ISSN (Online): 2230-7893
www.IJCEM.org

Soft Computing, Artificial Intelligence, Fuzzy Logic & Genetic Algorithm in Bioinformatics

Swati Jain, Abhishek Pandey
Faculty of CS, Takshshila Institute of Engineering & Technology, Jabalpur

Abstract

Soft computing is creating several possibilities in bioinformatics, especially by generating low-cost, low precision (approximate), good solutions. Bioinformatics is an interdisciplinary research area that is the interface between the biological and computational sciences. Bioinformatics deals with algorithms, databases and information systems, web technologies, artificial intelligence and soft computing, information and computation theory, structural biology, software engineering, data mining, image processing, modeling and simulation, discrete mathematics, control and system theory, circuit theory, and statistics. Despite of a high number of techniques specifically dedicated to bioinformatics problems as well as many successful applications, we are in the beginning of a process to massively integrate the aspects and experiences in the different core subjects such as biology, medicine, computer science, engineering, and mathematics. Recently the use of soft computing tools for solving bioinformatics problems have been gaining the attention of researchers because of their ability to handle imprecision, uncertainty in large and complex search spaces. The paper will focus on soft computing paradigm in bioinformatics with particular emphasis on integrative research.

Keywords: *Bioinformatics, Soft computing, Artificial neural network, Fuzzy logic, Genetic algorithms*

1. Introduction

Advancement in soft computing techniques demonstrates the high standards of technology, algorithms, and tools in bioinformatics for dedicated purposes such as reliable and parallel genome sequencing, fast sequence comparison, search in databases, automated gene identification, efficient modeling and storage of heterogeneous data, etc. The basic problems in bioinformatics like protein structure prediction, multiple alignment, phylogenetic inference etc. are mostly NP-hard in nature. For all these problems, soft computing offers on promising approach to achieve efficient and reliable heuristic solution. On the other side the continuous development of high quality biotechnology, e.g. micro-array techniques and mass spectrometry, which provide complex patterns for the direct characterization of cell processes, offers further promising opportunities for advanced research in bioinformatics. So bioinformatics must cross the border towards a massive integration of the aspects and experience in the different core subjects like computer science and statistics etc. for an integrated understanding of relevant processes in systems biology. This puts new challenges not only on appropriate data storage, visualization, and retrieval of heterogeneous information, but also on soft computing methods and tools used in this context, which must adequately process and integrate heterogeneous information into a global picture.

2. Bioinformatics

Bioinformatics is the application of computer technology to the management and analysis of biological data. The result is that computers are being used to gather, store, analyze and merge biological data. The ultimate goal of bioinformatics is to uncover the wealth of biological information hidden in the mass of data and obtain a clearer insight into the fundamental biology of organisms. This new knowledge could have profound impacts on fields as varied as human health, agriculture, the environment, energy and biotechnology.[48]

2.1. Tasks of Bioinformatics

Different biological problems considered within the scope of bioinformatics involve the study of genes, proteins, nucleic acid structure prediction, and molecular design with docking. A broad classification of the various bioinformatics tasks is given as follows.

1. Alignment and comparison of DNA, RNA, and protein sequences.
2. Gene mapping on chromosomes.
3. Gene finding and promoter identification from DNA sequences.
4. Interpretation of gene expression and micro-array data.
5. Gene regulatory network identification.
6. Construction of phylogenetic trees for studying evolutionary relationship.
7. DNA structure prediction.
8. RNA structure prediction.
9. Protein structure prediction and classification.
10. Molecular design and molecular docking.

2.2. Applications of Bioinformatics

Bioinformatics has found its applications in many areas. It helps in providing practical tools to explore proteins and DNA in number of other ways. Bio-computing is useful in recognition techniques to detect similarity between

sequences and hence to interrelate structures and functions. Another important application of bioinformatics is the direct prediction of protein 3-Dimensional structure from the linear amino acid sequence. It also simplifies the problem of understanding complex genomes by analyzing simple organisms and then applying the same principles to more complicated ones. This would result in identifying potential drug targets by checking homologies of essential microbial proteins. Bioinformatics is useful in designing drugs.

