

Chapter 2

Bacteria Inspired Algorithms

Abstract In this chapter, we present a set of algorithms that are inspired by the different bacteria behavioural patterns, i.e., bacterial foraging algorithm (BFA), bacterial colony chemotaxis (BCC) algorithm, superbug algorithm (SuA), bacterial colony optimization (BCO) algorithm, and viral system (VS) algorithm. We first describe the general knowledge of bacteria foraging behaviour in [Sect. 2.1](#). Then, the fundamentals and performances of BFA, BCC algorithm, SuA, BCO algorithm, and VS algorithm are introduced in [Sects. 2.2](#) and [2.3](#), respectively. Finally, [Sect. 2.4](#) summarises this chapter.

2.1 Introduction

It is fair to say that bacteria can be observed almost everywhere, from the most hospitable surroundings to the most hostile environment (Helden et al. [2012](#)). Some of them are harmful but majority of them are beneficial to the nature. They comprise various attributes (e.g., shape, texture, and metabolism) and different behavioural patterns (e.g., foraging, reproduction, and movement) (Modrow et al. [2013](#); Giguère et al. [2013](#)). In addition to purely scientific aspects related to the bacteria, those characteristics also initiate computer scientists to develop algorithms for the solution of optimization problems. The first attempts appeared in the late 1970s [e.g., bacterial chemotaxis algorithm that proposed by Bremermann ([1974](#))]. Since then, similar ideas have attracted a steadily increasing amount of research. In this chapter, a set of bacteria inspired algorithms are collected and introduced as follows:

- [Section 2.2](#): Bacterial Foraging Algorithm.
- [Section 2.3.1](#): Bacterial Colony Chemotaxis.
- [Section 2.3.2](#): Superbug Algorithm.
- [Section 2.3.3](#): Bacterial Colony Optimization.
- [Section 2.3.4](#): Viral System.

The effectiveness of these newly developed algorithms are validated through the testing on a wide range of benchmark functions and engineering design problems, and also a detailed comparison with various traditional performance leading computational intelligence (CI) algorithms, such as particle swarm optimization (PSO), genetic algorithm (GA), differential evolution (DE), evolutionary algorithm (EA), fuzzy system (FS), ant colony optimization (ACO), and simulated annealing (SA).

2.1.1 Bacteria

Bacteria are micro-organisms that include a large domain. For example, there are roughly 5,000 operational taxonomic unit of bacteria for every gram of soil (Maheshwari 2012). In addition, they have a wide range of shapes, such as spheres, spirals, and rods. Also, they live in everywhere, such as in plants, animals, and human's body. Due to their diverse ecology, they exhibit multifarious functional characters beneficial in nature.

2.1.2 Bacterial Foraging Behaviour

The bacterial foraging behaviour (i.e., bacterial movement) has been studied by biologists for many years, but there are very few reports until 70s (Berg and Brown 1972; Adler 1975). During foraging, there are two types of bacterial motility: i.e., flagellum-dependent and flagellum-independent. In addition, biologists found that bacterial movement was not random and arbitrary. Instead, bacterial cells exhibited directed movement toward certain stimuli and away from others (Adler 1975). This behaviour is called bacterial chemotaxis in the literature. Since its discovery, chemotactic behaviour has stimulated the curiosity of numerous investigators. Nowadays, those research outcomes have been adopted by the models simulating bacterial foraging patterns (Paton et al. 2004).

2.2 Bacterial Foraging Algorithm

2.2.1 Fundamentals of Bacterial Foraging Algorithm

Bacterial foraging algorithm (BFA) was originally proposed by Passino (2002) in which the foraging strategy of *E. Coli* bacteria has been simulated. Typically, the BFA consists of four main mechanisms: chemotaxis, swarming, reproduction, and elimination-dispersal event. The main steps of BFA are outlined below (Passino 2002):

- Chemotaxis: The movement of *E. Coli* bacteria can be performed through two different ways: swimming which means the movement in the same direction, and tumbling refers the movement in a random direction. The movement of the i th bacterium after one step is given by Eqs. 2.1 and 2.2, respectively (Passino 2002):

$$\theta^i(j+1, k, l) = \theta^i(j, k, l) + C(i)\phi(j), \quad (2.1)$$

$$\begin{cases} \theta^i(j+1, k, l) > \theta^i(j, k, l), & \text{swimming in which } \phi(j) = \phi(j-1) \\ \theta^i(j+1, k, l) < \theta^i(j, k, l), & \text{tumbling in which } \phi(j) \in [0, 2\pi] \end{cases} \quad (2.2)$$

Where $\theta^i(j, k, l)$ denotes the location of i th bacterium at j th chemotactic, k th reproductive and l th elimination and dispersal step, $C(i)$ is the length of unit walk, and $\phi(j)$ is the direction angle of the j th step.

- Swarming: Under the stresses circumstances, the bacteria release attractants to signal bacteria to swarm together, while they also release a repellent to signal others to be at a minimum distance from it. The cell to cell signalling can be represented by Eq. 2.3 (Passino 2002):

$$\begin{aligned} J_{cc}(\theta, P(j, k, l)) &= \sum_{i=1}^S J_{cc}^i(\theta, \theta^i(j, k, l)) \\ &= \sum_{i=1}^S \left[-d_{attract} \exp \left(-\omega_{attract} \sum_{m=1}^P (\theta_m - \theta_m^i)^2 \right) \right] \\ &\quad + \sum_{i=1}^S \left[h_{repellant} \exp \left(-\omega_{repellant} \sum_{m=1}^P (\theta_m - \theta_m^i)^2 \right) \right]. \end{aligned} \quad (2.3)$$

Where $J_{cc}(\theta, P(j, k, l))$ is the objective function value to be added to the actual objective function, S is the total number of bacteria, P is the number of variables to be optimized which are present in each bacterium, $\theta = [\theta_1, \theta_2, \dots, \theta_P]^T$ denotes a point in the P -dimensional search domain, $d_{attract}$ is the depth of the attractant released by the cell, $\omega_{attract}$ is a measure of the width of the attractant signal, $h_{repellant}$ is the height of the repellent effect (i.e., $h_{repellant} = d_{attract}$), and $\omega_{repellant}$ is the measure of the width of the repellent.

