Report Number: WUCS-99-09

1999-01-01

# Pattern Matching Techniques and Their Applications to Computational Molecular Biology - A Review

Eric C. Rouchka

Pattern matching techniques have been useful in solving many problems associated with computer science, including data compression (Chrochemore and Lecroq, 1996), data encryption (RSA Laboratories, 1993), and computer vision (Grimson and Huttenlocher, 1990). In recent years, developments in molecular biology have led to large scale sequencing of genomic DNA. Since this data is being produced in such rapid fasion, tools to analyze DNA segments are desired. The goal here is to discuss various techniques and tools for solving various pattern matching questions in computational biology, including optimal sequence alignment, multiple sequence alignment, and buidling models to describe sequence families... **Read complete abstract on page 2.**

# Pattern Matching Techniques and Their Applications to Computational Molecular Biology - A Review

Eric C. Rouchka

Complete Abstract:

Pattern matching techniques have been useful in solving many problems associated with computer science, including data compression (Chrochemore and Lecroq, 1996), data encryption (RSA Laboratories, 1993), and computer vision (Grimson and Huttenlocher, 1990). In recent years, developments in molecular biology have led to large scale sequencing of genomic DNA. Since this data is being produced in such rapid fasion, tools to analyze DNA segments are desired. The goal here is to discuss various techniques and tools for solving various pattern matching questions in computational biology, including optimal sequence alignment, multiple sequence alignment, and buidling models to describe sequence families using Hidden Markov Models (HMMs) and regular expressions.

# Pattern Matching Techniques and Their Applications to Computational Molecular Biology - A Review

Eric C. Rouchka
WUCS-99-09

March 4, 1999

Department of Computer Science
Washington University
Campus Box 1045
One Brookings Drive
Saint Louis, MO 63130-4899

Institute for Biomedical Computing
Washington University
700 S. Euclid Avenue
Saint Louis, MO 63110

ecr@ibc.wustl.edu

# Pattern Matching Techniques and Their Applications to Computational Molecular Biology -- A Review

**Eric C. Rouchka**
Institute for Biomedical Computing
Washington University
700 South Euclid Avenue
St. Louis, MO 63110-1012, USA
Email: ecr@ibc.wustl.edu

### Abstract

Pattern matching techniques have been useful in solving many problems associated with computer science, including data compression (Chrochemore and Lecroq, 1996), data encryption (RSA Laboratories, 1993), and computer vision (Grimson and Huttenlocher, 1990). In recent years, developments in molecular biology have led to large scale sequencing of genomic DNA. Since this data is being produced in such rapid fashion, tools to analyze DNA segments are desired. The goal here is to discuss various techniques and tools for solving various pattern matching questions in computational biology, including optimal sequence alignment, multiple sequence alignment, and building models to describe sequence families using Hidden Markov Models (HMMs) and regular expressions.

**Keywords:** Optimal sequence alignment, multiple sequence alignment, pattern matching, dynamic programming, Hidden Markov Models

## Introduction

Pattern detection problems have their roots in many specific computer science fields. Included among these are voice recognition, handwriting recognition, object recognition, and sequence/subsequence detection.

Pattern matching questions in computational biology arise from the desire to know different characteristics about DNA sequences. One problem attempts to find the best way to align two biological sequences. Another involves finding any common subsequences, such as promoters or functional motifs, within a given set of sequences. A third problem attempts to determine how well a sequence fits into a given pattern. A final question involves the construction of a model to describe a given set of functionally similar sequences.

We attempt to discuss several methods to answer these questions. The solutions presented develop statistical pattern matching methods, create models on untrained data, maintain data structures, and use some form of induction.

