World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:1, No:5, 2007

Multiple Sequence Alignment Using Optimization Algorithms

M. F. Omar, R. A. Salam, R. Abdullah, N. A. Rashid

Abstract**—**Proteins or genes that have similar sequences are likely to perform the same function. One of the most widely used techniques for sequence comparison is sequence alignment. Sequence alignment allows mismatches and insertion/deletion, which represents biological mutations. Sequence alignment is usually performed only on two sequences. Multiple sequence alignment, is a natural extension of two-sequence alignment. In multiple sequence alignment, the emphasis is to find optimal alignment for a group of sequences. Several applicable techniques were observed in this research, from traditional method such as dynamic programming to the extend of widely used stochastic optimization method such as Genetic Algorithms (GAs) and Simulated Annealing. A framework with combination of Genetic Algorithm and Simulated Annealing is presented to solve Multiple Sequence Alignment problem. The Genetic Algorithm phase will try to find new region of solution while Simulated Annealing can be considered as an alignment improver for any near optimal solution produced by GAs.

Keywords**—**Simulated Annealing, Genetic Algorithm, Sequence alignment, Multiple Sequence Alignment.

I. Introduction

ACCORDING to Luscombe [1], bioinformatics can be defined as conceptualizing biology in terms of macromolecules (in the sense of physical-chemistry) and then applying "informatics" techniques (derived from disciplines such as applied mathematics, computer science, and statistics) to understand and organize the information associated with these molecules, on a large-scale. In short, bioinformatics leverage the techniques borrowed from computer science to solve problems in molecular biology.

This exciting area is a new field, and the pace of research is driven by the large and rapidly increasing amount of data being produced for example, efforts to sequence the genomes of a variety of organisms. The areas where computer science can be applied range from assembly of sequence fragments, analysis of DNA, RNA and protein sequences, prediction and analysis of protein sequence and function, and the analysis and simulation of general metabolic function and regulation [1], [2].

Manuscript received April 21, 2004.

M. F. Omar is with Universiti Utara Malaysia, Perlis, Malaysia. (ijatz@yahoo.com).

R. A. Salam is with Universiti Sains Malaysia, 11800 Penang, Malaysia. (Tel: +60-4-6532486; Fax: +60-4-6573335; email: rosalina@cs.usm.my).

R. Abdullah is with Universiti Sains Malaysia, 11800 Penang, Malaysia. (email: rosni@cs.usm.my).

N. A. Rashid is with Universiti Sains Malaysia, 11800 Penang, Malaysia. (email: nuraini@cs.usm.my).

One of the branches in bioinformatics is sequence similarity which is a subset of sequence analysis. Molecular sequence data is a rich source of knowledge capable of teaching us about the structure, function and evolution of biological macromolecules. It is vital to drug engineers and pharmacists to understand molecular sequence analysis. Sequence analysis is assisted by searching the similarity between the given sequences.

Proteins or genes that have similar sequences are likely to perform the same function. Proteins are the building blocks for all cells while DNA stores all genetic information. The primary structure of a protein is a linear chain of amino acids. There are twenty amino acids [3], denoted by *A*, *R*, *N*, *D*, *C*, *Q*, *E*, *G*, *H*, *I*, *L*, *K*, *M*, *F*, *P*, *S*, *T*, *W*, *Y*, and *V*. DNA molecules are chains of nucleotides. There are four different types of nucleotides [4], denoted by *A*, *T*, *G*, *C* . Therefore, both proteins and DNA molecules can be represented as strings of letters from relatively small alphabets.

Multiple Sequence Alignment (MSA), is an extension of two-sequence/pairwise sequence alignment [5]. Nowadays, multiple sequence alignment is an important tool in molecular biology and it provides key information for sequence analysis. There are several uses of MSA; finding sequence to determine patterns that characterize protein/gene families; detecting homology between new sequences and known protein/gene family sequences; predicting secondary and tertiary structures of new protein sequences; predicting function of new sequences and molecular evolutionary analysis [6].

As the name suggests, in multiple sequence alignment, we would like to find an optimal alignment for a collection of sequences. The following section will briefly define the problem of multiple sequence alignment.

The problem of MSA falls in the class of NP-Complete problem [5], [7]. It is necessary to clearly define the sequence alignment problem. Let say, there is a set of string where the number of string is *k*. So, given *k* string $S= \{S_1,S_2,S_3,\ldots\ldots S_k\}$, we try to find an optimal alignment for those sequences. That is to say, the purpose is to find $S' =\{S'_1,S'_2,\ldots\ldots S'_k\}$ in optimal way such that [5]:

- S'_i is an extension of S_1 by inserting or padding gaps/spaces,
- $\forall i,j\ : |S'_i| = |S'_j|$, and

- $\Sigma_i\Sigma j$ *sim(S'₁, S'ⱼ)* is maximized or $\Sigma_i\Sigma j$ *cost(S'i, S'ⱼ)* is minimized where *sim(X,Y)* and *cost(X,Y)* are some sequence similarity and alignment cost functions defined for sequences *X* and *Y*.

In MSA, we have to define a scoring scheme for evaluating matching letters, mismatching letters and gap penalties. The representation of the sequences must also be taken into consideration. The goal of sequence alignment is to produce an alignment such that the total score is optimal. The basic rule is, we can insert gaps at any positions in the sequences, but the order of sequence must be preserved.

There are two types of alignment [6], global and local. In global alignment, attempts are made to detect the best alignment of the entire sequences. In local alignment, the best alignment is constructed for segments of sequences with the highest density of matches, while the rest of the sequences are ignored. In this paper, only global alignment was investigated. So, from this section onwards, the word alignment will refer to global alignment.

