Studies on Global Optimization Approach for General Variational Inequality Problems

Mend-Amar MAJIG

Studies on Global Optimization Approach for General Variational Inequality Problems

Mend-Amar MAJIG

Submitted in partial fulfillment of

the requirement for the degree of

Doctor of Informatics

Kyoto University
Kyoto, Japan
January 2009

Preface

In this thesis, we develop several methods for solving the general (not necessarily monotone) variational inequality problem. The variational inequality problem, which may be regarded as an extension of the nonlinear programming problem, has now become one of the core subjects in optimization, and indeed it covers various classes of optimization problems including the nonlinear system of equations and equilibrium problems of many kinds. Besides its use as these problems, the variational inequality problem has a wide range of important application fields in engineering and economics.

Since its emergence in 1960s, the variational inequality problem has been widely explored by mathematicians, engineers and economists. Although there have been plenty of methods proposed for solving the variational inequality problem, most of them restrict themselves by imposing some assumptions such as monotonicity on the problem. For the general variational inequality problem, there are very few methods, and especially methods implementable in practice are even fewer.

The relationship between the variational inequality problem and global optimization was first pointed out by Auslender in 1976 when he introduced the gap function. Since then several other global optimization reformulations of the variational inequality problem have been discovered and all of them possess the same important property that the global minimum value of the equivalent problem is zero provided the variational inequality problem has a solution. Moreover, the advances in computer technology and the emergence of the heuristic methods make it possible to employ global optimization methods which are traditionally considered very expensive for solving the variational inequality problem.

The main contribution of this thesis is to investigate the possibility of using global optimization methods for solving the general variational inequality problem. We develop methods, from deterministic to heuristic, for directly solving the equivalent global optimization problems. Moreover, we propose a modified and practically implementable version of the classical fast convergent Josephy-Newton method and employ a heuristic method as a subproblem solver.

Another important contribution is to develop a heuristic method that is capable of detecting as many as possible, hopefully all, solutions of the general variational inequality problem. The adaptive fitness function techniques for the evolutionary algorithm is introduced as a part of this method.

Kyoto, Japan Mend-Amar Majig
January 2009

Acknowledgements

I would like to express my deepest gratitude to my supervisor, Professor Masao Fukushima of Kyoto University, one of the greatest scientists in optimization and a truly wonderful person, for many things. I thank him for accepting me to be one of his students, for guiding my research, and for reading and correcting my draft manuscripts carefully. Although I may have often troubled him, he is always very understanding and encouraging. His sincere and caring attitude toward his students is so admirable and it makes me want to be a person like him. Without his continual guidance and persistent support, this dissertation would not have been possible.

I would like to extend my appreciation to my former supervisor Professor Enkhbat Rentsen of National University of Mongolia. In fact, it was his inspiration that intrigued me to study optimization several years ago, and during my stay in Kyoto University he continuously supported and advised me through his encouraging e-mails.

I would like to thank Associate Professor Nobuo Yamashita of Kyoto University. His insight to the optimization field is remarkable and the suggestions he gave me always led to better results in my research.

My especial thanks to Mrs Fumie Yagura of Fukushima Laboratory, who always took care of me and helped me to handle all my faculty related documents.

I would also like to thank all members of Fukushima Laboratory, including Assistant Professor Shunsuke Hayashi, former and current students, for a friendly and warm environment they offered me. I have received a great deal of help from them and it made my life in Japan easier. Indeed, I am really proud of being a member of Fukushima Laboratory of Kyoto University.

I am particularly thankful to Japanese Ministry of Education, Culture, Sports, Science and Technology and Kyoto University for supporting me with scholarships and facilities during my study in Japan.

Finally, I thank my family, parents, a brother and a sister, for their everlasting love and relentless support.

Contents

Chapter 1

Introduction

1.1 Variational Inequality Problem

Let S be a nonempty closed convex set in $\Re^n$ and F be a continuous mapping from $\Re^n$ into itself. The *variational inequality problem* (VIP for short) is to find a vector $x^* \in S$ such that

$$\langle F(x^*), x - x^* \rangle \geq 0, \quad \forall x \in S, \tag{1.1.1}$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product in $\Re^n$. This problem is denoted by VI(S, F) and its solution set is denoted by SOL(S, F).

When $S \equiv \Re^n$, (1.1.1) reduces to the following system of nonlinear equations:

$$F(x) = 0. \tag{1.1.2}$$

When F is a gradient of some differentiable function f, i.e., $F(x) \equiv \nabla f(x)$, then (1.1.1) represents the optimality condition of the following optimization problem:

$$\min f(x) \text{ subject to } x \in S. \tag{1.1.3}$$

Besides being an extension to the nonlinear programming problem and the system of nonlinear equations, the variational inequality problem is a host to various types of equilibrium problems and has a great deal of application fields [8, 17, 33, 54, 69, 73, 78, 98].

Theoretical aspects of the VIP have been studied extensively [2, 3, 6, 9, 10, 11, 12, 25, 32, 35, 44, 45, 46, 47, 49, 50, 57, 59, 65, 66, 67, 76, 77, 79, 80, 84, 85, 92, 93, 94, 95, 99] since its emergence and many algorithms, such as projection methods, interior and smoothing methods and equation reduction methods, have been proposed to solve it [11, 18, 19, 20, 23, 32, 39, 68, 70, 71, 72, 74, 83, 101]. However, the validity and efficiency of those algorithms often depend on the monotonicity-like assumption on the mapping F.

Although some algorithms have been proposed for solving general (not necessarily monotone) VIPs [1, 40, 41, 48, 83, 88, 96], they are primarily designed to solve the particular VI(S, F) in which the constraint set S is the non-negative orthant $\Re^n_+$. This special problem, VI$(\Re^n_+, F)$, is called the *nonlinear complementarity problem*, and is denoted NCP(F). Some methods [40, 41, 88] may be applied to the general VIP by reformulating the problem as a complementarity problem through its KKT system. But those approaches increase the dimension of the problem by introducing Lagrangian multipliers.

Among the methods for solving the VIP, a popular idea is to reformulate the VIP as some well known class of problems such as nonlinear systems of equations or optimization problems and deal with their equivalent problems by using classical methods [22, 23, 35, 40, 41, 42, 43, 66, 72, 83, 84, 85, 97, 102, 106, 107]. It has been known for long that the VIP can be reformulated as an optimization problem with zero global minimum value. But global optimization methods have been considered computationally very expensive and little attention has been paid to using global optimization approaches for solving VIPs.

The advances in computer technology and the emergence of meta-heuristic methods [7, 13, 28, 51, 52] in global optimization make us reconsider the use of global optimization methods for solving VIPs. Solving the global optimization problem derived from a VIP has several advantages.

Firstly, since a global solution of the equivalent optimization problem will always be the true solution to VIP, we do not need any extra assumptions as most of the VIP methods require. In this way we can deal with the VIP under a very general setting.

Secondly, the known global minimum value of the equivalent problem is particularly helpful especially when we use heuristic approaches. Due to the absence of optimality conditions for general global minima, heuristic methods for global optimization usually use some upper limit for computational expenses as a stopping condition. Since the global minimum value is known to be zero, we can use this information in the stopping condition.

Moreover, heuristic methods of global optimization will give us opportunity to search with much more flexibility. As a result we may develop a method for finding as many as possible solutions of the general VIP, which is one of the first methods of this kind.

1.2 Global Optimization Problem

For a given set S and a continuous function f, *the global optimization problem* is to find $x^* \in S$ such that

$$f(x^*) \leq f(x), \ \forall x \in S. \tag{1.2.1}$$

The point x^* satisfying (1.2.1) is called a global minimum for f over the set S, and $f(x^*)$ is called the global minimum value.

The global optimization problem is one of the fundamental subjects in optimization and its practical applications are diverse [37]. In the presence of multiple non-global solutions, the classical optimization methods are not guaranteed to find the global solutions. The methods for solving global optimization problems fall into three categories; deterministic methods, stochastic methods and heuristic methods [37, 13, 75, 87, 89, 90].

The deterministic methods, including the branch and bound methods, the cutting plane methods, have a theoretical guarantee of finding the global solution. But most of the deterministic methods are proposed for particular classes of the problems and take advantage of the special structure of the problems. Besides, the deterministic methods are considered very expensive.

The stochastic methods, including the simulated annealing and the Monte-Carlo sampling, search the solution randomly and there is no guarantee that the algorithm can find the solution in a finite number of steps. Nevertheless, for some type of problems the stochastic methods work better than the deterministic methods.

The advances in the computer technology in last two decades make it possible to solve global optimization problems once considered very hard. Meantime, the introduction of meta-heuristics for global optimization has brought us to a new era in global optimization solver [13, 28, 51]. The heuristic methods search global solutions in an intelligent way. Although there is no guarantee for the convergence of the method to a solution, heuristic methods work very well in practice, and very often prevail the other two. Methods in meta-heuristics are flexible, and they can deal with very general global optimization problems.

The main drawback of stochastic and heuristic methods is that they suffer from the difficulty in determining stopping conditions, and they usually use some upper limits on computational expenses. But when we use heuristic methods to solve an equivalent optimization problem of the VIP, we have a natural stopping condition based on the fact that the global minimum value is zero.

In the thesis, we concentrate on using global optimization methods and techniques for solving the general variational inequality problem. Using a global optimization approach for solving VIPs has several advantages.

First of all, the global minimum value of the equivalent problems is known to be zero beforehand. So we can get rid of the difficulty that most global optimization problems face in determining when to stop.

Secondly, when we try to obtain not just one solution, but multiple solutions, the zero global minimum value is very important. The adaptive fitness function technique we develop has the property that when we construct the merit function by using some adaptive functions, the global minimum value will remain intact because of the zero global minimum value.

We will employ both deterministic and heuristics methods for solving the equivalent global optimization problems of VIPs.

1.3 Outline of the Thesis

In the subsequent chapters, we explore the basic properties of the VIP and its relationship with global optimization, and develop several methods for solving the general VIP. Below we summarize the main contributions in each chapter.

In Chapter 2, we give a brief introduction to the VIP, its solution analysis, and its relation to a global optimization problem. Equivalent unconstrained global optimization problems based on merit functions will be discussed in detail.

In Chapter 3, we present a heuristic method to find as many as possible solutions of the general VIP. We propose a hybrid evolutionary algorithm that incorporates local search in promising regions. In order to prevent searching process from returning to the already detected global or local solutions, we employ some adaptive fitness function techniques.

In Chapter 4, we propose a new practical version of globally convergent Josephy-Newton based method for VIPs. By means of some additional bound constraint, the method ensures existence of solutions to subproblems, which is an essential property not enjoyed by the classical Josephy-Newton method. Under appropriate conditions, global and locally superlinear convergence properties of the proposed method are established. Furthermore, we develop an evolutionary algorithm for solving general VIPs with bounded polyhedral constraints, which can be used to solve the modified Josephy-Newton subproblems when the constraint set is polyhedral.

In Chapter 5, we present a deterministic approach for solving VIPs. Under some suitable assumptions, the Lipschitz continuity for merit functions is shown. Then the classical Lipschitzian branch and bound method is applied to solve merit function minimization problems.

In Chapter 6, we will summarize the findings in the thesis and discuss some possible future directions.

Chapter 2

Preliminaries

2.1 Notations

First we introduce some basic notations which will be used in the subsequent chapters.

$\Re^n$ is the n dimensional Euclidean space.
$\|\cdot\|$ is the Euclidean norm defined by

$$\|x\| := \sqrt{x_1^2 + x_2^2 + \cdots + x_n^2}, \text{ for } x = (x_1, x_2, \ldots, x_n).$$

$\|\cdot\|_\infty$ is the l_∞ norm defined by

$$\|x\|_\infty := \max_{1 \le i \le n} |x_i|, \text{ for } x = (x_1, x_2, \ldots, x_n).$$

$B(\bar{x}, \delta)$ is the closed ball with center $\bar{x}$ and radius δ defined by

$$B(\bar{x}, \delta) := \{x \in \Re^n |\ \|x - \bar{x}\| \le \delta\}.$$

The notation $x \perp y$ means vector x is perpendicular to vector y.
An $n \times n$ matrix G is called positive definite, if and only if

$$\langle x, Gx \rangle > 0,\ \forall x \in \Re^n,\ x \neq 0.$$

For a given symmetric, positive definite matrix G, $\|\cdot\|_G$ is the G norm defined by

$$\|x\|_G := \langle x, Gx \rangle.$$

For a given closed convex set $S \subseteq \Re^n$, a vector $x \in \Re^n$ and a symmetric positive definite matrix G, the projection of x onto S under the G norm, denoted $\Pi_{S,G}(x)$, is defined as the unique solution of the following strictly convex minimization problem

$$\begin{aligned}&\text{minimize } \tfrac{1}{2}\|y-x\|_G^2\\&\text{subject to } y \in S.\end{aligned}$$

When G is the identity matrix, i.e., $G \equiv I$, $\Pi_{S,G}(x)$ is called the Euclidean projection and denoted simply by $\Pi_S(x)$.
For a differentiable function $f : \Re^n \to \Re$, $\nabla f(x)$ denotes its gradient at point x defined by

$$\nabla f(x) := \begin{pmatrix} \dfrac{\partial f(x)}{\partial x_1} \\ \vdots \\ \dfrac{\partial f(x)}{\partial x_n} \end{pmatrix},$$

where $\dfrac{\partial f(x)}{\partial x_i}$ is the partial derivative of f at x associated with its i-th component.
For a differentiable vector valued function $F : \Re^n \to \Re^m$, $\nabla F(x)$ denotes its transposed Jacobian matrix at point x defined by

$$\nabla F(x) := (\nabla F_1(x) \dots \nabla F_n(x)) = \begin{pmatrix} \dfrac{\partial F_1(x)}{\partial x_1} & \dots & \dfrac{\partial F_m(x)}{\partial x_1} \\ \vdots & \ddots & \vdots \\ \dfrac{\partial F_1(x)}{\partial x_n} & \dots & \dfrac{\partial F_m(x)}{\partial x_n} \end{pmatrix}.$$

2.2 Solution Analysis of VIPs

The most fundamental result on the existence of a solution to the VI(S, F) requires the compactness and convexity of the set S and continuity of the mapping F.

Theorem 2.2.1. *[15, 34] Let $S \subseteq \Re^n$ be a nonempty and convex set and let $F : S \to \Re^n$ be a continuous mapping. If S is compact, then there exists a solution to the problem VI(S, F).*

From this theorem, many other results can be derived by replacing the compactness of S with some additional conditions on F. Among these additional conditions, the monotonicity plays a crucial role. Below we introduce the definition of the monotonicity and its versions.

Definition 2.2.2. *[82] The mapping $F : \Re^n \to \Re^n$ is said to be*

(a) monotone over a set S if

$$\langle F(x) - F(y), x - y\rangle \geq 0, \ \forall x, y \in S; \tag{2.2.1}$$

(b) strictly monotone over S if

$$\langle F(x) - F(y), x - y\rangle > 0, \ \forall x, y \in S, \ x \neq y; \tag{2.2.2}$$

(c) strongly monotone over S if there exists an $\alpha > 0$ such that

$$\langle F(x) - F(y), x - y\rangle \geq \alpha\|x - y\|^2, \ \forall x, y \in S; \tag{2.2.3}$$

(d) coercive with respect to S if there exists a vector $x^0 \in S$ such that

$$\lim_{x\in S, \ \|x\|\to\infty} \frac{\langle F(x), x - x^0\rangle}{\|x\|} = \infty. \tag{2.2.4}$$

The concept of monotonicity is closely related with the concept of convexity. In fact, when F is a gradient mapping of a real-valued function f, i.e., $F(x) = \nabla f(x), \ \forall x \in S$, then the mapping F is monotone on S if and only if f is a convex function on S.

In general, the variational inequality problem may have more than one solution, but under the strict monotonicity assumption on F, there cannot be more than one solution.

Proposition 2.2.3. *[32] If F is strictly monotone on S, then the problem VI(S, F) has at most one solution.*

Note that the strict monotonicity assumption does not guarantee the existence of the solution to VI(S, F). The following two theorems show that under a bit stronger conditions, the existence is guaranteed.

Theorem 2.2.4. *[34, 80] Let $S \subseteq \Re^n$ be a nonempty, closed, convex set and let $F : S \to \Re^n$ be a continuous mapping. If F is coercive with respect to S, then the problem VI(S, F) has a nonempty compact solution set.*

The following theorem suggests the uniqueness of the solution.

Theorem 2.2.5. *[34, 80] Let $S \subseteq \Re^n$ be a nonempty, closed, convex set and let $F : S \to \Re^n$ be a continuous mapping. If F is strongly monotone with respect to S, then there exists a unique solution to the problem VI(S, F).*

Next we consider the concept of an isolated solution.

Definition 2.2.6. *[104] A solution x^* of the variational inequality problem VI (S, F) is isolated if there exists a neighborhood U of x^* such that x^* is the only solution of the VI(S, F) in U.*

The following result gives a sufficient condition for a solution of the problem VI(S, F) to be isolated.

Proposition 2.2.7. *[104] Let $F : \Re^n \to \Re^n$ be a continuously differentiable mapping and let x^* be a solution to the problem VI(S, F). If the Jacobian matrix $\nabla F(x^*)$ is positive definite, then x^* is isolated.*

We close this subsection with some discussion about the KKT system of the variational inequality problem. We consider the case where the constraint set S is represented by finitely many differentiable inequalities and equations:

$$S := \{x \in \Re^n |\ h(x) = 0,\ g(x) \leq 0\}, \tag{2.2.5}$$

with $h : \Re^n \to \Re^l$ is an affine function and $g : \Re^n \to \Re^m$ is a vector-valued continuously differentiable convex function. When we consider the KKT system for an optimization problem, a condition so called constraint qualification on the constraint set is crucial for the validity of the KKT system. One of the most popular constraint qualifications is Slater's constraint qualification. We say that the generalized Slater's constraint qualification holds for the set S given by (2.2.5) if there exists a $\bar{x} \in S$ such that $g(\bar{x}) < 0$.

Proposition 2.2.8. *[17] Let S be given by (2.2.5) where the function $h := (h_1, \ldots, h_l)$ is affine and the function $g := (g_1, \ldots, g_m)$ is continuously differentiable and convex. Let F be a mapping from S into $\Re^n$. The following two statements are valid.*

(a) Let x be a solution to VI(S, F). If the generalized Slater's constraint qualification holds for the set S, then there exist vectors $\mu \in \Re^l$ and $\lambda \in \Re^m$ such that

$$\begin{aligned} 0 &= F(x) + \sum_{j=1}^{l} \mu_j \nabla h_j(x) + \sum_{i=1}^{m} \lambda_i \nabla g_i(x) \\ 0 &= h(x) \\ 0 &\leq \lambda \perp g(x) \leq 0. \end{aligned} \tag{2.2.6}$$

(b) Conversely, if (x, μ, λ) satisfies (2.2.6), then x solves the VI (S, F).

This KKT system itself may be regarded as a box constrained variational inequality problem. Hence, instead of solving the original VIP with general convex constraints, we can deal with a box constrained VIP with higher dimension.

2.3 Equivalent Reformulations of VIPs

2.3.1 Equation Reformulations

The first two reformulations are based on the projection operator. Let us define the following two mapping:

- *the natural mapping -* $F_S^{\text{nat}}(v) := v - \Pi_S(v - F(v))$,
- *the normal mapping -* $F_S^{\text{nor}}(v) := F(\Pi_S(v)) + v - \Pi_S(v)$.

The following proposition will give us equivalent reformulations of VIP as systems of equations.