## Globally Aligning Two Sequences

The most widely used approach in aligning two sequences of biological importance is to use a dynamic programming technique (Needleman and Wunch, 1970; Sellers, 1974; Smith and Waterman, 1981). Dynamic programming incorporates a recursive algorithm using a two-dimensional score matrix with $M+1$ columns and $N+1$ rows, where $M$ and $N$ are the lengths of the two sequences to be compared. Figure 1 shows an initial score matrix. There are two stages involved in aligning two sequences using dynamic programming. The first is a matrix fill step that calculates an alignment score. The second is a traceback that recovers the optimal alignment. Such an approach can be thought of as an alignment graph finding the cost of a shortest source-to-sink path, where the source is located in the upper left and the sink is located in the lower right.

| | | G | A | A | T | T | C | A | G | T | T | A |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| G | 0 | | | | | | | | | | | |
| G | 0 | | | | | | | | | | | |
| A | 0 | | | | | | | | | | | |
| T | 0 | | | | | | | | | | | |
| C | 0 | | | | | | | | | | | |
| G | 0 | | | | | | | | | | | |
| A | 0 | | | | | | | | | | | |

**Figure 1: Initial score matrix.** Shown in this figure is the initial score matrix for aligning the sequence GGATCGA to the sequence GAATTCAGTTA. Note that the first row and first column are initially filled with zeros.

Calculation of the score matrix using a symbol-based scheme takes $O(MN)$ time. Let $C$ be the alignment score matrix. Position $C_{i,j}$ contains the best alignment between $a_1..a_i$ and $b_1..b_j$ where $A=a_1...a_M$ is the first sequence and $B=b_1..b_N$ is the second sequence. For symbol-based schemes, there are three edges that reach any given position. These correspond to substitution events that model aligned pairs of residues, deletion events which

model gaps in the input sequence $A$, and insertion events that model gaps in the sequence $B$. A scoring function $\delta$ is incorporated to score the following functions:

$\delta(a,b)$: scores an alignment of symbol $a$ with symbol $b$
$\delta(\varepsilon,b)$: scores an alignment of a gap with symbol $b$
$\delta(a,\varepsilon)$: scores an alignment of symbol $a$ with a gap

The cost of reaching any point in the matrix $C$ is calculated as the minimum of a substitution event, deletion event, or insertion event. This is captured in Equation 1. Figure 2 shows the filled score matrix using one particular scoring scheme.

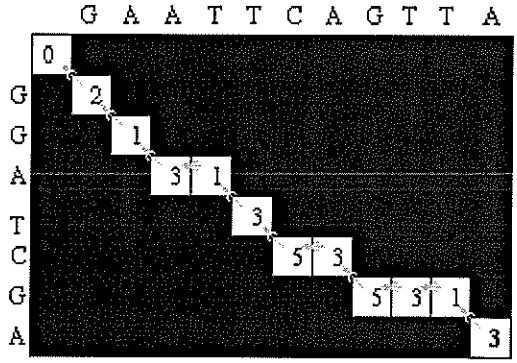
$$C_{i,j} = \min(C_{i-1,j-1} + \delta(a_i, b_j), \\ C_{i-1,j} + \delta(a_i, \varepsilon), \\ C_{i,j-1} + \delta(\varepsilon, b_j))$$

**Equation 1: Score matrix equation for symbol-based scheme.** The score matrix is calculated using the minimum of a substitution event, deletion event, or insertion event.

| | | G | A | A | T | T | C | A | G | T | T | A |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| G | 0 | 2 | 0 | -1 | -1 | -1 | -1 | -1 | 2 | 0 | -1 | -1 |
| G | 0 | 2 | 1 | -1 | -2 | -2 | -2 | -2 | 1 | 1 | -1 | -2 |
| A | 0 | 0 | 4 | 3 | 1 | -1 | -3 | 0 | -1 | 0 | 0 | 1 |
| T | 0 | -1 | 2 | 3 | 5 | 3 | 1 | -1 | -1 | 1 | 2 | 0 |
| C | 0 | -1 | 0 | 1 | 3 | 4 | 5 | 3 | 1 | -1 | 0 | 1 |
| G | 0 | 2 | 0 | -1 | 1 | 2 | 3 | 4 | 5 | 3 | 1 | -1 |
| A | 0 | 0 | 4 | 2 | 0 | 0 | 1 | 5 | 3 | 4 | 2 | 3 |

**Figure 2: Final score matrix.** Shown in this figure is the final score matrix for aligning the sequence GGATCGA to the sequence GAATTCAGTTA. A symbol-based scheme is used with the scoring function as follows: if $a=b$, $\delta(a,b) = 2$, otherwise $\delta(a,b) = -1$; $\delta(a,\varepsilon) = \delta(\varepsilon,b) = -2$. The arrows indicate the predecessor from which the score in each box is derived.

