Chapter 2

A Globally Convergent Improved BFGS Method for Generalized Nash Equilibrium Problems

2.1 Introduction

Generalized Nash Equilibrium Problem (GNEP) is a non-cooperative game in which the strategy set of each player may depend on the strategies of the rival player. It was first formally introduced by Debreu [2] as a social equilibrium in 1952, and later as an abstract economy [3] in 1954. In the early 50's, Nash [12, 13] introduced a notion of equilibrium, called Nash equilibrium, for non-cooperative N-player games where the payoff function of each player depends on the others' strategies. Arrow and Debreu [3] extended this notion to the generalized Nash equilibrium for games where both the payoff function and the set of feasible strategies depend on others' strategies. GNEPs have been a major area of research during the last two decades, which have several real-world applications in the areas of economics, computer science, and engineering, e.g., the abstract economy model [3], a power allocation problem in telecommunications [4], a competition among countries that arises from the Kyoto protocol to reduce the air pollution [5], social science [14], energy problems [15–17], wireless communication [6,7], cloud computing [8], electricity generation [9], etc. Robinson [10, 11] discussed the problem of measuring the effectiveness in optimization-based combat models and gave several mathematical formulations. All these applications have motivated the evolution of the generalized Nash equilibrium concept and its use in complex games that now require a deep understanding of theoretical and computational mathematics.

2.2 Motivation

The convergence and efficiency of the BFGS method have motivated many researchers to study and improve the BFGS method [51–54]. In BFGS methods, a stationary point may be easily missed if the step size is large, or a cycle may be generated among several points if the step size is small. To overcome these drawbacks, Yuan et al. [55] proposed the modified-weak Wolfe-Powell (MWWP) line search technique and used it to prove the global convergence of the BFGS method for general functions. Yang et al. [45] proposed an improved BFGS method using an MWWP line search technique and showed a detailed numerical performance compared with the original BFGS method using a weak Wolfe-Powell (WWP) line search technique. The numerical performance [45] shows that the improved BFGS method with the MWWP line search technique has better problem-solving capability than the standard BFGS algorithm based on the WWP line search technique. Therefore, in this chapter, we propose to solve GNEPs with an improved BFGS method using the Armijo-Goldstein and MWWP line search techniques.

Facchinei et al. [1] analyzed GNEPs with shared constraints and proposed Newton-type methods– semi-smooth Newton methods and Levenberg-Marquardt method to solve them. The semi-smooth Newton method in [1] converges Q-quadratically, but they have a drawback: they do not converge globally. Solving a system of linear (or nonlinear) equations by the semi-smooth Newton method at each stage can be expensive if the number of unknowns is large and may not be justified when the initial guess is far from a solution. This motivates us to develop an improved BFGS method that consumes lesser computation costs (number of iterations and CPU time). Therefore, we aim to solve GNEPs using an improved BFGS method such that it converges globally. To minimize the computation costs, we use Armijo-type line search techniques, which are cost-effective compared to the Wolfe-type line search techniques. Therefore, we solve GNEPs by BFGS method using the two line search techniques: Armijo-Goldstein and MWWP [45], and provide their numerical performances.

2.3 Contributions

In this chapter, we have proposed an improved BFGS method to solve GNEPs, which is globally convergent. The novelty and contribution of this chapter are as follows:

- With the help of Fischer-Burmeister C-function [42], we formulate a smooth merit function for solving GNEPs under consideration.
- Step-wise algorithms of the proposed BFGS methods, with MWWP and Armijo-Goldstein rule, in the GNEP set-up are provided.
- Well-definedness and global convergence of the proposed two algorithms are given.
- Numerical performance of the studied methods on some academic and practical GNEPs are provided.

2.4 Improved BFGS method

The BFGS method is known as an effective and favorable solver for finding a minimum of a continuously differentiable function. A general structure of the commonly used quasi-Newton method—BFGS technique– is given below.

Consider an optimization problem

$$\min_{x\in\mathbb{R}^n} f(x), \text{ where } f \text{ is a differentiable real-valued function.}$$

The main steps in the BFGS method are as follows.

1. *Find a descent direction*

   Find a descent direction d_k that solves the system $B_k d_k = -g_k$, where $g_k = \nabla f(x_k)$ is the gradient of f at x_k, and B_k is an approximation of the Hessian $\nabla^2 f(x_k)$.

2. *Find a step-length*

   Find a step-length $\alpha_k \in \mathbb{R}$ along the descent direction d_k. The step length can be obtained using a line search technique: exact or inexact. The next iterative point is obtained by

   $$x_{k+1} = x_k + \alpha_k d_k.$$

3. *Update of Hessian approximation matrix*

   Hessian approximation matrix B_{k+1} can be updated by the standard BFGS update formula:

   $$B_{k+1} = B_k + \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{s_k^T y_k},$$

   where $s_k = x_{k+1} - x_k$ and $y_k = \nabla f(x_{k+1}) - \nabla f(x_k) = g_{k+1} - g_k$ are such that they satisfy the secant equation

   $$B_{k+1} s_k = y_k.$$

4. Iteration continues until a stopping criterion is satisfied.

The convergence and efficiency of the BFGS method have motivated many researchers to study and improve the BFGS method [51–54]. In BFGS methods, a stationary point may be easily missed if the step size is large, or a cycle may be generated among several points if the step size is small. To overcome these drawbacks, Yuan et al. [55] proposed the modified-weak Wolfe-Powell (MWWP) line search technique and used it to prove the global convergence of the BFGS method for general functions. Yang et al. [45] proposed an improved BFGS method using an MWWP line search technique and showed a detailed numerical performance compared with the original BFGS method using a weak Wolfe-Powell (WWP) line search technique. The numerical performance [45] shows that the improved BFGS method with the MWWP line search technique has better problem-solving capability than the standard BFGS algorithm based on the WWP line search technique. Therefore, in this chapter, we propose to solve GNEPs with an improved BFGS method using the Armijo-Goldstein and MWWP line search techniques.

The conventional BFGS update formula is

$$B_{k+1} = B_k - \frac{B_k s_k {s_k}^\top B_k}{{s_k}^\top B_k s_k} + \frac{y_k {y_k}^\top}{{y_k}^\top s_k}, \tag{2.1}$$

where $s_k = z_{k+1} - z_k$ and $y_k = \nabla\Phi(z_{k+1}) - \nabla\Phi(z_k)$. The BFGS formula is updated by Yuan et al. [56] is given by

$$B_{k+1} = B_k - \frac{B_k s_k {s_k}^\top B_k}{{s_k}^\top B_k s_k} + \frac{y_k^* {y_k^*}^\top}{{y_k^*}^\top s_k}, \tag{2.2}$$

where $y_k^* = y_k + a_k^* s_k$, $a_k^* = \max\{\bar{a}_k, 0\}$, $s_k = z_{k+1} - z_k$, $y_k = \nabla\Phi(z_{k+1}) - \nabla\Phi(z_k)$ and

$$\bar{a}_k = \tfrac{1}{\|s_k\|^2} \left\{ 6[\Phi(z_k) - \Phi(z_k + \alpha_k d_k)] + 3[\nabla\Phi(z_k) + \nabla\Phi(z_k + \alpha_k d_k)]^\top s_k \right\}. \tag{2.3}$$

