International Journal of Applied Information Systems (IJAIS) – ISSN : 2249-0868
Foundation of Computer Science FCS, New York, USA
Volume 2– No.4, May 2012 – www.ijais.org

Integration of Bacteria Foraging Optimization and Case Base Reasoning for Ground Water Possibility Detection

Chandni Kapoor
M.Tech Student (CSE)
D.A.V.I.E.T, Jalandhar

Harpreet Bajaj
Assistant Professor
D.A.V.I.E.T, Jalandhar

Navdeep Kaur
Assistant professor
K.C.E.T, Amritsar

ABSTRACT
Bacterial Foraging Optimization (BFO) is a population-based numerical optimization algorithm. This technique is proposed by K.M. Passino in 2002 to handle complex problems of the real world. Case-based reasoning is a unique platform of concepts and techniques that touch upon some of the basic issues concerning to knowledge representation, reasoning and learning from experience. In this work, we have integrated Bacteria Foraging Optimization with Case Based Reasoning to detect ground water possibility in a given area. An algorithm has been proposed in this work. A problem case whose ground water possibility is to be determined is input to the system, BFO retrieves the best matching case from the Case Base and the ground water possibility of that Case is proposed as a solution to the problem case.

Keywords
Bacterial Foraging Optimization, Chemotactic, Case base reasoning, Case Retrieval, Ground water possibility.

1. INTRODUCTION
Bacterial Foraging Optimization (BFO) is a novel optimization algorithm based on the social foraging behavior of E. coli bacteria. The motile bacteria such as E. coli and Salmonella propel themselves by rotating their flagella. To move forward, the flagella counterclockwise rotate and the organism "swims" (or "runs").While a clockwise rotation of the flagellum causes the bacterium randomly "tumble" itself in a new direction and then swims again [1]. An alternation between "swim" and "tumble" enables the bacterium search for nutrients in random directions. Swimming is more frequent as the bacterium approaches a nutrient gradient. Tumbling, hence direction changes, is more frequent as the bacterium moves away from some food to search for more. Basically, bacterial chemotaxis is a complex combination of swimming and tumbling that keeps bacteria in places of higher concentration of nutrients. Bacterial chemotaxis can also be considered as the optimization process of the exploitation of known resources, and costly exploration for new, potentially more valuable resources [2].

2. CLASSICAL BFO ALGORITHM
The original Bacterial Foraging Optimization system consists of three principal mechanisms, namely, chemotaxis, reproduction, and elimination-dispersal.Chemotactic consists of Swim and Tumble. Each of these processes is described as follows.

2.1 Chemo taxis
In the original BFO, a unit walk with random direction represents a "tumble" and a unit walk with the same direction in the last step indicates a "run." Suppose $\theta^i(j,k,l)$ represents the bacterium at jth chemotactic, kth reproductive, and lth elimination-dispersal step. C(i) is the chemotactic step size during each run or tumble (i.e., run-length unit). Then in each computational chemotactic step, the movement of the ith bacterium can be represented as

$$\theta^i(j+1,k,l) = \theta^i(j,k,l) + C(i)\frac{\Delta(i)}{\sqrt{\Delta^T(i)\Delta(i)}}, \quad \ldots(1)$$

where Δ(i) is the direction vector of the jth chemotactic step. When the bacterial movement is run, Δ(i) is the same with the last chemotactic step; otherwise, Δ(i) is a random vector whose elements lie in [−1, 1]. With the activity of run or tumble taken at each step of the chemotaxis process, a step fitness, denoted as J (i, j, k, l), will be evaluated [8].

2.2 Reproduction
The health status of each bacterium is calculated as the sum of the step fitness during its life, that is, $\sum_{j=1}^{Nc} J(i,j,k,l)$, where Nc is the maximum step in a chemotaxis process. All bacteria are sorted in reverse order according to health status. In the reproduction step, only the first half of population survives and a surviving bacterium splits into two identical ones, which are then placed in the same locations. Thus, the population of bacteria keeps constant [7].

