# HARD PROBLEMS IN GENE SEQUENCE ANALYSIS: CLASSICAL APPROACHES AND SUITABILITY OF GENETIC ALGORITHMS

L. Jantschi[1,3], S.D. Bolboaca[2,3], R.E. Sestras[3]
Technical University of Cluj-Napoca, Cluj, Romania[1]
"Iuliu Haţieganu" University of Medicine and Pharmacy Cluj-Napoca, Cluj, Romania[2]
University of Agricultural Sciences and Veterinary Medicine Cluj-Napoca, Cluj, Romania[3]
Correspondence to: Sorana D. Bolboaca
E-mail: sbolboaca@umfcluj.ro

## ABSTRACT
*Genetic algorithms are based on observations of natural phenomena as well as on the simulation of the artificial selection of organisms with multiple loci controlling a measurable trait. Genetic algorithms evolved into complex and strong informatics tools able to deal with hard problems of decision, classification, optimization, or/and simulation. We aimed to show how genetic algorithms can be used to solve hard problems on gene sequence analysis.*

**Keywords:** gene sequence analysis, genetic algorithm, hard problem, sequence alignment, classification

## Introduction
Systems occurring in nature are considered the most complex systems because they are the result of evolutionary processes (15). Nils Aall Barricelli applied evolutionary strategies to computer algorithms (5). Three years later, Alex Fraser published his first paper on the simulation of the artificial selection of organisms with multiple loci controlling a measurable trait (24). Fraser's simulations included all the essential elements of modern genetic algorithms.

Evolutionary algorithms are inspired by natural processes and are developed in order to optimize difficult or hard problems (9). A hard problem is defined as a problem with exponential complexity; classical algorithms are not able to provide an optimum solution to this kind of problems in real time (21).

Two different evolutionary algorithms were introduced in the 1970s: genetic algorithms (GAs) (7, 31) and evolution strategies (53, 57). Holland investigated the adaptation rather than the optimization of hard problems by studying the genetic algorithm. He applied the decision theory to the discrete domain. In contrast, Rechenberg and Schwefel investigated mutation in very small populations in order to optimize continuous parameters (53, 57). During the same period, two heuristics were introduced for solving hard problems that do not require the optimum solution: tabu search (26) and simulated annealing (16).

The aim of this study was to show how GAs can be used to solve gene hard problems from the field of sequence analysis. The following were studied: the classification of hard problems in gene sequence analysis; how genetic algorithms work; the usefulness of genetic algorithms in sequence alignment; the results of the classification of sequence alignments using genetic algorithms.

### Imposed Problems
Different bioinformatics methods are used to determine the biological function and/or structure of genes and of encoded proteins. Sequence analysis is an automated computer-based method that comprises the following steps (20):

- ◊ Sequence alignment: comparison of sequences in terms of similarity and dissimilarity;
- ◊ Sequence identification: identification of gene-structures, reading frames, introns (regions that are not translated into proteins), exons (the part of the open reading frame that codes a specific portion of the complete protein) and regulatory elements;
- ◊ Prediction of protein structures;
- ◊ Genome mapping;
- ◊ Comparisons of homologous sequences for constructing the molecular phylogeny.

In chemistry, sequence analysis comprises techniques used to determine the sequence of a polymer made of several monomers. In molecular biology and genetics this process is called "sequencing".

Multiple Sequence Alignment (MSA) is applied on three or more biological sequences (e.g. protein, deoxyribonucleic acid (DNA), or ribonucleic acid (RNA). It is assumed that the investigated sequences had an evolutionary relationship (a common ancestor). The simultaneous alignment of many nucleic acids or amino acid sequences is one of the most commonly used techniques in sequence analysis.

MSA was performed by using dynamic programming methods (49, 64) (no more than three sequences due to the computation requirements of the method) (28), heuristics (22, 62), Carrillor and Lipman Algorithm (12) or its modification (45).