Proposition 2.3.1. *[92, 93, 94] Let $S \subseteq \Re^n$ be a closed convex set and let $F : S \to \Re^n$ be any mapping. Then x is a solution to VI(S, F) if and only if any of the following two statements holds:*

(a) x satisfies

$$F_S^{nat}(x) = 0; \tag{2.3.1}$$

(b) there exists a vector z such that $x = \Pi_S(z)$ and

$$F_S^{nor}(z) = 0. \tag{2.3.2}$$

The next reformulation is based on the KKT system of the VIP. We have seen in the previous section that, under some suitable condition, for a finitely representable set

$$S := \{x \in \Re^n |\ h(x) = 0,\ g(x) \le 0\},$$

if x is a solution to VI(S, F), it satisfies with some $(\mu, \lambda) \in \Re^l \times \Re^m$ the following KKT system

$$\begin{array}{rcl} 0 & = & F(x) + \sum\limits_{j=1}^{l} \mu_j \nabla h_j(x) + \sum\limits_{i=1}^{m} \lambda_i \nabla g_i(x) \\ 0 & = & h(x) \\ 0 & \le & \lambda \ \perp \ g(x) \le 0. \end{array} \tag{2.3.3}$$

By using the so-called complementarity function, or C function for short, we can reformulate this system as a nonlinear system of equations.

Definition 2.3.2. *A function $\psi : \Re^2 \to \Re$ is called a C-function, if for any pair $(a, b) \in \Re^2$,*

$$\psi(a, b) = 0 \Longleftrightarrow (a, b) \ge 0 \textit{ and } ab = 0.$$

With the Fischer-Burmeister C function defined by $\psi_{\mathrm{FB}}(a, b) := \sqrt{a^2 + b^2} - a - b$, we can redefine the KKT system (2.3.3) as the following system of nonlinear equations:

$$\left(\begin{array}{c} F(x) + \sum\limits_{j=1}^{l} \mu_j \nabla h_j(x) + \sum\limits_{i=1}^{m} \lambda_i \nabla g_i(x) \\ h(x) \\ \psi_{\mathrm{FB}}(g(x), \lambda) \end{array} \right) = 0, \tag{2.3.4}$$

where $\psi_{\mathrm{FB}}(g(x), \lambda) = (\psi_{\mathrm{FB}}(g_1(x), \lambda_1), \dots, \psi_{\mathrm{FB}}(g_m(x), \lambda_m))^T$.

2.3.2 Merit Functions

Definition 2.3.3. *[17, 22] A merit function for the VI (S, F) on a closed set $X \supseteq S$ is a nonnegative function $\theta : X \to \Re_+$ such that $x^* \in$ SOL(S, F) if and only if $x^* \in X$ and $\theta(x^*) = 0$, that is, if and only if the solutions of the VI (S, F) coincide with the global solutions of the problem*

$$\textit{minimize } \theta(x)$$

$$\textit{subject to } x \in X$$

and the optimal objective value of this problem is zero.

When SOL(S, F) is empty, then either the global optimal value of θ over X is positive or θ has no global minima on X. Below we consider some well known merit functions.

Gap function [3, 35]
The gap function is defined by

$$\theta_{\text{gap}}(x) = \sup_{y \in S} \langle F(x), x - y \rangle. \tag{2.3.5}$$

This function is considered the first of the merit functions and several other merit functions have been created by modifying this function [22, 23, 43]. It is not difficult to see that $\theta_{\text{gap}}(x) \geq 0,\ \forall x \in S$, and that $x^* \in \text{SOL}(S, F)$ if and only if $\theta(x^*) = 0$. Thus θ_{gap} is a merit function for the VI (S, F) on S.

To evaluate a gap function value, we have to maximize a linear function over the set S. Unless S is a polyhedral set, this is not a trivial task. In general, the gap function is not differentiable.

The next function we consider is the regularized gap function which is a modification of the gap function.

Regularized gap function [23]
The regularized gap function is defined by

$$\theta_c(x) = \sup_{y \in S} \{\langle F(x), x - y \rangle - \frac{c}{2} \langle x - y, G(x - y) \rangle\}, \tag{2.3.6}$$

where G is a symmetric, positive definite matrix and c is a positive scalar. To evaluate a regularized gap function value we have to solve the following convex programming problem.

$$\begin{aligned} &\text{maximize } \langle F(x), x - y \rangle - \frac{c}{2} \langle x - y, G(x - y) \rangle \\ &\text{subject to } y \in S. \end{aligned} \tag{2.3.7}$$

The objective function in (2.3.7) is strongly concave, so the regularized gap function is well defined, and from the definition of the function it is seen that

$$\theta_b(x) \leq \theta_a(x),\ \forall x \in \Re^n,$$

for any two scalars $b > a > 0$. Moreover, the regularized gap function is differentiable, whenever so is F, and its gradient is given by

$$\nabla \theta_c(x) = F(x) + (\nabla F(x)^T - cG)^T (x - \Pi_{S,G}(x - c^{-1} G^{-1} F(x))).$$

It is important to note that the nonnegativity of the regularized gap function θ_c is valid only on S; moreover, in order for a zero of θ_c to be a solution of the VI (S, F), it is essential that such a zero belongs to S. The constraint $x \in S$ is needed in order to ensure that a global minimizer of the program with zero objective value solves the VI (S, F).

Linearized gap function [102]
One of the drawbacks of the regularized gap function is that the evaluation of this function requires a calculation of a projection onto the set S, which cannot be accomplished by a finite procedure, unless S is polyhedral. However, in the case of a finitely representable set S, we can consider a less expensive merit function. Assume that

$$S := \{x \in \Re^n | g_i(x) \leq 0, \; i = 1, \ldots, m\}, \tag{2.3.8}$$

where each $g_j, \; i = 1, \ldots, m$ is a convex and continuously differentiable function. Linear equality constraints can be included without difficulty, but are omitted for simplicity. We define the polyhedral approximation to S at the point x by

$$\Gamma(x) := \{y \in \Re^n | \; g_i(x) + \langle \nabla g_i(x), y - x\rangle \leq 0, \; i = 1, \ldots, m\},$$

and define the linearized gap function $\theta_c^{\text{lin}}(x)$ by

$$\theta_c^{\text{lin}}(x) = \sup_{y \in \Gamma(x)} \{\langle F(x), x - y\rangle - \frac{c}{2}\langle x - y, G(x - y)\rangle\}. \tag{2.3.9}$$

Since $S \subseteq \Gamma(x), \; \forall x \in \Re^n$,

$$\theta_c^{\text{lin}}(x) \geq \theta_c(x), \; \forall x \in \Re^n,$$

and the following proposition shows that under some suitable constraint qualifications for S, $\theta_c^{\text{lin}}(x)$ is indeed a merit function for VI(S, F).

Proposition 2.3.4. *[102] Let S be defined by (2.3.8), where each $g_i, \; i = 1, \ldots, m$ is a convex and continuously differentiable function. Let $c > 0$ be a given scalar and G be a given symmetric positive definite matrix. If $x \in S$ and $\theta_c^{lin}(x) = 0$, then $x \in$SOL(S, F). Conversely, if $x \in$ SOL(S, F) and the generalized Slater's constraint qualification holds for S, then $\theta_c^{lin}(x) = 0$.*

The evaluation of the linearized gap function requires solving a convex quadratic programming problem. Although this function is not differentiable, under some suitable condition, it can be shown that the function is directionally differentiable for every direction.

2.3.3 Unconstrained Global Optimization Reformulations

In this subsection, we will consider several unconstrained global optimization reformulations to VIPs.

Equation based reformulations

The equation reformulations we have considered in the previous sections directly lead us to some unconstrained global optimization reformulations of VIPs.
Suppose that for the VI(S, F) we have the following equation reformulation:

$$H(x) = 0. \tag{2.3.10}$$

If we consider the function

$$\theta(x) := \|H(x)\|^{\alpha}, \tag{2.3.11}$$

where α is any positive integer, then this is an unconstrained merit function for the VIP, i.e.,

$$\min \theta(x) \text{ subject to } x \in \Re^n, \tag{2.3.12}$$

whose global minimum value is zero. In the special case where $H(x) = F^{\text{nat}}(x)$ and $\alpha = 1$, the function θ is called the natural residual function and denoted $\theta_{\text{nat}}(x)$, i.e.,

$$\theta_{\text{nat}}(x) := \|F^{\text{nat}}(x)\| = \|x - \Pi_S(x - F(x))\|.$$

Penalty based reformulation

The merit functions we have considered so far are all parts of some constrained global optimization problems

$$\min \theta(x) \;\text{ subject to } x \in S. \tag{2.3.13}$$

For a finitely representable set $S := \{x \in \Re^n |\; h(x) = 0,\; g(x) \leq 0\}$, we can consider the following penalty based reformulation for the variational inequality problem. Let us define the penalty function for problem (2.3.13) by

$$p(x) := \theta(x) + \rho \max\{g_1(x), \ldots, g_m(x), |h_1(x)|, |h_2(x)|, \ldots, |h_l(x)|\}. \tag{2.3.14}$$

The following theorem shows that under some assumptions, the function (2.3.14) can serve as an unconstrained merit function for VI(S, F).

Theorem 2.3.5. *[5] Under some suitable constraint qualifications, there exists a $\rho > 0$ for which the solution sets of the problem*

$$\textit{min } p(x) \textit{ subject to } x \in \Re^n \tag{2.3.15}$$

and problem (2.3.13) *are identical.*

Note that the global minimum value of the problem (2.3.15) for an appropriate ρ is again zero.

The D-gap merit functions [106]
The D-gap function is defined by

$$\theta_{ab}(x) := \theta_a(x) - \theta_b(x),\ \forall x \in \Re^n, \tag{2.3.16}$$

where a and b are given scalars satisfying $b > a > 0$.

The following theorem shows that the D-gap function is an unconstrained merit function of the VI (S, F).

Theorem 2.3.6. *[43, 106] Let $F : \Re^n \to \Re^n$ be continuous and S be a closed convex subset of $\Re^n$. For $b > a > 0$, the D-gap function θ_{ab} is continuous on $\Re^n$ and*

(a) $\theta_{ab}(x) \geq 0$ for all x in $\Re^n$;

(b) $\theta_{ab}(x) = 0$ if and only if $x \in$ SOL(S, F);

(c) if F is continuously differentiable on $\Re^n$, then so is the D-gap function θ_{ab}.

As an alternative to this D-gap function, we may also consider the linearized D-gap function

$$\theta_{ab}^{\text{lin}}(x) = \theta_a^{\text{lin}}(x) - \theta_b^{\text{lin}}(x),$$

which remains directionally differentiable under some suitable condition. It is not difficult to see that this function is indeed a merit function. Evaluation of the D-gap function requires solving two convex programming problem, while that of the linearized D-gap function requires solving two convex quadratic programming problem. With either the D-gap function or the linearized D-gap function, we can reformulate the VI(S, F) as an unconstrained global optimization problem

$$\text{minimize } \theta_{ab}(x) \text{ subject to } x \in \Re^n \tag{2.3.17}$$

or

$$\text{minimize } \theta_{ab}^{lin}(x) \text{ subject to } x \in \Re^n \tag{2.3.18}$$

whose global minimum value is zero.

All the unconstrained global optimization reformulations we have considered result in the global optimization problems whose global minimum values are known to be zero. This is the motivation of our global optimization approach to VIPs.

In the following three sections, we develop three methods for solving general VIPs. All of them are based on solving an equivalent global optimization problem we have just considered. In two of them, we directly solve the equivalent global optimization problem for VIPs. The other one solves the equivalent global optimization problem as a subproblem of a particular fast convergent method.

Chapter 3

Hybrid Evolutionary Algorithm for VIPs

3.1 Introduction

In this chapter, we propose a heuristic method for finding as many as possible, hopefully all, solutions of the general VIP. To achieve this, we first use an optimization reformulation of the VIP based either on a merit function or on its KKT system. In either case, the VIP is reformulated as the following box constrained optimization problem with *zero global minimum value* [17, 22]:

$$\min\ f(x) \quad \text{s.t. } x \in D, \tag{3.1.1}$$

where f is a real-valued function and the set D is defined as $D = \{x \in R^n |\ l \leq x \leq u\}$. Here $l, u \in R^n \cup \{\pm\infty\}$ are, possibly infinite, lower and upper bounds on the variable.

In order to find all global solutions of problem (3.1.1), we propose a population-based *hybrid evolutionary algorithm* (HEA) that incorporates local search in promising regions. The proposed method tries to keep and improve diversity of good trial points in the population set while searching for global minimizers of the objective function. Moreover, every time a global or local solution, or an unpromising trial point is detected by the HEA, the objective function of the problem is locally modified around this point to prevent the searching process from returning back to the vicinity of this solution again. Actually, the proposed HEA invokes some known strategies of hybrid metaheuristics [51, 89, 103] with some modifications to fit the general VIP.

The tunneling function method for finding a global minimum of a non-convex function was first introduced in [55]. The main idea of this method is that every time a local solution is detected in computation, it tries to construct a new objective function which has the same global minima as the original function but the detected local minimum is no longer a local minimum for the new function. The new function is treated as the objective function in the next search stage, and this process is repeated until a global solution is found. Another idea to escape from a detected local minimum, called the filled function method, is proposed in [26, 27, 105]. Instead of constructing a tunnel at a detected local minimum, it considers a new objective function which has a local maximum at the detected local solution and has no stationary point at local solutions worse (having greater original objective function value) than the detected one. Both tunneling and filled function techniques are applied to the general nonlinear complementarity problem in [40], where a semi-smooth Newton method is presented.

In our method, the tunneling function technique is used not only for escaping from the detected local minimum, but also and more importantly for escaping from a detected global minimum and its basin to search for other global minimizers. However, the direct use of the tunneling function technique at a detected *global* minimum can be effective only if it is an exact solution of the problem. In practice, we can only expect to find approximations of global minima. To cope with this difficulty, before using the tunneling modification at a detected approximate global minimum, we suggest first to use another modification of the objective function, which is called a *hump* function and helps capture the exact global minimum near the detected approximate solution, and then construct a *hump-tunneling* function to which the HEA is applied.

The global optimal value of problem (3.1.1) is known to be zero. The proposed method HEA exploits this fact in two ways. First, it helps the HEA to determine whether a solution is global or not. Second and more importantly, if a modified objective function (either by a tunneling or by a hump-tunneling function) has at least one common global minimum with the original objective function, it must also have the zero global minimum value, i.e., the global minimum value of the objective functions will remain the same during the computation except when there are no other common global minimizers.

The organization of this chapter is as follows: In Section 3.2, we first give a description of the evolutionary algorithm and main procedures used in it. In Section 3.3, we introduce the adaptive fitness function techniques that enable us to find multiple solutions for the VIP. The basic ideas behind the tunneling and hump function techniques are also contained in this section. In Section 3.4, we present some diversification techniques. In Section 3.5, we discuss our HEA and its elements in detail. Then we present some numerical results for well known test problems in Section 3.6 and conclude the chapter in Section 3.7.

3.2 Evolutionary Algorithm

An evolutionary algorithm is based on the idea of imitating the evolutionary process observed in nature. Encouraged by the roles of reproduction, mutation and survival in the evolution of living things, an evolutionary algorithm tries to combine and change elements of existing solutions in order to create a new solution with some of the features of parents, and selects next candidate solutions among them [13, 28, 36, 103]. An evolutionary algorithm for optimization is different from classical optimization methods in several aspects. First of all, it depends on random sampling, i.e., the method is non-deterministic. So there is no theoretical guarantee for the method to find an optimal solution. Secondly, an evolutionary algorithm works with a *population* of candidate solutions, meanwhile classical optimization methods usually maintain a single best solution found so far. The use of population sets helps the evolutionary algorithm avoid being trapped at a local solution.

Basic scheme of an evolutionary algorithm is given in Figure 3.1. It relies on procedures such as Parents selection, Crossover and Mutation, and Survival selection [13, 28].

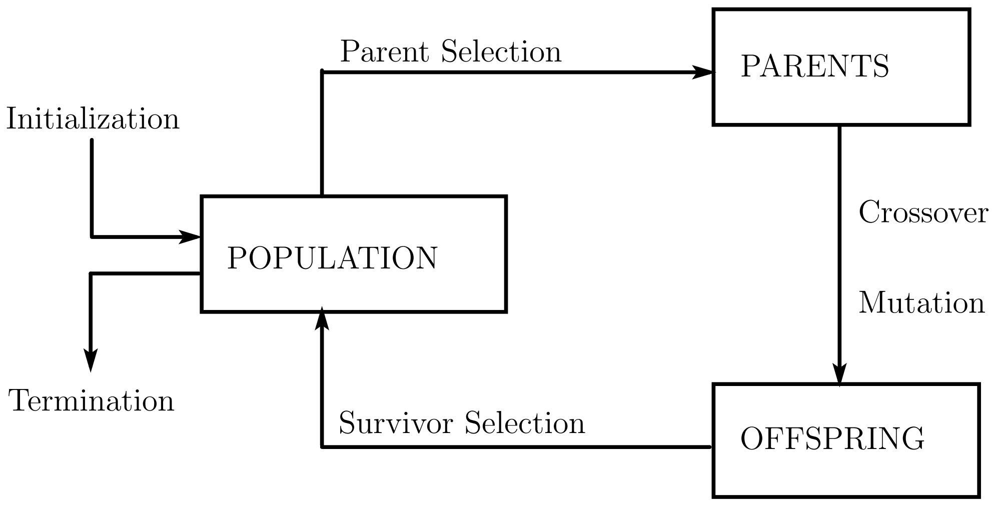

Figure 3.1: Basic scheme of an evolutionary algorithm

Now we elaborate on the procedures shown in Figure 3.1.

Initialization. We choose the parameters and the fitness function by which solutions can be assessed, and establish an initial population set. To generate the initial population set we use either a random distribution or a controlled random distribution.

Parents selection. Parents selection procedure collects parents in a set called parents pool proportionally to their fitness values. The smaller the fitness function value, the higher the probability to be chosen as a parent. Thus, it is possible for one member to dominate all the others and get selected a high proportion of the time.

Crossover and Mutation. The purpose of *crossover* is to produce children who are expected to possess better properties than their parents. Good results can be obtained with a random matching of the individuals [13, 28]. Moreover, random changes or *mutations* are made periodically for some members of the current population, thereby yielding a new candidate solution.

Survival selection. An evolutionary algorithm performs a selection process in which the most fit members of the population survive, and the least fit members are eliminated. This process is done with the help of the fitness function and leads the population towards ever-better solutions.

Termination. An evolutionary algorithm has no optimality condition for the solution of the problem, so we never know whether the solution we have found is a real optimal solution or not, unless we already knew the global minimum value of the problem beforehand. So in general, a prescribed upper limit on the number of function evaluations is used for termination. Once the number of function evaluations hits this upper limit, the algorithm stops and the best solution found so far is regarded as a global minimum.

A drawback of any evolutionary algorithm is that a solution is judged better only in comparison to currently known other solutions; such an algorithm actually has no reasonable way to test whether a solution is, even local, optimal. This drawback will disappear when the minimum objective value is known, and the global optimization problem considered here precisely meets this requirement.

3.3 Adaptive Fitness Function Techniques

When the evolutionary search gets stuck around some local or global solution, unless we intervene, it is very likely that the searching process stops around that point, even for other runs with different initial population sets. Adaptive fitness functions discussed below are proposed to prevent the searching process wandering around detected solution uselessly and give opportunity to continue searching in other regions.

First we consider the following two types of functions. Let $f_c(x)$ be the current fitness function used in the HEA and $\bar{x}$ be a point around which the function $f_c(x)$ is to be modified.

Tunneling function. [4, 55, 56] Consider the following function:

$$f_t(x, \bar{x}) := f_c(x) \cdot \exp\Big(\frac{1}{\|x - \bar{x}\|^2}\Big). \tag{3.3.1}$$

This function is called a *tunneling function* because of its behavior around the point $\bar{x}$. For the sake of computational convenience, instead of directly using the function $f_t(x, \bar{x})$, we use the following approximation of this function:

$$\bar{f}_t(x, \bar{x}) := f_c(x) \cdot \exp\Big(\frac{1}{\varepsilon_t + \frac{1}{\rho_t^2}\|x - \bar{x}\|^2}\Big), \tag{3.3.2}$$

where ε_t and ρ_t are positive parameters that control the degree and the range of modification. If $\bar{x}$ is not a global minimum, since the objective function value is zero at any global minimum, the modified function $\bar{f}_t(x, \bar{x})$ has the same global minima as the function $f_c(x)$ has.

