Two-level Nonlinear Integer Programming Problem through Genetic Algorithms for Obtaining Stackelberg Solutions

Md. Abul Kalam Azad, Masatoshi Sakawa, Kosuke Kato & Hideki Katagiri
Department of Artificial Complex Systems Engineering, Graduate School of Engineering
Hiroshima University, 1-4-1 Kagamiyama, Higashi Hiroshima 739-8527, Japan
(azad, sakawa, kato, katagiri)@msl.sys.hiroshima-u.ac.jp

***Abstract*- In the present paper, we focus on two-level nonlinear integer programming problems (TLNLIPPs) in which there exist a decision maker (the leader, DM1) with integer decision variables at the upper level and another decision maker (the follower, DM2) with integer decision variables at the lower level. Various approaches for two-level programming problems could exist according to situations, which the DMs are placed in. Assuming the noncooperative relationship between the DM at the upper level and one at the lower level, in this paper, genetic algorithms for TLNLIPPs is proposed to obtain Stackelberg solutions for the DMs. Furthermore, the feasibility of the proposed method is shown by applying it to illustrative numerical examples.**

I. INTRODUCTION

In this paper, we consider two-level nonlinear integer programming problems (TLNLIPPs) for obtaining Stackelberg solutions in which the decision makers control integer decision variables at each level and do not have any motivation to cooperate with each other. It is assumed that the decision maker at the upper level (the leader, DM1) and the decision maker at the lower level (the follower, DM2) completely know their objective functions and the constraints of the problem and they do not have any motivation to cooperate with each other, and the leader first makes a decision and then the follower specifies a decision so as to optimize the objective function of itself with full knowledge of the decision of the leader. On this assumption the leader also makes a decision such that the objective function of the leader is optimized. Then a solution defined as the above mentioned procedure is called a Stackelberg (equilibrium) solutions, which has been employed as a solution concept for two-level mathematical programming problem [3, 4, 5, 6, 7].

W. F. Bialas and M. H. Karwan [5] proposed four algorithms based on vertex enumeration and Kuhn-Tucker approches to solve two-level linear programming problems for obtaining Stackelberg solutions where two of them can provide local optimal solutions, and remaining two yield global optima. It is known that finding Stackelberg solutions of the two-level programming problem is strongly NP-hard [7] i.e. in proportion as scale of the problem, computational time exceedingly increases. To reduce computational time, Nishizaki et al. [3, 4] proposed computational methods through genetic algorithms for obtaining Stackelberg solutions to two-level zero-one and mixed zero-one linear programming problems. But all the methods concern with only linear programming problems. Jan et al. [6] proposed a solution method for obtaining Stackelberg solutions to nonlinear integer bilevel programming problems. They consider linear constraints only.

Two-level nonlinear integer programming problems (TLNLIPPs) can be formulated as large-scale mathematical programming problems involving integer decision variables, nonlinear objective functions and nonlinear constraint functions. Since a general solution method does not exist for nonlinear integer programming problems like the branch and bound method for linear ones, a solution method peculiar to each problem has been proposed. As a general-purpose solution method for nonlinear integer programming problems, we propose the usage of genetic algorithms with double strings based on reference solution updating (GADSRSU) [1, 2]. Under these circumstances, in this paper, for obtaining Stackelberg solutions in noncooperative relationship between the DMs, solution method through proposed GADSRSU is presented for two-level nonlinear integer programming problems (TLNLIPPs). Furthermore, the feasibility of the proposed method is shown through illustrative numerical examples with different number of variables.

II. PROBLEM FORMULATION

A two-level nonlinear integer programming problem for obtaining Stackelberg solutions is generally formulated as:

$$\left.\begin{array}{lll} \underset{\text{Leader (DM1)}}{\text{Upper level:}} & \underset{x_1}{\text{minimize}} & f_1(\boldsymbol{x}_1, \boldsymbol{x}_2) \text{ where } \boldsymbol{x}_2 \text{ solves} \\ \underset{\text{Follower (DM2)}}{\text{Lower level:}} & \underset{x_2}{\text{minimize}} & f_2(\boldsymbol{x}_1, \boldsymbol{x}_2) \\ & \text{subject to} & g_i(\boldsymbol{x}_1, \boldsymbol{x}_2) \le 0, i = 1, ..., m \\ & & x_{lj} \in \{ 0, 1, ..., v_{lj} \}, \\ & & \quad l = 1, 2, j = 1, ..., n_l \\ & & \boldsymbol{x}_1 \in X_1, \ \boldsymbol{x}_2 \in X_2 \end{array}\right\} \quad (1)$$