Before we go further, let us examine some mathematical terminology adapted from Martin Tompa lecture notes [5]. Assuming that these are two sequences;

```
Seq1 a  c  -  -  b  c  d  b
Seq2 -  c  a  d  b  -  d  -
```

Special character "-" denote the insertion of a space, representing deletion from its sequence or insertion in the other sequence. The scoring function of aligning is $\sigma(x,y)$ where x and y are each single character or space. For any two distinct characters x and y, $\sigma(y,y)= +2$ and $\sigma(x,y) = \sigma(-,y) = \sigma(x,-) = -1$. From a scoring scheme we can calculate the scoring function for both sequences → 3.(2) + 5.(-1) = 1.

If S is a string, then $|S|$ denotes the ith character of S. For example, if S = acbcbd, then $|S|$ = 6 and $S|3|$ = b. Let say we have string S and T. An *alignment A* maps S into strings S' and T' that may contain space characters, where $|S'| = |T'|$, and the removal of spaces from S' and T'(without changing the order of remaining characters) leaves S and T respectively. The value of alignment A is

$$.\sum_{i=1}^{l} \sigma(S'[i], T'[i]) .$$

where $l = |S'| = |T'|$ In the example above, if S = acbcdb and T=cadbd, then S'=ac--bcdb and T'= -cadb-d-. The above alignment is called pairwise alignment. The remaining part of this study will use the same mathematical terminology to depict the function used in MSA.

II. Motivation

There exist three categories of optimization algorithm for multiple alignment [8]; exact, progressive and iterative. Numerous MSA programs have been applied using many techniques and algorithms. Most commonly used techniques are progressive and iterative techniques. The exact method suffers from inexact sequence alignment and lead to an aggressive research on progressive and iterative algorithms.

A. *Exact*

One of the algorithm that use exact method is MSA program based on Carillo and Lipman algorithm that make it possible to align up to ten closely related sequence [9]. This algorithm use slower and upper bounds tighter than guaranteed ones. However, these methods suffer a major drawback that is, it is not guaranteed to reach mathematical optimum [11]. The limitation of MSA program has been solved when Stoye described a new divide and conquer algorithm [11]. Divide and conquer heuristically cut the sequences at the right points, so that the produced alignment remain as close as possible to optimal.

B. *Progressive*

Progressive alignment constitutes one of the simplest ways to align sequence. Basically, most progressive alignment methods heavily rely on dynamic programming to perform multiple alignment starting with the most related sequences and then progressively adding less related sequences to the initial alignment. The existence of several progressive program has broadened up the aligning techniques. This approach has the advantages of speed and simplicity [8]. However the major problem with progressive alignment method is that errors in the initial alignments are the most closely related sequence propagated to the multiple alignment [8].

In the context of pairwise sequence comparison, dynamic programming alignment is generated by starting at the ends of two sequences and attempt to match all possible pairs of characters between the sequences. This is followed by a scoring scheme for matches, mismatches, and gaps. The representation of sequence used in pairwise alignment is in the form of matrices. Initially, a scoring table is constructed in order to fill in the scoring value in each i column and j row. Dynamic Programming fills in the values for V(i,j) from top to bottom and left to right.

Another aspect in programming is how to recover the alignment. The solution to retrace steps back from the (n,m) entry, determining which preceding entries were responsible for the current one. The concept in n-dimensional remain the same when there is a start path through a lattice until it reach the end point.

ClustalW is a straightforward progressive alignment strategy where sequence are added one by one to the multiple alignment according to the order indicated by a pre-computed dendogram. The main shortcoming of this strategy is that once sequence has been aligned, alignment will never be modified even if it conflicts with the sequence added later.

C. *Iterative*

Iterative alignment methods depend on algorithms that are able to produce an alignment and to refine through a series of cycles(iterations) until no more improvement can be made [8]. Iterative methods can be deterministic or stochastic, depending on the strategy used to improve the alignment. The simplest iterative strategies are deterministic. It involves extracting the sequence one by one from multiple alignments and realigning them to the remaining sequences. This procedure is terminated when no improvement can be made(convergence). Stochastic iterative methods include Hidden Markov Model(HMM), simulated annealing and evolutionary computation such as genetic algorithms(GAs) and evolutionary programming. Their main advantage is to allow for a good separation between the optimization process and evaluation criteria [8]. It is the objective function that defines the aim of any optimization procedure.

D. Genetic Algorithms

In iterative multiple sequence alignment, Genetic Algorithms(GAs) show a promising result in sequence alignment. Another way is to hybrid GA with other method such as dynamic programming. Dynamic programming as an alignment improver. Some of them outperform ClustalW with an impressing result.

GAs are adaptive methods which may be used to solve search and optimization problems. They are based on the genetic process of biological organisms [20]. Since 1960s, there has been increasing effort to mimic living being to develop powerful algorithms for difficult optimization problem [19]. GA are adaptive methods which may be used to solve search and optimization problems. GA are inspired by the mechanism of natural selection where stronger individuals are likely the winners in a competing environments. It is powerful and broadly applicable stochastic search and optimization techniques and is perhaps the most widely known types of evolutionary computation methods today. In general, a GA has five basic components [19]:

- A genetic representation of solution to the problem
- A way to create an initial population of solutions
- An evaluation function rating solutions in terms of their fitness
- Genetic operators that alter the genetic composition of children during the reproduction
- Values for the parameters of genetic algorithms

To successfully apply GA on MSA, one would need to map the potential solutions (i.e. alignments) into a representation/data structure that could be easily manipulated (in terms of recombination and mutation). On top of that, a proper evaluation function or objective function should be defined, which in this case is related to the overall alignment score of the sequences. In one sense, we can see that GA for MSA somewhat model a brute force approach of trying numerous possible solution in a somewhat guided-randomized manner. Examples of GA-based methods to align multiple sequences are such as SAGA [22], Isokawa [23], GA and Dynamic Programming [24]. However, SAGA [22] and Isokawa [23] use different representations.