Now let $\bar{x}$ be an isolated global minimum of $f_c(x)$. Our purpose is to construct a new fitness function which has the same global minimizers as the fitness function $f_c(x)$ has, except $\bar{x}$. Moreover, we require the new function to have no solution around $\bar{x}$. In principle, we may use the tunneling function (3.3.2). If $\bar{x}$ is an exact global solution, i.e., $f_c(\bar{x}) = 0$, then under mild condition the function $f_t(x, \bar{x})$ can satisfy our requirements. But, if $\bar{x}$ is just an approximation of a global solution $\bar{x}^*$, as one may expect in practice, then it may not be appropriate to use the tunneling function modification $f_t(x, \bar{x})$, because the exact solution $\bar{x}^*$ still satisfies $f_t(\bar{x}^*, \bar{x}) = 0$ unexpectedly (see Figure 3.2).

Below we propose a possible remedy to overcome the above-mentioned drawback of the tunneling function method.

Hump-Tunneling function. Consider first the following modified function:

$$f_h(x, \bar{x}) := f_c(x) + \alpha_h \max\Big\{0, 1 - \frac{1}{\rho_h^2}\|x - \bar{x}\|^2\Big\}, \tag{3.3.3}$$

where $\alpha_h, \rho_h > 0$ are some parameters. We call this function the *hump function* and this function may enable us to escape from the region around $\bar{x}$ even when $\bar{x}$ is an approximation of a global solution.

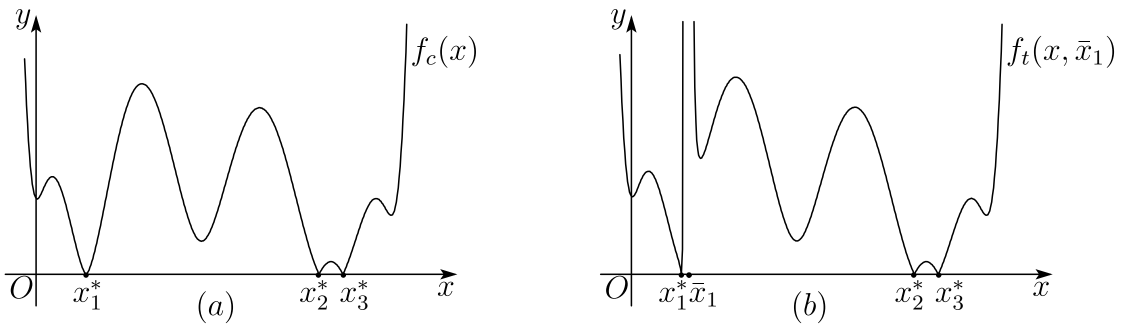

Figure 3.2: (a) Graphs of a function and (b) its tunneling modification at an approximate solution $\bar{x}_1$.

However, it is not clear how we can determine the parameter ρ_h appropriately. If we set it smaller than necessary, the modified function may not be very useful because of a narrow range of modification, and if we set it large, it may affect some other global solutions near $\bar{x}$, if any, and may make them non-global solutions any more (see Figure 3.3.(a)). Although it is not very appropriate to use either tunneling or hump function method individually, it may be effective to use a combination of these two functions.

We first take a sufficiently small positive scalar $\bar{\rho}_h$ and define a hump function $f_h(x, \bar{x})$ as in (3.3.3). Then we construct the following function:

$$\begin{aligned} \bar{f}_{ht}(x, \bar{x}) &:= f_h(x, \bar{x}) \cdot \exp\Big(\frac{1}{\varepsilon_t + \frac{1}{\rho_t^2}\|x - \bar{x}\|^2}\Big) \\ &= \Big(f_c(x) + \alpha_h \max\Big\{0, 1 - \frac{1}{\bar{\rho}_h^2}\|x - \bar{x}\|^2\Big\}\Big) \cdot \exp\Big(\frac{1}{\varepsilon_t + \frac{1}{\rho_t^2}\|x - \bar{x}\|^2}\Big). \quad (3.3.4) \end{aligned}$$

We call this function the *hump-tunneling function* and zero points of this function coincide with those of the function $f_c(x)$ except for those zeros in $B(\bar{x}, \bar{\rho}_h)$. Choosing $\bar{\rho}_h$ small enough, we can avoid affecting other global solutions near $\bar{x}$ (see Figure 3.3.(b)).

Now we explain when and how to use these functions in detail. The HEA collects the points at which the detected global or local solutions, or unpromising trial points in the set S_{mod} of modification points. Once one of those points is detected, the method adds it to S_{mod} and modifies the objective function around this point in order to avoid returning to it in the further search. Moreover, the HEA collects the detected global solutions in the set S_{glob} and it will play an important role in the algorithm.

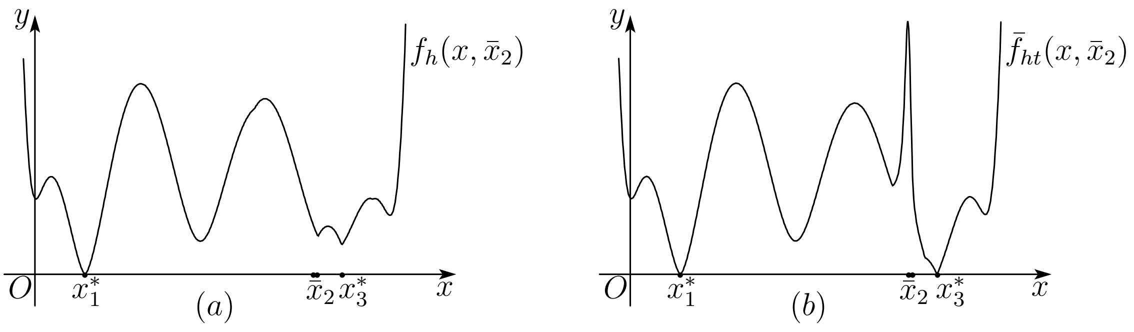

Figure 3.3: (a) Graphs of a hump function and (b) a hump-tunneling function constructed through modification at an approximate solution $\bar{x}_2$ of the function of Figure 3.2.(a).

Let us consider the modifications.

1. If $\bar{x}$ is an global solution, then we set

$$S_{\text{mod}} := S_{\text{mod}} \cup \{\bar{x}\}, \quad S_{\text{glob}} := S_{\text{glob}} \cup \{\bar{x}\},$$

$$f_c(x) := \Big(f(x) + \sum_{x_g \in S_{\text{glob}}} \max\Big\{0, 1 - \frac{1}{\bar{\rho}_h^2}\|x - x_g\|^2\Big\}\Big)$$

$$\cdot \exp\Big(\sum_{x_m \in S_{\text{mod}}} \frac{1}{\varepsilon_t + \frac{1}{\rho_t^2}\|x - x_m\|^2}\Big).$$

Before considering the next type of modification, let us introduce the concepts of semi-local solutions and unpromising trial points.

Definition 3.3.1. *If after a certain number of evolutionary generations and local searches, the best candidate solution in the population set $\mathcal{P}$ has not been improved and it is not a global minimum, then we say the point is a semi-local solution.*

Definition 3.3.2. *Let $f(x)$ and $f_c(x)$ be the original and the current fitness functions, respectively, and $\bar{x}$ be a trial point. Suppose a local search is executed on the original objective function f with the starting point $\bar{x}$. If the current fitness function value increases after the local search, then we say that $\bar{x}$ is an unpromising trial point.*

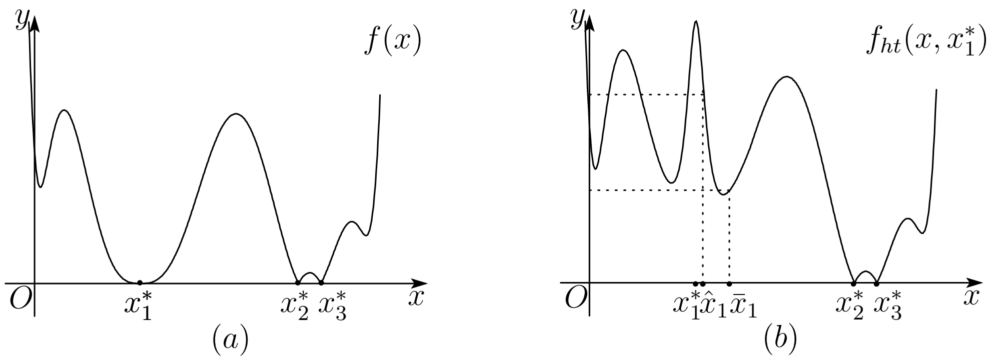

Figure 3.4: (a) the original function and (b) an illustration of unpromising trial point .

Figure 3.4.(b) illustrates an unpromising trial point. Let $\bar{x}_1$ be a global solution and $\hat{x}_1$ be obtained by local search applied to the original function from the starting point $\bar{x}_1$. Then since the modified function value increases after the local search, $\bar{x}_1$ is unpromising.

2. Suppose $\bar{x}$ is a semi-local solution. Since it may still attract the population set, we need to modify the function around this point. A similar observation applies when $\bar{x}$ is an unpromising trial point, and we also modify the function. In either case, we set

$$S_{\text{mod}} := S_{\text{mod}} \cup \{\bar{x}\}, \quad f_c(x) := f_c(x) \cdot \exp\Big(\frac{1}{\varepsilon_t + \frac{1}{\rho_t^2}\|x - \bar{x}\|^2}\Big).$$

Determining the tunneling parameters. The choice of the tunneling parameters is very important when constructing a new fitness function. Even with the hump-tunneling function, it is not easy to fully escape from the basin of the already detected solutions. If we choose the tunneling range parameter ρ_t smaller than enough, the tunneling will be not effective and the algorithm will need more modifications to leave fully the basin of this solution. On the other hand, if we choose this parameter inappropriately big, then it also may affect the other solutions basins and may make them narrower. So here we consider a heuristic approach to determine appropriate parameters ε_t, ρ_t.

Assume that $\bar{x}$ is a trial solution on which the tunneling is to be constructed. First we will determine the basin of this point, so this will in turn give the range of the tunneling.

Choose a parameter $\tau > 1$ and execute the following procedures.
Generate r random vectors $d_i^j,\ j = 1, 2, \dots, r$ with unit length and check the condition

$$f(\bar{x} + \tau^i d_i^j) < f(\bar{x} + \tau^{i+1} d_i^j),\ \forall j \in \{1, \dots, r\} \tag{3.3.5}$$

If (3.3.5) holds for $i = 0$, then repeat the above procedure by setting $i := i + 1$ until (3.3.5) is violated. As soon as (3.3.5) is violated, let $\bar{\delta} := \tau^{i-1}$ and $\tilde{x} = \bar{x} + \bar{\delta}\bar{d}$, where $\bar{d}$ is one of the random vectors violating the condition (3.3.5).

If (3.3.5) does not hold for $i = 0$, then repeat the above procedure by setting $i := i - 1$ until (3.3.5) is satisfied. As soon as (3.3.5) is satisfied, let $\bar{\delta} := \tau^{i}$ and $\tilde{x} = \bar{x} + \bar{\delta}\bar{d}$, where $\bar{d}$ is one of the random vectors last considered in checking (3.3.5).

Now we may deduce that the basin of the point $\bar{x}$ includes the ball $B(\bar{x}, \bar{\delta})$. So our tunneling should affect more the region inside $B(\bar{x}, \bar{\delta})$ and less outside it. For example, we require that the new fitness function at the point of tunneling has a value at least twice as much as that at the point $\tilde{x}$, i.e.,

$$\bar{f}_{ht}(\bar{x}) \geq 2\bar{f}_{ht}(\tilde{x}),$$

and at the point $\tilde{x}$ it has a value no greater than $\exp(\frac{1}{4})$ times $f(\tilde{x}) - f(\bar{x})$, i.e.,

$$\bar{f}_{ht}(\tilde{x}) \leq \exp(\frac{1}{4})(f(\tilde{x}) - f(\bar{x})).$$

Then simple calculation gives us possible choices

$$\rho_t = 0.25\bar{\delta}, \ \varepsilon_t = \min\{1, -2 + 2\sqrt{1 + \frac{1}{\log 2(f(\tilde{x}) - f(\bar{x}))}}\}.$$

Collecting all the procedures given in this section, we denote by $\mathbf{AFF}(f_c, \bar{x}, S_{\text{mod}}, S_{\text{glob}}, \mathcal{P})$ the fitness function modification procedure. This procedure yields a new fitness function, which is a modification of the former fitness function f_c on $\bar{x}$, with the corresponding changes in the sets $S_{\text{mod}}, S_{\text{glob}}$ and $\mathcal{P}$.

3.4 Population Update Rules

In order to search for many global solutions simultaneously, the proposed evolutionary algorithm first tries to keep diversity in the population set. Due to the rules of accepting a newly produced trial solution to survive in the population set, most evolutionary algorithms have the tendency that population sets eventually cluster around only a few solutions. This is because, in ordinary evolutionary algorithms, a new trial solution is usually accepted to survive and replace some solution in the population set, if it is better than that. Although some algorithms such as scatter search method [51, 52] try to keep diversity, the number of different good points in the population set is still small (even if the objective function has many global solutions) and the remaining points are usually just diversity points. The HEA uses the Population Update Rules, which are novel types of criteria for accepting new trial solutions to survive in the population set, and tries to keep diversity while searching for promising points.

Here we propose two different techniques to update the population set, which are aimed to keep diversity while searching for global solutions. The first one is somewhat heuristic and depends on the structure of the population set. The second one is based on some tolerance parameter for the distance between trial points.

Population Update 1. Consider a set of points $\mathcal{P} := \{x^1, x^2, \dots, x^M\}$ sorted according to their objective function values. Let x be a trial solution used to update the population set.
1. If $f(x) \geq f(x^M)$, i.e., x is worse than the worst element in $\mathcal{P}$, then discard x.
2. If $f(x) \leq f(x^1)$, i.e., x is better than the best element in $\mathcal{P}$, then add x to $\mathcal{P}$ and delete the closest point to x in $\mathcal{P}$.
3. If $f(x^i) \leq f(x) < f(x^{i+1})$, then let

$$k := \operatorname*{argmin}_{1 \leq j \leq i} \|x - x^j\|, \quad l := \operatorname*{argmin}_{i+1 \leq j \leq M} \|x - x^j\|.$$

Namely, x^k is the closest point to x among such points in $\mathcal{P}$ that their objective function values are smaller than $f(x)$, while x^l is the closest point to x among such points in $\mathcal{P}$ that their objective function values are greater than $f(x)$.
If $\|x - x^k\| \leq \|x^k - x^l\|$, then discard x.
If $\|x - x^k\| > \|x^k - x^l\|$ and $\|x - x^l\| \leq \|x^k - x^l\|$, then delete x^l from $\mathcal{P}$ and add x to $\mathcal{P}$ in the $(i+1)$-th position.
Otherwise, delete x^M from $\mathcal{P}$ and add x to $\mathcal{P}$ in the $(i+1)$-th position.

Population Update 2. Let $\mathcal{P} := \{x^1, x^2, \dots, x^M\}$ be a set of points sorted according to their function values, and $\varepsilon_D > 0$ be a fixed tolerance for the distance. Let x be a trial solution. Define

$$B(x, \varepsilon) := \{y \in R^n | \; \|x - y\| < \varepsilon\}, \quad k(i) := \operatorname*{argmin}_{1 \leq j \leq i} \|x - x^j\|.$$

1. If $f(x) \leq f(x^1)$, then add x to the set $\mathcal{P}$ and delete from $\mathcal{P}$ all the points x^j satisfying $x^j \in B(x, \varepsilon_D)$. If there is no such element in $\mathcal{P}$, then delete x^M from $\mathcal{P}$. If there are many, add new trial solutions generated by using Diversification Generation Method to $\mathcal{P}$ to keep the size of the population set $\mathcal{P}$ equal to M.

2. If $f(x^i) < f(x) \leq f(x^{i+1})$, then do the following:
If $x \in B(x^{k(i)}, \varepsilon_D)$, then discard x. Otherwise, add the point x to $\mathcal{P}$, and delete all the elements $x^j,\ j = i+1, \ldots, M$ of $\mathcal{P}$ satisfying $x^j \in B(x, \varepsilon_D)$. If there is no such element in $\mathcal{P}$, then delete x^M from $\mathcal{P}$. If there are many, add new trial solutions generated by using Diversification Generation Method to $\mathcal{P}$ to keep the size of the population set $\mathcal{P}$ equal to M.

If $\varepsilon_D = 0$, then the Population Update Rule 2 will coincide with the ordinary update rule used in the genetic algorithm that accepts a child to survive if it is better than an element in the population.

3.5 Hybrid Evolutionary Algorithm

In this section, we describe our hybrid evolutionary algorithm HEA for the optimization problem (3.1.1) with *zero* global minimum value:

$$\min f(x) \ \text{ s.t. } x \in D,$$

where the constraint set is defined as $D := \{x \in R^n |\ l \leq x \leq u\}$ with $l, u \in R^n \cup \{\pm\infty\}$. Assume that f is continuously differentiable. Our purpose is to design an evolutionary algorithm which is able to find as many solutions as possible of problem (3.1.1).
First we elaborate the steps used in the HEA.

Diversification Generation Method. The purpose of the diversification generation [51] is to generate a well distributed set of trial solutions. The basic Diversification Generation method uses controlled randomization and frequency memory to generate a set of diverse solutions. This can be accomplished by dividing the range $[l_i, u_i]$ of each variable into 4 sub-ranges of equal size. Then, a solution is constructed in two steps. First a subrange is randomly selected. The probability of selecting a subrange is determined to be inversely proportional to its frequency count. Then a value is randomly generated within the selected subrange.

Crossover and Mutation Procedure. The purpose of crossover is to produce children who are expected to possess better properties than their parents. Good results can be obtained with a random matching of the individuals [13, 28]. Some well known crossovers are the following [36].

Single-point crossover: One crossover position (coordinate) in the vector of variables (genes) is randomly selected and the variables situated after this point are exchanged between individuals, thus producing two offsprings.

Multi-point crossover: Some crossover positions are chosen, and then the variables between successive crossover points are exchanged among the two parents to produce new offsprings.

Intermediate recombination: The values of the offspring variables are chosen from the values of the parents variables according to some rule.

Although we could use various types of existing crossovers, we propose another crossover which may hopefully be more appropriate to our problem. We have seen in the Chapter 2 the fact that

$$x \text{ solves the VI}(D,F) \iff x = \Pi_D(x - F(x)),$$

where $\Pi_D(z) := \underset{y\in D}{\operatorname{argmin}} \|z-y\|$ is the projection of point z on D. If we denote the mapping $H(x) := \Pi_D(x - F(x))$, then we have

$$\|x - \Pi_D(x - F(x))\| = \|x - H(x)\| = \sqrt{\sum_{i=1}^{n}(x_i - H_i(x))^2}.$$

Here $H_i(x)$ is i-th component of the vector $H(x)$. According to this formula, we may tell to some extent the quality of the gene x_i, that is, the smaller the value $|x_i - H_i(x)|$, the better the gene. In particular, the equalities $x_i - H_i(x) = 0,\ i = 1,\dots,n,$ hold at any solution of the VIP. Taking into account these properties, we propose the following crossover and mutation.