2.3 Elimination and Dispersal
The chemotaxis provides a basis for local search, and the reproduction process speeds up the convergence which has been simulated by the classical BFO. While to a large extent, only chemotaxis and reproduction are not enough for global optima searching. Since bacteria may get stuck around the initial positions or local optima, it is possible for the diversity of BFO to change either gradually or suddenly to eliminate the accidents of being trapped into the local optima. In BFO, the dispersion event happens after a certain number of reproduction processes. Then some bacteria are chosen, according to a preset probability Ped, to be killed and moved to another position within the environment [9], [8].

The original BFO algorithm is briefly outlined step by step as follows.

Step 1. Initialize parameters n, S, Nc, Ns, Nre, Ned, Ped, C(i) (i = 1, 2, . . . , S), θi, where
n = dimension of the search space,
S = the number of bacteria in the colony,
Nc = chemotactic steps,
Ns = swim steps,
Nre = reproductive steps,
Ned = elimination and dispersal steps,
Ped = probability of elimination,
C(i)= the run-length unit (i.e., the size of the step taken in each run or tumble).

Step 2. Elimination-dispersal loop: l = l +1.

Step 3. Reproduction loop: k = k +1.

Step 4. Chemotaxis loop: j = j +1.

Substep 4.1 For i = 1, 2. . . S, take a chemotactic step for bacterium i as follows:

Substep 4.2 Compute fitness function, J (i, j, k, l).

Substep 4.3 Let J last = J (i, j, k, l) to save this value since we may find better value via a run.

Substep 4.4 Tumble. Generate a random vector $\Delta(i) \in Rn \setminus$ with each element Δm (i), m = 1, 2. . . n, a random number on [−1, 1].

Substep 4.5 Move. Let

$$\theta^i(j+1,k,l) = \theta^i(j,k,l) + C(i)\frac{\Delta(i)}{\sqrt{\Delta^T(i)\Delta(i)}} \quad \ldots (2)$$

This results in a step of size C(i) in the direction of the tumble for bacterium i.

Substep 4.6 Compute J (i, j +1, k, l) with θi (j+1, k, l).

Substep 4.7 Swimming.
(i) Let m = 0 (counter for swim length).
(ii) While m < Ns (if has not climbed down too long), the following hold.
- Let m = m + 1.
  - If J(i, j +1, k, l) < Jlast, let Jlast = J(i, j +1,k,l) then another step of size C(i) in this same direction will be taken as eq. (2) and use the new generated. θi(j +1, k, l) to compute the new J (i, j +1, k, l).
  - Else let m = Ns.

Substep 4.8 Go to next bacterium (i +1). If i≠S, go to (Substep 4.2) to process the next bacterium.

Step 5. If j < Nc , go to Step (3). In this case, continue chemotaxis since the life of the bacteria is not over.

Step 6. Reproduction.

Substep 6.1 For the given k and l, and for each i = 1, 2, . . . , S, let

$$J^i_{health} = \sum_{j=1}^{Nc+1} J(i,j,k,l) \quad \ldots (3)$$

be the health of the bacteria. Sort bacteria in order of ascending values of Jhealth.

Substep 6.2 The Sr bacteria with the highest Jhealth value die and the other Sr bacteria with the best values split and the copies that are made are placed at the same location as their parent. .

Step 7. If k < Nre, go to Step 2. In this case the number of specified reproduction steps is not reached and start the next generation in the chemotactic loop.

Step 8. Elimination-dispersal: for i = 1, 2, . . . , S, with probability ped, eliminate and disperse each bacterium, which results in keeping the number of bacteria in the population constant. To do this, if a bacterium is eliminated, simply disperse one to a random location on the optimization domain. If l <Ned, then go to Step 2. otherwise end.