After the score matrix has been filled, the traceback algorithm determines the actual alignment(s) that result in the maximum score. When a simple scoring scheme is used, there will likely be multiple maximal alignments. The traceback step begins in the lower right-hand corner of the matrix. In this step, all neighbor cells that could be direct predecessors are considered. For the symbol-based scheme, you can consider the neighbor to the left (a gap in the second sequence has occurred), the diagonal neighbor (a match or mismatch), or the neighbor above (a gap in the first sequence has occurred). The arrows in Figure 2 indicate the possible predecessor nodes. The traceback algorithm chooses as the next cell in the sequence one of the possible predecessors from which the current score can be derived. Once the cell in the upper left-hand corner has been reached, traceback is finished. Figure 3 indicates one possible traceback route leading to a maximal alignment using the scoring scheme described in Figure 2.




```
G A A T T C A G T T A
|   |   | |   |     |
G G A _ T C _ G _ _ A
```

**Figure 3: Optimal sequence alignment.** This figure indicates one possible sequence of events that leads to the maximum alignment score between GGATCGA and GAATTCAGTTA.

### Allowing Concave Gap Penalties

The problem with symbol-based schemes is they don't weight gaps based on length. Many short, isolated gaps may occur when clustered gaps are desired. Knight and Myers (1992) present a solution to this problem using a concave cap scheme. Concave gap penalties give preference to longer gaps by not penalizing adding onto a gap (gap extension) as much as creating a new gap (gap initiation). In this scheme, the cost of a gap, $\omega$, is a function of its length, $k$. The general form of a concave gap penalty is $\omega(k) = \alpha + \beta * f(k)$ where $f(k)$ is a concave function. One example is $\omega(k) = \alpha + \beta \log k$ where $\alpha,\beta > 0$.

For concave gap schemes, the dynamic programming matrix can be filled out in $O(MN(M+N))$ time. This is due to the fact that for each insertion or deletion event, all nodes in a column or row must be considered. Knight and Myers (1992) reduce this to time of $O(MN(\log M + \log N))$ by using the notion of "Minimum Envelopes".

Minimum envelopes reduce the search space by considering the deletion (insertion) terms over a whole column (row). For any point in the dynamic programming

matrix, all positions in a column are considered as a possible minimum deletion path. Rather than computing this each time for each position in a column, deletion terms for the whole column are stored. This is used to compute a minimum envelope.

Minimum envelopes contain a list of possible candidates for the minimum contribution in a column according to a transformed version of the curve $\alpha+\omega(\beta+x)$ where $\alpha=C_{k,j}$; $\beta= i - k$; $i$ is the current column, $j$ is the current row, and $k$ is calculated for each column. This is done similarly for the insertion elements of each row. Minimum envelopes are revisited when matching a sequence to a regular expression.

### Affine Gap penalties

Affine gap models are a specific example of using a concave gap scoring function (Fitch and Smith, 1983). For the affine gap model, the penalty is assigned according to the function $\omega(k) = \alpha + \beta k$ where $\alpha,\beta > 0$. Calculation of sequence alignment using affine gap penalties can be found in $O(MN)$ time. This is accomplished by using an additional parameter at each position in the score matrix which contains the length of the gap to that point.

## Locally Aligning Multiple Sequences

Some biological problems deal with discovering how multiple sequences relate to one another. Detection of protein binding sites or promoter regions is one specific application. Information measures are applied in expectation/maximization (EM) algorithms (Stormo and Hartzell, 1989; Lawrence and Reilly, 1990; Cardon and Stormo, 1992) to detect the most likely alignment between sequences. The basis behind EM algorithms is to compute maximum likelihood estimates from missing or incomplete data (Dempster, et. al., 1977).

EM algorithms can be useful in detecting locally conserved regions (motifs). However, since the algorithms involve a maximization step, the solutions provided can fall into the pitfall of winding up in a local maximum solution without ever considering the global maximum.

### Gibbs Sampling Approach

A sampling strategy known as Gibbs Sampling (Lawrence, et. al., 1993) has been devised to help alleviate the local maximum problem. Gibbs Sampling is a stochastic iterative technique used to detect and align locally conserved regions (motifs) in multiple sequences of amino acids or nucleic acids.