Let (p^1, p^2) be a pair of solutions used to produce new trial solutions.
Crossover. Let $\bar{p}$ denote the vector whose coordinates are given by

$$\bar{p}_j := \begin{cases} p_j^1, & \text{if } |p_j^1 - H_j(p^1)| \le |p_j^2 - H_j(p^2)|, \\ p_j^2, & \text{otherwise,} \end{cases} \qquad j = 1,\dots,n.$$

Choose random numbers r_1, r_2 from the interval $[0,1]$. Define two new trial solutions as follows:
If $\bar{p} \neq p^1$ or $\bar{p} \neq p^2$, then

$$c^i := p^i + r_i(\bar{p} - p^i),\ i = 1,2.$$

Otherwise,

$$c^1 := p^1 + r_1(p^2 - p^1), \quad c^2 := \begin{cases} \Pi_D(p^1 - r_2(p^2 - p^1)), & \text{if } \bar{p} = p^1; \\ \Pi_D(p^2 - r_2(p^1 - p^2)), & \text{if } \bar{p} = p^2. \end{cases}$$

Mutation. Choose random numbers r_1, r_2 from the interval $[0,1]$. Define two new trial solutions as follows:

$$c^i := p^i + r_i(H(p^i) - p^i),\ i = 1, 2.$$

In the HEA, we use the above Crossover and Mutation in addition to the multi-point crossover to generate the children.

Now we will describe the evolutionary algorithm with adaptive fitness function. We first discuss parameters and procedures that will be used in the algorithm.

M – number of elements in the population set,

l_N – number of best points to which local search is applied,

l_s – maximum number of steps per local search,

$\bar{N}, \eta$ – parameters used to determine semi-global solutions,

$Crossover[(p^1, p^2)] + Mutation$ – the mating procedure for the pair (p^1, p^2) and possible mutation for the resulted children pair,

Local Search $(f(x), \bar{x}, l_s)$ – a local search process for the function $f(x)$ starting from the point $\bar{x}$ with the number of steps l_s.

To check whether a point is semi-local or not, we use $\bar{N}$ evolutionary generations and a local search step. Here we use a set B whose elements represent the historical data of the best points in the population set during the last $\bar{N}$ generations.

To terminate our EA, we use the following three different criteria.

S1. The number of function evaluations exceeds the pre-determined upper limit,

S2. The number of detected global solutions exceeds the pre-determined number,

S3. Let N_s be a pre-specified positive integer. The most recently added N_s elements of the set S_{mod} of modification points were not new global solutions.

If one of those criteria is satisfied, then we terminate the main algorithm.

The main loop of the proposed algorithm is stated as follows.

Algorithm 3.5.1. *HEA*

1. Initialization. *Choose parameters $M, l_N, l_s, \bar{N}, \varepsilon > 0$ and $\eta \in (0,1)$. Generate the population set $\mathcal{P}$ by using some diversity generation method. Let the set of modification points and the set of global solutions be $S_{mod} := \emptyset$ and $S_{glob} := \emptyset$, respectively. Define the current fitness function as*

$$f_c(x) := f(x).$$

Sort the elements in $\mathcal{P}$ in ascending order of their current fitness function values, i.e.,

$$f_c(x^1) \leq f_c(x^2) \leq \cdots \leq f_c(x^M).$$

Set the generation counters $t := 1$ and $s := 1$.

2. Parents Selection. *Generate a parents pool*

$$\mathcal{P}' := \{(x^i, x^j) | x^i, x^j \in \mathcal{P},\ x^i \neq x^j\}.$$

3. Crossover and Mutation. *Select a pair $(p^1, p^2) \in \mathcal{P}'$ and generate a pair as*

$$(c^1, c^2) \longleftarrow Crossover[(p^1, p^2)] + Mutation.$$

4. Population Update. *Using the Population Update Rule with c^1 and c^2, update the population set $\mathcal{P}$. Delete the pair (p^1, p^2) from the parents pool $\mathcal{P}'$. If $\mathcal{P}' = \emptyset$, then let $N := min\{s, \bar{N}\}$ and*

$$B := \{b^1, b^2, \ldots, b^N\} \leftarrow \{x^1, b^1, \ldots, b^{(N-1)}\},\ s := s + 1$$

and go to Step 5; otherwise go to Step 3.

5. Intensification. *If, during the last $\bar{N}$ generations of evolution, the fitness function has not been modified and the best point in the population set has not been improved enough, i.e.,*

$$s \geq \bar{N} \text{ and } \left| f_c(b^{\bar{N}}) - f_c(b^1) \right| \leq \eta(1 + f_c(b^1)),$$

then choose $x^1, x^2, ..., x^{l_N} \in \mathcal{P}$ *and for each* x^i, $i = 1, 2, ..., l_N$, *perform the following procedure:*

$$\bar{x}^i \longleftarrow \textit{Local Search } (f(x), x^i, l_s).$$

If x^i *is an unpromising trial point, then construct a new fitness function by*

$$f_c(x) := \boldsymbol{AFF}(f_c, x^i, S_{mod}, S_{glob}, \mathcal{P}).$$

Otherwise, $\mathcal{P} := \mathcal{P} \cup \{\bar{x}^i\} \backslash \{x^i\}$.

If the fitness function is modified at least once during the above procedure, then set $s := 1$. *Go to Step 6.*

6. Semi-local solutions and adaptation. *If* $x^1 \in \mathcal{P}$ *is a semi-local solution, i.e.,*

$$s \geq \bar{N} \text{ and } \left| f_c(b^{\bar{N}}) - f_c(x^1) \right| \leq \eta(1 + f_c(x^1)),$$

then construct a new fitness function by

$$f_c(x) := \boldsymbol{AFF}(f_c, x^1, S_{mod}, S_{glob}, \mathcal{P})$$

and set $s := 1$. *Otherwise, let* $B := \{b^1, b^2, \ldots, b^{\bar{N}}\} \leftarrow \{x^1, b^1, \ldots, b^{(\bar{N}-1)}\}$. *Proceed to Step 7 with* $(f_c(x), \mathcal{P})$.

7. Stopping Condition. *If one of the stopping conditions holds, then terminate the algorithm and refine the global solutions in* S_{glob} *by some local search method. Otherwise, set* $t := t + 1$ *and go to Step 2.*

Remark 3.5.2. *This algorithm can be used for solving a more general class of problems. Particularly, we can employ the HEA for solving an extension of VIP, the quasi variational inequality problem (QVIP). For a vector valued mapping* F *and a set valued mapping* S, *the QVIP is to find a vector* $x^* \in S(x^*)$ *such that*

$$\langle F(x^*), x - x^* \rangle \geq 0, \quad \forall x \in S(x^*). \tag{3.5.1}$$

Unlike the ordinary VIP, the constraint set S in the QVIP varies depending on the variable x. When the constraint set S is constant, i.e., $S(x) \equiv S$, the QVIP reduces to the ordinary VIP. Although the application field of this type of problem is diverse including the generalized Nash equilibrium problem, there hardly exist practically implementable methods for the QVIP yet. Fortunately, the linearized gap function and the linearized D-gap function for the VIP discussed in Chapter 2 still apply for the QVIP [24], and allow us to reformulate the QVIP as an unconstrained optimization problem with zero global minimum value, which can be solved by using the HEA.

3.6 Numerical Experiments

The performance of the HEA was tested on a number of well known test problems in the MCPLIB library. To show the efficiency of the HEA we have used only those problems which have multiple solutions, and for each problem we made 20 trials with different initial populations. The programming code for the algorithm was written in MATLAB and run on a computer Pentium 4, Microprocessor.

The regularized gap function is used to reformulate MCPLIB test problems as optimization problems, and for local search in the HEA, we employ MATLAB's command `fmincon` combined with an active set detecting strategy. Moreover, the HEA is supposed to use a finite box for generating diversity points in the Diversification Generation Method, whereas most of the test problems are mixed complementarity problems which have no lower or upper bound. To deal with such problems, we use a fixed finite box defined inside the original box in the Diversification Generation Method, while using the original box constraint in the Crossover Mutation Procedure and in the local search.

In general, it is difficult to universally determine suitable values of HEA parameters for every problem, because they are highly problem dependent. Nevertheless, through testing many times on various test problems, we suggest possible choices of the parameters as shown in Table 3.1.

We have two versions of the HEA; HEA_1 and HEA_2 that use Population Update Rule 1 and Rule 2, respectively. We ran the HEA versions for all the chosen test problems with the general parameter settings mentioned in Table 3.1 and put the numerical results in Tables 3.2 and 3.3. The columns in these tables have the following meanings:

Table 3.1: Parameter Settings.

Parameters	Definition	Value
M	number of elements in population	$\min\{2n+4, 20\}$
l_N	number of best points for which local search is used	2
l_s	maximum number of steps per local search	$\min\{2n, 30\}$
$\bar{N}, \eta$	parameters controlling local search in HEA	3, 10^{-3}
ε	tolerance parameter for the objective function in HEA	10^{-6}
N_{max}	maximum number of ineffective local transformations	10
N_{gmax}	maximum number of global solutions to be found	20
NF_{max}	maximum number of function evaluations	$5n10^4$
ε_D	distance tolerance used in Population Update Rule 2	$n/5$
ε_t, ρ_t	tunneling parameters used in (3.3.2) and (3.3.4)	0.1, 2
α_h, ρ_h	humping parameters used in (3.3.4)	1, 0.3

Problem:	name of the test problem,
n:	dimension of the test problem,
K_{min}, K_{av}, K_{max}:	minimum, average, maximum numbers of solutions found by the algorithm,
N_{gen}:	average number of generations,
N_{loc}:	average number of local steps taken,
NF:	average number of function evaluations,
N_f:	average number of function evaluations when the last global solution is obtained.

The results reported in Tables 3.2 and 3.3 show that the HEA is promising. For most of test problems, the average numbers of obtained global solutions (K_{av}) are close to the maximum numbers of obtained global solutions (K_{max}), and this implies that the HEA versions are capable of finding multiple solutions. Moreover, the average numbers of generations are reasonable compared with the problem dimensions and the numbers of obtained global solutions. The HEA versions use three different stopping conditions. Specifically, if the number of global solutions or that of ineffective transformations (i.e., the number of subsequently detected local solutions or unpromising trial points) or that of function evaluations exceeds their respective pre-specified limits $N_{gmax}, N_{max}, NF_{max}$, then the algorithm is terminated. We observe in both tables that the HEA versions find global solutions in a relatively small number of function evaluations (N_f), and after that, the algorithms were still running until one of the termination conditions is met in order to check whether there are any other solution left or not.

Table 3.2 reveals that the HEA$_1$ finds no less than N_{gmax} (=20) global solutions for five test problems. In fact, we may conclude that these problems have infinitely many solutions, and by setting N_{gmax} bigger, it is possible to find as many solutions as one may want. For the other five problems, the algorithm was terminated because the number of ineffective local transformations exceeded $N_{max}(= 10)$.

Table 3.2: Numerical Experiments for the HEA with Population Update Rule 1.

Problem	n	K_{min}	K_{av}	K_{max}	N_{gen}	N_{loc}	NF	N_f
badfree	5	20	20	20	31	14	3260	3260
games	16	20	20	20	74	432	32859	32859
kojshin	4	2	2	2	38	115	4023	1474
mathinum	3	1	14.4	20	255	858	30402	23066
mathisum	4	1	1.9	2	52	184	6751	2974
ne-hard	3	2	3.1	4	185	956	31223	16068
powell	16	4	11.4	20	96	1644	45740	36676
powell_mcp	8	2	5.1	9	110	1109	24585	10553
scarfasum	14	2	2.7	3	80	1157	45508	25139
sppe	27	20	20	20	325	471	138475	138475

Table 3.3 shows that, the performance of the HEA$_2$ is promising except for problem *mathinum*, for which it occasionally failed to find a global solution despite the fact that this problem has at least twenty solutions as shown in Tables 3.2 and 3.3. It happened because the number of ineffective local transformations reached its limit $N_{max}(= 10)$ before finding a global solution. By increasing N_{max}, we could improve the performance of the HEA$_2$ for this problem. Another possible reason for the unexpected performance of the HEA$_2$ for

problem *mathinum* is the choice of parameter ε_D. As we mentioned earlier in Section 3.4, the Population Update Rule 2 is an extension of the standard genetic algorithm selection mechanism and it tries to prevent the population from prematurely converging to one or only a few points. However, we found that the choice $\varepsilon = n/5$ was not appropriate for problem *mathinum*, since the HEA$_2$ could not escape from the undesirable property that the population converges to only a few points. We have observed that, by increasing ε_D, we could improve the performance of the HEA$_2$ for this problem.

Table 3.3: Numerical Experiments for the HEA with Population Update Rule 2.

Problem	n	K_{min}	K_{av}	K_{max}	N_{gen}	N_{loc}	NF	N_f
badfree	5	20	20	20	15	0	2946	2946
games	16	20	20	20	31	132	25529	25529
kojshin	4	2	2	2	28	92	5786	1986
mathinum	3	0	4.3	20	794	1008	132280	41290
mathisum	4	1	1.95	2	48	95	9299	2728
ne-hard	3	2	3.3	4	82	261	17930	10746
powell	16	5	14	20	71	1160	49965	34619
powell_mcp	8	3	6.9	11	107	1123	32349	12887
scarfasum	14	1	1.6	3	49	714	55479	22456
sppe	27	20	20	20	225	1563	260650	260650

Finally, we make some remarks on the comparison between the results shown in Tables 3.2 and 3.3 in terms of the numbers of obtained global solutions and computational costs. Generally, the HEA versions are neutral in terms of the numbers of obtained global solutions, since these numbers are almost the same for six problems out of ten. For problems *powell* and *powell_mcp*, the HEA$_2$ was able to find more global solutions than the HEA$_1$. However, for problems *mathinum* and *scarfasum*, the HEA$_1$ performed better than the HEA$_2$ in terms of the numbers of obtained global solutions and the numbers of function evaluations. On the other hand, the HEA$_2$ did not use local search in all runs for problem *badfree*, while the HEA$_1$ required more local search steps than the HEA$_2$ for seven out of ten problems.

3.7 Conclusion

In this chapter, we have presented a new population-based algorithm HEA that is designed to find as many solutions as possible of the general VIP. New types of population update schemes in the evolutionary algorithm and a hump-tunneling technique for escaping from detected solutions have also been proposed. The computational results for some well known test problems show that the HEA method is capable of locating many solutions in an acceptable number of function evaluations. Moreover, the numerical results indicate that, the more solutions a problem has, the better the HEA method works. Finally, it is worth mentioning that one can use the HEA for finding solutions of a system of equations or a global optimization problem with known minimum objective value.

Chapter 4

Restricted-Step Josephy Newton Method for VIPs

4.1 Introduction

Our next approach is a combination of heuristic and deterministic approaches. Consider the variational inequality problem, to find a vector $x^* \in S$ such that

$$\langle F(x^*), x - x^* \rangle \geq 0, \quad \forall x \in S. \tag{4.1.1}$$

In this chapter, we present two methods for general VIPs. These methods are designed with quite different ideas, but they are linked in that the second method can be used as a subprocedure within the first method.

The first method is a modification of the classical Josephy-Newton method for VIPs. The Josephy-Newton method is an iterative method that successively solves sub-linearized variational inequalities and local superlinear convergence of the method has been established [39]. Globalization of the Josephy-Newton method by combining it with some line search methods for equivalent optimization problem of VIP has also been designed [17, 101]. A difficulty of using those approaches in practice is that solvability of the subproblems is not guaranteed, i.e., in some case the subproblems have no solution. Furthermore, even when the subproblems have solutions, it is not easy to find them because the subproblems themselves are general variational inequalities with the same constraint set S.

Our method adopts the idea that an extra bound constraint (such as a box or a ball) is added to the original constraint set in the subproblems so that the Josephy-Newton subproblems have always solutions. In order to get a global convergence property, we combine the Josephy-Newton method with the gradient projection method for the equivalent optimization problem and we employ the regularized gap function for VIP (4.1.1) as a merit function. Although the solvability of the subproblems is ensured by means of the above mentioned modification, we still need some effective method for solving those subproblems.

The second method we develop is to meet this requirement. It is an evolutionary algorithm designed to solve the VIP with bounded polyhedral constraints and also relies on the equivalent global optimization problem of the VIP. The global minimum value of the equivalent optimization problem is known to be zero and the algorithm uses this fact as a stopping condition. Unlike the heuristic method developed in the previous chapter, this method directly deals with the special structure of the polyhedral constraint set. Moreover, crossover and mutation procedures amenable to the polyhedral constraint set are devised as well as the initial population generation procedure. In addition, we use the adaptive fitness function procedure to increase the validity of the algorithm. In general, an evolutionary algorithm is expensive computationally. However, our target here is to solve VIPs with bounded polyhedral sets, of which size is presumably small. For such problems, we may expect that the algorithm can find a solution with a reasonable amount of computations.

The organization of the chapter is as follows. In Section 4.2, we propose the restricted-step Josephy-Newton method and discuss its convergence properties. Section 4.3 presents the evolutionary algorithm for solving VIPs with bounded polyhedral constraints. Appropriate crossover and mutation procedures as well as the fitness function modification procedure will also be discussed. In Section 4.4, we report numerical results of the methods for some test problems. Section 4.5 concludes the chapter.

4.2 Restricted-Step Josephy-Newton Method

In this section we present the restricted-step Josephy-Newton method for solving the general VIP. Fist we will briefly review the classical Josephy-Newton method [39]. Let us consider the VIP (4.1.1). At the k-th iteration, the Josephy-Newton method solves the linearized sub-VIP

$$\langle F^k(x), x' - x\rangle \geq 0, \quad \forall x' \in S \tag{4.2.1}$$

to get the next iteration point x^{k+1}. Here F^k is the linear approximation of the mapping F at x^k, i.e.,

$$F^k(x) := F(x^k) + F'(x^k)(x - x^k).$$

It has been shown [39] that this algorithm is locally superlinearly convergent and, by combining it with a descent method for a merit function, there have been developed some methods which are globally convergent and locally superlinearly convergent [17, 101]. A drawback of these methods is that the solvability of (4.2.1) is not guaranteed and even if it has a solution, without some kind of monotonicity assumption on (4.2.1) we may fail to solve it. We propose here a modification of the Josephy-Newton method which is implementable in practice.

We consider the Josephy-Newton subproblem modified as

$$\langle F^k(x), x' - x \rangle \geq 0, \quad \forall x' \in S \cap B(x^k, \delta), \tag{4.2.2}$$

where $\delta > 0$ is a fixed scalar and $B(x^k, \delta) := \{x \in \Re^n | \; \|x - x^k\| \leq \delta\}$. We combine this method with the gradient projection method for the equivalent optimization problem to VIP (4.1.1)

$$\min \theta(x) \text{ s.t } x \in S. \tag{4.2.3}$$

Lemma 4.2.1. *A point x^k is a solution of VIP (4.1.1) if and only if it is a solution of the modified Josephy-Newton subproblem (4.2.2).*

Proof. Let x^k be a solution to VIP (4.1.1). Then $\langle F(x^k), x' - x^k \rangle \geq 0, \; \forall x' \in S$, and since $x^k \in S \cap B(x^k, \delta)$ and $F^k(x^k) = F(x^k)$, x^k is solution to (4.2.2).

Next, let x^k be a solution of the modified Josephy-Newton subproblem (4.2.2) and show it is a solution of VIP (4.1.1). Assume to the contrary that there exists a point $\bar{x} \in S$ such that $\langle F(x^k), \bar{x} - x^k \rangle < 0$. Then the point $\tilde{x} := x^k + \min\{\delta, 1\} \dfrac{\bar{x} - x^k}{\|\bar{x} - x^k\|}$ is in $S \cap B(x^k, \delta)$ and

$$\langle F^k(x^k), \tilde{x} - x^k \rangle = \frac{\min\{\delta, 1\}}{\|\bar{x} - x^k\|} \langle F(x^k), \bar{x} - x^k \rangle < 0.$$

This contradicts the fact that x^k is a solution of (4.2.2). $\Box$

The use of the subproblem (4.2.2) has some advantages. First of all, this problem always has a solution because the constraint set is bounded and closed convex. Secondly, we may develop some effective method for solving it when the parameter δ is of appropriate size. In general, the choice of the parameter δ is important for the efficiency of the algorithm. If δ is inappropriately big, then the procedure of solving the sub-VIP (4.2.2) may not be very effective. On the other hand, if we choose δ too small, then the improvement of the solution might also be very small, resulting in slow convergence. One possible way to cope with this situation is to adjust the parameter δ appropriately during the iterations in a way similar to the trust region method in nonlinear optimization. We use some upper limit δ_{max} and lower limit δ_{min} for the parameter δ to prevent it from becoming too large or too small. In the next section, we will present an algorithm for solving sub-VIP (4.2.2).