- Reproduction: After N_c chemotaxis steps, the reproduction step should be performed. The fitness value of the bacteria is stored in an ascending order. The working principle is the least health bacteria eventually die and the remaining bacteria (i.e., healthiest bacteria) will be divided into two identical ones and placed at the same location.

- Dispersion and elimination: For the purpose to avoid local optima, dispersion and elimination process is performed after a certain number of reproduction steps. According to a present probability (p_{ed}), a bacterium is chosen to be dispersed and moved to another position within the environment.

Taking into account the key phases described above, the steps of implementing BFA can be summarized as follows (Passino 2002; Boussaïd et al. 2013; El-Abd 2012; Tang and Wu 2009):

- Step 1: Defining the optimization problem, and initializing the optimization parameters.
- Step 2: Iterative algorithm for optimization. In this step the bacterial population, chemotaxis loop ($j = j + 1$), reproduction loop ($k = k + 1$), and elimination and dispersal operations loop ($l = l + 1$) are performed.
- Step 3: If $j < N_c$, go to the chemotaxis process.
- Step 4: Reproduction.
- Step 5: If $k < N_{re}$, go to reproduction process.
- Step 6: Elimination-dispersal.
- Step 7: If $l < N_{ed}$, then go to elimination-dispersal process; otherwise end.

2.2.2 Performance of BFA

In order to show how the BFA performs, a set of experimental studies were adopted in the literature (Passino 2002). Computational results showed that BFA gives some promising results.

2.3 Emerging Bacterial Inspired Algorithms

In addition to the aforementioned BFA, the characteristics of this interesting biological organisms also motivate researchers to develop other bacterial inspired innovative CI algorithms.

2.3.1 Bacterial Colony Chemotaxis Algorithm

2.3.1.1 Fundamentals of Bacterial Colony Chemotaxis Algorithm

Bacterial colony chemotaxis (BCC) algorithm was originally proposed by Li et al. (2005) that is based on bacterial chemotaxis algorithm (Bremermann and Anderson 1991) and incorporated the communication mechanisms between the colony members. There are several variants and applications relative to BCC can be found in the literature (Li et al. 2009; Li and Li 2012; Sun et al. 2012; Lu et al.

2013; Irizarry 2011). The basic steps of BCC are as follows (Lu et al. 2013; Müller et al. 2002; Sun et al. 2012):

- Step 1: Initial Bacterial Positions. Generate N bacteria randomly in the search space, the velocity is assumed constant by Eq. 2.4 (Lu et al. 2013; Müller et al. 2002):

$$v = \text{const.} \quad (2.4)$$

- Step 2: Optimize by a single bacterium.

Compute the duration of the trajectory τ from the distribution of a random variable with an exponential probability density function via Eq. 2.5 (Lu et al. 2013; Müller et al. 2002):

$$P(X = \tau) = \frac{1}{T} e^{-\tau/T}, \quad (2.5)$$

where the expectation value $\mu = E(X) = T$, and the variance $\sigma^2 = \text{Var}(X) = T^2$. The time T is given by Eq. 2.6 (Müller et al. 2002):

$$T = \begin{cases} T_0 & \text{for } \frac{f_{pr}}{l_{pr}} \geq 0, \\ T_0 \left(1 + b \cdot \left\{ \frac{f_{pr}}{l_{pr}} \right\} \right) & \text{for } \frac{f_{pr}}{l_{pr}} < 0, \end{cases} \quad (2.6)$$

where T_0 is minimal mean time, f_{pr} is the difference between the actual and the previous function value, l_{pr} is the length of the previous step and $l_{pr} = |\bar{x}_{pr}|$, \bar{x}_{pr} denotes vector connecting the previous and the actual position in the parameter space, and b is dimensionless parameter.

The position of a bacterium is defined by x with a radius and angles via Eq. 2.7 (Lu et al. 2013):

$$\begin{aligned} x_1 &= \gamma \prod_{s=1}^{n-1} \cos(\phi_s), \\ x_i &= \gamma \sin(\phi_{i-1}) \gamma \prod_{s=i}^{n-1} \cos(\phi_s), \quad \text{for } i = 2, 3, \dots, n, \\ x_n &= \gamma \sin(\cos(\phi_{n-1})), \end{aligned} \quad (2.7)$$

where γ is the radius, and $\Phi = \{\phi_1, \phi_2, \dots, \phi_{n-1}\}$ denote angles. According to the Gaussian distribution, the angel (ϕ_i) between the previous and the new direction for turning left or right are defined by Eqs. 2.8 and 2.9, respectively (Lu et al. 2013):

$$P(X_i = \phi_i, v_i = \mu_i) = \frac{1}{\sigma_i \sqrt{2\pi}} \exp \left[-\frac{(\phi_i - v_i)^2}{2\sigma_i^2} \right], \quad (2.8)$$

$$P(X_i = \phi_i, v_i = -\mu_i) = \frac{1}{\sigma_i \sqrt{2\pi}} \exp \left[-\frac{(\phi_i - v_i)^2}{2\sigma_i^2} \right], \quad (2.9)$$

where $\phi_i \in [0^\circ, 180^\circ]$. The expectation value and the standard deviation are defined by Eqs. 2.10 and 2.11, respectively (Lu et al. 2013):

$$\begin{aligned} \mu &= E(X_i) = 62^\circ(1 - \cos(\theta)), \\ \sigma_i &= \sqrt{\text{Var}(X_i)} = 26^\circ(1 - \cos(\theta)), \end{aligned} \quad (2.10)$$

$$\cos(\theta) = \begin{cases} 0 & \text{for } \frac{f_{pr}}{l_{pr}} \geq 0, \\ \exp(-\tau_c / \tau_{pr}) & \text{for } \frac{f_{pr}}{l_{pr}} < 0, \end{cases} \quad (2.11)$$

where τ_{pr} is the duration of the previous step, and τ_c is the correlation time.

Compute the new position (\bar{x}_{new}) via Eqs. 2.12 and 2.13, respectively (Lu et al. 2013):

$$\bar{x}_{new} = \bar{x}_{old} + \bar{n}_u \cdot l, \quad (2.12)$$

$$l = v \cdot t, \quad (2.13)$$

where l is the length of the path, and \bar{n}_u is the normalized new direction vector.