E. Simulated Annealing

Simulated annealing(SA) is a generalization of the Monte Carlo method for examining the equations of state and frozen states of n-body systems [25]. The concept is based on the manner in which liquids freeze or metals recrystalise in the process of annealing. During the annealing state, process will melt and initially it will start at high temperature and it will slowly cooled down so that the system will always be at thermodynamic equilibrium. As cooling proceeds, the system becomes more ordered and approaches a "frozen" ground state at T=0. Hence the process can be thought of as an adiabatic approach to the lowest energy state. If the initial temperature of the system is too low or cooling is done insufficiently, slowly the system may become quenched forming defects or freezing out in meta stable states (i.e. trapped in a local minimum energy state) [25].

In the original Metropolis scheme was that an initial state of a thermodynamic system was chosen at energy E and temperature T. Holding T constant, the initial configuration is perturbed and the change in energy, dE is computed. If the change in energy is negative the new configuration is accepted. If 1the change in energy is positive it is accepted with a probability given by the Boltzmann factor exp -(dE/T) [21]. These processes are repeated for a number of times to obtain good sampling statistics for the current temperature. The temperature is decremented and the entire process was repeated until a frozen state is achieved at T=0.

By analogy the generalization of this Monte Carlo approach to combinatorial problems is straight forward [25]. The current state of the thermodynamic system is analogous to the current solution to the combinatorial problem, the energy equation for the thermodynamic system is analogous to the objective function, and ground state is analogous to the global minimum. The major problem in implementing the algorithm is that there is no obvious analogy for the temperature T with respect to a free parameter in the combinatorial problem. Furthermore, avoidance of entrainment in local minima is dependent on the "annealing schedule", the choice of initial temperature, how many iterations are performed at each temperature, and how much the temperature is decremented at each step as cooling proceeds. An example that uses SA [13] works by iteratively improving a new multiple sequence alignment calculated using the mode.

III. Methodology

The proposed new methodology for multiple sequence alignment is shown in Fig. 1. This system is a hybrid system of Genetic Algorithm and Simulated Annealing. There are six modules in the system starting with Initialize Population, Evaluate Population Structure, Selection Reproduction, Operators and Clean Gap, Selection Replace and Alignment Improver. This section will describe each of the module in more detail.

A. System Module

Input File

Initially, a data file in FASTA format is read. Sequences in FASTA formatted files are preceded by a line starting with >. The first word on this line is the name of the sequence. The rest of the line is a description of the sequence. The remaining lines contain the sequence itself. Blank lines in a FASTA file are ignored, and so are spaces or other gap symbols (dashes, underscores, and periods) in a sequence.

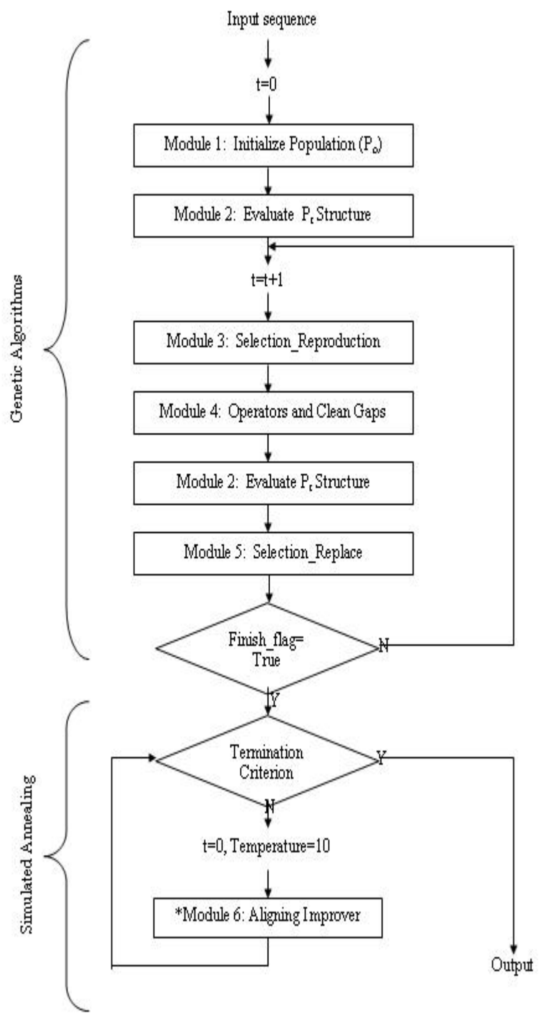

Input sequence
t=0
Module 1: Initialize Population (P0)
Module 2: Evaluate Pt Structure
t=t+1
Module 3: Selection_Reproduction
Module 4: Operators and Clean Gaps
Module 2: Evaluate Pt Structure
Module 5: Selection_Replace
Finish_flag= True
N
Y
Termination Criterion
Y
N
t=0, Temperature=10
*Module 6: Aligning Improver
Output
Genetic Algorithms
Simulated Annealing

FIGURE 1
SYSTEM OVERVIEW.

Fasta files containing multiple sequences are just the same, with one sequence listed right after another. This format is accepted by many multiple sequence alignment programs. A description of the FASTA format can be found at http://www.ncbi.nlm.nih.gov/BLAST/fasta.html. There are several websites that offer proteomic and genomic data in FASTA format. The data can be accessed at http://www.people.virginia.edu/~wrp/cshl99/mcclure-seqs and also at http://rsdb.csie.ncu.edu.tw/tools/msa.htm.

Module1: Initialize Population.

Objective: To populate solution randomly.

Description:

This module generates a random solution. Each individual in population represent *S'*, a sequence with gaps (-) inserted randomly.