where $\boldsymbol{x}_1 \in X_1$ is an n_1 dimensional integer decision variable column vector for the decision maker at the upper level (leader), $\boldsymbol{x}_2 \in X_2$ is an n_2 dimensional integer decision variable column vector for the decision maker at the lower level (follower), and objective functions $f_l(\boldsymbol{x}_1, \boldsymbol{x}_2)$, $l = 1, 2,$ constraint functions $g_i(\boldsymbol{x}_1, \boldsymbol{x}_2), i = 1, ... m$ are real-valued functions. $f_1(\boldsymbol{x}_1, \boldsymbol{x}_2)$ is the leader's objective function and $f_2(\boldsymbol{x}_1, \boldsymbol{x}_2)$ is the follower's objective function.

Problem (1) can be interpreted as follows: On the assumption that the follower chooses a decision $\boldsymbol{x}_2$ with respect to the leader's decision $\boldsymbol{x}_1$ such that the objective function $f_2(\boldsymbol{x}_1, \boldsymbol{x}_2)$ is minimized, the leader first chooses a decision $\boldsymbol{x}_1$ so as to minimize the objective function $f_1(\boldsymbol{x}_1, \boldsymbol{x}_2)$. On this assumption the obtained solution is called Stackelberg solutions.

For example a manufacturer has several plants and warehouses located in the different parts of the country. The manufacturer subcontracts to a forwarding agent in order to transport products from factories to warehouses. The manufacturer wants to minimize total cost on the other hand the forwarding agent also wants to minimize transportation cost. Here the manufacturer is a decision maker at the upper level (leader, DM1) and the agent is a decision maker at the lower level (follower, DM2) and they do not have any motivation to cooperate with each other.

In the noncooperative relation between the DMs, Nishizaki et al. proposed two-level genetic algorithms for two-level linear zero-one and mixed zero-one programming problems for obtaining Stackelberg solutions [3, 4].

In this paper, focusing on the case of noncooperative relation between decision maker at the upper level (leader, DM1) and decision maker at the lower level (follower, DM2), we present the computational method through two-level genetic algorithms [3, 4] with double strings based on reference solution updating (GADSRSU) [1, 2] for obtaining Stackelberg solutions of problem (1).

III. Solution Procedures

In problem (1), a set $S(\boldsymbol{x}_1)$ of feasible decisions of the follower for $\boldsymbol{x}_1$ specified by the leader is represented by:

$$S(\boldsymbol{x}_1) = \{ \boldsymbol{x}_2 \mid g_i(\hat{\boldsymbol{x}}_1, \boldsymbol{x}_2) \le 0, i = 1, ..., m, \boldsymbol{x}_2 \in X_2 \} \quad (2)$$

and a set $R(\boldsymbol{x}_1)$ of rational responses of the follower for $\boldsymbol{x}_1$ is represented by:

$$R(\boldsymbol{x}_1) = \left\{ \boldsymbol{x}_2 \mid \boldsymbol{x}_2 \in \underset{\boldsymbol{x}_2 \in S(\boldsymbol{x}_1)}{\arg\min} f_2(\boldsymbol{x}_1, \boldsymbol{x}_2) \right\}, \quad (3)$$

where $\arg\min_{\boldsymbol{x}_2 \in S(\boldsymbol{x}_1)} f_2(\boldsymbol{x}_1, \boldsymbol{x}_2)$ denotes the set of $\boldsymbol{x}_2 \in S(\boldsymbol{x}_1)$ minimizing the function $f_2(\boldsymbol{x}_1, \boldsymbol{x}_2)$. A set $S(X_1)$ of feasible decisions of the leader is defined by:

$$S(X_1) = \{ \boldsymbol{x}_1 \mid \text{ There exists } \boldsymbol{x}_2 \text{ such that } g_i(\boldsymbol{x}_1, \boldsymbol{x}_2) \le 0, \\ i = 1, ..., m, \boldsymbol{x}_1 \in X_1, \boldsymbol{x}_2 \in X_2 \}. \quad (4)$$

