WHAT IS THE EXACTLY EQUIVALENT KNAPSACK TO AN ASSIGNMENT PROBLEM ?

By Dr. J - C. PANAYIOTOPOULOS

Department of Computers, Piraeus Graduate School of Industrial Studies

ABSTRACT. This paper gives the exactly equivalent 0 - 1 Knapsack to an Assignment problem of oıder n; consequently all the Assignment problems can be solved rapidly using the equivalent Knapsack and a computer.

1. INTRODUCTION

The Assignment problem is the special type of linear programming problem where the resourses are being allocated to the activities on a one-to-one basis [1]. To formulate this problem in mathematical programming terms, define the activity variables as :

$$x_{ij} = 1, \text{ if } i \text{ is performed by } j$$

$$= 0, \text{ otherwise}$$

$$\text{for } i = 1, 2, \ldots, n \text{ and } j = 1, 2, \ldots, n$$

then the optimization model is :

$$(1) \qquad \text{optimize } \sum_i \sum_j c_{ij} \cdot x_{ij}$$

supject to,

$$(2) \qquad \sum_j x_{ij} = 1 \qquad \text{(all i)}$$

$$(3) \qquad \sum_i x_{ij} = 1 \qquad \text{(all j)}$$

$$(4) \qquad x_{ij} \in \{0, 1\} \qquad \text{(all i and j)}$$

where the c_{ij} is a cost (or profit) for assignee i to assignment j.

The 0 - 1 Knapsack is a pure integer program, the model of which is as follows,

(5) optimize $\sum_t c_t \cdot y_t \qquad t = 1, 2, \ldots, T$

supject to,

(6) $\sum_t a_t \cdot y_t = \mathbf{b}$

(7) $y_t \in \{0, 1\}$ for all t

where c_t, a_t in this paper are all positive integers [2].

It is Known that the 0 - 1 Knapsack can be solved by the Branch - and - Bound algorithm easily [3] ; but the Assignment proplem for large n is difficult to be solved. Consequently the following question is arising :

> «How can the model model (1), (2), (3) and (4) be transformed into an equivalent Knapsack (5), (6) and (7) and what is the exact values of c_t, a_t,b and T ?».

This paper deals with the above question and gives a complete answer.

2. TRANSFORMING THE ASSIGNMENT PROBLEM INTO AN EQUIVALENT INTEGER PROGRAM

We write the (2) and (3) as follows :

$$
\begin{array}{ll}
1 & x_{11} + \cdots + x_{1n} = 1 \\
\cdot & \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot \\
n & x_{n1} + \cdots + x_{nn} = 1 \\
n+1 & x_{11} + \cdots + x_{n1} = 1 \\
\cdot & \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot \\
2n & x_{1n} + \cdots + x_{nn} = 1
\end{array}
$$

So, we get 2n conditions. We correspond the variables x_{ij} to the variables y_λ, $\lambda = 1, 2, \ldots, n^2$ where,

(8) $\lambda = (i - 1)\, n + j$

We define the martix $M = (m_{\theta\lambda})$, $\vartheta = 1, 2, \ldots, 2n \quad \lambda = 1,2, \ldots, n^2$ so that :

$$m_{\theta\lambda} = 1, \text{ if } y_\lambda \text{ occurs into condition } \vartheta$$
$$= 0, \text{ otherwise.}$$

Consequently in [3] we got the following :

L e m m a 2.1. We have $m_{\vartheta\lambda} = 1$ iff one of the following cases holds :

(i) $\vartheta = 1, 2, \ldots, n$ and $\lambda = (\vartheta - 1)\, n + \lambda_1$, $\lambda_1 = 1, 2, \ldots, n$.

(ii) $\vartheta = n+u$, $u = 1, 2, \ldots, n$ and $\lambda = \lambda_2 n + u$, $\lambda_2 = 0, 1, 2, \ldots, n-1$.

We see, that on the one hand the matrix M does not depend on the form of the conditions (2) and (3), and on the other hand the construction of M is possible in the use of computer. So, we get the equivalent integer program,

$$(9) \qquad \text{optimize} \sum_{\lambda} c_\lambda \cdot y_\lambda, \quad \lambda = 1, 2, \ldots, N$$

subject to,

$$(10) \qquad \sum_{\lambda} m_{\vartheta\lambda} \cdot y_\lambda = 1 \quad (\text{all } \vartheta = 1, 2, \ldots, M)$$

$$(11) \qquad y_\lambda \in \{ 0, 1 \} \text{ for all } \lambda$$

where $N = n^2$ and $c_\lambda = c_{ij}$, $\lambda = (i - 1)\, n + j$.

