Artificial Immune System for Solving Generalized Geometric Problems: A Preliminary Results

Jui-Yu Wu
Department of Industrial Engineering and Management
Yuan Ze University
135 Yuan-Tung Road, Chung-Li 320, Taiwan R.O.C.
886-3-463-8800 EXT: 2516-2505
s909508@mail.yzu.edu.tw

Yun-Kung Chung
Department of Industrial Engineering and Management
Yuan Ze University
135 Yuan-Tung Road, Chung-Li 320, Taiwan R.O.C.
886-3-463-8800 EXT: 2503
ieychung@saturn.yzu.edu.tw

ABSTRACT
Generalized geometric programming (GGP) is an optimization method in which the objective function and constraints are nonconvex functions. Thus, a GGP problem includes multiple local optima in its solution space. When using conventional nonlinear programming methods to solve a GGP problem, local optimum may be found, or the procedure may be mathematically tedious. To find the global optimum of a GGP problem, a bio-immune-based approach is considered. This study presents an artificial immune system (AIS) including: an operator to control the number of antigen-specific antibodies based on an idiotypic network hypothesis; an editing operator of receptor with a Cauchy distributed random number, and a bone marrow operator used to generate diverse antibodies. The AIS method was tested with a set of published GGP problems, and their solutions were compared with the known global GGP solutions. The testing results show that the proposed approach potentially converges to the global solutions.

Categories and Subject Descriptors
G.1.6 **[Numerical Analysis]**: Optimization – *constrained optimization, nonlinear programming*

General Terms
Algorithms

Keywords
Generalized geometric programming, artificial immune system, nonlinear programming

1. INTRODUCTION
An immune system (IS) learns, memorizes, identifies and eliminates invading foreign materials such as viruses, pathogens and bacteria (called antigens, **Ags**), protecting the human body from disease. The natural IS has inspired an emerging computation field called artificial immune systems (AISs) [1]. In recent years, AISs have been applied to various function optimization problems such as multimodal and constrained nonlinear programming (NLP) optimizations [2, 3, 4], concluding that AISs work well to function optimization problems.

Duffin et al., introduced geometric programming (GP) whose name originally comes from the arithmetic-geometric mean inequality [5]. GP is a class of NLP in which the objective function and constraints are termed as posynomials, indicating that the formulation of GP is positive polynomial. Significantly, in GP the global minimum of the primal minimization problem with nonlinear constraints can be determined by solving the dual maximization problem with linear constraints. Subsequently, generalized geometric programming (GGP) was presented when GP formulation failed in many important engineering applications. Unfortunately, both primal and dual GGP formulation hold highly nonconvex functions, implying that they have multiple local optima on a complex constraint surface. To solve GGP problems, Maranas and Floudas presented a deterministic global optimization method including exponential variable transformation; convex lower bounding; variable scaling, and a branch and bound framework [6]. Although their optimization method yields a global solution to GGP problems, it may be mathematically tedious.

As mentioned above, this study presents an effective and efficient AIS approach. This study transforms a constrained GGP problem into an unconstrained problem with exterior penalty function, and then uses the proposed AIS to yield the optimal solution of the unconstrained GGP problem. The quality of proposed AIS solution is evaluated by comparing with the published global solution [7].

2. IMMUNE SYSTEM
Human immunity is divided into two types: innate and adaptive immunities. Innate immunity uses macrophages to ingest and digest **Ags** that invade the human body to provide an immediate host defense. The immune response has a feature that cannot be altered by repeated exposure to specific **Ags**, indicating that the response has no immunological memory. After innate immunity is achieved, adaptive immunity is activated. Conversely, adaptive immunity has an immunological memory for specific **Ags**. The adaptive immune response is composed of antigen-specific reactions of T and B cells (lymphocytes), which take part in cell-

GECCO'05, June 25–29, 2005, Washington, DC, USA.

mediated and homoral immunities respectively [8]. T cells have two types: cytotoix (killer) and helper. Cytotoix cells kill virally-infected cells. Th1 helper cells secrete cytokine to activate macrophages, and Th2 helper cells stimulate and suppress B cells to secrete antigen-specific antibodies (**Abs**) [9]. The **Abs** are receptors located on the B cells' surface. Each B cell can only produce one variety of **Abs**. T and B cells, which recognize and eradicate invading **Ags** to maintain the health of the host, circulate throughout the human body.

2.1 Operation of Adaptive Immune System

2.1.1 Ab-Ag and Ab-Ab Recognition

Figure 1 illustrates an **Ag** and an **Ab**. The **Ag** has multiple epitopes, which can be recognized by the **Ab**, on its surface. The **Ab** in figure 1 shows heavy **(H)** and light **(L)** polypeptide chains. Both chains are composed of variable **(V)** and constant **(C)** regions. The **V**-region recognizes the **Ag**, while the **C**-region is responsible for a variety of effector functions. A portion locating on the **V**-region of an **Ab** is called a paratope, which can match with the epitope of an **Ag**. The **Ab** with the **Ag** has a high **Ab-Ag** affinity when the paratope and the epitope have complementary shapes.