The aims of Bioinformatics are:
1. To organize data in a way that allows researchers to access existing information and to submit new entries as they are produced
2. To develop tools and resources that aid in the analysis and management of data.
3. To use this data to analyze and interpret the results in a biologically meaningful manner.
4. To help researchers in the pharmaceutical industry in understanding the protein structures to make the drug design easy.

2.3. Algorithms in Bioinformatics

This discussion sheds light on algorithms that are of interest to biologists. The following are some of the most important algorithmic trends in bioinformatics:

1. Finding similarities among strings (such as proteins of different organisms).
2. Detecting certain patterns within strings (such as genes, introns, and a-helices).
3. Finding similarities among parts of spatial structures (such as motifs).
4. Constructing trees (called phylogenetic trees expressing the evolution of organisms whose DNA or proteins are currently known.
5. Classifying new data according to previously clustered sets of annotated data.
6. Reasoning about microarray data and the corresponding behavior of pathways.

The first three trends can be viewed as instances of pattern matching. However, pattern matching in biology differs from its counterpart in computer science. DNA strings contain millions of symbols, and small local differences may be tolerated. The pattern itself may not be exactly known, because it may involve inserted, deleted, or replacement symbols. Regular expressions are useful for specifying a multitude of patterns and are ubiquitous in bioinformatics. However, what biologists really need is to be able to infer these regular expressions from typical sequences and establish the likelihood of the patterns being detected in new sequences.

Two other algorithmic trends relevant to this discussion are related to micro-arrays and biologists' interest in computational linguistics. Recall that the main goal of analyzing micro-array data is to establish relationships among gene behavior, possible protein interactions, and the effects of a cell's environment. From a computer science perspective, that goal is amounted to the generation of parts of a program (flowchart) from data. This was also an early goal of program synthesis. However, it should be stressed that biological data is vast and noisy, spurring development of new heuristic based techniques (such as Bayesian nets, SVM, Fuzzy logic and evolutionary algorithms) is required.

Information about the relationships among genes is often buried in countless articles describing the results of biological experiments. In the case of protein interaction, pharmaceutical companies have teams whose task is to search the available literature and "manually" detect phrases of interest. Efforts have been made to computerize these searches. Their implementation requires expertise in biology, computational linguistics and heuristic based methodologies.

Based on the above discussion, it is mandatory to have a machine learning/soft computing based approach for various tasks in bioinformatics. This paper will focus on, how the soft computing techniques suits in bioinformatics. The next section will discuss the concept of soft computing, their constituents and the prominent application to bioinformatics.

3. Soft Computing

The principal notion in soft computing is that precision and certainty carry a cost, and that computation, reasoning, and decision-making should exploit the tolerance for imprecision, uncertainty, approximate reasoning, and partial truth for obtaining low-cost solutions. Soft computing is a consortium of methodologies that work synergistically and provides, in one form or another, flexible information processing capabilities for handling real life ambiguous situations. The constituents of soft computing are: Fuzzy Logic (FZ), Artificial Neural Networks (ANN), Evolutionary Algorithms (EAs) (including genetic algorithms (GAs), genetic programming (GP), evolutionary strategies (ES)), Support Vector Machines (SVM), Simulated Annealing (SA), Ant Colony Optimization (ACO) and Tabu Search (TS). In this paper, the application of the main constituent of the soft computing methods like fuzzy set, artificial neural network and genetic algorithm in bioinformatics have been briefly discussed.

3.1. Why Soft Computing Techniques in Bioinformatics

There are a number of reasons why soft computing approaches are widely used in practice, especially in bioinformatics [1] [9][17].

