SEARCHING FOR REMOTELY HOMOLOGOUS SEQUENCES IN PROTEIN DATABASES WITH HYBRID PSI-BLAST

DISSERTATION

Presented in Partial Fulfillment of the Requirements for
the Degree of Doctor of Philosophy in the
Graduate School of The Ohio State University

By

Yuheng Li,

* * * * *

The Ohio State University

2006

Dissertation Committee:

Mario Lauria, Adviser
Ralf Bundschuh
Will Ray

Approved by

__________________________
Adviser
Biophysics Graduate Program
© Copyright by

Yuheng Li

2006
ABSTRACT

Sequence alignment is one of the fundamental techniques used in molecular biology. It has been widely used in many biological applications, such as protein classification, gene finding, homology modeling, structure and function prediction, phylogenetic analysis and database annotation. In high sensitivity sequence homology database searches, progressive sequence model refinement by means of iterative searches is an effective method and is currently employed in many popular tools such as PSI-BLAST and SAM. Recently, a novel alignment algorithm has been proposed that offers features expected to improve the sensitivity of such iterative approaches, specifically a well-characterized theory of its statistics even in the presence of position-specific gap costs. We have demonstrated that the new hybrid alignment algorithm is ready to be used as the alignment core of PSI-BLAST. We also evaluated the accuracy of two proposed approaches to edge effect correction in short sequence alignment statistics that turns out to be one of the crucial issues in developing a hybrid-alignment based version of PSI-BLAST. In addition, we have exploited other benefits of the hybrid alignment. We show that incorporating information about the suboptimal alignments, otherwise ignored in PSI-BLAST, already improves the sensitivity of PSI-BLAST. In one experiment, we have found a set of sequences on which our tool disagrees with the classification given by SCOP. Careful examination points to a possible misclassification in SCOP. Cross-referencing with two other methods of
protein structure classification, CATH and DALI, supports this view, indicating that the enriched information from suboptimal alignments is valuable for detecting more weakly related sequences. Finally, we have integrated position-specific gap penalties in PSI-BLAST, which is intentionally left out due to a theoretical limitation of its underlying Smith-Waterman score statistics. We also investigated several strategies to adjust the position-based gap costs derived from the forward-backward algorithm. The results show that the degree of conservedness calculated as a localized relative entropy from the position-specific substitution matrix is the most effective. Such enhancements further improve the sensitivity of PSI-BLAST for remote homology detection in database searches.
This thesis is dedicated to my wife, Na, and our son Alvin. They have always been my joy and encouragement.

I also dedicate this work to my parents, who have been very strict, but also made great sacrifices to their children.
I am deeply indebted to many individuals who have helped me in my pursuit of the graduate education.

First of all, I would like to thank my adviser and dissertation committee chair, Dr. Lauria. If without his consistent support, encouragement and guidance, this work could not have been accomplished. I have also learned a lot from his great personality and insightful knowledge. I will certainly not forget to mention the delicious cookies that he brought for us from Europe after his vacations.

It has been a pleasure to work with Dr. Bundschuh over the years. I have benefited a lot from discussions with him on many topics in this dissertational research. His intuitive explanations of some complicated concepts always amazed me and encouraged me to think more deeply.

I also would like to thank Dr. Ray and Dr. Yuan for their stimulating discussions on many topics not just limited to my dissertation. Special thanks to Dr. Clanton, the director of the biophysics graduate program, for his excellent work in taking care of the needs of the students in the program, including myself. As an international student, I have faced many difficulties during my graduate education. Dr. Clanton is always ready to offer great help, practical advice and warm encouragement.

I am grateful to Nicholas, who is now Dr. Chia. We have cooperated on the late part of the Hybrid PSI-BLAST project and had many interesting discussions.
Finally, I want to acknowledge the sacrifices that my wife has made for me through the years of grad school. Her love and support is always with me in those stressful times. I would also want to take this opportunity to express my sincerest gratitude to my parents and my parents-in-law for their unconditional belief in me and help in taking care of Alvin, my naughty but nonetheless genius son.
VITA

October 15, 1976............................Born in Jing Men, Hubei Province, China
July, 1999.................................B.S., Technical Physics, Beijing University, Beijing, China
October, 2000 – Sept, 2001............Graduate Teaching Associate,
Biophysics Program, The Ohio State University, Columbus, USA
October, 2001 – August, 2002............Graduate Teaching Associate,
Department of Chemistry, The Ohio State University, Columbus, USA
September, 2002 – present.................Graduate Research Associate,
Department of Computer Science and Engineering, The Ohio State University, Columbus, USA

PUBLICATIONS

Research Publications


**FIELDS OF STUDY**

Major Field: Biophysics

*Bioinformatics and Computational Biology*

Studies in Hybrid Alignment: Prof. Mario Lauria and Prof. Ralf Bundschuh
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract</td>
<td>ii</td>
</tr>
<tr>
<td>Dedication</td>
<td>iv</td>
</tr>
<tr>
<td>Acknowledgments</td>
<td>v</td>
</tr>
<tr>
<td>Vita</td>
<td>vii</td>
</tr>
<tr>
<td>List of Tables</td>
<td>xii</td>
</tr>
<tr>
<td>List of Figures</td>
<td>xiii</td>
</tr>
<tr>
<td>Chapters:</td>
<td></td>
</tr>
<tr>
<td>1. INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Motivation</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Goals of this thesis</td>
<td>4</td>
</tr>
<tr>
<td>1.3 Contributions</td>
<td>5</td>
</tr>
<tr>
<td>1.4 Organization of this thesis</td>
<td>6</td>
</tr>
<tr>
<td>2. BACKGROUND</td>
<td>8</td>
</tr>
<tr>
<td>2.1 Introduction</td>
<td>8</td>
</tr>
<tr>
<td>2.2 Biological background</td>
<td>9</td>
</tr>
<tr>
<td>2.2.1 Protein sequences</td>
<td>9</td>
</tr>
<tr>
<td>2.2.2 Protein structures and functions</td>
<td>12</td>
</tr>
<tr>
<td>2.2.3 Protein homology</td>
<td>16</td>
</tr>
<tr>
<td>2.2.4 Applications of protein homology</td>
<td>18</td>
</tr>
<tr>
<td>2.3 Background on sequence alignment</td>
<td>19</td>
</tr>
<tr>
<td>2.3.1 Overview of sequence alignment methods</td>
<td>21</td>
</tr>
<tr>
<td>Section</td>
<td>Title</td>
</tr>
<tr>
<td>---------</td>
<td>--------</td>
</tr>
<tr>
<td>2.3.2</td>
<td>Sequence alignment algorithms</td>
</tr>
<tr>
<td>2.3.3</td>
<td>Alignment score statistics</td>
</tr>
<tr>
<td>2.4</td>
<td>Survey of some existing sequence alignment tools</td>
</tr>
<tr>
<td>2.4.1</td>
<td>BLAST and FASTA</td>
</tr>
<tr>
<td>2.4.2</td>
<td>PSI-BLAST</td>
</tr>
<tr>
<td>2.4.3</td>
<td>HMMER and SAM</td>
</tr>
<tr>
<td>2.4.4</td>
<td>CLUSTALW</td>
</tr>
<tr>
<td>3.1</td>
<td>Introduction</td>
</tr>
<tr>
<td>3.2</td>
<td>Hybrid model topology</td>
</tr>
<tr>
<td>3.3</td>
<td>Hybrid alignment statistics</td>
</tr>
<tr>
<td>3.4</td>
<td>Forward-backward algorithm</td>
</tr>
<tr>
<td>3.5</td>
<td>Posterior decoding</td>
</tr>
<tr>
<td>4.1</td>
<td>Introduction</td>
</tr>
<tr>
<td>4.2</td>
<td>Implementation stages</td>
</tr>
<tr>
<td>4.2.1</td>
<td>Stage I: Initial incorporation of hybrid algorithm in NCBI PSI-BLAST</td>
</tr>
<tr>
<td>4.2.2</td>
<td>Stage II: Utilization of suboptimal alignments in building emission probabilities</td>
</tr>
<tr>
<td>4.2.3</td>
<td>Stage III: Incorporation of position specific transition probabilities</td>
</tr>
<tr>
<td>5.1</td>
<td>Introduction</td>
</tr>
<tr>
<td>5.2</td>
<td>Performance assessment</td>
</tr>
<tr>
<td>5.3</td>
<td>Database search heuristics</td>
</tr>
<tr>
<td>5.4</td>
<td>Finite-size correction</td>
</tr>
<tr>
<td>5.5</td>
<td>Sequence weighting</td>
</tr>
<tr>
<td>5.6</td>
<td>Model building and regularization</td>
</tr>
<tr>
<td>5.6.1</td>
<td>Emission probabilities estimation</td>
</tr>
<tr>
<td>5.6.2</td>
<td>Transition probabilities estimation</td>
</tr>
<tr>
<td>6.1</td>
<td>Compatibility studies</td>
</tr>
<tr>
<td>6.2</td>
<td>Finite-size correction</td>
</tr>
<tr>
<td>6.3</td>
<td>Parallelization of Hybrid PSI-BLAST</td>
</tr>
</tbody>
</table>
LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1 Twenty common amino acids found in proteins with their 3-letter and 1-letter code. The last column shows amino acid chemical properties as (+/-) charged, polar or hydrophobic.</td>
<td>10</td>
</tr>
<tr>
<td>5.1 Example of position specific weighting (PSW) and generalized position specific weighting (GPSW).</td>
<td>81</td>
</tr>
<tr>
<td>6.1 Ambiguous sequences that have been identified by the conservative counting method during the evaluation of the performance of emHybrid PSI-BLAST. These sequences have an ( E )-value less than 0.001. The first column is the names of those sequences, the second column is the SCOP classification of each sequence. Four codes, which are separated by dot, represent \textit{Class}, \textit{Fold}, \textit{Superfamily}, and \textit{Family}, respectively. Conventionally, sequences from the same superfamly are considered homologs. Otherwise, they are non-homologs. The last column is the CATH classification, with each number representing \textit{Class}, \textit{Architecture}, \textit{Topology}, and \textit{Homologous superfamily}, respectively. In CATH, sequences are considered homologs if they are classified in the same Homologous superfamily. Discrepancies are evident for the classification on these sequences by SCOP and CATH.</td>
<td>128</td>
</tr>
<tr>
<td>6.2 The ROC values for different choices of two-level gap opening gap cost scheme. It shows that the choice of the upper bound of the gap opening cost is more important, because the variation of ROC is about 1% due to changes in the lower bound of the gap opening cost and threshold, while it increases to 10% at ( EPQ = 1 ) if we change the upper bound instead. The upper bound of the gap open cost at 13 works the best among all the choices.</td>
<td>131</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1 Central dogma in biology. DNA can be replicated or be transcribed into RNA, which can also be reverse-transcribed back into DNA. Protein is produced by translating RNA.</td>
<td>10</td>
</tr>
<tr>
<td>2.2 Diagram of amino acid. The central carbon $C_\alpha$ connects to four functional groups in amino acid. These groups are a hydrogen (−H) atom, an amino group (−$NH_2$), a carboxyl group (−COOH) and a side chain (−$R$).</td>
<td>11</td>
</tr>
<tr>
<td>2.3 Example of a multiple alignment of part of Cytochrome b from different species.</td>
<td>21</td>
</tr>
<tr>
<td>2.4 BLOSUM62 substitution score matrix</td>
<td>23</td>
</tr>
<tr>
<td>2.5 Dotplot between a tropomyosin sequence and itself. Lines along diagonals represent identical segments. Lines along the anti-diagonals are inverted regions. Short parallel segments around position 180 are the repeats.</td>
<td>27</td>
</tr>
<tr>
<td>2.6 Gumbel distribution</td>
<td>30</td>
</tr>
<tr>
<td>3.1 Model topology for hybrid alignment. Black solid squares represent match states (M), Blue solid squares denote insert states (I) and red circles depict delete states (D). The diagram on the left shows the linear structure of the hidden markov model of the hybrid alignment. The diagram on the right shows the expanded 2D structure of the topology when a subject sequence is to be aligned with the query model. Two dummy states, i.e. the begin state (B) and the end state (E), are not shown.</td>
<td>42</td>
</tr>
</tbody>
</table>
3.2 Alignment lattice with an alignment path example .................................. 44

3.3 A schematic diagram of the forward and the backward algorithms. For illustration purpose, some of the transitions are omitted. The backward algorithm executes backward along the reversed transitions of the forward algorithm ................................................................. 47

4.1 PSI-BLAST execution flow chart ............................................................. 54

4.2 Hybrid PSI-BLAST execution flow chart after stage I ............................... 59

4.3 Hybrid PSI-BLAST execution flow chart after stage II ............................. 62

5.1 A schematic drawing about PSI-BLAST heuristics for extension on a seed. The gapless extension starts from the seed and extends on the diagonal to both directions. It then stops when the alignment score falls off the maximum score obtained so far by more than a certain threshold. Then the extension resumes with gaps allowed. Such extension will search other diagonals .......................................................... 73

5.2 An example of the “soft-band” forward search heuristics. The score landscape is calculated from a full scale forward alignment. A seed is identified at (30, 14) on the alignment lattice. The thick blue line is the boundary of the gapped extension, which adjusts its position along the dominant alignment path. It also locates the correct end point at (128, 140), which is even on a different diagonal as the seed ........................................ 75

5.3 A example of the $M \rightarrow M$ probabilities in a model (of sequence d1kzfzfa..) from SUPER90 v1.69 dataset and the corresponding model built by the forward-backward algorithm. There is an abundance of structures in the SUPER90 model as demonstrated by the peaks. The peaks derived from the forward-backward alignment are fewer in number and not as deep as those in SUPER90 model ........................................ 87

5.4 Using transition probabilities from the previous round as the pseudo-count $\eta_0$ works better than using the non-position specific values given in the command line. For this experiment, the ASTRAL40 dataset was used ........................................ 89
5.5 Relationship between average relative entropy $\bar{H}$ and $M \rightarrow M$ probability. As shown in the graph, these two quantities have a positive correlation for positions with high relative entropy, which tend to have high $M \rightarrow M$ probabilities as well. But for positions with low relative entropies, there is a wide distribution of possible $M \rightarrow M$ probabilities. (Plot produced with the help from Nicholas Chia.)

