



Particle Swarm Optimisation for Protein Motif Discovery

BILL C. H. CHANG
ASANGA RATNAWEERA
SAMAN K. HALGAMUGE

Mechatronics Research Group, Mechanical and Manufacturing Engineering, University of Melbourne, Australia

HARRY C. WATSON

Thermofluids Research Group, Mechanical and Manufacturing Engineering, University of Melbourne, Australia

Submitted March 5, 2003; Revised December 17, 2003

Abstract. In this paper, a modified particle swarm optimisation algorithm is proposed for protein sequence motif discovery. Protein sequences are represented as a chain of symbols and a protein sequence motif is a short sequence that exists in most of the protein sequence families. Protein sequence symbols are converted into numbers using a one to one amino acid translation table. The simulation uses EGF protein and C2H2 Zinc Finger protein families obtained from the PROSITE database. Simulation results show that the modified particle swarm optimisation algorithm is effective in obtaining global optimum sequence patterns, achieving 96.9 and 99.5 classification accuracy respectively in EGF and C2H2 Zinc Finger protein families. A better true positive hit result is achieved when compared to the motifs published in PROSITE database.

Keywords: particle swarm optimisation, protein sequence motif, motif discovery, symbolic data optimisation

Introduction

With the close completion of the Human Genome Project (HGP), there is a great amount of interest in the analysis of genomic data. In 2000, the editors at the international journal, *Science*, have put the genomic projects research at the top of the scientific development for the year [22].

Apart from understanding the patterns exist in DNA sequences, research in protein functional properties is also very important. Protein is a chain of amino acids and its sequence is represented by a series of English alphabet each representing an amino acid. A Protein sequence motif, signature or consensus pattern, is a short sequence that is embedded within the sequences of a same protein family [7]. A motif can be used to classify unknown protein sequences into their corresponding protein family/families for further biological analysis otherwise long experimental research steps. In past years, many algorithms for finding protein sequence motifs have been proposed. Some use real biological data to test against their algorithms [18, 34, 37, 10], while some studies tackle the problem of protein motif identification using artificially generated data [30, 35, 9].

Sequence motif discovery algorithms can be generally categorized into 3 types: (1) String Alignment algorithms, (2) Exhaustive enumeration algorithms, and (3) Heuristic methods.

String alignment algorithms [41, 14, 29] find sequence motifs by minimizing a cost function which is related to the edit distances between sequences. Multiple alignment of sequences is a NP-hard problem and its computational time increases exponentially with the sequence size. Local search algorithms such as Gibbs sampling [26], expectation maximization [3, 27] may end up in a local optimum instead of finding the best motif [9]. Exhaustive enumeration algorithms [6, 8, 17] are guaranteed to find the optimal motif, but run in exponential time with respect to the length of motif. Heuristic methods [21, 36, 40] can have a better performance but are usually less flexible.

There are a number of protein motif databases such as PROSITE [20], BLOCKS [19], PRINTS [2] and PFAM [4] are available. The differences between them are the methods used to generate sequence motifs and profiles. PROSITE database is used in this paper because of its high biological significance of the patterns. Each PROSITE entry is well documented and provided with its biological information.

In this paper, extension of a particle swarm optimisation algorithm for symbolic data is introduced, which aims to find protein sequence motifs that are unique to a family of proteins. In the following sections, the algorithm, simulation results and discussion are presented.

Background on particle swarm optimization

Particle swarm optimization method is an evolutionary optimization technique first introduced by Kennedy and Eberhart [23] in 1995. Particle swarm optimization method mimics the development of this technique was inspired by the animal social behaviors such as school of fish, flock of birds etc and mimics the way they find food sources and prevent from predators.

Particle swarm algorithm starts with the random initialization of a population of individuals (particles) in the search space and works on the social behavior of the particles in the swarm. Therefore, it finds the global best solution by simply adjusting the trajectory of each individual towards its own best location and towards the best particle of the swarm at each time step (generation) [23, 12, 13]. However, the trajectory of each individual in the search space is adjusted by dynamically altering the velocity of each particle, according to its own flying experience and the flying experience of the other particles in the search space.

