Mathematical Problems in Engineering
Volume 7, pp. 155–175
Reprints available directly from the publisher
Photocopying permitted by license only

An Optimal Control Approach to Manpower Planning Problem*

H. W. J. LEE [a,†], X. Q. CAI [b] and K. L. TEO [a]

[a]*Department of Applied Mathematics, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong;* [b]*Department of Systems Engineering and Engineering Management, The Chinese University of Hong Kong, Shatin, NT, Hong Kong*

(Received 2 August 2000; In final form 23 October 2000)

A manpower planning problem is studied in this paper. The model includes scheduling different types of workers over different tasks, employing and terminating different types of workers, and assigning different types of workers to various trainning programmes. The aim is to find an optimal way to do all these while keeping the time-varying demand for minimum number of workers working on each different tasks satisfied. The problem is posed as an optimal discrete-valued control problem in discrete time. A novel numerical scheme is proposed to solve the problem, and an illustrative example is provided.

Keywords: Optimal control; Manpower planning; CPET; DMISER

1. INTRODUCTION

This paper considers a manpower planning problem. Suppose there are n different types of workers and there are m different kinds of work (tasks) to perform. Each type of workers must be able to perform at least one specific task. The wages, the costs to employ, and the costs to terminate employment are different for each type of workers. Suppose type i workers can perform tasks $\{i_1, i_2, \ldots, i_p\}$ and suppose type j

*Acknowledgment to the Research Committee of The Hong Kong Polytechnic University

†Corresponding author. Tel.: (852) 2766-6951, Fax: (852) 2362-9045, e-mail: majlee@polyu.edu.hk

workers can perform not only tasks $\{i_1, i_2, \ldots, i_p\}$ but more, say, tasks $\{i_1, i_2, \ldots, i_p, j_1, j_2, \ldots, j_q\}$, then, the wages, the costs to employ, the costs to terminate employment for type j workers would be higher than those of type i. Some organizations would assign workers to undergo various trainning programmes. Thus, workers of the "lower wages" type can be "upgraded" to a "better wages" type with the additional skills gained from the trainning programme. Workers entered into the trainning programme, although not engaging in the workforce, would still be receiving their old wages until they have completed the trainning programme. After that, they would be receiving the "better wages", as they are upgraded to a "better wages" type.

Different tasks have different time-varying manpower demand requirements to satisfy. That is, for every task, there is a minimum number of workers required to work on that task at each specific time. The objective is thus to find the optimal combined policy of employment, scheduling, and trainning over the planning horizon such that the cost is minimized subject to these specifications and requirements.

In [7, 5], the problem of finding the optimal policy of the training of manpower is presented. However, the model involved only considers two types of workers, namely the "skilled" and the "unskilled" labour. Moreover, the employment policy, the scheduling of workers, and the manpower demand constraints are not included in the model.

Staff scheduling has attracted considerable attention in the literature. Nevertheless, the majority of the literature considers mainly the requirement to allocate available manpower to meet the demands for staff. In general, this problem is formulated as a static, integer optimization model. An early survey is given in [1]. [2] gives another survey on the works published in recent years. [6] considers full-time and part-time employees, but only a single type of jobs. [11] considers a heterogeneous workforce, and propose a heuristic to tackle the problem. [12] investigates a single shift scheduling problem, which contains multiple categories of workers whose capabilities can be structured hierarchically. Genetic algorithms and simulated annealing have also been applied to tackle the staff scheduling problem; see, for example [3, 4, 13, 16].

In this paper, we aim to provide a more complete mathematical model of this problem. It turns out naturally as a discrete time optimal discrete-valued control problem. The control transformation technique proposed in [10] is used to convert the problem into a sequence of standard discrete-time optimal control problems. Each of these discrete-time optimal control problems can be solved by the discrete-time optimal control technique presented in [14], and hence the companion software package DMISER3.2 [8] is applicable.

A numerical example is also provided for illustration.

2. PROBLEM FORMULATION

Consider the discrete time setting where the planning horizon is over $t = 0, 1, 2, \ldots, T$. Let

$$z(t) = \begin{bmatrix} z_1(t) \\ z_2(t) \\ \vdots \\ z_n(t) \end{bmatrix}$$

where $z_j(t)$ denotes the number of type j workers engaging in the workforce at time t. Let

$$\boldsymbol{\alpha}(t) = \begin{bmatrix} \alpha_1(t) \\ \alpha_2(t) \\ \vdots \\ \alpha_n(t) \end{bmatrix}, \quad \boldsymbol{\beta}(t) = \begin{bmatrix} \beta_1(t) \\ \beta_2(t) \\ \vdots \\ \beta_n(t) \end{bmatrix}$$

where $\alpha_j(t)$ denotes the number of type j workers newly employed at time t, and $\beta_j(t)$ denotes the number of type j workers being terminated at time $t \in \{0, 1, 2, \ldots, T-1\}$. We assume that for all $t = 0, 1, 2, \ldots, T-1$, and for all $j = 1, 2, \ldots, n$,

$$\alpha_j(t) \in \{0, 1, 2, \ldots, \bar{\alpha}\} \tag{2.1}$$

$$\beta_j(t) \in \{0, 1, 2, \ldots, \bar{\beta}\} \tag{2.2}$$

Let $w_{ij}(t)$ be the number of type i workers assigned to perform task j at time t, and for all $i = 1, 2, \ldots, n$, and $j = 1, 2, \ldots, m$, and $t = 0, 1, 2, \ldots, T$,

$$w_{ij}(t) \in \{0, 1, 2, \ldots, \overline{w_{ij}}\} \tag{2.3}$$

Note that if a type i worker is not able to work on task k, then $w_{ik}(t) = 0$, $\forall t \in \{0, 1, 2, \ldots, T\}$. Let $\mathcal{I}$ be the index set containing all (i,j) pairs such that $w_{ij}(t)$ are not those constrained as zero for all $t = 0, 1, 2, \ldots, T$. Moreover, we define $\Omega : \mathcal{I} \mapsto \{1, 2, \ldots, \|\mathcal{I}\|\}$ as a one-to-one map. We now define $\boldsymbol{w}(t)$ as a vector collecting all $w_{ij}(t)$ such that $[\boldsymbol{w}(t)]_{\Omega(i,j)} = w_{ij}(t)$.