6.1 Comparison of Hybrid PSI-BLAST performance for different gap costs. The curves show the trade-off between the errors per query and the coverage for Hybrid PSI-BLAST on a “gold standard” database using different gap costs. While all curves are relatively close together, a cost of $11 + 1 \cdot k$ for a gap of length $k$ seems to lead to the best performance.

6.2 Comparison of the NCBI and the Hybrid version of PSI-BLAST. The curves show the trade-off between the errors per query and the coverage for NCBI and Hybrid PSI-BLAST on a “gold standard” database. For small coverages Hybrid PSI-BLAST is slightly superior while for high coverages the NCBI PSI-BLAST performs better.

6.3 Comparison of the NCBI and the Hybrid version of PSI-BLAST on a large database ASTRAL40NRtrim. The curves show the trade-off between the errors per query and the coverage for NCBI and Hybrid PSI-BLAST for those sequence pairs the homology of which is known from structural considerations. For small coverages Hybrid PSI-BLAST is slightly inferior while for high coverages the two algorithms perform nearly identically.

6.4 Comparison of two formulas for edge effect correction. Both graphs show the dependence between the $E$-value cutoff and the number of errors per query, i.e., the number of non-homologous sequence pairs with an $E$-value lower than the cutoff divided by the total number of sequences in the database. The dotted line corresponds to hybrid alignment with $E$-values calculated according to Eq. (5.4) while the solid line corresponds to hybrid alignment with $E$-values calculated according to Eq. (5.5). The dash-dotted line is the result of BLAST 2.0 and the dashed line is the identity corresponding to an ideal algorithm. Both graphs show data for the ASTRAL40 database and the BLOSUM62 scoring matrix. In (a) the cost of a gap of length $k$ is $11 + k$ while in (b) it is $9 + 2k$. In both cases BLAST 2.0 and Eq. (5.5) yield good estimates of the $E$-value while Eq. (5.4) is clearly inferior for hybrid alignment.
6.5 Speedup of the first stage implementation of Hybrid PSI-BLAST by parallelization ........................................ 105

6.6 Efficiency graphs of the parallelized Hybrid PSI-BLAST after implementation stage I on a sample of 100 sequences chosen at random from ASTRAL40 dataset. ........................................ 106

6.7 Execution times for the 100 individual query sequences. .......................... 108

6.8 Miss rate vs. Alignment score plot. It is clear that the new heuristics, which uses an adaptive banded search, works better than the old heuristics approach, since the latter may potentially miss a good sequence candidate for hybrid alignment due to the limited region covered on the alignment lattice ........................................ 111

6.9 The plot of the effectiveness of both heuristics approaches. The effectiveness is defined as the average execution time per passed sequence \(AET\). It shows that the new heuristics spends less time on average than the old heuristics to identify a potential hit. ........................................ 112

6.10 Comparison for different sequence weighting schemes. As illustrated in the plot, the generalized position-specific weighting (GPSW) outperforms the equal weighting method, because the latter essentially does not take the database bias into account. Interestingly, the weighting scheme used by NCBI PSI-BLAST works better than the GPSW approach in the low coverage region, but worse than GPSW in the high coverage region. ........................................ 114

6.11 Performance of the block-wise GPSW shows that such weighting does little change to the ROC curves. However, it improves the performance in the high coverage region, while slightly loses performance in the low coverage region. The block-wise GPSW with \(Y = 0.5\) seems work best overall. ........................................ 116

6.12 emHybrid PSI-BLAST performance with different choices of pseudo-count constant \(\beta\). The larger the constant, the more the emphasis on the prior belief of the amino acid substitution probabilities. The default value 10 used in NCBI PSI-BLAST works best for emHybrid PSI-BLAST as well. ........................................ 119
6.13 Performance comparison among NCBI PSI-BLAST, preHybrid PSI-BLAST, and emHybrid PSI-BLAST. The straight counting method is used in calculating EPQ. preHybrid PSI-BLAST works best in the low coverage region, but worse in the high coverage region. emHybrid PSI-BLAST is comparable to or slightly better than NCBI PSI-BLAST.

6.14 Performance comparison among NCBI PSI-BLAST, preHybrid PSI-BLAST, and emHybrid PSI-BLAST. The conservative counting scheme is used instead. emHybrid PSI-BLAST outperforms both NCBI PSI-BLAST and preHybrid PSI-BLAST over the entire range of coverage.

6.15 Difference between two error counting methods on emHybrid PSI-BLAST. The conservative counting scheme reports less errors than the straight counting as expected. Those disputable sequences that have an E-value less than 0.001 are collected and examined more closely. When two other protein structure classification systems, namely CATH and DALI, have been used to test their relationship, almost all of the “errors” found by the straight counting are classified as “homologs” according to CATH and DALI.

6.16 Difference between two error counting methods on NCBI PSI-BLAST. There is hardly any change, which implies that almost all the false hits identified by the straight counting belong to even different folds. Such errors are true non-homologs.

6.17 Performance comparison among emHybrid PSI-BLAST, Hybrid PSI-BLAST with the initial incorporation of the position-specific transition probability matrix, and Hybrid PSI-BLAST with subsequent adjustment on the matrix. emHybrid PSI-BLAST works the best among them, but the adjusted transition profiles improve over those which only use raw values.

6.18 Performance comparison among emHybrid PSI-BLAST, Hybrid PSI-BLAST with the adjustment on the raw position-specific transition probability matrix, and Hybrid PSI-BLAST with additional capping on the adjusted gap opening costs. emHybrid PSI-BLAST still works the best when coverage is low. However, the limitation that prevents the gap opening costs from going up too high improves the performance of Hybrid PSI-BLAST and makes it even better than emHybrid PSI-BLAST at high coverages.
6.19 Performance of Hybrid PSI-BLAST with different values of threshold \( h \). The choice of 0.9 works best among all the tested values.

6.20 Performance of Hybrid PSI-BLAST with different values for the cap on \( M \rightarrow M \) probability. The threshold \( h \) is set to 0.9. The performance is the best when \( M \rightarrow M \) is capped at 0.93, which roughly corresponding to a gap opening cost of 13. This result is consistent with the data from the capping experiment, which also achieves the optimal performance with a cap of 13.

6.21 Performance of different versions of PSI-BLAST: emHybrid PSI-BLAST is the version produced in stage II, which incorporates the information in the suboptimal alignments into the building of new models; Hybrid PSI-BLAST is the stage III version, which is built upon emHybrid PSI-BLAST and is enhanced with regularized position specific gap penalties.
CHAPTER 1

INTRODUCTION

1.1 Motivation

This dissertation is focused on the development of a new generation of sequence alignment tool, which is able to automatically detect much weaker sequence homologies in databases than currently available tools. Many applications of sequence alignments such as functional and structural prediction, protein homology modeling, and large scale database annotation will benefit from the very ability to identify more remote homologous sequences without much human intervention. Ultimately, a better understanding of gene functions and biological processes through highly sensitive sequence alignment will advance human health.

Homology when used to describe the relationship between sequences means those sequences are descendants of a common ancestry. However, we cannot verify the existence of such common ancestor sequence by experiments. Scientists instead collect other evidences to infer the homology relationship. Similarity between sequences has long being used as such evidence.
Trying to gain valuable insight on properties of unknown objects by comparing them to similar objects with known characteristics is a time-honored technique. Sequence alignment is the application of such comparison technique to the molecular biology domain. High similarity between closely related sequences is relatively easy to detect by our eyes, whereas moderately weak sequence similarity can be uncovered with aid of computers through sequence alignment algorithms, such as BLAST and FASTA. Distant homology requires more careful analysis of the whole family of related sequences. Models such as position specific scoring matrices (PSSM) and hidden Markov models (HMM) are used to capture the essential properties of a sequence family.

Such computer-aided approaches assign some quantitative measures, called scores, to characterize the underlying similarities. Usually, the higher the score and thus similarity, the more likely that the sequences are homologous. However, sequences can achieve a very good score just by chance. To address this problem, various statistical approaches have been applied to characterize the significance of the scores. One common approach is to obtain the distribution of scores from comparison of random sequences, from which the probability of acquiring a certain score by pure chance is derived. For the renown Smith-Waterman sequence alignment algorithm, the alignment score distribution for random sequences is analytically proven to follow the so-called Gumbel distribution, a special case of Extreme Value Distribution (EVD), when no gap is allowed in the alignment. Even when gap is permitted, the score distribution was empirically found to still follow the Gumbel form in many numerical studies.
The Gumbel distribution is fully characterized by two distribution parameters, namely \( \lambda \) and \( K \). \( \lambda \) is the scale parameter, which controls the more important tail part, which is often the most interesting region people care about. \( K \) is the placement parameter, determining where the mode is. Unfortunately, these two parameters depend on the individual scoring matrix used for the alignment and are usually derived through computationally intensive simulation processes. It is the main reason that one of the most popular sequence alignment tool, BLAST, is restricted to only a handful of scoring matrices, which have precomputed \( \lambda \) and \( K \) values.

PSSM approaches, such as PSI-BLAST, suffer from this drawback as well. Because PSI-BLAST is an iterative sequence alignment tool, which dynamically refines its scoring matrix from iteration to iteration. To avoid simulations needed for fitting those two parameters at each iteration, PSI-BLAST adopts a heuristic method by scaling the scoring matrix by a factor and only use non-position specific gap penalties. This restriction is considered to potentially hamper the sensitivity of PSI-BLAST program. As for the HMM methods, the score distribution is even less understood, but has a Gumbel like tail. So HMMER, another well known sequence alignment and modeling tool, calibrates its significance assessment routine by fitting the scores from random sequences to the tail of a Gumbel form.

Recently, Yu and Hwa [60] have proposed an alternative sequence alignment algorithm to the Smith-Waterman algorithm. Since it is the ”hybrid” of the probabilistic modeling approach, such as HMMs, and of the score-optimization approach, such as Smith-Waterman algorithm, the authors have named it hybrid algorithm. The main feature of hybrid algorithm comes from its well characterized alignment statistics for
the score distribution of the null model (random sequences). Theoretical and empirical studies have shown that its score distribution follows a Gumbel distribution with the key parameter $\lambda$ taking a fixed value for a wide range of scoring systems, even including position-specific gap costs. So this universal statistic can benefit many bioinformatics applications, such as BLAST and PSI-BLAST. It can naturally extend applicability of BLAST to other scoring matrices and relieve PSI-BLAST from its restriction to non-position specific gap penalties. In fact, a prototype of hybrid algorithm has been implemented in BLAST program by Yu, et al. The hybrid version of BLAST performs comparably to the original BLAST, but can now use any scoring system.

1.2 Goals of this thesis

This dissertation seeks to further investigate the properties of hybrid alignment in the context of position-specific scoring systems, including position-specific gap costs. Another purpose of this thesis is to demonstrate the applicability of hybrid algorithm by implementing it in the framework of PSI-BLAST. The specific aims that this work wants to achieve are listed as follows:

- Incorporating position-specific gap costs in hybrid version of PSI-BLAST (Hybrid PSI-BLAST)

- Explore the model building strategies that can be used in Hybrid PSI-BLAST

- Evaluate the performance of Hybrid PSI-BLAST and compare it to other sequence alignment algorithms
Ultimately, this thesis aims at significantly increasing the sensitivity of PSI-BLAST, so that much weaker sequence homology can be detected in database searches. If the promised significant improvement is realized, we hope that the hybrid algorithm will be incorporated into the PSI-BLAST package distributed by the National Center for Biotechnology Information (NCBI). Given that the NCBI tools have become the most widely used sequence database search engines, our work has the potential to benefit the worldwide research community.

1.3 Contributions

This dissertation describes the process of the development and implementation of hybrid alignment in PSI-BLAST, and the incorporation of position specific gap penalties into the PSI-BLAST package. The capability to handle position-specific gap penalties was missing from the original PSI-BLAST (NCBI PSI-BLAST) due to the lack of a theory for the Smith-Waterman alignment score statistics. This thesis also proposes a modified heuristics to speed up the database searches for hybrid alignment, because the different nature of the hybrid alignment and the Smith-Waterman alignment renders the old search heuristics no longer suitable for hybrid alignment.

New methods for constructing position-specific scoring matrices have been investigated for Hybrid PSI-BLAST. For example, suboptimal alignments that are ignored in NCBI PSI-BLAST can now be used to build more sensitive substitution matrices. Creation of position specific gap penalties have been studied extensively and a number of measures have been adopted to produce high quality gap cost schemes. Great care has been taken in modeling the protein family during the iterative search loops.
This dissertation proposes a generalization of the position specific weight scheme for sequence weighting in the model building phase, which handles not only multiple alignment derived from optimal pair-wise alignments but also a generalized probabilistic multiple alignment. In studying the alignment statistics of hybrid alignment, an alternative length correction formula for statistical significance assessment that works better for hybrid alignment is used in place of the Altschul-Karlin’s formula.