Representation

The population is represented as an array of sequence where each sequence was encoded as an array of character over the alphabet *A*, *R*, *N*, *D*, *C*, *Q*, *E*, *G*, *H*, *I*, *L*, *K*, *M*, *F*, *P*, *S*, *T*, *W*, *Y*, and *V* as a protein sequence. The symbol "-" will refer to the gap in the alignment which represent an insertion or a deletion of an amino acid residue. Each sequence has their own length. Therefore there must be a mechanism to limit the length of the column. Let say we have a set of sequence S = $\{S_1,S_2,S_3,\ldots\ldots S_n\}$. So from here we can limit the column by w = $(1.5 * \max \{|S_1|,|S_2|,|S_3|,\ldots\ldots|S_n|\})$ meaning that we will restore some free column in order to add/delete gaps.

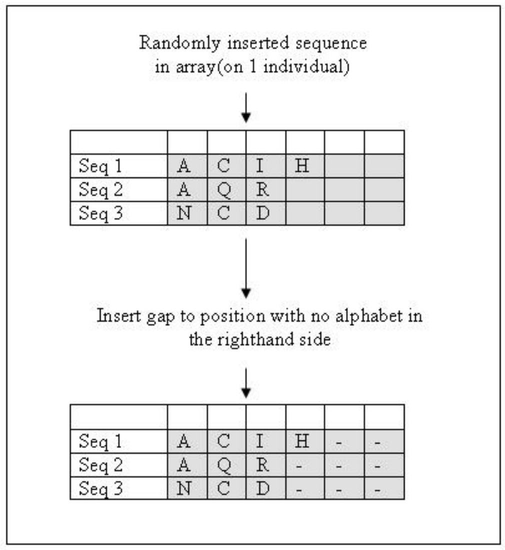

Randomly inserted sequence in array(on 1 individual)
Seq 1 A C I H
Seq 2 A Q R
Seq 3 N C D
Insert gap to position with no alphabet in the righthand side
Seq 1 A C I H - -
Seq 2 A Q R - - -
Seq 3 N C D - - -

FIGURE 2
POPULATION INITIALIZATION (FOR 1 INDIVIDUAL).

Population Initialization

The population of the initial parent alignment matrices was generated via random initialization. All the sequences, S_n, are restored in an array randomly. All positions that are not associated with amino acid were filled with a gap. This is shown in Fig. 2. The size of population is decided by user. By default the population size is 10.

Module2: Evaluate P_t Structure.

Objective: Evaluate and assign a scoring function to each individual in population.

Description:

Each individual in a population will have a scheme of scoring function. Individual that scores a high fitness function (F) will survive for the next iteration. The Scoring function is as follows:

$$. \text{ Cost} = \sum_{i=1}^{l} \sigma(\, S'[i],\, T'[i]) \,.$$

where $l = |S'| = |T'|$, $\sigma(x,x)=0$ and $\sigma(x,y)= \sigma(-,y)\ \sigma(x,-)=1$. This means that each similarity between amino acid will get 0 point (no cost). If there is a difference in the alphabet, the score will be –1. Fitness Value must be in the range 0 –100%.

Module3: Selection_Reproduction.

Objective: To select two chromosomes (individual) which have the best fitness function.

Description:

This module chooses two individuals for mating process. The selection probability for each individual is proportional to the fitness function value. In this case, the fitter the individual, the more likely it will be chosen at random (i.e. the better scoring function, the better chance to be pick up). Below are the steps taken for selecting two individuals for mating buffer.

- Compute selection probabilities for the current population based on fitness value.
- Select two individual randomly based on the selection probabilities to obtain clones which may then be subjected to mutation or recombination.

Module4: Operators and Clean Gap

Objective: To explore new region of solution and eliminate unused gaps

Description:

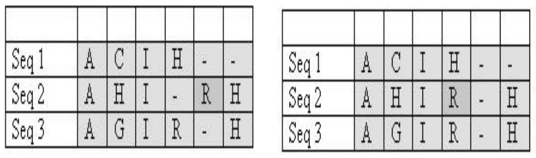
Crossover

The crossover operator will use point-to-point crossover that uses one of SAGA operator where the operator take two alignment genomes from the population and randomly select a fully matched (no gap) column. This operator will try to swap gaps with amino acids, such that the offspring produce a feasible sequence.

Fig. 3 shows a one-point cross over. The crossover point will be selected randomly. After crossover, Child 1 and Child 2 are evaluated. The fittest offspring will survive in the next iteration.

Mutation

This is the background operator. The mutation operator picks a random amino acid from a randomly chosen row (sequence) in the alignment and checks whether one of its neighbors has a gap. If this is the case, the algorithms swaps (2-opt) the selected amino acid with a gap neighbor. If both neighbors are gaps, one of them will be picked randomly. This is shown in Fig. 4.



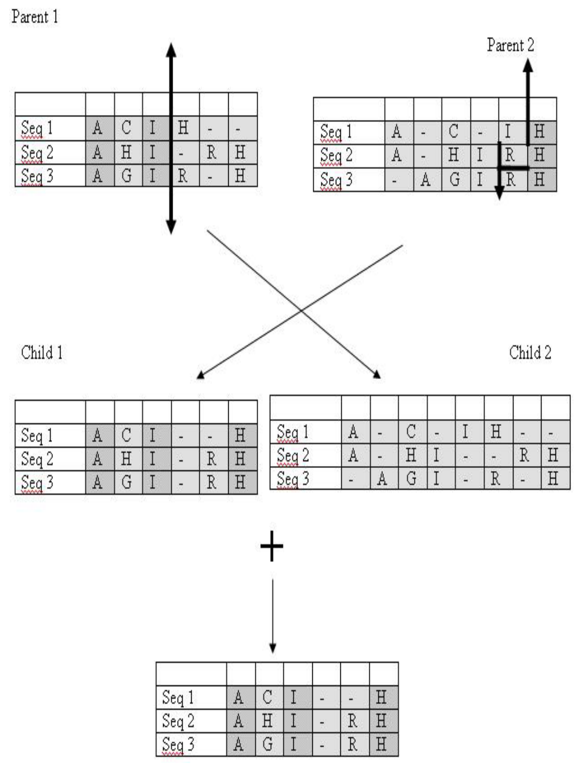


FIGURE 3
ONE POINT CROSSOVER.