3. CASE BASE REASONING

CBR involves reasoning from prior examples: retaining a memory of previous problems and their solutions and solving new problems by reference to that knowledge. Generally, a case-based reasoned will be presented with a problem, either by a user or by a program or system. The case-based reasoned then searches its memory of past cases (called the case base) and attempts to find a case that has the same problem specification as the case under analysis. If the reasoner cannot find an identical case in its case base, it will attempt to find a case or multiple cases that most closely match the current case under analysis [3].

In a CBR system, the problem solving life cycle consists of the following four parts .

(a) **Retrieve:** Determine most similar previously experienced cases (e.g. problem-solution-outcome triples) stored in the case base, whose problem is examined to be similar.

(b) **Reuse:** Solve the new problem by re-using information and knowledge in the retrieved cases by copying or integrating the solutions.

(c) **Revise:** Evaluate the applicability of the proposed solution in the real-world or adapting the solutions retrieved in an attempt to solve the new problem.

(d) **Retaining:** Update case base with new learned case for future problem solving [4].

3.1 Case Retrieval Method

The case searching and matching is a key step in case retrieving and it directly influences the re trieval efficiency and accuracy. In fact, the case retrieval in essence is to find the most similar case in the case base to target case.

K-Nearest Neighbor (KNN) is widely used for its advantages of clear physical concept and simple calculation for case searching [3]. A sum of similarities is calculated according to the similarity between each feature in problem case and the cases in case base.

$$Sim(P, C) = \sum f(Pa,Ca) * wa$$

where

$$\begin{cases} f(P_a, C_a) = 1, \text{ if } P_a = C_a \\ f(P_a, C_a) = 0, Otherwise \end{cases}$$

Where, Sim (P,C) represents the similarity degree of problem case(P) and case(C) stored in case base, a= attribute of case, n = number of attributes, Pa = problem case, Ca=cases stored in case base and wa = weight of attribute a, the more important attributes should assigned larger weights then less important ones. The larger the weighted sum is, the more similar the two cases will be.

4. PROPOSED WORK

In our research, we have integrated Bacteria Foraging Optimization with Case Based Reasoning to detect the ground water possibility in a given area. The Bacteria Foraging optimization has been used for Case Retrieval from the given CBR system. A problem case is entered to the system whose ground water possibility is to be detected in terms of Low, High, and Moderate. Bacteria Foraging Optimization algorithm retrieves the most similar case from Case Base and decision of that most similar case (retrieved from case base) is proposed as a solution to problem case.

4.1 Integration of BFO with CBR

In a CBR system, the key factors that influence the retrieving of similar cases from case base include knowledge representation; attribute description and similarity measures definition [3]. In this CBR system, geographical parameters and their corresponding solutions, i.e. the possibility of ground water (High, Moderate, and Low) are stored as cases in the case base (knowledge representation). The cases are collected from the expert knowledge. The case is represented by the ordered pair C= {F, D}, where F is the geographical parameter of case C and F = {Geology, Landform, Soil Type, Lineament, Slope, Land Type}, D is the decision attribute, i.e. the possibility of ground water in the given region (attribute description).

In the proposed System, the value of geographical parameters (F) of Case(C) is considered to be the position of bacteria in multidimensional search space and all the bacteria are categorized in D categories depending upon their current position(F's values). The problem case whose solution (D's value) is to be found is considered to be food that bacteria is searching for. Thus, in proposed work, there is only one food source and each bacteria is trying to approach it. At the end of the algorithm, the food will belong to bacteria (with category D) which is closest to food.

5. PROPOSED ALGORITHM

The proposed algorithm for integration BFO with CBR is discussed in Fig 1. The parameters used in algorithm are as follows:

n = dimensions of search space (number of attributes of a case in Case Base)
S = no of bacteria in colony
Nc = Chemotactic steps
Nre = reproductive steps
Sr = fraction of bacteria to be eliminated in a reproduction step

Initially, S bacteria are assigned random positions (S cases from Case Base). Then during chemotaxis and reproduction (step 2 to 5) steps of proposed algorithm, Nre*Nc*S cases are taken from the case base. Thus,

S + Nre*NC*S = total size of Case Base

This formula ensures that each case from case base is considered exactly once.