1.4 Organization of this thesis

Chapter 2 presents the background information on biological properties of proteins and basics for sequence alignment. A survey of some existing popular sequence alignment tools is also given in that chapter. The detailed introduction to hybrid algorithm can be found in Chapter 3, which describes the underlying hidden markov model of hybrid algorithm and the featured statistical properties of its alignment score distributions. In addition, it documents a frame-work for deriving new scoring system, which is employed in Chapter 4. Chapter 4 gives a detailed account of the development and implementation process, which took place in three stages. The following chapter, Chapter 5 focuses on some of the important issues that have arisen in the process, including performance assessment, database search heuristics, finite-size correction, sequence weighting and construction of new scoring systems. In Chapter 6, an extensive analysis is carried out on the results of compatibility studies of the hybrid alignment with the PSI-BLAST package, on the performance assessment of the new database search heuristics, and on the effect of the alternative finite-size correction formula for alignment statistics. Results from the investigation of sequence weight
schemes and model building methods are also presented in Chapter 6. Conclusion of this dissertation and possible future directions are described in Chapter 7.
CHAPTER 2

BACKGROUND

2.1 Introduction

Bioinformatics is one of the fastest growing interdisciplinary fields of science, and one that is acting as a strong force in changing the landscape of modern life sciences. It brings together experts from many disciplines, such as biology, chemistry, physics, computer science, engineering and mathematics. Given the breadth of topics encompassed by these fields, this chapter is not intended to be a comprehensive introduction to bioinformatics. It rather focuses on the necessary background that would assist an average reader to understand and critically analyze this thesis.

As mentioned in chapter 1, the question being asked is how to search for much weaker homologs of a given protein sequence. In this chapter, we will first introduce the reader to what a protein is and why seeking remote homologs is an important question. We will also give an overview of the history of the sequence alignment research and some of the commonly used methods.
2.2 Biological background

Proteins are biological molecules essential to life. They are the basic building blocks of life and involved in almost all aspects of our activities. For example, hemoglobin is transporting oxygen to all parts of our body at any given moment, actin and myosin make us move around, DNA polymerase copies our genetic material faithfully, and rhodopsin receives light signals that enable our perception of light. Hence, the study of proteins has been at the heart of biological research for many years. In the post-genomic era, hundreds and thousands of new proteins are discovered at an astonishing rate. As of July, 2006, over two million protein sequences covering more than three thousand organisms have been deposited in RefSeq database maintained by National Center for Biotechnology Information (NCBI). The sheer amount of data presents a golden opportunity for discovery and at the same time confronts us with even greater challenges of revealing the properties of those uncharacterized proteins.

2.2.1 Protein sequences

So what exactly are proteins? Simply put, proteins are macromolecules consisting of chains of amino acids. The blue prints of proteins are stored in cell’s DNA molecules, which are first transcribed into RNA molecules and then RNA molecules are translated into protein products. This relationship is the central dogma of biology and is shown schematically in Figure 2.1.

Amino acids are the components of proteins. There are 20 common amino acids naturally occurring in proteins. Scientists often use three-letter or one-letter abbreviations of their names, which are listed below in Table 2.1.
Figure 2.1: Central dogma in biology. DNA can be replicated or be transcribed into RNA, which can also be reverse-transcribed back into DNA. Protein is produced by translating RNA.

<table>
<thead>
<tr>
<th>Amino Acid</th>
<th>3-letter code</th>
<th>1-letter code</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alanine</td>
<td>Ala</td>
<td>A</td>
<td>Hydrophobic</td>
</tr>
<tr>
<td>Arginine</td>
<td>Arg</td>
<td>R</td>
<td>+ Charged</td>
</tr>
<tr>
<td>Asparagine</td>
<td>Asn</td>
<td>N</td>
<td>Polar</td>
</tr>
<tr>
<td>Aspartic acid</td>
<td>Asp</td>
<td>D</td>
<td>- Charged</td>
</tr>
<tr>
<td>Cysteine</td>
<td>Cys</td>
<td>C</td>
<td>Polar</td>
</tr>
<tr>
<td>Glutamic acid</td>
<td>Glu</td>
<td>E</td>
<td>- Charged</td>
</tr>
<tr>
<td>Glutamine</td>
<td>Gln</td>
<td>Q</td>
<td>Polar</td>
</tr>
<tr>
<td>Glycine</td>
<td>Gly</td>
<td>G</td>
<td>- Changed</td>
</tr>
<tr>
<td>Histidine</td>
<td>His</td>
<td>H</td>
<td>Polar, + Charged</td>
</tr>
<tr>
<td>Isoleucine</td>
<td>Ile</td>
<td>I</td>
<td>Hydrophobic</td>
</tr>
<tr>
<td>Leucine</td>
<td>Leu</td>
<td>L</td>
<td>Hydrophobic</td>
</tr>
<tr>
<td>Lysine</td>
<td>Lys</td>
<td>K</td>
<td>+ Charged</td>
</tr>
<tr>
<td>Methionine</td>
<td>Met</td>
<td>M</td>
<td>Hydrophobic</td>
</tr>
<tr>
<td>Phenylalanine</td>
<td>Phe</td>
<td>F</td>
<td>Hydrophobic</td>
</tr>
<tr>
<td>Proline</td>
<td>Pro</td>
<td>P</td>
<td>Hydrophobic</td>
</tr>
<tr>
<td>Serine</td>
<td>Ser</td>
<td>S</td>
<td>Polar</td>
</tr>
<tr>
<td>Threonine</td>
<td>Thr</td>
<td>T</td>
<td>Polar</td>
</tr>
<tr>
<td>Tryptophan</td>
<td>Trp</td>
<td>W</td>
<td>Polar</td>
</tr>
<tr>
<td>Tyrosine</td>
<td>Tyr</td>
<td>Y</td>
<td>Polar</td>
</tr>
<tr>
<td>Valine</td>
<td>Val</td>
<td>V</td>
<td>Hydrophobic</td>
</tr>
</tbody>
</table>

Table 2.1: Twenty common amino acids found in proteins with their 3-letter and 1-letter code. The last column shows amino acid chemical properties as (+/-) charged, polar or hydrophobic.
All of these amino acids have similar structures, shown in Figure 2.2. They consist of a central carbon atom, called $\alpha$ carbon or $C_\alpha$, and four other functional groups, which are connected to this $\alpha$ carbon. These four groups are a hydrogen ($-H$) atom, an amino group ($-NH_2$), a carboxyl group ($-COOH$) and a side chain ($-R$). It is the different types of side chains that differentiate these twenty types of amino acids. Each type of side chain confers to each type of amino acid distinctive physical and chemical properties, a sample of which are also shown partially in Table 2.1.

To produce a protein molecule, tens or hundreds or even thousands of amino acids are covalently linked through the condensation reaction between the carboxyl group of one amino acid molecule and the amino group of another amino acid molecule. The resulting linkage is often called peptide bond and is one of the strongest chemical bonds known so far. The contiguous chain is also referred to as polypeptide chain. The free amino group of the peptide is called the $N$ terminus and is often referred to as the start of a protein. The end of a protein is the free carboxyl group of the chain.
and is also called \textit{C terminus}. Scientists sometimes refer to amino acids in proteins as \textit{residues} since they are no longer intact amino acid molecules. Proteins consist of different number of residues, ranging from around 40 to 2000 residues. Typical proteins have a length of approximately 200 residues.

\subsection*{2.2.2 Protein structures and functions}

Proteins are not merely sequences of residues lining up in one dimension, but rather have more compact structures that are essential in carrying out their biological functions. People study protein structures in four levels, namely \textit{primary}, \textit{secondary}, \textit{tertiary} and \textit{quaternary}, which are described in more detail below:

\textbf{Primary structure} The primary structure of protein only considers the sequence of the amino acid residues. When the one-letter code is used to describe a protein sequence, a string of letters from an alphabet of size 20 will be sufficient to represent the protein. Conventionally, the string starts from the N terminus to the C terminus when read from left to right. The advantages of such notation lie in its simplicity and capability of demonstrating the major features of proteins. However, the disadvantages are also obvious, as such representation of proteins ignores the interaction between nonadjacent amino acids. For example, the intramolecule sulfur bridge linking distant cysteine residues cannot be seen from the protein sequences.

\textbf{Secondary structure} The secondary structure of protein refers to the regular structure units often found in proteins, such as $\alpha$-helices, $\beta$-sheets and loops, which can be recognized by the shape and by the hydrogen bond patterns between the backbone atoms. Among these three common secondary structure units,
α-helices are segments of spirals, with 3.6 amino acids per turn on average. The diameter of a typical α-helix is around 10Å. They can act as anchors for membrane proteins or serve as the tunnels for ion channels. β-sheets consist of flat segments, sometimes called β-strands. Depending on the relative directions of the adjacent β-strands, β-sheets can be either parallel or anti-parallel. To form an antiparallel β-sheet, the protein chain has to fold back via a short turn called β-turn. In a parallel β-sheet, all β-strands are pointing in the same direction with respect to the N- and C-termini.

Loops, unlike the α-helices and β-sheets with well defined structural features, are often called random coils. They have no specific shape but a statistical distribution of all possible conformations. Loops usually connect between α-helices and β-sheets. However, loops are no less important than α-helices or β-sheets structures. As a matter of fact, some neurodegenerative diseases such as Huntington’s disease may be related to random coil formation in certain proteins.

The representation of protein secondary structures provides more information about the spacial arrangement of the amino acids and functional elements of proteins. It also highlights the correspondence between the sequence and the structural features.

**Tertiary structure** The tertiary structure of proteins is often called fold, referring to the 3D configuration of the proteins and is able to reveal the otherwise unseen long range interactions between residues. Many globular proteins are
usually folded into a generally compact, orderly three-dimensional conformation. The word *fold* is also used to indicate a structural family, within which given proteins share similar structures. Since there are countless possibilities for a protein sequence to adopt a particular arrangement in 3D space, but only certain conformations or even one unique configuration is biologically active, clues of proteins biological functions can be inferred by examining their tertiary structures.

**Quaternary structure** There are many proteins structures that consist of several polypeptide chains, which cannot form a stable structure by themselves. However, through cooperative interactions among all those chains, the assembled protein complex then becomes stable and biologically active. In addition to the tertiary structure of the subunits, the quaternary structure refers to such assembly of individual substructures into a multi-subunit complex. For example, proteins with quaternary structure include DNA polymerase, ion channels and hemoglobin. Protein complexes with two subunits are called dimers, three subunits trimers, and so on.

Different types of forces play different roles in defining each hierarchy of protein structures. For example, covalent bonds, which is a type of chemical bonds, bring together protein’s primary structure. Hydrogen bonds, on the other hand, are considered a type of electrostatic force and are much weaker than covalent bonds. The secondary structure that forms through hydrogen bonds is thus easier to break than the primary structure. Even weaker forces, such as hydrophobic interactions or Van der Waals force, maintain the tertiary and quaternary structures for proteins. One exception is the sulfur bridge formed between paired cysteine residues that are very
important in establishing the higher level structures. Interestingly, the evidence shows that the structure of a protein is ultimately determined by its amino acid sequence. The classical RNase refolding experiment has demonstrated this unique property of proteins.

Proteins participate in a wide range of biological functions, such as metabolism, signal transduction, immunization, function regularization, and etc. Enzymes are a particular type of proteins that can greatly speed up the reactions that may normally take days, months or even years to finish. This class of proteins usually only acts on specific substrates and has much higher specificity than other catalysts. Structural proteins, such as collagen, are the main components of cartilage, bones, teeth and skins. Regulatory proteins, such as cycling, regulate the life cycle of cells. Signaling proteins such as insulin and hormones, stimulate biological processes. Defensive proteins protect us from toxic or virus attacks. Practically all the molecular processes that happen in organisms are controlled by proteins.

During interactions with other substances, proteins usually identify their target through complementary shape, or favorable interactions between residues on their surfaces. Typically, only a small portion with limited number of residues are directly involved in contact with the reacting compound. This small portion of the protein is often called functional site or active site. Residues that are involved are known as functional residues. Mutant proteins that have mutation at the active site often loose their normal biological activity and cause adverse effects. For example, sickle-cell disease is caused by a single nucleotide mutation from A to T, which produces a valine (V) in place of a glutamic acid (E) in patients’ hemoglobin. Thus, those functional residues tend to be strongly conserved throughout the evolutionary history
and across different species. Some other residues that are not directly playing roles in the reaction may still be important in providing structural framework for the active site and may also conserved to a certain degree.

2.2.3 Protein homology

From Darwin’s evolutionary hypothesis, the proteins that exist today can be viewed as descendants of ancient proteins. During the history of evolution, proteins have evolved with the changing environment. Mutations that have altered the DNA sequences are propagated into the protein products. So people use the term “homology” to indicate that two given proteins share a common ancestor. However, it is not always possible to tell whether two proteins are homologous, as we are unable to trace back billions of years to verify that these two proteins indeed evolved from a common ancestor. It is possible though that we may still be able to conclude reasonably that the proteins are homologous based on their sequence similarity. For example, if they have almost identical sequences, it is most likely that these two proteins are descendants of an ancient protein. Unfortunately, sequence similarity is not a conclusive measure of homology, because there can be other explanations. For instance, short sequences may be similar just by pure chance or similar sequences have been chosen by nature to preferentially bind a particular substrate, such as transcription factors. Sometimes, people use percent identity or percent similarity to quantify the similarity between two sequences, but they are by no means a confirmation of homology and can only serve as indirect evidence to support or reject the homology hypothesis. Partial homology can arise from the fact that only a fraction of the sequences are derived
from a common ancestry, while the rest does not. This can happen when a segment of the coding frame of one gene is transposed and integrated into another gene.

There are two types of homology based on the mechanism that separated the two homologous sequences. If they were separated by speciation event, meaning that two proteins are from different species, they are called orthologs. On the other hand, if these two proteins were separated by gene duplication, they are called paralogs. Orthologs often carry out similar biological function, which is not always the case for paralogs. The reason may stem from the lack of selective pressure upon one copy of the paralogous gene, which can mutate relatively more freely and acquire new functions. For example, hemoglobins have four different subtypes, namely hemoglobin A, hemoglobin A2, hemoglobin S, and hemoglobin F, which are paralogs to each other. They all have the basic function of oxygen transportation, but they diverged slightly in affinity to oxygen to accommodate different environment. As an example, fetal hemoglobin (hemoglobin F) has a higher affinity to oxygen than adult hemoglobin as developing fetus need better access to oxygen from the mother’s bloodstream.