Multiple alignments are used to predict the secondary or tertiary structure of new sequences (37); to analyze homology (60); to construct phylogenetic trees (67), to find protein

families (48); and to suggest primers for polymerase chain reaction (PCR) (66).

**Genetic Algorithm Characteristics**

Genetic Algorithms (GAs) are adaptive heuristic search algorithms based on the evolutionary ideas of natural selection and genetics. GAs are designed to simulate the natural processes required for evolution, especially those which follow the "soft inheritance" principle of Jean-Baptiste Lamarck (42) and the "survival of the fittest" principle of Charles Darwin (15). In nature, the individuals' competition for scanty resources results in the fittest individuals dominating over the weaker ones.

Genetic algorithms are implemented as computer simulations in which a population of abstract representations (chromosomes or genotypes) of candidate solutions (individuals, creatures, or phenotypes) is subject to an optimization problem in order for better solutions to be obtained. GAs simulate the survival of the fittest among consecutive generations of individuals for solving a problem. Each generation consists of a population of character strings analogous to the DNA chromosomes. Each individual represents a point in a search space and a possible solution. The individuals in the population evolve. GAs are based on an analogy with the genetic structure and behaviour of chromosomes within a population of individuals.

There are many variants and adaptations of GAs in order to improve performances for a given type of problem. The following are examples of using GAs for solving hard problems in biological sciences: ant colony optimization (8), bacteriologic algorithms (6), the cross-entropy method (18), cultural algorithms (40), evolution strategies (58), evolutionary programming (23), extremal optimization (2), Gaussian adaptation (39), genetic programming (4), memetic algorithm (61), hybrid search (17), etc.

A sample of a given size of chromosomes (entry 1 in **Table 1**) must be generated in order to use a classical GA for solving a problem. A GA must have an evaluation function in order to assess chromosome fitness and assign it a value. The GA iterates as follows:

◊ Repeat:
- Step1: Select two chromosomes (sometimes according to their fitness - better fitness followed by better selection chances) by using a probability mass function - entry 1, **Table 1**;
- Step2: Crossover the parents by using a probability mass function and create offspring - entry 2, **Table 1**;
- Step3: Mutate the offspring by using a probability mass function - entry 3, **Table 1**;
- Step4: Add the offspring to the sample;
- Step5: Assess the fitness of the new members using the evaluation function;
- Step6: Delete one or more members from the sample based on their fitness by using a probability mass function (steady-state selection is applied);

◊ Until the best fitness of a sample member satisfies the end condition.

Encoding, crossover and mutation are presented in **Table 1**. Selection and fitness are shown in **Table 2**.

Other related approaches include support vector machines (10), rough sets (34), SPLASH (11), or probabilistic relational models (59).

**TABLE 1**

Encoding, crossover and mutation in genetic algorithms

| Operator | Example | | Comments |
|---|---|---|---|
| Encoding | Chromosome_1 | U A G G A G | Encode two chromosomes (U, A, G, C are the genes here) |
| | Chromosome_2 | C G G G A A | |
| Crossover | Offspring_1 | **C G** G G A G | Select crossover points (here are 0 and 2) then interchange genes |
| | Offspring_2 | **U A** G G A A | |
| Mutation | Offspring_3 | C G **C A** A G | Mutate offspring randomly (A into C and GG into CA) |
| | Offspring_4 | U **C** G G A A | |

**TABLE 2**

Fitness and selection in genetic algorithms

| Method | Fitness score | Selection | Comments |
|---|---|---|---|
| Proportional | $f_i$=Fitness(Chromosome_i) | $p_i=f_i/\Sigma_i f_i$ | The chance of reproduction is proportional to the fitness (using the probability) |
| Deterministic | | $i \mid f_i$ = max. or min. | The best or worst individuals are reproduced (elitism) |
| Tournament | | $(f_i,f_j)$ | Pairs of individuals compete for selection |
| Normalization | $g_i=(f_i-N_0)(f_{max.}-f_{min.})/(N_1-N_0)$ | $p_i=g_i/\Sigma_i g_i$ | A fixed scale $[N_0,N_1]$ normalizes fitness between different generations |
| Ranking | $h_i$=Rank($f_i$)($f_{max.}-f_{min.}$)/Size | $p_i=h_i/\Sigma_i h_i$ | The reproduction chance is proportional to the fitness rank |