- Step 3: Optimize by the bacterial colony.

Initialize the number of bacteria colony and the parameters, such as the initial starting precision (ε_{begin}), the constant of updating precision (ε_{cons}), and the target precision (ε_{end}).

The best neighbour centre position of the i th bacterium is given via Eq. 2.14 (Sun et al. 2012):

$$center_position(i) = rand() \cdot dis(\bar{x}_{i,t}, Center(\bar{x}_{i,t})), \quad (2.14)$$

where $dis(\bar{x}_{i,t}, Center(\bar{x}_{i,t}))$ is the distance between the bacterium i and the centre position, and $rand()$ is a random number meeting the uniform distribution in the interval of $(0, 1)$.

Based on the single bacterium movement process, the next position of i th bacterium can be defined by Eqs. 2.15 and 2.16, respectively (Sun et al. 2012):

$$individual_position(i) = current_position(i) + next(i), \quad (2.15)$$

$$next(i) = v \cdot d \cdot |\bar{n}_u|, \quad (2.16)$$

where v denotes the velocity, d is the duration, and \bar{n}_u is the normalized new direction vector.

- Step 4: Bacterial colony move to the new location. Compare the objective function fitness value between *center_position(i)* and *individual_position(i)*, then *i*th bacterium chooses the better value position to move.
- Step 5: If the target precision (ϵ_{end}) is gained end, otherwise turn to Step 2.

2.3.1.2 Performance of BCC

To illustrate the performance of BCC, a set of experimental studies were adopted in (Li and Li 2012; Sun et al. 2012). Computational results showed that the proposed algorithm performs well.

2.3.2 Superbug Algorithm

2.3.2.1 Fundamentals of Superbug Algorithm

Superbug algorithm (SuA) was originally proposed in (Anandaraman et al. 2012) that is based on the findings of epidemic and antiviral research (Lindsay and Holden 2006; Low et al. 1999; Bisen and Raghuvanshi 2013; Gong 2013; Desai and Meanwell 2013; Cook 2013; Morrow et al. 2012; Lahtinen et al. 2012; Helden et al. 2012). To implement SuA, the following steps need to be performed (Anandaraman et al. 2012):

- Step 1: Generating initial population.
- Step 2: Mutating the bacteria (two mechanisms are introduced in SuA, namely, inverse and pairwise interchange mutation).
- Step 3: Transferring gene between bacteria for the purpose of enhancing the fitness level.
- Step 4: Performing single point mutation on the modified bacteria.

2.3.2.2 Performance of SuA

The core concept underlying the SuA is the antibiotic resistance that a bacterium has developed during the procedure of mutation. To illustrate the performance of SuA, Anandaraman et al. (2012) applied it to the flexible manufacturing system scheduling problem. In comparison with other CI techniques (e.g., GA), the SuA showed a better performance in majority of the cases. At the end of their study, Anandaraman et al. (2012) suggested that it is possible to apply SuA to other multi-machine and dynamic job shop scheduling problems.

2.3.3 Bacterial Colony Optimization Algorithm

2.3.3.1 Fundamentals of Bacterial Colony Optimization Algorithm

Bacterial colony optimization (BCO) algorithm was originally proposed by Niu and Wang (2012) which is inspired by five basic behaviours of *E. Coli* bacteria in their whole lifecycle, i.e., chemotaxis, communication, elimination, reproduction, and migration. The main steps of BCO are outlined as follows (Niu and Wang 2012):

- Chemotaxis and communication model: In BCO, the movements of bacteria (i.e., runs and tumbles) are accompanied with communication mechanism in which three types of information (i.e., group information, personal previous information, and a random direction) are exchanged to guide them in direction and ways of movement. The bacterium runs or tumbles with communication process can be formulated via Eqs. 2.17 and 2.18, respectively (Niu and Wang 2012):

$$Position_i(T) = Position_i(T - 1) + R_i \cdot (Run_{Info}) + R\Delta(i). \quad (2.17)$$

$$Position_i(T) = Position_i(T - 1) + R_i \cdot (Tumb_{Info}) + R\Delta(i). \quad (2.18)$$

- Elimination and reproduction model: Based to the bacteria searching ability, each bacterium is marked with an energy degree. According to this level, the decisions of elimination and reproduction are defined via Eq. 2.19 (Niu and Wang 2012):

$$\begin{cases} \text{if } L_i > L_{given}, \text{ and } i \in \text{healthy}, & \text{then } i \in \text{Candidate}_{repr}, \\ \text{if } L_i < L_{given}, \text{ and } i \in \text{healthy}, & \text{then } i \in \text{Candidate}_{eli}, \\ \text{if } i \in \text{unhealthy}, & \text{then } i \in \text{Candidate}_{eli}, \end{cases} \quad (2.19)$$

where L_i denotes the i th bacterium's level of energy.

- Migration model: To avoid local optimum, the BCO's migration model is defined via Eq. 2.20 (Niu and Wang 2012):

$$Position_i(T) = rand \cdot (ub - lb) + lb, \quad (2.20)$$

where ub and lb are the upper and lower boundary, respectively, and $rand \in (0, 1)$.

In addition, in the process of tumbling, a stochastic direction are incorporated into actually running process. Therefore, the position of each bacterium is updated via Eqs. 2.21–2.23, respectively (Niu and Wang 2012):

$$\begin{aligned}
Position_i(T) = & Position_i(T-1) + C(i) \\
& * [f_i \cdot (G_{best} - Position_i(T-1)) + (1 - f_i)] \\
& * (P_{best_i} - Position_i(T-1)) + turbulent_i],
\end{aligned} \tag{2.21}$$

$$\begin{aligned}
Position_i(T) = & Position_i(T-1) + C(i) \\
& * [f_i \cdot (G_{best} - Position_i(T-1)) + (1 - f_i)] \\
& * (P_{best_i} - Position_i(T-1))],
\end{aligned} \tag{2.22}$$

$$C(i) = C_{\min} + \left(\frac{iter_{\max} - iter_j}{iter_{\max}} \right)^n \cdot (C_{\max} - C_{\min}), \tag{2.23}$$

where $turbulent_i$ is the turbulent direction variance of the i th bacterium, $f_i \in (0, 1)$, G_{best} and P_{best_i} are the globe best and personal best position of the i th bacterium, respectively, $C(i)$ is the chemotaxis step size of the i th bacterium, $iter_{\max}$ is the maximal number of iterations, and $iter_j$ is the current number of iterations.