Proteins with similar sequences are often grouped into families to indicate possible homology relationship. One good example is the SCOP (Structural Classification Of Proteins) database, which classifies proteins based on their sequence and structure similarities into four levels:

1. **Class:** General classification of the structure content of proteins, such as \( \alpha \)-helices only proteins, \( \beta \)-sheets only proteins, \( \alpha + \beta \) proteins which have both \( \alpha \)-helices and mainly parallel \( \beta \)-sheets (as \( \beta - \alpha - \beta \) units), and \( \alpha/\beta \) proteins which contain both \( \alpha \)-helices and mainly antiparallel \( \beta \)-sheets (as separate \( \alpha \) and \( \beta \) domains)
2. **Fold:** Proteins have the same major secondary structures in the same arrangement and with the same topological connections. Proteins that were placed together in the same fold category may not have a common evolutionary origin, because the structural similarity could result from the physics and chemistry of proteins favoring certain packing arrangements and chain topologies.

3. **Superfamily:** Proteins have a probable common evolutionary origin. They often have low sequence identity, but possess structural and functional features that suggest a probable common evolutionary origin. For example, one superfamily consists of actin, the ATPase domain of the heat shock protein, and hexokinase.

4. **Family:** Proteins are clearly evolutionarily related. They typically have percent identity of 30% or higher. In the case of the absence of high sequence identity, similar functions and structures provide definitive evidence of common descent. For example, the family of globins have some members share only 15% sequence identity.

### 2.2.4 Applications of protein homology

Homology is a very important concept. Scientists use it to solve many problems. For example, vital proteins such as ribosome, DNA polymerase and RNA polymerase are present in many organisms, from the most primitive bacteria to the most complex mammals. By examining the conservation and variation of those homologous sequences from different species, scientists can build a phylogenetic tree and estimate the divergence time for each species. One basic assumption is that the core regions of the proteins are conserved across all lineages of species and perform similar
functions, while additional subunits for regulation and protein-protein interaction are added later in the evolution.

Given the close relationship between protein sequences and their structures, and the structures and their functions, homology information can provide clues for predicting protein structural and functional properties. Protein homology modeling is a good example. It combines sequence analysis and molecular modeling to predict the 3D structures of proteins based on homologous sequences with known structures. In this approach, finding the right homologous sequences and building accurate alignments are critical to the success of the method.

The homology concept can also help in annotating biological databases. Traditional protein characterization methods, such as SDS PAGE, chromatography, NMR, X-ray and Mass Spectrometry, are very accurate but far too slow to catch up with the large number of uncharacterized sequence data found in databases. As an example, around 30% of the entries in the non-redundant database are named putative, hypothetical or predicted proteins. Automated methods have been developed to assign meaning to those data based on their possible homology counterparts. Although such automated methods bear the risk of higher error rate, they can nevertheless provide some direction for further and more accurate characterization.

2.3 Background on sequence alignment

Scientists have developed many methods to detect the homology relationship between biological entities, such as DNA, RNA and proteins. Since it is relatively easier to obtain sequence information than structural and functional data, sequence alignment has been a fundamental technique in molecular biology for revealing similarities
among biological molecules. Its importance application has been increasing, given the explosive growth of the sequence databases.

Sequence alignment is the process of arranging sequences in such a way that regions of similarity are clustered together, which may indicate possible homology relationship among the sequences. Depending on the number of sequences being compared, sequence alignment is usually called pair-wise alignment for comparing two sequences and multiple alignment for three or more sequences. Figure 2.3 shows an example of multiple alignment of 7 sequences. In this example, there are several columns, such as column 4, column 9 and the last column, that contain the same type of amino acid, and we call them matches. Columns that contain different amino acids are then called mismatches, such as the first three columns. There are also columns that contain two additional symbols (“-” and “.”), which are used to indicate that amino acids are deleted (“-”) or inserted (“.”) in those columns. Deletion and insertion are often called gaps, since they introduce non-amino acid symbols into the alignment. If the aligned sequences are homologous, when interpreted from the evolutionary perspective, mismatches are point mutations in which one amino acid is substituted by another amino acid, deletions represent loss of amino acids for a protein sequence, and insertion means that the protein has acquired additional amino acids. In many cases, we do not care about which sequence came first in time as insertion and deletion are relative terms, so scientists also use the term indels to refer indifferently to either an insertion or a deletion. For example, deletions from one sequence can also be viewed as insertions in another sequences if the frame of reference is changed.
2.3.1 Overview of sequence alignment methods

When the sequence data was scarce, scientists were able to align very short or very similar sequences by hand. However, as sequence data become abundant and it became necessary to align lengthy, highly variable or large numbers of sequences, the time and complexity required to solve the task quickly scaled up beyond the capacity of sole human effort. To overcome this problem, scientists have turned to computers and a lot of effort has gone into designing and developing efficient algorithms and software tools.

The first theoretical framework for sequence alignment algorithms was established by Needleman and Wunsch in 1970 [40]. They studied the optimization of alignment between two sequences over their entire length. The technique they used is called dynamic programming by computer scientists. About a decade later, Smith and
Waterman [53] proposed a local optimization version of the Needleman-Wunsch approach, called the Smith-Waterman algorithm. It aims at finding the local similarities between two sequences instead of the global similarity. Thus the Needleman-Wunsch algorithm is also called global alignment algorithm and the Smith-Waterman algorithm is referred to as local alignment algorithm. Both algorithms use some kind of scoring matrix to assign a score to the pairing of the sequences and optimize over all possible scores. Figure 2.4 shows the BLOSUM62 matrix, which is constructed from a set of sequences with no more than 62% sequence identity. It is widely used as the default matrix in many sequence alignment packages. In common practice, the Smith-Waterman algorithm is more popular than the Needleman-Wunsch algorithm, as it is often the case that only a fraction of the sequences are very similar while the rest is unrelated. For example, one often finds a large number of introns which may be just random residues in genome structure analysis; similarly just a few regions are conserved on distantly related sequences just a few regions are conserved.

As the field continues to grow, many other approaches have been proposed, such as profile alignment [26, 34, 51], hidden markov models (HMM) [20, 28, 29, 21], neural networks [49, 48], Bayesian systems [61], genetic algorithms [12], voting algorithms [13], and etc. In addition, scientists have been trying to integrate information beyond pure sequence data to help improve the accuracy and sensitivity of the sequence alignment. For example, the secondary structure or tertiary structure data from either experimental results or theoretical predictions have been used in sequence alignment. Functional annotation data, such as ontology information and enzymatic data have been used in sequence alignment as well [46]. Meanwhile, different softwares have been developed to facilitate the use of sequence alignment algorithms and
|    | A | R | N | D | C | Q | E | G | H | I | L | K | M | F | P | S | T | W | Y | V | B | Z | X * |
| A  | 4 | -1 | -2 | -2 | 0 | -1 | -1 | 0 | -2 | -1 | -1 | -1 | -1 | -2 | -1 | 1 | 0 | -3 | -2 | 0 | -2 | -1 | 0 | -4 |
| R  | -1 | 5 | 0 | -2 | -3 | 1 | 0 | -2 | 0 | -3 | -2 | 2 | -1 | -3 | -2 | -1 | -1 | -3 | -2 | -3 | -1 | 0 | -1 | -4 |
| N  | -2 | 0 | 6 | 1 | -3 | 0 | 0 | 0 | 1 | -3 | -3 | 0 | -2 | -3 | -2 | 1 | 0 | -4 | -2 | -3 | 3 | 0 | -1 | -4 |
| D  | -2 | 2 | 1 | 6 | -3 | 0 | 2 | -1 | -1 | -3 | -4 | 1 | -3 | -3 | -1 | 0 | 1 | -4 | -3 | -3 | 4 | 1 | 1 | -4 |
| C  | 0 | -3 | -3 | 9 | -3 | -4 | -3 | -3 | -1 | -1 | -3 | -1 | -2 | -3 | -1 | -1 | -2 | -1 | -2 | -1 | -3 | -3 | -2 | -4 |
| Q  | -1 | 1 | 0 | 0 | -3 | 5 | 2 | -2 | 0 | -3 | -2 | 1 | 0 | -3 | -1 | 0 | -1 | -2 | -1 | -2 | 0 | 3 | -1 | -4 |
| E  | -1 | 0 | 0 | 2 | -4 | 2 | 5 | -2 | 0 | -3 | -3 | 1 | 2 | -3 | -3 | 1 | 0 | -1 | -3 | -2 | -2 | 1 | 4 | -1 | -4 |
| G  | 0 | -2 | 0 | -1 | -3 | -2 | -2 | 6 | -2 | -4 | -4 | 2 | -3 | -1 | -2 | 0 | 2 | -2 | -3 | -3 | -1 | 2 | -1 | -4 |
| H  | -2 | 0 | 1 | -1 | -3 | 0 | 0 | -2 | 8 | -3 | -3 | -1 | -2 | -1 | -2 | -1 | -2 | -2 | 2 | -3 | 0 | 0 | -1 | -4 |
| I  | -1 | -3 | -3 | -3 | -1 | -3 | -3 | -4 | -3 | 4 | 2 | -3 | 1 | 0 | 3 | -2 | -1 | -3 | -1 | 3 | -3 | -3 | -1 | -4 |
| L  | -1 | -2 | -3 | -4 | -1 | -2 | -3 | -4 | -3 | 2 | 4 | -2 | 2 | 0 | 3 | -2 | -1 | -2 | -1 | 1 | 4 | -3 | -1 | -4 |
| K  | -1 | 2 | 0 | -1 | -3 | 1 | 1 | 2 | 1 | -2 | -1 | -3 | 2 | 5 | -1 | 3 | -1 | 0 | -1 | -3 | -2 | -2 | 0 | 1 | -1 | -4 |
| N  | -1 | 1 | -2 | -3 | -1 | 0 | -2 | -3 | -2 | 1 | 2 | -1 | 5 | 0 | -2 | -1 | -1 | -1 | 1 | -3 | -1 | -1 | -4 |
| F  | -2 | -3 | -3 | -3 | -2 | -3 | -3 | -3 | -1 | 0 | 0 | -3 | 0 | 6 | -4 | -2 | -2 | 1 | 3 | -1 | -3 | -3 | -1 | -4 |
| E  | -1 | -2 | -2 | 1 | -3 | -1 | -1 | -2 | -2 | -3 | -3 | -1 | -2 | -4 | 7 | -1 | -4 | -3 | -2 | -2 | 1 | -2 | -4 |
| S  | 1 | 1 | 1 | 0 | -1 | 0 | 0 | 0 | 1 | -2 | -2 | 2 | 0 | -1 | -2 | -1 | 4 | 1 | -3 | -2 | -2 | 0 | 0 | 0 | -4 |
| T  | 0 | -1 | 0 | -1 | -1 | -1 | -1 | -2 | -1 | -1 | -1 | -2 | -1 | -1 | -2 | 1 | 1 | 5 | -2 | -2 | 0 | -1 | -1 | -4 |
| W  | -3 | -3 | -4 | -4 | -2 | -2 | -3 | -2 | -3 | -3 | -1 | 1 | -9 | -3 | 2 | 1 | 1 | -2 | 3 | -4 | 3 | -3 | -2 | -4 |
| Y  | -2 | -2 | -3 | -2 | -1 | -2 | -3 | -2 | -3 | -3 | -1 | 1 | -9 | -3 | -3 | 2 | 1 | 1 | -2 | 3 | -4 | 3 | -3 | -2 | -4 |
| V  | 0 | -3 | -3 | -3 | -1 | -2 | -2 | -3 | -3 | -3 | -1 | 2 | 1 | -1 | -2 | -2 | 0 | -3 | -3 | 4 | -3 | -2 | -1 | -4 |
| B  | -2 | -1 | 3 | -4 | -3 | 0 | 1 | -1 | 0 | -3 | -4 | 0 | -3 | -3 | 2 | 0 | -1 | -4 | -3 | -3 | 4 | 1 | 1 | -4 |
| Z  | -1 | 0 | 0 | -3 | 5 | -4 | 2 | 0 | -3 | -3 | 1 | -1 | -3 | -1 | -2 | 0 | 1 | -3 | -2 | -2 | 1 | 4 | -1 | -4 |
| X  | 0 | -1 | -1 | -1 | -2 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | 0 | 0 | -2 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |
| *  | -4 | -4 | -4 | -4 | -4 | -4 | -4 | -4 | -4 | -4 | -4 | -4 | -4 | -4 | -4 | -4 | -4 | -4 | -4 | -4 | -4 | -4 | -4 | -4 |

Figure 2.4: BLOSUM62 substitution score matrix
the interpretation of alignment results. The GCG package developed at Accelrys integrated many tools, including programs for sequence alignment, for comprehensive sequence analysis. BLAST, which is a collection of tools developed at NCBI, is probably the most widely used sequence alignment and database search program. All kinds of databases have also been established for sequence analysis, such as PFAM, SCOP, PROSITE, etc.

Although sequence alignment is now considered a mature technique, there are still many challenges facing researchers. For example, several studies have shown that many of the sequence alignment tools can only cover around 20% of the homology sequences in the databases [7, 29, 50], indicating that from the sensitivity point of view the sequence alignment algorithms have quite some room for improvement. In addition, there are still no good methods to identify homologous sequences with very low percentage of sequence identity. The continuing success in the search for homology relies on how far we can push the boundary of our detection power.

Often we are only interested in certain species and need to work with proteins of specific families. So it is usually desirable to build alignment scoring matrices that fit their unique requirements. However, to construct a high quality scoring matrix is a non-trivial task. A lot of knowledge and effort has to be put into building such elaborate scoring systems.