The position and the velocity of the i th particle in the d -dimensional search space can be represented as $X_i = (x_{i1}, x_{i2}, x_{i3}, \dots, x_{id})$ and $V_i = (v_{i1}, v_{i2}, v_{i3}, \dots, v_{id})$ respectively. Each particle has its best position $P_i = (p_{i1}, p_{i2}, p_{i3}, \dots, p_{id})$ correspondent to the personal best fitness value obtained so far at time t , with reference to the user defined objective function. The global best particle, which represents the fittest particle found so far at time t in the entire swarm is denoted by P_g . The new velocity of each particle is calculated according to the following equation.

$$v_{id(t+1)} = \omega v_{id(t)} + c_1 \times \text{rand}() \times (p_{id} - x_{id}) + c_2 \times \text{Rand}() \times (p_{gd} - x_{id}) \quad (1)$$

where c_1 and c_2 are constants and are known as acceleration coefficients, ω is called the inertia factor and $\text{rand}()$ and $\text{Rand}()$ are two sparely generated uniformly distributed random numbers in the range of [0, 1].

At each iteration (generation) the position of each particle is updated according to the following equation

$$x_{id} = x_{id} + v_{id} \quad (2)$$

The inertial weight factor provides the necessary diversity to the swarm by changing the momentum of particles and hence avoids the stagnation of particles at local optima. The empirical investigations conducted by Eberhart and Shi [16] shows the importance of decreasing the value of ω from a higher value (usually 0.9 to 0.4) during the search. Further, they have found improved performance for most of the commonly used benchmarks when a random inertia weight factor is used [15].

It is common practice in particle swarm optimization to define a maximum velocity for each modulus of velocity vector. Usually the upper limit of the each modulus of position vector is used as the maximum velocity. This helps to control the unnecessary excessive roaming of particles outside the predefined search space.

Since the introduction of the particle swarm optimization concept in 1995, there has been a considerable amount of work done in developing the original version. However, most of these developments are based on empirical simulations on numerical benchmarks. Kennedy [24] introduced the stereotypical behavior and beliefs of groups of humans to the particle swarm algorithm, through cluster analysis. Bergh and Engelbrech [39] proposed the concept “cooperative particle swarm” by splitting the full solution vector into small vectors and each sub vector was optimized using separate particle swarm optimization. Angeline [1] introduced a hybrid particle swarm optimization strategy by adding a standard selection mechanism from evolutionary programming and Lovbejerg, Rasmussen and Krink [28] introduced another hybrid method, in which the new particles are generated through breeding among existing particles.

Since the particle swarm optimization concept is relatively new, this has not been applied on wide range of practical applications as other evolutionary algorithms yet. However, there are few attempts to apply particle swarm optimization concept to evolve weights and the structure of artificial neural networks as reported in [23, 15]. Furthermore, Yoshida [42] has applied this algorithm for reactive power and voltage control in power system, by modeling the power system as a mixed-integer non-linear optimization problem. Laskari [25] investigated about the possibilities of applying the particle swarm technique on integer programming problems and, concluded that the particle swarm can handle those problems more efficiently, when compared to conventional Branch and Bound techniques. Blackwell and Bentley [5] introduced the concept of SWARMUSIC” by developing a novel interactive and improvising musical system using the particle swarm algorithm. Application of the particle swarm optimization method on numerical control end milling optimization, and in the field of electromagnetics were also reported [38, 11].

Based on some previous developments on the original particle swarm optimization concept, the authors have also reported number of extensions [32, 33]. In this work, the Self-Organizing Hierarchical Particle Swarm Optimizer with time varying acceleration coefficients (**HPSO-TVAC**) strategy developed by the authors is used. Under this particle swarm optimization strategy, a series of particles swarm optimizers are automatically generated inside the main particle swarm optimizer according to the behavior of the particles in the search space, until the convergence criteria is met. The pseudo code for **HPSO-TVAC** is shown in Figure 1.

```

HPSO-TVAC
begin
  initialise the population
  while (termination condition = false)
    do
      for (i=1 to number of particles)
        evaluate the fitness: =f(x)
        update  $P_{id}$  and  $P_{gd}$ 
        for d =1 to number of dimensions
          calculate the new velocity  $v_{id} = c_1 * rand1() * (p_{id} - x_{id}) + c_2 * rand2() * (p_{gd} - x_{id})$ 
          if ( $v_{id} = 0$ )
            if ( $rand3() < 0.5$ )
               $v_{id} = rand4() * v$ 
            else
               $v_{id} = - rand5() * v$ 
            end if
          end if
           $v_{id} = sign(v_{id}) * min(abs(v_{id}), v_{max})$ 
          update the position
        increase d
      increase i
    end do
  end

```

Figure 1. Pseudo code for **HPSO-TVAC**.