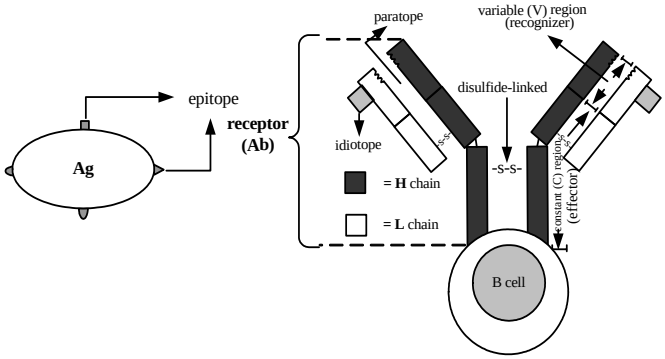

Figure 1. An Ab and an Ag representation

The **Ab** in Figure 1 also has two immunogentic idiotopes (antigenic determinants) that can be recognized by other **Abs**. From this perspective, Jerne [10] developed the immune theory of idiotypic network, as shown in Figure 2.

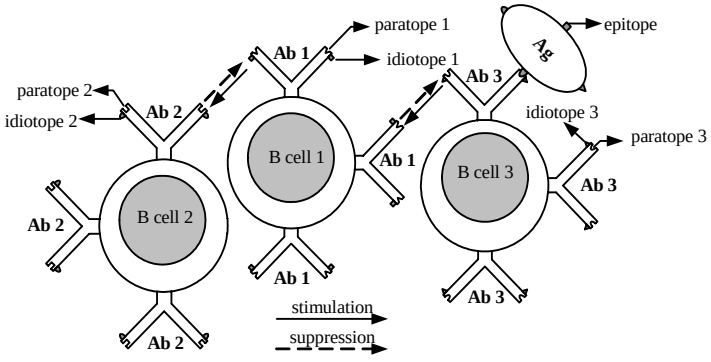

Figure 2. Idiotypic network hypothesis

The figure illustrates that B-cells are not isolated, instead the different **Abs** interact with each other to form an idiotypic network. For example, the epitope of the invading **Ag** matches with paratope 3 of **Ab 3** on the B cell 3 surface, and then the idiotope 3 of **Ab 3** stimulates (solid line in the figure) B cell 1 through paratope 1 of **Ab 1**, while B cell 1 simultaneously suppresses (dashed line in the figure) B cell 3 with the paratope 1 of **Ab 1**, and so forth. After Jerne presented his differential equation to describe the dynamics of a set of identical B cells, Perelson [11] reviewed and developed a general network described by:

$$RIP = B - D + R \tag{1}$$

where

RIP = rate of increase of B cells population

B = influx from bone marrow

D = death rate of unstimulated B cells

R = reproduction rate of stimulated B cells

The last term in Eq. (1) involves **Ab-Ag** and **Ag-Ag** recognition information. The equation maintains the number of diverse **Abs** and is used in this study. Additionally, this study considers an **Ab** as equal to a B cell.

2.1.2 Clonal Selection

Clonal selection is based on the description of basic principles of an adaptive immune response to an antigenic stimulus. Figure 3 depicts the clonal selection principle [4].

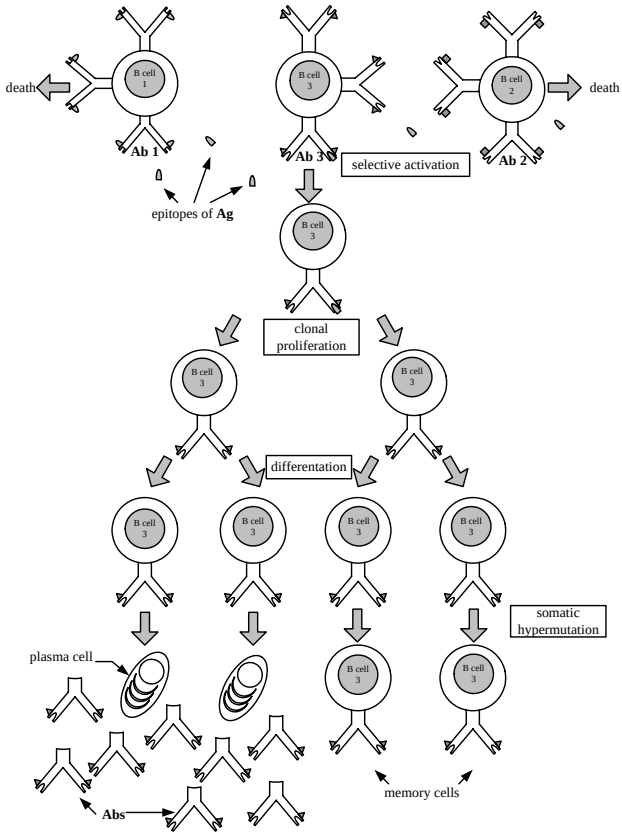

Figure 3. Clonal selection principle

As seen from Figure 3, the antigen-specific **Ab 3** with invading **Ag** has high **Ab-Ag** affinity, activating the antigen-specific B cell 3. By contrast, antigen-specific B cells (B cell 1 and 2 in the

figure) with low **Ab-Ag** affinity gradually die in the IS. The antigen-specific B cell 3 activated gets into the germinal centers of lymphoid follicles to undergo proliferation and differentiation. The antigen-specific B cell has two differentiation pathways: plasma and memory cells, as described below.

(1) Plasma cell: The antigen-specific B cells are transformed into effector cells such as plasma cells that can secrete a large number of antigen-specific **Abs** to destroy specific **Ags**.