5.1 Chemotactic

In each chemotactic step, bacterium tumbles to a new position, $\theta^i(j+1,k)$

$\theta^i(j+1,k)$ = Case from Case Base that has not yet been considered

Where, $\theta^i(j+1,k)$ is the position of ith bacterium at jth chemotactic and kth reproductive step and bacterium moves to this new position. Any bacterium can only tumble to predefined positions (total cases in case base). Among all these positions, bacterium can tumble to any random position that has not been considered.

5.2. Fitness Function

The fitness function J (i, j, k) (in step 3.2 of proposed algorithm) of i^{th} bacterium at j^{th} chemotactic and k^{th} reproductive step is given by:

$$J(i,j,k) = Sim(F,\theta^i(j,k)) = \sum_{a=1}^{n} f(F_a,\theta_a^i(j,k)) * w_a$$

$$\text{Where } \begin{cases} f(F_a,\theta_a^i(j,k)) = 1, if\ F_a = \theta_a^i(j,k) \\ f(F_a,\theta_a^i(j,k)) = 0, otherwise \end{cases}$$

Where F is the position of food and θ^i (j, k) is the position of current bacteria. If the position of bacteria and food along ath dimension is same, then similarity value ($f(F_a,\theta_a^i(j,k))$) is taken to be 1,otherwise is taken to be zero. Each dimension is assigned different weightage (wa) because each dimension (attribute of a case) has different importance in bringing the bacteria near to food.

Total weight used in step 6 of proposed algorithm is given by:

$$\text{total weight} = \sum_{a=1}^{n} w_a$$

Once the current fitness J (i, j+1, k) has been found (step 3.5) for ith bacteria, this is compared with its previous fitness, J (i, j, k). If the current fitness (J (i, j+1, k)) of i th bacteria is less than its previous fitness (J (i, j, k)), then the bacteria has moved farther from food in this chemotaxis step. So the bacteria is moved back to its previous position ($\theta^i(j,k)$) with current fitness updated back to previous fitness (J (i, j, k)) (step 3.6).

5.3 Reproduction

In the proposed work, the health status of each bacterium is its current fitness value. All the bacteria are sorted in descending order of their health status (as in step 4). In the reproduction step, first half (Sr=0.5 in our research) of the fittest bacteria survive and the other half (least fit bacteria) are removed. Each of fitter (surviving) bacteria is split into two identical ones and are placed in same location as their parent.

Algorithm CBR Using BFO

1. Place the bacteria at random position and the food at its defined position in search space.
2. (Reproduction step k+1) For k = k + 1
3. (Chemotaxis step j+1) For j = j + 1
   3.1 For i = 1,2,…S, take chemotaxis step for bacteria i as follows:
   3.2 Compute the fitness function of the Previous (jth) Chemotaxis step:

   J (i ,j, k) =Sim (F, θ^i (j, k))

   where F is the position of food and θ^i (j, k) is the position of ith bacteria
   3.3 (Tumble) Tumble to random new position,

   $\theta^i(j+1,k)$.

   3.4 (Move) Move the bacteria to new position $\theta^i(j+1,k)$
   3.5 Compute J (i ,j+1,k) using $\theta^i(j+1,k)$ (as in step 3.2).
   3.6 If J (i, j+1, k) < J (i, j, k) then:
   (i) (Move Back) Move bacteria back to its previous position:

   $\theta^i(j+1,k) = \theta^i(j,k)$

   (ii) Upadate Fitness function
   J (i, j+1, k) = J (i ,j, k)
   end if
   3.7 Go to next bacteria i +1 (for of step 3.1 ends)
   3.8 Store the current fitness of ith bacteria in Jc (i). (chemotaxis loop of step 3 ends).
4. (Reproduction step) for given k and for each i = 1,2..S, Let