2.3.3.2 Performance of BCO

To test the effectiveness of BCO, a set of well-known test functions were adopted in (Niu and Wang 2012). Compared with five other algorithms [i.e., PSO, GA, BFA, bacterial foraging optimization with linear decreasing chemotaxis step (BFO-LDC), and bacterial foraging optimization with nonlinear decreasing chemotaxis step (BFO-NDC)], computational results showed that BCO performs significantly better than four other algorithms (i.e., GA, BFA, BFO-LDC, and BFO-NDC) in all test functions, and BCO obtains better results than PSO in most of functions.

2.3.4 Viral System Algorithm

2.3.4.1 Fundamentals of Viral System Algorithm

Viral system (VS) algorithm was originally proposed by Cortés et al. (2008) which is based on viral infection processes. Two mechanisms called replication and infection are employed to the VS algorithm. The main steps of VS are as follows (Cortés et al. 2008, 2012; Ituarte-Villarreal and Espiritu 2011):

- Step 1: Initialisation the VS components. Each VS is defined by three components, i.e., a set of viruses, an organism, and an interaction between them. In addition, each virus includes four components, i.e., state, input, output, and process, and each organism includes two components, i.e., state and process. Overall, the VS can be described via Eqs. 2.24–2.27, respectively (Cortés et al. 2008):

$$VS = \langle \text{Virus}, \text{Organism}, \text{Interaction} \rangle, \quad (2.24)$$

$$\text{Virus} = \{\text{Virus}_1, \text{Virus}_2, \dots, \text{Virus}_n\}, \quad (2.25)$$

$$\text{Virus}_i = \langle \text{State}_i, \text{Input}_i, \text{Output}_i, \text{Process}_i \rangle, \quad (2.26)$$

$$\text{Organism} = \langle \text{State}_0, \text{Process}_0 \rangle, \quad (2.27)$$

where State_i denotes the characterises of the virus, Input_i identifies the information that the virus can collect from the organism, Output_i denotes the actions that the virus can take, Process_i represents autonomous behaviour of the virus that changing the State_i , State_0 characterises the organism state in each instant, consisting of clinical picture and the lowest healthy cell, and Process_0 represents the autonomous behaviour of the organism that tries to protect itself from the infection threat, consisting of antigen liberation.

- Step 2: Population construction. The set of feasible solutions in a specific space is given by Eq. 2.28 (Cortés et al. 2008):

$$K = \{x : g_i(x) \leq 0, \forall i = 1, 2, \dots, n\}, \quad (2.28)$$

where $x \in K$ denotes the feasible solutions and has been called a cell.

- Step 3: Define type of virus infection: selective or massive infection.

In case of massive infection: $Y - A$ cells of the neighbourhood are infected, and must be incorporated into the clinical picture. If there is not enough free space in the population, it will randomly erase the necessary cells from the $Y - A$ selected cells.

In case of selective infection: One only cell from the neighbourhood is selected according to the selective selection. The antigenic response of such cell is evaluated as a Bernoulli process (A). In case of antigenic response a lysogenic replication is initiated.

- Step 4: Define type of evolution of the virus: i.e., the lytic replication and the lysogenic replication.

In case of the lytic replication: This process starts only after a specific number of nucleus-capsids haven been replicated.

Calculate the limit number of nucleus-capsids replication (LNR) in a cell x by Eq. 2.29 (Cortés et al. 2008):

$$\text{LNR}_{\text{cell}-x} = \text{LNR}^0 \cdot \left(\frac{f(x) - f(\hat{x})}{f(\hat{x})} \right), \quad (2.29)$$

where \hat{x} is the cell that produces the best known result of the problem (in terms of $f(x)$), x is the infected cell being analysed, and LNR^0 is the initial value for LNR.

In each iteration, a number of virus replications (NR) takes place. The number of replications per iteration is calculated as function of a binomial variable (Z), adding its value to the total NR. The probability of replicating exactly z nucleus-capsids ($P(Z = z)$), as well as the average ($E(Z)$), and variance ($Var(Z)$) are given by Eqs. 2.30–2.32, respectively (Cortés et al. 2008):

$$P(Z = z) = \binom{LNR}{z} p_r^z (1 - p_r)^{LNR-z}, \quad (2.30)$$

$$E(Z) = p_r LNR, \quad (2.31)$$

$$Var(Z) = p_r(1 - p_r)LNR, \quad (2.32)$$

where p_r is the single probability of one replication, and Z is a binomial random variable, i.e., $Z = \text{Bin}(LNR, p_r)$.

Once the bacterium border is broken liberating the viruses, each one of the viruses has a probability (p_i) of infecting other new cells of the neighbourhood. The probability of infecting exactly y nucleus-capsids ($P(Y = y)$), as well as the average ($E(Y)$), and variance ($Var(Y)$) are given via Eqs. 2.33–2.35, respectively (Cortés et al. 2008):

$$P(Y = y) = \binom{|V(x)|}{y} p_i^y (1 - p_i)^{|V(x)|-y}, \quad (2.33)$$

$$E(Y) = p_i |V(x)|, \quad (2.34)$$

$$Var(Y) = p_i(1 - p_i)|V(x)|, \quad (2.35)$$

where $|V(x)|$ is the feasible solution of the neighbourhood, Y is a binomial random variable representing the cells infected by the virus in the neighbourhood, i.e., $Y = \text{Bin}(|V(x)|, p_i)$.

Each one of the infected cells in the clinical picture has a probability of developing antibodies against the infection based on a Bernoulli probability distribution ($p_{an} : A(x) = \text{Ber}(p_{an})$). So, the total population of infected cells generating antibodies is characterised by a binomial distribution ($p_{an} : A(\text{population}) = \text{Bin}(n, p_{an})$). The probability of finding exactly a immune cells ($P(A = a)$), as well as the average ($E(A)$), and variance ($Var(A)$) are given by Eqs. 2.36–2.38, respectively (Cortés et al. 2008):

$$P(A = a) = \binom{|V(x)|}{a} p_{an}^a (1 - p_{an})^{|V(x)|-a}, \quad (2.36)$$

$$E(A) = p_{an} |V(x)|, \quad (2.37)$$

$$Var(A) = p_{an}(1 - p_{an})|V(x)|, \quad (2.38)$$

where a denotes the immune cell, and x is an infected cell.