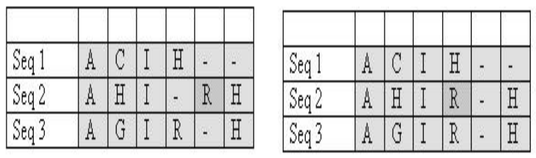
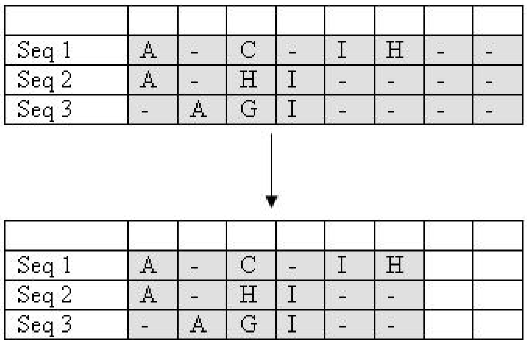


FIGURE 4
2 – OPT MUTATION.

CleanGap

This operator will clean up the gap on the right hand side since they are disrupting the current alignment. So, with the new alignment, it is easy to evaluate the new individuals. This is shown in Fig. 5



Seq 1	A	-	C	-	I	H	-	-
Seq 2	A	-	H	I	-	-	-	-
Seq 3	-	A	G	I	-	-	-	-

Seq 1	A	-	C	-	I	H		
Seq 2	A	-	H	I	-	-		
Seq 3	-	A	G	I	-	-		

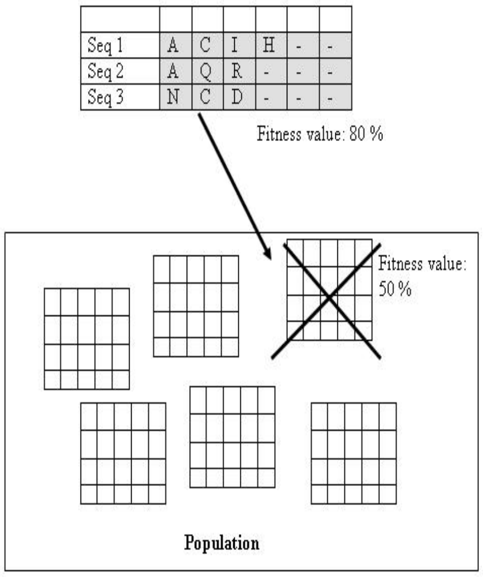
FIGURE 5.
CLEAN GAP – CLEAN UNWANTED GAPS

Module5: Selection_Replace

Objective: Replace old chromosome with new offspring

Description:

This module will replace old chromosome that have less fitness function and insert new offspring to the population. Let say, if we have a crossover, a new offspring will be chosen based on fitness function. This new offspring will replace the old individual that have the least fitness value in population. This can be seen in Fig. 6



Seq 1	A	C	I	H	-	-
Seq 2	A	Q	R	-	-	-
Seq 3	N	C	D	-	-	-

FIGURE 6
SELECTION_REPLACE: REPLACE THE LEAST FITNESS INDIVIDUAL WITH A NEW INDIVIDUAL.

Module6: Aligning Improver

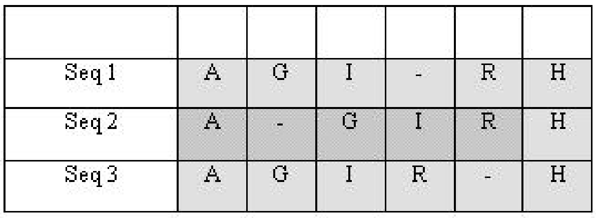
Objective: Improve alignment quality from a single solution produced from GA (if required).

Description:

If the fitness value is less than a threshold value (e.g. <70% fitness). The program will proceed to simulated annealing phase to align a solution produced from the GA phase. This technique does not maintain population. From here, Simulated Annealing will use the same representation and the same fitness function used in GA phase. One main reason of using *Aligning Improver* (SA phase) is to avoid local minima. In the hybrid system of GA/DP mentioned earlier, it produces average results. This is due to the problem of the progressive method in Dynamic Programming that gets easily trapped in local minima. If this happens then the process cannot be repeated once it is at the middle of the progressive alignment. This can be overcome with the use of SA.

Neighborhood Structure

The rule in performing the structure is to preserve the old solution and try to improve it from there. In simulated annealing there is a need to describe the neighborhood structure. This is shown in Fig. 7. Initially, we will pick a random acid amino with gaps from a random row. Let say we have N= {A – G I R}. We will start at a high "temperature," where the temperature is the SA parameter that mimics the effect of a fast moving particle in a hot object like a hot molten metal, thereby permitting the ball to make very high bounces and being able to bounce over any mountain to access any valley, given enough bounces. As the temperature is made relatively colder, the ball cannot bounce so high, and it can also settle to become trapped in relatively smaller ranges of valleys. In this case we use t = 10.



Seq 1	A	G	I	-	R	H
Seq 2	A	-	G	I	R	H
Seq 3	A	G	I	R	-	H

FIGURE 7
NEIGHBORHOOD STRUCTURE.

Performing Swaps

Performing swaps is shown in Fig. 8. Data were taken from Fig. 7 Assuming N = {A – G I R}, five different solutions can be produced from the swapping process. This is shown below:

N= { A – G I R }.