An important property of formula (2.2) is that B_{k+1} remains positive definite as long as ${y_k^*}^\top s_k > 0$ (see [57]). The condition ${y_k^*}^\top s_k > 0$ assured to hold if the step-size is determined by Wolfe-type line search technique (see [45]):

$$\left.\begin{aligned} &\Phi(z_k + \alpha_k d_k) \leq \Phi(z_k) + \delta_1 \alpha_k \nabla\Phi(z_k)^\top d_k \\ &\nabla\Phi(z_k + \alpha_k d_k)^\top d_k \geq \delta_2 \nabla\Phi(z_k)^\top d_k, \end{aligned}\right\} \tag{2.4}$$

where δ_1, δ_2 are positive constants such that $\delta_1 < \delta_2 < 1$. Yuan et al. [55] have improved the weak Wolfe-Powell (WWP) line search technique and studied the new line search technique: Modified Weak Wolfe-Powell (MWWP) line search technique [55], which has global convergence once used in a BFGS method. MWWP is formulated as follows:

$$\left.\begin{aligned} &\Phi(z_k + \alpha_k d_k) \leq \Phi(z_k) + c\alpha_k \nabla\Phi(z_k)^\top d_k + \alpha_k \min\left\{-c_1 \nabla\Phi(z_k)^\top d_k, \frac{c\alpha_k \|d_k\|^2}{2}\right\} \\ &\nabla\Phi(z_k + \alpha_k d_k)^\top d_k \geq c_2 \nabla\Phi(z_k)^\top d_k + \min\left\{-c_1 \nabla\Phi(z_k)^\top d_k, c\alpha_k \|d_k\|^2\right\} \end{aligned}\right\}, \tag{2.5}$$

where $c \in (0,1)$, $\alpha_k > 0$, $c_1 \in (0,c)$ and $c_2 \in (c,1)$. For results based on this improved line search, one can refer to [45, 58].

In comparison to Wolfe-type line search techniques, the Armijo-Goldstein line search techniques have a better speed of convergence and are better suited for quasi-Newton methods. In this chapter, we use the Armijo-Goldstein line search technique to compute the step length α_k that is the largest value in the set $\{\bar{c}\, 2^{-i} \mid \bar{c} \in [0,\infty)$ is fixed, and $i = 0, 1, 2, \ldots\}$ for which the inequalities

$$\left.\begin{aligned} &\Phi(z_k + \alpha_k d_k) \leq \Phi(z_k) + c\alpha_k \nabla\Phi(z_k)^\top d_k \\ &\Phi(z_k + \alpha_k d_k) \geq \Phi(z_k) + (1-c)\alpha_k \nabla\Phi(z_k)^\top d_k \end{aligned}\right\} \tag{2.6}$$

are satisfied with $c \in (0, \frac{1}{2})$. However, the Armijo-Goldstein conditions do not ensure ${y_k^*}^\top s_k > 0$, and therefore B_{k+1} is not necessarily positive definite even if B_k is positive definite. Hence, we write the BFGS update formula B_k with the following form

$$B_{k+1} = \begin{cases} B_k - \frac{B_k s_k {s_k}^\top B_k}{{s_k}^\top B_k s_k} + \frac{y_k^* {y_k^*}^\top}{{y_k^*}^\top s_k} & \text{if } \frac{{y_k^*}^\top s_k}{\|s_k\|^2} \geq \beta \|\nabla \Phi(z_k)\|^\alpha, \\ B_k & \text{otherwise,} \end{cases} \tag{2.7}$$

where β and α are positive constants.

Now, we consider the player convex GNEP. Thus, we have reformulated GNEP (1.7)

$$F(x, \lambda) = \begin{pmatrix} L(x, \lambda) \\ \Phi(-g(x), \lambda) \end{pmatrix} = 0. \tag{2.8}$$

The reformulated system (2.8) becomes

$$F(x, \lambda) = \begin{pmatrix} \nabla_{x^1} L_1(x^1, \boldsymbol{x}^{-1}, \lambda^1) \\ \nabla_{x^2} L_1(x^2, \boldsymbol{x}^{-2}, \lambda^2) \\ \vdots \\ \nabla_{x^N} L_N(x^N, \boldsymbol{x}^{-N}, \lambda^N) \\ \Phi(-g^1(x^1, \boldsymbol{x}^{-1}), \lambda^1) \\ \Phi(-g^2(x^2, \boldsymbol{x}^{-2}), \lambda^2) \\ \vdots \\ \Phi(-g^N(x^N, \boldsymbol{x}^{-N}), \lambda^N) \end{pmatrix} = 0. \tag{2.9}$$

Here, we use a smooth complementarity function with the help of Fischer-Burmeister C-function [42]. The Fischer-Burmeister function $\Psi : \mathbb{R}^2 \to \mathbb{R}$ is defined by

$$\Psi(x, y) = \sqrt{x^2 + y^2} - (x + y),$$

which is a convex function but not differentiable at $(0, 0)$. Thus, we take (see [59, 60]) the complementarity function as

$$\Phi(x,y) = \Psi(x,y)^2, \tag{2.10}$$

which is known to be differentiable everywhere, and its gradient is globally Lipschitz continuous and semismooth [59]. A survey on several other merit functions and their basic and desirable properties can be found in [60]. With the $\Phi(x,y)$ in (2.10), the reformulated system (2.9) becomes a system of smooth equations. Since Φ is continuously differentiable everywhere, the system (2.9) becomes a system of smooth equations $F(z) = 0$. With the help of $F(x,\lambda) : \mathbb{R}^{n+m} \to \mathbb{R}^{n+m}$ in (2.9) and $z = (x,\lambda) \in \mathbb{R}^{n+m}$, consider the merit function

$$\Phi(z) = \tfrac{1}{2}\|F(z)\|^2. \tag{2.11}$$

Here, $\Phi(z)$ is a square of the norm of a differentiable function. Therefore, Φ is a differentiable function and we can write $\nabla\Phi(z_k) = \nabla F(z_k)^\top F(z_k)$. We will solve the smooth system

$$\Phi(z) = 0 \tag{2.12}$$

using BFGS method.

2.5 Improved BFGS methods to solve GNEPs

Algorithm 1 Improved BFGS method using Armijo-Goldstein line search technique to solve the smooth system (2.12)

Step 0: Initialization

Take initial Hessian approximation matrix $B_0 = I_{(n+m)\times(n+m)}$, and any $c \in (0, 0.5)$, $\bar{c} \in [1, \infty)$

Choose $z_0 = (x_0, \lambda_0) \in \mathbb{R}^n \times \mathbb{R}^m$.

Take constants $\alpha > 0$, $\beta > 0$, and set the iteration counter $k = 0$.

Provide the termination scalar $\epsilon > 0$.

Step 1: Termination condition

If $\|\nabla\Phi(z_k)\| < \epsilon$, then stop and output z_k as an ϵ-precise solution to system (2.12).

Step 2: Descent direction

Find a solution $d_k \in \mathbb{R}^{n+m}$ of the system

$$B_k d = -\nabla\Phi(z_k). \tag{2.13}$$

Step 3: Step length

Find a step length $\alpha_k = \bar{c}\, 2^{-i_k}$ using Armijo-Goldstein line search technique:

Choose i_k, the smallest non-negative integer i such that

$$\left.\begin{array}{l}\Phi(z_k + \bar{c}\, 2^{-i_k} d_k) \le \Phi(z_k) + c\bar{c}\, 2^{-i_k} \nabla\Phi(z_k)^\top d_k \\ \Phi(z_k + \bar{c}\, 2^{-i_k} d_k) \ge \Phi(z_k) + (1-c)\bar{c}\, 2^{-i_k} \nabla\Phi(z_k)^\top d_k\end{array}\right\} \tag{2.14}$$

Step 4: Intermediate computation

$z_{k+1} = z_k + \alpha_k d_k$, $s_k = z_{k+1} - z_k$ and $y_k = \nabla\Phi(z_{k+1}) - \nabla\Phi(z_k)$. Compute $\bar{a}_k$ by (2.3), $a_k^* = \max\{\bar{a}_k, 0\}$ and $y_k^* = y_k + a_k^* s_k$.