With the rapid increase of the number of the sequence databases and the size of those databases, it is increasingly a challenge to search such huge reservoir of sequence information to find what users need. The challenge is two folds. Firstly, users want the results as quickly as possible. Secondly, the results should be as accurate as possible. So there is a tradeoff between speed and accuracy. To speed up the searches, many
heuristics methods have been proposed, such as building indexes to locate identical matches, using filters to reduce the number of sequences being aligned, and employing more powerful computers and parallel search approaches. To improve the accuracy of the searches, a popular approach is to study the statistics of the alignment scores, and construct scoring matrices that are more specific to the sequence families.

For pairwise alignment, the exact dynamic programming technique can find the optimal alignment with a computational complexity of $O(N^2)$, where $N$ is the average length of the sequences. However, when more sequences are added into the multiple alignment, the time complexity for aligning them increases exponentially. Exhaustive dynamic programming can only find the optimal alignment in $O(N^k)$ time, where $k$ is the number of sequences. So it is clear that such approach cannot be used in practice. Consequently, many heuristics approaches and different alignment methods have been proposed [4, 58].

Most of the widely used alignment algorithms adopt some kind of markov process to model the alignment, and have become popular due to the relatively good results they can produce. However the markov process is not a strictly accurate biological model, since it only considers interactions between adjacent amino acids. As mentioned in section 2.2.2, the secondary and tertiary protein structures are maintained through hydrogen bonds, hydrophobic reactions and Van der Waals force, which involve distant amino acids. Thus many effort have been focused on other avenues, such as genetic algorithm and voting algorithms.
2.3.2 Sequence alignment algorithms

The Dot Plot method, presented by Gibbs and McIntyre in 1970 [19], can be used to align two sequences in a qualitative manner. It can easily visualize certain sequences features, such as exact matches, insertions, deletions, repeats or inverted repeats. Figure 2.5 shows an example of a dotplot between a tropomyosin sequence and itself. To draw such a dot plot, one sequence is placed horizontally along the bottom of a two-dimensional matrix, and another sequence is written vertically along the leftmost column of the matrix. Such matrix is usually referred to as alignment lattice. By comparing the corresponding amino acids at each point, a dot is printed with different intensity according to the similarity of the two residues. If there are regions between two sequences that are very similar, a diagonal line will be shown in correspondence of the regions. The advantage of dotplot is its simplicity and effectiveness in revealing many of the interesting sequence features. So scientists often draw a dot plot as the first step in sequence analysis. However, a dotplot is too simple for more advanced tasks. For example, the comparison is not quantified and may not always be consistent as it relies on experimenters to put gaps in order to achieve better alignment.

In searching for more effective and efficient alignment algorithms, Needleman and Wunsch [40] proposed a new algorithm using the so-called dynamic programming technique. This new algorithm was later named the Needleman-Wunsch algorithm. The dynamic programming approach first calculates the optimal solution for aligning the initial portion of the sequences, and then incrementally builds the optimal solution for progressively larger initial portion of the two sequences until the entire sequences are covered. Such technique greatly reduces the number of operations needed to
Figure 2.5: Dotplot between a tropomyosin sequence and itself. Lines along diagonals represent identical segments. Lines along the anti-diagonals are inverted regions. Short parallel segments around position 180 are the repeats.
find out the optimal solution for the whole sequence, since the brute force method would have a $O(3^N)$ time complexity, whereas dynamic programming only requires $O(N^2)$. In the optimization process, Needleman and Wunsch used a similarity matrix and a gap penalty to calculate a score for each candidate alignment, which acts as a measure of quality. The later Smith-Waterman algorithm proposed by Smith and Waterman takes a different approach. Instead of optimizing the alignment score over the entire sequences, it finds the local maxima. This method is particularly useful to align two distantly related sequences, since only part of the sequences is normally conserved over the evolution. The Smith-Waterman algorithm can also handle the cases where gene shuffle or transposition result in conserved domains within the whole sequence of proteins. Thus it have become the most popular algorithm used in sequence alignment.

The early substitution matrices are symmetric matrices and usually consist of 20 by 20 entries, since they only represent the relative likelihood of each amino acid changing to each other amino acid and it does not take into account where this substitution took place. Studies have shown that different regions of the same protein undertake different evolutionary pressure. For example, the outer loop region of a protein changes more frequently than the core region. So instead of using the 20 by 20 matrix, new matrices have been proposed with 20 by $L$ entries, where $L$ is the length of the protein. Thus the same amino acid substitution at different positions may produce different scores. This new kind of substitution scoring matrices are also called profiles.
Hidden markov models can be considered a profile alignment method as well, but they provide an elegant statistical framework, which allows users to ask many interesting questions, such as what is the probability of having a particular configuration of the alignment or what is the probability of aligning a particular residue at a given position.

Genetic algorithms model the alignment process in a way that is analog to the evolution of sequences. By simulating the insertion, deletion, substitution and even cross-over events on sequences, the initial sequence will produce offspring and evolve with some constraints. Those offspring have a certain probability to survive, mutate, and produce more descents. After many generations, an alignment can be obtained by comparing the initial and remaining sequences.

2.3.3 Alignment score statistics

Many sequence alignment tools usually assign a score to the alignment to indicate the closeness of biological relationship. Ideally, the larger the score, the closer in evolutionary time the sequences being compared. However, when searching a database, it is possible to achieve a high score by pure chance. Therefore, a more reliable quantitative criterion for estimating biological closeness of sequences is the $E$- or $p$-value, which assesses the statistical significance of obtaining a similar or better score from random sequences of same length as the assigned ones.

Iterative sequence alignment tools, such as PSI-BLAST or SAM, find a list of hits ordered by their scores and build a multiple sequence alignment based on pairwise sequence alignments between each of the hit in the list and the query sequence. A new scoring system is then derived from the multiple sequence alignment and is used
Gumbel Distribution

Figure 2.6: Gumbel distribution
for next iteration. The crucial step between iterations is deciding which of the hits to keep as putative members of the family (and thus to be later included in the multiple alignment) and which of the hits to reject as irrelevant. The common practice is to use a cutoff in the $E$-value or $p$-value.

$E$-value denotes the quality of an alignment as it is the average number of sequences that one would have expected if the sequences in the database were randomly chosen and thus they were unrelated sequences. Computing the $E$-value requires the statistical distribution of alignment scores of randomly chosen sequences to be known. For gapless alignment, i.e., alignment without gaps (insertions or deletions), the alignment score distribution of random sequences is known. It has been analytically proven [27, 28, 16] that the expected number of gapless local alignment of two sequences of length $M$ and $N$ with a score larger than $\Sigma$, i.e., the $E$-value, follows the universal form when $M$ and $N$ approaching infinity:

$$E(\Sigma) = KMNe^{-\lambda\Sigma}. \quad (2.1)$$

This form neither depends on the scoring parameters nor on the sequence model, i.e., the frequencies with which each amino acid appears in the random sequences, as long as only local alignments are considered. However, the two parameters $\lambda$ and $K$ do depend on the scoring parameters. The Karlin-Altschul theory [27, 28, 16] describes this dependence. Thus, an $E$-value can be assigned to a gapless alignment without any further need for computation, which made the original version of BLAST so successful.

Unfortunately in order to detect weak sequence homologies, gaps must be allowed [44]. According to many numerical studies [14, 36, 57, 2, 41], in the presence
of gaps the $E$-values still follow the universal form Eq. (2.1). However, the numerical values of the two parameters $\lambda$ and $K$ are not known.

There are various approaches to solving this dilemma. For large gap costs, approximate analytical formulas exist for $\lambda$ [37, 52]. For a small sub-class of scoring systems, an analytical formula for $\lambda$ that is valid for all gap costs [10] has been derived. The current version of PSI-BLAST uses a heuristic method to estimate $\lambda$ for different scoring matrices but at fixed gap cost [4, 50]. In addition, there are numerical approaches [8, 9] that rapidly determine $\lambda$.

However, all of these approaches are either heuristic or restricted to certain ranges of values of the alignment parameters. A possible escape route from this dilemma is an alternative alignment algorithm that has been proposed by Yu and Hwa [60]. The algorithm is called hybrid alignment since it is a combination of the Smith-Waterman algorithm and probabilistic schemes such as hidden Markov models. In hybrid alignment the score assigned to a sequence pair is obtained from the summation over all the possible alignments rather than from the most probable alignment as in the Smith-Waterman algorithm. Nevertheless the $E$-values are still calculated according to Eq. (2.1) with the parameter $\lambda$ taking the universal value $\lambda = 1$ that is completely independent of the scoring system. This simplification of the statistics does not decrease the sensitivity of the algorithm compared to the traditional Smith-Waterman algorithm [59]. The basic computational complexity of the alternative algorithm is the same as for the Smith-Waterman algorithm and it can be combined with heuristic schemes similar to the ones used in BLAST to reduce the computational effort. Most importantly, the theoretical prediction of the universal form Eq. (2.1) with $\lambda = 1$ holds even for position-dependent gap costs. This prediction has also been
numerically verified [59] for a large range of scoring systems with position-specific gap costs taken from the PFAM [5] database. The inability to calculate E-values when using position-specific gap costs is precisely the reason why PSI-BLAST does not allow them, in spite of the expectation that using a position-specific gap costs would increase sensitivity significantly. Thus, using hybrid alignment in PSI-BLAST would not only provide a theoretical basis for the calculation of E-values with the current fixed gap cost scoring systems but also enable us to utilize the suboptimal alignments and open the possibility of incorporation of sensitivity enhancing position-specific gap costs.

2.4 Survey of some existing sequence alignment tools

Since the adoption of computers in sequence analysis, a large amount of work has gone into the development of sequence alignment software. This is still an ongoing endeavor as users keep asking for faster, more accurate, and more specific tools to help them interpret their ever increasing amounts of sequence data. This section offers a survey on some of the most popular sequence alignment tools, discussing their underlying algorithms, advantages and possible pitfalls.

2.4.1 BLAST and FASTA

FASTA is probably one of the earliest sequence alignment software packages, which was first described by Lipman in 1985. It was originally named FASTP because it was designed for protein sequence alignment. In 1988, additional features such as DNA sequence alignment, and protein/DNA sequence alignment were added into the package and it was then renamed FASTA. The current FASTA package provides rapid heuristic methods for searching similar sequences in large database. The FASTA file
format is widely used by many other database search tools and sequence analysis softwares. FASTA implements a Smith-Waterman type of algorithm to search local sequence similarities among a query sequence and sequences in a database, which basically carries out the following four steps: a) find the pairs of identical letters, called words; b) rescore region without gaps around the words using a substitution score matrix, such as the PAM and the BLOSUM matrix and keep top scoring segments; c) eliminate segments that are unlikely to be part of the optimal alignment; d) use dynamic programming to optimize the alignment in a narrow band that cover the top scoring segments by “gluing” them with gaps. FASTA plots the initial alignment scores of each database sequence in a histogram and calculates the statistical significance of the scores. FASTA is freely available to the public and can be downloaded from ftp.virginia.edu/pub/fasta.

**BLAST**, which stands for **B**asic **L**ocal **A**lignment **S**earch **T**ool, is one of the most widely used bioinformatics programs. The advantages of BLAST are two folds: its emphasis on speed over sensitivity makes searching huge databases practical and its ability to answer fundamental questions that are of interest to users. BLAST was developed by Altschul et al. at the National Center for Biotechnology Information (NCBI). It has a web interface (http://www.ncbi.nlm.nih.gov/BLAST/) for users to specify their sequences to search for similar ones in popular online databases. Similar to FASTA, BLAST searches for high scoring local alignment between a query sequence and a database using a heuristic approach, which is an approximation of the Smith-Waterman algorithm. The exhaustive Smith-Waterman search is too slow when searching against very large databases, while the heuristic approach runs over 50 times faster with slightly reduced sensitivity. Generally speaking, the BLAST
algorithm can be divided into three stages: a) search for exact matches of a short segment between query and sequences in database, which has been optimized with indexing; b) extend the matches in both directions without gaps c) A banded extension with gaps resumes from where the gapless extension stops and the alignment is constructed by a variation of the Smith-Waterman algorithm. BLAST adopts a different approach than FASTA in determining the statistical significance of the results. Instead of referring to a empirical histogram, BLAST uses an equation to compute an $E$-value from the alignment score to indicate the average number of random sequences that would achieve such score or higher in the database.

Both FASTA and BLAST use approximations of the Smith-Waterman algorithm in their alignments, which sacrifice sensitivity for performance. However, the loss of sensitivity is minimized due to the efficient heuristics. On the other hand, only non-position specific scoring matrices are used by FASTA and BLAST, which put them to disadvantages when compared to other profile based methods, such as PSI-BLAST, HMMER and SAM.

2.4.2 PSI-BLAST

PSI-BLAST is an enhanced version of BLAST and was first described by Altschul et al. in 1997. PSI-BLAST stands for **P**osition **S**pecific **I**terative BLAST. It is a profile alignment tool, since a position specific scoring matrix (PSSM) is constructed from a multiple alignment of the selected list of hits in an initial BLAST search. The PSSM is then used to search the database for more similar sequences. PSI-BLAST also refines the profile through iterations when new sequences are added into the selected list. Because PSI-BLAST uses sequences from the same family to build the
scoring matrix in contrast to BLAST which only uses predefined non-position specific scoring matrix, PSI-BLAST is more sensitive than BLAST as it is able to find more distantly related sequences. However, a limitation on the alignment score statistics of the underline Smith-Waterman-like algorithm restricts it from using position specific gap cost.

2.4.3 HMMER and SAM

Along another line of approach, HMMER was developed by Eddy et al. in 1995 for searching weak sequence similarities in databases. It was rewritten later based on a new model architecture. HMMER employs a hidden markov model to represent the protein family, which can be viewed as another type of profile. The hidden markov model has a very nice statistical interpretation and consists of an emission probability matrix and a transition probability matrix. The emission probability matrix specifies the probability for an amino acid to be “emitted” from one position, while the transition probability matrix gives the probability of switching from one state to another state. In contrast to BLAST and PSI-BLAST, which start with one query sequence, HMMER starts with a multiple sequence alignment, then builds an HMM from the alignment, which is then used to search the database.