The restricted-step Josephy-Newton algorithm, referred to as Algorithm rJN, is formally stated as follows.

Algorithm 4.2.2. *Algorithm rJN*

1. Initialization. *Choose* $x^0 \in S$ *and set parameters* $\varepsilon_1 > 0$, $\varepsilon_2 > 0$, $p > 1$, $\gamma \in (0,1)$, $\sigma \in (0,1)$, $\rho > 0$, $\beta \in (0,1)$, $0 < \delta_{min} < \delta_{max}$ *and a negative integer* i_{min}. *Set* $\delta^0 := \delta_{max}$ *and the iteration counter* $k := 0$.

2. Stopping condition. *If* $\theta(x^k) \le \varepsilon_1$ *or* $\|x^k - \Pi_S(x^k - \nabla\theta(x^k))\| \le \varepsilon_2$, *then STOP.*

3. Sub-VIP. *Find a solution* $\bar{x}^k$ *of the sub-VIP (4.2.2) with* δ^k, *and let* $d^k := \bar{x}^k - x^k$.

(a) If the condition

$$\theta(x^k + d^k) \le \sigma \cdot \theta(x^k) \tag{4.2.4}$$

is satisfied, then set $x^{k+1} := \bar{x}^k$ *and* $\delta^{k+1} := \begin{cases} \min\{2\delta^k, \delta_{max}\}, & \text{if } \|x^{k+1} - x^k\| = \delta^k; \\ \max\{\delta_{min}, \|x^{k+1} - x^k\|\}, & \text{otherwise.} \end{cases}$

Set $k := k+1$ *and go to Step 2.*

(b) Otherwise, check the condition

$$\nabla\theta(x^k)^T d^k \le -\rho \cdot \|d^k\|^p, \tag{4.2.5}$$

and if it holds, then find the smallest nonnegative integer i_k *such that*

$$\theta(x^k + \beta^{i_k} d^k) \le \theta(x^k) + \gamma\beta^{i_k}\nabla\theta(x^k)^T d^k. \tag{4.2.6}$$

If $i_k = 0$, then redefine i_k by finding the largest nonpositive integer in $[i_{min}, 0]$ satisfying (4.2.6) and

$$\theta(x^k + \beta^{i_k - 1} d^k) > \theta(x^k + \beta^{i_k} d^k),\ x^k + \beta^{i_k} d^k \in S. \tag{4.2.7}$$

Set $x^{k+1} := x^k + \beta^{i_k} d^k$, $\delta^{k+1} := mid\{\delta_{min}, \|x^{k+1} - x^k\|, \delta_{max}\}$. Set $k := k+1$ and go to Step 2.

(c) If either of conditions (4.2.4) and (4.2.5) is not satisfied, then go to Step 4.

4. Gradient projection step. *Find x^{k+1} by the following gradient projection procedure: Define $x^k(\alpha) := \Pi_S(x^k - \dfrac{\alpha \delta^k}{\|\nabla \theta(x^k)\|} \nabla \theta(x^k))$ and find the smallest nonnegative integer i_k such that*

$$\theta(x^k(\beta^{i_k})) \le \theta(x^k) + \gamma \nabla \theta(x^k)^T (x^k(\beta^{i_k}) - x^k). \tag{4.2.8}$$

If $i_k = 0$, then redefine i_k by finding the largest nonpositive integer in $[i_{min}, 0]$ satisfying (4.2.8) and

$$\theta(x^k(\beta^{i_k - 1})) > \theta(x^k(\beta^{i_k})). \tag{4.2.9}$$

Set $x^{k+1} := x^k(\beta^{i_k})$, $\delta^{k+1} := mid\{\delta_{min}, \|x^{k+1} - x^k\|, \delta_{max}\}$, $k := k+1$ and go to Step 2.

Remark 4.2.3. *In Step 3(b) and Step 4, when $i_k = 0$ is accepted by (4.2.6) or (4.2.8), we try to find a stepsize larger than unity in the hope of achieving a further reduction in the function value. Note that this modification does not affect the theoretical convergence properties of the algorithm.*

In the algorithm, we use a gradient projection step when the Josephy-Newton subproblem fails to give a useful direction. The idea of combining Newton-type methods with gradient related methods, is an important technique to make Newton-type methods globally convergent and can be found in many literatures [17, 41, 88, 101]. When the iteration approaches a solution satisfying a certain regularity condition, Newton type methods will take effect and make the convergence faster.

The next theorem establishes the global and local superlinear/quadratic convergence properties of the restricted-step Josephy-Newton method.

Theorem 4.2.4. *Let F be a continuously differentiable mapping and S be a closed convex set. Let $\{x^k\}$ be an infinite sequence generated by Algorithm rJN. Then*

(a) every accumulation point of $\{x^k\}$ is a stationary point of the optimization problem (4.2.3).

(b) if x^* is an accumulation point of $\{x^k\}$ such that $F'(x^*)$ is positive definite, then x^* is a solution of the VIP (4.1.1) and the whole sequence $\{x^k\}$ converges to this point. Furthermore, if $p > 2$ and $\gamma < \frac{1}{2}$ in Algorithm rJN, then the following statements hold:

(i) eventually the unit step size for the direction d^k is accepted so that $x^{k+1} = \bar{x}^k$.

(ii) the convergence rate is Q-superlinear; furthermore, if the Jacobian $F'(x)$ is Lipschitz continuous in a neighborhood of x^*, the convergence rate is Q-quadratic.

Proof. Part (a) can be proved by using standard arguments on descent methods. For part (b), if we show that eventually the subproblems (4.2.1) and (4.2.2) have identical solution sets, then everything will essentially follow from Theorem 10.4.23 of [17].

Since F is continuously differentiable and $F'(x^*)$ is positive definite, there exists a scalar $\delta_1 > 0$ such that $F'(x^k)$ is positive definite for any $x^k \in B(x^*, \delta_1)$. This implies that the mapping $F^k(x) = F(x^k) + F'(x^k)(x - x^k)$ is strongly monotone for all k large enough, and hence the subproblems (4.2.1) and (4.2.2) both have unique solutions.

Theorem 7.3.3 of [17] states that there exists a positive scalar $\bar{\varepsilon}$ such that for every $\varepsilon \in (0, \bar{\varepsilon}]$ a $\delta_2 > 0$ exists such that for every $x^k \in S \cap B(x^*, \delta_2)$, the problem (4.2.1) has a unique solution in $B(x^k, \varepsilon)$. If we choose $\varepsilon = \min\{\bar{\varepsilon}, \frac{\delta_{min}}{2}\}$, then we can ensure that the solution of (4.2.1) is in $B(x^k, \delta_{min})$. Therefore, this solution is also a solution of (4.2.2), and since they both have unique solutions, their solution sets are identical. □

4.3 Evolutionary Algorithm for VIPs with a Bounded Constraint Set

In this section, we present a method for solving the following VIP: Find $x \in D$ such that

$$\langle \bar{F}(x), x' - x \rangle \geq 0, \ \forall x' \in D, \tag{4.3.1}$$

where $\bar{F} : \Re^n \rightarrow \Re^n$ and $D \subseteq \Re^n$. If we let $\bar{F}(x) := F^k(x)$ and $D := S \cap B(x^k, \delta)$, then this VIP reduces to the Josephy-Newton subproblems (4.2.1) in the previous section. We will confine ourselves to the case where S is a polyhedral set and the norm used to define $B(x^k, \delta)$ is the l_∞ norm. Then the set D is a bounded polyhedral set represented by a system of linear inequalities, that is,

$$D := \{x \in \Re^n | Ax \leq b\},$$

where $A \in \Re^{m \times n}$ and $b \in \Re^m$.

Then the VIP (4.3.1) can be reformulated as the following global optimization problem:

$$\min \bar{\theta}(x) \text{ s.t } x \in D, \tag{4.3.2}$$

where $\bar{\theta}$ is the gap function associated with (4.2.2). The evaluation of gap function in this case requires solving a linear programming problem. The method presented here follows the main framework of the evolutionary algorithm [13, 28] and uses some additional procedures to exploit the special features of the problem such as the known minimum value and the polyhedrality of the constraint set.

Firstly, the global minimum value of the problem is known to be zero if the VIP (4.3.1) has a solution. We use this fact not only as a stopping condition but also to improve the validity of the evolutionary algorithm. The evolutionary algorithm is a population-based algorithm. It makes use of a fitness function to evaluate the goodness of a solution, thereby keeping the population set consisting of promising solutions. If, during the search, the population set gets stuck around a point which is not a global minimum, we need to direct the population set to another possible region which may contain a global solution. But if we use the same fitness function all the time, even with a different initial population set, the algorithm is very likely to converge back to this non-global solution. So we modify the fitness function by increasing the function value around this non-global solution. More specifically, we use the tunneling function technique [4] and consider a new fitness function

$$f_t(x, \bar{x}) := f_c(x) \cdot \exp\Big(\frac{1}{\|x - \bar{x}\|^2}\Big), \tag{4.3.3}$$

where $f_c(x)$ is the current fitness function and $\bar{x}$ is the above-mentioned non-global solution. The modified fitness function $f_t(x, \bar{x})$ will have the same exact global minimum points as the original function.

Next we use crossover and mutation procedures that are amenable to the polyhedral constraint set $D = \{x \in \Re^n | Ax \leq b\}$. Let $a_i^T,\ i = 1, \ldots, m$ be the rows of matrix A, and $b_i,\ i = 1, \ldots, m$ be the components of vector b. Suppose we have two points $x \notin D$ and $y \in \operatorname{int} D$, where $\operatorname{int} D$ denotes the interior of D. First we determine the point where the line segment $[x, y]$ intersects the boundary of D. Let us define the index sets $I(x) := \{i \mid \langle a_i, x\rangle - b_i > 0\}$. Then the point of intersection is determined as

$$\bar{x} := y + \frac{-\langle a_{i(x,y)}, y\rangle + b_{i(x,y)}}{\langle a_{i(x,y)}, x - y\rangle}(x - y),$$

where $i(x, y) := \operatorname*{argmin}_{i \in I(x)} \dfrac{-\langle a_i, y\rangle + b_i}{\langle a_i, x\rangle - b_i}$. During the evolutionary search, points generated by the crossover and mutation procedures may go outside the constraint set D, and we need to bring them back into the set D. Because the projection operator may project many different points onto the same point on the boundary, it may be helpful to use points of intersection computed by the above mentioned procedure instead of projection.

To describe our procedures, it will be convenient to define the mapping $P : \Re^n \times \operatorname{int} D \to D$ by

$$P(x, y) := \begin{cases} x, & \text{for } x \in D; \\ y + \dfrac{-\langle a_{i(x,y)}, y\rangle + b_{i(x,y)}}{\langle a_{i(x,y)}, x - y\rangle}(x - y), & \text{for } x \notin D. \end{cases} \tag{4.3.4}$$

The coefficient $\dfrac{-\langle a_{i(x,y)}, y\rangle + b_{i(x,y)}}{\langle a_{i(x,y)}, x - y\rangle}$ in (4.3.4) is well defined and actually bounded. In fact, we have

$$\frac{-\langle a_{i(x,y)}, y\rangle + b_{i(x,y)}}{\langle a_{i(x,y)}, x - y\rangle} = \frac{-\langle a_{i(x,y)}, y\rangle + b_{i(x,y)}}{-\langle a_{i(x,y)}, y\rangle + b_{i(x,y)} + \langle a_{i(x,y)}, x\rangle - b_{i(x,y)}} = \left(1 + \frac{\langle a_{i(x,y)}, x\rangle - b_{i(x,y)}}{-\langle a_{i(x,y)}, y\rangle + b_{i(x,y)}}\right)^{-1},$$

which is bounded by the definition of $i(x, y)$.

Initial population generation. First we determine the analytical center x^0 of the set D and a box D_1 containing D inside. Finding such a box can be done easily by solving the $2n$ linear programming problems

$$l_i := \min_{x \in D} \langle e_i, x\rangle, \quad u_i := \max_{x \in D} \langle e_i, x\rangle, \quad i = 1, 2, \ldots, n,$$

where e_i is the i-th unit vector, and then letting $D_1 := \{x \in \Re^n |\ l_i \leq x_i \leq u_i,\ i = 1, \ldots, n\}$. Moreover the analytical center x^0 of D will be determined by

$$x^0 := \operatorname*{argmin}_{x \in D} \sum_{i=1}^{m} -\log(b_i - \langle a_i, x \rangle).$$

Then we will generate points randomly in the box D_1 and bring them inside the set D. Specifically, let $x \in D_1$ be a generated point. Then we compute a new point x_{new} as follows and add it to the population set:

$$x_{new} := x^0 + \alpha \cdot (P(x, x^0) - x^0),$$

where $P(\cdot, \cdot)$ is defined by (4.3.4) and $\alpha \in (0, 1)$ is a random scalar. Clearly all points in the population set lie in the interior of D.

Crossover. Let x^1 and x^2 be parents to mate and $x_c \in \Re^n$ be a point created by the intermediate recombination of x^1 and x^2, whose components are corresponding components of either x^1 or x^2. Then we produce two children as follows:

$$c^1 := x^1 + \alpha_1 \cdot (P(x^1, x_c) - x^1), \quad c^2 := x^2 + \alpha_2 \cdot (P(x^2, x_c) - x^2),$$

where $\alpha_1, \alpha_2 \in (0, 1)$ are random scalars.
Mutation. We choose some component of a newly produced child and change it randomly in a certain range.

Now we state our evolutionary algorithm, called Algorithm EA, for problem (4.3.2), thereby solving the general VIP (4.3.1) with a bounded polyhedral constraint.

Algorithm 4.3.1. *Algorithm EA*

1. Initialization. *Choose a population size M and fix parameters $l_N, l_s, \bar{N}, \eta, \mathrm{NF}_{max}, \varepsilon > 0$. Construct an initial population set $\mathcal{P}$ and let the initial fitness function be the original objective function of (4.3.2),*

$$f_c(x) := \bar{\theta}(x).$$

Evaluate the trial points in $\mathcal{P}$ and order them according to their fitness function values so that x^1 is the best solution and x^M is the worst, i.e.,

$$f_c(x^1) \leq f_c(x^2) \leq \cdots \leq f_c(x^M).$$

Set the generation counters $t := 1$ and $s := 0$.

2. Parents Pool Generation. *Generate a parents pool $\mathcal{P}'$, which consists of all different pairs from the population set $\mathcal{P}$.*

3. Crossover and Mutation. *Select a pair (p^1, p^2) from the parents pool $\mathcal{P}'$. Apply the Crossover and Mutation Procedure to the pair (p^1, p^2) to obtain two new trial points c^1, c^2.*

4. Survival Selection. *If the newly produced point c^1 or c^2 is better than any point $\bar{p}$ in the population, then replace $\bar{p}$ by that new point. Delete the pair (p^1, p^2) from the parents pool $\mathcal{P}'$. If $\mathcal{P}' = \emptyset$, then reorder the population set according to their fitness function value and let*

$$N := min\{s, \bar{N}\},\ B := \{b^1, b^2, \ldots, b^N\} \leftarrow \{x^1, b^1, \ldots, b^{(N-1)}\},\ s := s + 1$$

and go to Step 5; otherwise go to Step 3.

5. Intensification. *If, during the last $\bar{N}$ generations of evolution, the fitness function has not been modified and the best point in the population set has not been improved enough, i.e.,*

$$s \geq \bar{N} \text{ and } |f_c(b^{\bar{N}}) - f_c(b^1)| \leq \eta(1 + f_c(b^1)),$$

then choose $x^1, x^2, ..., x^{l_N} \in \mathcal{P}$ and for each x^i, $i = 1, 2, ..., l_N$, perform a local search on the original objective function $\bar{\theta}(x)$:

$$\bar{x}^i \longleftarrow \text{Local Search } (\bar{\theta}(x), x^i, l_s),$$

where l_s is the maximum number of steps in the local search. If $f_c(x^i) < f_c(\bar{x}^i)$, i.e, after the local search the current fitness function value increases, then construct a new fitness function by

$$f_c(x) := f_c(x) \cdot exp\Big(\frac{1}{\|x - x^i\|^2}\Big).$$

Otherwise, let $\mathcal{P} := \mathcal{P} \cup \{\bar{x}^i\} \backslash \{x^i\}$. If the fitness function is modified at least once during the above procedure, then set $s := 1$. Reorder the points in the population set according to *their fitness function values and let $t := t + 1$. Go to Step 6.*

6. Stopping Condition. *If $f_c(x^1) < \varepsilon$ or the number of function evaluations exceeds the pre-specified limit NF_{max}, then terminate the algorithm and refine the global solution x^1 by some local search method.*

If the fitness function of the problem has not been modified and $x^1 \in \mathcal{P}$ has not been improved enough during the last $\bar{N}$ generations of evolution and a local search, i.e.,

$$s \geq \bar{N} \text{ and } |f_c(b^{\bar{N}}) - f_c(x^1)| \leq \eta(1 + f_c(x^1)),$$

then construct a new fitness function by

$$f_c(x) := f_c(x) \cdot exp\Big(\frac{1}{\|x - x^1\|^2}\Big)$$

and set $s := 1$. Otherwise, let $B := \{b^1, b^2, \ldots, b^{\bar{N}}\} \leftarrow \{x^1, b^1, \ldots, b^{(\bar{N}-1)}\}$. Proceed to Step 2 with $(f_c(x), \mathcal{P})$.

4.4 Numerical Experiments

The performance of the restricted-step Josephy-Newton method was tested on a number of test problems. The lack of test problems for general variational inequality problems enforces us to generate test problems by ourselves. We did this task by modifying, i.e., adding some extra constraints to, well known mixed complementarity test problems in the MCPLIB library [14]. The test problems used in our numerical experiments are included in the Appendix A. The programming code for the algorithm was written in MATLAB and run on a computer Pentium 4 Microprocessor.

The algorithm is terminated if one of the following two conditions is satisfied:

$$\theta(x^*) \leq \varepsilon_1, \quad \|x^* - \Pi_S(x^* - \nabla\theta(x^*))\| \leq \varepsilon_2.$$

In the implementation, we have used the following parameter settings:

$$\beta = 0.5,\ \sigma = 0.5,\ \gamma = 0.49,\ p = 2.1,\ \rho = 0.5,\ \delta_{min} = 0.2n,\ \delta_{max} = 10\delta_{min},\ i_{min} = -10$$

and

$$\varepsilon_1 = 10^{-12}, \quad \varepsilon_2 = 10^{-6}.$$

The parameter settings of the evolutionary algorithm EA used to solve sub-VIPs are displayed in Table 4.1. The last two parameters in Table 4.1 are used in the stopping conditions of EA, and they both depend on the current objective value in the main iteration, where n is the dimension of the problem and x^k is the current point of the main iteration. In the early stage of the main iterations, i.e, when $\theta(x^k)$ is relatively large, solving subproblems with high precision is not so effective from the viewpoint of computational expense. As the iteration process converges to a solution, the tolerance ε_{EA} in EA will decrease, while the maximum number of function evaluations allowed will increase.

Table 4.1: Parameter Settings.