Where i is a particle in a d -dimensional search space and x and v represent the position and velocity vectors respectively. Vector p represents the best individual according to a user defined fitness function and g is the global best particle. v is the re-initialization velocity and $\text{randi}(\cdot)$, $i = 1, 2, \dots, 4$ are separately generated, uniformly distributed random numbers in the range $[0, 1]$.

Particle swarm optimisation for protein sequence data

Protein sequences

Protein sequences are chains of amino acids and there exists 20 different types of amino acid. Each amino acid is represented by an English alphabet from A to Z, excluding B, J, O, U, X, and Z. A sequence data can be interpreted as a series of *events*, E_i , separated by their *event intervals*, I_{ij} . A sequence can be described as:

$$E_1 - I_{1,2} - E_2 - I_{2,3} - E_3 \cdots - I_{(n-1),n} - E_n$$

For example, the sequence *CGC* has three events *C*, *G* and *C*. The event intervals between *C* and *G*, and *G* and *C* are both zero (i.e. $I_{1,2} = I_{2,3} = 0$). The concept of *event intervals* is important when the search pattern contains *wild cards*. A wild card, usually represented by letter “X” in molecular biology literature, can be matched to any other symbols. For instance, sequences *TXG* and *TSG* are considered to be an “exact match” as *X* can be matched to *S* (in this case) or any of the possible symbols/events. Since *X* is a wild card and not an identified event for the search algorithm, the sequence *TXG* has only two events, *T* and *G* separated by an event interval of one.

The aim of this algorithm is to find a consensus pattern, or motif, from sequences belonging to the same family. This motif can be either a rigid or flexible pattern. A rigid pattern may be $S-x(5)-T$, where there exist a fixed number of gaps/wildcards (in this case, 5) between two events *S* and *T*. In a flexible pattern, the value of event interval is represented by a lower bound and an upper bound, such as $x(2, 4)$.

Converting protein sequence data into numerical form

In the case of protein motif discovery, data are in symbolic form and need to be converted into numeric forms to implement particle swarm optimisation. The amino acids symbols are converted into numbers as shown in Table 1.

Therefore, a short sequence *T-S-Q* is translated into 16-15-13. Furthermore, by adopting the sequence structure described above, which includes event intervals, *T-S-Q* is translated into 16-0-15-0-13.

Particle swarm algorithm and fitness function for protein motif discovery

There are two major steps considered in protein motif discovery: *initial motif generation*, and *motif optimisation*. *Initial Motif Generation* generates a rigid motif pattern with satisfactory classification accuracy. After this initial motif is generated (e.g., $A-x(5)-C$), *Motif*

Table 1. Amino Acid (AA) coding system for PSO optimisation

AA symbol	AA number	AA symbol	AA number	AA symbol	AA number
A	0	I	7	R	14
C	1	K	8	S	15
D	2	L	9	T	16
E	3	M	10	V	17
F	4	N	11	W	18
G	5	P	12	Y	19
H	6	Q	13		

Optimisation step then optimises the event interval distances to increase its classification rate (e.g., A-x(2, 6)-C).

Initial motif generation. The HPSO-TVAC algorithm is slightly modified for protein sequence discovery application. With the symbol-to-number amino acid coding system described above, there is a distance relationship between each of the amino acid. However, without considering biological similarities between amino acids, the distance between any two amino acids should be the same. For example, the distance between amino acid A and C should be the same as the distance between A and Y. The downside of this coding system is that, a particle may be “stuck” in local minimum at A, whereas the global minimum is at Y. Even though the particle swarm strategy does allow for small random movement when the computed velocity is 0 (in order to escape the local minimum), all the particles will be attracted towards A (0) and never be able to reach the global minimum at Y (19). In order to make particles “travel” through the data space better, the algorithm is modified as follows:

$$\begin{aligned} \text{If } x_{id} < X_{LB}, x_{id} &= X_{UB}, \\ \text{If } x_{id} > X_{UB}, x_{id} &= X_{LB}, \end{aligned}$$

where x_{id} is the next particle position, X_{LB} is the lower bound of particle x_i in dimension d , and X_{UB} is the upper bound of particle x_i in dimension d .

