 같다.

문제의 크기	문제의 수	평균CPU시간
10 × 20	10	5.14
10 × 30	10	34.6
10 × 50	3	210.
10 × 100	3	> 780

참 고 문 헌

1. Balas, E., "An additive algorithm for solving linear program with zero-one variables", Operations Research 13 (1965).
2. Balas, E. & Zemel, E., "An algorithm for solving linear program with zero-one knapsack-problem", Operations Research 28 (1980).
3. Faaland, B.H., "The multi-period knapsack problem", Operations Research 27 (1980).
4. Fisher, M.L., "The Lagrangian relaxation method for solving integer programming problem", Management Science 27 (1981).
5. Gioffrion, A., "Integer programming by implicit enumeration and Balas 's method", SIAM 9 (1967).
6. Glover, F., "A multiphase dual algorithm for the zero-one integer programming problem", Operations Research 13 (1965).
7. Hung and Fish, "An algorithm for 0-1 multi-phase knapsacl

problem", Naval Research Logistics Quarterly 25 (1978).

8. Nauss, R.M., "An efficient algorithm for the 0-1 knapsack problem", Management Science 23 (1976).
9. Roodman, G.M., "Post optimality analysis in zero-one programming by implicit enumeration", Naval Research Logistics Quarterly 19 (1972).
10. Salkin, H.M., Integer programming, Addison-Wesley, 1975.
11. Weigartner, H.M. and Ness, D.N., "Methods for the solution of the multi-dimensional o/1 knapsack problem", Operations Research 15 (1967).