Parameters	Definition	Value
M	number of elements in population	10
l_N	number of best points for which local search is used	1
l_s	maximum number of steps in local search	20
$\bar{N}, \eta$	parameters controlling local search in EA	3, 10^{-3}
ε_t, ρ_t	tunneling parameters	0.1, 2
ε_{EA}	tolerance parameters for the objective function in EA	$\min\{10^{-6}, 10^{-2} \cdot \theta(x^k)\}$
NF_{max}	maximum number of function evaluations in EA	$\mathrm{mid}\{100n, 400n, \frac{400n}{\theta(x^k)^{0.25}}\}$

The results for the test problems are shown in Table 4.2. The columns in this table have the following meanings:

Problem:	name of the test problem,
n:	dimension of the test problem,
t:	number of main iterations,
N_θ:	number of rJN directions accepted by the condition (4.2.4),
N_d:	number of rJN directions accepted by the condition (4.2.6),
N_{mon}:	number of monotone sub-VIPs solved,
N_{inact}:	number of sub-VIPs where the constraint $\|x - x^k\| \leq \delta$ was inactive at the solution,
N_{unit}:	number of times rJN unit stepsize was taken,
$\theta(x^*)$:	regularized gap function value at the obtained solution,
NF:	number of function evaluations,
NF_{EA}:	number of function evaluations required to solve the sub-VIPs by EA.

As shown in Table 4.2, the main termination criterion

$$\theta(x^*) \leq \varepsilon_1$$

was satisfied for all test problems. Moreover, the number of iterations was quite small overall and the solutions of sub-VIPs were used very often. All the problems except *choi** made use of the directions determined from the solution of sub-VIPs. As for the problem *choi**, the method solved sub-VIPs 8 times, and 7 of them gave successful directions. The column 6

Table 4.2: Numerical Experiments for the Restricted-Step Josephy-Newton Method.

Problem	n	t	N_θ	N_d	N_{mon}	N_{inact}	N_{unit}	$\theta(x^*)$	NF	NF_{EA}
badfree*	5	3	3	0	0	2	3	4.5391e-18	4	6556
choi*	13	8	7	0	5	4	7	7.4270e-13	19	29095
explcp*	16	12	11	1	0	0	12	8.8818e-16	13	4527
josephy*	4	4	4	0	0	3	4	2.8813e-13	5	3968
kojshin*	4	4	4	0	0	3	4	1.4279e-16	5	2979
mathinum*	3	8	6	2	0	5	8	2.9116e-16	9	7944
mathisum*	4	10	8	2	0	3	9	6.0942e-16	12	11321
nash*	10	8	8	0	8	6	8	1.0027e-13	9	18871

(N_{mon}) shows the number of monotone sub-VIPs we encountered. Most of the test problems have non-monotone sub-VIPs, while every sub-VIP of the problem *nash** was monotone.

This suggests that the problem *nash** itself is monotone. The column 7 (N_{inact}) shows the number of sub-VIPs that had solutions interior to additional box constraints. The column 8 (N_{unit}) gives the number of times the unit step size or a longer step size was accepted in the direction determined by the solutions of sub-VIPs. For all problems except *mathisum** and *choi**, such a step size was accepted all the times.

The last two columns of the table show the number of function evaluations required to solve all the sub-VIPs. The column *NF* shows the number of actual function evaluations of F, while NF_{EA} shows the number of merit function evaluations for sub-VIPs (4.2.2). Since the evaluation of those merit functions requires to solve a linear programming problem, these numbers equal the number of linear programming problems solved.

4.5 Conclusion

In this chapter we have proposed a practically implementable, globally convergent and locally superlinearly convergent method for solving the general variational inequality problem with polyhedral constraints. Our restricted-step Josephy-Newton method is a modification of the classical Josephy-Newton method and it solves the linearized sub-VIPs with additional box constraints. Moreover, an evolutionary algorithm for solving those sub-VIPs is presented that can effectively deal with bounded polyhedral constraints. Numerical results show that the proposed approach is able to solve various test problems of non-monotone VIPs successfully.

Chapter 5

Lipschitzian Branch and Bound Method for VIPs

5.1 Introduction

In this chapter, we develop another deterministic global optimization approach for solving general VIP. Consider the following variational inequality problem:
to find a vector $x^* \in S$ such that

$$\langle F(x^*), x - x^* \rangle \geq 0, \quad \forall x \in S. \tag{5.1.1}$$

We employ a branch and bound method for an equivalent global optimization problem to VIP. The branch and bound method was first proposed by Land and Doig in 1960 for linear programming [53], since then the usage of this method expanded to other fields. Nowadays, the branch and bound method is regarded as one of the best deterministic methods of global optimization. The basic idea of branch and bound method is that it divides the searching region into sub-regions on which the upper and lower bounds of the function can be determined. Then the regions with higher lower bounds than an upper bound of some other region can be safely discarded from the computation since those regions do not include the solution, and the dividing process continues on the remaining sub-regions themselves. The iteration stops when the lowest upper bound of sub-regions equals an upper bound of corresponding region and any element in that region can be regarded as a solution.

In our approach, we use the branch and bound method from Lipschitzian optimization side. Under some mild conditions, we show that some of the merit functions introduced in Chapter 2 have Lipschitzian properties. Then the Lipschitzian condition gives us upper and lower bounds for the merit function on a certain set.

Using the Lipschitzian branch and bound method for VIP (5.1.1) has several advantages. First of all, theoretical convergence of the method is guaranteed. Secondly, the equivalent global optimization problem to VIP has the zero global minimum value. This means we have a natural lower bound for the objective function, and when determining the useless subregions instead of comparing the lower bounds of regions with the highest upper bound, we can just compare them with zero. A branch and bound algorithm was also developed for linear complementarity problem by Al-Khayyal [1], but in a quite different manner.

This chapter is organized as follows. In Section 5.2, we consider the gap function and the natural residual function, and show their Lipschitzian properties. In Section 5.3, we describe a global optimization method based on a branch and bound method for solving the VIP. In Section 5.4, we present some numerical results with the proposed method for some test problems. Section 5.5 concludes the chapter.

5.2 Lipschitz Continuity

Consider the VI(S, F) defined by (5.1.1), where $F : \Re^n \to \Re^n$ is a continuous, but not necessarily monotone, mapping and S is a nonempty closed convex set in $\Re^n$.
In Chapter 2, we have seen that with the following merit functions, the VIP can be reformulated as a global optimization problem with zero global minimum value.

- Gap function:

$$\theta_{\text{gap}}(x) = \sup_{y \in S} \langle F(x), x - y \rangle. \tag{5.2.1}$$

- Natural residual function:

$$\theta_{\text{nat}}(x) = \|x - \Pi_S(x - F(x))\|, \tag{5.2.2}$$

  where $\Pi_S(z) := \operatorname*{argmin}_{y \in S} \|z - y\|$ denotes the projection of point z on S.

Assume that F is Lipschitz continuous on the set S, i.e., there exists a constant $L_F > 0$ such that

$$\|F(x) - F(y)\| \le L_F \|x - y\|, \ \forall x, y \in S.$$

Moreover, we assume that the set S is compact. Under these assumptions, the VIP (5.1.1) is guaranteed to have at least one solution [17]. Denote by L_1 and L_2 the global maximum values of continuous functions $\|F(y)\|$ and $\|x-y\|$, respectively, over the compact set S, i.e.,

$$L_1 = \max_{y\in S}\|F(y)\|,\ L_2 = \max_{x,y\in S}\|x-y\|, \tag{5.2.3}$$

where $\|\cdot\|$ denotes the Euclidean norm.

Lemma 5.2.1. *If F is Lipschitz continuous on S with constant L_F and S is a compact set, then the gap function θ_{gap} defined by (5.2.1) is Lipschitz continuous with constant $L_{gap} := L_F L_2 + L_1$ on S.*

Proof. For $\forall x, y \in S$, we have

$$\begin{aligned}
\theta_{\text{gap}}(x) - \theta_{\text{gap}}(y) &= \max_{z\in S}\langle F(x), x-z\rangle - \max_{z\in S}\langle F(y), y-z\rangle \\
&\le \max_{z\in S}[\langle F(x), x-z\rangle - \langle F(y), y-z\rangle] \\
&= \max_{z\in S}[\langle F(x)-F(y), x-z\rangle + \langle F(y), x-y\rangle] \\
&\le \max_{z\in S}\|F(x)-F(y)\|\|x-z\| + \|F(y)\|\|x-y\| \\
&\le (L_F\max_{z\in S}\|x-z\| + \|F(y)\|)\|x-y\| \\
&\le (L_F L_2 + L_1)\|x-y\| = L_{\text{gap}}\|x-y\|.
\end{aligned}$$

On the other hand, we have

$$\begin{aligned}
\theta_{\text{gap}}(x) - \theta_{\text{gap}}(y) &= \max_{z\in S}\langle F(x), x-z\rangle - \max_{z\in S}\langle F(y), y-z\rangle \\
&\ge -\max_{z\in S}[\langle F(y), y-z\rangle - \langle F(x), x-z\rangle] \\
&= -\max_{z\in S}[\langle F(y)-F(x), y-z\rangle + \langle F(x), y-x\rangle] \\
&\ge -(\max_{z\in S}\|F(x)-F(y)\|\|y-z\| + \|F(x)\|\|x-y\|) \\
&\ge -(L_F\max_{z\in S}\|y-z\| + \|F(x)\|)\|x-y\| \\
&\ge -(L_F L_2 + L_1)\|x-y\| = -L_{\text{gap}}\|x-y\|.
\end{aligned}$$

Hence we conclude that

$$|\theta_{\text{gap}}(x) - \theta_{\text{gap}}(y)| \le L_{\text{gap}}\|x-y\|,\ \forall x, y \in S.$$

This completes the proof.

Lemma 5.2.2. *If F is Lipschitz continuous on S with constant L_F and S is a compact set, then the natural residual function θ_{nat} defined by (5.2.2) is Lipschitz continuous with constant $L_{nat} := L_F + 2$.*

Proof. Let us take $\forall x, y \in S$ and check the Lipschitz condition:

$$\begin{aligned} |\theta_{\text{nat}}(x) - \theta_{\text{nat}}(y)| &= |\|x - H(x)\| - \|y - H(y)\|| \\ &\leq \|(x - H(x)) - (y - H(y))\| \\ &\leq \|x - y\| + \|H(x) - H(y)\| \\ &= \|x - y\| + \|\Pi_S(x - F(x)) - \Pi_S(y - F(y))\|. \end{aligned}$$

Since the projection operator is non-expansive, we have

$$\begin{aligned} |\theta_{\text{nat}}(x) - \theta_{\text{nat}}(y)| &\leq \|x - y\| + \|\Pi_S(x - F(x)) - \Pi_S(y - F(y))\| \\ &\leq \|x - y\| + \|(x - F(x) - (y - F(y))\| \\ &\leq \|x - y\| + \|x - y\| + \|F(x) - F(y)\| \\ &\leq 2\|x - y\| + L_F\|x - y\| = L_{\text{nat}}\|x - y\|. \end{aligned}$$

The proof is completed.

Remark 5.2.3. *It is also possible to show the Lipschitz continuity of the differentiable merit function (2.3.5) and one may combine the Lipschitz optimization approach with some gradient related method.*

5.3 Lipschitzian Branch and Bound Method

When F is Lipschitz continuous, Lemmas 5.2.1 and 5.2.2 allow us to reduce the VIP to a Lipschitz optimization problem. Although Lipschitz optimization problems with box constraints have been much studied [30, 31, 37, 38, 75], application to VIP is scarce. For this reason, we restrict ourselves to VIP with box constraints. More specifically, we consider the VI(D, F) with

$$D := \{y \in R^n | l_i \leq y_i \leq u_i,\ i = 1, ..., n\}.$$

The best known and most studied algorithm for Lipschitz optimization problems is the branch and bound method [37, 38]. In this method, the feasible set D is subsequently divided into more and more refined parts, over which lower and upper bounds of the minimum objective function value are determined. Parts of the feasible set D with lower bounds exceeding *zero* (global minimum value of merit function) are discarded from further consideration, since these parts cannot contain a solution of VIP. Negative lower bounds may also be helpful in selecting parts of D that have a good chance to contain a solution, as we consider below.

Description of the branch and bound method is given as follows [37, 38]:

Branch and bound method

Let us define a rectangle $R = \{o, d, \lambda, m, \upsilon\}$ with origin $o(R)$, diagonal endpoint $d(R)$, Lipschitz lower bound $\lambda(R)$, best known solution $m(R) = \operatorname*{argmin}_{x \in K(R)} \theta(x)$ and best known upper bound $\upsilon(R) = \min_{x \in K(R)} \theta(x)$, where $K(R) = \{o(R), d(R), (o(R) + d(R))/2, \dots\}$ is a set of control points associated with the rectangle. In particular, for a function θ satisfying the Lipschitz condition with constant L, $\lambda(R)$ is computed as

$$\begin{aligned}\lambda(R) = \max\{ \quad & \max(\theta(o(R)), \theta(d(R))) - L\|d(R) - o(R)\|, \\ & \theta((o(R) + d(R))/2) - L/2\|d(R) - o(R)\|, \dots\}.\end{aligned}$$

The priority queue $\Omega := \{R^1, R^2, \dots\}$ of rectangles which may contain a solution will also be used in the method and the priority may be defined in many ways. In our numerical experiments, we use two different priorities, *diving* and *the most expansive bound.*

diving strategy - the smaller the lower bound value $\upsilon(R)$, the better the rectangle R.

the most expansive bound strategy - the higher the difference between upper and lower bounds $\lambda(R) - \upsilon(R)$, the better the rectangle R.

Algorithm 5.3.1. *Algorithm LBB*

1.Initialization. $R = D$; $\Lambda_0 = \lambda(R)$; $\Upsilon_0 = \upsilon(R)$; $s_0 = m(R)$; $\Omega = \{R\}$; $k := 0$;

2. Stopping Condition. *If* $\Upsilon_k < \Lambda_k + \varepsilon$, *then terminate the algorithm and* s_k *is a solution. Otherwise, go to the next step.*

3. Selection. *Select a candidate* R *from the priority queue* Ω *according to the one of the strategies.*

4. ***Branching.*** *Divide R into two rectangle (R_1, R_2) by halving it.*

5. Bounding. *Determine $\lambda(R_i), \upsilon(R_i), m(R_i)$ for $i = 1, 2$ and insert R_1, R_2 in the priority queue Ω. Update the upper and the lower bounds, and the current solution:*

$$\begin{aligned}
\Lambda_{k+1} &:= \max\{\Lambda_k, \lambda(R_1), \lambda(R_2)\},\\
\Upsilon_{k+1} &:= \min\{\Upsilon_k, \upsilon(R_1), \upsilon(R_2)\},\\
s_{k+1} &:= \mathit{argmin}\{\theta(s)|\ s \in \{s_k, m(R_1), m(R_2)\}\}.
\end{aligned}$$

Set $k := k + 1$ and go to Step 2.

The efficiency of the branch and bound algorithm substantially depends on the Lipschitz constant. There are some procedures for estimating the Lipschitz constant, such as those proposed by Strongin [100] and Meewella and Mayne [75]. Such constants may also be estimated by using interval analysis [90, 91].

Since the minimum value of the merit function is known to be zero, we concentrate more on decreasing the upper bound Υ_k than decreasing the difference $\Upsilon_k - \Lambda_k$ and we use the stopping condition $\Upsilon_k < \varepsilon$ instead of $\Upsilon_k - \Lambda_k < \varepsilon$. Moreover, at every pre-specified number of, say three, iterations, we select the next rectangle R from the priority queue Ω with the lowest upper bound $\upsilon(R)$, instead of selecting it by the diving strategy or the most expansive bound strategy.

Convergence of the algorithm is ensured by the following theorem.

Theorem 5.3.2. *([37]) If $\varepsilon = 0$ in the branch and bound algorithm and an infinite sequence is generated, then*

$$\Lambda = \lim_{k\to\infty} \Lambda_k = \lim_{k\to\infty} \theta(s_k) = \lim_{k\to\infty} \Upsilon_k = \Upsilon$$

and every accumulation point s^* of the sequence $\{s_k\}$ is an optimal solution of the optimization problem

$$\mathit{min}\ \theta(x)\ \mathit{s.t}\ x \in D.$$

5.4 Numerical Experiments

In this section, we implement the branch and bound method for two types of test problems. The first one is the affine variational inequality problem in which F is given by $F(x) = Px+q$, where $P \in \Re^{n\times n}$ and $q \in \Re^n$. For this kind of problems, it is not difficult to compute Lipschitz constants for merit functions. In fact, we have

$$\begin{aligned} L_1 &= \max_{x\in D} \|F(x)\| \\ &= \max_{x\in D} \|Px+q\| \\ &\leq \min\{\|Pl+q\| + \|P\|\|u-l\|, \|P\|\|c(l,u)\|\}, \end{aligned}$$

where $c(l,u)$ is a vector with components $c_i(l,u) = \max\{|(u+P^\dagger q)_i|, |(l+P^\dagger q)_i|\},\ i = 1,...,n$, and $P^\dagger$ stands for the Moore-Penrose inverse of P. So by letting

$$L_F = \|P\| \text{ and } L_2 = \|u-l\|,$$

we have from Lemmas 5.2.1 and 5.2.2,

$$\begin{aligned} L_{\text{gap}} &= L_F L_2 + L_1 \\ &\leq \|P\|\|u-l\| + \min\{\|Pl+q\| + \|P\|\|u-l\|, \|P\|\|c\|\} \end{aligned}$$

and

$$L_{\text{nat}} = L_F + 2 = \|P\| + 2.$$

Thus, L_{gap} depends substantially on the diameter of the set D. Therefore, for a rectangle $R \subset D$, it is sometimes computationally helpful to use different estimates of Lipschitz constant instead of using only L_{gap}. For example, in the interval $[l^i, u^i] \subset [l,u]$, Lipschitz constant for the gap function θ_{gap} may be estimated as follows:

$$\begin{aligned} L^i_{gap} &= L_F L_2 + L^i_1 \\ &\leq \|P\|\|u-l\| + \min\{\|Pl^i+q\| + \|P\|\|u^i-l^i\|, \|P\|\|c(l^i,u^i)\|\}. \end{aligned}$$

To use this formula without trouble, however, one has to have some guarantee, at least, that $L^i_{gap} < L_{\text{gap}}$.

The other type of problems we have tested are non-smooth nonlinear complementarity problems involving a mapping of the form $F(x) = P|x| + q$, where $P \in \Re^{n\times n}$, $q \in \Re^n$ and $|x|$ denotes the vector with components $|x_i|,\ i = 1,\ldots,n$.

When the branch and bound method is executed, one often obtains a good feasible point in an early stage and most of the computational effort is spent to verify its quality or prove optimality. In addition to the stopping condition $\Upsilon_k < \varepsilon$, we use another stopping condition - upper limit NF_{max} for the number of function evaluations. Once the number of function evaluations reaches NF_{max}, we terminate the algorithm. In our computational experiments, we set

$$\varepsilon = 10^{-6} \text{ and } NF_{max} = 10000n,$$

where n is the dimension of the problem.

We solved six VIPs in the Appendix B up to dimension 10. The program was coded in Matlab and run on a computer Pentium 4 Microprocessor. Tables 5.1 and 5.2 show the numerical results. Columns of the tables have the following meaning.

Problem:	name of the test problem,
n:	dimension of the test problem,
func:	merit function used,
L :	Lipschitz constant of the merit function,
$\theta^* \approx$:	the function value at the obtained solution,
$\lvert\Omega\rvert$:	the number of elements in the priority queue, i.e., the number of remaining rectangles which possibly contain a solution,
NF :	the number of function evaluations.