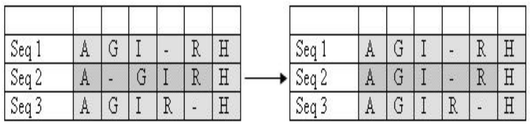
Solution0 = A – G I R

Solution1 = – A G I R

Solution2 = A G – I R

Solution3 = A G I – R

Solution4 = A G I R –



Seq 1	A	G	I	-	R	H
Seq 2	A	-	G	I	R	H
Seq 3	A	G	I	R	-	H

Seq 1	A	G	I	-	R	H
Seq 2	A	G	I	-	R	H
Seq 3	A	G	I	R	-	H

FIGURE 8
PERFORMING SWAPS.

Fig. 8 shows a simple complete example. If seq1 is {A G I – R H}, swapping will remain the first sequence and it will start to process the second sequence. The second sequence chooses solution3 which is the same as seq1. The next step is to follow the next generated solution which is solution4. Therefore, the new seq3 is {A G I R - H}.

Downhill Move

SA's is based on energy minimization. Energy in SA's represents the cost consumed in timetabling problem. Usually, when changes were accepted, the cost is lower than the previous cost. This step is called downhill move. Firstly, differences in energy (dEnergy) between the current solution and the old solution are calculated. If the dEnergy is greater than zero, it is either the move is accepted or it will choose the downhill.

Uphill Move

If the probability is lower than the value of X ($0<X<1$) then the changes (go uphill) were accepted, otherwise changes were rejected.

Cooling Schedule

The annealing schedule determines the degree of uphill movement permitted during the search and is thus critical to the algorithm's performance. The principle underlying is the choice of a suitable annealing schedule is easily stated: the initial temperature should be high enough to *melt* the system completely and should be reduced towards its *freezing point* as the search progresses. Initially, `COOLING_RATE = 0.9` is declared. At the end of the iteration, the temperature will be cooled down using the following formula:

. t = COOLING_RATE * t .

where `t` is the temperature.

IV. Implementation and Results

Visual Basic and Microsoft XP were used as a platform for the implementation. Fig. 9 shows the interface of the prototype. Firstly, the sequence file is read and kept in an array. This is shown in Fig. 10.

Several tests were performed for evaluating and comparing purposes. The implementation used the protein data sets used by Horg [24], that can be accessed at http://rsdb.csie.ncu.edu.tw/tools/msa.htm. These are protein data sets that were extracted from Swiss-Prot 39.16.

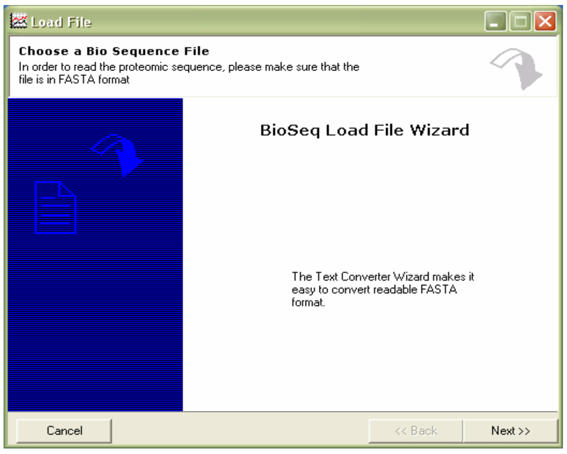
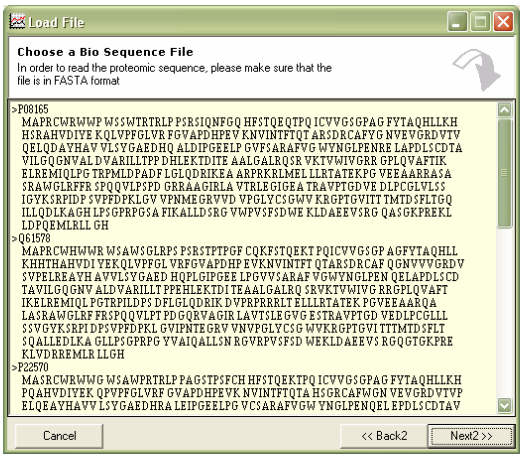

Load File
Choose a Bio Sequence File
In order to read the proteomic sequence, please make sure that the file is in FASTA format
BioSeq Load File Wizard
The Text Converter Wizard makes it easy to convert readable FASTA format.
Cancel
<< Back
Next >>

FIGURE 9
CHOOSING A SEQUENCE FILE.