In the proposed method to obtain Stackelberg solutions the leader first specifies his decision $\boldsymbol{x}_1$, then the follower's rational response $\boldsymbol{x}_2 \in R(\boldsymbol{x}_1)$ can be obtained through genetic algorithms. We assume that, for any $\boldsymbol{x}_1 \in S(X_1)$, the set $R(\boldsymbol{x}_1)$ of the follower's rational responses is a singleton. Let $\boldsymbol{x}_2(\boldsymbol{x}_1)$ denote a rational response, and then a pair $(\boldsymbol{x}_1, \boldsymbol{x}_2(\boldsymbol{x}_1))$ becomes a Stackelberg solutions of problem (1). To obtain $\boldsymbol{x}_2(\boldsymbol{x}_1)$ the following problem (5) is solved to minimize the follower's objective function under the assumption that the leader specifies his decision $\boldsymbol{x}_1$.

$$\left.\begin{array}{lll} \underset{\text{Follower (DM2)}}{\text{Lower level:}} & \underset{x_2}{\text{minimize}} & f_2(\hat{\boldsymbol{x}}_1, \boldsymbol{x}_2) \\ & \text{subject to} & g_i(\hat{\boldsymbol{x}}_1, \boldsymbol{x}_2) \le 0, i = 1, ..., m \\ & & x_{2j} \in \{ 0, 1, ..., v_{2j} \}, \\ & & \quad j = 1, ..., n_2 \\ & & \boldsymbol{x}_2 \in X_2 \end{array}\right\} \quad (5)$$

IV. Genetic Algorithms with Double Strings based on Reference Solution Updating

To obtain Stackelberg solutions to the problem (1), a solution procedure through two-level genetic algorithms is proposed in which search procedures are based on the mechanics of natural selection and natural genetics. It should be noticed that genetic algorithms have received much attention as a promising approximate computational method for large scale optimization problems. In the proposed method for obtaining Stackelberg solutions to two-level nonlinear integer programming problem (1) through two-level genetic algorithms, double strings representation [1, 2] of each individual is used. In two-level genetic algorithms [3, 4], the upper level genetic algorithms search the best individual for decision maker at the upper level (leader, DM1) and the lower level genetic algorithms search the best

individual for decision maker at the lower level (follower, DM2) corresponding to each individual specified by the leader.

In this section, we mention genetic algorithms with double strings based on reference solution updating (GADSRSU) proposed as a general solution method for nonlinear integer programming problem defined as (6).

$$\left.\begin{array}{ll} \text{minimize} & f(\boldsymbol{x}) \\ \text{subject to} & g_i(\boldsymbol{x}) \le 0, i = 1, ..., m \\ & x_j \in \{ 0, 1, ..., v_j \}, \ j = 1, ..., n \end{array}\right\} \quad (6)$$

In (6) $\boldsymbol{x}$ is an n dimensional integer decision variable vector, $f(\boldsymbol{x})$, $g_i(\boldsymbol{x})$, i = 1,…, m are nonlinear functions.

A. Individual Representation

The individual representation by double strings shown in Fig. 1 is adopted in GADSRSU.

Indices	$s(1)$	$s(2)$	$\cdots$	$s(j)$	$\cdots$	$s(n)$
Values	$y_{s(1)}$	$y_{s(2)}$	$\cdots$	$y_{s(j)}$	$\cdots$	$y_{s(n)}$

$= \mathbf{s}$

Fig. 1 Double Strings Representation.

In the figure, each of $s(j)$, j = 1,…, n is the index of an element in a solution vector and each of $y_{s(j)} \in \{ 0, 1, ..., v_j \}$, j = 1,…, n is the value of the element, respectively.