1. Traditionally, a human being builds such an expert system by collecting knowledge from specific experts. The experts can always explain what factors they use to assess a situation, however, it is often difficult for the experts to say what rules they use (for example, for disease analysis and control). This problem can be resolved by soft computing mechanisms. Soft computing mechanism can extract the description of the hidden situation in terms of those factors and then fire rules that match the expert's behavior.
2. Systems often produce results different from the desired ones. This may be caused by unknown properties or functions of inputs during the design of systems. This situation always occurs in the biological world because of the complexities and mysteries of life sciences. However, with its capability of dynamic improvement, soft computing can cope with this problem.
3. In molecular biology research, new data and concepts are generated every day, and those new data and concepts update or replace the old ones. Soft computing can be easily adapted to a changing environment. This benefits system designers, as they do not need to redesign systems whenever the environment changes.
4. Missing and noisy data is one characteristic of biological data. The conventional computer techniques fail to handle this. Soft computing based techniques are able to deal with missing and noisy data.
5. With advances in biotechnology, huge volumes of biological data are generated. In addition, it is possible that important hidden relationships and correlations exist in the data.

Soft computing methods are designed to handle very large data sets, and can be used to extract such relationships.

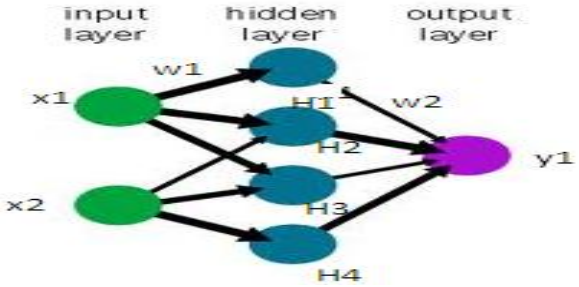
3.2. Relevance of Artificial Neural Network in Bioinformatics

An Artificial Neural Network (ANN) is an information processing model that is able to capture and represent complex input-output relationships. The motivation the development of the ANN technique came from a desire for an intelligent artificial system that could process information in the same way the human brain. Its novel structure is represented as multiple layers of simple processing elements, operating in parallel to solve specific problems. ANNs resemble human brain in two respects: learning process and storing experiential knowledge. An artificial neural network learns and classifies a problem through repeated adjustments of the connecting weights between the elements. In other words, an ANN learns from examples and generalizes the learning beyond the examples supplied.

Artificial neural network applications have recently received considerable attention. The methodology of modeling, or estimation, is somewhat comparable to statistical modeling. Neural networks should not, however, be heralded as a substitute for statistical modeling, but rather as a complementary effort or an alternative approach to fitting non-linear data.

The behavior of the neural network depends largely on the interaction between the different neurons. The basic architecture consists of three types of neuron layers: input, hidden, and output layers. A typical neural network (shown in Figure 1) is composed of input units X1, X2,... corresponding to independent variables, a hidden layer known as the first layer, and an output layer (second layer) whose output units Y1,... correspond to dependent variables (expected number of accidents per time period).

Figure 1: A simplified Artificial Neural Network



In between are hidden units H1, H2, … corresponding to intermediate variables. These interact by means of weight matrices W(1) and W(2) with adjustable weights. The values of the hidden units are obtained from the formulas:

$$H_j = f\left(\sum_k W_{jk}^{(1)} X_k \right)$$

$$Y_i = f\left(\sum_j W_{ij}^{(2)} H_j \right).$$

In One multiplies the first weight matrix by the input vector X = (X1, X2,...) and then applies an activation function f to each component of the result. Likewise the values of the output units are obtained by applying the second weight matrix to the vector H = (H1, H2,...) of hidden unit values, and then applying the activation function f to each component of the result. In this way one obtains an output vector Y= (Y1, Y2,...).The activation function f is typically of sigmoid form and may be a logistic function, hyperbolic tangent, etc.:

$$f(u) = \frac{1}{1 + e^{-u}}, \qquad f(u) = \frac{e^{u} - e^{-u}}{e^{u} + e^{-u}}.$$

Usually the activation function is taken to be the same for all components but it need not be.