Gibbs Sampling maintains a pattern description, or profile, of the nucleic acid or amino acid composition of each position in the motif as well as the frequencies as they occur in the remaining sequence. The goal is to find the most probable pattern by maximizing the ratio of the corresponding pattern probability to background probability.

There are three basic steps in the Gibbs sampler: initializing motif positions, predictive update, and sampling. To initialize the motif positions, random starting positions for the motif must be chosen in each sequence. Figure 4 indicates an initial motif alignment.

TCAGAACCAGTTATA**AATTTA**TCATTTCCTTCTCCACTCCT
CCCACGCA**GCCGCC**CCTCCTCCCCGGTCACTGACTGGTCCTG
TCGACCCTCTGAACCTATCAGGGACCA**CAGTCA**GCCAGGCAAG
AAAACACTTGAG**GGAGCA**GATAACTGGGCCAACCATGACTC
GGGTGAATGGTACTGCT**GATTAC**AACCTCTGGTGCTGC
AGCCTAGAGT**GATGAC**TCCTATCTGGGTCCCCAGCAGGA
GCCTCAGGATCCAGCACACAT**TATCAC**AAACTTAGTGTCCA
CATTATCAC**AAACTT**AGTGTCCATCCATCACTGCTGACCCT
TCGGAACAAGGCAAA**GGCTAT**AAAAAAAATTAAGCAGC
GCCCCTTCCCCA**CACTAT**CTCAATGCAAATATCTGTCTGAAACGGTTC

**Figure 4: Initial Motif Alignment.** The bases in blue indicate an initial random motif alignment for a six-base motif. The motif contained within these sequences is a six-base GATA-1 binding site with the consensus (T/A)GATA(A/G) (Ominchinski, JG., et. al., 1995; Orkin, 1995).

Once the motif positions are initialized, Gibbs Sampling iterates through the predictive update and sampling steps. In the predictive update step, a sequence is chosen at random. The motif element is taken out of that sequence, and the motif and background profiles are updated. In the sampling step, the probability for each possible motif starting position in the sequence chosen in the predictive update step is calculated. One of the positions is sampled back into the model based on the normalized probabilities. After the predictive update and sampling steps have been iterated through several times, an alignment results. Figure 2 illustrates an example of a final alignment.

TCAGAACCAGTTATAAAT**TTATCA**TTTCCTTCTCCACTCCT
CCCACGCAGCCGCCCTCCTCCC**CGGTCA**CTGACTGGTCCTG
TCGACCCTCTGAAC**CTATCA**GGGACCACAGTCAGCCAGGCAAG
AAAACACTTGAGGGAGC**AGATAA**CTGGGCCAACCATGACTC
GGGTGAA**TGGTAC**TGCTGATTACAACCTCTGGTGCTGC
GCCTAGAGTGATGACTC**CTATCT**GGGTCCCCAGCAGGA
GCCTCAGGATCCAGCACACA**TTATCA**CAAACTTAGTGTCCA
A**TTATCA**CAAACTTAGTGTCCATCCATCACTGCTGACCCT
TCGGAACAAGGCAAAGG**CTATAA**AAAAAATTAAGCAGC
GCCCCTTCCCCACA**CTATCT**CAATGCAAATATCTGTCTGAAACGGTTC

**Figure 5: Final Motif Alignment.** The bases in blue indicate the final alignment upon termination of the Gibbs Sampler. Note that some sequences contain the approximations of both the consensus (T/A)GATA(A/G) and its complement (T/C)TATC(A/T).

Once a final alignment is presented, a profile describing the motif is available. A profile describes the motif, position by position, by giving the frequency at which each residue occurs in each of the motif positions. Profiles are useful in describing and searching for families of sequences (Schneider, et. al., 1986; Gribskov et. al., 1987; Gribskov, et. al, 1990; Schneider and Stephens, 1990; Henikoff and Henikoff, 1991; Tatusov, et. al., 1994; Pietrokvoski et. al. 1996; Gribskov and Veretnik, 1996).

Implementations of the Gibbs Sampler run in $N$ linear time. The approximate running time is $TNLW$ where $T$ is the number of iterations through the sampler (typically around 100), $N$ is the number of input sequences, $L$ is the average length of the $N$ sequences, and $W$ is the width of the motif.