Step 5: Hessian approximation

Calculate the Hessian approximation B_{k+1} by (2.7), where B_{k+1} satisfies the quasi-Newton relation (see [53]):

$$B_{k+1}(z_{k+1} - z_k) = y_k + a_k^* s_k.$$

Update $z_{k+1} \leftarrow z_k + 2^{-i_k} d_k$, $k \leftarrow k + 1$ and go to Step 1.

Algorithm 2 Improved BFGS method using MWWP line search technique to solve the smooth system (2.12)

Step 0: Initialization

Take initial Hessian approximation matrix $B_0 = I_{(n+m)\times(n+m)}$, and any $c \in (0, 0.5)$, $\bar{c} \in [1, \infty)$ $c_1 \in (0, c)$, $c_2 \in (c, 1)$.

Choose $z_0 = (x_0, \lambda_0) \in \mathbb{R}^n \times \mathbb{R}^m$.

Set the iteration counter $k = 0$.

Provide the termination scalar $\epsilon > 0$.

Step 1: Termination condition

If $\|\nabla\Phi(z_k)\| < \epsilon$, then stop and output z_k as an ϵ-precise solution to system (2.12).

Step 2: Descent direction

Find a solution $d_k \in \mathbb{R}^{n+m}$ of the system

$$B_k d = -\nabla\Phi(z_k). \tag{2.15}$$

Step 3: Step length

Find a step length $\alpha_k = \bar{c}\ 2^{-i_k}$ using MWWP line search technique:

Choose i_k, the smallest non-negative integer i such that

$$\left.\begin{array}{r}\Phi(z_k + \bar{c}\ 2^{-i_k} d_k) \le \Phi(z_k) + c\bar{c}\ 2^{-i_k} \nabla\Phi(z_k)^\top d_k + \bar{c}\ 2^{-i_k} \qquad\qquad\qquad\qquad \\ \min\left\{-c_1 \nabla\Phi(z_k)^\top d_k, \dfrac{c\bar{c}\ 2^{-i_k}\|d_k\|^2}{2}\right\} \\ \nabla\Phi(z_k + \bar{c}\ 2^{-i_k} d_k)^\top d_k \ge c_2 \nabla\Phi(z_k)^\top d_k + \min\left\{-c_1 \nabla\Phi(z_k)^\top d_k, c\bar{c}\ 2^{-i_k}\|d_k\|^2\right\}\end{array}\right\} \tag{2.16}$$

Step 4: Intermediate computation

$z_{k+1} = z_k + \alpha_k d_k$, $s_k = z_{k+1} - z_k$ and $y_k = \nabla\Phi(z_{k+1}) - \nabla\Phi(z_k)$. Compute $\bar{a}_k$ by (2.3), $a_k^* = \max\{\bar{a}_k, 0\}$ and $y_k^* = y_k + a_k^* s_k$.

Step 5: Hessian approximation

Calculate the Hessian approximation B_{k+1} by (2.2), where B_{k+1} satisfies the quasi-Newton relation (see [53]):

$$B_{k+1}(z_{k+1} - z_k) = y_k + a_k^* s_k.$$

Update $z_{k+1} \leftarrow z_k + 2^{-i_k} d_k$, $k \leftarrow k+1$ and go to Step 1.

2.6 Convergence analysis

In the following section, we show the well-definedness of Algorithms 1 and 2 and establish their global convergence under the following assumption on the GNEP under consideration.

Before reaching the main convergence result, we prove some subsidiary properties of $\{B_k\}$ that facilitate obtaining the main result.

Lemma 2.1 *If the sequence $\{B_k\}$ is obtained by (2.7), in Algorithm 1, then the matrix B_k is positive definite for every $k = 0, 1, 2, \ldots$.*

Proof: Note that $B_0 = I_{(m+n)\times(m+n)}$ is positive definite. Let $\{z_k\}$ be the sequence obtained by Algorithm 1, and the BFGS matrix is updated by (2.7), and B_k is a positive definite matrix for some $k > 0$. We show that B_{k+1} is positive definite. This will complete the proof.

If $\frac{y_k^{*\top} s_k}{\|s_k\|^2} \geq \beta \|\nabla \Phi(z_k)\|^{\alpha}$, then evidently, $y_k^{*\top} s_k > 0$, and hence (see p.6 [55])

$$B_{k+1} = B_k - \frac{B_k s_k {s_k}^\top B_k}{{s_k}^\top B_k s_k} + \frac{y_k^* {y_k^*}^\top}{{y_k^*}^\top s_k} \text{ is positive definite.}$$

In the second case, as $B_{k+1} = B_k$, B_{k+1} is obviously positive definite.

□

Lemma 2.2 *Let the sequence $\{B_k\}$ is obtained by (2.2) in Algorithm 2. Then, the matrix B_k is positive definite for every $k = 0, 1, 2, \ldots$.*

Proof: Note that $B_0 = I_{(m+n)\times(m+n)}$ is positive definite. Suppose that the matrix B_k is a positive definite matrix for some $k > 0$. We prove that B_{k+1} is positive definite. To prove that B_{k+1} is positive definite, we need to show that $y_k^* \top s_k > 0$. Using (2.16),

$B_k d_k = -\nabla\Phi(z_k)$, $\Phi_{k+1} = \Phi(z_k + \bar{c}2^{-i_k} d_k)$, $\Phi_k = \Phi(z_k)$, and we have

$$\begin{aligned}
y_k^{*\top} s_k &= (y_k + a_k^* s_k)^\top s_k \\
&= y_k^\top s_k + s_k^\top a_k^* s_k \\
&= \nabla\Phi_{k+1}^\top s_k - \nabla\Phi_k^\top s_k + s_k^\top a_k^{*\top} s_k \\
&\geq c_2 \nabla\Phi_k^\top s_k + \min\left\{-c_1 \nabla\Phi_k^\top s_k, c s_k \|d_k\|^2\right\} - \nabla\Phi_k^\top s_k + s_k^\top a_k^{*\top} s_k \\
&= -(1-c_2)\nabla\Phi_k^\top s_k + \min\left\{-c_1 \nabla\Phi_k^\top s_k, c s_k \|d_k\|^2\right\} + s_k^\top a_k^{*\top} s_k \\
&\geq -(1-c_2)\nabla\Phi_k^\top s_k > 0.
\end{aligned}$$

(Here, we have used $\min\left\{-c_1 \nabla\Phi_k^\top s_k, c s_k \|d_k\|^2\right\} \geq 0$ and $-\nabla\Phi_k^\top d_k = d_k^\top B_k d_k > 0$, as B_k is positive definite matrix). Hence, B_{k+1} is a positive definite matrix (see p.6 [55]). □

In Algorithm 1, the Hessian approximation B_k is updated by the BFGS update formula (2.7). It is found (Lemma 2.1) that the Hessian approximation B_k in Algorithm 1 is a symmetric and positive definite matrix for all k. The descent direction d_k, obtained from (2.13), and the step-length, calculated from (2.14), together imply that $\{\Phi(z_k)\}$ is a monotonic non-increasing sequence. Using (2.11), the sequence $\{\Phi(z_k)\}$ is bounded below. Hence $\{\Phi(z_k)\}$ is a convergent sequence.