Values of W(1) and W(2) are assumed at the initial iteration. The accuracy of the estimated output is improved by an iterative learning process in which the outputs for various input vectors are compared with targets (observed frequency of accidents) and an average error term E is computed:

$$E = \frac{\sum_{n=1}^{N} (Y^{(n)} - T^{(n)})^2}{N}.$$

Here

N = Number of highway sites or observations

Y(n) = Estimated number of accidents at site n for n = 1, 2,..., N

T(n) = Observed number of accidents at site n for n = 1, 2,..., N.

After one pass through all observations (the training set), a gradient descent method may be used to calculate improved values of the weights W(1) and W(2), values that make E smaller. After reevaluation of the weights with the gradient descent method, successive passes can be made and the weights further adjusted until the error is reduced to a satisfactory level. The computation thus has two modes, the mapping mode, in which outputs are computed, and the learning mode, in which weights are adjusted to minimize E. Although the method may not necessarily converge to a global minimum, it generally gets quite close to one if an adequate number of hidden units are employed.

The most delicate part of neural network modeling is generalization, the development of a model that is reliable in predicting future accidents. Overfitting (i.e., getting weights for which E is so small on the training set that even random variation is accounted for) can be minimized by having two validation samples in addition to the training sample. According to Smith and Thakar [38], the data set should be divided into three subsets: 40% for training, 30% to prevent overfitting, and 30% for testing. Training on the training set should stop at the epoch when the error E computed on the second set begins to rise (the second set is not used for training but merely to decide when to stop training). Then the third set is used to see how well the model performs. The cross validation helps to optimize the fit in three ways: by limiting/optimizing the number of hidden units, by limiting/optimizing the number of iterations, and by inhibiting network use of large weights.

The major advantages and disadvantages of neural networks in modeling applications are as follows:

Advantages

1. An ability to learn how to do tasks based on the data given for training or initial experience.
2. An ANN can create its own organisation or representation of the information it receives during learning time.
3. ANN computations may be carried out in parallel, and special hardware devices are being designed and manufactured which take advantage of this capability.
4. Partial destruction of a network leads to the corresponding degradation of performance. However, some network capabilities may be retained even with major network damage.

Applications

Neural networks have been widely used in biology since the early 1980s. They can be used to:

1. Predict the translation initiation sites in DNA sequences [23].
2. Explain the theory of neural networks using applications in biology [9].
3. Predict immunologically interesting peptides by combining an evolutionary algorithm [11].
4. Study human TAP transporter [10].
5. Carry out pattern classification and signal processing successfully in bioinformatics; in fact, a large number of applications of neural network can be found in this area.
6. Perform protein sequence classification; neural networks are applied to protein sequence classification by extracting features from protein data and using them in combination with the Bayesian neural network (BNN) [13].
7. Predict protein secondary structure prediction [30] [46].
8. Analyze the gene expression patterns as an alternative to hierarchical clusters [14]. Gene expression can even be analyzed using a single layer neural network [1]. Protein folds recognition using ANN and SVM [20].

In summary, a neural network is presented with a pattern on its input nodes, and the network produces an output pattern based on its learning algorithm during the training phase. Once trained, the neural network can be applied to classify new input patterns. This makes neural networks suitable for the analysis of gene expression patterns, prediction of protein structure, and other related processes in bioinformatics.

3.3. Relevance of Fuzzy Logic in Bioinformatics

Fuzzy logic is a relatively new technique (first appeared in 1970s) for solving engineering control problems. This technique can be easily used to implement systems ranging from simple, small or even embedded up to large networked ones. It can be used to be implemented in either software or hardware The key idea of fuzzy logic is that it uses a simple and easy way in order to get the output(s) from the input(s), actually the outputs are related to the inputs using if-statements and this is the secret behind the easiness of this technique. The most fascinating thing about Fuzzy logic is that it accepts the uncertainties that are inherited in the realistic inputs and it deals with these uncertainties in such away their affect is negligible and thus resulting in a precise outputs. Fuzzy logic is said to be the control methodology that mimics how a person decides but only much faster. One of the many advantages of fuzzy logic is that it really simplifies complex systems.