The Gibbs sampler does have some shortcomings. One is that the Gibbs sampler expects exactly one occurrence of the motif in each sequence. This is taken care of in a later version (Neuwald, et. al., 1995). The sequences being studied must be divergent enough to detect the differences. There also needs to be a sufficient enough data set (at least 10 sequences) for the Gibbs sampler to work. An optimal alignment is not guaranteed, due to the inherent sampling strategy. Also, an approximate motif width must be known.

## Hidden Markov Models

Hidden Markov Models (HMMs) describe a family of related objects by defining a probability distribution over all of the objects used to create the model. HMMs historically have their roots in speech processing for the purpose of recognition of words given the valid phonemes of a language (Rabiner, 1989).

One particular implementation of HMMs in speech is the SPHINX Speech Recognition System (Lee, et. al., 1990). In SPHINX, the goal is to create a system that can accurately determine the continuous speech of an independent speaker in which a large vocabulary is used.

### HMM Structure in Sequence Analysis

Pattern matching in speech recognition can be applied to sequence analysis in a straightforward manner. The basic building block of words in speech is the phone. In molecular sequence analysis, it is either amino acids or nucleic acids. Phones are used to build words. Similarly, amino acids or nucleic acids are used to build motifs. The difference is that in natural languages, the words are known, while in sequence analysis, we may only know a small portion of the motifs.

HMMs are generalized profiles. In sequence analysis, an HMM can be thought of as a structure that generates sequences by a random process. In particular, it can be thought of as a finite state machine. For sequence analysis, HMMs generally consist of three states: match states ($m$), delete states ($d$), and insert states ($i$). Figure 6 shows the structure of HMMs as discussed in Krough et. al., 1990.

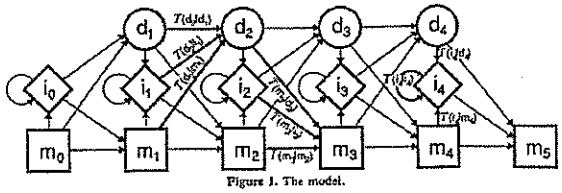



**Figure 6: Generalized Hidden Markov Model.** This figure (taken from Krough, et. al., 1990) shows the state structure of an HMM used in sequence analysis.

### States and Transitions in Hidden Markov Models

Each match state $m$ generates a letter $x$ from the alphabet $\Sigma$ according to a probability distribution. When nucleic acids are used, $\Sigma$ is a four-letter alphabet. In the case of amino acids, $\Sigma$ consists of 20 elements.

There is the possibility that a gap exists in the alignment of sequences. Hidden Markov models allow for such events through the delete states, $d$.

It is also possible that one or more sequences in an alignment contain a series of inserted residues differing from the other sequences. This is taken care of through the insert states, $i$. The insert states emit a letter $x$ from the alphabet $\Sigma$. This differs from the current match state $m$ because there is a different probability distribution being used, and the insertion state $i$ also allows the possibility of staying at the current state.

Each of the states in an HMM have three outgoing transitions. Transitions from delete and match states move forward in the model. Since insertion states allow for the insertion of more than one element, there is one transition associated with insertion states that stays at the same state in the HMM.

Hidden Markov Models are useful in sequence analysis because they model molecular sequence properties in three ways. First, HMMs indicate a sequence of positions, each with its own distribution over the alphabet (amino acids or nucleic acids) being used. Secondly, they allow the possibility of inserted or deleted segments. Finally, they allow for the possibility that continuing an insertion or deletion is more likely than starting one.

### Training and construction of HMMs

A framework for describing a family of related biological sequences is now in place. The state and transition probabilities for a particular instance of an HMM are created through what is referred to as training. HMMs are constructed from a set of similar yet unaligned primary sequences using an expectation-maximization forward-backward algorithm discussed as follows. The first step is to create an initial model by assigning probabilities for the transitions and emission states. Using the current model, consider all possible paths for each training sequence to get new probabilities for the transitions and emission states. Create a new model incorporating the updated probabilities. Continue by repeating the steps until the model converges to equilibrium.

The end product should be a model that assigns large probabilities to the members of the family of sequences it describes. For any given sequence, the probability that it belongs to the model is the sum over all paths that could produce that sequence.