B. Decoding Algorithm

Let N_1 and N_2 be the number of population (pop_size) of the leader and the follower, respectively. The two sets of individuals $\boldsymbol{S}_{\boldsymbol{x}_1}$ and $\boldsymbol{S}_{\boldsymbol{x}_2}$ with the dimensions n_1 and n_2 respectively are generated randomly. With each of the individuals $\boldsymbol{S}_{\boldsymbol{x}_1}$ of the leader, the follower's rational response $R(\boldsymbol{x}_1)$ is determined by solving (5). Unfortunately, however, since the direct mapping of the leader individuals can generate infeasible solution $\boldsymbol{x}_1$ which yield problem (5) without any feasible solutions $\boldsymbol{x}_2$. To eliminate such solutions, as in [2], a decoding algorithm of double strings for nonlinear integer programming problem using a reference solution ($\boldsymbol{x}_1^0$, $\boldsymbol{x}_2^0$), which is a feasible solution of problem (1) and used as the origin of decoding for the leader, is constructed as follows.

Decoding algorithm for leader using reference solution

Let n_1, N_1 and ($\boldsymbol{x}_1^0$, $\boldsymbol{x}_2^0$) be the number of variables, the number of individuals in the population of leader and the reference solution, respectively.

Step 1: If the index of an individual to be decoded is in $\{1, ..., \lfloor N_1/2 \rfloor\}$, go to step 2. Otherwise, go to step 8.

Step 2: Let $j := 1$, $\boldsymbol{x}_1 := \{\mathbf{0}, \ldots, \mathbf{0}\}$, $l := 1$.

Step 3: Let $x_{1S(j)} := y_{1S(j)}$.

Step 4: If $g_i(\boldsymbol{x}_1, \boldsymbol{x}_2) \le 0$, i = 1, …, m, let $l := j$, $j := j$+1, and go to step 5. Otherwise, let $j := j$+1, and go to step 5.

Step 5: If $j \le n_1$, go to step 3. Otherwise, go to step 6.

Step 6: $l > 0$, go to step 7. Otherwise, go to step 8.

Step 7: By substituting $x_{1S(j)} := y_{1S(j)}$, $1 \le j \le l$ and $x_{1S(j)} := 0$, $l < j \le n_1$, we obtain a feasible solution $\boldsymbol{x}_1$ corresponding to the individual $S_{\boldsymbol{x}_1}$ and stop.

Step 8: Let $j := 1$, $\boldsymbol{x}_1 := \boldsymbol{x}_1^0$.

Step 9: Let $x_{1S(j)} := y_{1S(j)}$. If $y_{1S(j)} = x_{1S(j)}^0$, let $j := j$+1, and go to step 11. If $y_{1S(j)} \ne x_{1S(j)}^0$, go to step 10.

Step 10: If $g_i(\boldsymbol{x}_1, \boldsymbol{x}_2) \le 0$, i = 1, …, m, let $j := j$+1, and go to step 11. Otherwise, let $\boldsymbol{x}_{1S(j)} := \boldsymbol{x}_{1S(j)}^0$, $j := j$+1, and go to step 11.

Step 11: If $j \le n_1$, go to step 9. Otherwise, we obtain a feasible solution $\boldsymbol{x}_1$ from the individual $S_{\boldsymbol{x}_1}$ and stop.

For each decoded $\boldsymbol{x}_1$, to obtain follower's rational responses $\boldsymbol{x}_2(\boldsymbol{x}_1)$ the problem (5) is to be solved through genetic algorithms. Since this direct mapping of individuals $\boldsymbol{S}_{\boldsymbol{x}_2}$ also may generate infeasible solutions, a decoding algorithm using reference solution is proposed for the follower.

Decoding algorithm for follower using reference solution

Let n_2, N_2 and ($\boldsymbol{x}_1^0$, $\boldsymbol{x}_2^0$) be the number of variables, the number of individuals in the population of follower and the reference solution, respectively.

Step 1: If the index of an individual to be decoded is in $\{1, ..., \lfloor N_2/2 \rfloor\}$, go to step 2. Otherwise, go to step 8.

Step 2: Let $j := 1$, $\boldsymbol{x}_2 := \{\mathbf{0}, \ldots, \mathbf{0}\}$, $l := 1$.

Step 3: Let $x_{2S(j)} := y_{2S(j)}$.

Step 4: If $g_i(\hat{\boldsymbol{x}}_1, \boldsymbol{x}_2) \le 0$, i = 1, …, m, let $l := j$, $j := j$+1, and go to step 5. Otherwise, let $j := j$+1, and go to step 5.

Step 5: If $j \le n_2$, go to step 3. Otherwise, go to step 6.