One may be surprised if told that he is using fuzzy logic statements (descriptions) almost every day.

For example when you say: “John is fat” and "Tom is tall", you are giving non-accurate descriptions about those people which turn to be exactly fuzzy logic descriptions, but if you say for example: "John is 80 kg" and "Tom is 190 cm", you are giving a certain and exact numbers which are NOT fuzzy descriptions. The difference between fuzzy logic and other type of logics is in terms of precision and significance. FL is a technique in which the significance is the most important while in other logics the precision is the important aspect.

There are several reasons behind the increasing use of this type of methodology in the world.

First of all, Fuzzy Logic reduces the design steps and simplifies complexity that might arise since the first step is to understand and characterize the system behavior by using knowledge and experience.

The concept of Fuzzy Logic (FL) was conceived by Lotfi Zadeh, a professor at the University of California at Berkley, and presented not as a control methodology, but as a way of processing data by allowing partial set membership rather than crisp set membership or non-membership. FL provides a simple way to arrive at a definite conclusion based upon vague, ambiguous, imprecise, noisy, or missing input information. It mimics human control logic.

Applications

Fuzzy systems have been successfully applied to several areas in practice. In bioinformatics, fuzzy systems play an important role for building knowledge-based systems. Most systems involve fuzzy logic-based and fuzzy rule-based models. They can control and analyze processes and diagnose and make decisions in biomedical sciences [7] [21].

There are many application areas in biomedical science and bioinformatics, where fuzzy logic techniques can be applied successfully. Some of the important uses of fuzzy logic are listed below:

1. To increase the flexibility of protein motifs [33].
2. To study differences between polynucleotides [43].
3. To analyze experimental expression data using fuzzy adaptive resonance theory [42].
4. To align sequences based on a fuzzy recast of a dynamic programming algorithm [37].
5. DNA sequencing using genetic fuzzy systems [16].
6. To cluster genes from micro-array data [13] [29].
7. To predict proteins sub-cellular locations from their dipeptide composition using fuzzy k- nearest neighbors algorithm. [24]
8. To simulate complex traits influenced by genes with fuzzy-valued e.ects in pedigreed populations [15].
9. To attribute cluster membership values to genes applying a fuzzy partitioning method, fuzzy C-means. [19]
10. To analyze gene expression data [45].

3.4. Relevance of Genetic Algorithms in Bioinformatics

Genetic algorithms [18][31], a biologically inspired technology, are randomized search and optimization techniques guided by the principles of evolution and natural genetics. They are efficient, adaptive, and robust search processes, producing near optimal solutions, and have a large degree of implicit parallelism. Therefore, the application of GAs for solving certain problems of bioinformatics, which need optimization of computation requirements, and robust, fast and close approximate solutions, appears to be appropriate and natural [25]. Moreover, the errors generated in experiments with bioinformatics data can be handled with the robust characteristics of GAs. To some extent, such errors may be regarded as contributing to genetic diversity, a desirable property. The problem of integrating GAs and bioinformatics constitutes a new research area.

GAs is executed iteratively on a set of coded solutions, called population, with three basic operators: selection/reproduction, crossover, and mutation. They use only the payoff (objective function) information and probabilistic transition rules for moving to the next iteration. Of all the evolutionarily inspired approaches, Gas seem particularly suited to implementation using DNA, protein, and other bioinformatics tasks [35]. This is because GAs are generally based on manipulating populations of bit-strings using both crossover and point-wise mutation.