C o r r o l a r y 2.2. For every $\vartheta = 1, 2, \ldots, 2n$ we get :

$$\sum_{\lambda} m_{\vartheta\lambda} = n, \quad \lambda = 1, 2, \ldots, n^2.$$

3. CONVERTING THE (9), (10), (11) PROGRAM

Consider the integer programming model (9), (10) and (11). Such a model can be transformed into an equivalent problem where the M constraints in (10) are replaced by a single constraint. For easy of exposition we illustrate how to aggregate two constraints [4] :

$$(12) \qquad \sum_{t=1}^{T} S_t \cdot x_t = b_1 \quad \text{and} \quad \sum_{t=1}^{T} R_t \cdot x_t = b_2$$

where $x_t \in \{ 0, 1, 2, \ldots, U_t \}$ and all S_t, R_t, b_1, b_2 are integer - valued. Let,

$$m_1 = \sum_{t} (\max \{ 0, S_t \}) \cdot U_t - b_1$$

$$m_2 = \sum_{t} (\min \{ 0, S_t \}) \cdot U_t - b_1$$

$$m = \max \{ m_1, |m_2| \}.$$

Then we can replace (12) by :

$$(13) \qquad \sum_{t} (S_t + M \cdot R_t) \cdot x_t = b_1 + M \cdot b_2 = B$$

where M is any integer such that $|M| > m$.

Assume the two first constraints of (10) :

$$(14) \qquad \sum_{j=1}^{N} m_{2j} \cdot y_j = 1 \qquad \text{and} \qquad \sum_{j=1}^{N} m_{1j} \cdot y_j = 1.$$

Then, because of (13) and 2.2 we get,

$$m_1 \quad \sum_{j=1}^{N} (\max \{ 0, m_{2j} \}) \cdot U_j - 1 = n - 1$$

$$m_2 = 0 - 1 = -1$$

$$m = n - 1 \quad ; \quad \text{so, } M = n$$

$$\sum_{j=1}^{N} (m_{2j} + m_{1j} \cdot n) \cdot y_j = 1 + n \cdot 1 = B_2, \ (\vartheta = 2).$$

We set $B_1 = 1$ and $L_j^1 = m_{1j}$. Thus we have a new constraint instead of (14) ; let :

$$(15) \qquad \sum_{j=1}^{N} L_j^2 \cdot y_j = B_2$$

We write the $\vartheta = 3$ constraint of (10) and (15) :

$$(16) \qquad \sum_{j=1}^{N} m_{3j} \cdot y_j = 1 \quad \text{and} \quad \sum_{j=1}^{N} L_j^2 \cdot y_j = B_2.$$

Then, because of (13) and 2.2 we get $M = n$ again, and

$$(17) \qquad \sum_{j=1}^{N} (m_{3j} + L_j^2 \cdot n) \cdot y_j = 1 + n \cdot B_2 = B_3$$

If we continue the above technique, at the end ($\vartheta = M$) we get the final single constraint :

$$\sum_{j=1}^{N} (m_{Mj} + L_j^{M-1} \cdot n) \, y_j = 1 + n \cdot B_{M-1} = B_M$$

L e m m a 3.1. The constraints (10) are equivalent to the single constraint ($\vartheta = M$) :

$$\sum_{j=1}^{N} (m_{\theta j} + L_j^{\theta-1} \cdot n) \cdot y_j = 1 + n \cdot B_{\theta-1} = B_\theta$$

where $\vartheta = 2, 3, \ldots, M$, $B_1 = 1$, $L_j^1 = m_{1j}$, $\forall j = 1, 2, \ldots, N$ and,

$$\sum_{j=1}^{N} L_j^{\theta-1} \cdot y_j = B_{\theta-1} \quad .$$

Theorem 3.2. The Assignment problem (1), (2), (3) and (4) is equivalent to the following 0—1 Knapsack problem :

(18) optimize $\sum_{\lambda=1}^{n^2} c_\lambda \cdot y_\lambda$, $\lambda = (i-1)\,n + j$, $i, j = 1,2,\ldots, n$

subject to :

(19) $$\sum_{\mu=1}^{n} \sum_{p=1}^{n} (n^{2n-\mu} + n^{n-p}) \cdot y_{(\mu-1)n+p} = \sum_{\varphi=0}^{2n-1} n^\varphi$$

(20) $y_\lambda \in \{0, 1\}$, $\forall \lambda = 1, 2, \ldots, n^2$.

Proof. From 3.1 we get,

$$B_M = 1 + n \cdot B_{M-1} = 1+n+n^2 \cdot B_{M-2} = \ldots = 1+n+n^2+\ldots+n^{M-1} \cdot B_1 =$$

$$= n^0 + n + n^2 + \ldots n^{2n-1} \cdot 1 = \sum_{\varphi=0}^{2n-1} n^\varphi$$

So, we get the right - hand side of (19). Assume that d_θ, $\vartheta = 1, 2, \ldots, 2n$ are the respectively coefficients of y_{j0} in 3.1; then there exist w and w_0 such that, $w \neq w_0 \in \{1, 2, \ldots, 2n\}$ and $m_{wj_0} = m_{w_0j_0} = 1$, $m_{\theta j_0} = 0$, $\forall \vartheta \neq w, w_0$

Consequently we have :

$$d_2 = m_{2j_0} + L^1_{j_0} \cdot n = 0 + 0 \cdot n = 0$$

. . . .

$$d_w = 1 + 0 \cdot n = 1$$

$$d_{w+1} = 0 + 1 \cdot n = n$$

. . . .

$$d_n = n^{n-w}$$

$$d_{n+1} = n^{n-w+1}$$

. . . .

$$d_{w_0} = 1 + n^{n-w+w_0-n} = 1 + n^{w_0-w} = 1 + n^n$$

. . . .

$$d_{2n} = n^{2n-w_0} + n^{2n-w_0+n} = n^{2n-w_0} + n^{3n-w_0}$$

We set $p = \mu = w_0 - n$ into the left - hand side of (19) :

$$n^{2n-w_0+n} + n^{n-w_0+n} = n^{3n-w_0} + n^{2n-w_0} = d_{2n}.$$

Therefore the proof of 3.2 is complete.