From (2.14), we have

$$\frac{\Phi(z_k) - \Phi(z_k + \bar{c}\, 2^{-i_k} d_k)}{1-c} \leq -\bar{c}\, 2^{-i_k} \nabla\Phi(z_k)^\top d_k \leq \frac{\Phi(z_k) - \Phi(z_k + \bar{c}\, 2^{-i_k} d_k)}{c}. \tag{2.17}$$

Therefore, using

$$\lim_{k\to\infty} \{\Phi(z_k) - \Phi(z_k + \bar{c}\, 2^{-i_k} d_k)\} = 0,$$

we have

$$-\lim_{k\to\infty} \bar{c}\, 2^{-i_k} \nabla\Phi(z_k)^\top d_k = -\lim_{k\to\infty} \nabla\Phi(z_k)^\top s^k = 0. \tag{2.18}$$

By results (2.18) and Lemma 2.2 we prove convergence of Algorithms 1 and 2 in the next section.

In Algorithm 2, we have used Wolfe-type line search techniques. Due to this, the Hessian approximation B_k, updated by BFGS update formula (2.2), in Algorithm 2 is symmetric and positive definite matrix for all k (Lemma 2.2). The descent direction d_k, obtained from (2.15), and the step-length, calculated from (2.16), together implies that $\{\Phi(z_k)\}$ is a monotonic non-increasing sequence. Also, using (2.11), $\{\Phi(z_k)\}$ is bounded, and hence $\{\Phi(z_k)\}$ is a convergent sequence.

Assumption 1 *1. Consider the level set*

$$\Omega = \{z \in \mathbb{R}^{n+m} \mid \Phi(z) \le \Phi(z_0)\} \text{ is bounded.}$$

2. The function Φ is continuously differentiable.

3. There exists a constant $L > 0$ for which

$$\nabla\Phi(z_1) - \nabla\Phi(z_2)\| \le L\|z_1 - z_2\|, \quad \text{for all } z_1,\, z_2 \in \Omega. \tag{2.19}$$

Consider a sequence $\{z_k\}$ which is obtained by Algorithm 1. Then, the sequence $\{\Phi(z_k)\}$ is a monotonic non-increasing sequence, i.e.,

$$\Phi(z_0) \ge \Phi(z_1) \ge \Phi(z_2) \ge \cdots .$$

Therefore, the sequence $\{z_k\}$ is obtained by Algorithm 1 is lies in Ω. Define the index set

$$\bar{K} = \left\{ k \,\middle|\, \frac{y_k^{*\top} s_k}{\|s_k\|^2} \ge \beta\|\nabla\Phi(z_k)\|^\alpha \right\}, \tag{2.20}$$

where α and β are positive constants. Then, we can rewrite the Hessian approximation update formula (2.7) as

$$B_{k+1} = \begin{cases} B_k - \frac{B_k s_k s_k{}^\top B_k}{s_k{}^\top B_k s_k} + \frac{y_k^* y_k^{*\top}}{y_k^{*\top} s_k}, & \text{if } k \in \bar{K}, \\ B_k, & \text{otherwise.} \end{cases} \tag{2.21}$$

Now, it is essential to show that Algorithms 1 and 2 are well-defined. Notice that the well-definedness of Algorithm 1 is dependent on the existence of a d for (2.13) and an i_k for (2.14). In Theorem 2.1 and Theorem 2.2 below, we show the existence of both d and i_k, respectively.

In the similar way of the proofs of Theorem 2.1 and Theorem 2.2, the well-definedness of Algorithm 2 is followed. Note that if sequence $\{z_k\}$ is obtained by Algorithm 2, and the Hessian approximation matrix B_k is updated by (2.21), then using Lemma 2.2, the matrix B_k is always positive definite. Therefore, the system (2.15) always has a unique solution d_k. Also, under Assumption 1, the system (2.16) is well-defined (see [61]). Thus, we can see that Algorithms 1 and 2 are well-defined.

Theorem 2.1 *Assume that Assumption 1 holds for $\Phi(z)$. Consider a sequence $\{z_k\}$ which is obtained by Algorithm 1. Let the Hessian approximation matrix B_k be updated by (2.7). Then, there exists d_k such that (2.13) is true.*

Proof: Given sequence, $\{z_k\}$ is obtained by Algorithm 1, and the Hessian approximation matrix B_k is updated by (2.7). Therefore, if the matrix B_k is positive definite, the matrix B_{k+1} is also positive definite for every $k = 0, 1, 2, \ldots$ (see Lemma 2.1). Therefore, by Assumption 1 for $\Phi(z)$, the system (2.13) has a unique solution

$$d_k = -B_k^{-1} \nabla \Phi(z_k).$$

□

Theorem 2.2 *Assume that $\Phi(z)$ is continuously differentiable and bounded below. If $\nabla\Phi(z_k)^\top d_k < 0$, then there exists an $\alpha^* > 0$ such that*

$$\left.\begin{aligned} \Phi(z_k+\alpha^* d_k) &\le \Phi(z_k) + c\alpha^*\nabla\Phi(z_k)^\top d_k \\ \textit{and } \Phi(z_k+\alpha^* d_k) &\ge \Phi(z_k) + (1-c)\alpha^*\nabla\Phi(z_k)^\top d_k, \end{aligned}\right\} \tag{2.22}$$

where $c \in (0, 0.5)$.

Proof: We denote $\Phi_k = \Phi(z_k)$ and $\Phi_{k+1} = \Phi(z_k + \alpha d_k)$. By the hypothesis, Φ_k and Φ_{k+1} are bounded below, $c \in (0, 0.5)$ and $\nabla\Phi_k^\top d_k < 0$. For $\alpha > 0$, define

$$\begin{aligned} \psi_1(\alpha) &= \Phi(z_k+\alpha d_k) - \Phi(z_k) - c\alpha\nabla\Phi(z_k)^\top d_k \\ \text{and } \psi_2(\alpha) &= \Phi(z_k+\alpha d_k) - \Phi(z_k) - (1-c)\alpha\nabla\Phi(z_k)^\top d_k. \end{aligned}$$

Also, we have

$$\lim_{\alpha\to\infty+}\psi_1(\alpha) = \lim_{\alpha\to\infty+}\psi_2(\alpha) = +\infty \text{ and } \psi_1(0) = \psi_2(0) = 0. \tag{2.23}$$

Note that both of $\psi_1(\alpha)$ and $\psi_2(\alpha)$ are continuous and

$$\psi_2(\alpha) = \psi_1(\alpha) - (1-2c)\alpha\nabla\Phi_k^\top d_k. \tag{2.24}$$

For all sufficiently small positive α, we get

$$\begin{aligned} \psi_1(\alpha) &= \Phi(z_k+\alpha d_k) - \Phi(z_k) - c\alpha\nabla\Phi(z_k)^\top d_k \\ &= \Phi_k + \alpha\nabla\Phi_k^\top d_k + o(\alpha) - \Phi_k - c\alpha\nabla\Phi_k^\top d_k \\ &= (1-c)\alpha\nabla\Phi_k^\top d_k + o(\alpha) < 0. \end{aligned}$$

Also, we have

$$\begin{aligned}\psi_2(\alpha) &= \Phi(z_k+\alpha d_k) - \Phi(z_k) - (1-c)\alpha\nabla\Phi(z_k)^\top d_k\\ &= c\alpha\nabla\Phi_k^\top d_k + o(\alpha) < 0.\end{aligned}$$

Thus, for $\alpha \to 0+$, we have $\psi_1(\alpha) < 0$ and $\psi_2(\alpha) < 0$.

Therefore, using (2.23), there exist constants $\rho_1 > 0$ and $\rho_2 > 0$ for which $\psi_1(\rho_1) = 0$ and $\psi_2(\rho_2) = 0$. Taking ρ_1 and ρ_2 to be the infimum positive root of ψ_1 and ψ_2, respectively, we can assume that there is no zero of $\psi_1(\alpha)$ in $(0, \rho_1)$ and no zero of $\psi_2(\alpha)$ in $(0, \rho_2)$.