Since different tasks have different time-varying manpower demand requirements to satisfy, we have, for all $t = 0, 1, 2, \ldots, T$, the following all-time-step constraints

$$\sum_{i=1}^{n} w_{ij}(t) \geq d_j(t), \quad j = 1, 2, \ldots, m, \tag{2.4}$$

$$\sum_{j=1}^{m} w_{ij}(t) = z_i(t), \quad i = 1, 2, \ldots, n. \tag{2.5}$$

where $d_j(t)$, for $t = 0, 1, 2, \ldots, T$, is the time-varying manpower demand requirements for task j. That is to say, at time t, there should be at least $d_j(t)$ workers assigned to perform task j. We now define

$$\boldsymbol{\eta} = \begin{bmatrix} \eta_1 \\ \eta_2 \\ \vdots \\ \eta_n \end{bmatrix}, \quad \boldsymbol{\sigma} = \begin{bmatrix} \sigma_1 \\ \sigma_2 \\ \vdots \\ \sigma_n \end{bmatrix}, \quad \boldsymbol{r} = \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_n \end{bmatrix}$$

where η_i, σ_i, and r_i are, respectively, the cost to employ a new type i worker, the cost to terminate a type i worker, and the wage of a type i worker during employment.

Remark 2.1 In reality, we note that $\boldsymbol{\eta}$, $\boldsymbol{\sigma}$, and $\boldsymbol{r}$ should be time dependent. However, for short term planning, we may assume that these quantities are constants. This assumption is made so as to simplify the complexity of the problem concerned. Our formulation may be considered as an approximation to the actual situation. Further research is needed for a more general situation.

In general, not all types of workers can be assigned to trainning programmes for "upgrading". Among the n types of workers, suppose there are only n' (where $n' \leq n$) types of workers that can be assigned to leave the workforce to undergo a trainning programme to be "upgraded". Let $\mu_{a_j}(t)$ be the number of type j workers assigned to leave the workforce to undergo its trainning programme at time t, where $a_j \in \{1, 2, \ldots, n'\}$ is the corresponding index. Note that a_j may not be defined for all $j \in \{1, 2, \ldots, n\}$. If a_j is defined for some $j \in \{1, 2, \ldots, n\}$, it means type j workers can be assigned to a trainning programme for "upgrading", otherwise, there is no such trainning programme for type j workers. Without loss of generality, we assume that, if a_{j_1} and a_{j_2} are both defined, with $j_1 < j_2$, and a_j is not defined for all $j \in \{j_1+1, j_1+2, \ldots, j_2-1\}$, then $a_{j_1}+1 = a_{j_2}$. We can write this in a more compact form. Let

$$\boldsymbol{\mu}(t) = \begin{bmatrix} \mu_1(t) \\ \mu_2(t) \\ \vdots \\ \mu_{n'}(t) \end{bmatrix},$$

where

$$\mu_i(t) \in \{0, 1, 2, \ldots, \overline{\mu_i}\} \tag{2.6}$$

for all $t \in \{0, 1, 2, \ldots, T-1\}$. Associated with $\boldsymbol{\mu}$ is its cost vector $\boldsymbol{\zeta} = \begin{bmatrix} \zeta_1 \\ \zeta_2 \\ \vdots \\ \zeta_{n'} \end{bmatrix}$. Once workers leave the workforce for trainning, it would take them a certain number of time steps before they can rejoin the workforce with their "upgraded" skills. Each trainning programme may have a different length of duration. Suppose for some j, a_j is defined, and the trainning programme takes $\tau_{a_j}+1$ time steps to complete, we then define

$$y_1^{a_j}(t), y_2^{a_j}(t), \ldots, y_{\tau_a - 1}^{a_j}(t)$$

as the trainning states for type j workers.

The dynamics of the trainning states for type j workers can be written as:

$$\begin{aligned} y_1^{a_j}(t+1) &= \mu_{a_j}(t) \\ y_2^{a_j}(t+1) &= y_1^{a_j}(t) \\ y_3^{a_j}(t+1) &= y_2^{a_j}(t) \\ &\vdots \quad \vdots \\ y_{\tau_{a_j}+1}^{a_j}(t+1) &= y_{\tau_{a_j}}^{a_j}(t), \end{aligned}$$

or equivalently,

$$\begin{bmatrix} y_1^{a_j}(t+1) \\ y_2^{a_j}(t+1) \\ y_3^{a_j}(t+1) \\ \vdots \\ y_{\tau_{a_j}+1}^{a_j}(t+1) \end{bmatrix} = \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & & \ddots & & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{bmatrix} \begin{bmatrix} y_1^{a_j}(t) \\ y_2^{a_j}(t) \\ y_3^{a_j}(t) \\ \vdots \\ y_{\tau_{a_j}+1}^{a_j}(t) \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ \vdots \\ \vdots \\ 0 \end{bmatrix} \mu_{a_j}(t). \tag{2.7}$$

To simplify the notations, if a_j is defined for some $j \in \{1, 2, \ldots, n\}$, we let

$$\boldsymbol{y}^{a_j}(t) = \begin{bmatrix} y_1^{a_j}(t) \\ y_2^{a_j}(t) \\ y_3^{a_j}(t) \\ \vdots \\ y_{\tau_{a_j}+1}^{a_j}(t) \end{bmatrix},$$

and define $\boldsymbol{C}_{a_j}$ as a $(\tau_{a_j}+1) \times (\tau_{a_j}+1)$ matrix:

$$[\boldsymbol{C}_{a_j}]_{i,k} = \begin{cases} 1, & i = k+1 \\ 0, & \text{otherwise}, \end{cases}$$

and $\boldsymbol{D}_{a_j}$ as a $(\tau_{a_j}+1) \times 1$ vector:

$$[\boldsymbol{D}_{a_j}]_i = \begin{cases} 1, & i = 1 \\ 0, & \text{otherwise}. \end{cases}$$

Hence, (2.7) can be rewritten as:

$$\boldsymbol{y}^{a_j}(t+1) = \boldsymbol{C}_{a_j}\boldsymbol{y}^{a_j}(t) + \boldsymbol{D}_{a_j}\mu_{a_j}(t).$$

Let $\boldsymbol{y}(t)$ be a vector collecting all $\boldsymbol{y}^k(t)$ for all $k = 1, 2, \ldots, n'$,

$$\boldsymbol{y}(t) = [\boldsymbol{y}^1(t)^\top, \boldsymbol{y}^2(t)^\top, \ldots, \boldsymbol{y}^{n'}(t)^\top]^\top.$$

Note that $\boldsymbol{y}(t) \in \mathbb{R}^{n''}$ where $n'' = \sum_{p=1}^{n'}(\tau_p + 1)$.