4. THE ALGORITHM

We give an algorithm to solve the Assignment problem (1), (2), (3) and (4) according to 3.2, the steps of which are ordered as follows :

STEP 0. The n and the martix $A = (c_{ij})$, $i, j = 1, 2, \ldots, n$ are given.

STEP 1. We define the matrices :

$$Y = (y_\lambda) \quad , \quad C = (c_\lambda) = 0$$

where $\lambda = (i - 1)\, n + j \quad , \quad \lambda = 1, 2, \ldots, n^2$

STEP 2. We set :

$$c_\lambda = c_{ij} \text{ where } \lambda = (i - 1)\, n + j, \quad \forall\, i, j$$

STEP 3. We solve the Knapsack :

$$\text{optimize } C \cdot Y$$

subject to,

$$\sum_i \sum_j (n^{2n-i} + n^{n-j}) \cdot y_\lambda = \sum_{\varphi=o}^{2n-1} n^\varphi$$

$$y_\lambda \in \{0, 1\} \text{ for all } \lambda.$$

STEP 4. If $y_\lambda = 1$ then the respective variable $x_{ij} = 1$, where $\lambda = (i - 1)\, n + j$; otherwise $x_{ij} = 0$. END.

An available computer was used and the program of the above algorithm was coded in machine language ; so, in case n = 1500 we got the optimal solution after 7 minutes machine time work. But, when we tryed to solve the problem without the above algorithm, then we got the optimal solution after 12 minutes.

I l l u s t r a t i o n 4.1. An easy example will clarify the details of the procedure. Consider :

$$\text{maximize } (3x_{11} + 4x_{12} + 5x_{13} + 3x_{14} + 5x_{21} + 4x_{22} + 3x_{23} + 2x_{24} + 6x_{31} + 7x_{32} + 8x_{33} + x_{34} + x_{41} + 3x_{42} + 2x_{43} + 5x_{44})$$

subject to,

$$\sum_j x_{ij} = 1 \quad \text{(all i)} \quad \text{and} \quad \sum_i x_{ij} = 1 \quad \text{(all j)}$$

$$x_{ij} \in \{0, 1\}, \ \forall\, i, j = 1, 2, 3, 4.$$

STEP 0. The $n = 4$ and the martix :

$$A = \begin{bmatrix} 3 & 4 & 5 & 3 \\ 5 & 4 & 3 & 2 \\ 6 & 7 & 8 & 1 \\ 1 & 3 & 2 & 5 \end{bmatrix}$$

are given.

STEP 1. We define the matrices :

$$Y = (y_\lambda) \ , \ \lambda = 1, 2, \ldots, 16$$

$$C = (c_\lambda) = 0 \ , \ \lambda = (i - 1) \cdot 4 + j \ , \ \forall \ i,j = 1, 2, 3, 4.$$

STEP 2. We get :

$$C = [3, 4, 5, 3, 5, 4, 3, 2, 6, 7, 8, 1, 1, 3, 2, 5]$$

STEP 3. We solve the Knapsack :

$$\text{maximize } C \cdot Y$$

subject to,

$$16448\ y_1 + 16400\ y_2 + 16388\ y_3 + 16385\ y_4 + 4160\ y_5 + 4112\ y_6 + 4100\ y_7 + 4097\ y_8 + 1088\ y_9 + 1040\ y_{10} + 1028\ y_{11} + 1025\ y_{12} + 320\ y_{13} + 272\ y_{14} + 260\ y_{15} + 257\ y_{16} = 21845.$$

$$y_\lambda = 0 \text{ or } 1.$$

STEP 4. Optimal solution: $y_2 = y_5 = y_{11} = y_{16} = 1$; consequently we get :

$$x_{12} = x_{21} = x_{33} = x_{44} = 1.$$

where the maximum value is 22. END.

REFERENCES

[1] Wagner H. M., Principles of Operations Research P.H.I. London, 1975.

[2] Greenberg H. and Hegerich R.L., «A Branch Search Algorithm for the Knapsack Problem», Manag. Science, vol. 16, 1970, pp. 327 - 332.

[3] Panayiotopoulos J-C., «About tke Maximum Production of a Company», Hellenic Math. Review, vol. 6, 1977, pp. 140 - 152.

[4] Padberg M. W. «Equivalent Knapsack - Type Formulation of Bounded Integer Linear Programs : An Alternative Approach». Naval Research Logistics Quart, vol. 19, 1972, pp. 699 - 708.