Step 6: $l > 0$, go to step 7. Otherwise, go to step 8.

Step 7: By substituting $x_{2S(j)} := y_{2S(j)}$, $1 \le j \le l$ and $x_{2S(j)} := 0$, $l < j \le n_2$, we obtain a feasible solution $\boldsymbol{x}_2$ corresponding to the individual $S_{\boldsymbol{x}_2}$ and stop.

Step 8: Let $j := 1$, $\boldsymbol{x}_2 := \boldsymbol{x}_2^0$.

Step 9: Let $x_{2S(j)} := y_{2S(j)}$. If $y_{2S(j)} = x_{2S(j)}^0$, let $j := j$+1, and go to step 11. If $y_{2S(j)} \ne x_{2S(j)}^0$, go to step 10.

Step 10: If $g_i(\hat{\boldsymbol{x}}_1, \boldsymbol{x}_2) \le 0$, i = 1, …, m, let $j := j$+1, and go to step 11. Otherwise, let $\boldsymbol{x}_{2S(j)} := \boldsymbol{x}_{2S(j)}^0$, $j := j$+1, and go to step 11.

Step 11: If $j \le n_2$, go to step 9. Otherwise, we obtain a feasible solution $\boldsymbol{x}_2$ from the individual $S_{\boldsymbol{x}_2}$ and stop.

These decoding algorithms enable us to decode each of the

individuals represented by the double strings to the corresponding feasible solution. However, the diversity of the solution $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ greatly depend on the reference solution, because solutions obtained by the decoding algorithms using reference solution tend to concentrate around the reference solution. To overcome such situations, the reference solution updating procedure [1, 2] is adopted here.

C. Fitness Function

Nature obeys the principle of Darwinian "survival of fittest", the individuals with high fitness values will, on average, reproduce more often than those low fitness values. For obtaining Stackelberg solutions to two-level nonlinear integer programming problem (1) through GADSRSU, the objective function value is used as the fitness value f of an individual S for the leader and the follower. When the variance of fitness in a population is small, it is often observed that the ordinary roulette wheel selection does not work well because there is little difference between the probability of a good individual surviving and that of a bad one surviving [1, 2]. In order to overcome this problem, the linear scaling [1, 2] is adopted here. The fitness $F_1(S)$ of the leader and the fitness $F_2(S)$ of the follower are obtained by using the following linear scaling

$$F_l(S) = a_l f_l(S) + b_l$$

where $f_l(S)$, $l = 1, 2$ are the fitness values of the leader and the follower with respect to each decoded individual S.

D. Genetic Operators

For obtaining Stackelberg solutions to two-level nonlinear integer programming problem (1) through GADSRSU, three genetic operators such as reproduction, partially matched crossover (PMX) and mutation are adopted for the leader and the follower.

i. Reproduction

As a reproduction operator the elitist expected value selection is adopted here. The elitist expected value selection is a combination of elitist preserving selection and expected value selection.

Elitist preserving selection: One or more individuals with the largest fitness up to the current population is unconditionally preserved in the next generation.

Expected value selection: Let N denote the number of individuals in the population. The expected value of the number of the ith individual $\boldsymbol{S}_i$ in the next population is calculated as [1, 2]:

$$N_i = \frac{f(\boldsymbol{S}_i)}{\sum_{i=1}^{N} f(\boldsymbol{S}_i)} \times N \tag{7}$$

In the expected value selection, the integral part of $N_i(=\lfloor N_i \rfloor)$ denotes the definite number of individuals $\boldsymbol{S}_i$ preserved in the next population. While, using the fractional part of $N_i(= N_i - \lfloor N_i \rfloor)$, the probability to preserve $\boldsymbol{S}_i$, in the next population is determined by

$$\frac{N_i - \lfloor N_i \rfloor}{\sum_{i=1}^{N} (N_i - \lfloor N_i \rfloor)} \tag{8}$$

ii. Crossover

It is well recognized that the main distinguishing feature of genetic algorithms is the use of crossover. Crossover, also called recombination, is an operator which creates new individuals from the current population. The main role of this operator is to combine together pieces of information coming from different individuals in the population. Actually, it recombines genetic material of two parent individuals to create offspring for the next generation. In GADSRSU, partially matched crossover (PMX) [1, 2] is used.