Advantages

1. Several tasks in bioinformatics involve optimization of different criteria (such as energy, alignment score, and overlap strength), thereby making the application of Gas more natural and appropriate.
2. Problems of bioinformatics seldom need the exact optimum solution; rather, they require robust, fast, and close approximate solutions, which GAs are known to provide efficiently.
3. GAs can process, in parallel, populations billions times larger than is usual for conventional computation. The usual expectation is that larger populations can sustain larger ranges of genetic variation, and thus can generate high-fitness individuals in fewer generations.
4. Laboratory operations on DNA inherently involve errors. These are more tolerable in executing evolutionary algorithms than in executing deterministic algorithms.

Applications

The most suitable applications of GAs in bioinformatics are:

1. Alignment and comparison of DNA, RNA, and protein sequences [14][39][40] [41].
2. Gene mappings in chromosomes [39][29].
3. Gene finding and promoter identification from DNA sequences [44].
4. Interpretation of gene expression and micro array data [28][22].
5. Gene regulatory network identification [32][39].
6. Construction of phylogenetic tree for studying evolutionary relationship [3].
7. DNA structure prediction [34].
8. RNA structure prediction [8][41].
9. Protein structure prediction and clustering [12][36].
10. Molecular design and molecular docking [17] [23][27].

4. Conclusions

Bioinformatics is a developing interdisciplinary science. The involvement of other sciences (such as computer science) holds great promise; this century's major research and development efforts will likely be in the biological and health sciences. Computer science departments planning to diversify their offerings can thus only gain through early entry into bioinformatics. Even using minimal resources, such efforts are wise, as computer science graduates will enhance their employment qualifications. Still unclear is whether bioinformatics will eventually become an integral part of computer science (in the same way as, say, computer graphics and databases) or will develop into an independent application. Regardless of the outcome, computer scientists are sure to benefit from being active and assertive partners with biologists.

Acknowledgments

This work received support from the Department of Computer Science & Engineering, Takshshila Institute of Engineering and Technology.

References

[1] Narayanan, E. Keedwell, and B. Olsson (2003), "Artificial Intelligence Techniques for Bioinformatics", Applied Bioinformatics, Vol.1, No. 4, pp. 191-222.

[2] P. Gultyaev, V. Batenburg and C. W. A. Pleij (1995), "The computer simulation of RNA folding pathways using a genetic algorithm", J. Mol. Biol., Vol. 250, pp. 37–51.

[3] R. Lemmon and M. C. Milinkovitch (2002), "The metapopulation genetic algorithm: An efficient solution for the problem of large phylogeny estimation", Proc. Nat. Acad. Sci., Vol. 99, No. 16, pp. 10516–10521.

[4] S. Wu and I. Garibay (2002), "The proportional genetic algorithm: Gene expression in a genetic algorithm", Genetic Programm. Evol. Hardware, Vol.3, No. 2, pp. 157–192.

[5] Salamov and V. Solovyev (1995), "Prediction of protein secondary structure by combining nearest-neighbor algorithms and multiple sequence alignments", J. Mol. Biol., Vol. 247, pp. 11–15.

[6] A. Skourikhine (2000), "Phylogenetic tree reconstruction using self-adaptive genetic algorithm", In IEEE Int. Symp. Bio-Informatics and Biomedical Engineering, pp.129–134.

[7] Adriaenssens V., Baetsb B. D., Goethalsa P. L. M. and Pauwa N. D. (2004), "Fuzzy rule-based models for decision support in ecosystem management", Science of The Total Environment, Vol. 319, pp. 1-12.

[8] Shapiro, J. C. Wu, D. Bengali and M. J. Potts (2001), "The massively parallel genetic algorithm for RNA folding: MIMD implementation and population variation", Bioinformatics, vol. 17, no. 2, pp. 137–148.

[9] Baldi P. and Brunak S. (1998), "Bioinformatics: the Machine Learning Approach", MIT Press.

[10] Brusic V., van Endert P., Zeleznikow J., Daniel S., Hammer J., and Petrovsky N. 1. (1999), "A neural

network model approach to the study of human TAP transporter", In Silico Biol. 1: pp. 9–21.