(2) Memory cell: The antigen-specific B cell in the germinal centers undergoes an **Ab-Ag** affinity maturation process through somatic hypermutation to increase affinity for specific **Ags**. The somatic hypermutation can be achieved with the single-point mutation operations within the **V**-region of the antigen-specific **Ab**. The single-point mutation performs the local exploration. Additionally, the receptor editing has been presented to the **Ab-Ag** affinity maturation process to decrease the possibility of trapping into local optima on a very rough **Ab-Ag** affinity landscape. As discussed above, the hypermutation and the receptor editing play complementary roles in the **Ab-Ag** affinity maturation of **Abs** [12]. These antigen-specific B cells with high **Ab-Ag** affinity become memory cells in the human IS. The memory cells are transformed into plasma cells to secrete many antigen-specific **Abs** when the same or similar specific **Ags** invade the human body again. The immune response (secondary response) is stronger and faster than the primary response when the antigen-specific **Abs** first encounter the **Ags**.

2.1.3 Generation of Diverse Abs

In the human, bone marrow is an important organism for generating new **Abs** to maintain the number of **Abs** in the IS. Each **Ab** has two identical **H** and **L** chains, as defined in Session 2.1.1. The **H** chain's **V**-region consists of gene segments of variable $\mathbf{V_H}$, diverse $\mathbf{D_H}$ and joining $\mathbf{J_H}$, while the **L** chain's **V**-region is composed of gene segments of variable $\mathbf{V_L}$ and joining $\mathbf{J_L}$ [13]. These gene segments are individual libraries that generate diverse **Abs**. The paratope of **Ab** is formed by recombining $\mathbf{V_H}$ $\mathbf{D_H}$ $\mathbf{J_H}$ and $\mathbf{V_L}$ $\mathbf{J_L}$. The metaphor of gene segment rearrangement has inspired development of a bone marrow operator in this study.

3. GENERALIZED GEOMETRIC PROGRAMMING

The mathematical formulation of a primal GGP is described as follows:

$$\text{Minimize} \quad g_0(\mathbf{x}) = \sum_{t=1}^{T_0} \sigma_{0t} c_{0t} \prod_{n=1}^{N} x_n^{a_{0tn}} \tag{2}$$

$$\text{subject to} \quad g_m(\mathbf{x}) = \sum_{t=1}^{T_m} \sigma_{mt} c_{mt} \prod_{n=1}^{N} x_n^{a_{mtn}} \le \sigma_m \tag{3}$$

$$x_n > 0, \quad n = 1,2,\ldots,N \tag{4}$$

where

$g_0(\mathbf{x})$ = GGP objective function

N = number of decision variables

$\mathbf{x} = [x_1, x_2, \cdots, x_N]^T$, decision variables

M = number of inequality constraints

T_m = number of terms in mth constraints

$c_{mt} > 0$, $\sigma_{mt} = \pm 1$, $\sigma_m = \pm 1$, $t = 1,2,\ldots,T_m$, a_{mtn} = real number (stand for exponentiation here), $m = 0,1,\ldots,M$

Traditionally, two approaches in solving GGP problems are general: posynomial condensation [14] and pseudo-duality [15]. Posynomial condensation condenses the multiple polynomial terms of GGP to a posynomial with a single term and obtains optimal solution by iteratively calculating a sequence of posynomial approximations. In pseudo-duality, a weaker duality theory is applied to this class of NLP. These two approaches guarantee convergence to local minima at best.

4. PENALTY FUNCTION METHODS

Penalty functions are popularly used in evolutionary algorithms, such as evolution strategy and genetic algorithms (GAs), to solve constrained optimization problems. A survey to penalty function techniques can be found in [16]. A general constrained NLP formulation can be written as follows:

$$\text{Minimize} \quad \hat{g}_0(\mathbf{x}) \tag{5}$$

$$\text{subject to} \quad \hat{g}_m(\mathbf{x}) \le 0, \quad m = 1,2,\ldots,M \tag{6}$$

$$\hat{h}_k(\mathbf{x}) = 0, \quad k = 1,2,\ldots,K \tag{7}$$

where

$\hat{g}_0(\mathbf{x})$ = objective function

K = number of equality constraints

Exterior and interior penalty functions are the two most common penalty functions for handling the constraints of an NLP problem. These two penalty functions work by obtaining a pseudo-objective function with the original NLP objective function and constraints and penalize the pseudo-objective function when the constraints are violated. Table 1 compares the exterior with the interior functions. The exterior penalty function is employed in this study, since it does not require a feasible starting point and is conceptually easy to understand.

Table 1. Comparison of the exterior with the interior penalty function

Types \ Features	Starting Point	Convergence Direction
Exterior penalty function $f(\mathbf{x},\rho) = \hat{g}_0(\mathbf{x}) + \rho\left\{\sum_{m=1}^{M}\{\max[0,\hat{g}_m(\mathbf{x})]\}^2 + \sum_{k=1}^{K}\left[\hat{h}_k(\mathbf{x})\right]^2\right\}$ ρ = penalty parameter	Infeasible	From infeasible region to feasible region
Interior penalty function $f(\mathbf{x},\rho) = \hat{g}_0(\mathbf{x}) - \rho\sum_{m=1}^{M}\frac{1}{\hat{g}_m(\mathbf{x})} + \frac{1}{\rho^{1/2}}\sum_{k=1}^{K}\left[\hat{h}_k(\mathbf{x})\right]^2$	Feasible	From feasible region to constraint bound

5. METHOD

This section presents an AIS scheme including several IS operators for solving primal GGP problems, as shown in Figure 4 and described below.