SAM is another hidden markov based approach to sequence alignment. SAM is the acronym for Sequence Alignment and Modeling System. As HMMER, SAM uses a linear hidden markov model to represent a multiple sequence alignment. But unlike HMMER, it first uses a query sequence to search a database for a list of potentially related sequences and then builds a HMM from them. Later on, this HMM is used to search the database for more weakly related sequences. It can also search the
database iteratively just as PSI-BLAST, but it is much slower. SAM was developed by Krogh et.al. at the University of California, Santa Cruz.

2.4.4 CLUSTALW

The above mentioned software packages are mostly used for pairwise sequence alignment or database searches. **CLUSTALW** was designed to construct multiple sequence alignment for a given list of input sequences. It builds the multiple sequence alignment in three main steps: a) pairwise alignments between every pair of the input sequences; b) creation of a phylogenetic tree from those pairwise alignments; c) construction of a multiple sequence alignment with the guide of the phylogenetic tree. CLUSTALW is a progressive and heuristic approach, because it expands the multiple alignment by adding sequences progressively and it does not guarantee that the final result is the optimal solution.
CHAPTER 3

HYBRID ALIGNMENT

3.1 Introduction

The hybrid algorithm was introduced by Yu and Hwa in their 2001 paper [60]. It is a combination of the optimization alignment approach exemplified by the Smith-Waterman algorithm and the probabilistic approach exemplified by hidden markov models. This is why the alignment given by the hybrid algorithm is also referred to as "semi-probabilistic alignment". When a sequence is aligned to the model, the hybrid algorithm calculates the probability of every possible path on the alignment lattice just as the probabilistic algorithms. But instead of summing over the likelihood of all the possible paths across the whole lattice, hybrid algorithm checks the summations of the likelihood of those paths that end at the same point and only picks the maximum value, which is an optimization process similar to the Smith-Waterman algorithm.

The underlying structure of the hybrid algorithm can be viewed as a hidden markov model. The detailed topology will be described in section 3.2. However, this HMM model basically contains a sequence of columns and each column has three states, namely Match (M) state, Insert (I) state and Delete (D) state. At
each Match state, the model can ‘emit’ (produce) residues according to the so-called emission probabilities. These probabilities correspond to the target probability proposed by Altschul et al. [2], which can be translated into substitution scores that are used in score-based alignments such as BLAST. Between two adjacent columns, there are a total of nine possible transitions from one column to another, namely, \( \{ M \rightarrow M, M \rightarrow I, M \rightarrow D \} \), \( \{ I \rightarrow M, I \rightarrow I, I \rightarrow D \} \), and \( \{ D \rightarrow M, D \rightarrow I, D \rightarrow D \} \). These transitions, except for the \( M \rightarrow M \) transition, characterize the indel (insertion/deletion) events and serve similar roles as gap costs in optimization alignment algorithms but with additional levels of control. For example, the gap opening costs dictate the probability to open a gap in either query or subject sequence, whereas the likelihood of opening up a gap just in query sequence is controlled by \( M \rightarrow I \) transition probability. On the other hand, the \( M \rightarrow D \) transition probability tells how likely is that the gap will be opened in subject sequence. Similarly, the \( I \rightarrow I \) transition probability gives the likelihood of extending an existing gap in query sequence. However, back to back gaps are normally not allowed in pairwise alignments, so the \( D \rightarrow I \) and \( I \rightarrow D \) transitions are not enabled at the same time, i.e. one of them needs to be set to zero. In our implementation of hybrid algorithm, we actually set both \( D \rightarrow I \) and \( I \rightarrow D \) transitions to zero, which disallows consecutive runs of gaps on query and subject sequences.

The salient feature of this combination of optimization algorithms and probabilistic algorithms is that the alignment score statistics for hybrid algorithm is better characterized than either of the approaches. The alignment statistics for the optimization algorithms is mathematically proven to obey the Gumbel distribution for gapless alignment, and even the Gumbel distribution parameters \( \lambda \) and \( K \) can be
calculated analytically. It can also be shown numerically that the same type of distribution still holds for alignment with gaps, however in this case the two parameters are no longer computable. Costly simulations are often needed to obtain their values, especially for the key parameter $\lambda$. On the other hand, even the shape of the alignment statistics for probabilistic alignment is not known, let alone the distribution parameters. On the contrary, the null statistics for hybrid alignment is in Gumbel form, and more importantly the key parameter $\lambda$ is fixed at value $1$ for many scoring systems, including the scoring system with position-specific gap penalties. We will discuss the hybrid alignment statistics in section 3.3.

By redefining sequence alignment in the statistical framework of the hidden markov model, hybrid algorithm has laid a solid theoretical foundation. At the same time, we can investigate many interesting questions using various statistical analysis methods, such as Bayesian statistics and the forward-backward algorithm. For instance, we can calculate the probability of aligning any residue at any given position, or the probability for any transition between two states. One crucial aspect of this method is that we can build a model with the input from some training sequences using the forward-backward algorithm and the posterior decoding method, which are the topics of section 3.4 and section 3.5.

### 3.2 Hybrid model topology

In this section, we will formally define the topology of the model for hybrid alignment, which is depicted in Figure 3.1. Let the length of the model be $L$. Then there are $L$ nodes in the model and they are numbered sequentially from $1$ to $L$. Each node consists of three states: Match state ($M$), Insert state ($I$) and Delete state ($D$).
Delete state is a silent state and there is no residue to emit. The Match state and Insert state are active states, i.e. they will produce residues according to their emission probabilities. The emission probability \( P_{j}^{M,e}(a) \) denotes how likely residue \( a \) will be emitted from the Match state at node \( j \), and \( P_{j}^{I,e}(a) \) characterizes such probability for the Insert state. Normally, we assume that residues are inserted randomly, so we use the background probability \( p_{a} \) for \( P_{j}^{I,e}(a) \). When calculating the statistics for hybrid alignments, we are not just interested in the raw probabilities of the events, but rather their likelihood with respect to the random events (background). Thus the quantities used for calculation in hybrid alignment are referred to as emission weights, which are the ratio of the emission probabilities to the background probabilities. For the Insert state, the emission weight for any amino acid is always 1. So whenever we use emission weights later, we will be only referring to the Match state. We also use \( q_{j,a} \) to represent the emission probability \( P_{j}^{M,e}(a) \) and \( \omega_{j,a} = q_{j,a}/p_{a} \) for the corresponding emission weight.

From the Match state of any node \( j \), the next stop can be the Insert state at the same node, or the Match/Delete state of the subsequent node (skipping nodes is not permitted unless the starting node is the Begin state, \( B \)). Transition probabilities are represented by \( \eta_{j}, \mu_{j}^{I2} \) and \( \mu_{j}^{D2} \) for \( M_{j} \rightarrow M_{j+1}, M_{j} \rightarrow I_{j} \) and \( M_{j} \rightarrow D_{j+1} \) transitions, respectively. Since we disable the transitions between Insert state and Delete state, other transitions from node \( j \) to node \( j+1 \) are \( I_{j} \rightarrow M_{j+1}, D_{j} \rightarrow M_{j+1} \) and \( D_{j} \rightarrow D_{j} \) with the corresponding transition probabilities \( \eta_{j}^{I1}, \eta_{j}^{D1} \) and \( \nu_{j}^{D} \), respectively. The transition \( I_{j} \rightarrow I_{j} \) loops at node \( j \) and the probability for such transition is \( \nu_{j}^{I} \). There are two dummy states: Begin state (\( B \)) and End state (\( E \)), which represent the start and end of alignments. The transitions from the Begin state to every Match state
Figure 3.1: Model topology for hybrid alignment. Black solid squares represent match states (M), Blue solid squares denote insert states (I) and red circles depict delete states (D). The diagram on the left shows the linear structure of the hidden markov model of the hybrid alignment. The diagram on the right shows the expanded 2D structure of the topology when a subject sequence is to be aligned with the query model. Two dummy states, i.e. the begin state (B) and the end state (E), are not shown.
serve as the free insertion module, which allows the alignment to start from anywhere within the model. There are transitions from every Match state to the End state, which does not restrict alignment to be finished at the end of the model. So the topology of hybrid alignment describes the local alignment, but at the meantime it is also very flexible as it can accommodate global alignment as well if the probabilities for all transitions from the Begin state except $B \rightarrow M_1$ and all transitions to the End state except $M_L \rightarrow E$ are set to zero.

When a sequence is to be aligned to the model, it will be much easier for us to see the alignment if we expand the topology along the sequence. So the expanded model becomes a two dimensional alignment lattice, which is shown on the right side of Figure 3.1. Figure 3.3 illustrates the correspondence between transition probabilities and elements of the lattice. Any alignment between the model and the sequence can be represented by a path on the lattice. One example is shown in Figure 3.2. The overall likelihood to obtain such alignment can be calculated by multiplying the corresponding emission weights and transition probabilities along the alignment path. It is obvious that this alignment example is a local alignment with respect to both the sequence and the model.

3.3 Hybrid alignment statistics

If we recall the introduction to the probabilistic alignment in Chapter 1, its alignment statistics of the probabilistic alignment, which is the distribution of the likelihood score, is not well characterized. Although it is suspected to be Gumbel distributed, there is no theoretical justification for this assumption. In fact, Yu and
Figure 3.2: Alignment lattice with an alignment path example
Hwa have shown numerically that the distribution of the likelihood score of the probabilistic alignment indeed does not follow the Gumbel form, and it has a flatter tail as compared to the Gumbel distribution. One possible explanation put forward by Yu and Hwa is that the score calculated by the probabilistic alignment is a sum of a large number of correlated terms, while the Gumbel distribution derives from the maximum of a large number of uncorrelated terms [60].

In hybrid alignment, a slight modification of the probabilistic alignment approach results in a much better characterized alignment score statistics. Since the key feature of the Gumbel distribution is to represent the maximum of a large number of uncorrelated terms, the hybrid alignment was designed to follow an optimization approach in that it calculates the maximum log-likelihood score $\Phi$.

$$\Phi = \max_{i,j} \{ \log Z_{i,j} \}$$

where $Z_{i,j}$ can be viewed as the local log-likelihood score, which is the sum of the likelihood scores of all alignment paths that end at the point $(i, j)$ on the alignment lattice. The local log-likelihood scores are statistically independent, i.e. they can be considered as uncorrelated terms.

Under some weak assumptions, the distribution of the maximum of these uncorrelated local log-likelihood scores is of Gumbel form as expected. By taking this optimization approach, one additional advantage is that the key parameter $\lambda$ of the Gumbel distribution is also known to be fixed asymptotically at value 1 for a wide range of scoring matrices and parameters. Thus computation-intensive simulations for fitting $\lambda$ are no longer necessary if hybrid alignment is used. The mathematical deduction of this powerful result can be found in Yu and Hwa’s paper [60].
3.4 Forward-backward algorithm

The forward-backward algorithm, which is also known as the Baum-Welch algorithm, was originally proposed by Baum [6] in studying natural language processing. It can be used to estimate the parameters of a hidden markov model (HMM). With the introduction of HMMs to the field of sequence analysis, the forward-backward algorithm has been widely used in many bioinformatics tools. Because of the close tie between hybrid algorithm and hidden markov models, the forward-backward algorithm can be used in hybrid alignment to learn model parameters. As the name suggests, the forward-backward algorithm consists of two parts: the forward algorithm and the backward algorithm. Both of them can be implemented by dynamic programming techniques just as the Smith-Waterman algorithm and the Needleman-Wunsch algorithm. In fact, the forward algorithm is rather close to the Needleman-Wunsch algorithm in that it replaces the "max" operation in the Needleman-Wunsch algorithm with the "sum" operation. It thus calculates the sum of the likelihoods of all alignments as opposed to just the probability of the optimal alignment. This change has a negligible effect if the probability of the optimal alignment is dominant, however, it is expected to capture more information if there are many suboptimal alignments with probabilities comparable to the probability of the optimal alignment. The latter case is very common when aligning two remote homologs. The forward algorithm computes the sum of the likelihoods of all alignments ending in node $j$ of the HMM and letter $i$ of the sequence for all pairs $(i,j)$. The backward algorithm uses a similar dynamic programming scheme as the forward algorithm to calculate the sum of the probabilities of all alignments that start at node $j$ of the HMM and letter $i$ of the sequence. Figure 3.3 shows a schematic diagram for calculating such likelihoods. By
Figure 3.3: A schematic diagram of the forward and the backward algorithms. For illustration purpose, some of the transitions are omitted. The backward algorithm executes backward along the reversed transitions of the forward algorithm.

Coupling the forward and the backward algorithms, the posterior probability of the occurrence of any amino acid at every model position and the transition probabilities can be estimated, a process called posterior decoding.

Here is a brief summary of the forward-backward algorithm in the context of hybrid alignment. Let $\Omega$ be the model of length $L$, and $X = (x_0 x_1 \cdots x_{n-1})$ be the sequence to be aligned with the model. The letter $x_i$ is chosen from a finite alphabet
On the alignment lattice, the forward algorithm can be used to calculate the total weights $f_{i,j}^S$ of all alignment paths reaching any state $S$ at any position $(i, j)$, where $S \in \{M, I, D\}$, $i = 1 \ldots n$, $j = 1 \ldots L$ via the following recursions:

\[

text{for } S = M:
\begin{align*}
    f_{i,j}^M &= 1 + \eta_j - 1 \omega_j(x_{i-1}) f_{i-1,j-1}^M \\
    &+ \mu^I_{j-1} f_{i-1,j-1}^I + \mu^D_{j-1} f_{i-1,j-1}^D
\end{align*}
\]

\[

text{for } S = I:
\begin{align*}
    f_{i,j}^I &= \mu^I_{j-1} f_{i-1,j-1}^M + \nu_I^I f_{i-1,j-1}^I
\end{align*}
\]

\[

text{for } S = D:
\begin{align*}
    f_{i,j}^D &= \mu^D_{j-1} f_{i,j-1}^M + \nu_D^D f_{i,j-1}^D
\end{align*}
\]

Here the constant 1 accounts for the possibility of starting the alignment in the middle of the model. The boundary conditions are: $f_{i,j}^M = f_{i,0}^M = 1$ and $f_{i,j}^D = f_{i,0}^D = 0$, where $i = 0 \ldots n$, $j = 0 \ldots L$.