Load File
Choose a Bio Sequence File
In order to read the proteomic sequence, please make sure that the file is in FASTA format
Cancel
<< Back2
Next2 >>

```
>P08165
  MAPRCWRWWP WSSWTRTRLP PSRSIQNFGQ HFSTQEQTPQ ICVVGSGPAG FYTAQHLLKH
  HSRAHVDIYE KQLVPFGLVR FGVAPDHPEV KNVINTFTQT ARSDRCAFYG NVEVGRDVTV
  QELQDAYHAV VLSYGAEDHQ ALDIPGEELP GVFSARAFVG WYNGLPENRE LAPDLSCDTA
  VILGQGNVAL DVARILLTPP DHLEKTDITE AALGALRQSR VKTVWIVGRR GPLQVAFTIK
  ELREMIQLPG TRPMLDPADF LGLQDRIKEA ARPRKRLMEL LLRTATEKPG VEEAARRASA
  SRAWGLRFFR SPQQVLPSPD GRRAAGIRLA VTRLEGIGEA TRAVPTGDVE DLPCGLVLSS
  IGYKSRPIDP SVPFDPKLGV VPNMEGRVVD VPGLYCSGWV KRGPTGVITT TMTDSFLTGQ
  ILLQDLKAGH LPSGPRPGSA FIKALLDSRG VWPVSFSDWE KLDAEEVSRG QASGKPREKL
  LDPQEMLRLL GH
>Q61578
  MAPRCWHWWR WSAWSGLRPS PSRSTPTPGF CQKFSTQEKT PQICVVGSGP AGFYTAQHLL
  KHHTHAHVDI YEKQLVPFGL VRFGVAPDHP EVKNVINTFT QTARSDRCAF QGNVVVGRDV
  SVPELREAYH AVVLSYGAED HQPLGIPGEE LPGVVSARAF VGWYNGLPEN QELAPDLSCD
  TAVILGQGNV ALDVARILLT PPEHLEKTDI TEAALGALRQ SRVKTVWIVG RRGPLQVAFT
  IKELREMIQL PGTRPILDPS DFLGLQDRIK DVPRPRRRLT ELLLRTATEK PGVEEAARQA
  LASRAWGLRF FRSPQQVLPT PDGQRVAGIR LAVTSLEGVG ESTRAVPTGD VEDLPCGLLL
  SSVGYKSRPI DPSVPFDPKL GVIPNTEGRV VNVPGLYCSG WVKRGPTGVI TTTMTDSFLT
  SQALLEDLKA GLLPSGPRPG YVAIQALLSN RGVRPVSFSD WEKLDAEEVS RGQGTGKPRE
  KLVDRREMLR LLGH
>P22570
  MASRCWRWWG WSAWPRTRLP PAGSTPSFCH HFSTQEKTPQ ICVVGSGPAG FYTAQHLLKH
  PQAHVDIYEK QPVPFGLVRF GVAPDHPEVK NVINTFTQTA HSGRCAFWGN VEVGRDVTVP
  ELQEAYHAVV LSYGAEDHRA LEIPGEELPG VCSARAFVGW YNGLPENQEL EPDLSCDTAV
```

FIGURE 10
SEQUENCE FILE IS READ AND PUT IN AN ARRAY.

Altogether, 8 protein datasets were used. Similarity measures for all the three stages, that are the pre-alignment, GA, GA and Simulated Annealing is shown in Table 1. Similarity can be defined as:

. *Similarity = totally match column/maxLength* .

The crossover rate is 70%. It will exchange a region from child 1 to another region from child 2 (vice versa).While mutation will try to swap (2-opt) gaps and characters within one child. Mutation rate=1%. If there is no crossover or mutation, the child will be copied exactly from its parent.

TABLE I
LIST OF PROTEIN IDS AND THEIR SIMILARITY MEASURE FOR PRE-ALIGNMENT.

Protein ID	*Pre-Alignment*	*Genetic Algorithm*	*GA/ Simulated Annealing*
P1	0.0192	0.25	0.28
P2	0.0020	0.56	0.56
P3	0.6462	0.74	0.74
P4	0.0019	0.4	0.4
P5	0.0028	0.5	0.5
P6	0.0049	0.63	0.63
P7	0.0022	0.422	0.422
P8	0	0.32	0.32

As can be seen from the table, the aligning improver that is Simulated Annealing does not show significant improvement on our algorithm. This is because the local minima which sometimes occur during the alignment process do not happen with the test data we used. Whether the local minima occur or not is determined by the threshold value that we set. The threshold value is 70%. However, we still proceed with SA even though the threshold value is greater than 70%. The main reason of using SA is to avoid local minima. However, our test data were not sufficient. A more comprehensive test needs to be carried out to prove our algorithm. It will require a larger dataset to prove that SA will produce better results.

V. CONCLUSIONS

Multiple Sequence Alignment relies very much on optimization algorithms. The combination of Genetic Algorithm and Simulated Annealing is a new way that can be used to solve MSA assignment. Genetic Algorithm will try to find new region of feasible solution while Simulated Annealing will act as aligning improver. Simulated annealing also helps to prevent local minima problem compared to the Dynamic Programming.

Experiments were conducted just by selecting a few data sets. Further tests need to be carried out to prove that the use of SA can produce better results. The current experiments showed that GA itself is sufficient to solve the problem. However, this is not true since SA can avoid local minima.

There are other aspects of the system that need to be improved. The cooling schedule needs to be tested thoroughly to get better results. Another factor is due to the gap insertion process, where a new operator can increase the performance of the system. Our future work will focus more on the simulated annealing stage and also to introduce a new operator for the gap insertion stage.

REFERENCES

[1] N. M. Luscombe, D. Greebaum, M. Gerstein, (2001). *What is bioinformatics? A proposed definition and overview of the field.* Department of Molecular Biophysics and Biochemistry, Yale University, USA.

[2] R. Musick, T. Slezak, T. Crithlow, (2001). An Overview of Bioinformatics Research at Lawrence Livermore National Laboratory. Joint Genome Institute / Computing Application Organization .Lawrence Livermore National Laboratory.

[3] *Online Lectures on Bioinformatics* (September 2003) [Online], Available: http://lectures.molgen.mpg.de/Biol/index.html.

[4] P. C. Turner, A. G. McLennan, A. D Bates, M. R. H. White, (1997). Nucleic Acid Structure. *Instant Notes in Molecular Biology* C1:31-35. Bios Scientific Publishers Limited, Liverpool.

[5] M. Tompa .(2000). *Lecture Notes on Biological Sequence Analysis*. Technical Report #2000-06-01.Department of Computer Science and Engineering, University of Washington.

[6] W. Zhong(2003). *Using Travel Salesman Problem Algorithms To determine Multiple Sequence Alignment Orders*. Master Thesis, University of Georgia: Athens, Georgia.

[7] C. Karostensky, G. Gonnet. *Near Optimal Multiple Sequence Alignment using a Traveling Salesman Problem approach*. Swiss Federal Institute of Technology, Institute of Scientific Computing. ETH Zurich, Switzerland (2000).