Hence, the dynamics of the whole of $\boldsymbol{y}(t)$ can be written as:

$$\boldsymbol{y}(t+1) = \boldsymbol{C}\boldsymbol{y}(t) + \boldsymbol{D}\boldsymbol{\mu}(t)$$

with initial condition

$$\boldsymbol{y}(0) = \boldsymbol{y}_0,$$

where

$$\boldsymbol{C} = \begin{bmatrix} \boldsymbol{C}_1 & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{C}_2 & \cdots & \boldsymbol{0} \\ \vdots & & \ddots & \vdots \\ \boldsymbol{0} & \cdots & \boldsymbol{0} & \boldsymbol{C}_{n'} \end{bmatrix}_{n'' \times n''}, \text{and}$$

$$\boldsymbol{D} = \begin{bmatrix} \boldsymbol{D}_1 & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{D}_2 & \cdots & \boldsymbol{0} \\ \vdots & & \ddots & \vdots \\ \boldsymbol{0} & \cdots & \boldsymbol{0} & \boldsymbol{D}_{n'} \end{bmatrix}_{n'' \times n'}$$

The overall dynamics of the model is thus given by

$$\begin{bmatrix} \boldsymbol{z}(t+1) \\ \boldsymbol{y}(t+1) \end{bmatrix} = \begin{bmatrix} \boldsymbol{z}(t) + \boldsymbol{\alpha}(t) - \boldsymbol{\beta}(t) - \boldsymbol{A}\boldsymbol{\mu}(t) + \boldsymbol{B}\boldsymbol{y}(t) \\ \boldsymbol{C}\boldsymbol{y}(t) + \boldsymbol{D}\boldsymbol{\mu}(t) \end{bmatrix}$$

$$= \begin{bmatrix} \boldsymbol{I} & \boldsymbol{B} \\ \boldsymbol{0} & \boldsymbol{C} \end{bmatrix} \begin{bmatrix} \boldsymbol{z}(t) \\ \boldsymbol{y}(t) \end{bmatrix} + \begin{bmatrix} \boldsymbol{I} & -\boldsymbol{I} & -\boldsymbol{A} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{D} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha}(t) \\ \boldsymbol{\beta}(t) \\ \boldsymbol{\mu}(t) \\ \boldsymbol{w}(t) \end{bmatrix} \quad (2.8)$$

for $t=0,1,2,\dots,T-1$ with the initial condition

$$\begin{bmatrix} z(0) \\ y(0) \end{bmatrix} = \begin{bmatrix} z_0 \\ y_0 \end{bmatrix},$$

where $\boldsymbol{A}$ and $\boldsymbol{B}$ are $n \times n'$ and $n \times n''$ matrices defined by

$$[A]_{j,k} = \begin{cases} 1, & \text{if } a_j \text{ is defined and } a_j = k, \\ 0, & \text{otherwsie,} \end{cases}$$

and

$$[\boldsymbol{B}]_{j,k} = \begin{cases} 1 & k \in b(j) \\ 0 & \text{otherwise} \end{cases}$$

respectively, $b(\cdot)$ is the index set defined as: if $b(j)=\{i_1, i_2, \dots, i_h\}$, then $y_{i_1}, y_{i_2}, \dots, y_{i_h}$ would leave the trainning dynamics and rejoin the workforce as type j workers.

We can write (2.8) in a more compact form:

$$\boldsymbol{x}(t+1) = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{B} \\ \boldsymbol{0} & \boldsymbol{C} \end{bmatrix} \boldsymbol{x}(t) + \begin{bmatrix} \boldsymbol{I} & -\boldsymbol{I} & -\boldsymbol{A} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{D} & \boldsymbol{0} \end{bmatrix} \boldsymbol{u}(t) \tag{2.9}$$

for $t=0,1,2,\dots,T-1$ with the initial condition

$$\boldsymbol{x}(0) = \boldsymbol{x}_0, \tag{2.10}$$

where

$$\boldsymbol{x}(t) = \begin{bmatrix} z(t) \\ y(t) \end{bmatrix}, \quad \boldsymbol{u}(t) = \begin{bmatrix} \boldsymbol{\alpha}(t) \\ \boldsymbol{\beta}(t) \\ \boldsymbol{\mu}(t) \\ \boldsymbol{w}(t) \end{bmatrix}. \tag{2.11}$$

Note that our decision variables $\boldsymbol{\alpha}(t)$, $\boldsymbol{\beta}(t)$, $\boldsymbol{\mu}(t)$, are for all $t=0, 1,2,\dots,T-1$, whereas $w_{ij}(t)$ for each $(i,j)\in\mathcal{I}$ are for all $t=0, 1,2,\dots,T$. Therefore, to be more precise, we define

$$\boldsymbol{u}(t) = \begin{bmatrix} \boldsymbol{\alpha}(t) \\ \boldsymbol{\beta}(t) \\ \boldsymbol{\mu}(t) \\ \boldsymbol{w}(t) \end{bmatrix} \quad \text{for } t = 0,1,2,\dots,T-1,$$

and

$$\xi = \boldsymbol{w}(T). \tag{2.12}$$

We can now rewrite constraints (2.4) and (2.5) as:

$$\gamma_j^{\top} \boldsymbol{u}(t) \geq d_j(t), \quad j = 1, 2, \ldots, m, \; t = 0, 1, 2, \ldots, T-1 \tag{2.13}$$

$$\gamma_j^{\top} \xi \geq d_j(T), \quad j = 1, 2, \ldots, m, \tag{2.14}$$

$$\boldsymbol{\rho}_i^{\top} \boldsymbol{u}(t) = [\boldsymbol{x}(t)]_i, \quad i = 1, 2, \ldots, n, \; t = 0, 1, 2, \ldots, T-1 \tag{2.15}$$

$$\boldsymbol{\rho}_i^{\top} \xi = [\boldsymbol{x}(T)]_i, \quad i = 1, 2, \ldots, n, \tag{2.16}$$