Another modification is the calculation of particle velocity when v_{id} is 0. Since the input space of particles is discrete, v_{id} is set to either +1, -1 or 0. The modified algorithm is shown as in Figure 2.

Fitness function

Fitness function is very important in optimisation problems as the global optimum should have a maximum or minimum fitness function value over all the other possible input values. Also, the fitness function values of solutions close to global optimum should also have a fitness value close to that of global optimum. For protein sequence discovery, various fitness functions were tested:

```

if ( $v_{id} = 0$ )
    if ( $\text{rand3}() < 0.2$ )
         $v_{id} = 1$ 
    else if ( $\text{rand3}() > 0.8$ )
         $v_{id} = -1$ 
    else
         $v_{id} = 0$ 
    end if
end if

```

Figure 2. Pseudo code for modified particle velocity algorithm.

- (1) When two sequences are compared, an exact match is given 1 point, and the rests are given 0 points.
- (2) When two sequences are compared, each of the matched symbols is given 1 point, and the rests are given 0 points,
- (3) Same as (2), and bonus points are given for exact matched sequence.

For example, considering the three data sequence: TSCG, TSCG, TSCH and CSCC, and four particles: TSCG, TSCH, TSCC and CCCC, their fitness values in these three methods are:

Fitness functions	TSCG	TSCH	TSCC	CCCC
(1)	2	1	0	0
(2)	$4 + 4 + 3 + 2 = 13$	$3 + 3 + 4 + 2 = 12$	$3 + 3 + 3 + 3 = 12$	$1 + 1 + 1 + 3 = 6$
(3)	$8 + 8 + 3 + 2 = 21$	$3 + 3 + 8 + 2 = 16$	$3 + 3 + 3 + 3 = 12$	$1 + 1 + 1 + 3 = 6$

Fitness function (1) is not desirable, as CSCC and CCCC both have the same fitness value even though CSCC is closer to the four data sequences than CCCC. This fitness function value creates a “spiked” fitness function value space where the fitness values of solutions close to global minimum is the same as other random values.

Fitness function (3) is preferred than (2) because in motif discovery, exact match is more desirable than partial matches. In fitness function (2), both TSCH and TSCC have the same fitness function value even though TSCH has 1 exact match in the database whereas TSCC does not. In fitness function (3), the above problem is overcome as bonus points are awarded for exact matches. In this paper, fitness function (3) is used in the simulation.

Motif optimisation. After the initial motif has been generated, it is then optimised by tuning event intervals. Particles represent the event interval distances of the initial motif, and they are initialized around the values of initial motif. The fitness function used for this step is the same as in *Initial Motif Generation*, with an additional rule:

If two particles have the same fitness function value, then the best particle is the one with a smaller interval gap values (interval upper bound value – interval lower bound value).

The reason to have a smaller interval gap value is to reduce false positives identified by the motif pattern.

Simulation and results

In this simulation, protein data from PROSITE database (release 39, PROSITE website) is used. Each PROSITE entry has been carefully grouped and documented by an expert, using biological information [18]. Two protein families—C2H2 Zinc Finger Protein and EGF Proteins—are used to demonstrate the effectiveness of the particle swarm optimisation algorithm. Motif patterns generated using the proposed algorithm are then compared with that of PROSITE and combinatorial method with neuro-fuzzy [10] motif patterns. PROSITE motif patterns are generated by performing multiple sequence alignment over the selected biological significant regions. For comparison purposes, the size of a particle is predetermined so that its generated motif pattern has a same structure (lengthwise) as the PROSITE motif.