In the case of the lysogenic replication, calculate the limit number of interaction (LIT) in a cell x via Eq. 2.39 (Cortés et al. 2008):

$$LIT_{cell-x} = LIT^0 \cdot \left(\frac{f(x) - f(\hat{x})}{f(\hat{x})} \right), \quad (2.39)$$

where LIT^0 is the initial value for LIT.

- Step 5: Ending. The VS algorithm ended according to two criteria, i.e., the collapse and death of the organism, or the isolation of the virus.

2.3.4.2 Performance of VS

To test the proposed algorithm, a well-known NP-Compete problem, i.e., the Steiner problem, is adopted in (Cortés et al. 2008). Compared with TS and GA, VS clearly improves the results from GA and for several cases VS obtains better results than TS.

2.4 Conclusions

In this chapter, we introduced five bacteria inspired CI algorithms. They stem from two background: BFA, BCC algorithm an BCO algorithm are currently bred by the further understanding of bacterial foraging patterns, while SuA and VS algorithm are mainly motivated by the viral research. Although they are newly introduced CI method, we have witnessed the following rapid spreading of at least one of them, i.e., BFA:

First, several enhanced versions of BFA can be found in the literature as outlined below:

- Adaptive BFA (Majhi et al. 2009; Sanyal et al. 2011; Sathya and Kayalvizhi 2011d; Panigrahi and Pandi 2009).
- Amended BFA (Sathya and Kayalvizhi 2011a).
- BFA with varying population (Li et al. 2010).
- Fuzzy adaptive BFA (Venkaiah and Kumar 2011).
- Fuzzy dominance based BFA (Panigrahi et al. 2010).
- Hybrid BFA and differential evolution (Pandi et al. 2010).
- Hybrid BFA and genetic algorithm (Nayak et al. 2012; Kim et al. 2007).
- Hybrid BFA and particle swarm optimization (Hooshmand and Mohkami 2011; Saber 2012).
- Hybrid BFA, differential evolution, and particle swarm optimization (Vaisakh et al. 2012).

- Modified BFA (Verma et al. 2011; Sathya and Kayalvizhi 2011b; Hota et al. 2010; Deshpande et al. 2011; Biswas et al. 2010a, b).
- Multi-colony BFA (Chen et al. 2010).
- Multiobjective BFA (Panigrahi et al. 2011).
- Oppositional based BFA (Mai and Ling 2011).
- Other hybrid BFA (Lee and Lee 2012; Panda et al. 2011; Hooshmand et al. 2012; Rajni and Chana 2013).
- Quantum inspired bacterial swarming optimization (Cao and Gao 2012).
- Rule based BFA (Mishra et al. 2007).
- Self-adaptation BFA (Su et al. 2010).
- Simplified BFA (Muñoz et al. 2010).
- Synergetic bacterial swarming optimization (Chatzis and Koukas 2011).
- Velocity modulated BFA (Gollapudi et al. 2011).

Second, the BFA has also been successfully applied to a variety of optimization problems as listed below:

- Circuit design optimization (Chatterjee et al. 2010).
- Communication optimization (Su et al. 2010; Chen et al. 2010).
- Data mining (Lee and Lee 2012).
- Fuzzy system design optimization (Kamyab and Bahrololoum 2012).
- Image processing (Verma et al. 2011, 2013; Maitra and Chatterjee 2008; Sanyal et al. 2011; Panda et al. 2011; Sathya and Kayalvizhi 2011a, b, c, d).
- Inventory management (Deshpande et al. 2011).
- Manufacturing cell formation (Nouri et al. 2010; Nouri and Hong 2012).
- Motor control optimization (Bhushan and Singh 2011; Sakthivel and Subramanian 2012; Sakthivel et al. 2011).
- Nonlinear system identification (Majhi and Panda 2010).
- Power system optimization (Tang et al. 2006; Ulagammai et al. 2007; Mishra et al. 2007; Panigrahi and Pandi 2009; Pandi et al. 2010; Hota et al. 2010; Panigrahi et al. 2010; Ali and Abd-Elazim 2011; Tabatabaei and Vahidi 2011; Venkaiah and Kumar 2011; Panigrahi et al. 2011; Hooshmand and Mohkami 2011; Abd-Elazim and Ali 2012; Saber 2012; Hooshmand et al. 2012; Kumar and Jayabarathi 2012; Vaisakh et al. 2012).
- Robot control optimization (Turdudiev et al. 2010; Supriyono and Tokhi 2012).
- Scheduling optimization (Nayak et al. 2012; Vivekanandan and Ramyachitra 2012; Rajni and Chana 2013).
- Stock market prediction (Majhi et al. 2009).

Interested readers are referred to them, together with several excellent reviews (e.g., Tang and Wu 2009; Boussaïd et al. 2013; Agrawal et al. 2011; Niu et al. 2010a, b), as a starting point for a further exploration and exploitation of these bacteria inspired algorithms.