Let $\bar{\alpha}$ be the global minimizer of $\psi_1(\alpha)$ in $[0, \rho_1]$. The minimum value cannot occur at the endpoints because $\psi_1(0) = 0$ and $\psi_1(\rho_1) = 0$, and there exists $\alpha \in (0, \rho_1]$ that satisfies $\psi_1(\alpha) < 0$. Therefore, there exists at least one local minimizer $\alpha^* \in (0, \rho_1)$ such that $\psi_1(\alpha^*) < 0$, and $\psi_1(\alpha) < 0$ for every $\alpha \in (0, \rho_1)$. Then,

$$\begin{aligned}&\psi_1(\alpha^*) = \Phi(z_k+\alpha^* d_k) - \Phi(z_k) - c\alpha^*\nabla\Phi(z_k)^\top d_k < 0\\ \Longrightarrow\ &\Phi(z_k+\alpha^* d_k) < \Phi(z_k) + c\alpha^*\nabla\Phi(z_k)^\top d_k.\end{aligned}$$

Next, we prove that $\rho_1 > \rho_2$. Using $\psi_2(\rho_2) = 0$, we have

$$\begin{aligned}&\psi_2(\rho_2) = \psi_1(\rho_2) - (1-2c)\rho_2\nabla\Phi_k^\top d_k = 0\\ \Longrightarrow\ &\psi_1(\rho_2) = (1-2c)\rho_2\nabla\Phi_k^\top d_k < 0.\end{aligned}$$

Therefore, $\rho_2 \in (0, \rho_1)$, $\rho_2 < \rho_1$. If we choose $\alpha^* = \rho_2 + \epsilon$ where $\epsilon > 0$ is such a quantity $\alpha^* \notin (0, \rho_2)$ and $\alpha^* < \rho_1$), then $\psi_1(\alpha^*) < 0$ and $\psi_2(\alpha^*) > 0$. Therefore,

$$\begin{aligned}&\psi_1(\alpha^*) = \Phi(z_k+\alpha^* d_k) - \Phi(z_k) - c\alpha^*\nabla\Phi(z_k)^\top d_k < 0\\ \Longrightarrow\ &\Phi(z_k+\alpha^* d_k) < \Phi(z_k) + c\alpha^*\nabla\Phi(z_k)^\top d_k,\end{aligned}$$

and

$$\psi_2(\alpha^*) = \Phi(z_k + \alpha^* d_k) - \Phi(z_k) - (1-c)\alpha^* \nabla\Phi(z_k)^\top d_k > 0$$
$$\implies \Phi(z_k + \alpha^* d_k) > \Phi(z_k) + (1-c)\alpha^* \nabla\Phi(z_k)^\top d_k.$$

Thus, there exists a constant $\alpha^* \in [\rho_2, \rho_1]$ that satisfies (2.22).

□

Using Theorem 2.1 and Theorem 2.2, we note that for every iteration k in Algorithm 1, there exist d_k and α_k that satisfy (2.13) and (2.14), respectively.

Theorem 2.3 *Let Assumption 1 holds for Φ. Let $\{z_k\}$ is obtained by Algorithm 1. If there are positive constants $\beta_1, \beta_2, \beta_3 > 0$ for which the relations*

$$\|B_k s_k\| \leq \beta_1 \|s_k\| \text{ and } \beta_2 \|s_k\|^2 \leq s_k^\top B_k s_k \leq \beta_3 \|s_k\|^2 \tag{2.25}$$

follow for infinitely many k, then

$$\liminf_{k\to\infty} \|\nabla\Phi(z_k)\| = 0. \tag{2.26}$$

Proof: Since we have $s_k = z_{k+1} - z_k = \alpha_k d_k$, one can notice that (2.25) follows when s_k is replaced with d_k, i.e.,

$$\begin{aligned} B_k s_k &= B_k \alpha_k d_k = \alpha_k B_k d_k, \\ \text{and } &\|B_k s_k\| \leq \beta_1 \|s_k\| \\ \implies &\|\alpha_k B_k d_k\| \leq \beta_1 \|\alpha_k d_k\| \\ \implies &|\alpha_k| \|B_k d_k\| \leq \beta_1 |\alpha_k| \|d_k\| \\ \implies &\|B_k d_k\| \leq \beta_1 \|d_k\|. \end{aligned} \tag{2.27}$$

Also,

$$s_k^\top B_k s_k = \alpha_k^2 d_k^\top B_k d_k \text{ and } \|s_k\|^2 = \alpha_k^2 \|d_k\|^2.$$

From (2.25), we have

$$\beta_2 \alpha_k^2 \|d_k\|^2 \leq \alpha_k^2 d_k^\top B_k d_k \leq \beta_3 \alpha_k^2 \|d_k\|^2, \tag{2.28}$$

$$\implies \beta_2 \|d_k\|^2 \leq d_k^\top B_k d_k \leq \beta_3 \|d_k\|^2. \tag{2.29}$$

Thus, d_k satisfies (2.25). Let K be the set of indices k's such that (2.25) holds. Therefore, $B_k d_k = -\nabla\Phi(z_k)$ and (2.25) implies

$$\|\nabla\Phi(z_k)\| \leq \beta_1 \|d_k\|$$

and

$$\beta_2 \|d_k\|^2 \leq d_k^\top B_k d_k \leq \|d_k\| \|B_k d_k\| = \|d_k\| \|\nabla\Phi(z_k)\|$$
$$\text{i.e., } \beta_2 \|d_k\| \leq \|\nabla\Phi(z_k)\|.$$

Thus,

$$\beta_2 \|d_k\| \leq \|\nabla\Phi(z_k)\| \leq \beta_1 \|d_k\|. \tag{2.30}$$

The step length α_k is the largest value in the set $\left\{\bar{c}\; 2^{-i_k} \;\middle|\; \bar{c} \in [1,\infty) \text{ is fixed, and } i = 0, 1, \ldots\right\}$ such that i_k satisfies (2.14). Therefore, from Armijo rule (2.14) we have

$$\Phi(z_k + \bar{c}\; 2^{-(i_k-1)} d_k) > \Phi(z_k) + c\bar{c}\; 2^{-(i_k-1)} \nabla\Phi(z_k)^\top d_k. \tag{2.31}$$

By mean-value theorem, there is a $\theta_k \in (0,1)$ for which $\Phi(z_k + \bar{c}\, 2^{-(i_k-1)} d_k) - \Phi(z_k)$

$$
\begin{aligned}
&= \bar{c}\, 2^{-(i_k-1)} \nabla\Phi(z_k + \theta_k \bar{c}\, 2^{-(i_k-1)} d_k)^\top d_k \\
&= \bar{c}\, 2^{-(i_k-1)} [\nabla\Phi(z_k)^\top d_k + \left(\nabla\Phi(z_k + \theta_k \bar{c}\, 2^{-(i_k-1)} d_k) - \nabla\Phi(z_k)\right)^\top d_k] \\
&\leq \bar{c}\, 2^{-(i_k-1)} [\nabla\Phi(z_k)^\top d_k + \|\nabla\Phi(z_k + \theta_k \bar{c}\, 2^{-(i_k-1)} d_k) - \nabla\Phi(z_k)\| \|d_k\|] \\
&\leq \bar{c}\, 2^{-(i_k-1)} [\nabla\Phi(z_k)^\top d_k + L\|\theta_k \bar{c}\, 2^{-(i_k-1)} d_k\| \|d_k\|] \\
&\leq \bar{c}\, 2^{-(i_k-1)} \nabla\Phi(z_k)^\top d_k + L\bar{c}\,^2 2^{-2(i_k-1)} \|d_k\|^2, \qquad (2.32)
\end{aligned}
$$

where $L > 0$ is the Lipschitz constant, and the second last inequality is obtained using (2.19). Using (2.32) in (2.31), for any $k \in K$,

$$
\begin{aligned}
&\bar{c}\, 2^{-(i_k-1)} \nabla\Phi(z_k)^\top d_k + L\bar{c}\,^2 2^{-2(i_k-1)} \|d_k\|^2 > c\bar{c}\, 2^{-(i_k-1)} \nabla\Phi(z_k)^\top d_k \\
\Longrightarrow\ & 2^{-(i_k)} > \frac{-(1-c)\nabla\Phi(z_k)^\top d_k}{2\bar{c}\, L\|d_k\|^2} = \frac{(1-c) d_k^\top B_k d_k}{2\bar{c}\, L\|d_k\|^2} \\
\Longrightarrow\ & 2^{-(i_k)} \geq \frac{(1-c)}{2\bar{c}} \beta_2 L^{-1} > 0. \qquad (2.33)
\end{aligned}
$$