Partially Matched Crossover (PMX) for double strings

Step 0: Let $i := 1$.

Step 1: Choose X and Y as parent individuals from the current population. Then, make copies X' and Y' of X and Y, respectively.

Step 2: If a random number **rand**() in [0,1] is less than or equal to the probability of crossover p_c, go to step 3. Otherwise, go to step 8.

Step 3: Choose two crossover points h and k (h < k) from $\{1, 2,\ldots, n\}$ at random. Let $j := h$.

Step 4: Find j' such that $S_{X'}(j') = S_Y(j)$. Then, interchange $\left(S_{X'}(j), y_{S_{X'}(j)}\right)^T$ with $\left(S_{X'}(j'), y_{S_{X'}(j')}\right)^T$ and let $j := j+1$.

Step 5: If $j > k$, go to step 6. Otherwise, return to step 4.

Step 6: For every j from h to k, let $y_{S_{X'}(j)} := y_{S_Y(j)}$ and go to step 7.

Step 7: Carry out the same operations as steps 4, 5, 6 for Y' and X.

Step 8: Preserve X' and Y' as offspring of X and Y respectively, and let $i := i+1$.

Step 9: If $i > N$, go to step 10. Otherwise, return to step 1.

Step 10: Choose $N \cdot G$ individuals randomly from $2 \cdot N$ offspring preserved by step 9 and substitute those for $N \cdot G$ individuals in the current population, where G denotes the generation gap.

iii. Mutation

It is considered that mutation plays the role of local random search in genetic algorithms. In GADSRSU, two

mutation operators (bit-reverse type and inversion) are used. The procedures of mutation of bit-reverse type and inversion [1, 2] are summarized as follows.

Mutation of bit-reverse type for double strings

Step 0: Let $i := 1$.

Step 1: Let $j := 1$.

Step 2: If a random number **rand**() in [0,1] is less than or equal to the probability of mutation p_m, go to step 3. Otherwise go to step 4.

Step 3: Determine $y_{S(j)}$ randomly according to the uniform distribution in $[0, v_j]$, and go to step 4.

Step 4: If $j < n$, let $j := j+1$ and return to step 2. Otherwise, go to step 5.

Step 5: If $i < N$, let $i := i+1$ and return to step 1. Otherwise, stop.

Inversion for double strings

Step 0: Let i := 1.

Step 1: If a random number **rand**() in [0,1] is less than or equal to the probability of inversion p_i, go to step 2. Otherwise, go to step 5.

Step 2: After choosing two inversion points h and k ($h < k$) form $\{1, 2,\ldots, n\}$, let $j := h$.

Step 3: Interchange $\left(S(j), y_{S(j)}\right)^T$ with $\left(S(k-(j-h)), y_{S(k-(j-h))}\right)^T$ and let $j := j+1$.

Step 4: If $j < h + \lfloor (k-h+1)/2 \rfloor$, return to step 3. Otherwise, go to step 5.

Step 5: If $i < N$, let $i := i+1$ and return to step 1. Otherwise, stop.

V. Proposed Algorithm Through Two-Level GADSRSU

To obtain Stackelberg solutions of problem (1), the leader first specifies his decision $\boldsymbol{S}_{\boldsymbol{x}_1}$ of individuals after decoded from N_1 individuals, then for each decoded individual of the leader, the follower's rational response $\boldsymbol{x}_2 \in R(\boldsymbol{x}_1)$ can be obtained by solving problem (5) through GADSRSU. Let $\boldsymbol{x}_2(\boldsymbol{x}_1)$ denote a rational response. After some predefined generations a pair $(\boldsymbol{x}_1^*, \boldsymbol{x}_2^*(\boldsymbol{x}_1^*))$ is obtained which becomes a Stackelberg solutions of problem (1). The proposed solution procedures through two-level GADSRSU are summarized as follows.

Step 0: Determine values of the parameters used in two-level GADSRSU: the population size N_1 and N_2 for the leader and the follower respectively, the generation gap G, the probability of crossover p_c, the probability of mutation p_m, the probability of inversion p_i, the scaling constant c_{mult}, the parameter for feasible solution θ, the parameter for reference solution updating η and the maximal search generation M_1 and M_2 for the leader and the follower respectively. Also determine the upper bound of the decision variables v for the leader and the follower.