The emission weight $\omega_j(x_i)$ is the ratio of the emission probability $q_j x_i$ of letter $x_i$ over its background probability $p_{x_i}$ at model position $j$, i.e., $\omega_j(x_i) = q_j x_i / p_{x_i}$.

The transition probabilities $\eta_j, \eta_j \mu^I_{j-1}, \mu^I_j, \eta_j \mu^D_{j-1}, \mu^D_j, \nu^I_j$ and $\nu^D_j$ satisfy the following constraints:

\[
\begin{align*}
    \eta_j + \mu^I_j + \mu^D_j &= 1 \\
    \eta_j \mu^I_{j-1} + \nu^I_j &= 1 \\
    \eta_j \mu^D_{j-1} + \nu^D_j &= 1
\end{align*}
\]

Using the forward algorithm we can find the position $(s_E, m_E)$ at which the local likelihood score $Z_{i,j}$ is maximized across the entire lattice, where $Z_{i,j}$ is defined as follows:

\[
    Z_{i,j} = f_{i,j}^M + \mu^I_{j-1} f_{i,j}^I + \mu^D_{j-1} f_{i,j}^D
\]

After we chose $(s_E, m_E)$ to be the end of the alignment, the backward algorithm can be used to calculate the total weight $b_{i,j}^S$ of all alignment paths starting from state.
S at position \((i, j)\) and ending at the point \((s_E, m_E)\), where \(1 \leq i \leq s_E, 1 \leq j \leq m_E\), and \(S \in \{M, I, D\}\) via the recursions:

\[
\begin{align*}
b_{i-1,j-1}^M &= \eta_{j-1} \omega_{j-1} (x_{i-1}) b_{i,j}^M + \mu_{j-1}^I b_{i,j-1}^I + \mu_{j-1}^D b_{i-1,j}^D \\
b_{i-1,j-1}^I &= \mu_{j-1}^I \eta_{j-1} \omega_{j-1} (x_{i-1}) b_{i,j}^M + \nu_{j-1}^I b_{i,j-1}^I \\
b_{i-1,j-1}^D &= \mu_{j-1}^D \eta_{j-1} \omega_{j-1} (x_{i-1}) b_{i,j}^M + \nu_{j-1}^D b_{i,j-1}^I
\end{align*}
\]

The boundary conditions are: \(b_{s_E,m_E}^M = 1, b_{s_E,m_E}^I = \mu_{m_E}^I, b_{s_E,m_E}^D = \mu_{m_E}^D\) and \(b_{s_E+1,j}^S = b_{s_E+1,j}^D = 0\), where \(i = 0 \ldots s_E, j = 0 \ldots m_E\).

### 3.5 Posterior decoding

Once the forward and backward weight matrices are calculated, we can perform various statistical analysis. For example, we can derive the probability of any letter \(a\) appearing at model position \(j\), which is computed as follows:

\[
Pr(a, j, X|\Omega) = \frac{\sum_{x_i=a} f_{i,j}^M \cdot b_{i,j}^M}{Z_{s_E, m_E}}
\]

The first term of the numerator denotes the sum of the weights of all alignment paths that end with letter \(a\) in sequence \(X\) at the \(j\)th position of the model, and the second term is the sum of weights of all alignment paths that start at this position and stop at the end of the alignment \((s_E, m_E)\). So the numerator represents the total weight of all the alignment paths that have letter \(a\) matched to the \(j\)th position of the model, whereas the denominator is the total weight of all paths. The ratio of the two is then the probability of having an alignment path that shows \(a\) at model position \(j\). If we have multiple sequences to be aligned to the model, the average number of occurrence
of letter $a$ at position $j$, $E_j(a)$, can be obtained by summing $Pr(a, j, X|\Omega)$ over those sequences, hence:

$$E_j(a) = \sum_X Pr(a, j, X|\Omega)$$

We can also ask for the probability of visiting a single state, say $M_j$, by calculating the following quantity:

$$Pr(M_j, X|\Omega) = \frac{\sum_i f^M_{i,j} \cdot b^M_{i,j}}{Z_{sE, mE}} = \sum_a Pr(a, j, X|\Omega)$$

Under the same rational, we are able to calculate probabilities involving transitions from state to state. For example, the average number of transitions from $M_j$ to $M_{j+1}$ can be derived as follows. First we sum over all the weights of alignment paths that have such a transition, which is $\sum_i f^M_{i,j} \cdot \eta \cdot \omega_j(x_i) \cdot b^M_{i+1,j+1}$. Then the probability of reaching $M_{j+1}$ from $M_j$ and the average number of such observed transitions are formulated as follows:

$$Pr(M_j \rightarrow M_{j+1}, X|\Omega) = \frac{\sum_i f^M_{i,j} \cdot \eta \cdot \omega_j(x_i) \cdot b^M_{i+1,j+1}}{Z_{sE, mE}}$$

$$Tr(M_j \rightarrow M_{j+1}) = \sum_X Pr(M_j \rightarrow M_{j+1}, X|\Omega)$$

Similarly, the probabilities of other transitions are computable. Please refer to the Appendix A for details. Thus the forward-backward algorithm will essentially construct a multiple sequence alignment using the underlying HMM as the template, which can be viewed as a generalization of the traditional multiple sequence alignment built from optimal pair-wise alignments. Techniques such as counting methods that operate on conventional multiple sequences can also applied to this generalized alignment to create new emission and transition probability matrices. A more elaborated discussion of this topic can be found in chapter 5.
4.1 Introduction

Yu and Hwa have laid the theoretical foundation for hybrid alignment [60]. They also investigated many properties of the hybrid alignment, including the finite-size correction and the alignment score statistics. The alignment statistics was initially verified by direct simulations for sequences of length between 100 and 1000. The scoring systems used in their study were two PAM substitution matrices and various affine gap penalties, which are defined by \( \text{Gap Cost} = c_1 + c_2 \cdot (l - 1) \), where \( c_1 \) is gap opening cost, \( c_2 \) is gap extension cost and \( l \) is the gap length. Subsequently, they have also numerically verified that the hybrid alignment statistics indeed follow Gumbel distribution with the \( \lambda \) asymptotically being a fixed value 1 for thousands of PFAM models.

In light of this very promising property of the hybrid alignment, there were attempts to apply it to search homologous sequences in databases. Yu and Hwa have touched this issue in their original paper about hybrid. They started with correlated sequences generated according to the model described in their 2001 paper [60]. This model is based on matrices of different PAM distances and various gap opening costs.
They showed that using their crude homology models the hybrid method performed comparably well to the Smith-Waterman algorithm, but significantly better than the Viterbi algorithm. Other studies have shown similar results for the implementation of hybrid alignment in BLASTP program, which is part of the BLAST package for protein sequence homology detection.

However, all of the previous results on sensitivity of hybrid alignment are in the context of non-position specific scoring systems. Recent studies have shown that profile based search algorithms are usually two to three times more sensitive than non-profiled based algorithms [21]. So it has been long expected that hybrid alignment will perform even better when position specific scoring systems are used for homology searches, since theoretical prediction points out that hybrid alignment statistics still hold in those situations. The main focus of our research has been to study the behavior of hybrid alignment in searching weakly related sequences using position-based substitution scores and position-specific gap costs.

We choose PSI-BLAST as the framework to study hybrid alignment in databases searches. There are several reasons to consider PSI-BLAST. First, it provides a natural testbed for us to implement hybrid alignment in the context of position-based scoring system. Second, we intended to make available hybrid alignment for sequence analysis to the research community. PSI-BLAST is one of the most widely used sequence database search tools, so the implementation of hybrid alignment in PSI-BLAST will make it immediately available to a large number of users. Third, we have some previous experience of the BLAST package, so it will take less effort to deal with PSI-BLAST code. Fourth, the PSI-BLAST is known for its speed and sensitivity in database searches, so we have a very good benchmark to compare against.
There are two main goals that we want to achieve for the study of the hybrid alignment in the framework of PSI-BLAST. First, we want to see how sensitive the hybrid alignment will be when it is coupled with position-specific scoring systems. Secondly, we would like to build position-specific scoring systems inside the PSI-BLAST code using various options provided by hybrid alignment, so that hybrid alignment can be used in a real world application for sequence database searches. We did not implement all the hybrid features in a single step, because the required changes to many aspects of the alignment code of PSI-BLAST was a nontrivial task. In order to make the process manageable, we carried out the implementation in three stages, which are described in detail in section 4.2. After each stage, we compared performance of the implementation of hybrid alignment to PSI-BLAST. The methods used for assessing the performance of various sequence database search algorithms are summarized in section 5.2.

4.2 Implementation stages

PSI-BLAST is a very big program composed of hundreds of source files and containing many blocks of functions. Figure 4.1 illustrates the execution flow of PSI-BLAST. In the first iteration of a PSI-BLAST run, the program takes some non-position specific substitution matrix, such as BLOSUM62, and uses it to perform a regular BLASTP search. This process involves the following major steps:

1. **Word finding**: All possible words of a predefined length $k$ are generated and only those words that have a score larger than some threshold $T$ will be kept. At default, the size of the word is 3 and the threshold $T$ is set to 11. Those
Figure 4.1: PSI-BLAST execution flow chart
selected words contain not only exact matches, but also close matches, which greatly increases the sensitivity of the program.

2. **Gapless extension**: Once the words are located on the alignment lattice, the program looks for any two such words on the same diagonal within a window of size $A$. Then a seed is chosen, which is usually in the middle of those two words, and the alignment is extended from the seed on the diagonal at both ends. Since the extension is restricted to a single diagonal, no gaps are allowed at this point. Gapless extension quickly identifies the “hot” spots, where regions of potential high similarity can be found.

3. **Extension with gaps**: The gapless extension will stop after the alignment score drops off to certain extent from the highest value obtained so far. The next step is to try further extensions where gaps are allowed. This step will permit the identification of not only highly similar segment, but also moderately or weakly related regions.

4. **Scoring and E-value calculation**: After the filtering, if a candidate alignment has a preliminary score larger enough to warrant reporting, it will enter the final scoring phase, where the expensive Smith-Waterman alignment is carried out between the sequence and the query. The final alignment score is used to obtain an $E$-value, indicating the statistical significance of this sequence during the search.

After the first run, PSI-BLAST chooses sequences with an $E$-value below an inclusion threshold and constructs a new substitution score matrix for the next iteration. This new score matrix is no longer a general purpose one like the BLOSUM matrices
and is instead position-specific. Ideally, the sequences chosen belong to the same protein family as the query, and thus it is crucial to choose only the right ones. The steps involved in building such new matrix are listed as follows:

1. **Backtracking**: Optimal pair-wise alignments are extracted from backtracking the alignment along paths on the Smith-Waterman matrix.

2. **Multiple sequence alignment**: Using the query as a template, all the optimal pair-wise alignments between the subject sequences and the query are stitched together, forming a multiple sequence alignment. Although this kind of multiple sequence alignment may not be as sophisticated as the one built by specific multiple alignment tools, it has been shown to work reasonably well.

3. **Building substitution scoring matrix**: Once a multiple alignment is constructed, the number of occurrences of every amino acid is counted and frequencies of amino acid occurrence are derived and converted to new substitution scores. This calculation is performed for each column and produces a vector of substitution scores for each position of the query sequence. Thus the new substitution matrix becomes the profile of the sequence family and captures position-specific features.

Previously, hybrid alignment has been implemented in BLASTP. However, such implementation only contained the data structures and functions for alignments with non-position based substitution score matrices and affine gap costs. So in order to enable hybrid to function in PSI-BLAST, our initial implementation involved adding necessary data structures for position-based scoring matrix and corresponding modification of the scoring functions. This represented the first stage whose aim was
to examine the interaction between hybrid alignment and PSI-BLAST. Once hybrid alignment could be stabilized in PSI-BLAST, we began to incorporate more features and to study the sensitivity of hybrid alignment in the context of position-specific scoring systems. In section 4.2.1, we describes the changes in our initial incorporation of hybrid alignment in PSI-BLAST. The second stage of our implementation is explained in section 4.2.2, which involved implementing the forward-backward algorithm and building of new position-specific emission weights matrix for hybrid alignment. The final stage is depicted in section 4.2.3 and involved the incorporation of position-specific transition probabilities and other issues related to building better models for protein families.

4.2.1 Stage I: Initial incorporation of hybrid algorithm in NCBI PSI-BLAST

We started our implementation of hybrid alignment in PSI-BLAST from Hybrid BLASTP, which is the previous work of Ralf Bundschuh. Hybrid BLASTP performs the sequence alignment without position-specific features. However, it provided us with a suitable starting point. With only slight modification of the code, we were able to add some position-specific features to the hybrid alignment in PSI-BLAST. In Bundschuh’s code, hybrid alignment had its own data structure to hold the non-position specific emission weights matrix, and there were specific scoring functions that made use of this data structure to calculate the scores of the alignments. In order to use a position-based emission weights matrix, we first added the corresponding fields into the dedicated hybrid alignment data structure. Then we modified those scoring functions so that they could use the position-specific emission weight matrix from the second round of PSI-BLAST runs. To fill this matrix, we had to get
the values from the substitution score matrix built by native PSI-BLAST code. So we also added one function to convert the PSI-BLAST position-specific substitution score matrix to hybrid position-specific emission weights matrix. The difference between these two types of matrices is trivial, since PSI-BLAST score matrix contains 

$$\frac{1}{\lambda} \log \frac{q_{j,a}}{p_a}$$

i.e. the log-odd likelihood of amino acid $a$ appearing at $j$th position scaled by the Gumbel parameter $\lambda$, where hybrid emission weight matrix contains only the likelihood $\frac{q_{j,a}}{p_a}$. Here $q_{j,a}$ is the target frequency of amino acid $a$ at position $j$ and $p_a$ is its background probability.