where

$$[\gamma_j]_k = \begin{cases} 0, & k \in \{1, 2, \ldots, 2n + n'\} \\ 1, & \Omega^{-1}(k - (2n + n')) = (\cdot, j) \in \mathcal{I} \\ 0, & \text{otherwise,} \end{cases}$$

and

$$[\boldsymbol{\rho}_i]_k = \begin{cases} 0, & k \in \{1, 2, \ldots, 2n + n'\} \\ 1, & \Omega^{-1}(k - (2n + n')) = (i, \cdot) \in \mathcal{I} \\ 0, & \text{otherwise,} \end{cases}$$

and we can rewrite the discrete range sets (2.1), (2.2), (2.6), (2.3) as:

$$u_k(t) \in \{0, 1, 2, \ldots, \overline{u_k}\}, \quad t = 0, 1, 2, \ldots, T-1 \tag{2.17}$$

$$\xi_k \in \{0, 1, 2, \ldots, \overline{\xi_k}\}. \tag{2.18}$$

Remark 2.2 Note that $\overline{u_k}$ and $\overline{\xi_k}$ should be state dependent, *i.e.*, depending on $\boldsymbol{x}(t)$. However, as a first approximation towards the mathematical formulation of this important problem, we assume that they are constants so as to avoid further complexity.

Since workers entering the trainning programme would still be receiving their old wages until their trainning is completed, we can then compute the cost associated with $\boldsymbol{y}(t)$ by

$$\boldsymbol{r}^{\top} H_f(\boldsymbol{A}\boldsymbol{D}^{\top})\boldsymbol{y}(t)$$

where $H_f(\boldsymbol{AD}^\top)$ is defined as replacing "0" by "1" following any "1" along the row of the matrix $\boldsymbol{AD}^\top$ until reaching just before a column which contains a "1". For example,

$$\text{if } \boldsymbol{AD}^\top = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad \text{then}$$

$$H_f(\boldsymbol{AD}^\top) = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

Let

$$\boldsymbol{c}^\top = [\boldsymbol{r}^\top, \boldsymbol{r}^\top H_f(\boldsymbol{AD}^\top)].$$

The total cost is thus given by

$$\boldsymbol{c}^\top \begin{bmatrix} z(T) \\ y(T) \end{bmatrix} + \sum_{t=0}^{T-1} \left\{ \boldsymbol{c}^\top \begin{bmatrix} z(t) \\ y(t) \end{bmatrix} + \boldsymbol{\eta}^\top \boldsymbol{\alpha}(t) + \boldsymbol{\sigma}^\top \boldsymbol{\beta}(t) + \boldsymbol{\zeta}^\top \boldsymbol{\mu}(t) \right\},$$

or more compactly,

$$\boldsymbol{c}^\top \boldsymbol{x}(T) + \sum_{t=0}^{T-1} \{ \boldsymbol{c}^\top \boldsymbol{x}(t) + \boldsymbol{e}^\top \boldsymbol{u}(t) \},$$

where

$$\boldsymbol{e}^\top = [\boldsymbol{\eta}^\top, \boldsymbol{\sigma}^\top, \boldsymbol{\zeta}^\top, \boldsymbol{0}^\top].$$

In practice, we usually take

$$\boldsymbol{e}^\top = [\boldsymbol{\eta}^\top, \boldsymbol{\sigma}^\top, \boldsymbol{\zeta}^\top, \boldsymbol{\varepsilon}^\top],$$

where $\boldsymbol{\varepsilon}^\top$ is a row of insignificantly small and not evenly distributed numbers instead of exactly zeros.

Now, we can formulate our problem as:

Subject to the set of difference Eq. (2.9) with initial condition (2.10), the constraints (2.13), (2.14), (2.15), (2.16), and

$$x_i(t) \geq 0, \quad \text{for } i = 1, 2, \ldots, n + n'', \tag{2.19}$$

find a discrete-valued control $\boldsymbol{u}(\cdot)$ with the range specified in (2.17), and a system parameter ξ with the range specified in (2.18) such that the cost

$$J(\boldsymbol{u}(\cdot), \xi) = \boldsymbol{c}^{\top}\boldsymbol{x}(T) + \sum_{t=0}^{T-1}\{\boldsymbol{c}^{\top}\boldsymbol{x}(t) + \boldsymbol{e}^{\top}\boldsymbol{u}(t)\}, \tag{2.20}$$

is minimized. Note that each component of $\boldsymbol{u}(\cdot)$ and ξ is integer-valued. Thus, this is an optimal discrete-valued control problem in discrete time. We refer this as problem **(P)**.

3. A COMPUTATIONAL PROCEDURE

Problem **(P)** is a discrete time optimal discrete-valued control problem. It cannot be handled directly by standard technique as reported in [14, 8]. A problem transformation is needed.

In [10], a novel method for solving a general class of nonlinear integer programming problems is introduced. Basically, the method transforms the problem with integer decision variables into a discrete-valued optimal control problem, and then followed by another problem transformation, called the Control Parametrization Enhancing Transform (CPET), to obtain a standard optimization problem involving only continuous values. The combinatorial nature of the original problem is thus avoided. Motivated by these results, we aim to replace each $u_k(\cdot)$ and ξ_k in Problem **(P)** by some control sequences with continuous range.

For each k, suppose $u_k(t)$ is replaced by $\{(\sum_{p=1}^{\overline{u_k}+1} p \cdot v_p^k(t)) - 1\}$ for $t = 0, 1, 2, \ldots, T-1$, and ξ_k is replaced by $\{(\sum_{p=1}^{\overline{\xi_k}+1} p \cdot \vartheta_p^k) - 1\}$, where

$$v_p^k(t) \in [0, 1] \quad \text{for all } p \in \{1, 2, \ldots, \overline{u_k} + 1\},$$
$$k \in \{1, 2, \ldots, 2n + n' + \|\mathcal{I}\|\}$$

and

$$\vartheta_p^k \in [0,1] \quad \text{for all } p \in \{1,2,\ldots,\overline{\xi_k}+1\}, k \in \{1,2,\ldots,\|\mathcal{I}\|\}$$

and we confine $v_p^k(t)$ and ϑ_p^k by

$$\sum_{p=1}^{\overline{u_k}+1} v_p^k(t) = 1 \quad \text{for } t = 0,1,2,\ldots,T-1, \quad \text{and} \sum_{p=1}^{\overline{\xi_k}+1} \vartheta_p^k = 1 \tag{3.1}$$