Step 1: Find the feasible solution $(\boldsymbol{x}_1^0, \boldsymbol{x}_2^0)$ of (1) through GADSRSU without considering decoding algorithm.

Step 2: For the leader's decision variable $\boldsymbol{x}_1$, generate N_1 individuals $\boldsymbol{S}_{\boldsymbol{x}_1}$ at random and form an initial population.

Step 3: For each individual $S_{\boldsymbol{x}_1}$, after it is decoded from $S_{\boldsymbol{x}_1}$ to $\boldsymbol{x}_1$, the following procedure is repeated.

Step 3-1: Generate N_2 initial individuals $\boldsymbol{S}_{\boldsymbol{x}_2}$ for the decision variable $\boldsymbol{x}_2$ of the follower at random.

Step 3-2: After each individual $S_{\boldsymbol{x}_2}$ with decoded $\boldsymbol{x}_1$ is decoded from $(\boldsymbol{x}_1, S_{\boldsymbol{x}_2})$ to $(\boldsymbol{x}_1, \boldsymbol{x}_2)$, the follower's fitness of $(\boldsymbol{x}_1, S_{\boldsymbol{x}_2})$ is computed through a decoded $(\boldsymbol{x}_1, \boldsymbol{x}_2)$.

Step 3-3: Go to step 4 if the procedure of steps 3-1 to 3-4 for each $\boldsymbol{x}_1$ is repeated M_2 times, which is the maximal number of generations specified in advance.

Step 3-4: Apply a reproduction operator, and thereafter apply crossover and mutation operators to $\boldsymbol{S}_{\boldsymbol{x}_2}$ according to the probabilities of crossover and mutation. Return to step 3-2.

Step 4: The leader's fitness of each individual $S_{\boldsymbol{x}_1}$ is computed through a decoded $(\boldsymbol{x}_1, \boldsymbol{x}_2(\boldsymbol{x}_1))$.

Step 5: The algorithm stops if the procedure of steps 3 to 6 is repeated M_1 times, which is the maximal number of generations specified in advance, and then a pair $(\boldsymbol{x}_1^*, \boldsymbol{x}_2^*(\boldsymbol{x}_1^*))$ corresponding to an individual with the maximal fitness of the leader is an approximate Stackelberg solutions to the problem (1).

Step 6: Apply a reproduction operator, and thereafter apply crossover and mutation operators to $\boldsymbol{S}_{\boldsymbol{x}_1}$ according to the probabilities of crossover and mutation. Return to step 3.

VI. Numerical Example

$$
\left.
\begin{aligned}
&\underset{\text{DM1: } \boldsymbol{x}_1}{\text{maximize}} \quad f_1(\boldsymbol{x}_1, \boldsymbol{x}_2) = \prod_{j=1}^{n}\left[1-(1-r_j)^{x_j}\right] \quad \text{where } \boldsymbol{x}_2 \text{ solves} \\
&\underset{\text{DM2: } \boldsymbol{x}_2}{\text{minimize}} \quad f_2(\boldsymbol{x}_1, \boldsymbol{x}_2) = \sum_{j=1}^{n} q_j \left[x_j + \exp\left(\frac{x_j}{4}\right)\right] \\
&\text{subject to} \quad g_1(\boldsymbol{x}_1, \boldsymbol{x}_2) = \sum_{j=1}^{n} p_j x_j^2 - P \le 0 \\
&\qquad\qquad g_2(\boldsymbol{x}_1, \boldsymbol{x}_2) = \sum_{j=1}^{n} w_j x_j \exp\left(\frac{x_j}{4}\right) - W \le 0 \\
&\qquad\qquad x_{lj} = \{1, 2, ..., 10\},\ j = 1, ..., n,\ l = 1, 2
\end{aligned}
\right\} \tag{9}
$$

For obtaining Stackelberg solutions to two-level nonlinear

integer programming problems, we have considered the above two-level nonlinear integer programming problem to test the proposed algorithm.