EGF protein

“The functional significance of epidermal growth factor (EGF) domains in what appear to be unrelated proteins is not yet clear. However, a common feature is that these repeats are found in the extracellular domain of membrane-bound proteins or in proteins known to be secreted (exception: prostaglandin G/H synthase).” [31].

The EGF protein has 446 sequences and they are all used for motif extraction experiment in this study. A population of 100 particles is randomly initialized, and there are 100 iterations. In the initial motif generation step, each value of a particle represents either an event (amino acid) or an event interval. 10 sets of simulation are performed and number of dimensions (d) for a particle is set to 7 for algorithm comparison purposes. The best particle (the pattern with highest fitness value) is then selected as the initial motif. The results is shown in Table 2.

From the simulation results, the particle C-x(1)-C-x(5)-G-x(2)-C (1-1-1-5-5-2-1) is selected as the initial motif for this protein sequence family as it has the best fitness value. With this initial motif, 100 particles of dimension 6 (3 event intervals, each with a upper and lower bound) are initialized around $\{[1, 1], [5, 5], [2, 2]\}$ and PSO is applied again. This optimisation can be completed quite quickly compared to the previous step and only 1 simulation run is required (because of a smaller solution space). The results of this optimisation is shown in Table 3 and is comparable to the Combinatorial Method with Neuro-Fuzzy optimisation technique.

Table 2. Simulation results for 10 runs of initial motif generation for EGF protein sequences

No.	Best particle	Fitness	Match	No.	Best particle	Fitness	Match
1	9-5-9-11-15-4-9	1336	28	6	1-1-1-5-5-2-1	2836	314
2	1-1-1-2-5-2-5	2706	279	7	9-1-9-0-9-0-9	1780	106
3	5-0-9-5-9-7-5	1318	29	8	9-5-9-3-0-4-5	1330	30
4	9-5-9-0-9-6-14	1465	54	9	1-1-1-5-5-2-1	2836	314
5	12-2-11-0-5-2-1	1615	88	10	9-1-5-7-9-4-15	1377	30

“Match” indicates the number of identical matches between particle (motif) and database. Each particle in the “Best Particle” column represents a motif candidate using the notation $E_1 - I_{1,2} - E_2 - I_{2,3} - E_3 \dots - I_{(n-1),n} - E_n$, where E_i is amino acid (see Table 1 for coding system) and I_{ij} is the gap distance between E_i and E_j .

Table 3. Classification performance of PROSITE [20], Combinatorial Method with Neuro-Fuzzy optimisation [10] and PSO optimisation motif pattern for EGF proteins

Motif	Exact matches
PROSITE published motif C-x(1)-C-x(5)-G-x(2)-C	314/446 (70.4%)
Combinatorial Method with Neuro-Fuzzy optimisation C-x(1,4)-C-x(3,7)-G-x(1,4)-C	432/446 (96.9%)
PSO optimisation C-x(1,4)-C-x(3,7)-G-x(1,4)-C	432/446 (96.9%)

C2H2 Zinc Finger protein

“C2H2 are nucleic acid-binding protein structures first identified in the *Xenopus* transcription factor TFIIIA. These domains have since been found in numerous nucleic acid-binding proteins. A zinc finger domain is composed of 25 to 30 amino-acid residues. There are two cysteine or histidine residues at both extremities of the domain, which are involved in the tetrahedral coordination of a zinc atom” [31].

The C2H2 Zinc Finger protein has 418 sequences and they are all used for motif extraction experiment in this study. A similar approach is taken that an initial population of 100 and 100 iterations are used in a simulation. The results of 10 set of simulation is shown in Table 4, and from the results, the pattern C-x(2)-C-x(12)-H-x(3)-H (1-2-1-12-6-3-6) is selected as

Table 4. Simulation results for 10 runs of initial motif generation for C2H2 zinc finger protein sequences

No.	Best particle	Fitness	Match	No.	Best particle	Fitness	Match
1	12-5-1-3-4-7-14	1955	164	6	3-7-1-5-13-8-14	1478	95
2	12-7-14-10-6-2-16	1490	65	7	3-4-1-3-5-3-15	1786	137
3	1-2-1-12-6-3-6	2971	352	8	8-7-9-5-7-2-5	1820	142
4	12-5-1-7-2-7-16	1298	49	9	6-7-8-8-2-15	1746	131
5	12-5-1-7-0-6-14	1309	43	10	3-0-8-7-5-8-9	1953	169