The last inequality in (2.33) is obtained using (2.25). It is clear from the inequality (2.33) that the step-length $\{\bar{c}\, 2^{-i_k}\}_{k \in K}$ is bounded away from zero.

Therefore, from (2.13) and (2.18), we have $d^\top B_k d_k = -\nabla\Phi(z_k)^\top d_k \to 0$ as $k \to \infty$, $k \in K$. Using (2.30) and (2.25), we get

$$
\liminf_{k \to \infty} \|\nabla\Phi(z_k)\| = 0.
$$

□

To establish the main theorem, we use the below result.

Lemma 2.3 *(See [62]) Let B_k be updated by the BFGS formula*

$$B_{k+1} = B_k + \frac{y_k^* y_k^{*\top}}{y_k^{*\top} s_k} - \frac{B_k s_k s_k^\top B_k}{s_k^\top B_k s_k}.$$

Suppose B_0 is symmetric and positive definite and there are positive constants $m \leq M$ such that for all $k \geq 0, y_k^*$ and s_k follow the inequalities

$$\frac{y_k^{*\top} s_k}{\|s_k\|^2} \geq m \ \textit{and} \ \frac{\|y_k^*\|^2}{y_k^{*\top} s_k} \leq M. \tag{2.34}$$

Then, there exist constants $\beta_1, \beta_2, \beta_3 > 0$ such that for any positive integer t, (2.25) satisfy for at least $[t/2]$ values of $k \in \{1, 2, \ldots, t\}$.

Theorem 2.4 *Let Assumption 1 holds and $\{z_k\}$ is obtained by Algorithm 1, then*

$$\liminf_{k\to\infty} \|\nabla\Phi(z_k)\| = 0 \tag{2.35}$$

Proof: By Theorem 2.3, we show that there are infinitely many indices k which follows (2.25).

If $\bar{K}$ in (2.20) is a finite set, then B_k remains constant after a finite number of iterations. As B_k is symmetric and positive definite for every k, it is clear that there are constants $\beta_1, \beta_2, \beta_3 > 0$ for which (2.25) follows for infinitely many k. Therefore, using Theorem 2.3, we get

$$\liminf_{k\to\infty} \|\nabla\Phi(z_k)\| = 0.$$

Assume the case when $\bar{K}$ is an infinite set. To prove the result for this case, we assume the contrary that (2.35) is not true, i.e., there exists a constant $\delta > 0$ for which

$\|\nabla\Phi(z_k)\| > \delta$, for each $k \in \bar{K}$. Therefore, using (2.20), we get

$$\frac{{y_k^*}^\top s_k}{\|s_k\|^2} \geq \beta\delta^\alpha = m, \tag{2.36}$$

$$\text{i.e., } {y_k^*}^\top s_k \geq \beta\delta^\alpha \|s_k\|^2 \text{ for every } k \in \bar{K}. \tag{2.37}$$

From Assumption 1, we have

$$\|y_k\|^2 \leq \|s_k\|^2, \text{ for every } z_1, z_2 \in \Omega.$$

From (2.14),

$$-c\nabla\Phi(z_k)^\top s_k \leq \Phi(z_k) - \Phi(z_k + s_k) \leq -(1-c)\nabla\Phi(z_k)^\top s_k < \nabla\Phi(z_k)^\top s_k. \tag{2.38}$$

Now,

$$\begin{aligned}
\bar{a}_k &= \frac{6(\Phi(z_k) - \Phi(z_k + s_k)) + 3(\nabla\Phi(z_k) + \nabla\Phi(z_k + s_k))^\top s_k}{\|s_k\|^2} \\
&\leq \frac{6(-\nabla\Phi(z_k)^\top s_k) + 3(2\nabla\Phi(z_k) + \nabla\Phi(z_k + s_k) - \nabla\Phi(z_k))^\top s_k}{\|s_k\|^2} \\
&\leq \frac{3(\nabla\Phi(z_k + s_k) - \nabla\Phi(z_k))^\top s_k}{\|s_k\|^2}.
\end{aligned}$$

Therefore, using (2.19), we have

$$|\bar{a}_k| \leq \frac{3\|(\nabla\Phi(z_k + s_k) - \nabla\Phi(z_k))^\top s_k\|}{\|s_k\|^2} \leq \frac{3L\|s_k\|^2}{\|s_k\|^2} = 3L. \tag{2.39}$$

When $\bar{a}_k \leq 0$, $y_k^* = y_k$. Then, we have $\|y_k^*\| = \|y_k\| \leq L\|s_k\|$ and

$$\frac{\|y_k^*\|^2}{{y_k^*}^\top s_k} \leq \frac{L\|s_k\|^2}{\beta\delta^\alpha\|s_k\|^2} = \frac{L}{\beta\delta^\alpha} = M_1. \tag{2.40}$$

When $\bar{a}_k > 0$, $y_k^* = y_k + \bar{a}_k s_k$. Then, we have $\|y_k^*\| \leq \|y_k\| + |\bar{a}_k|\|s_k\| \leq L\|s_k\| + 3L\|s_k\| =$

$4L\|s_k\|$ and

$$\frac{\|y_k^*\|^2}{y_k^{*\top} s_k} \le \frac{16L^2\|s_k\|^2}{\beta\delta^\alpha\|s_k\|^2} = \frac{16L^2}{\beta\delta^\alpha} = M_2. \tag{2.41}$$

Applying Lemma 2.3 with (2.36) and (2.40), (2.41) to the matrix sequence $\{B_k\}_{k\in\bar{K}}$, then there are constants $\beta_1, \beta_2, \beta_3 > 0$ for which (2.25) holds for infinitely many k. Therefore, using Theorem 2.3, we have

$$\liminf_{k\to\infty} \|\nabla\Phi(z_k)\| = 0,$$

which contradicts our assumption that there exists a constant $\delta > 0$ such that $\|\nabla\Phi(z_k)\| > \delta$, for each k.

Therefore, in the case where $\bar{K}$ is an infinite set, we have

$$\liminf_{k\to\infty} \|\nabla\Phi(z_k)\| = 0,$$

which completes the proof. □

Theorem 2.5 *If Assumption 1 holds and $\{z_k\}$ is obtained by Algorithm 2, then*

$$\liminf_{k\to\infty} \|\nabla\Phi(z_k)\| = 0 \tag{2.42}$$

Proof: The proof is analogous to that of Theorem 2.4. □

2.7 Numerical Results

In the following section, we solve five generalized Nash equilibrium problems using Algorithms 1 and 2. Yang et al. [45] reported a numerical comparison of the BFGS method with MWWP and WWP line search techniques. Here, we provide numerical performances of the BFGS method with Armijo-Goldstein (Algorithm 1) and MWWP rules (Algorithm 2) in the GNEP set-up.