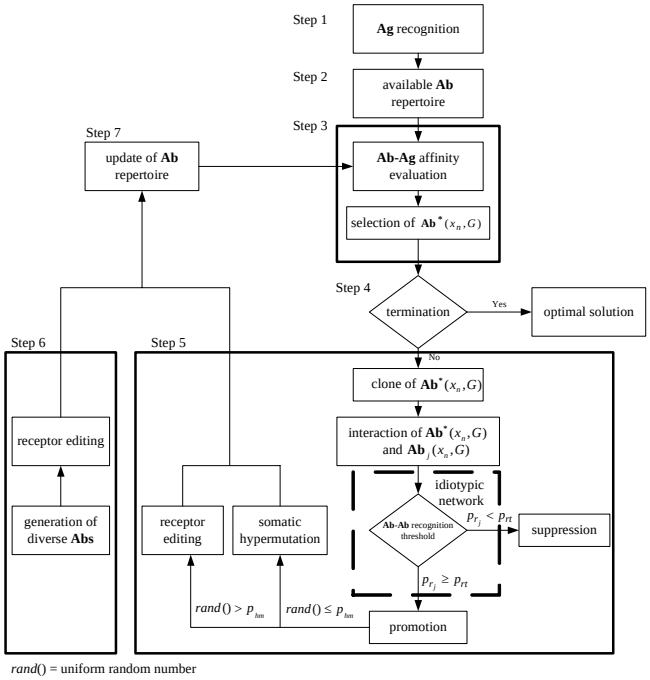

Figure 4. AIS scheme

Step 1: **Ag** Recognition

Figure 5 illustrates an **Ag** and an **Ab** representation used in the proposed AIS. The **Ag** epitope in the figure denotes the parameters c_{mt}, a_{mtn}, σ_{mt} and σ_m of a GGP problem; both the paratope and idiotope of the **Ab** describe the GGP decision variables. Because a GGP problem is a continuous optimization problem, using real numbers to represent its decision variables may yield a more accurate solution than obtained using binary encoding. Therefore, real numbers are used to represent **Ag** and **Ab**.

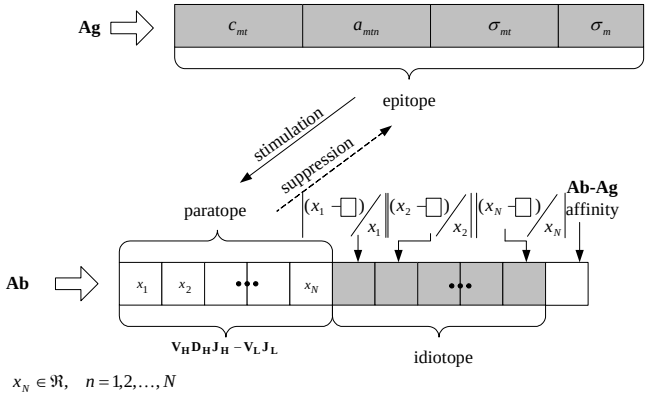

Figure 5. An Ab and Ag representation

Step 2: Available **Ab** Repertoire

An available **Ab** repertoire is formed from repertoire size (rs). Each **Ab** in the repertoire is created within $[x_n^l, x_n^u]$, where x_n^l and x_n^u are respectively the lower and upper bounds of the decision variable x_n. Like the population size in GAs, the **Abs** in a repertoire converges prematurely when rs is too small, while the computation time increases when rs is too large. This study found that $rs = 100$ worked well. The level of constraint violation of the **Abs** in the created repertoire is computed by:

$$vio_j = \sum_{m=1}^{M} \max[0, g_m(\mathbf{x}) - 1], \quad j = 1,2,\ldots,rs \tag{8}$$

and then sorted in ascending order by the vio_j. The **Ab** with the smallest constraint violation level is selected as a starting point.

Step 3: **Ab-Ag** Affinity Evaluation

A pseudo-objective function is obtained by exterior penalty function, as follows:

$$\text{maximum } affinity_j = \frac{1}{g_0(\mathbf{x}) + \rho \sum_{m=1}^{M} \{\max[0, g_m(\mathbf{x}) - 1]\}^2}, \; j = 1,2,\ldots,rs \tag{9}$$

Equation (9) is then applied to evaluate **Ab-Ag** affinity in this step. This study determined that each constraint of the best AIS solution obtained from a testing GGP problem must be accurate by up to five decimal places. To yield the resolution, $\rho = 10^9$ was found. Referring to Eq. (9), $\rho = 10^9$ can significantly penalize a solution obtained from an infeasible region and can provide a high-quality feasible solution.

After the **Ab-Ag** affinities of **Abs** in the **Ab** repertoire of current generation (G) are measured, the **Ab** with the highest **Ab-Ag** affinity is selected to undergo clonal selection in Step 5. The best **Ab** at G is defined by $\mathbf{Ab}^*(x_n, G), n = 1,2,\ldots,N$.

Step 4: Termination

The proposed AIS uses the maximum number of generations $(G_{\max})$ as its termination criterion. If the termination is met, then the AIS stops and an optimal solution in the form of the **Abs** is obtained; otherwise, the diverse **Abs** is developed in Steps 5-6.