A. Result and Discussion

The numerical experiments were performed on a personal computer (processor: Intel 1 MHz, memory: 512 MB, OS: Windows 2000) using Visual C/C++ compiler (version 6.0). The parameter values used in two-level GADSRSU for solving two-level nonlinear integer programming problem (9) for obtaining Stackelberg solutions were set as follows: N_1 = 50, N_2 = 30, M_1 = 500, M_2 = 200, θ = 5.0, c_{mult} = 1.8, η = 0.1, p_c = 0.9, G = 0.9, p_m = 0.05, p_i = 0.03 and v = 10. Several problems with different number of variables were considered to test the proposed algorithm for solving (9). The data were generated randomly. The results are summarized in Table 1 after 10 trails.

Table 1 Simulation results of problem (9)

Prob. No.	Upper level DM		Lower level DM		Time (sec.)
	Varia.	Obj.	Varia.	Obj.	
1	4	0.79	4	2986.80	332.81
2	5	0.84	5	3285.10	454.02
3	8	0.96	7	4706.99	692.80
4	10	0.78	10	6480.43	1023.53
5	15	0.49	15	10855.20	1819.12

Considering the second problem where the decision maker at the upper level (leader, DM1) controls 5 integer variables and the decision maker at the lower level (follower, DM2) controls other 5 integer variables. By using upper level genetic algorithms, the leader first specifies his decisions. For each decision of the leader, by using lower level genetic algorithms the follower's rational responses are obtained. After maximal search generations of the follower, the follower's objective is found and it is 3285.10. Then the leader's objective is also found after maximal search generations of the leader and it is 0.84.

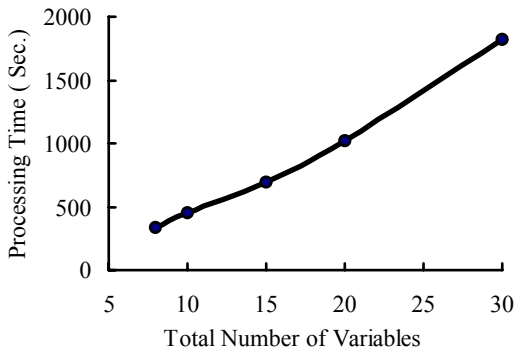

Fig. 2 Computational Time of Problem (9).

Fig. 2 shows the computational time of problem (9) with different total number of variables. From the figure it seems that the computational time increases almost polynomially with increasing the number of variables in the test problem.

VII. Conclusion

In this paper, focusing on two-level nonlinear integer programming problems (TLNLIPPs) in which there exist a decision maker (the leader, DM1) with integer decision variables at the upper level and another decision maker (the follower, DM2) with integer decision variables at the lower level, solution procedures through two-level genetic algorithms with double strings based on reference solution updating (GADSRSU) are presented for obtaining Stackelberg solutions. Furthermore, the feasibility of the proposed method is shown by applying it to illustrative numerical examples.

References

[1] M. Sakawa, Genetic Algorithms and Fuzzy Multiobjective Optimization, *Kluwer Academic Publishers, Boston, USA*, 2002.

[2] M. Sakawa, K. Kato, Genetic Algorithms with double strings for 0-1 programming problems, *European Journal of Operational Research*, vol. 144, pp. 581-597, 2003.

[3] I. Nishizaki, M. Sakawa and K. Kato, Computational methods through Genetic Algorithms for obtaining Stackelberg solutions to Two-level Zero-One Programming Problems, *Proceedings of 2000 IEEE International Conference on Industrial Electronics, Control and Instrumentation*, pp. 2750-2755, 2000.

[4] I. Nishizaki and M. Sakawa, Computational Methods through Genetic Algorithms for obtaining Stackelberg solutions to Two-Level Mixed Zero-One Programming Problems, *Cybernetics and Systems: An International Journal*, Vol. 31, pp. 203-221, 2000.

[5] W. F. Bialas and M. H. Karwan, Two-Level Linear Programming, *Management Science*, vol. 30, no. 8, pp. 1004-1020, 1984.

[6] R. Jan and M. Chern, Nonlinear Integer Bilevel Programming, *European Journal of Operational Research*, vol. 72, pp. 574-587, 1994.

[7] K. Shimizu, Y. Ishizuka and J.F. Bard, Nondifferentiable and Two-Level Mathematical Programming, *Kluwer Academic Publishers, Boston, USA*, 1997.