In identifying numerical performances, Algorithms 1 and 2 are coded in MATLAB software (version: 9.12.0.2009381 (R2022a)) on a CPU of i5-10th generation. During the compilation of algorithms, we use a stopping condition $\|\nabla\Phi(z_k)\| < \epsilon = 10^{-8}$. Other algorithmic parameter values are as follows:

- for choosing the step-length α_k, we take $\bar{c} = 100$,
- $c = 0.01$, $c_1 = \frac{c}{3}$, $\theta = 1 - c$,
- $\beta = 10^{-8}$, and
- for the Step 4 in Algorithm 1, in applying (2.7), we take

$$\alpha = \begin{cases} 0.01, & \text{if } \|\nabla\Phi(z_k)\| \geq 1 \\ 3, & \text{otherwise.} \end{cases}$$

The parameters β and α are used only in the caution BFGS-update matrix (2.1) in Algorithm 1. In our experiments, algorithms stop whenever $\|\nabla\Phi(z_k)\| < \epsilon$ or the non-negative integer $i > 200$ in step-length α_k.

In the comparison tables, we have specified some regions for starting points. We have randomly taken 300 starting points from each of the specified regions and presented the minimum, median, and maximum of the number of iterations and CPU time consumed by Algorithms 1 and 2.

Problem 2.1 *Consider the following game with two players:*

$$\left.\begin{array}{r}\min\limits_{x_1}\,(x_1-1)^2 \\ \\ \textit{subject to } x_1+x_2\le 1\end{array}\right\} \qquad \textit{and} \qquad \left\{\begin{array}{l}\min\limits_{x_2}\,\left(x_2-\frac{1}{2}\right)^2 \\ \\ \textit{subject to } x_1+x_2\le 1.\end{array}\right.$$

This problem was considered by Facchinei et al. [1]. This problem has infinitely many solutions, which are

$$S = \{(\bar{s}, 1-\bar{s}) \mid \bar{s} \in [0.5, 1]\}.$$

This GNEP has many equilibria: $(0.2, 0.3)$, $(0.50, 0.49)$,$(0.61, 0.38)$, $(0.56, 0.43)$, *etc. A numerical comparison of the performance of Algorithms 1 and 2 on this problem is given in Table 2.1.*

Table 2.1: Performances of Algorithm 1 and Algorithm 2 on Problem 2.1

Region of initial point	Algorithm 1						Algorithm 2					
	Iteration number			Computation time			Iteration number			Computation time		
	Min	Median	Max	Min	Median	Max	Min	Median	Max	Min	Median	Max
$\|x_0\| \le 1$	33	35	40	9.8155611	11.5301793	12.9420471	33	39.5	44	10.0071156	11.9353416	15.1988879
$1 < \|x_0\| \le 5$	33	43	64	9.9780662	12.8930529	19.1935193	35	45	71	10.5464406	13.5859148	25.2763049
$5 < \|x_0\| \le 15$	46	59	71	14.0252074	17.8076591	21.5618371	51	67	90	15.3653121	20.1413819	29.9882825
$15 < \|x_0\| \le 50$	60	74	80	18.1086301	21.1665489	23.1277177	61	68	84	21.4922147	25.0860578	32.3190021
$50 < \|x_0\| \le 100$	62	71	81	18.6511131	24.4064391	29.0292234	68	80	85	22.3850595	24.0805074	35.9274275

In the numerical comparison, we have taken initial points randomly from each of the specified regions and verified the global convergence of the method. Also, we have compared the minimum, median, and maximum of the consumed number of iterations and computation costs in each specified region by solving GNEPs by both Algorithms 1 and 2. Though the Problem 2.1 is not a large-scale problem, we can see a difference in computation costs consumed by both Algorithms 1 and 2. If the initial points are near the origin, both algorithms 1 and 2 perform mostly similar and consume almost the same computation costs. However, when the initial points are far from the origin, we can see a difference in computation costs consumed by both algorithms. Note that Algorithm 1 takes lesser CPU time compared to Algorithm 2.

Problem 2.2 *Consider the following GNEP that has two players and one shared constraint $s(x) = x_1 + x_2 - 15 \leq 0$:*

$$\left.\begin{aligned} &\min_{x_1} x_1^2 + \tfrac{8}{3}x_1x_2 - 34x_1 \\ &\textit{subject to } x_1 + x_2 \leq 15,\ 0 \leq x_1 \leq 10 \end{aligned}\right\} \quad \textit{and} \quad \left\{\begin{aligned} &\min_{x_2} x_2^2 + \tfrac{5}{4}x_1x_2 - \tfrac{97}{4}x_2 \\ &\textit{subject to } x_1 + x_2 \leq 15,\ 0 \leq x_2 \leq 10. \end{aligned}\right.$$

This game was introduced by Harker [63]. This GNEP has an infinite number of solutions. Using random multi-starting values, we found that Algorithm 1 converges globally. We have compared the computation costs in solving GNEPs by Algorithms 1 and 2 in Table 2.2.

Table 2.2: Performances of Algorithm 1 and Algorithm 2 on Problem 2.2

Region of initial point	Algorithm 1						Algorithm 2					
	Iteration number			Computation time			Iteration number			Computation time		
	Min	Median	Max	Min	Median	Max	Min	Median	Max	Min	Median	Max
$\|x_0\| \leq 1$	46	55	62	367.8947949	430.0820592	481.6043705	35	59	70	410.8706878	691.6027966	822.7638993
$1 < \|x_0\| \leq 5$	42	60.5	72	331.6576986	472.4338520	562.6860231	53	58	73	621.8719814	682.5887432	846.2415418
$5 < \|x_0\| \leq 15$	49	65	85	384.0940428	508.4177229	666.8262221	61	80	95	729.9097584	947.6941509	1115.2065741
$15 < \|x_0\| \leq 50$	70	78	95	551.4164915	613.2027723	748.5838472	56	71	89	670.0887578	845.1442606	1057.3772041
$50 < \|x_0\| \leq 100$	95	104	123	765.6869913	834.1185455	984.4251712	81	95	98	946.8039653	1150.9772041	1300.8525742

We can see that both Algorithms 1 and 2 converge globally, but we can see a big difference in the minimum, median, and maximum of consumed CPU time by both algorithms. Clearly, Algorithm 1 takes lesser computation costs compared to Algorithm 2 in each specified region for the initial point. Also, in numerical comparison, we can observe that Algorithm 1 takes lesser CPU time compared to Algorithm 2.

Problem 2.3 *The following problem depicts a game with two players and single shared constraint:*

$$\left.\begin{array}{c}\min\limits_{x_1} \frac{1}{2}x_1^2 - x_1x_2 \\ \textit{subject to } x_1 + x_2 \geq 1, \\ x_1 \geq 0\end{array}\right\} \quad \textit{and} \quad \left\{\begin{array}{c}\min\limits_{x_2} x_2^2 + x_1x_2 \\ \textit{subject to } x_1 + x_2 \geq 1, \\ x_2 \geq 0.\end{array}\right.$$

This problem has been introduced by Rosen [64]. In this problem, there are two constraints $h_1(x_1) = -x_1$ *and* $h_2(x_2) = -x_2$ *that depends only on the variables of a single player, and there is one shared constraint* $s(x_1, x_2) = 1 - x_1 - x_2$*. The performance of Algorithms 1 and 2 is depicted in Table 2.3.*

Table 2.3: Performances of Algorithm 1 and Algorithm 2 on Problem 2.3

Region of initial point	Algorithm 1						Algorithm 2					
	Iteration number			Computation time			Iteration number			Computation time		
	Min	Median	Max	Min	Median	Max	Min	Median	Max	Min	Median	Max
$\|x_0\| \leq 1$	30	35	43	190.7484509	233.5142974	280.0147215	35	40.5	46	279.2283089	364.9990586	380.3896140
$1 < \|x_0\| \leq 5$	35	43.5	46	211.6401868	263.1287694	277.8278983	37	42.5	47	288.3394868	382.7355782	406.0382182
$5 < \|x_0\| \leq 15$	34	46	50	206.7644701	279.6569243	305.4529313	36	45	47	324.7146608	407.6843183	430.5627083
$15 < \|x_0\| \leq 50$	45	50.5	61	276.7633172	311.7309502	373.2828701	50	59.5	86	460.2240536	544.4629340	772.6540426
$50 < \|x_0\| \leq 100$	47	58	64	286.6669676	354.7094388	391.3671929	49	62	69	452.0246391	560.8488727	636.3005903

Here, we observe that both Algorithms 1 and 2 take almost the same number of iterations but have a big difference in their consumed CPU time. We have provided regions where we have randomly taken initial points. We can see that Algorithm 1 is cost-effective compared to Algorithm 2.