Step 5:Clonal Selection

In this step, the $\mathbf{Ab}^*(x_n, G)$ selected in Step 3 plays the role of a specific **Ag**. The $\mathbf{Ab}^*(x_n, G)$ is recognized by other **Abs** in the repertoire of G. This study presents a so-called idiotypic network operator based on the idiotypic network hypothesis [10] to control the number of antigen-specific **Abs**. The operator is defined by:

$$p_{r_j} = \frac{1}{N} \sum_{n=1}^{N} \frac{1}{e^{d_n}} \tag{10}$$

$$d_n = \left| \frac{\mathbf{Ab}^*(x_n, G) - \mathbf{Ab}_j(x_n, G)}{\mathbf{Ab}^*(x_n, G)} \right|, \; j = 1,2,\ldots,rs, \;\; n = 1,2,\ldots,N \quad (11)$$

where

p_{r_j} = rate of recognition, within 0 and 1

$\mathbf{Ab}_j(x_n, G)$ = the jth **Ab** of representing decision variables x_n in the **Ab** repertoire of G

A large value of p_{r_j} means that the $\mathbf{Ab}_j(x_n, G)$ can effectively recognize the $\mathbf{Ab}^*(x_n, G)$. To determine the degree of the **Ab-Ab** recognition, a predetermined threshold value p_{rt} is used in the proposed AIS. These **Abs** with p_{r_j} which are equal to or larger than the p_{rt} is promoted, whereas others are suppressed. This study uses a high p_{rt} to eliminate inferior **Abs**, the p_{rt} = 0.999 is employed.

Subsequently, the promoted **Abs** is divided into two subsets based on a somatic hypermutation probability (p_{hm}). To generate a uniform random number for each promoted **Ab**, the promoted **Abs** are performed somatic hypermutation when their random number is equal to or smaller than a predetermined p_{hm} value, while other **Abs** undergo receptor editing. The operators of somatic hypermutation and receptor editing are described below.

(1) Somatic hypermutation

This study uses multi-non-uniform mutation [17] as the somatic hypermutation operator, which can be expressed as follows:

$$\hat{x}_n = \begin{cases} x_n + (x_n^u - x_n) A(G), \; if \; U(0,1) < 0.5 \\ x_n - (x_n + x_n^l) A(G), \; if \; U(0,1) \geq 0.5 \\ x_n, \qquad \text{otherwise} \end{cases} \quad (12)$$

where

$A(G) = \left[U_1(0,1) \left(1 - \frac{G}{G_{\max}} \right) \right]^b$ = perturbation factor

x_n = decision variable found in the solution space

$\hat{x}_n$ = perturbed decision variable

$U(0,1)$ and $U_1(0,1)$ = uniform random number

b = shape parameter, often is b = 2

The hypermuation has two tasks, uniform search and local fine-tuning.

(2) Receptor Editing

The random number of a standard Cauchy distribution $C(0,1)$ is used to implement receptor (**Ab**) editing, since it can create a large ("strong") perturbation to let the current points escape from a local **Ab-Ag** affinity landscape. The proposed Cauchy receptor editing is defined by:

$$\hat{\mathbf{x}} = \mathbf{x} + \{[U_2(0,1)] \times [1 - U_3(0,1)]\}^{\alpha} \times [\mathbf{C(0,1)} \text{ random number}] \quad (13)$$

where

$\mathbf{C(0,1)} = [C_1(0,1), C_2(0,1), \ldots, C_n(0,1)]^T, \; n = 1,2,\ldots,N$

$U_2(0,1)$ and $U_3(0,1)$ = uniform random number

α = a nonnegative exponent constant, often is 2

The receptor is responsible for performing local fine-tuning and large perturbation.

Step 6: Generation of Diverse **Abs**

The paratope of an **Ab** can be created by recombining $\mathbf{V_H}$ $\mathbf{D_H}$ $\mathbf{J_H}$ with $\mathbf{V_L}$ $\mathbf{J_L}$. Thus, this study presents a bone marrow operator based on the metaphor of gene segment rearrangement to synthesize diverse **Abs** to recruit the number of **Abs** eliminated in Step 5. The operator randomly selects two **Abs** from the promoted **Abs** and randomly chooses a recombination point from gene segments of the paratope of selected **Abs**. These two chosen gene segments are each given a random number from a standard normal distribution $N(0,1)$ and are exchanged to create two new **Abs**. The gene segments of the idiotopes corresponding with selected **Abs** are also exchanged. Finally, to increase diversity, the receptor editing operation is performed on the two new **Abs**.

Step 7: Update of **Ab** repertoire

A new **Ab** repertoire is created based on Eq. (1), consisting of the **Abs** generated from Steps 5 and 6. This step computes the **Ab-Ag** affinities of **Abs** in generated **Ab** repertoire and sorts them in descending order. If the highest **Ab-Ag** affinity in the sorted **Ab** repertoire is larger than the best **Ab-Ag** affinity at G , then half of the **Abs** in the sorted **Ab** repertoire are selected to create an **Ab** repertoire based on rs . The **Ab** repertoire becomes the **Ab** repertoire of the next generation. Conversely, if the highest **Ab-Ag** affinity in the sorted **Ab** repertoire is smaller than the best **Ab-Ag** affinity at G , then the **Ab** repertoire at G remains in the next generation. This process not only keeps the $\mathbf{Ab}^*(x_n, G)$ but also allow some secondary **Abs** to survive.