The numerical results reported in Tables 5.1 and 5.2 show that the branch and bound method was able to solve all the test problems. As we see in the last columns of both tables, the number of function evaluations required to find an approximate solution of each test problem is generally not so large. Particularly, for problems (P1),(P3),(P5) and (P6), the method works very well. Problems (P3) and (P6) have a solution at a vertex of the box constraint set, while problems (P1) and (P5) happen to have a solution at a vertex of some sub-rectangle emerged in an earlier stage of branching. Moreover, for all problems, the Lipschitz constant L_{nat} of θ_{nat} is much smaller than that of θ_{gap}. This is because of the fact that L_{gap} largely depends on the size of the constraint set, while L_{nat} does not depend on it. The stopping condition $\Upsilon_k < \varepsilon$ was satisfied for most of the test problems and approximation of solutions was reasonably accurate. Only for the θ_{gap} reformulations of problems (P2), (P4) and (P6), the maximum number of function evaluations NF_{max} was used to terminate the iteration.

In Table 5.1, we see that, for the diving strategy, the θ_{nat} reformulation works better than the θ_{gap} reformulation for all problems but (P3). We can see this fact in the last two columns of Table 5.1, i.e., the number of function evaluations and the number of remaining rectangles in the priority queue Ω for the θ_{nat} reformulation are smaller than those of the θ_{gap} reformulation. The reason for this may be explained by the fact that L_{nat} is usually much smaller than L_{gap}. Moreover, for the θ_{gap} reformulation, the method with the diving strategy was not able to solve problem (P6), while the θ_{nat} reformulation was solved easily

Table 5.1: Numerical Experiments of the *Algorithm LBB* with the Diving Strategy

Problem	n	*func*	L	$\theta^* \approx$	$\|\Omega\|$	*NF*
$P1$	2	θ_{gap}	9.8994	4.7684e-7	184	1464
		θ_{nat}	4.0000	6.7435e-7	117	560
$P2$	3	θ_{gap}	31.7177	1.9073e-6	3275	30000
		θ_{nat}	5.3027	4.7684e-7	471	11668
$P3$	4	θ_{gap}	20.1923	7.6294e-7	131	532
		θ_{nat}	5.8587	8.2591e-7	483	2056
$P4$	3	θ_{gap}	49.1393	7.5338e-5	3238	30000
		θ_{nat}	7.3629	8.5963e-7	788	6448
$P5$	5	θ_{gap}	55.4700	9.2387e-7	232	928
		θ_{nat}	12.7750	6.7435e-7	219	876
$P6$	10	θ_{gap}	391.0476	36.7296e-0	12688	100000
		θ_{nat}	17.6376	9.6294e-7	419	1676

by the same strategy. This shows that the diving strategy is not suitable for problem (P6) with the θ_{gap} reformulation. The θ_{gap} reformulation of problems (P2) and (P4) required the upper limit NF_{max} for the number of function evaluations to terminate the iteration, while the the stopping condition $\Upsilon_k < \varepsilon$ was satisfied before the number of function evaluations reached its upper limit NF_{max} in all the other cases.

In Table 5.2, which shows the results for the most expansive bound strategy, we see that the θ_{nat} reformulation works better for problems (P1), (P2), (P4) and (P6) in terms of both the numbers of function evaluations and remaining rectangles in Ω, while the θ_{gap} reformulation performs well for problems (P3) and (P5). The θ_{gap} reformulations of problems (P2) and (P4) used NF_{max} to terminate the iteration, while the stopping condition $\Upsilon_k < \varepsilon$ was satisfied before the number of function evaluations reached its upper limit NF_{max} in all

Table 5.2: Numerical Experiments for the *Algorithm LBB* with the Most Expansive Bound Strategy

Problem	n	*func*	L	$\theta^* \approx$	$\|\Omega\|$	*NF*
$P1$	2	θ_{gap}	9.8994	4.7684e-7	133	1436
		θ_{nat}	4.0000	6.7435e-7	74	456
$P2$	3	θ_{gap}	31.7177	1.9073e-6	1320	30000
		θ_{nat}	5.3027	4.7684e-7	171	12200
$P3$	4	θ_{gap}	20.1923	7.6294e-7	127	580
		θ_{nat}	5.8587	8.2591e-7	385	2952
$P4$	3	θ_{gap}	49.1393	7.5338e-5	1432	30000
		θ_{nat}	7.3629	8.5963e-7	375	6552
$P5$	5	θ_{gap}	55.4700	9.2387e-7	211	848
		θ_{nat}	12.7750	6.7435e-7	226	904
$P6$	10	θ_{gap}	391.0476	7.6722e-7	537	2148
		θ_{nat}	17.6376	9.4619e-7	443	1772

the other cases. For problems (P2) and (P4), the θ_{nat} reformulation works much better than the θ_{gap} reformulation, as we can see in the last two columns of Table 5.2. We may deduce that for problems having a solution in a general position, Lipschitz constant plays a more important role for effective computation.

Finally, we discuss some comparison of results shown in the two tables. For the θ_{gap} reformulation, the method with the most expansive bound strategy works consistently better than the method with the diving strategy. Comparing the number of function evaluations for the reformulation θ_{nat} in Tables 5.1 and 5.2, we observe that the diving strategy works better for all problems except (P1). With the most expansive bound strategy for θ_{gap}, the method could solve problem (P6) without trouble. For problems (P1)-(P4), the most expansive bound strategy leaves less rectangles in the priority queue Ω.

5.5 Conclusion

In this chapter, we have considered the possibility of applying Lipschitz optimization approach to general VIPs. Under some suitable conditions, we have shown that the equivalent global optimization reformulations of VIPs satisfy the Lipschitz condition, and presented a branch and bound algorithm designed particularly for the latter problem. Numerical results for some test problems are reported to illustrate the behavior of the proposed approach.

Chapter 6

Conclusions

In this thesis, we have developed several global optimization based methods for solving the general VIP. From classic methods to heuristics, we have studied the possibility of using global optimization methods for solving the equivalent optimization problem to VIP. The main advantage of the equivalent global optimization reformulation to VIP is the global minimum value of the equivalent problem is known to be zero. Since there are no general criteria for global optimal solutions, most global optimization methods suffer from the difficulty in determining when to terminate the algorithm. The information about the global minimum value is crucial for our methods and it greatly enhances the performance of the methods. Moreover, directly dealing with the equivalent global optimization problem enables us to deal with VIP under very general settings.

Below, we describe the main contributions of the thesis chapter by chapter.

In Chapter 3, we have proposed a heuristic method for solving the general VIP. Special procedures such as adaptive fitness function techniques and the population update rules have been introduced. The proposed method is capable of detecting as many as possible, hopefully all solutions of the general VIP. Numerical experiments for some well known test problems show that the method works very well in practice.

In Chapter 4, we have proposed a modified and practically implementable version of the classical Josephy-Newton method. By imposing some extra bounds to the traditional Josephy-Newton subproblems, we ensure the existence of solutions to subproblems. Under appropriate conditions, global and locally superlinear convergence properties of the proposed method are established. Furthermore, we have developed an evolutionary algorithm for solving general VIPs with bounded polyhedral constraints, which can be used to solve the Josephy-Newton subproblems when the constraint set is polyhedral. Numerical experiments show that the proposed approach is able to solve various test problems successfully.

In Chapter 5, we have proposed a deterministic approach for solving the general VIP. Assuming the mapping F is Lipschitz continuous, we have shown that some of merit functions are Lipschitz continuous. Then the Lipschitzian branch and bound method has been employed for solving the merit function minimization problem. The known global minimum value of zero gives us the natural lower bound, so we have concentrated only the upper bound minimization part. Numerical experiments for some test problems are reported to illustrate the behavior of the proposed approach.

Finally, we discuss some future research directions concerning the general VIP.

In Chapter 3, we have considered an evolutionary algorithm for solving the general VIP. Similarly, it is interesting to study other heuristic methods such as Tabu search and the simulated annealing for solving the general VIP. Combining these methods with adaptive fitness function techniques, new methods can be developed.

Another issue left without answer in Chapter 3 is to construct an exact tunneling function. The tunneling function employed in the adaptive function techniques is somewhat heuristic and we have to determine the appropriate tunneling parameters. Establishing an exact tunneling function would be very helpful, not only in solving VIP, but also in other optimization fields that use the tunneling function.

In Chapter 4, the Josephy-Newton method linearizes only a part of the original VIP and it results in subproblems with the same constraint set. When we use some polyhedral approximation to the constraint set, the existence of a solution to subproblems is in danger even if the original constraint set is compact. By imposing some extra bound to this linearized constraint sets we may consider a fully linearized version of the Josephy-Newton method with tractable subproblems.

The next direction for general VIP is to consider large scale problems. When the size of the problem is very big, most of the existing methods face the computational difficulty. The parallel version of our methods or some decomposition methods will be useful to deal with such problems.

Appendix A

Test Problems for the Restricted-Step Josephy-Newton Method

The test problems used in the numerical experiments for the restricted-step Josephy-Newton method are displayed here. We have used some standard test problems from the MCPLIB [14] to construct our test problems by modifying the constraint sets. In particular, modifications have been made in such a way that the solutions of the original test problems become infeasible to the new test problems.

- **badfree***

  $n = 5$ and $F(x) = Mx + q$, where

$$M = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 \end{pmatrix}, \quad q = \begin{pmatrix} -1 \\ -1 \\ -0.5 \\ -0.5 \\ -1 \end{pmatrix}.$$

  In this example, the original constraint set X was

$$X := \{x \in \Re^n |\ x_i \geq 0,\ i = 1, \dots, n-1\}.$$

  We have modified this by adding two linear constraints:

$$S = \{x \in X |\ \sum_{i=1}^{n} x_i \leq n, \quad \sum_{i=1}^{n} i x_i \geq n+1\}.$$

  We use the feasible starting point $x^0 = \left(\frac{n-1}{n}, \frac{n-1}{n}, \dots, \frac{n-1}{n}\right)^T$.

- **choi***
  $n = 13$ and $F(x) = (\nabla_{x_j} \Pi_j(x))_{j=1}^n$, where

$$\Pi_j(x) = (x_j - C_j)\frac{Q}{I}\sum_{i=1}^{I} Pr_{ij},$$

with

$$Pr_{ij} = \frac{\exp\{-\chi DU_{ij}\}}{\sum\limits_{m=1}^{J} \exp\{-\chi DU_{im}\} + K}$$

and

$$DU_{ij} = \sum_{n=1}^{N} v_{in}(a_{jn} - b_{in})^2 + w_i x_j + b_i.$$

For the details of the coefficients given above, see [8]. In this example, the original constraint set X was

$$X := \{x \in \Re^n |\ l_i \le x_i \le u_i\},$$

where

$l = (0.4,\ 0.1328,\ 0.4,\ 0.1275,\ 0.0975,\ 0.1172,\ 0.1541,\ 0.4,\ 0.301,\ 0.4,\ 0.4,\ 0.26,\ 0.2383)$,

$u = (1.2,\ 0.3984,\ 1.2,\ 0.3825,\ 0.2925,\ 0.3516,\ 0.4623,\ 1.2,\ 0.903,\ 1.2,\ 1.2,\ 0.78,\ 0.7149)$.

We have modified this by adding two linear constraints:

$$S = \{x \in X |\ \sum_{i=1}^{n} x_i \ge 5, \quad \sum_{i=1}^{n} x_i \le 10\}.$$

We use the feasible starting point $x^0 = \dfrac{(l+u)}{2}$.

- **explcp***
  $n = 16$ and $F(x) = Mx + q$, where

$$M = \begin{pmatrix} 1 & 2 & \dots & 2 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 2 \\ 0 & \dots & 0 & 1 \end{pmatrix}, \quad q = \begin{pmatrix} -1 \\ \vdots \\ -1 \end{pmatrix}.$$

In this example, the original constraint set X was

$$X := \{x \in \Re^n |\ x_i \ge 0,\ i = 1, \dots, n\}.$$

We have modified this by adding two linear constraints:

$$S = \{x \in X | \sum_{i=1}^{n} x_i \geq 2, \quad \sum_{i=1}^{n} x_i \leq n\}.$$

We use the feasible starting point $x^0 = \left(\frac{1}{2}, \frac{1}{2}, \ldots, \frac{1}{2}\right)^T$.

- **josephy***

$n = 4$ and

$$F(x) = \begin{pmatrix} 3x_1^2 + 2x_2^2 + 2x_1x_2 + x_3 + 3x_4 - 6 \\ 2x_1^2 + x_2^2 + x_1 + 3x_3 + 2x_4 - 2 \\ 3x_1^2 + 2x_2^2 + x_1x_2 + 2x_3 + 3x_4 - 1 \\ x_1^2 + 3x_2^2 + 2x_3 + 3x_4 - 3 \end{pmatrix}.$$

In this example, the original constraint set X was

$$X := \{x \in \Re^n | \ x_i \geq 0, \ i = 1, \ldots, n\}.$$

We have modified this by adding two linear constraints:

$$S = \{x \in X | \sum_{i=1}^{n} ix_i \geq n, \quad \sum_{i=1}^{n} x_i \leq n - 1\}.$$

We use the feasible starting point $x^0 = \left(\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}\right)^T$.

- **kojshin***

$n = 4$ and

$$F(x) = \begin{pmatrix} 3x_1^2 + 2x_2^2 + 2x_1x_2 + x_3 + 3x_4 - 6 \\ 2x_1^2 + x_2^2 + x_1 + 10x_3 + 2x_4 - 2 \\ 3x_1^2 + 2x_2^2 + x_1x_2 + 2x_3 + 9x_4 - 9 \\ x_1^2 + 3x_2^2 + 2x_3 + 3x_4 - 3 \end{pmatrix}.$$

In this example, the original constraint set X was

$$X := \{x \in \Re^n | \ x_i \geq 0, \ i = 1, \ldots, n\}.$$

We have modified this by adding two linear constraints:

$$S = \{x \in X | \sum_{i=1}^{n} ix_i \geq n, \quad \sum_{i=1}^{n} x_i \leq n - 1\}.$$

We use the feasible starting point $x^0 = \left(\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}\right)^T$.

- **mathinum*** and **mathisum***

  Both *mathinum* and *mathisum* test problems originate from the Mathiesen's Walrasian equilibrium problem with

$$F(x) = \begin{pmatrix} -x_1 + x_2 + x_3 \\ x_4 - \dfrac{0.9(5x_2 + 3x_3)}{x_1} \\ 5 - x_4 - \dfrac{0.1(5x_2 + 3x_3)}{x_2} \\ 3 - x_4 \end{pmatrix}.$$

  The dimension of the problem is $n = 4$ and the original constraint set X is given by

$$X := \{x \in \Re^n |\ x_i \geq 0,\ i = 1, \ldots, n\}.$$

  (a). **mathinum***
  Setting $x_1 = 4.2$, we have *mathinum* test problem with dimension $n = 3$. We have modified the problem by adding two linear constraints:

$$S = \{x \in X |\ \sum_{i=1}^{n} x_i \geq 10, \quad \sum_{i=1}^{n} x_i \leq 10n\}.$$

  We use the feasible starting point $x^0 = (4, 4, 4)^T$.
  (b). **mathisum***
  Setting $\sum_{i=1}^{n-1} x_i = 1$, we have *mathisum* test problem with dimension $n = 4$. We have modified the problem by adding a linear constraint:

$$S = \{x \in X |\ \sum_{i=1}^{n} x_i \leq n\}.$$

  We use the feasible starting point $x^0 = \left(\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}\right)^T$.

- **nash***

  $n = 10$, $\gamma = 1.2$, $c = (5,\ 3,\ 8,\ 5,\ 1,\ 3,\ 7,\ 4,\ 6,\ 3)$, $L = (10,\ 10, \ldots,\ 10)$, $\beta = (1.2,\ 1,\ 0.9,\ 0.6,\ 1.5,\ 1,\ 0.7,\ 1.1,\ 0.95,\ 0.75)$ and

$$F(x) = \left(c_i + (L_i x_i)^{\frac{1}{\beta_i}} - p(x) + x_i \frac{p(x)}{\gamma s(x)}\right)_{i=1}^{n},$$

  where

$$s(x) = \sum_{j=1}^{n} x_j, \quad p(x) = 5000^{\frac{1}{\gamma}} s(x)^{-\frac{1}{\gamma}}.$$

In this example, the original constraint set X was

$$X := \{x \in \Re^n |\ x_i \geq 0,\ i = 1, \ldots, n\}.$$

We have modified this by adding two linear constraints:

$$S = \{x \in X |\ \sum_{i=1}^{n} x_i \geq 1, \quad \sum_{i=1}^{n} x_i \leq 4n\}.$$

We use the feasible starting point $x^0 = (1, 1, \ldots, 1)^T$.

Appendix B

Test Problems for the Lipschitzian Branch and Bound Method

The test problems used in the numerical experiments for the Lipschitzian branch and bound method are displayed here. The first three test problems, which are originally linear complementarity problems, are taken from [21] and are modified by introducing some box constraints.

- **Test Problem 1**

$$P1(n=2): \ F(x) = Px + q, \ D := \{x \in \Re^n | a \le x \le b\},$$

where

$$P = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}, \quad q = \begin{pmatrix} -1 \\ -1 \end{pmatrix}, \quad a = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \quad b = \begin{pmatrix} 1.5 \\ 1.5 \end{pmatrix}.$$

- **Test Problem 2**

$$P2(n=3): \quad F(x) = Px + q, \quad D := \{x \in \Re^n | a \le x \le b\},$$

where

$$P = \begin{pmatrix} 0 & -1 & 2 \\ 2 & 0 & -2 \\ -1 & 1 & 0 \end{pmatrix}, \ q = \begin{pmatrix} -3 \\ 6 \\ -1 \end{pmatrix}, \ a = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \ b = \begin{pmatrix} 2 \\ 2 \\ 2.5 \end{pmatrix}.$$

- **Test Problem 3**

$$P3(n=4): \quad F(x) = Px + q, \quad D := \{x \in \Re^n | a \le x \le b\},$$

where

$$P = \begin{pmatrix} 0 & 0 & 1 & 2 \\ 0 & 0 & 3 & 1.5 \\ 1 & 2 & 0 & 0 \\ 3 & 1.5 & 0 & 0 \end{pmatrix}, \; q = \begin{pmatrix} -0.1 \\ -0.1 \\ -0.1 \\ -0.1 \end{pmatrix}, \; a = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \; b = \begin{pmatrix} 2 \\ 2 \\ 2 \\ 2 \end{pmatrix}.$$

The other three test problems are nonlinear, non-smooth VIPs given as follows.