Table 5. Classification performance of PROSITE [20], Combinatorial Method with Neuro-Fuzzy optimisation [10] and PSO optimisation motif pattern for C2H2 Zinc Finger proteins

Motif	Exact matches
PROSITE published motif C-x(2,4)-C-x(3)-[LIVMFYWC]-x(8)-H-x(3,5)-H	414/418 (98.8%)
Combinatorial Method with Neuro-Fuzzy optimisation C-x(1,4)-C-x(11,15)-H-x(2,5)-H	416/418 (99.5%)
PSO optimisation C-x(1,4)-C-x(11,15)-H-x(2,5)-H	416/418 (99.5%)

the initial motif after simulations. 100 Particles are then initialised around $\{[2, 2], [12, 12], [3, 3]\}$ to find the optimum flexible pattern. After 17 iterations, the optimum flexible pattern is obtained and is shown in Table 5.

Discussion

The simulation results show that PSO algorithm can be successfully applied to symbolic sequence data. For both EGF proteins and C2H2 Zinc Finger proteins, optimum solutions are obtained and they are the same as solutions obtained by Combinatorial Method with Neuro-Fuzzy Optimisation. The advantages of PSO algorithm is that it is much faster in motif optimisation step (second step) when compared to Neuro-Fuzzy Optimisation. Both these techniques are fully automatic compared to PROSITE which are compiled manually.

The disadvantages of PSO algorithm is particles are sometimes “stuck” in local minimum and therefore, may never reach the global optimum in a simulation. This problem however, can be overcome by running more simulations or introduce more initial particles at the start. Simulation time can increase exponentially with the increased initial particle population.

Determination of fitness function for optimisation algorithm can be quite difficult. The goal of fitness function is to give the optimum solution a global maximum/minimum fitness value. This is not a difficult criterion considering the fact that a protein sequence motif is a short sequence pattern that exists most frequently in a protein sequence family. However, a good fitness function should also give the “closer to optimum fitness values” for patterns that are close to the global optimum (i.e., motif). This is very important in optimisation tasks because it would almost be impossible to optimise if a closer to optimum solution is not detected. This is a more challenging task considering protein sequences are combinations of symbols and the number of matches say for AAA and AAC can vary significantly. The fitness function proposed in this paper has been successfully applied, however, this may be extended.

Conclusion and future work

In this paper, particle swarm optimisation algorithm has been successfully applied to protein sequence motif discovery problem. The simulation results indicate that this algorithm can be used to obtain global optimum protein sequence motifs.

In future work, fitness function formulation and amino acid coding system with biological focus should be further investigated. By including biological information into the optimisation algorithm, biological functional properties of proteins may be interpreted in a more meaningful way, and hence, obtaining better protein sequence motifs in biological sense.