[11] Brusic V., Rudy G., Honeyman G., Hammer J. and Harrison L.(1998), " Prediction of MHC class II-binding peptides using an evolutionary Algorithm and artificial neural network", Bioinformatics, 14, pp. 121-130.

[12] Chen, L. H. Wang, C. Kao, M. Ouhyoung and W. Chen (1998), "Molecular binding in structure- based drug design: A case study of the population based annealing genetic algorithms", In Proc. IEEE Int. Conf. Tools with Artificial Intelligence, pp. 328–335.

[13] H. Wu and J. McLarty (2000), "Neural Networks and Genome Informatics", Elsevier Science.

[14] Zhang (1994), "A genetic algorithm for molecular sequence comparison", In Proc. IEEE Int. Conf. Systems, Man, and Cybernetics, Vol. 2, pp. 1926–1931.

[15] Carleos C., Rodriguez F., Lamelas H., Baro J. A. (2003), "Simulating complex traits influenced by genes with fuzzy-valued effects in pedigreed populations", ioinformatics, 19(1): 144–148.

[16] Cord´on O., Gomide F., Herrera F., Ho.mann F., Magdalena L. (2004), "Ten years of genetic fuzzy systems: current framework and new trends", Fuzzy Sets and Systems, 141(1): 5–31.

[17] E. Clark and D. R. Westhead (1996), "Evolutionary algorithms in computer aided molecular design", J. Comput.-Aided Mol. Design, Vol. 10, No. 4, pp. 337–358.

[18] Goldberg (1989), "Genetic Algorithms in Optimization, Search, and Machine Learning", Reading, MA: Addison-Wesley.

[19] Demb´el´e D., Kastner P. (2003), "Fuzzy C-means method for clustering microarray data", Bioinformatics, 19(8): 973–980.

[20] Ding C. H. Q. and Dubchak I. (2001), "Multi-class protein fold recognition using support vector machines and neural networks", Bioinformatics, 17: 349–358.

[21] Edwin Lughofer and Carlos Guardiola (2008), "Applying Evolving Fuzzy Models with Adaptive Local Error Bars to On-Line Fault Detection", International Workshop on Genetic and Evolving Fuzzy Systems.

[22] H. K. Tsai, J. M. Yang, Y. F. Tsai and C. Y. Kao (2004), "An evolutionary approach for gene expression patterns", IEEE Trans. Inf. Technol. Biomed., Vol. 8, No. 2, pp. 69–78.

[23] Hatzigeorgiou, A. G. Reckzo, M. (2004), "Signal peptide prediction on DNA sequences with artificial neural networks", Biomedical Circuits and Systems.

[24] Huang Y., Li Y. (2004), "Prediction of protein subcellular locations using fuzzy k-NN method", Bioinformatics, 20(1): 21–28.

[25] J. Setubal and J. Meidanis (1999), "Introduction to Computational Molecular iology", Boston, MA: Thomson.

[26] J. D. Szustakowski and Z. Weng (2000), "Protein structure alignment using a genetic algorithm", Proteins, Vol. 38, No. 4, pp. 428–440.

[27] J. M. Yang and C. Y. Kao (2000), "A family competition evolutionary Algorithm for automated docking of flexible ligands to proteins", IEEE Trans. Inf. Technol. Biomed., Vol. 4, No. 3, pp. 225–237.

[28] J. Quackenbush (2001), "Computational analysis of microarray data", Nat. Rev. Genetics, Vol. 2, pp. 418–427.

[29] J. W. Fickett (1996), "Finding genes by computer: The state of the art", Trends Genetics, Vol. 12, No. 8, pp. 316–320.

[30] K. Chenand, L. Kurgan (2007), "PFRES: protein fold classification by using evolutionary information and predicted secondary structure", Bioinformatics, 23(21): 2843 – 2850.

[31] L. B. Booker, D. E. Goldberg and J. H. Holland (1989), "Classifier systems and genetic algorithms", Artif. Intell., Vol. 40, No. 1–3, pp. 235–282.