Once an HMM is built, databases can be searched for new members. A negative log likelihood score (NLL) for each sequence can be scored as according to equation 2.

$$NLL = -\log(\Pr(sequence \mid \mathrm{mod}\, el))$$

**Equation 2: Calculation of negative log likelihood.**

When the NLL is normalized according to the length of the sequence, it is used to determine whether or not the sequence is a member of the family. The time it takes to search a database is proportional to the number of residues in the database times the length of the model.

## Regular Expression Pattern Matching

Many sequence databases have been set up for nucleic acid and amino acid sequences. Among these are GenBank (Benson, et. al., 1998), European Molecular Biology Laboratory (EMBL) Nucleotide Sequence Database (Stoesser, et. al., 1999), Protein Data Bank (PDB) (Abola, et. al., 1997), and SWISS-PROT (Bairoch and Apweiler, 1999). PROSITE (Hofmann, et. al., 1999) and BLOCKS (Pietrokovski, et. al., 1996) are examples of tools that can be used to search databases for the occurrences of sequences that are described by a particular regular expression. Since the databases are rather large, an optimal alignment of a single sequence to a regular expression in a minimal number of operations is desired.

A sequence can be tested for an alignment to a regular expression, *R*, using an alignment graph where each row is formed from the state of an ε-NFA *F* constructed from *R* as described in Knight and Myers, 1992. Figure 8 shows one such ε-NFA.

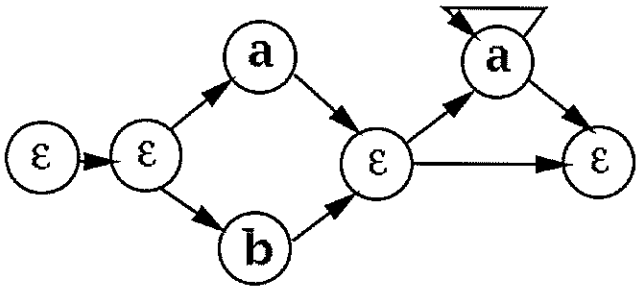



**Figure 7: ε-NFA describing the language (a|b) a*.** This figure describes the language encoded by the regular expression (a|b)a*.

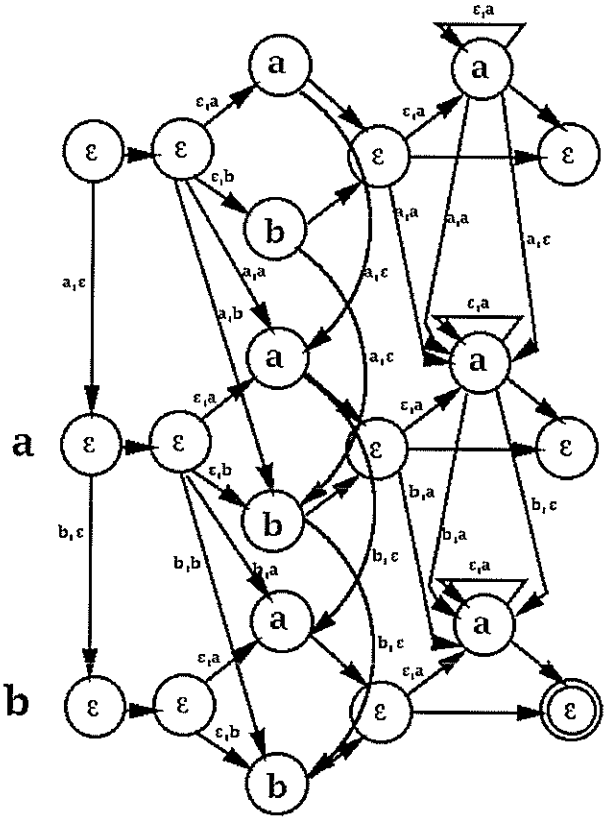



**Figure 8: Alignment graph.** This figure indicates the alignment graph for the sequence ab to the language described by the regular expression (a|b)a*, which is described above in Figure 4. (Adapted from Knight, et. al., 1992.)

**Regular expression matching without gap penalties**

The shortest cost path at state $s$ in the $i^{th}$ copy of the NFA is computed as the minimum of a match, insert, or delete event. A match is calculated as the minimum transition from a state in the $(i-1)^{st}$ copy of the NFA to state $s$, matching the symbol of the transition with the symbol of $s$. Insertions are the minimum transition from a state $t$ in the $i^{th}$ copy of the NFA to state $s$, matching a gap to the symbol of state $s$. A deletion is the minimum transition from a state $t$ in the $(i-1)^{st}$ copy of the NFA to $s$, matching the symbol of the transition with a gap.