As far as sequence analysis is concerned, the objective function (a measure of overall alignment quality) is not used to demonstrate that one alignment is preferred over another or that the best possible alignment, given a set of parameters, was found. Therefore, progressive alignment (55), which provides two main alternatives, could be used:

- ◊ Hidden Markov models (41, 46) simultaneously find an alignment and a probability model of substitutions, insertions and deletions that are most self consistent;
- ◊ Objective functions (OFs) measure multiple alignment quality and find the best scoring alignment (19, 27). This approach has a further advantage: it may be used to optimise any OF. The alignments can be evaluated using an OF, which is a measure of multiple alignment quality (**Table 3**).

The OF must deal with the following issues when used to solve a gene sequence alignment problem (see **Table 3**):

- ◊ Matches and gaps: two objectives - maximizing matches and minimizing gaps. A match may have a different biological relevance (weight) than a gap.
- ◊ Sequence length: matches (and gaps) increase as sequence lengths increase.
- ◊ Sequence shifts: shifting of a sequence will produce gaps at the beginning and end of the aligned sequences; these gaps must be treated separately.

There are many different approaches to constructing an OF. Karlin and Altschul (38) presented four types of scores:

1. Based on charges: not all amino acids present the same partial (or apparent) charge in a given environment (such as in blood serum or muscle cells). Charge values may be obtained by averaging the values of an experiment; alternatively, the pK (or pK-7) value (acid dissociation constant) may be used.
2. Based on matches of a given amino-acid (e.g. A in **Table 3**).
3. Derived from target frequencies: different weights match different amino acids;
4. Based on structure alphabets: when amino acids are partitioned into classes (such as internal, external and ambivalent).

Classical GAs are slightly changed in order to solve a specific problem. Thus, Notredame and Higgins (50) reported a software package called SAGA (Sequence Alignment by Genetic Algorithm) that uses a scheduling scheme to control the usage of 22 different operators for combining alignments or mutating them between generations. They implemented the cost of a multiple alignment (A) as a linear superposition of costs between pairs of aligned sub-sequences as OF:

$$OF(A) = \sum_{i=2}^{N} \sum_{j=1}^{i-1} W_i \cdot W_j \cdot Cost\ (A_i, A_j)$$

where $W_i$ and $W_j$ are weights of the $A_i$ and $A_j$ sub-sequences (in sequences); the Cost(·,·) function includes gap opening and extension penalties for opening and extending the gaps.

Altschul (1) made an extensive review describing the different ways of scoring gaps in a multiple alignment. Two related questions derived from sequence alignment:

1. Is the alignment significant according to certain statistical models?
2. How stable is the alignment? (Which are the alternative alignments with similar alignment scores?)

The first question is related with the probability of observing any particular alignment solely by chance. This difficult problem has solutions under certain conditions (65). The second question regards alignment interpretation (results obtained by Vingron and Argos) (63).

When there is additional information (e.g. the secondary structure of one protein from two aligned chains) the complexity of the problem decreases. Such alignments include non-local interactions and the solution proved to be a hard problem (43). Under these conditions, the objective functions must take into account this new challenge. Corpet and Michot (14) proposed an OF with two position-specific gap penalties: GOS (penalty for opening a gap between two stacked pairs); GO (penalty for opening a gap in non-structured regions), and GEP (penalty for the gap length). Corpet and Michot (14) suggested the following predefined weights: GO=5, GOS=8, GEP=0.3, and computed the total gap penalty as:

$$GapP(A)=a(A)\cdot GOS + b(A)\cdot GO + c(A)\cdot GEP$$

where *a* is the number of gaps between stacked pairs in stems, *b* is the number of other non-terminal gaps and *c* is the total length of all non-terminal gaps. The alignment score (the OF) is calculated as follows:

$$OF(A) = Pr(A) + \lambda\cdot Se(A) - GapP(A)$$

where Pr(·) is a function of the aligned pairs of residues in the alignment, Se(·) is based on the secondary structure and it evaluates the stability of the folding induced by the master in the slave sequence. Parameter λ (positive constant) balances the contribution of primary and secondary structure information.