References

1. P. J. Angeline, "Using selection to improve particle swarm optimisation," in Proceedings of the IEEE International Conference on Evolutionary Computation, IEEE Inc.: New York, NY, USA, 1998, pp. 84–89.
2. T. Attwood, D. Flower, A. Lewis, J. Mabey, S. Morgan, P. Scordis, J. Selley, and W. Wright, "PRINTS prepares for the new millenium," *Nucleic Acids Research* vol. 27, pp. 220–225, 1999.
3. T. Bailey and C. Elkan, "Unsupervised learning of multiple motifs in biopolymers using expectation maximisation," *Machine Learning* vol. 21, nos. 1/2, pp. 51–80, 1995.
4. A. Bateman, E. Birney, R. Durbin, S. Eddy, R. Finn, and E. Sonnhammer, "Pfam 3.1: 1313 multiple alignments and profile HMMs match the majority of proteins," *Nucleic Acids Research* vol. 27, pp. 260–262, 1999.
5. T. M. Blackwell and P. Bentley, "Improvised music with swarms," in Proceedings of the IEEE Congress on Evolutionary Computation, Honolulu, Hawaii, USA, 2002.
6. M. Blanchette, B. Schwikowski, and M. Tompa, "An exact algorithm to identify motifs in orthologous sequences from multiple species," in Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology, P. Bourne et al. (eds.), AAAI Press: San Diego, CA, 2000, pp. 37–45.
7. P. Bork and E. Koonin, "Protein sequence motifs," *Curr. Opin. Struct. Biol.* vol. 6, pp. 366–376, 1996.
8. A. Brazma, I. Jonassen, J. Vilo, and E. Ukkonen, "Predicting gene regulatory elements *in silico* on a genomic scale," *Genomic Research* vol. 15, pp. 1202–1215, 1998.
9. J. Buhler and M. Tompa, "Finding motifs using random projections," in Proceedings of RECOMB 2001, ACM, Montreal, Canada, 2001, pp. 69–76.
10. B. Chang and S. Halgamuge, "Protein Motif Extraction with Neuro-Fuzzy Optimisation," *Bioinformatics* vol. 18, pp. 1084–1090, 2002.
11. G. Ciuprina, D. Ioan, and I. Munteanu, "Use of intelligent-particle swarm optimization in electromagnetics," *IEEE Transactions on Magnetics* vol. 38, no. 2, pp. 1037–1040, 2002.
12. M. Clerc, "The swarm and the queen: Towards a deterministic and adaptive particle swarm optimisation," in Proceedings of the IEEE International Congress on Evolutionary Computation, IEEE Neural Networks Council (ed.), IEEE Inc.: Washington DC, 1999, vol. 3, p. 1951.
13. M. Clerc and J. Kennedy, "The particle swarm—Explosion, stability and convergence in a multi-dimensional complex space," *IEEE Trans. Evolutionary Computation* vol. 6, pp. 58–73, 2002.
14. A. Delcoigne and P. Hansen, "Sequence comparison by dynamic programming," *Biometrika* vol. 62, pp. 661–664, 1975.
15. R. C. Eberhart and Y. Shi, "Particle swarm optimisation: Developments, applications and resources," in Proceedings of the IEEE International Conference on Evolutionary Computation, IEEE Inc.: Piscataway, NJ, USA, 2001, vol. 1, pp. 81–86.
16. R. C. Eberhart and Y. Shi, "Comparing inertia weights and construction factors in particle swarm optimisation," in Proceedings of the IEEE International Congress on Evolutionary Computation, IEEE Neural Networks Council (ed.), Jolla, California, USA, 2000, vol. 1, pp. 84–88.
17. D. Galas, M. Eggert, and M. Waterman, "Rigorous pattern-recognition methods for DNA sequences: Analysis of promoter sequences from *Escherichia coli*," *Journal of Molecular Biology* vol. 186, no. 1, pp. 117–128, 1985.
18. R. Hart, A. Royyuru, S. Stolovitzky, and A. Califano, "Systematic and automated discovery of patterns in PROSITE families," in RECOMB 2000, ACM: Tokyo, Japan, 2000, pp. 147–154.
19. S. Henikoff, J. Henikoff, and S. Pietrokovski, "Blocks: A non-redundant database of protein alignment blocks derived from multiple compilations," *Bioinformatics* vol. 15, pp. 471–479, 1999.
20. K. Hoffman, P. Bucher, L. Falquet, and A. Bairoch, "The PROSITE database, its status in 1999," *Nucleic Acids Research* vol. 27, pp. 215–219, 1999.