References

- Abd-Elazim, S. M., & Ali, E. S. (2012). Coordinated design of PSSs and SVC via bacteria foraging optimization algorithm in a multimachine power system. *Electrical Power and Energy Systems*, 41, 44–53.
- Adler, J. (1975). Chemotaxis in bacteria. *Annual Reviews of Biochemistry*, 44, 341–356.
- Agrawal, V., Sharma, H. & Bansal, J. C. (2011, December 20–22). Bacterial foraging optimization: A survey. In *The International Conference on Soft Computing for Problem Solving (SOCPROS)* (pp. 227–242). Berlin: Springer.
- Ali, E. S., & Abd-Elazim, S. M. (2011). Bacteria foraging optimization algorithm based load frequency controller for interconnected power system. *Electrical Power and Energy Systems*, 33, 633–638.
- Anandaraman, C., Sankar, A. V. M., & Natarajan, R. (2012). A new evolutionary algorithm based on bacterial evolution and its applications for scheduling a flexible manufacturing system. *Jurnal Teknik Industri*, 14, 1–12.
- Berg, H. C., & Brown, D. A. (1972). Chemotaxis in *Escherichia coli* analyzed by three-dimensional tracking. *Nature*, 239, 500–504.
- Bhushan, B., & Singh, M. (2011). Adaptive control of DC motor using bacterial foraging algorithm. *Applied Soft Computing*, 11, 4913–4920.
- Bisen, P. S. & Raghuvanshi, R. (2013). *Emerging epidemics: Management and control*. Hoboken: Wiley, ISBN 978-1-118-39323-9.
- Biswas, A., Das, S., Abraham, A., & Dasgupta, S. (2010a). Analysis of the reproduction operator in an artificial bacterial foraging system. *Applied Mathematics and Computation*, 215, 3343–3355.
- Biswas, A., Das, S., Abraham, A., & Dasgupta, S. (2010b). Stability analysis of the reproduction operator in bacterial foraging optimization. *Theoretical Computer Science*, 411, 2127–2139.
- Boussaïd, I., Lepagnot, J., & Siarry, P. (2013). A survey on optimization metaheuristics. *Information Sciences*, 237, 82–117.
- Bremermann, H. J. (1974). Chemotaxis and optimization. *Journal of Franklin Institute*, 297, 397–404.
- Bremermann, H. J. & Anderson, R. W. (1991). How the brain adjusts synapses-maybe. In: Boyer, R. S. (Ed.) *Automated reasoning: Essays in honor of Woody Bledsoe*. Norwell: Kluwer.
- Cao, J. & Gao, H. (2012). A quantum-inspired bacterial swarming optimization algorithm for discrete optimization problems. In: Y. Tan, Y. Shi, & Z. Ji (Eds.), *ICSI 2012, Part I, LNCS 7331* (pp. 29–36). Berlin: Springer.
- Chatterjee, A., Fakhfakh, M., & Siarry, P. (2010). Design of second-generation current conveyors employing bacterial foraging optimization. *Microelectronics Journal*, 41, 616–626.
- Chatzis, S. P., & Koukas, S. (2011). Numerical optimization using synergetic swarms of foraging bacterial populations. *Expert Systems with Applications*, 38, 15332–15343.
- Chen, H., Zhu, Y., & Hu, K. (2010). Multi-colony bacteria foraging optimization with cell-to-cell communication for RFID network planning. *Applied Soft Computing*, 10, 539–547.
- Cook, N. (Ed.). (2013). *Viruses in food and water: Risks, surveillance and control*. Cambridge: Woodhead Publishing Limited. ISBN 978-0-85709-430-8.
- Cortés, P., García, J. M., Muñuzuri, J., & Onieva, L. (2008). Viral systems: A new bio-inspired optimisation approach. *Computers & Operations Research*, 35, 2840–2860.
- Cortés, P., García, J. M., Muñuzuri, J. & Guadix, J. (2012). Viral system algorithm: foundations and comparison between selective and massive infections. *Transactions of the Institute of Measurement and Control*. doi:[10.1177/0142331211402897](https://doi.org/10.1177/0142331211402897).
- Desai, M. C., & Meanwell, N. A. (Eds.). (2013). *Successful strategies for the discovery of antiviral drugs*. Cambridge: The Royal Society of Chemistry. ISBN 978-1-84973-657-2.
- Deshpande, P., Shukla, D., & Tiwari, M. K. (2011). Fuzzy goal programming for inventory management: A bacterial foraging approach. *European Journal of Operational Research*, 212, 325–336.

- El-Abd, M. (2012). Performance assessment of foraging algorithms vs. evolutionary algorithms. *Information Sciences*, 182, 243–263.
- Giguère, S., Prescott, J. F., & Dowling, P. M. (Eds.). (2013). *Antimicrobial therapy in veterinary medicine*. Chichester: Wiley. ISBN 978-0-470-96302-9.
- Gollapudi, S. V. R. S., Pattnaika, S. S., Bajpai, O. P., Devi, S., & Bakwad, K. M. (2011). Velocity modulated bacterial foraging optimization technique (VMBFO). *Applied Soft Computing*, 11, 154–165.
- Gong, E. Y., et al. (2013). *Antiviral methods and protocols*. New York: Springer. ISBN 978-1-62703-483-8.
- Helden, J. V., Toussaint, A., & Thieffry, D. (Eds.). (2012). *Bacterial molecular networks: Methods and protocols*. New York: Springer. ISBN 978-1-61779-360-8.
- Hooshmand, R. A., & Mohkami, H. (2011). New optimal placement of capacitors and dispersed generators using bacterial foraging oriented by particle swarm optimization algorithm in distribution systems. *Electrical Engineering*, 93, 43–53.
- Hooshmand, R.-A., Parastegari, M., & Morshed, M. J. (2012). Emission, reserve and economic load dispatch problem with non-smooth and non-convex cost functions using the hybrid bacterial foraging–Nelder–Mead algorithm. *Applied Energy*, 89, 443–453.
- Hota, P. K., Barisal, A. K., & Chakrabarti, R. (2010). Economic emission load dispatch through fuzzy based bacterial foraging algorithm. *Electrical Power and Energy Systems*, 32, 794–803.
- Irizarry, R. (2011). Global and dynamic optimization using the artificial chemical process paradigm and fast Monte Carlo methods for the solution of population balance models. In: Dritsas, I. (Ed.) *Stochastic optimization—Seeing the optimal for the uncertain*, (Chapter 16). Rijeka: InTech, ISBN 978-953-307-829-8.
- Ituarte-Villarreal, C. M. & Espiritu, J. F. (2011). Wind turbine placement in a wind farm using a viral based optimization algorithm. In *41st International Conference on Computers & Industrial Engineering* (pp. 672–677). Los Angeles, CA, USA, 23–26 October 2011.
- Kamyab, S., & Bahrololoum, A. (2012). Designing of rule base for a TSK-fuzzy system using bacterial foraging optimization algorithm (BFOA). *Procedia-Social and Behavioral Sciences*, 32, 176–183.
- Kim, D. H., Abraham, A., & Cho, J. H. (2007). A hybrid genetic algorithm and bacterial foraging approach for global optimization. *Information Sciences*, 177, 3918–3937.
- Kumar, K. S., & Jayabarathi, T. (2012). Power system reconfiguration and loss minimization for an distribution systems using bacterial foraging optimization algorithm. *Electrical Power and Energy Systems*, 36, 13–17.
- Lahtinen, S., Salminen, S., Ouwehand, A., & Wright, A. V. (Eds.). (2012). *Lactic acid bacteria: Microbiological and functional aspects*. Boca Raton: Taylor & Francis Group LLC. ISBN 978-1-4398-3678-1.
- Lee, C.-Y. & Lee, Z.-J. (2012). A novel algorithm applied to classify unbalanced data. *Applied Soft Computing*. doi:[10.1016/j.asoc.2012.03.051](https://doi.org/10.1016/j.asoc.2012.03.051).
- Li, Y., & Li, G. (2012). A new mean shift algorithm based on bacterial colony chemotaxis. *International Journal of Fuzzy Systems*, 14, 257–263.
- Li, W.-, Wang, H., & Zou, Z. J. (2005). Function optimization method based on bacterial colony chemotaxis. *Journal of Circuits and Systems*, 10, 58–63.
- Li, G.-Q., Liao, H.-L. & Chen, H.-H. (2009). Improved bacterial colony chemotaxis algorithm and its application in available transfer capability. In *Fifth International Conference on Natural Computation (ICNC)* (pp. 286–291), 14–16 August 2009, Tianjin, China, IEEE.
- Li, M. S., Ji, T. Y., Tang, W. J., Wu, Q. H., & Saunders, J. R. (2010). Bacterial foraging algorithm with varying population. *BioSystems*, 100, 185–197.
- Lindsay, J. A., & Holden, M. T. G. (2006). Understanding the rise of the superbug: Investigation of the evolution and genomic variation of *Staphylococcus aureus*. *Functional & Integrative Genomics*, 6, 186–201.
- Low, D. E., Kellner, J. D., & Wright, G. D. (1999). Superbugs: How they evolve and minimize the cost of resistance. *Current Infectious Disease Reports*, 1, 464–469.