respectively. We then define $\boldsymbol{v}_k(s,t)$ as:

$$\boldsymbol{v}_k(s,t) = \begin{cases} v_1^k(t), & 0 \leq s < 1 \\ v_2^k(t), & 1 \leq s < 2 \\ \vdots & \vdots \\ v_{\overline{u_k}+1}^k(t), & \overline{u_k} \leq s < \overline{u_k}+1 \end{cases}$$

and $\boldsymbol{\vartheta}_k(s)$ as:

$$\boldsymbol{\vartheta}_k(s) = \begin{cases} \vartheta_{1,k}, & 0 \leq s < 1 \\ \vartheta_{2,k}, & 1 \leq s < 2 \\ \vdots & \vdots \\ \vartheta_{\overline{\xi_k}+1,k}, & \overline{\xi_k} \leq s < \overline{\xi_k}+1, \end{cases}$$

Thus, from (3.1), we have

$$\int_0^{\overline{u_k}+1} \boldsymbol{v}_k(s,t)ds = 1, \quad k \in \{1,2,\ldots,2n+n'+\|\mathcal{I}\|\},$$
$$t \in \{0,1,2,\ldots,T-1\}$$

and

$$\int_0^{\overline{\xi_k}+1} \boldsymbol{\vartheta}_k(s)ds = 1, \quad k \in \{1,2,\ldots,\|\mathcal{I}\|\}.$$

For each $k \in \{1,2,\ldots,2n+n'+\|\mathcal{I}\|\}$, and $t \in \{0,1,2,\ldots,T-1\}$ we define the variance of $\boldsymbol{v}_k(\cdot,t)$ as:

$$\operatorname{var}(\boldsymbol{v}_k(\cdot,t)) = \int_0^{\overline{u_k}+1} s^2 \boldsymbol{v}_k(s,t)ds - \left(\int_0^{\overline{u_k}+1} s\boldsymbol{v}_k(s,t)ds\right)^2,$$

and for each $k \in \{1,2,\ldots,\|\mathcal{I}\|\}$, we define the variance of $\boldsymbol{\vartheta}_k(\cdot)$ as:

$$\operatorname{var}(\boldsymbol{\vartheta}_k(\cdot)) = \int_0^{\overline{\xi_k}+1} s^2 \boldsymbol{\vartheta}_k(s)ds - \left(\int_0^{\overline{\xi_k}+1} s\vartheta_k(s)ds\right)^2.$$

In view of [10], the following two remarks are in order.

Remark 3.1 For each $t \in \{0, 1, 2, \ldots, T-1\}$, $k \in \{1, 2, \ldots, 2n + n' + \|\mathcal{I}\|\}$, the minimum value of $\mathrm{var}(\boldsymbol{v}_k(\cdot, t))$ is $(1/12)$. Moreover, $\mathrm{var}(\boldsymbol{v}_k(\cdot, t)) = (1/12)$ if and only if $v_p^k(t) = 1$ for just one value of p and $v_q^k(t) = 0$ for all $q \neq p$.

Remark 3.2 For each $k \in \{1, 2, \ldots, 2n + n' + \|\mathcal{I}\|\}$, the minimum value of $\mathrm{var}(\boldsymbol{\vartheta}_k(\cdot))$ is $(1/12)$. Moreover, $\mathrm{var}(\boldsymbol{\vartheta}_k(\cdot)) = (1/12)$ if and only if $\vartheta_p^k = 1$ for just one value of p and $\vartheta_q^k = 0$ for all $q \neq p$.

From Remark 3.1 and Remark 3.2, we note that $\mathrm{var}(\boldsymbol{v}_k(\cdot, t)) = (1/12)$ (respectively $\mathrm{var}(\boldsymbol{\vartheta}_k(\cdot)) = (1/12)$) if and only if $u_k(t)$ (respectively ξ_k) takes on integer value in $\{0, 1, 2, \ldots, \overline{u_k}\}$ (respectively $\{0, 1, 2, \ldots, \overline{\xi_k}\}$). Thus, by replacing each discrete-valued control variable $u_k(t)$ with

$$\left(\sum_{p=1}^{\overline{u_k}+1} p \cdot v_p^k(t) \right) - 1 \tag{3.2}$$

and replacing each discrete-valued system parameters $\xi_k(t)$ with

$$\left(\sum_{p=1}^{\overline{\xi_k}+1} p \cdot \vartheta_p^k \right) - 1 \tag{3.3}$$

and by imposing the additional constraints

$$\int_0^{\overline{u_k}+1} \boldsymbol{v}_k(s, t)\,ds = 1, \quad k \in \{1, 2, \ldots, 2n + n' + \|\mathcal{I}\|\}, \quad t \in \{0, 1, 2, \ldots, T-1\}, \tag{3.4}$$

$$\int_0^{\overline{\xi_k}+1} \boldsymbol{\vartheta}_k(s)\,ds = 1, \quad k \in \{1, 2, \ldots, \|\mathcal{I}\|\}, \tag{3.5}$$

and

$$\mathrm{var}(\boldsymbol{v}_k(\cdot, t)) = \frac{1}{12}, \quad k \in \{1, 2, \ldots, 2n + n' + \|\mathcal{I}\|\}, \quad t \in \{0, 1, 2, \ldots, T-1\}, \tag{3.6}$$

$$\mathrm{var}(\boldsymbol{\vartheta}_k(\cdot)) = \frac{1}{12}, \quad k \in \{1, 2, \ldots, \|\mathcal{I}\|\}, \tag{3.7}$$

we obtain a standard discrete time optimal control problem, which is equivalent to problem $(\boldsymbol{P})$. Denote this problem as problem $(\boldsymbol{P}')$.

In practice, we do not solve problem $(\boldsymbol{P}')$ directly in its present form. Rather, we would replace constraints (3.6) by

$$\operatorname{var}(\boldsymbol{v}_k(\cdot,t)) \le \frac{1}{12} + \delta, \quad \text{for each } t \in \{0,1,2,\ldots,T-1\}$$
$$k \in \{1,2,\ldots,2n+n'+\|\mathcal{I}\|\}, \tag{3.8}$$

and constraints (3.7) by

$$\operatorname{var}(\boldsymbol{\vartheta}_k(\cdot)) \le \frac{1}{12} + \delta, \quad \text{for each } k \in \{1,2,\ldots,\|\mathcal{I}\|\} \tag{3.9}$$

where $\delta > 0$ is a small positive real number. This newly introduced parameter δ is used for the "Variance Constraints Relaxation" as mentioned in [10]. We denote this new problem as problem $(\boldsymbol{P}''_\delta)$.