The shortest cost path of aligning a sequence to the NFA can be computed in $O(MP)$ time using a two sweep algorithm where $M$ is the length of the sequence and $P$ is the length of the regular expression. The first sweep computes the recurrence at each vertex ignoring Kleene closures while the second sweep includes the ignored Kleene closures. The minimum of the two gives the shortest path cost.

**Regular expression matching with gap penalties**

Regular expression pattern matching with gap penalties employs the same ideas. The difference is that all possible states $t$ with a path to $s$ are considered, instead of only those with direct transitions. This is an $O(MP(M+P))$ algorithm when $\omega$ is monotonic increasing. The best path between vertices $(i,t)$ and $(i,s)$ modeling the least cost gap is the path from $t$ to $s$ in $F$ with the fewest non-ε symbols. Using the concept of minimum envelopes discussed earlier reduces this to an $O(MP(logM + logP))$ algorithm by using nesting trees based on the location of alternations and Kleene closures in the NFA. Details of the regular expression pattern matching algorithms can be found in Knight and Myers, 1992.

## Discussion

In tying all of these methods together, it is helpful to consider when to use each one of them. Dynamic programming sequence alignment is useful when comparing two sequences and global alignments are desired. The downfalls are that the size of the sequences matters for space and alignments of three or more sequences using this approach can be time and space consuming.

Gibbs Sampling is useful when comparing a set of distantly related sequences. It is used to find local similarities (motifs) and performs best when no information about the sequences is known.

Hidden Markov Models are useful to describe a family of related sequences. They handle multiple motifs and gaps within them, whereas Gibbs Sampling only allows conserved gaps. HMMs are a useful way to store the description of a family.

Regular expressions are useful to describe smaller motif models. They help to condense data, which in turn saves space and reduces search time. However, regular expressions are probably not as useful for larger families.

## Acknowledgements

The material for this review article was taken from a presentation for an oral qualifying exam in the Computer Science Department at Washington University in St. Louis. I would like to thank David States, Ron Cytron, and Dr. Subhash Suri for their time in participating on the exam committee.

## References

Abola, E.E., Sussman, J.L., Prilusky, J., Manning, N.O., (1997) "Protein Data Bank Archives of Three-Dimensional Macromolecular Structures." in *Methods of Enzymology* (Carter, C.W. Jr., Sweet, R.M., eds.), Academic Press, San Diego.

Bairoch, A., Apweiler, R. (1999) "The SWISS-PROT protein sequence data bank and its supplement TrEMBL in 1999." *Nucleic Acids Research*, **27**(1):49-54.

Benson, D.A., Boguski, M.S., Lipman, D.J., Ostell, J., Ouellette, B.F., (1998) "GenBank." *Nucleic Acids Research*, **26**(1):1-7.

Cardon, L.R., Stormo, G., (1992) "Expectation maximization algorithm for identifying protein-binding sites with variable lengths from unaligned DNA fragments." *Journal of Molecular Biology*, **223**:159-170.

Crochemore, M., Lecroq, T., (1996) "Patterm matching and text data compression algorithms." *ACM Computing Surveys.*, **28**(1): 39-41.

Dempster, A.P., Laird, N.M., Rubin, D.B., (1977) "Maximum Likelihood from Incomplete Data via the EM Algorithm." *The Journal of the Royal Statistical Society, Series B*, 39:1-38.

Fitch W. Smith, T.F. (1983) "Optimal sequence alignments." *Proceedings of the National Academy of Sciences of the United States of America*, **80**:1382-1386.

Gribskov, M., Luthy, R., Eisenberg, D., (1990) "Profile Analysis." *Methods in Enzymology*, **183**:146-159.

Gribskov, M., McLachlan, A.D., Eisenberg, D., (1987) "Profile analysis: detection of distantly related proteins." *Proceedings of the National Academy of Sciences of the United States of America*, **84**(13)4355-4358.

Gribskov, M., Veretnik, S., (1996) "Identification of sequence pattern with profile analysis." *Methods in Enzymology*, **266**:198-212.