   $J^i_{health} = J_c(i)$

   be the health of bacteria. Sort the bacteria in descending order of J_{health}.
5. The bacteria with Sr lowest J_{health} values die and other bacteria with Sr best J_{health} are split and copies that are made are placed at the same location as their parents. (reproduction loop of step 2 ends)
6. Pick up the bacteria B with max (J_{health}) value. The Probability that the food F belongs to bacteria B (with category D) is:

   $= \frac{\max\ (J_{health})}{\text{total weight}} * 100$

Fig 1:- Proposed algorithm for integration of BFO with CBR

Thus half of the currently considered cases that are less similar to problem case are removed and other halves that are more similar are duplicated. This will help to find the most similar cases (from Case Base) to problem case.

5.4 Elimination Dispersal

Since there are fixed number of positions (cases in Case Base) that bacteria can tumble to and each position will be considered exactly once in chemotaxis step, bacteria can't get stuck in local optima . So elimination dispersal step is not required in the proposed algorithm.

6. RESULTS AND DISCUSSION

The algorithm for groundwater possibility detection using Bacteria Foraging Optimization as shown in Fig. 1 is coded in Matlab 7.0 [10].

6.1 Case Base

The case base used in our experiments as in Table 1 consists of six geological features (geology, land form, soil type, land use, lineament, slope) and depending upon the set of values of these features in a given area, the ground water possibility is recorded in fuzzy terms of high, moderate and low. This Case Base has been designed by domain experts by on the basis of their ground observations. In Table 1, first column gives the geological attributes used in case base and the second column indicates their corresponding possible values [6].

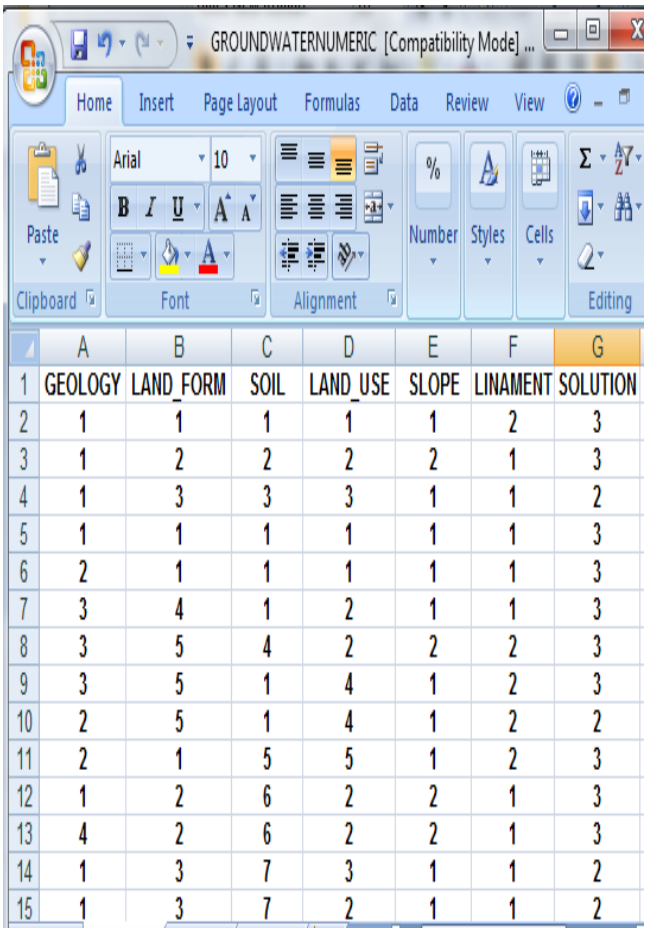
Table1. Six attributes used to design Case Base of ground water possibility detection