**TABLE 3**

Example of gene sequence alignment

| | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Two unaligned sequences | Sequence_1 | U | A | A | G | C | C | U | C | A | G | U | A | A |
| | Sequence_2 | A | A | C | C | C | U | C | A | U | A | | | |
| | | | | | | | | | | | | | | |
| A possible alignment of the sequences | Sequence_1 | U | **A** | **A** | G | **C** | **C** | **U** | **C** | **A** | G | **U** | **A** | A |
| | Sequence_2 | | **A** | **A** | C | **C** | **C** | **U** | **C** | **A** | | **U** | **A** | |

Notredame et al. (50) implemented the model for RNA sequence alignment proposed by Corpet and Michot (14) and observed that optimization was very difficult for $\lambda>0$ (the secondary structure was taken into account). Notredame et al. (50) reported good results using a *Homo sapiens* mitochondrion (X03205 and V00702) as protein with known structure and a mitochondrion from different species (*Drosophila virilis* X05914, *Apis mellifera* S51650, *Penicillium chrysogenum* L01493, etc.) as protein with unknown secondary structure. They showed that the best value for $\lambda$ parameter varied from 1 (*Oxytrichia nova* X03948, *Latimeria chalumnae* Z21921, *Xenopus laevis* M27605) to 6 (*Saccharomyces cerevisiae* V00702) for the best pair matching resulting from the reference alignment that varied from 66.6% to 84.9% in nine experiments (with an average statistics of 79.3±4.6% at 95% confidence).

**Software Applications**

Parsons et al. (52) developed and implemented a genetic algorithm for solving a DNA sequence assembly problem. The fragments were ordered by using a sorted order representation. Two fitness functions based on pairwise overlap strengths were implemented and tested. The first fitness function aimed to maximize the sum of overlap strengths in adjacent fragments. The second fitness function aimed to minimize the function described by Churchill et al. (13). The performances of the fitness functions were comparable; however, neither function appeared to represent the desired layout appropriately. The GA implementation suggested by Parsons et al. is a modified GA previously implemented by Grefenstettle (30). The GA implemented by Parsons et al. was better than the GA implemented by Huang (32).

Moore et al. (47) developed and applied a maximum-likelihood (ML) and Bayesian search using 61 plastid protein-coding genes on five major lineages of *mesangiosperms* for 45 taxa. A genetic algorithm was applied in order to perform rapid heuristic ML searches (the GARLI program) (68). For Bayesian searches, they used MrBayes 3 program (54) which implements a variant of Markov Chain Monte Carlo (MCMC) called Metropolis-Coupled MCMC (25). Huelsenbeck et al. (33) suggested that the Metropolis-Coupled MCMC was the most useful numerical method for approximating the posterior probability of a tree. The estimated ML parameters presented by Moore et al. (47) were assessed using the nonparametric BS approach (3). The phylogenetic tree analysis carried out by Moore et al. (47) revealed that GARLI estimated parameters were always extremely close to the fully optimized values.