Steps 3-7 are repeated until the termination criterion is met.

5.1 Comparison

This study combines the metaphor of clonal selection with the idiotypic networks hypothesis to design the AIS approach, as presented in Section 5. Although these two theories contradict each another, they are useful in developing a function optimization tool. Table 2 compares the existing AIS with the proposed AIS in this study.

Table 2. Comparison of AIS

Methods / Features	CLONALG [4, 18]	AiNet [2]	The proposed AIS
Metaphor	Clonal selection	Clonal selection and idiotypic network	Clonal selection and idiotypic network

Table 2. Comparison of AIS (cont.)

Features \ Methods	CLONALG [4, 18]	AiNet [2]	The proposed AIS
Representation	Binary	Real value	Real value
Selection of **Abs** with high **Ab-Ag** affinity	Yes	Yes	Yes
Ab-Ab recognition (suppression/stimulation)	No	Yes	Yes
Hypermutation	Yes	Yes	Yes
Receptor editing	Yes	No	Yes
Introduction of diverse **Abs**	Randomly generated	Randomly generated	Bone marrow operator
Application	Multimodal function optimization	Multimodal function optimization	Nonconvex function optimization

6. RESULTS

The proposed AIS described in the previous section was applied to three testing GGP problems taken from Floudas and Pardalos [7]. Maranas and Floudas [6], who presented a deterministic global optimization method based on the convex relaxation and branch and bound on a hyper-rectangle region, reported the global solution of these three GGP problems. The best solution of the proposed AIS was compared with the known global solution. The AIS software was written in MATLAB and executed on a Pentium 4 (2.4 GHz) PC. Five p_{hm}s were used to test each GGP problem. Different starting points were generated from 50 available **Ab** repertoires, and the proposed AIS was run 50 times with these starting points for each p_{hm}. The following parameters were used in all evaluations:

$$rs = 100,\ b = 2,\ \rho = 10^9,\ \alpha = 2 \text{ and } p_{rt} = 0.999.$$

6.1 Test Problem 1

$$\min \quad g_0(\mathbf{x}) = (0.5x_1x_2^{-1} - x_1 - 5x_2^{-1})$$
$$s.t.\ g_1(\mathbf{x}) = 0.01x_2x_3^{-1} + 0.01x_1 + 0.0005x_1x_3 \le 1$$
$$1 \le x_1, x_2, x_3 \le 100$$

According to Eq. (9), the **Ab-Ag** affinity can be expressed by:

$$affinity_j = \frac{1}{\left\{\begin{array}{l}(0.5x_1x_2^{-1} - x_1 - 5x_2^{-1}) \\ +10^9 \times \left\{\max\left[0, (0.01x_2x_3^{-1} + 0.01x_1 + 0.0005x_1x_3 - 1)\right]\right\}^2\end{array}\right\}}$$

In solving test problem 1, the proposed AIS takes the values $p_{hm} = \{0, 0.05, 0.1, 0.5, 1\}$, where $p_{hm} = 0$ indicates that the promoted **Abs** only edits the receptor and $p_{hm} = 1$ means that the promoted **Abs** performs only somatic hypermutation. Table 3 presents numerical results, including the best, the mean, the worst, and the standard deviation (SD). The best AIS solution was found at $p_{hm} = 0.1$ and $p_{hm} = 0.5$, indicating that neither receptor editing alone nor somatic hypermutation can yield the best solution. This study prefers the small p_{hm} ($p_{hm} = 0.1$) when $p_{hm} = 0.1$ and $p_{hm} = 0.5$ obtain similar results. Additionally, Table 4 presents the analysis of variance of the parameter p_{hm}. The table shows that the F of 10.59 for p_{hm}s was statistically significant at the $P < 0.05$ level, resulting from differences in the means of AIS solutions. Table 5 compares the known global solution with the best solution to test problem 1 obtained from the proposed AIS. The table indicates that the AIS took 4.406 seconds of CPU computation time to yield the best solution. The best AIS solution $g_0(\mathbf{x}) = -83.2497$ is worse than the known global solution $g_0(\mathbf{x}) = -83.2535$, but the precision of constraint of the AIS solution is more precise than that of the known global solution, meaning that the AIS solution still has specific applications.

Table 3. Numerical results from test problem 1

p_{hm}	0	0.05	0.1	0.5	1
Best	-83.2495	-83.2496	-83.2497	-83.2497	-83.2496
Mean	-83.2291	-83.2389	-83.2460	-83.2469	-83.2450
Worst	-83.1479	-83.1133	-83.2253	-83.2259	-83.2207
SD	0.0265	0.0226	0.0051	0.0040	0.0066

Table. 4 Analysis of variance of the parameter p_{hm} from test problem 1

Source	DF	Seq SS	Adj SS	Adj MS	F	P
p_{hm}s	4	0.011048	0.0110482	0.0027621	10.59	0.000
Error	245	0.063896	0.0638962	0.0002608		
Total	249	0.074944				

Table 5. Comparison of the know global solution with the best AIS solution from test problem 1

GGP Solution	The Best AIS Solution
	$p_{hm} = 0.1$ $G_{\max} = 1000$
x_1 = 88.2890 x_2 = 7.7737 x_3 = 1.3120 $g_0(\mathbf{x})$ = -83.2535	x_1 = 88.35549473 x_2 = 7.673224397 x_3 = 1.315935336 $g_0(\mathbf{x})$ = -83.2497 CPU time = 4.406 sec.
GGP Constraints $g_m(\mathbf{x})$	AIS Constraints $g_m(\mathbf{x})$
$g_1(\mathbf{x}) = 1.000058 \le 1$	$g_1(\mathbf{x}) = 1.000000 \le 1$

Figure 6 plots the **Ab-Ag** affinity against the number of generations. It shows that the **Ab-Ag** affinity increases with the number of generations. The AIS converges after the 900 generations.