- Lu, Z.-G., Feng, T., & Li, X.-P. (2013). Low-carbon emission/economic power dispatch using the multi-objective bacterial colony chemotaxis optimization algorithm considering carbon capture power plant. *Electrical Power and Energy Systems*, 53, 106–112.
- Maheshwari, D. K. (Ed.). (2012). *Bacteria in agrobiolgy: Plant probiotics*. Berlin: Springer. ISBN 978-3-642-27514-2.
- Mai, X.-F., & Ling, L. (2011). Bacterial foraging optimization algorithm based on opposition-based learning. *Energy Procedia*, 13, 5726–5732.
- Maitra, M., & Chatterjee, A. (2008). A novel technique for multilevel optimal magnetic resonance brain image thresholding using bacterial foraging. *Measurement*, 41, 1124–1134.
- Majhi, B., & Panda, G. (2010). Development of efficient identification scheme for nonlinear dynamic systems using swarm intelligence techniques. *Expert Systems with Applications*, 37, 556–566.
- Majhi, R., Panda, G., Majhi, B., & Sahoo, G. (2009). Efficient prediction of stock market indices using adaptive bacterial foraging optimization (ABFO) and BFO based techniques. *Expert Systems with Applications*, 36, 10097–10104.
- Mishra, S., Tripathy, M., & Nanda, J. (2007). Multi-machine power system stabilizer design by rule based bacteria foraging. *Electric Power Systems Research*, 77, 1595–1607.
- Modrow, S., Falke, D., Truyen, U., & Schätzl, H. (2013). *Molecular virology*. Berlin: Springer. ISBN 978-3-642-20717-4.
- Morrow, W. J. W., Sheikh, N. A., Schmidt, C. S., & Davies, D. H. (Eds.). (2012). *Vaccinology: Principles and practice*. Chichester: Blackwell Publishing Ltd. ISBN 978-1-4051-8574-5.
- Müller, S. D., Marchetto, J., Airaghi, S., & Koumoutsakos, P. (2002). Optimization based on bacterial chemotaxis. *IEEE Transactions on Evolutionary Computation*, 6, 16–29.
- Muñoz, M. A., Halgamuge, S. K., Alfonso, W., & Caicedo, E. F. (2010). Simplifying the bacteria foraging optimization algorithm. In *IEEE World Congress on Computational Intelligence (WCCI)* (pp. 4095–4101), 18–23 July 2010, CCIB, Barcelona, Spain, IEEE.
- Nayak, S. K., Padhy, S. K., & Panigrahi, S. P. (2012). A novel algorithm for dynamic task scheduling. *Future Generation Computer Systems*, 28, 709–717.
- Niu, B., & Wang, H. (2012). Bacterial colony optimization. *Discrete Dynamics in Nature and Society*, 2012, 1–28.
- Niu, B., Fan, Y., Tan, L., Rao, J. & Li, L. (2010a). A review of bacterial foraging optimization Part I: background and development. *Advanced intelligent computing theories and applications, communications in computer and information science* (Vol. 93, Part 26, pp. 535–543). Berlin: Springer.
- Niu, B., Fan, Y., Tan, L., Rao, J. & Li, L. (2010b). A review of bacterial foraging optimization Part II: Applications and challenges. *Advanced intelligent computing theories and applications, communications in computer and information science* (Vol. 93, Part 26, pp. 544–550). Berlin: Springer.
- Nouri, H. & Hong, T. S. (2012). A bacteria foraging algorithm based cell formation considering operation time. *Journal of Manufacturing Systems*, <http://dx.doi.org/10.1016/j.jmsy.2012.03.001>.
- Nouri, H., Tang, S. H., Tuah, B. T. H., & Anuar, M. K. (2010). BASE: A bacteria foraging algorithm for cell formation with sequence data. *Journal of Manufacturing Systems*, 29, 102–110.
- Panda, R., Naik, M. K., & Panigrahi, B. K. (2011). Face recognition using bacterial foraging strategy. *Swarm and Evolutionary Computation*, 1, 138–146.
- Pandi, V. R., Biswas, A., Dasgupta, S., & Panigrahi, B. K. (2010). A hybrid bacterial foraging and differential evolution algorithm for congestion management. *European Transactions on Electrical Power*, 20, 862–871.
- Panigrahi, B. K., & Pandi, V. R. (2009). Congestion management using adaptive bacterial foraging algorithm. *Energy Conversion and Management*, 50, 1202–1209.
- Panigrahi, B. K., Pandi, V. R., Das, S., & Das, S. (2010). Multiobjective fuzzy dominance based bacterial foraging algorithm to solve economic emission dispatch problem. *Energy*, 35, 4761–4770.