Pan et al. (51) developed GABRIEL (Genetic Analysis by Rules Incorporating Expert Logic), which is a rule-based system (including similarity, pattern, and proband based rules) designed to apply domain-specific and procedural knowledge systematically and uniformly in order to analyse and interpret data from DNA micro arrays. The effects of serum addition on the biology of human fibroblasts (29, 36, 44, 56) were used to analyse a dataset of 517 genes. The results revealed altered transcription in human foreskin fibroblasts following the addition of serum to growth-arrested cultures previously published by Iyer et al. (35). Pan et al. (51) used pattern-based rules to obtain the setting of the following parameters (not explicitly defined by Iyer et al.) (35): elevated, baseline, immediately, remained, and short period. Pan et al. (51) showed that the elevation required at least a 2-fold change in the gene expression of each time points, and a baseline zone between -1 and +1 (expressed as logarithms). They specified the immediate/early (I/E) response gene using a decision tree with time periods ranging from 15 min to 1 h (51). Pattern search analysis (PSA) was conducted using GAs (GABRIEL software) in order to detect data organized according to the interrelationships among component parts in gene expression profiles (data sets portraying the features of gene expression under specified conditions). PSA studies were able to reconstruct the results previously reported by Iyer et al. (35). Furthermore, when the continuity-proband rule was used (GABRIEL), additional continuities not found by Iyer et al. (35) were detected through the analysis of hierarchical clustering dendrograms. Pan et al. (51) remarked that the GA was able to distinguish between expression profiles with subtle differences not readily apparent by the visual scanning of data. Moreover, in cases where the results differed from the ones reported by Iyer et al. (35), the GABRIEL rule explanation function indicated the statistical or threshold parameters responsible for the differences. Iyer et al. (35) suggests that the key features of GABRIEL may be useful for analysing large data sets generated by other types of genomic and proteomic approaches.

## Conclusions

Genetic Algorithms can be implemented in a straightforward manner to solve hard problems derived from gene sequence analyses (sequence alignment, sequence databases, repeated sequence search, sequence comparisons). Recent advances in the integration of genetic algorithms with routines for maximum-likelihood estimation, Markov chain Monte Carlo simulations, and rules incorporating expert logic approaches proved able to investigate and explain hard questions of gene sequence analysis.

## Acknowledgments

The research was partly supported by national research grants (ID1051/UEFISCSU, ID0458/UEFISCSU, PCCP1177/CNMP).

## REFERENCES

1. **Altschul S.F.** (1989) J. Theor. Biol., **138**(3), 297-309.
2. **Bak P., Sneppen K.** (1993) Phys. Rev. Lett., **71**(24), 4083-4086.
3. **Baldwin B.G., Sanderson M.J.** (1998) Proc. Natl. Acad. Sci. USA, **95**(16), 9402-9406.