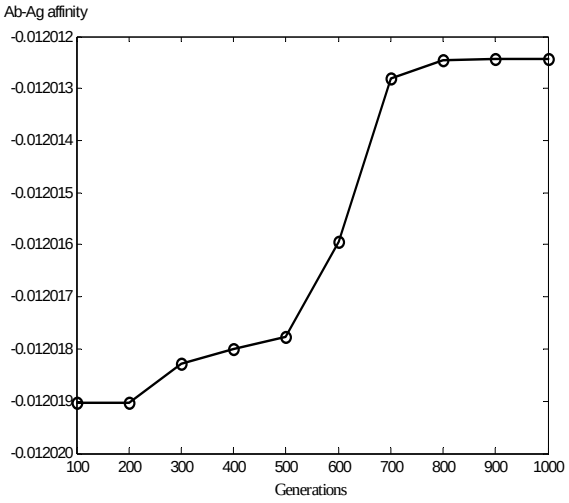

Figure 6. Ab-Ag affinity convergence through generations of test problem 1

6.2 Test Problem 2

$$\min \quad g_0(\mathbf{x}) = (-x_1 + 0.4x_1^{0.67}x_3^{-0.67})$$
$$s.t.\ g_1(\mathbf{x}) = 0.05882x_3x_4 + 0.1x_1 \le 1$$
$$g_2(\mathbf{x}) = 4x_2x_4^{-1} + 2x_2^{-0.71}x_4^{-1} + 0.05882x_2^{-1.3}x_3 \le 1$$
$$0.1 \le x_1, x_2, x_3, x_4 \le 10$$

Table 6 compares the known global solution with the best AIS solution, it demonstrating that the best AIS solution $g_0(\mathbf{x}) = -5.7398$ is the same as the known global solution; moreover, AIS solution yields a more precise constraint $g_2(\mathbf{x})$ than the known global solution. Furthermore, the AIS took 19.328 seconds of CPU computational time to solve the problem.

Table 6. Comparison of the know global solution with the best AIS solution from test problem 2

GGP Solution	The Best AIS Solution
	p_{hm} = 0.01 $G_{\max}$ = 3000
x_1 = 8.1267	x_1 = 8.12860927
x_2 = 0.6154	x_2 = 0.61371722
x_3 = 0.5650	x_3 = 0.56445552
x_4 = 5.6368	x_4 = 5.63650100
$g_0(\mathbf{x})$ = -5.7398	$g_0(\mathbf{x})$ = -5.7398
	CPU time = 19.328 sec.
GGP Constraints $g_m(\mathbf{x})$	AIS Constraints $g_m(\mathbf{x})$
$g_1(\mathbf{x})$ = 0.999999 ≤ 1	$g_1(\mathbf{x})$ = 0.999999 ≤ 1
$g_2(\mathbf{x})$ = 1.000007 ≤ 1	$g_2(\mathbf{x})$ = 0.999999 ≤ 1

6.3 Test Problem 3

$$\min \quad g_0(\mathbf{x}) = (5.3578x_3^2 + 0.8357x_1x_5 + 37.2392x_1)$$
$$s.t.\ g_1(\mathbf{x}) = 0.00002584x_3x_5 - 0.00006663x_2x_5 - 0.0000734x_1x_4 \le 1$$
$$g_2(\mathbf{x}) = 0.000853007x_2x_5 + 0.00009395x_1x_4 - 0.00033085x_3x_5 \le 1$$
$$g_3(\mathbf{x}) = 1330.3294x_2^{-1}x_5^{-1} - 0.42x_1x_5^{-1} - 0.30586x_2^{-1}x_3^2x_5^{-1} \le 1$$
$$g_4(\mathbf{x}) = 0.00024186x_2x_5 + 0.00010159x_1x_2 + 0.00007379x_3^2 \le 1$$
$$g_5(\mathbf{x}) = 2275.1327x_3^{-1}x_5^{-1} - 0.2668x_1x_5^{-1} - 0.40584x_4x_5^{-1} \le 1$$
$$g_6(\mathbf{x}) = 0.00029955x_3x_5 + 0.00007992x_1x_3 + 0.00012157x_3x_4 \le 1$$
$$78 \le x_1 \le 102, 33 \le x_2 \le 45, 27 \le x_3 \le 45,\ 27 \le x_4 \le 45, 27 \le x_5 \le 45$$

Table 7 lists the analytical results of the problem, showing that the AIS took 18.671 seconds to obtain the best solution. The best AIS solution $g_0(\mathbf{x}) = 10122.4782$ is better than the known global solution $g_0(\mathbf{x}) = 10122.6964$. The precision of the fifth constraint $g_5(\mathbf{x})$ is also higher in the AIS solution than in the known global solution.