[8] R. Choudry, (1999). *Application of Evolutionary Algorithms for Multiple Sequence Alignment*. Stanford University.

[9] D. J. Lipman, S. F. Altschul, J. D. Kececioglun, (1989). *A tool for Multiple Sequence Alignment*. Vol. 86.pp 4412-4415, Biochemistry. Proc. Natl. Acad. Sci. USA.

[10] M. Gribskov, J. Devereux, (1991). *Similarity and Homology*. Sequence Analysis Primer.3: 89-157. Stockton Press. NY.

[11] C. Notredame, (2002). *Recent Progress in Multiple Sequence Alignment: A Survey.*, Pharmacogenomics, 3(1):131-144.

[12] D. Higgins, (1999). *Multiple Sequence Alignment.* Genetics Databases.9:165-183 Academic Press, London, UK .

[13] J. D. Thompson, F. Plewniak, and O. Poch. (1999). *A comprehensive comparison of multiple sequence alignment programs*. Nucleic Acids Research, 27(13):2682–2690.

[14] *Introduction to hidden markov models* (June 2003) [Online], Available:www.scs.leeds.ac.uk/scsonly/teachingmaterials/HiddenMarkovModels/html dev/main.html.

[15] *Introduction to HMM* (April 2003) [Online] Available: http://www.cs.brown.edu/research/ai/dynamics/tutorial/Documents/HiddenMarkovModels.html.

[16] S. R. Eddy, (1998). *Profile Hidden Markov Models*. Bioinformatics Review. Vol. 14. 9: 755-763. Oxford University Press.

[17] P. Baldi, S. Brunak, Y. Chauven, A. Krogh (1997). *Hidden Markov Model For Human Genes: Periodic Pattern in Exon Sequence*. Theoretical and Computational Methods in Genome Research, 2: 15-32. Plenum Press, New York.

[18] S. R. Eddy, (1996). *Multiple Alignment Using Hidden Markov Models*. Dept. of Genetics, Washington University School of Medicine St. Louis, US.

[19] M. Gen, R. Cheng. (2000). *Genetic Algorithms and Engineering Optimization*. John Wiley & Sons: Canada.

[20] D. Beasley, D. R. Bull, R. R. Martin, (1993). *An Overview of Genetic Algorithms: Part 1, Fundamentals*. Inter-University Committee on Computing. 15(2) 58-69. [21] *CCS 501 Neural Network and Genetic Algorithm* http:// office1.cs.usm.my.

[21] C. Notredame, and D. G. Higgins, (1996). *SAGA: sequence alignment by genetic algorithm*, *Nuc. Acids Res., 24(8), 1515-1524.*

[22] M. Isokawa, M. Wayama, T. Shimizu, (1996). *Multiple Sequence Alignment using Genetic Algorithm*. Department of Information Science, Faculty of Science, Hirosaki University, Japan.

[23] J. T. Horng, C. N. Lin, B. H. Yang, and C. Y. Kao, (2001). *A genetic algorithm for multiple sequence alignment*. In *Poster* in German Conference on Bioinformatics.

[24] S. Kirkpatrick, C. D. J. Gellat, and M. P. Vecchi, (1983). *Optimization by Simulated Annealing*, Science, 220 pp. 671-680.

Mohd Faizal Omar was born in Penang, Malaysia on 22nd of June 1979. He received his Bachelor degree in Computer Science in 1999 from Universiti Utara Malaysia and his Masters degree in Computer Science in 2003 from Universiti Sains Malaysia.

He is currently a lecturer at Universiti Utara Malaysia and his research work is in the area of Artificial Intelligence.

Rosalina Abdul Salam was born in Penang, Malaysia on 13th of January 1968. She received her Bachelors degree in Computer Science in 1992 from Leeds Metropolitan University, United Kingdom. She received a scholarship to pursue her Masters degree and PhD. She received her Masters degree in Software Engineering from Sheffield University, United Kingdom in 1997. She completed her PhD in 2001 from Hull University in the area of artificial intelligence and image processing.

She was a system analyst in Intel Penang, from 1992 to 1995. She is currently a lecturer of the School of Computer Sciences, Universiti Sains Malaysia and a member of Artificial Intelligence Research Group. She has published more than 25 papers in journals and conferences. Her current research area is in the area of artificial intelligence, image processing and bioinformatics applications.

Rosni Abdullah was born in Penang, Malaysia on 3rd of June 1962. She received her Bachelors degree in Computer Science and Applied Mathematics and Masters degree in Computer Science from Western Michigan University, Kalamazoo, Michigan, U.S.A. in 1984 and 1986 respectively. She joined the School of Computer Sciences, Universiti Sains Malaysia in 1987 as a lecturer. She received an award from the university in 1993 to pursue her PhD at Loughborough University in United Kingdom in the area Parallel Algorithms.

She was promoted to Associate Professor in 2000. She has held several administrative positions such as First Year Coordinator, Programme Chairman and Deputy Dean for Postgraduate Studies and Research. She has published more than 50 papers in journals and conferences. She is currently the Dean of the School of Computer Sciences, Universiti Sains Malaysia and also Head of the Parallel and Distributed Processing Research Group. Her current research work is in the area of parallel algorithms for bioinformatics.

Nuraini Abdul Rashid was born in December 1963 in Singapore. She received her Bachelors degree in Computer Science from Mississippi Stat University, USA in 1985 and her Masters in Computer Science from Universiti Sains Malaysia in 1995.

She is a lecturer of the School of Computer Sciences, Universiti Sains Malaysia. She has published more than 20 papers in journals and conferences. She is currently doing her PhD in the area of Bioinformatics. Her interest is in Parallel Algorithms for Biological Pattern Recognition.