It can be shown that $(\boldsymbol{P}''_\delta)$ is equivalent to problem $(\boldsymbol{P}')$, or problem $(\boldsymbol{P})$, in the limiting case when δ tends to zero.

Note that all control variables $v_{p,k}(t)$, and all system parameters $\vartheta_{p,k}$ are ranging continuously in $[0,1]$. Hence, the combinatorial nature of the problem $(\boldsymbol{P})$ is avoided. Problem $(\boldsymbol{P}''_\delta)$ can thus be handled directly by standard optimal control technique.

In order to adopt standard technique (such as the one reported in [14, 8]) for solving Problem $(\boldsymbol{P}''_\delta)$, we re-cast the problem in canonical form first. For convenience, we define $\boldsymbol{v}(t)$ as:

$$\boldsymbol{v}(t) = \begin{bmatrix} v_1^1(t) \\ \vdots \\ v_{q'}^{q''}(t) \end{bmatrix}$$

where $q'' = 2n + n' + \|\mathcal{I}\|$ and $q' = \overline{u_{q''}} + 1$, and $\boldsymbol{\vartheta}$ as:

$$\boldsymbol{\vartheta} = \begin{bmatrix} \vartheta_1^1 \\ \vdots \\ \vartheta_{q'''}^{q''''} \end{bmatrix}$$

where $q'''' = \|\mathcal{I}\|$ and $q''' = \overline{\xi_{q''''}} + 1$. Hence, we can now rewrite (2.20) as:

$$G_0(\boldsymbol{v}(\cdot),\boldsymbol{\vartheta}) = \phi_0(\boldsymbol{x}(T),\boldsymbol{\vartheta}) + \sum_{t=0}^{T-1} g_0(t,\boldsymbol{x}(t),\boldsymbol{v}(t),\boldsymbol{\vartheta}), \tag{3.10}$$

the system dynamics (2.9) as:

$$x(t+1) = f(t, x(t), v(t), \vartheta) \tag{3.11}$$

for $t = 0, 1, 2, \ldots, T-1$ with the initial condition

$$x(0) = x_0,$$

constraints (2.15), (2.16), (3.4), (3.5), (2.13), (3.8) as:

$$G_k(v(\cdot), \vartheta) = \phi_k(x(T), \vartheta) + \sum_{t=0}^{T-1} g_k(t, x(t), v(t), \vartheta) = 0, \quad k = 1, \ldots, n_e \tag{3.12}$$

and constraints (2.14), (2.19), (3.9) as:

$$G_k(v(\cdot), \vartheta) = \phi_k(x(T), \vartheta) + \sum_{t=0}^{T-1} g_k(t, x(t), v(t), \vartheta) \geq 0, \quad k = n_e + 1, \ldots, n_g \tag{3.13}$$

with each component of the control $v(t)$ and system parameters ϑ confined in $[0, 1]$.

Note that among all canonical constraints (3.12), those arising from (2.13), (3.8) are of the form

$$G_k(v(\cdot), \vartheta) = \phi_k(x(T), \vartheta) + \sum_{t=0}^{T-1} g_k(t, x(t), v(t), \vartheta)$$
$$= 0 + \sum_{t=0}^{T-1} \min\{h_k(t, x(t), v(t)), 0\} = 0,$$

which are non-smooth. We approximate the non-smooth function $g_k = \min\{h_k, 0\}$ by a smooth one, $g_\varepsilon(h)$, where

$$g_{k,\varepsilon}(h) = \begin{cases} h_k, & \text{if } h_k \leq -\varepsilon, \\ -(h_k - \varepsilon)^2/4\varepsilon, & \text{if } -\varepsilon < h_k < \varepsilon, \\ 0, & \text{if } h_k \geq \varepsilon. \end{cases}$$

Define

$$G_{k,\varepsilon}(v(\cdot), \vartheta) = \sum_{t=0}^{T-1} g_{k,\varepsilon}(t, x(t), v(t))$$

and, we may approximate these canonical constraints (3.12) which arising from (2.13), (3.8) by

$$G_{k,\varepsilon}(\boldsymbol{v}(\cdot), \boldsymbol{\vartheta}) + \tau \geq 0.$$

In [14], it is shown that if τ is chosen as $\varepsilon/4$, then, for any $\varepsilon > 0$, any feasible solution of this approximate constraint is also a feasible solution of the original constraint.

The Hamiltonian for each canonical function $k = 0, 1, 2, \ldots, n_g$ is given by

$$\begin{aligned} H_\varepsilon^k(t, \boldsymbol{x}(t), \boldsymbol{v}(t), \boldsymbol{\vartheta}, \boldsymbol{\lambda}(t+1)) = {} & g_{k,\varepsilon}(t, \boldsymbol{x}(t), \boldsymbol{v}(t), \boldsymbol{\vartheta}) \\ & + [\boldsymbol{\lambda}^k(t+1)]^\top \boldsymbol{f}(t, \boldsymbol{x}(t), \boldsymbol{v}(t), \boldsymbol{\vartheta}), \end{aligned}$$

where

$$[\boldsymbol{\lambda}^k(t)]^\top = \frac{\partial H_\varepsilon^k}{\partial \boldsymbol{x}} = \frac{\partial g_{k,\varepsilon}}{\partial \boldsymbol{x}} + [\boldsymbol{\lambda}^k(t+1)]^\top \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}, \quad t = 0, 1, \ldots, \tau_k - 1 \tag{3.14}$$

and

$$[\boldsymbol{\lambda}^k(T)]^\top = \frac{\partial \phi_k(\boldsymbol{x}(T), \boldsymbol{\vartheta})}{\partial \boldsymbol{x}(T)}.$$