Attributes	Values
Geology	Sedimentary, Younger alluvium, Older alluvium, Igneous, Metamorphic
Landform	Floodplain, Intermontanevalley, Pediment, Alluvialfans ,Bajada, Pediplain, Buriedpediment, AlluvialPlain,DeltaicPlain, Wadi, Riverterraces, Oldmeander etc.
Soil	Sandyloam, Sandygravel, Coarsesand, Clayloam, Alluvialsand, Gravelsand, Gravel Sand Pebbles, Sand, Rocky etc.
Land use	Agriculturalland, Forest, Cultivatedland, Fallowland, Waterbody, Wasteland, Swampy land, Buildup, Urban ,Grass, Shrubs ,mixed vegetation etc.
Slope	Gentle, Steep
Lineament	Absent, Present

The possible values of each geological attribute (F) is assigned numerical values for simplicity and stored in first six columns and the numerical value of decision (D) of ground water possibility is stored in seventh columns of row. First column of table 1 gives the categorical attributes that are used in the dataset and the second column indicates the attribute values.
One row represents one case in Case Base.

Each case in the case base will be stored as shown in Table 2

Table 2: Cases stored in Case Base



	A	B	C	D	E	F	G
1	GEOLOGY	LAND_FORM	SOIL	LAND_USE	SLOPE	LINAMENT	SOLUTION
2	1	1	1	1	1	2	3
3	1	2	2	2	2	1	3
4	1	3	3	3	1	1	2
5	1	1	1	1	1	1	3
6	2	1	1	1	1	1	3
7	3	4	1	2	1	1	3
8	3	5	4	2	2	2	3
9	3	5	1	4	1	2	3
10	2	5	1	4	1	2	2
11	2	1	5	5	1	2	3
12	1	2	6	2	2	1	3
13	4	2	6	2	2	1	3
14	1	3	7	3	1	1	2
15	1	3	7	2	1	1	2

6.2 User Interface

The development of Graphical User Interface for ground water possibility retrieval system is accomplished using Matlab 7.0 (as in Fig. 2). It is similar to the interface used in [5][6].

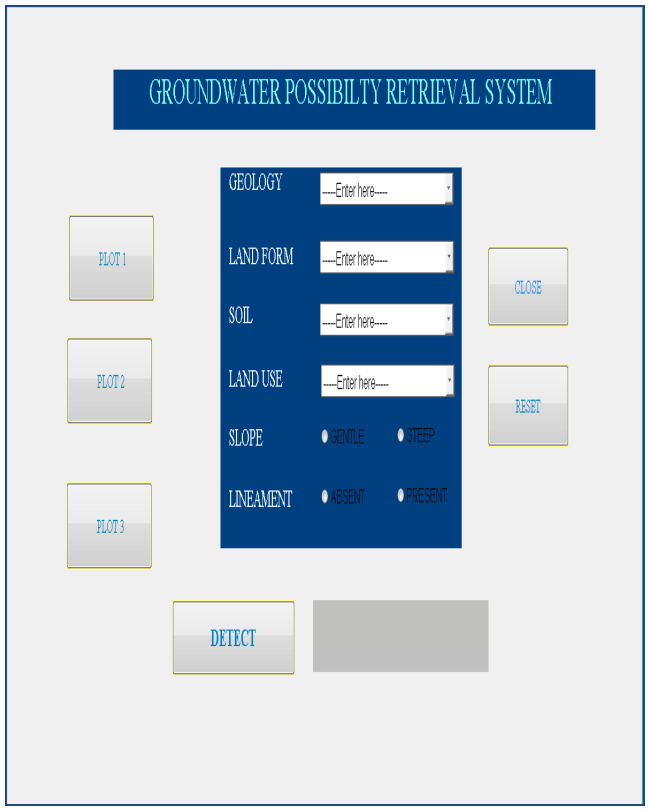


Fig 2:- GUI of Ground Water Possibility Detection

Table 3: Example of Query Entered by User

Geology	Metamorphic
LandForm	Floodplain
Soil	SandyLoam
Land Use	Cultivated land
Slope	Gentle
Lineament	absent
Decision	To be find

User is required to enter values for all the six attributes to predict ground water possibility in a given area. The example of query entered by user is shown in Table 3. This query entered by user becomes the position of food and all the cases in case base become the possible positions of bacteria in proposed Bacteria Foraging algorithm. The most similar case from case base is retrieved using proposed algorithm and decision (of ground water possibility) of that case is assigned as a decision to the query case.