[32] Leping Li 1,*, Yu Liang 1 and Robert L. Bass (2007), "GAPWM: a genetic algorithm method for optimizing a position weight matrix", Bioinformatics, 23(10): 1188-1194.

[33] Luis Taria, Chitta Barala, and Seungchan Kim (2008), "Fuzzy c-means clustering with prior biological knowledge", Journal of Biomedical Informatics.

[34] P. Baldi and P. F. Baisnee (2000), "Sequence analysis by additive scales: DNA structure for sequences and repeats of all lengths", Bioinformatics, Vol. 16, pp. 865–889.

[35] S. B. Needleman and C. D. Wunsch (1970), "A general method applicable to the search for

similarities in the amino acid sequence of two proteins", J. Mol. Biol., Vol. 48, pp. 443–453.

[36] S. Schulze-Kremer (2000), "Genetic algorithms and protein folding. Methods in molecular biology", Protein Structure Prediction: Methods and Protocols, Vol. 143, pp. 175–222.

[37] Schlosshauer M., Ohlsson M. (2002), "A novel approach to local reliability of sequence alignments", Bioinformatics, 18(6):847–854.

[38] Schneider Murray-Smith, R. and Thakar, S. (1993), "Combining case based reasoning with neural networks", In AAAI Workshop on Case Based Reasoning, Washington, D. C., USA.

[39] T. Hou, J. Wang, L. Chen and X. Xu (1999), "Automated docking of peptides and proteins by using a genetic algorithm combined with a tabu search", Protein Eng., Vol. 12, pp. 639–647.

[40] T. F. Smith and M. S. Waterman (2001), "Identification of common Informatics: Edmonton", AB, Canada: IMIA, pp. 83– 100.

[41] T. Murata and H. Ishibuchi (1996), "Positive and negative combination effects of crossover and mutation operators in sequencing problems", Evol.Comput., Vol. 20–22, pp. 170–175.

[42] Tomida S., Hanai T., Honda H., Kobayashi T. (2002), "Analysis of expression profile using fuzzy adaptive resonance theory", Bioinformatics, 18(8):1073–1083.

[43] Torres A, Nieto J. J. (2003), "The fuzzy polynucleotide space: basic properties", Bioinformatics, 19(5): 587–592.

[44] V. G. Levitsky and A. V. Katokhin (2003), "Recognition of eukaryotic promoters using a genetic algorithm based on iterative discriminant analysis", In Silico Biol., Vol. 3, No. 1–2, pp. 81–87.

[45] Woolf P. J., Wang Y. (2000), "A fuzzy logic approach to analyzing gene expression data", Physiological Genomics, 3(1): 9–15.

[46] Zhong Wei1, Altun Gulsah, Tian Xinmin, Harrison Robert, Tai Phang, Pan Yi (2007)," Parallel protein secondary structure prediction schemes using Pthread and OpenMP over hyper-threading technology", The Journal of Supercomputing, Vol. 41, No. 1, pp. 1-16.

[47] R K. Jena, M. Aqel, Pankaj Srivastava, "Soft Computing Methodologies in Bioinformatics", European Journal of Scientific Research, Vol.26 No.2 (2009), pp.189-203.

[48] WWW.123seminarsonly.com

Swati Jain received the BE degree in Computer Science Engineering, and the ME degree in Computer Engineering, from the RGPV university Bhopal, in 2007 and 2010 respectively. Currently she is an assistant professor at the department of Computer Science, Takshshila Institute of Technology, Jabalpur [M.P.], India. Her research interests include soft computing, artificial intelligence, neural network, fuzzy logic, genetic algorithm, computer graphics, data structure and software engineering. She is a member of the ISTE.

Abhishek Pandey BE [I.T.], ME[C.S.E] is an assistant professor at the Department of Computer Science, Takshshila Institute of Technology, Jabalpur[M.P.], India. His research interests include computer graphics, data structure and software engineering.