It can be shown that the gradient formula is given by

$$\frac{\partial G_{k,\varepsilon}(\boldsymbol{v}(\cdot), \boldsymbol{\vartheta})}{\partial v_j} = \sum_{t=0}^{T-1} \frac{\partial H_\varepsilon^k}{\partial v_j} \quad \text{for } j \in \left\{1, 2, \ldots, T \cdot \sum_{k=1}^{q''} (\overline{u_k} + 1)\right\} \tag{3.15}$$

where v_j is the jth component of the vector

$$\boldsymbol{v} = [\boldsymbol{v}(0)^\top, \boldsymbol{v}(1)^\top, \boldsymbol{v}(2)^\top, \ldots, \boldsymbol{v}(T-1)^\top]^\top,$$

and

$$\begin{aligned} & \frac{\partial G_{k,\varepsilon}(\boldsymbol{v}(\cdot), \boldsymbol{\vartheta})}{\partial \vartheta_j} = \frac{\partial \phi_k(\boldsymbol{x}(T), \boldsymbol{\vartheta})}{\partial \vartheta_j} + \sum_{t=0}^{T-1} \frac{\partial H_\varepsilon^k}{\partial \vartheta_j} \\ & \text{for } j \in \left\{1, 2, \ldots, \sum_{k=1}^{q''''} (\overline{\xi_k} + 1)\right\}. \end{aligned} \tag{3.16}$$

The problem can thus be solved using the Constrained Quasi-Newton method, or the Sequential Quadratic Programming technique. For this, we need the values of the cost function and all the costates as well as the gradient of each canonical functions corresponding to each $(\boldsymbol{\nu}, \boldsymbol{\vartheta}) \in \mathbb{R}^{\varrho} \times \mathbb{R}^{\varphi}$, where

$$\varrho = T \cdot \sum_{k=1}^{q''} (\overline{u_k} + 1) \quad \text{and} \quad \varphi = \sum_{k=1}^{q''''} (\overline{\xi_k} + 1).$$

The algorithm for computing the gradient can be achieved as follows:

Step 1 Given an initial guess of $\boldsymbol{\nu}$ and $\boldsymbol{\vartheta}$.
Step 2 Solve the system of the state difference Eq. (2.9) forward (with each $u_k(t)$ replaced by (3.2)) from $t=0$ to $t=T-1$.
Step 3 Solve the co-state difference Eq. (3.14) with boundary condition (3.15) backward from $t=T-1$ to $t=0$.
Step 4 Compute the gradient of G_k by (3.15) and (3.16).

We can now use the Constrained Quasi-Newton method, or the Sequential Quadratic Programming technique, to solve Problem $(\boldsymbol{P}''_{\delta})$. The software packages DMISER3.2 [8] was developed based on this approach.

4. AN ILLUSTRATIVE EXAMPLE

Suppose there are 2 tasks and 3 types of workers. Type 1 workers can do task 1 only, type 2 workers can do task 2 only, but type 3 workers can do both tasks. There are trainning programmes for type 1 and type 2 workers to upgrade into type 3 workers. The duration of the trainning programme for type 1 workers is 2 unit of time, whereas the duration of the trainning programme for type 2 workers is just 1 unit of time. Initially, there are 2 type 1 workers, 2 type 2 workers, but none of the type 3 workers. None of the workers are undergoing trainning programme at $t=0$. The management can only employ, or terminate, or assign to trainning, at most 2 workers of any type at any time. We are optimizing over the period

$t = 0, 1, 2, \ldots, 9$. The minimum demand of workers at each time t for each task is given as follows:

$$\begin{array}{ll} d_1(0) = 2 & d_2(0) = 2 \\ d_1(1) = 2 & d_2(1) = 3 \\ d_1(2) = 3 & d_2(2) = 1 \\ d_1(3) = 2 & d_2(3) = 2 \\ d_1(4) = 1 & d_2(4) = 7 \\ d_1(5) = 2 & d_2(5) = 2 \\ d_1(6) = 6 & d_2(6) = 2 \\ d_1(7) = 2 & d_2(7) = 8 \\ d_1(8) = 2 & d_2(8) = 5 \\ d_1(9) = 6 & d_2(9) = 1 \end{array}$$

The corresponding costs are given by

	Cost to newly employ	Cost to terminate	Wages	Cost to train
Type 1 workers	1.2	2.0	1.0	0.3
Type 2 workers	1.5	2.4	1.2	0.5
Type 3 workers	3.0	4.0	2.0	–

The system is given by

$$\begin{aligned} x_1(t+1) &= x_1(t) + u_1(t) - u_4(t) - u_7(t) \\ x_2(t+1) &= x_2(t) + u_2(t) - u_5(t) - u_8(t) \\ x_3(t+1) &= x_3(t) + u_3(t) - u_6(t) + x(5) + x(6) \end{aligned}$$

$$\begin{aligned} x_4(t+1) &= u_7(t) \\ x_5(t+1) &= x_4(t) \\ x_6(t+1) &= u_8(t) \end{aligned}$$

with constraints

$$\begin{aligned} u_9(t) + u_{10}(t) &= x_3(t) \\ x_1(t) + u_9(t) &\geq d_1(t) \\ x_2(t) + u_{10}(t) &\geq d_2(t) \\ x_i(t) &\geq 0 \quad i = 1, 2, \ldots, 6. \end{aligned}$$

In this example, since type 1 workers can only do task 1, and type 2 workers can only do task 2, we simply put $w_{11}(t) = x_1(t)$ and $w_{22}(t) = x_2(t)$.

$$u_k(t) \in \{0, 1, 2\} \quad k = 1, 2, \ldots, 8,$$
$$u_k(t) \in \{0, 1, 2, \ldots, \overline{u_k}\} \quad k = 9, 10.$$

We take $\overline{u_9} = \overline{u_{10}} = 9$.

Our initial guess to start the optimization is $v_{i,k}(t) = (1/\overline{u_k} + 1)$ for all k and for all $t = 0, 1, 2, \ldots, 9$. Note that this initial guess is not even feasible !!