Table 7. Comparison of the know global solution with the best AIS solution from test problem 3

GGP Solution	The Best AIS Solution
	p_{hm} = 0.01 $G_{\max}$ = 3000
x_1 = 78	x_1 = 78
x_2 = 33	x_2 = 33
x_3 = 29.998	x_3 = 29.99568144
x_4 = 45	x_4 = 44.99999986
x_5 = 36.7673	x_5 = 36.77538548
$g_0(\mathbf{x})$ = 10122.6964	$g_0(\mathbf{x})$ = 10122.4782
	CPU time = 18.671 sec.
GGP Constraints $g_m(\mathbf{x})$	AIS Constraints $g_m(\mathbf{x})$
$g_1(\mathbf{x})$ = -0.309978 ≤ 1	$g_1(\mathbf{x})$ = -0.309991 ≤ 1
$g_2(\mathbf{x})$ = 0.999826 ≤ 1	$g_2(\mathbf{x})$ = 1.000001 ≤ 1
$g_3(\mathbf{x})$ = -0.021419 ≤ 1	$g_3(\mathbf{x})$ = -0.021379 ≤ 1
$g_4(\mathbf{x})$ = 0.621349 ≤ 1	$g_4(\mathbf{x})$ = 0.621402 ≤ 1
$g_5(\mathbf{x})$ = 1.000062 ≤ 1	$g_5(\mathbf{x})$ = 1.000002 ≤ 1
$g_6(\mathbf{x})$ = 0.681496 ≤ 1	$g_6(\mathbf{x})$ = 0.681516 ≤ 1

6.4 Summary of Results

The performance of the proposed AIS is summarized as follows:

(1) Effectiveness: The AIS can yield a potential global solution that each constraint is accurate up to five decimal places (as shown in Tables 5-7).

(2) Efficiency: The AIS requires only a short CPU computational time (as shown in Tables 5-7).

(3) Simplicity: The AIS implementation is much simpler than most conventional GGP algorithms, since it has no complex differential calculation.

7. CONCLUSIONS AND FUTURE WORK

This study presented an AIS that involves several operators, inspired by bio-immune metaphor. The performance of the proposed AIS was measured by testing GGP problems. Numerical results indicate that the proposed method worked well. Future work should solve high-dimensional GGP problems and compare the computed results with those obtained using approaches that involve GAs and simulated annealing.

8. REFERENCES

[1] Dasgupta, D., and Attoh-Okine, N. Immunity-based systems: a survey. In IEEE international conference on System, men, and cybernetics (Orlando, FL USA, October 12-15, 1997). 1997, 369-374.

[2] de Castro, L. N., and Timmis, J. An artificial immune network for multimodal function optimization. In Proceedings of the congress on Evolutionary computation (Honolulu, HI USA, May 12-17, 2002). 2002, 699-704.

[3] Coello Coello, C. A., and Cruz Cortes, N. A parallel implementation of an artificial immune system to handle constraints in genetic algorithms: preliminary results. In Proceedings of the congress on Evolutionary computation (Honolulu, HI USA, May 12-17, 2002). 2002, 819-824.

[4] de Castro, L. N., and Von Zuben, F. J. Learning and optimization using the clonal selection principle. IEEE Transactions on Evolutionary Computation, 6, 3 (June 2002), 239-251.

[5] Duffin, R. J., Peterson, E. L., and Zener, C. *Geometric Programming*. Wiley, NY, 1967.

[6] Maranas, C. D., and Floudas, C. A. Global optimization in generalized geometric programming. Computers and Chemical Engineering, 21, 4 (1997), 351-369.

[7] Floudas, C. A., Pardalos, P. M., et al., *Handbook of Test Problems in Local and Global Optimization*. Kluwer, Boston, 1999.

[8] Coico, R., Sunshine, G., and Benjamini, E. *Immunology - A Short Course*. Wiley, NJ, 2003.

[9] Parkin, J., and Cohen, B. An overview of the immune system. The Lancet, 357, 9270 (June 2001), 1777-1789.

[10] Jerne, N. K. Idiotypic networks and other preconceived ideas. Immunological Reviews, 79, 1984, 5-24.

[11] Perelson, A. S. Immune network theory. Immunological Reviews, 110, 1989, 5-36.

[12] George, A. J. T., and Gray, D. Receptor editing during affinity maturation. Immunology Today, 20, 4 (April 1999), 196.

[13] Kouskoff, V., and Nemazee, D. Role of receptor and revision in shaping the B and T lymphocyte repertoire. Life Sciences, 69, 10 (July 2001), 1105-1113.

[14] Passy, U. Generalized weighted mean programming. SIAM Journal on Applied Mathematics, 20, 4 (June 1971), 763-778.

[15] Passy, U., and Wilde, D. J. Generalized polynomial optimization. SIAM Journal on Applied Mathematics, 15, 5 (September 1967), 1344-1356.

[16] Coello Coello, C. A. Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art. Computer Methods in Applied Mechanics and Engineering. 191, 11-12 (January 2002), 1245-1287.

[17] Michalewicz, Z., *Genetic Algorithms + Data Structures = Evolution Programs*. Springer, NY, 1999.

[18] de Castro, L. N., and Timmis, J., Artificial Immune Systems: A New Computational Intelligence Approach. Springer, NY, 2002.