4. **Banzhaf W., Nordin P., Keller R.E., Francone F.D.** (1997) Genetic Programming: An Introduction: On the Automatic Evolution of Computer Programs and Its Applications, Morgan Kaufmann Publishers, San Francisco, p. 450.
5. **Barricelli N.A.** (1954) Methodos, 45-68.
6. **Benoit B., Fleurey F., Jézéquel J.-M., Le Traon Y.** (2005) IEEE Software, **22**(2),76-82.
7. **Bosworth J., Norman F., Zeigler B.P.** (1972) Comparison of Genetic Algorithms with Conjugate Gradient Methods, NASA Contractor Reports, CR-2093.
8. **Bouktir T., Slimani L.** (2005) Leonardo J. Sci., **4**(7), 43-57.
9. **Bremermann H.J, Rogson J., Salaff S.** (1966) Global properties of evolution processes. In: Natural Automata and useful Simulations (H.H. Pattee, Ed.), Proceedings of Symposium on Fundamental Biological Models, Stanford University 1965, pp. 3-42.
10. **Brown M.P., Grundy W.N., Lin D., Cristianini N., Sugnet C.W., Furey T.S., Ares M.Jr., Haussler D.** (2000) Proc. Natl. Acad. Sci. USA, **97**(1), 262-267.
11. **Califano A.** (2000) Bioinformatics, **16**(4), 341-357.
12. **Carrillo H., Lipman D.** (1988) SIAM J. Appl. Math., **48**, 1073-1082.
13. **Churchill G., Burks C., Eggert M., Engle M.L., Waterman M.S.** (1993) Assembling DNA sequence fragments by shuffling and simulated annealing, Tech. Rep. LA-UR-93-2287, Los Alamos Scientific Laboratory Publication, LA-UR-2287, p. 25.
14. **Corpet F., Michot B.** (1994) Comput. Applicat. Biosci., **10**(4), 389-399.
15. **Darwin C.R.** (1859) On the origin of species by means of natural selection, J. Murray, London, p. 459.
16. **Davis L.** (1987) Genetic Algorithms and Simulated Annealing, M. Kaufmann, San Francisco, p. 216.
17. **Davis L.** (1991) The Handbook of Genetic Algorithms, VN Reinhold, New York, p. 385.
18. **De Boer P.-T., Kroese D.P., Mannor S., Rubinstein R.Y.** (2005) Ann. Oper. Res., **134**(1), 19-67.
19. **Do C.B., Mahabhashyam M.S.P., Brudno M., Batzoglou S.** (2005) Genome Res., **15**(2), 330-340.
20. **Durbin R., Eddy S., Krogh A., Mitchison G.** (2002) Biological sequence analysis. Probabilistic models of proteins and nucleic acids, 7th Ed., Cambridge University Press, Cambridge, UK, p. 356.
21. **Falkenauer E.** (1998) Genetic Algorithms and Grouping Problems, Wiley, New York, p. 220.
22. **Feng D., Doolittle R.F.** (1987) J. Mol. Evol., **25**, 351-360.
23. **Fogel L.J.** (1999) Intelligence Through Simulated Evolution: Forty Years of Evolutionary Programming, Wiley Interscience, New York, p. 162.
24. **Fraser A.** (1957) Aust. J. Biol. Sci., **10**, 484-491.
25. **Gilks W.R., Roberts G.O.** (1996) Strategies for improving MCMC, In: Markov Chain Monte Carlo in Practice: Interdisciplinary Statistics (W.R. Gilks, S. Richardson, D. Spiegelhalter, Eds.), Chapman & Hall, London.
26. **Glover F.** (1977) Decision Sci., **8**(1), 156-166.
27. **Gondro C., Kinghorn B.P.** (2007) Genet. Mol. Res., **6**(4), 964-982.
28. **Gotoh O.** (1986) J. Theor. Biol., **121**, 327-337.
29. **Greenberg M.E., Ziff E.B.** (1984) Nature, **311**(5985), 433-438.
30. **Grefenstette J.J.** (1984) Genesis: A system for using genetic search procedures. In: Proceedings of a Conference on Intelligent Systems and Machines, Rochester, MI, 161-165.
31. **Holland J.H.** (1975) Adaptation in Natural and Artificial Systems, Univ. of Michigan Press, Ann Arbor, p. 183.
32. **Huang X.** (1992) Genomics, **14**, 18-25.
33. **Huelsenbeck J.P., Ronquist F., Nielsen R., Bollback J.P.** (2001) Science, **294**(5550), 2310-2314.
34. **Hvidsten T.R., Komorowski J., Sandvik A.K., Laegreid A.** (2001) Pac. Symp. Biocomput., **6**, 299-310.
35. **Iyer V.R., Eisen M.B., Ross D.T., Schuler G., Moore T., Lee J.C., Trent J.M., Staudt L.M., Hudson J.Jr., Boguski M.S., Lashkari D., Shalon D., Botstein D., Brown P.O.** (1999) Science, **283**(5398), 83-87.
36. **Jähner D., Hunter T.** (1991) Mol. Cell. Biol., **11**(7), 3682-3690.
37. **Joshi R.R.** (2007) Current Bioinformatics, **2**(2), 113-131.
38. **Karlin S., Altschul S.F.** (1990) Proc. Natl. Acad. Sci. USA, **87**(6), 2264-2268.
39. **Kjellström G.** (1991) J. Optim. Theor. Appl., **71**(3), 589-597.
40. **Kobti Z., Reynolds R.G., Kohler T.** (2004) SwarmFest 8(online), p. 8.
41. **Krogh A., Brown M., Mian I.S., Sjolander K., Haussler D.** (1994) J. Mol. Biol., **235**(5), 1501-1531.
42. **Lamarck J.B.P.A.** (1830) An Exposition of Zoological Philosophy, JB Baillère, Paris, p. 420 and p. 450 (in french).
43. **Lathrop R.H.** (1994) Protein Engng., **7**(9), 1059-1068.
44. **Lau L.F., Nathans D.** (1987) Proc. Natl. Acad. Sci. USA, **84**(5), 1182-1186.
45. **Lipman D.J., Altschul S.F., Kececioglu J.D.** (1989) Proc. Natl. Acad. Sci USA, **86**(12), 4412-4415.
46. **Löytynoja A., Goldman N.** (2005) Proc. Natl. Acad. Sci. USA, **102**(30), 10557-10562.
47. **Moore M.J., Bell C.D., Soltis P.S., Soltis D.E.** (2007) Proc. Natl. Acad. Sci. USA **104**(49), 19363-19368.
48. **Mulder N.J., Apweiler R.** (2008) Curr. Protoc. Bioinformatics, **Suppl. 21**, 2.7.1-2.7.18.
49. **Murata M., Richardson J.S., Sussman J.L.** (1985) Proc. Natl. Acad. Sci. USA, **82**, 3073-3077.