The result of query in Table 3 is:
"**GROUND WATER POSSIBILITY IS HIGH WITH 75%**"
The percentage value denotes the amount of similarity between problem case and the solution case.

7. CONCLUSION AND FUTURE SCOPE

In this paper, we have presented Bacteria Foraging Optimization and Case Base Reasoning as an efficient technique to detect ground water possibility in a given area. Bacteria Foraging Optimization has been used as case retrieval method to retrieve the most similar case from Case Base. The decision of most similar case is provided as the decision to query case. The results obtained in this work are comparable to the work already done in this field. A system has been developed which can determine ground water level at any place without drilling holes by knowing its geological features only. The system developed is of great significance as the presence or absence of ground water in a given area has a direct effect on its real estate values. So the system can play a crucial role in economic sector of a nation. Moreover, it is very useful in military applications during the time of combats to find groundwater possibility in inaccessible areas like areas across the border of a nation. Movement of troops depends largely on such information as water is a basic need for survival.
Future research includes exploring Bacteria Foraging Optimization with different kind of Case Base Reasoning systems. Other possible applications may be processing medical data where integration of Bacteria Foraging and CBR system can be used to diagnose the disease of patients. Other areas such as oceanographic astronomical observations can be attacked. Further, a method may be proposed by using Bacteria Foraging to revise the retrieved solution and retain new solution into CBR system to solve new problems.

8. REFERENCES

[1] Adler, J. 1996. Chemotaxis in bacteria, Science, vol 153, pp. 708–716.

[2] Chen, H. , Zhu, Y. and Hu, K.. 2009 Cooperative Bacterial Foraging Optimization, Discrete Dynamics in Nature and Society,vol. 2, no.1, pp.501-517.

[3] Pal, S.K. and Shiu, Simon C. K. , 2009 Foundation of Soft case based reasoning, Wiley Series on Intelligent systems. Hoboken, New Jersey,.

[4] Agnar Aamodt and Enric Plaza, Foundational Issues, Methodological Variations, and System Approaches, Artificial Intelligence Communications, IOS Press, vol. 7, no. 1,pp. 39-59.

[5] Chunhua Yang, Hongqiu Zhu and Weihua Gui, , 2008 Permeability prediction model for imperial smelting furnace based on improved case-based reasoning, IEEE Proceedings of the 7th world congress on intelligent control and automation, June 25-27, Chongqing, China.

[6] Panchal, V.K. , Kundra, H. and Kaur,A. , 2009 An integrated approach to Biogeography Based Optimization with case based reasoning for retrieving Groundwater possibility In Proceedings of 8th Annual Asian Conference and Exhibition on Geospatial Information, Technology and Applications, Singapore.

[7] Swagatam Das, Arijit Biswas, Sambarta Dasgupta, and Ajith Abraham, 2009. Bacterial Foraging Optimization Algorithm: Theoretical Foundations, Analysis, and Applications, IEEE.

[8] Swagatam Das, Sambarta Dasgupta, Arijit Biswas, and Ajith Abraham, 2009. On Stability of the Chemotactic Dynamics in Bacterial-Foraging Optimization algorithm, IEEE Transaction on systems, mans, and cybernetics, VOL. 39, NO. 3

[9] Kevin M.Passino, 2010 Bacteria Foraging Optimization, Internantional journal of warm intelligence research.

[10] The MATLAB ver 7, The MathWorks, Inc.