- **Test Problem 4**

$$P4(n=3): \quad F(x) = P|x| + q, \quad D := \{x \in \Re^n | a \leq x \leq b\},$$

where

$$P = \begin{pmatrix} -2 & -1 & -3 \\ 0 & -1 & 1 \\ 3 & 3 & 1 \end{pmatrix}, \; q = \begin{pmatrix} 2 \\ -1 \\ -1 \end{pmatrix}, \; a = \begin{pmatrix} -1 \\ -1.5 \\ -1.5 \end{pmatrix}, \; b = \begin{pmatrix} 1 \\ 2 \\ 2 \end{pmatrix}.$$

- **Test Problem 5**

$$P5(n=5): \quad F(x) = P|x| + q, \quad D := \{x \in \Re^n | a \leq x \leq b\},$$

where

$$P = \begin{pmatrix} 3 & 1 & -1 & 1 & -3 \\ -3 & 0 & -2 & -5 & 2 \\ -1 & -1 & 4 & 1 & 0 \\ 0 & -4 & 3 & 3 & 4 \\ 2 & -1 & 4 & 5 & 2 \end{pmatrix}, \; q = \begin{pmatrix} -3 \\ -1 \\ -3 \\ 5 \\ -1 \end{pmatrix}, \; a = \begin{pmatrix} -1 \\ 1 \\ 1 \\ 0 \\ 0 \end{pmatrix}, \; b = \begin{pmatrix} 0 \\ 2 \\ 2 \\ 1 \\ 1 \end{pmatrix}.$$

- **Test Problem 6**

$$P5(n=10): \quad F(x) = Px + q, \quad D := \{x \in \Re^n | a \leq x \leq b\},$$

where

$$P = 0.1 \cdot \begin{pmatrix} -3 & 21 & 15 & 43 & -45 & -45 & 36 & -49 & -5 & 38 \\ -7 & 10 & -35 & 26 & -38 & -20 & 14 & -9 & -13 & -43 \\ 8 & 16 & -3 & -19 & 20 & -18 & -10 & -26 & 27 & -11 \\ 39 & -16 & 25 & -36 & 5 & 16 & 6 & 5 & -5 & -22 \\ -22 & 37 & -13 & -3 & 39 & 42 & 22 & -46 & -32 & 18 \\ -26 & 15 & 11 & 9 & 43 & -21 & 46 & 27 & 39 & -13 \\ 48 & 36 & 43 & -2 & -4 & 10 & 29 & -12 & -41 & -29 \\ -47 & 9 & -5 & 5 & -30 & -50 & -36 & 36 & 47 & 16 \\ 16 & 49 & -18 & 23 & -33 & 43 & 2 & 38 & 1 & 19 \\ -41 & -47 & 25 & 16 & -8 & 48 & -46 & 15 & 43 & 3 \end{pmatrix},$$

$$q = (16.6,\ -18.8,\ -44.2,\ 33.0,\ 37.3,\ 20.7,\ 46.6,\ -41.8,\ 44.5,\ -42.4)^T,$$

$$a = (-1,\ -1,\ -1,\ -1,\ -1,\ -1,\ -1,\ -1,\ -1,\ -1)^T,\ \ b = (2,\ 2,\ 2,\ 2,\ 2,\ 2,\ 2,\ 2,\ 2,\ 2)^T.$$

Bibliography

[1] Al-Khayyal, F.A. (1987), An implicit enumeration procedure for the general linear complementarity problem, *Mathematical Programming Study*, 31, 1–20.

[2] Auchmuty, G. (1989), Variational principles for variational inequalities, *Numerical Functional Analysis and Optimization*, 10, 863–874.

[3] Auslender, A. (1976), *Optimisation: Methodes Numeriques*, Masson, Paris.

[4] Barron, C. and Gomez, S. (1991), The exponential tunneling method, *Reporte de Investigacio'n IIMAS*, 1 (3), 1–23.

[5] Bertsekas, D.P. (1995), *Nonlinear Programming*, Athena Scientific, Belmont, Massachusetts.

[6] Carey, M. (1977), Integrability and mathematical programming models: a survey and parametric approach, *Econometrica*, 45, 1957–1976.

[7] Chak, C.K. and G. Feng. (1995), Accelerated genetic algorithms: Combined with local search techniques for fast and accurate global search, *IEEE Int. Conf. on Evolutionary Computation ICEC95*, Perth, Australia, 378–383.

[8] Choi, S.Ch., Desarbo, W.S. and Harker, P.T. (1990), Product positioning under price competition, *Management Science*, 36, 175–199.

[9] Cottle, R.W. (1964), Nonlinear Programs with Positively Bounded Jacobians, Ph.D. dissertation, Department of Mathematics, University of California, Berkeley, CA.

[10] Cottle, R.W. (1966), Nonlinear programs with positively bounded Jacobians, *SIAM Journal on Applied Mathematics*, 14, 147–158.

[11] Cottle, R.W. and Dantzig, G.B. (1968), Complementary pivot theory of mathematical programming, *Linear Algebra and Its Applications*, 1, 103–125.

[12] Cottle, R.W., Habetler, G.J. and Lemke, C.E. (1970), Quadratic forms semi-definite over convex cones, in: H.W. Kuhn, ed., *Proceedings of the Princeton Symposium on Mathematical Programming*, Princeton University Press, Princeton, NJ, 551–565.

[13] De Jong, K.A. (2005), *Evolutionary Computation*, The MIT Press, Cambridge, MA.

[14] Dirkse, S.P. and Ferris, M.C. (1995), Mcplib: a collection of nonlinear mixed complementarity problems, *Optimization Methods and Software*, 5, 319–345. http://www.gams.com/mpec/mcplib.htm.

[15] Eaves, B.C. (1971), On the basic theorem of complementarity, *Mathematical Programming,* 1, 68–75.

[16] Enkhbat, R. (1996), An algorithm for maximizing a convex function over a simple set, *Journal of Global Optimization*, 8, 379–391.

[17] Facchinei, F. and Pang, J.S. (2003), *Finite-Dimensional Variational Inequalities and Complementarity Problems*, Volumes I and II, Springer-Verlag, New York.

[18] Fischer, A. (1992), A special Newton-type optimization method, *Optimization*, 24, 269–284.

[19] Fischer, A. (1995), A Newton-type method for positive-semidefinite linear complementarity problems, *Journal of Optimization Theory and Applications*, 86, 585–608.

[20] Fischer, A. (1995), On the superlinear convergence of a Newton-type method for LCP under weak conditions, *Optimization Methods and Software*, 6, 83–107.

[21] Floudas, C.A., Pardalos P. et al. (1999), *Handbook of Test Problems in Local and Global Optimization*, Kluwer Academic Publisher, London.

[22] Fukushima, M. (1996), Merit functions for variational inequality and complementarity problems, in: G. Di Pillo and F. Giannessi, eds., *Nonlinear Optimization and Applications*, Plenum Press, New York, 155–170.

[23] Fukushima, M. (1992), Equivalent differentiable optimization problems and descent methods for asymmetric variational inequality problems, *Mathematical Programming,* 53, 99–110.

[24] Fukushima, M. (2007), A class of gap functions for quasi-variational inequality problems, *Journal of Industrial and Management Optimization*, 3, 165–171.

[25] Gabay, D. and Moulin, H. (1980), On the uniqueness and stability of Nash-equilibria in noncooperative games, in: A. Bensoussan, P. Kleindorfer and C.S. Tapiero, eds., *Applied Stochastic Control in Econometrics and Management Science*, North-Holland, Amsterdam, 271–292.

[26] Ge, R. (1990), A filled function method for finding a global minimizer of a function of several variables, *Mathematical Programming*, 46, 191–204.

[27] Ge, R. and Qin, Y. (1987), A class of filled functions for finding global minimizers of a function of several variables, *Journal of Optimization Theory and Applications*, 52(2), 240-252.

[28] Goldberg, D.E. (1989), *Genetic Algorithm in Search, Optimization and Machine Learning*, Addison-Wesley, Reading, Mass.

[29] Gomez, S., Solorzano, J., Castellanos, L. and Quintana, M.I. (2001), Tunneling and genetic algorithms for global optimization, in: N. Hadjisavvas and P. Pardalos, eds., *Advances in Convex Analysis and Global Optimization*, Kluwer Academic Publishers, 553–567.

[30] Hansen, P., Jaumard, B. and Li, S. (1992), Global optimization of univariate Lipschitz functions I,II, *Mathematical Programming*, 55, 251–292.

[31] Hansen, P., Jaumard, B. and Li, S. (1992), On the use of estimates of the Lipschitz constant in global optimization, *Journal of Optimization Theory and Applications*, 75, 195–200.

[32] Harker, P.T. and Pang, J.-S. (1990), Finite-dimensional variational inequality and nonlinear complementarity problems: A survey of theory, algorithms and applications, *Mathematical Programming*, 48, 161–220.

[33] Harker, P.T. (1984), A variational inequality approach for the determination of oligopolistic market equilibrium, *Mathematical Programming*, 30, 105–111.

[34] Hartman, P. and Stampacchia, G. (1966), On some nonlinear elliptic differential functional equations, *Acta Mathematica*, 115, 153–188.

[35] Hearn, D.W. (1982), The gap function of a convex program, *Operation Research Letters*, 1, 67–71.

[36] Herrera, F., Lozono, M. and Verdegay, J.L. (1998), Tackling real-coded genetic algorithms: Operators and tools for behavioural analysis, *Artificial Intelligence Review*, 12, 265–319.

[37] Horst, R., Pardalos, P. and Thoai, N.V. (2000), *Introduction to Global Optimization*, 2nd edition, Kluwer Academic Publisher, London.

[38] Horst, R. and Pardalos, P. (1995), *Handbook of Global Optimization*, Kluwer Academic Publisher, London.

[39] Josephy, N.H. (1979), Newton's method for generalized equations, *Techincal Summary Report 1965*, Mathematics Research Center, University of Wisconsin, Madison, WI.

[40] Kanzow, C. (2000), Global optimization techniques for mixed complementarity problems, *Journal of Global Optimization*, 16, 1–21.

[41] Kanzow, C. and Pieper, H. (1999), Jacobian smoothing methods for nonlinear complementarity problems, *SIAM Journal on Optimization*, 9, 342–373.

[42] Kanzow, C., Yamashita, N. and Fukushima, M. (1997), New NCP-functions and their properties, *Journal of Optimization Theory and Applications*, 94, 115–135.

[43] Kanzow, C. and Fukushima, M. (1998), Theoretical and numerical investigation of the D-gap function for box constrained variational inequalities, *Mathematical Programming*, 83, 55–87.

[44] Karamardian, S. (1971), Generalized complementarity problem, *Journal of Optimization Theory and Applications*, 8, 161–167.

[45] Karamardian, S. (1972), The complementarity problem, *Mathematical Programming*, 2, 107–129.

[46] Karamardian, S. (1976), Complementarity problems over cones with monotone and pseudomonotone maps, *Journal of Optimization Theory and Applications*, 18, 445–454.

[47] Karamardian, S. (1976), An existence theorem for the complementarity problem, *Journal of Optimization Theory and Applications*, 19, 227–232.

[48] Kostreva, N.M. and Zheng, Q. (1994), Integral global optimization method for solution of nonlinear complementarity problems, *Journal of Global Optimization*, 5, 181–193.

[49] Kyparisis, J. (1986), Uniqueness and differentiability of solutions of parametric nonlinear complementarity problems, *Mathematical Programming*, 36, 105–113.

[50] Kyparisis, J. (1987), Sensitivity analysis framework for variational inequalities, *Mathematical Programming*, 38, 203–213.

[51] Laguna, M. and Marti, R. (2003), *Scatter Search: Methodology and Implementation in C*, Kluwer Academic Publishers, Boston, MA.

[52] Laguna, M. and Marti, R. (2005), Experimental testing of advanced scatter search designs for global optimization of multimodal functions, *Journal of Global Optimization*, 33, 235–255.

[53] Land, A. H. and Doig, A. G. (1960), An automatic method for solving discrete programming problems, *Econometrica*, 28, 497–520.

[54] Lemke, C.E. and Howson, J.T. (1964), Equilibrium points of bimatrix games, *SIAM Review*, 12, 45–78.

[55] Levy, A.V. and Montalvo, A. (1985), The tunneling algorithm for the global minimization of functions, *SIAM Journal on Scientific and Statistical Computing*, 6, 15–29.

[56] Levy, A.V. and Gomez, S. (1985), The tunneling method applied to global optimization, in: P.T. Boggs, R.H. Byrd and R.B. Schnabel, eds., *Numerical Optimization*, SIAM, Philadelphia, 213–244.

[57] Lions, J.L. and Stampacchia, G. (1967), Variational inequalities, *Communications on Pure and Applied Mathematics*, 20, 493–519.

[58] Liu, Y. and Teo, K. (1999), A bridging method for global optimization, *Journal of Australian Mathematical Society Series B,* 41, 41-57.

[59] Luo, Z.Q. and Tseng, P. (1997), A new class of merit functions for the nonlinear complementarity problem, in: M.C. Ferris and J.S. Pang, eds., *Complementarity and Variational Problems: State of the Art*, SIAM, Philadelphia, 204–225.

[60] Majig, M., Hedar, A.R. and Fukushima, M. (2007), Hybrid evolutionary algorithm for solving general variational inequalities, *Journal of Global Optimization*, 38, 637–651.

[61] Majig, M., Barsbold, B., Enkhbat, R. and Fukushima, M., A global optimization approach for solving non-monotone variational inequality problems, to appear in *Optimization.*

[62] Majig, M., Hedar, A.R. and Fukushima, M., Hybrid evolutionary algorithm for solving global optimization problems, *Optimization and Optimal Control: Theory and Applications*, in: A. Chinchuluun, P.M. Pardalos, R. Enkhbat and I. Tseveendorj, eds., Springer Lecture Series, to appear.

[63] Majig, M. and Fukushima, M., Restricted-step Josephy-Newton method for general variational inequality problems with polyhedral constraints, to appear in *Pacific Journal of Optimization.*

[64] Majig, M. and Fukushima, M. (2008), Adaptive fitness function for evolutionary algorithm and its applications, *International Conference on Informatics Education and Research Knowledge-Circulating Society*, Kyoto, Japan, 119–124.

[65] Mancino, O. and Stampacchia, G. (1972), Convex programming and variational inequalities, *Journal of Optimization Theory and Application*, 9, 3–23.

[66] Mangasarian, O.L. (1976), Equivalence of the complementarity problem to a system of nonlinear equations, *SIAM Journal on Applied Mathematics*, 31, 89–92.

[67] Mangasarian, O.L. (1980), Locally unique solutions of quadratic programs, linear and nonlinear complementarity problems, *Mathematical Programming*, 19, 200–212.

[68] Marcotte, P. (1986), Gap-decreasing algorithms for monotone variational inequalities, *ORSA/TIMS Meeting*, Miami Beach.

[69] Marcotte, P. (1986), Network design with congestion effects: a case of bi-level programming, *Mathematical Programming*, 34, 142–162.

[70] Marcotte, P. and Dussault, J.P. (1985), A modified Newton method for solving variational inequalities, *Proceeding of the 24th IEEE Conference on Decision and Control*, 1433–1436.

[71] Marcotte, P. and Dussault, J.P. (1987), A note on a globally convergent Newton method for solving monotone variational inequalities, *Operations Research Letters*, 6, 35–42.

[72] Mastroeni, G. (2005), Gap functions and descent methods for Minty variational inequality, *Optimization and Control with Application*, in: L. Qi, K.L. Teo and X,Q. Yang, eds., Kluwer Academic Publisher, Boston, MA, 529–547.

[73] Mathiesen, L. (1985), Computation of economic equilibria by a sequence of linear complementarity problems, *Mathematical Programming Study*, 23, 144–162.

[74] Mathiesen, L. (1985), Computational experience in solving equilibrium models by a sequence of linear complementarity problems, *Operations Research*, 33, 1225–1250.

[75] Meewella, C.C. and Mayne, D.Q. (1989), Efficient domain partitioning method algorithms for global optimization of rational and Lipschitz continuous functions, *Journal of Optimization Theory and Applications*, 61, 247–270.

[76] Megiddo, N. and Kojima, M. (1977), On the existence and uniqueness of solutions in nonlinear complementarity theory, *Mathematical Programming*, 12, 110–130.

[77] Minty, G. (1962), Monotone (nonlinear) operators in Hilbert Space, *Duke Mathematical Journal*, 29, 341–346.

[78] Moré, J.J. (1971), The application of variational inequalities to complementarity problems and existence theorems, Technical Report 71–90, Department of Computer Sciences, Cornell University, Ithaca, NY.

[79] Moré, J.J. (1974), Classes of functions and feasibility conditions in nonlinear complementarity problems, *Mathematical Programming*, 6, 327–338.

[80] Moré, J.J. (1974), Coercivity conditions in nonlinear complementarity problems, *SIAM Review*, 17, 1–16.

[81] Moré, J.J. and Rheinboldt, W.C. (1973), On P- and S- functions and related classes of n-dimensional nonlinear mappings, *Linear Algebra and Its Applications*, 6, 45–68.

[82] Ortega, J.M. and Rheinboldt, W.C. (1970), *Iterative Solution of Nonlinear Equations in Several Variables*, Academic Press, New York.

[83] Pardalos, P.M. and Rosen, J.B. (1988), Global optimization approach to the linear complementarity problem, *SIAM Journal on Scientific and Statistical Computing*, 6, 341–353.

[84] Peng, J.M. (1997), Equivalence of variational inequality problems to unconstrained minimization, *Mathematical Programming*, 78, 347–355.

[85] Peng, J.M. and Yuan, Y. (1997) Unconstrained methods for generalized complementarity problems, *Journal of Computational Mathematics*, 15, 253–264.

[86] Pinter, J. (1992), Lipschitz global optimization: Some prospective appication, in: C. Floudas and P.M. Pardalos, eds., *Recent Advances in Global Optimization*, Princeton University Press.

[87] Piyavskii, S.A. (1972), An algorithm for finding the absolute extremum of a function, *USSR Computational Mathematics and Mathematical Physics*, 12, 57–67.

[88] Qi, H.D. and Liao, L.Z. (2000), A smoothing Newton method for general nonlinear complementarity problems, *Computational Optimization and Applications*, 17, 231–253.

[89] Rago, C. and Alidaee, B. (2005), *Metaheuristics Optimization via Memory and Evolution*, Kluwer Academic Publishers, Boston, MA.

[90] Ratchek, H. and Rokne, J. (1984), *Computer Methods for the Range of Functions*. Ellis Horwood, Chichester.

[91] Ratchek, H. and Rokne, J. (1988), *New Computer Methods for Global Optimization*. Ellis Horwood, Chichester.

[92] Robinson, S.M. (1992), Normal maps induced by linear transformations, *Mathematics of Operations Research*, 17, 691–714.

[93] Robinson, S.M. (1990), Homeomorphism conditions for normal maps of polyhedra, in: A. loffe, M. Marcus, and S. Reich, eds., *Optimization and Nonlinear Analysis* (Haifa), Pitman Research Notes in Mathematics Series 244, Longman House, Harlow, 240–248.

[94] Robinson, S.M. (1993), Nonsingularity and symmetry for linear normal maps, *Mathematical Programming*, 62, 415–425.

[95] Robinson, S.M. (1982), Generalized equations, in: A. Bachem, M. Grotschel and B. Korte, eds., *Mathematical Programming: The State of the Art*, Springer, Berlin, 346–367.

[96] Selami, H. and Robinson, S.M. (1997), Implementation of a continuation method for normal maps, *Mathematical Programming*, 76, 563–578.

[97] Smith, M.J. (1984), A descent algorithm for solving monotone variational inequality and monotone complementarity problems, *Journal of Optimization Theory and Application*, 44, 485–496.

[98] Smith, T.E. (1984), A solution condition for complementarity problems: with an application to spatial price equilibrium, *Applied Mathematics and Computation*, 15, 61–69.

[99] G. Stampacchia, G. (1968), Variational inequalities, in: *Theory and Applications of Monotone Operators*, Proceedings of the NATO Advanced Study Institute, Venice, Italy (Edizioni Oderisi, Gubbio, Italy), 102–192.

[100] Strongin, R.G. (1973), On the convergence of an algorithm for finding a global extremum. *Engineering Cybernetics*, 11, 549–555.

[101] Taji, K., Fukushima, M. and Ibaraki, T. (1993), A globally convergent Newton method for solving strongly monotone variational inequalities, *Mathematical Programming*, 58, 369–383.

[102] Taji, K. and Fukushima, M. (1996), A new merit function and a successive quadratic programming algorithm for variational inequality problems, *SIAM Journal on Optimization*, 6, 704–713.

[103] Talbi, E. (2002), A taxamony of hybrid metaheuristics, *Journal of Heuristics*, 8, 541–564.

[104] Tobin, R.L. (1986), Sensitivity analysis for variational inequalities, *Journal of Optimization Theory and Applications*, 48, 191–204.

[105] Xu, Z., Huang, H.X., Pardalos, P.M. and Xu, C.X. (2001), Filled functions for unconstrained global optimization, *Journal of Global Optimization*, 20, 49–65.

[106] Yamashita, N., Taji, K. and Fukushima, M. (1997), Unconstrained optimization reformulations of variational inequality problems, *Journal of Optimization Theory and Applications*, 92, 439–456.

[107] Zhu, D.L. and Marcotte, P. (1994), An extended descent framework for variational inequalities, *Journal of Optimization Theory and Applications*, 80, 349–360.