50. **Notredame C., O'Brien E.A., Higgins D.G.** (1997) Nucleic Acid Res., **25**(22), 4570-4580.
51. **Pan K.-H., Lih C.-J., Cohen S.N.** (2002) Proc. Natl. Acad. Sci. USA, **99**(4), 2118-2123.
52. **Parsons R., Forrest S., Burks C.** (1993) Genetic Algorithms for DNA Sequence Assembly, In: ISMB-93 Proceedings, 310-318.
53. **Rechenberg I.** (1973) Evolutionsstrategies - Optimierung technischer Systeme nach Prinzipien der biologischen Information, Frommann-Holzboog Verlag, Stuttgart.
54. **Ronquist F., Huelsenbeck J.P.** (2003) Bioinformatics, **19**(12), 1572-1574.
55. **Russell D.J., Otu H.H., Sayood K.** (2008) BMC Bioinformatics, **9**, Article no. 306.
56. **Ryder K., Lau L.F., Nathans D.** (1988) Proc. Natl. Acad. Sci. USA, **85**(5), 1487-1491.
57. **Schefel H.-P.** (1981) Numerical Optimization of Computer Models, John Wiley and Sons Ltd, New York, p. 398.
58. **Schwefel H.-P.** (1995) Evolution and Optimum Seeking, Wiley & Sons, New York, p. 456.
59. **Segal E., Taskar B., Gasch A., Friedman N., Koller D.** (2001) Bioinformatics, **17**(S1), 243-252.
60. **Shevchenko A., Valcu C.-M., Junqueira M.** (2009) J. Proteomics, **72**(2), 137-144.
61. **Smith J.E.** (2007) IEEE Trans. Syst. Man. Cy. B, **37**(1), 6-17.
62. **Taylor W.R.** (1986) J. Mol. Biol., **188**, 233-258.
63. **Vingron M., Argos P.** (1990) Protein Engng., **3**(7), 565-569.
64. **Waterman M.S.** (1984) Bull. Math. Biol., **46**, 473-500.
65. **Waterman M.S., Vingron M.** (1994) Proc. Natl. Acad. Sci. USA, **91**(11), 4625-4628.
66. **Yuan J.S., Burris J., Stewart N.R., Mentewab A., Neal Jr. C.N.** (2007) BMC Bioinformatics, **8**(Suppl. 7), Article no. S6.
67. **Zheng X., Qin Y., Wang J.** (2009) Math. Biosci., **217**(2), 159-166.
68. **Zwickl D.J.** (2006) Genetic algorithm approaches for the phylogenetic analysis of large biological sequence datasets under the maximum likelihood criterion. Ph.D. dissertation, The University of Texas at Austin.