Problem 2.4 *This problem is an internet switching model which was proposed by Kesselman et al. [65], where the traffic is generated by selfish users. The model depicts the behavior of users sharing the first-in-first-out buffer with bounded capacity. The utility of each user depends on its transmission rate and congestion level. Specifically, we consider that there are* N *users, and the buffer capacity is* B*. The user* v *controls the amount of his "packets" in the buffer, denoted by* $x^v \in [0, \infty)$*. The utility* *function for the player v $(v = 1, 2, 3, \ldots, N)$ is given by*

$$\theta_v(x^v, \mathbf{x}^{-v}) = -\frac{x^v}{x^1 + x^2 + \cdots + x^N}\left(1 - \frac{x^1 + x^2 + \cdots + x^N}{B}\right), \ (x^v, \mathbf{x}^{-v}) \in \mathbb{R}^N \tag{2.43}$$

and the constraints are

$$x^1 + x^2 + \cdots + x^N \leq B \text{ and } x^v \geq l_v,$$

where $l_v \geq 0$. Kesselman et al. [65] have shown that model (2.43) has a unique solution $\bar{x}^v = B(N-1)/N^2$, $v = 1, 2, \ldots, N$.

We take $N = 4$ players, $l_1 = 0.01$, $l_2 = 0.01$, $l_3 = 0.01$, $l_4 = 0.01$ and $B = 1$ for numerical computation. The problem has a total of 4 variables, and the system (2.12) pertaining to solve this GNEP involves nine variables. The GNEP problem (2.43) for four players has a unique solution $\bar{x}^1 = \bar{x}^2 = \bar{x}^3 = \bar{x}^4 = 0.1875$.
In employing Algorithms 1 and 2 on this problem, we randomly took the initial points from the region indicated in Table 2.4. For some initial points, it converges to $\bar{x}^1 = \bar{x}^2 = \bar{x}^3 = \bar{x}^4 = 0.20$. The numerical performance of Algorithms 1 and 2 on this GNEP is provided in Table 2.4, which depicts that Algorithms 1 is cost-efficient compared to Algorithm 2 corresponding to each specified region for initial points.

Table 2.4: Performances of Algorithm 1 and Algorithm 2 on Problem 2.4

Region of initial point	Algorithm 1						Algorithm 2					
	Iteration number			Computation time			Iteration number			Computation time		
	Min	Median	Max	Min	Median	Max	Min	Median	Max	Min	Median	Max
$\|x_0\| \leq 0.2$	110	130.5	135	60.8716676	69.9992739	80.7818287	120	143	170	78.5046025	82.9273549	105.4812018
$0.2 < \|x_0\| \leq 0.4$	91	109.5	128	48.3546525	57.6142135	67.3743387	102	125	140	58.7771746	71.6256916	84.7519417
$0.4 < \|x_0\| \leq 0.6$	60	81.5	87	32.7544586	43.5582642	46.6964562	70	91.5	96	45.1349918	60.5001818	81.0909437
$0.6 < \|x_0\| \leq 0.8$	50	76	88	27.1916965	40.7106048	47.1182006	61	83.5	101	45.7480215	60.4065841	81.3667261
$0.8 < \|x_0\| \leq 1$	89	106.5	121	48.0518683	57.2676848	70.8017430	98	111	147	72.8384523	81.3266689	101.3417314

Problem 2.5 *The following problem is another version of the internet switching model*

proposed by Kesselman et al. [65]. In this problem, there are $N = 3$ players. The variables corresponding to player v is $x^v \in \mathbb{R}$. The objective function of Player v is given by

$$\theta_v(x^v, \mathbf{x}^{-v}) = -\frac{x^v}{x^1 + x^2 + x^3}\left(1 - \frac{x^1 + x^2 + x^3}{B}\right), \; (x^v, \mathbf{x}^{-v}) \in \mathbb{R}^3, \tag{2.44}$$

where B is the buffer capacity. The constraints for the Player 1 are

$$0.3 \leq x^1 \leq 0.5,$$

and for the remaining players

$$x^1 + x^2 + x^3 \leq B, \text{ and } x^v \geq 0.01.$$

Here, we have taken the parameter $B = 1$ and solved this problem using both Algorithms 1 and 2. The system (2.12) has a total 8 variables. The GNEP problem (2.44) for 3 players has a unique solution $\bar{x}^1 = 0.2999$ and $\bar{x}^2 = \bar{x}^3 = 0.2055$. The numerical performance of both Algorithms 1 and 2 on this GNEP is shown in Table 2.5. Table 2.5 shows that Algorithm 1 takes lesser computation time and iteration numbers compared to Algorithm 2.

Table 2.5: Performances of Algorithm 1 and Algorithm 2 on Problem 2.5

Region of initial point	Algorithm 1						Algorithm 2					
	Iteration number			Computation time			Iteration number			Computation time		
	Min	Median	Max	Min	Median	Max	Min	Median	Max	Min	Median	Max
$\|x_0\| \leq 0.2$	142	145.5	180	60.5156996	69.8729060	75.1300861	146	187.5	200	75.4941229	95.8196587	110.6429374
$0.2 < \|x_0\| \leq 0.4$	134	142.5	126	54.5175096	57.8729938	70.8400814	140	155	186	65.1114390	89.4622365	96.7129967
$0.4 < \|x_0\| \leq 0.6$	90	94	126	36.6010346	38.1234327	51.1466707	98	114	150	45.6356939	59.8191949	71.2675442
$0.6 < \|x_0\| \leq 0.8$	82	125.5	146	33.2986903	51.0556914	59.3054762	92	139.5	175	47.4608546	64.4523775	91.8372476
$0.8 < \|x_0\| \leq 1$	109	130	161	44.2290509	52.8775267	65.5032156	120	137.5	181	54.4723432	68.4284650	95.1546866

2.8 Conclusion

In this chapter, we have solved GNEPs by an improved BFGS using the two line search techniques: The Armijo-Goldstein line search technique (see Algorithm 1) and MWWP line search technique (see Algorithm 2). We have reformulated GNEPs into a smooth system of equations (2.12), and by incorporating the merit function (2.11), we have solved GNEPs by improved BFGS method using two line search techniques. The BFGS method using MWWP line search technique converges globally and works well compared to other quasi-Newton methods (see a detailed comparison in [45]). However, we have used the Armijo-Goldstein line search technique to minimize computation costs. The improved BFGS method equipped with Armijo-Goldstein line search technique takes lesser computation costs than the MWWP-line search technique. We have solved five numerical problems using Algorithms 1 and 2, and have given a numerical comparison of both algorithms.

****