The converged solution computed by DMISER3.2 is as follows:

The control $\boldsymbol{u}(t)$:

	$t=0$	$t=1$	$t=2$	$t=3$	$t=4$	$t=5$	$t=6$	$t=7$	$t=8$	$t=9$
u_1	0	0	0	0	0	1	0	0	0	×
u_2	0	0	2	2	0	0	0	0	0	×
u_3	1	0	0	0	0	0	0	0	0	×
u_4	0	0	0	0	0	0	0	0	0	×
u_5	0	0	0	0	0	0	0	2	0	×
u_6	0	0	0	0	0	0	0	0	0	×
u_7	0	0	0	1	0	0	0	0	0	×
u_8	0	0	0	0	2	0	0	0	0	×
u_9	0	0	1	1	0	1	4	0	1	4
u_{10}	0	1	0	0	1	0	0	4	3	0

	$x_1(t)$	$x_2(t)$	$x_3(t)$	$x_4(t)$	$x_5(t)$	$x_6(t)$
$t=0$	2	2	0	0	0	0
$t=1$	2	2	1	0	0	0
$t=2$	2	2	1	0	0	0
$t=3$	2	4	1	0	0	0
$t=4$	1	6	1	1	0	0
$t=5$	1	4	1	0	1	2
$t=6$	2	4	4	0	0	0
$t=7$	2	4	4	0	0	0
$t=8$	2	2	4	0	0	0
$t=9$	2	2	4	0	0	0

Checking the minimum requirement demand constraints, *i.e.*, minimum number of workers engaged on each task

	Task 1 $x_1(t)+u_9(t)$ $(\geq d_1(t))$	Task 2 $x_2(t)+u_{10}(t)$ $(\geq d_2(t))$
$t=0$	$2(\geq 2)$	$2(\geq 2)$
$t=1$	$2(\geq 2)$	$3(\geq 3)$
$t=2$	$3(\geq 3)$	$2(\geq 1)$
$t=3$	$3(\geq 2)$	$4(\geq 2)$
$t=4$	$1(\geq 1)$	$7(\geq 7)$
$t=5$	$2(\geq 2)$	$4(\geq 2)$
$t=6$	$6(\geq 6)$	$4(\geq 2)$
$t=7$	$2(\geq 2)$	$8(\geq 8)$
$t=8$	$3(\geq 2)$	$5(\geq 5)$
$t=9$	$6(\geq 6)$	$2(\geq 1)$

Note that it satisfies all the demand requirements. The converged total cost is 119.1.

5. CONCLUSIONS

A novel mathematical formulation of a manpower planning problem is provided with which the model includes scheduling different types of workers over different tasks, empolying and terminating different types of workers, and assigning different types of workers to various trainning programmes. This can be considered as a first attempt towards the mathematical formulation of the real practical manpower planning problem, which is of large scale optimization nature, applicable to corporations in auto-industries, steel industries *etc.*, *etc.* In view of this, future research can be done to extend the applicability of the formulation and the computational technique to the national level of manpower management. People from different age-group, receiving different levels of education, can be considered as different "types of workers". The government may have different demands on what portions of the population should be participating in various sectors at different levels from time to time. This can be considered as different "tasks". Public money could be spent more optimally (to vocational institutes or universities) for manpower trainnings, when the demands of different tasks vary.

Acknowledgment

The authors are grateful to Professor N.U. Ahmed for his valuable comments and suggestions which have helped to improve the paper.

References

[1] Baker, K. (1976). Workforce Allocation in Cyclica Scheduling Problems: A Survey, *Operations Research Quarterly*, **27**, 155 – 167.
[2] Bechtold, S. E., Brusco, M. J. and Showalter, M. J. (1991). A Comparative Evaluation of Labour Tour Scheduling Methods, *Decision Sciences*, **22**, 683 – 699.
[3] Brusco, M. J. and Jacobs, L. W. (1993). A Simulated Annealing Approach to the Cyclic Staff-Scheduling Problem, *Naval Research Logistics*, **40**, 69 – 84.
[4] Cai, X. Q. and Li, K. N., A Genetic Algorithm for Scheduling Staff of Mixed skills under Multi-criteria, *European Journal of Operational Research*, (to appear).
[5] De Maio, A., Guardabassi, G., Locatelli, A. and Rinaldi, S., Optimal Manpower Trainning, *Computing Methods in Optimization Problems*, *Lecture Notes in Operations Research and Mathematical Economics*, **14**, Springer-Verlag, Berlin, 1969.
[6] Easton, F. F. and Rossin, D. F. (1991). Equivalent Alternate Solutions for Employee Scheduling Problems, *Decision Sciences*, **22**, 985 – 1007.
[7] Goh, C. J., Teo, K. L. and Goh, T. N. (1987). Optimal Planning of Manpower Trainning Programmes, *Opsearch*, **24**, 240 – 248.
[8] Jennings, L. S., Fisher, M. E., Teo, K. L. and Goh, C. J., *MISER3 Optimal Control Software version 2.0*: *Theory and User Manual*, 1997. (http://www.cado.uwa.edu.au/miser/manual.html)
[9] Lee, H. W. J., Teo, K. L., Rehbock, V. and Jennings, L. S. (1999). Control Parametrization Enhancing Technique for Optimal Discrete-Valued Control Problems, *Automatica*, **35**, 1401 – 1407.
[10] Lee, H. W. J., Teo, K. L. and Cai, X. Q. (1998). An Optimal Control Approach to Nonlinear Mixed Integer Programming Problems, *Computers and Mathematics with Applications*, **36**, 87 – 105.
[11] Loucks, J. S. and Jacobs, F. R. (1991). Equivalent Alternate Solutions for Employee Scheduling Problems, *Decision Sciences*, **22**, 985 – 1007.
[12] Narasimhan, R. (1997). An Algorithm for Single Shift Scheduling of Hierarchical Workforce, *European Journal of Operational Research*, **96**, 113 – 121.
[13] Tanomaru, J., Staff Scheduling by a Genetic Algorithm with Heuristic Operators, *Proceedings of the 1995 IEEE International Conference on Evolutionary Computation* (Perth, Australia), pp. 456 – 61, 29th Dec. 1995.
[14] Teo, K. L., Goh, C. J. and Wong, K. H., A Unified Computational Approach to Optimal Control Problems, *Essex*: *Longman Scientific and Technical*, UK, 1991.
[15] Teo, K. L., Jennings, L. S., Lee, H. W. J. and Rehbock, V. (1999). The Control Parametrization Enhancing Transform for Constrained Optimal Control Problems, *Journal of the Australian Mathematical Society*, *Series B*, **40**, 314 – 335.
[16] Whitley, D. (1989). The GENITOR Algorithm and Selection Presure: Why Rank-based Allocation of Reproductive Trials Is Best, *Proceedings of the Third International Conference on Genetic Algorithms*, pp. 116 – 121.