- Panigrahi, B. K., Pandi, V. R., Sharma, R., Das, S., & Das, S. (2011). Multiobjective bacteria foraging algorithm for electrical load dispatch problem. *Energy Conversion and Management*, 52, 1334–1342.
- Passino, K. M. (2002). Biomimicry of bacterial foraging for distributed optimization and control. *IEEE Control System Management*, 22, 52–67.
- Paton, R., Gregory, R., Vlachos, C., Palmer, J. W., Suanders, J., & Wu, Q. H. (2004). Evolvable social agents for bacterial systems modelling. *IEEE Transactions on Nanobioscience*, 3, 208–216.
- RAJNI, & CHANA, I. (2013). Bacterial foraging based hyper-heuristic for resource scheduling in grid computing. *Future Generation Computer Systems*, 29, 751–762.
- Saber, A. Y. (2012). Economic dispatch using particle swarm optimization with bacterial foraging effect. *Electrical Power and Energy Systems*, 34, 38–46.
- Sakthivel, V. P., & Subramanian, S. (2012). Bio-inspired optimization algorithms for parameter determination of three-phase induction motor. *COMPEL: The International Journal for Computation and Mathematics in Electrical and Electronic Engineering*, 31, 528–551.
- Sakthivel, V. P., Bhuvaneswari, R., & Subramanian, S. (2011). An accurate and economical approach for induction motor field efficiency estimation using bacterial foraging algorithm. *Measurement*, 44, 674–684.
- Sanyal, N., Chatterjee, A., & Munshi, S. (2011). An adaptive bacterial foraging algorithm for fuzzy entropy based image segmentation. *Expert Systems with Applications*, 38, 15489–15498.
- Sathya, P. D., & Kayalvizhi, R. (2011a). Amended bacterial foraging algorithm for multilevel thresholding of magnetic resonance brain images. *Measurement*, 44, 1828–1848.
- Sathya, P. D., & Kayalvizhi, R. (2011b). Modified bacterial foraging algorithm based multilevel thresholding for image segmentation. *Engineering Applications of Artificial Intelligence*, 24, 595–615.
- Sathya, P. D., & Kayalvizhi, R. (2011c). Optimal multilevel thresholding using bacterial foraging algorithm. *Expert Systems with Applications*, 38, 15549–15564.
- Sathya, P. D., & Kayalvizhi, R. (2011d). Optimal segmentation of brain MRI based on adaptive bacterial foraging algorithm. *Neurocomputing*, 74, 2299–2313.
- Su, T.-J., Cheng, J.-C., & Yu, C.-J. (2010). An adaptive channel equalizer using self-adaptation bacterial foraging optimization. *Optics Communications*, 283, 3911–3916.
- Sun, J.-Z., Geng, G.-H., Wang, S.-Y., & Zhou, M.-Q. (2012). Chaotic hybrid bacterial colony chemotaxis algorithm based on tent map. *Journal of Software*, 7, 1030–1037.
- Supriyono, H. & Tokhi, M. O. (2012). Parametric modelling approach using bacterial foraging algorithms for modelling of flexible manipulator systems. *Engineering Applications of Artificial Intelligence*, <http://dx.doi.org/10.1016/j.engappai.2012.03.004>.
- Tabatabaei, S. M., & Vahidi, B. (2011). Bacterial foraging solution based fuzzy logic decision for optimal capacitor allocation in radial distribution system. *Electric Power Systems Research*, 81, 1045–1050.
- Tang, W. J., & Wu, Q. H. (2009). Biologically inspired optimization: A review. *Transactions of the Institute of Measurement and Control*, 31, 495–515.
- Tang, W. J., Li, M. S., He, S., Wu, Q. H. & Saunders, J. R. (2006). Optimal power flow with dynamic loads using bacterial foraging algorithm. In *International Conference on Power System Technology (PowerCon)* (pp. 1–5), 2006. IEEE.
- Turduiev, M., Kirtay, M., Sousa, P., Gazi, V. & Marques, L. Chemical concentration map building through bacterial foraging optimization based search algorithm by mobile robots. In *IEEE International Conference on Systems, Man, and Cybernetics (IEEE SMC)* (pp. 3242–3249), 10–13 October 2010, Istanbul, Turkey, IEEE.
- Ulagammai, M., Venkatesh, P., Kannan, P. S., & Padhy, N. P. (2007). Application of bacterial foraging technique trained artificial and wavelet neural networks in load forecasting. *Neurocomputing*, 70, 2659–2667.

- Vaisakh, K., Praveena, P., Rao, S. R. M., & Meah, K. (2012). Solving dynamic economic dispatch problem with security constraints using bacterial foraging PSO-DE algorithm. *Electrical Power and Energy Systems*, 39, 56–67.
- Venkaiah, C., & Kumar, D. M. V. (2011). Fuzzy adaptive bacterial foraging congestion management using sensitivity based optimal active power re-scheduling of generators. *Applied Soft Computing*, 11, 4921–4930.
- Verma, O. P., Hanmandlu, M., Kumar, P., Chhabra, S., & Jindal, A. (2011). A novel bacterial foraging technique for edge detection. *Pattern Recognition Letters*, 32, 1187–1196.
- Verma, O. P., Hanmandlu, M., Sultania, A. K. & Parihar, A. S. (2013). A novel fuzzy system for edge detection in noisy image using bacterial foraging. *Multidimensional Systems and Signal Processing*. doi: [10.1007/s11045-011-0164-1](https://doi.org/10.1007/s11045-011-0164-1).
- Vivekanandan, K., & Ramyachitra, D. (2012). Bacteria foraging optimization for protein sequence analysis on the grid. *Future Generation Computer Systems*, 28, 647–656.

Innovative Computational Intelligence: A Rough Guide
to 134 Clever Algorithms

Xing, B.; Gao, W.-J.

2014, XXXIX, 451 p., Hardcover

ISBN: 978-3-319-03403-4