21. I. Jonassen and D. Higgins, "Finding flexible patterns in unaligned protein sequences," *Protein Science* pp. 1587–1595, 1995.
22. D. Kennedy, "Breakthrough of the year," *Science* vol. 290, no. 5500, p. 2255, 2000.
23. J. Kennedy and R. Barnhart, "Particle swarm optimisation," in *Proceedings of the IEEE International Conference on Neural Networks*, IEEE Neural Networks Council (ed.), IEEE Inc.: Perth, Western Australia, 1995, pp. 1942–1948.
24. J. Kennedy, "Stereotyping: Improving particle swarm performance with cluster analysis," in *Proceedings of the IEEE International Conference on Evolutionary Computation*, IEEE Neural Networks Council (ed.), IEEE Inc.: LA Jolla, California, USA, 2000, vol. 2, pp. 303–308.
25. E. C. Laskari, K. E. Parsopoulos, and M. N. Vrahatis, "Particle swarm optimisation for integer programming," in *Proceedings of the IEEE International Congress on Evolutionary Computation*, IEEE Inc.: Piscataway, NJ, USA, 2002, vol. 2, pp. 1582–1587.
26. C. Lawrence, S. Altschul, M. Boguski, J. Liu, A. Neuwald, and J. Wootton, "Detecting subtle sequence signals: A Gibbs sampling strategy for multiple alignment," *Science* vol. 262, pp. 208–214, 1993.
27. C. Lawrence and A. Reilly, "An expectation maximisation (EM) algorithm for the identification and characterisation of common sites in unaligned biopolymer sequences," *Proteins: Structures, Functions, and Genetics* vol. 7, pp. 41–51, 1990.
28. M. Lovbjerg and T. Krink, "Extending particle swarm optimizers with self-organized critically," in *Proceedings of the IEEE International Congress on Evolutionary Computation*, IEEE Inc.: Piscataway, NJ, USA, 2002, vol. 2, pp. 1588–1593.
29. S. Needleman and C. Wunsch, "A general method applicable to the search for similarities in the amino acid sequence of two proteins," *Journal of Molecular Biology* vol. 48, pp. 443–453, 1970.
30. P. Pevzner and S. Sze, "Combinatorial approaches to finding subtle signals in DNA sequences," in *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology*, 2000, pp. 269–287.
31. PROSITE website: www.expasy.ch/prosite/
32. A. C. Ratnaweera, S. K. Halgamuge, and H. C. Watson, "Particle swarm optimisation with time varying acceleration coefficients," in *Proceedings of the International Conference on Soft Computing and Intelligent Systems*, Tsukuba, Japan, 2002.
33. A. C. Ratnaweera, S. K. Halgamuge, and H. C. Watson, "Particle swarm optimisation with self adaptive acceleration coefficients," in *Proceedings of the 1st International Conference on Fuzzy Systems and Knowledge Discovery*, Lipo Wang, Saman K. Halgamuge, and Xin Yao (eds.), Singapore, 2002, vol. 1, pp. 264–268.
34. I. Rigoutsos and A. Floratos, "Motif discovery without alignment or enumeration," in *RECOMB 98*, ACM: New York, USA, 1998, pp. 221–227.
35. M. Sagot, "Spelling approximate repeated or common motifs using a suffix tree," in *Latin '98: Theoretical Informatics*, C. Lucchesi and A. Moura (eds.), vol. 1380 of *Lecture Notes in Computer Science*, Springer, 1998, pp. 111–127.
36. M. Sagot and A. Viari, "A double combinatorial approach to discovering patterns in biological sequences," in *Proceedings of the 7th, Symposium on Combinatorial Pattern Matching*, D. Hirschberg et al. (eds.), Laguna Beach, California, USA, 1996, pp. 186–208.
37. R. Smith and T. Smith, "Automatic generation of primary sequence patterns from sets of related protein sequences," *Nucleic Acid Research* pp. 118–122, 1990.
38. V. Tandon, H. E. Mounayri, and H. Kishawy, "NC end milling optimisation using evolutionary computation," *International Journal of Machine Tools and Manufacture* vol. 42, no. 5, pp. 595–605, 2002.
39. F. Van den Bergh and A. P. Engelbrecht, "Effect of swarm size on cooperative particle swarm optimizers," *South African Computer Journal* vol. 26, pp. 84–90, 2000.
40. J. Wang, T. Marr, D. Shasha, B. Shapiro, and G. Chirn, "Discovering active motifs in sets of related protein sequences and using them for classification," *Nucleic Acids Research* pp. 2769–2775, 1994.
41. M. Waterman, D. Galas, and R. Arratia, "Pattern recognition in several sequences: Consensus and alignment," *Bulletin of Mathematical Biology* vol. 46, pp. 512–527, 1984.
42. H. Yoshida, K. Kawata, Y. Fukuyama, and Y. Nakanishi, "A particle swarm optimisation for reactive power and voltage control considering voltage stability," in *Proceedings of the International Conference on Intelligent System Application to Power System*, Rio de Janeiro, Brazil, 1999, pp. 117–121.