Grimson, W.E.L., Huttenlocher, D.P., (1991) "On the verification of hypothesized matches in model-based recognition." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **13**(12):1201-1213.

Henikoff, S., Henikoff, J.G., (1991) "Automated assembly of protein blocks for database searching." *Nucleic Acids Research*, **19**(23):6565-6572.

Hofmann, K., Bucher, P., Falquet, L., Bairoch, A., (1999) "The PROSITE database, its status in 1999." *Nucleic Acids Research*, **27**(1):215-219.

Knight, J.R., Myers, E.W., (1992) "Approximate Regular Expression Pattern Matching with Concave Gap Penalties." *Proceedings of the Third Symposium on Combinatorial Pattern Matching*, 67-78.

Krogh, A., Brown, S., Mian, I.S., Sjölander, K., Haussler, D., (1994) "Hidden Markov Models in Computational Biology: Applications to Protein Modeling." *Journal of Molecular Biology*, **235**, 1501-1531.

Lawrence, C.E., Altschul, S.F., Boguski, M.S., Liu, J.S., Neuwald, A.F., Wootton, J.C., (1993) "Detecting Subtle Sequence Signals: A Gibbs Sampling Strategy for Multiple Alignment." *Science*, **262**, 208-214.

Lawrence, C.E., Reilly, A.A., (1990) "An expectation maximization (EM) algorithm for the identification and characterisation of common sites in unaligned biopolymer sequences." *Proteins*, 7:41-51.

Lee, K.F., Hon, H.W., Reddy, R., (1990) "An Overview of the SPHINX Speech Recognition System." *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **38**(1), 35-44.

Neeleman, S.B., Wunsch, C.D., (1970) "A General Method Applicable to the Search for Similarities in the Amino Acid Sequence of Two Proteins." *Journal of Molecular Biology*, **48**, 443-453.

Neuwald, A.F., Liu, J.S., Lawrence, C.E., (1995) "Gibbs motif sampling: detection of outer membrane repeats." *Protein Science*, **4**: 1618-1632.

Ominichinski, J.G., Clore, G.M., Schaad, O., Felsenfeld, G., Trainor, C., Appella, E., Stahl, S.J., Gronenborn, A.M., (1993) "NMR Structure of a Specific DNA Complex of Zn-Containing DNA Binding Domain of GATA-1." *Science*, **261**: 438-446.

Orkin, S.H., (1995) " Regulation of globin gene expression in erythroid cells." *European Journal of Biochemestry*, **231**: 271-281.

Pietrokovski, S., Henikoff, J.G., Henikoff, S., (1996) "The BLOCKS database -- a system for protein classification." *Nucleic Acids Research*, **24**:197-200.

Rabiner, L.R. (1989) "A tutorial on hidden Markov models and selected applications in speech recognition." *Proceedings of the IEEE*, **77**:257-286.

RSA Laboratories. (1993). "PKCS #1: RSA Encryption Standard. Version 1.5." *Technical Report.*

Schneider, T.D., Stormo, G.D., Gold, L., Ehrenfeucht (1986). "Information content of binding sites on nucleotide sequences." *Journal of Molecular Biology*, **188**:415-431.

Schneider, T.D., Stephens, R.M. (1990) "Sequence logos: a new way to display consensus sequences." *Nucleic Acids Research*, **18**:6097-6100.

Sellers, P.H., (1974) "On the theory of computation of evolutionary distances." *SIAM Journal of Applied Mathematics*, **26**:787-793.

Smith, T.E., Waterman, M.S., (1981) "Identification of Common Molecular Subsequences." *Journal of Molecular Biology*, **147**, 195-197.

Stoesser, G., Tuli, M.A., Lopez, R., Sterk, P., (1999) "The EMBL Nucleotide Sequence Database." *Nucleic Acids Research*, **27**(1):18-24.

Stormo, G.D., Hartzell, G.W., (1989) "Identifying protein-binding sites from unaligned DNA fragments." *Proceedings of the National Academy of Sciences of the United States of America*, **86**:1183-1187.

Tatusov, R.L., Altschul, S.F., Koonin, E.V. (1994) "Detection of conserved segments in proteins: iterative scanning of sequence databases with alignment blocks." *Proceedings of the National Academy of Sciences of the United States of America*, **91**:12091-12095.