We kept the user interface of PSI-BLAST, but added a command line option “-x” to enable the hybrid alignment. The default value of this option is set to FALSE. So in order to invoke hybrid alignment for PSI-BLAST, users need to set it to TRUE using “-x T”.

During this stage of implementation, we were more focused on the compatibility of hybrid alignment with PSI-BLAST code. So only the scoring routines were changed. This modification would affect the selection of sequences that are entered into the model building phase and thus ultimately affect the sensitivity of the program. The backtracking and model building were still carried out by PSI-BLAST routines using its heuristic implementation of the Smith-Waterman algorithm. Figure 4.2 demonstrates the new execution flow, which has a branch at the scoring step to hybrid alignment.
Figure 4.2: Hybrid PSI-BLAST execution flow chart after stage I
4.2.2 Stage II: Utilization of suboptimal alignments in building emission probabilities

After the successful initial implementation of hybrid alignment in PSI-BLAST, we proceeded on to the second stage, in which we completely replaced the Smith-Waterman alignment with hybrid alignment. This stage was more complicated than stage I, since all the steps related to model building were involved, including backtracking, multiple sequence alignment, scoring matrix building and matrix post-processing.

In PSI-BLAST, the backtracking step determines the path that gives the optimal alignment of the sequence to the profile, and also gives the alignment score of the sequence. Then a multiple sequence alignment is constructed by aligning each optimal pairwise alignment of the selected sequences to the model. In the multiple alignment PSI-BLAST counts the number of occurrence of every amino acid at each position and calculates the amino acid target frequencies $q_a$.

Although it is often assumed that the optimal alignment represents the biologically relevant alignment between a pair of sequences, this is only true for sequences sharing a high degree of sequence identity, and is no longer necessarily true for distantly related sequences, in which case the biologically meaningful alignment can have score close but not equal to the maximum score. One possible explanation is that the scoring systems used for sequence alignment cannot integrate all the information of the properties of sequences. For instance, the construction of the BLOSUM and PAM matrices does not take the information about the tertiary structure into account. It has been shown that the accuracy of the structure homology modeling and sequence-structure alignment can be improved by exploring the space of suboptimal
alignments. However, the commonly used sequence alignment tools generally produce only the optimal alignment. In addition, the number of suboptimal alignments are enormously large, owing to the combinatorial nature of the problem. So it is not clear how to search the space of the suboptimal alignments. The hidden markov model approach provides an elegant statistical framework to deal with suboptimal alignments. As illustrated in section 3.4, each suboptimal alignment path in the alignment lattice is associated with a probability, indicating its statistical significance. Hybrid alignment score contains the contributions not only from the optimal alignment, but also from other sub-optimal alignments. So the hybrid alignment inherently exploits the suboptimal alignment information to build new models.

We first implemented the forward-backward algorithm in PSI-BLAST as described in section 3.4, but limited only to the estimation of new position-specific emission probability matrix. Although estimating new position-specific transition probability matrix was also one of our final goals, we deferred this part to the next implementation stage, since we expected the performance of Hybrid PSI-BLAST would already benefit from the incorporation of the information of suboptimal alignments. The non-position specific affine gap penalties used in PSI-BLAST were kept in Hybrid PSI-BLAST. The Smith-Waterman algorithm was then completely replaced by the hybrid algorithm in stage II of our implementation. The amino acid target frequencies were calculated with the forward-backward algorithm as described; we also adopted the pseudocount approach to deal with the sample incompleteness problem. A more detailed account of this approach is describe in section 5.6. In the new flow chart for Hybrid PSI-BLAST, that branch representing the hybrid version is extended as shown in Figure 4.3.
Figure 4.3: Hybrid PSI-BLAST execution flow chart after stage II
One problem emerged in the implementation of the forward-backward algorithm was the overflow problem. Because we deal with floating point numbers and likelihoods, whenever the dynamic range of those values goes beyond the range of the floating point representation of our computers, the overflow problem occurs. We found the problem occurs more frequently for sequences that are very long and similar to the model, since the alignment scores tend to increase along with the length. We could not simply rescale all the values periodically as we proceeded with the alignment, because the early part of the alignment score will be reduced to zero and all the related information would be lost, and this information was needed later for constructing the new scoring matrix. So we use an array of scale factors to keep the values within the range of the floating point representation without losing such information.

4.2.3 Stage III: Incorporation of position specific transition probabilities

Despite it is well known that different parts of protein sequences are not subjected to the same evolutionary pressure, NCBI PSI-BLAST only use non-position specific gap penalties due to the theoretical limitation of its underlying Smith-Waterman algorithm as mentioned in section 2.3.3. Consequently, the indel events are handled uniformly for different regions of the sequences. In the last and third stage of our implementation, we aimed at alleviating this limitation of PSI-BLAST by implementing position-specific gap costs, i.e. the position-dependent transition matrix. We first augmented the hybrid data structure with a matrix of size $7 \times L$, which is used to store the probabilities for seven allowed transitions at every position of a model of length $L$. Then the scoring functions were modified accordingly so that these position-specific values are used instead of the non-position specific gap penalties. However, it is not
guaranteed that position-based gap costs will always outperform their non-position specific counterparts, because wrong transition probabilities could be assigned to wrong positions. So one has to pay more attention to the values being used for these transition probabilities. We found that it was very hard to get the optimal values for every model position, since there is no established theory for calculating those values. We basically followed the formulas outlined in section 3.4. By incorporating this important feature into Hybrid PSI-BLAST, we expected to be able to build better models that can greatly enhance the sensitivity of the program.

We also contributed two important algorithm modifications in carrying out the implementation of the Hybrid PSI-BLAST. First, we changed the gapped extension part of the search heuristics that are used in PSI-BLAST to identify possible candidate sequences. Second, we generalized the sequence weighting scheme that is used to compensate for the sequence bias in databases.

The gapped extension needed to be changed because it is not appropriate to extend the seed at both ends. Since the alignment score for hybrid is the sum of the paths which only end at the same position, and if we were still extending the seed both forward and backward, some of the paths that go through the shaded region as shown in Figure 5.1 would not be counted, resulting in a lower score and potential loss of true hits. So we instead only extend forward from the position of the seed at first. However, we cannot afford to carry out a full scale forward alignment due to the computational complexity of such operation, we instead restrict the forward extension only to those high probability paths within a band. The technique details are describe in section 5.3. If the restricted alignment score exceeds some predefined threshold, this forward process is stopped and the sequence will be marked for further
analysis. Otherwise, the maximum score point of the forward extension is identified. Since this point is presumed to be near the real end of the true unrestricted alignment, a backward extension from the point is performed to take into account some of the paths that may not be covered in the previous forward extension. The resulting score is again compared to the threshold to determine whether this sequence is a potential hit.

The original PSI-BLAST uses a sequence weighting scheme similar to the position-specific weighting method proposed by Henikoff [23], which uses the multiple alignment built from optimal pair-wise sequences alignments. Instead of having only one residue at every position of the optimal alignment, hybrid contains a vector of values indicating the probability of the occurrence of any residue at each position. Thus we had to devise another weighting scheme that can work on such data representation of the multiple alignment. We chose to generalize the position-specific weighting approach and a detailed account is given in 5.5.
CHAPTER 5

RESEARCH OBJECTIVES AND METHODOLOGY

5.1 Introduction

During the development and implementation of Hybrid PSI-BLAST, we encountered a host of very interesting research problems, namely finite-size correction of the alignment statistics, sequence weighting for biased databases, database search heuristics and protein family model building. Some of those important topics will be covered in this chapter.

The comparison of the performance of different database search algorithms is of great value as it can be used to demonstrate the strength or weakness of sequence alignment tools. Reliable and accurate performance assessment is essential for developing new programs. We describe in section 5.2 our methods for analyzing the performance of our implementation of hybrid alignment in different stages. We also compared Hybrid PSI-BLAST with the NCBI version of PSI-BLAST (version 2.0) to see whether hybrid alignment indeed improves the sensitivity of the latter.

Due to different nature of the hybrid alignment algorithm and the Smith-Waterman algorithm that are used as the search engines in Hybrid PSI-BLAST and PSI-BLAST respectively, we had to develop new database search heuristics for Hybrid PSI-BLAST.
In section 5.3, the details of our new approach is given with some examples. We also analyzed the effectiveness of the new heuristic method and compared it to the old approach.

The Gumbel statistics only holds asymptotically when the sequences being compared are of infinite length, which is not true for real databases. Thus the correction of the alignment statistics becomes crucial for aligning moderately long or short sequences. We have studied two previously proposed finite-size correction formula for hybrid alignment and demonstrated that Hwa’s correction equation works better than Altschul’s formula. This topic is fully explored in section 5.4.

Sequences in the real databases are far from randomly distributed, but rather present different biases toward certain protein families. For example, there is a large number of entries for certain protein types, such as hemoglobins, zinc finger proteins and kinases; also, most proteins come from a small number of species, such as E.coli, mouse and human. These biases stem from the limitations of current technologies or from historical reasons. However, this becomes a real concern when high quality models are to be derived from biased databases. To get around this problem, we need a way to balance the amount of information derived from popular versus rare sequence families. In section 5.5, we show how we generalized the position specific weight methods originally proposed by Henikoff [23] and adopted it for Hybrid PSI-BLAST.

Methods that are used for constructing position specific scoring matrix are studied extensively in section 5.6. We have focus on the transition probability matrix estimation, as it is one of the primary goals that we wanted to achieve in this work.
5.2 Performance assessment

In order to accurately evaluate the performance of various sequence alignment algorithms or tools, it would be ideal to have a database in which all the relationships among sequences are known. But this ideal database does not exist because it is impossible to trace back millions and billions of years to verify the true evolutionary history of the proteins or other biological sequences. Thus, we can only approximate the true relationships by inferring homology from structural and functional clues found so far. There have been many studies that have attempted to achieve this goal and SCOP is among the early attempts to create such a database. SCOP classification largely relies on the judgment of human experts. In many ways, it has been considered as the gold standard of protein structure classification, and thus it has been used in many studies for evaluation of sequence alignment algorithms [7].

We, too, initially relied on the SCOP classification for performance evaluation of our hybrid alignment implementations. However, at one point we began to realize that the SCOP classification has its own drawbacks that had to be addressed in the evaluation process. We decided that it may be dangerous to depend solely on SCOP for evaluation in all circumstances, and as a result we started consulting other protein classification systems, such as CATH and DALI, when we detected ambiguities.

The actual databases we used are part of the ASTRAL dataset (ASTRAL SCOP, http://astral.berkeley.edu/), a set of databases derived from SCOP with filtering at different sequence identity percentage. We generally followed the method described in Brenner’s paper [7] for assessing the performance. The ASTRAL compendium that contains only sequences with less than 40% pairwise sequence identity [11], ASTRAL40, was used in implementation stage I and II. Since it contains sequences that
are considered to be only distantly related, we did not need to worry about the sequence weighting scheme that had not been implemented until stage III. Its relatively small size also allowed us to quickly evaluate the performance under many different parameters and conditions. In addition, the low sequence identity shared among these sequences made it a hard test for evaluating the sensitivity of our implementations.

We used ASTRAL version 1.59 to derive the ASTRAL40 database in stage I of the implementation, which consisted of 4383 sequences. Then we carried out an all-vs-all comparison using every ASTRAL40 sequence in turn as a query. We found that one specific sequence, namely the representative of the superfamily c.11.1, was consistently misclassified by all versions of PSI-BLAST (Hybrid and NCBI) for nearly all parameter choices. So we suspected that its true relationship may not be correctly reflected by the SCOP classification, and we excluded this sequence in our performance evaluation. We later found that the newer release of ASTRAL indeed changed the classification of that sequence to c.10.3, which is even a different fold from the old assignment.

There were then 4382 sequences used as queries, and 88171 pairs that share the same superfamily indicating true homology according to SCOP. After pooling together the reported hits and ranking them by $E$-value, we counted the number of true homologs and the number of non-homologs below various $E$-value cutoffs. For each $E$-value cutoff, the coverage was computed by dividing the number of true homologs by the total number of true relationships in the database, which indicates what percentage of homologies the program is able to detect in the database. On the other hand, the errors per query (EPQ), which is the quotient of the number of non-homologs and the total number of queries, tells us how likely is that the program
will make a mistake. The plot of EPQ versus coverage as a parametric function of the cutoff $E$-value demonstrates the tradeoff between the sensitivity and selectivity of the program. Such plot is also known as ROC (receiver operating characteristics) plot [7].

\[
\text{Coverage} = \frac{\# \text{ of true homologs below E-value cutoff}}{\# \text{ of total homologs}} \\
\text{EPQ} = \frac{\# \text{ of non-homologs below E-value cutoff}}{\# \text{ of queries}}
\]

A concern about the accuracy of the SCOP classification has been recently raised by some researchers [20]. Specifically, the concern is about some of the proteins that are classified in different superfamilies in SCOP, but might actually still be homologs. Clarifying such misclassification is critical as sequence comparison algorithms become more and more sensitive, and begin to actually find such very weak homologies. Under the old evaluation scheme, an algorithm that finds such weak homolog would be penalized since the sequences identified would be classified as errors. To circumvent this difficulty, some suggested that any hit which is in a different superfamily but in the same fold as the query should be ignored instead of being counted as a non-homolog, because we do not have sufficient information to make the judgment of whether such a hit corresponds to a true homolog or not. Here, we refer to this new counting method as conservative counting and the old way of counting errors as straight counting.

Considering the drawbacks of benchmarking on a small and very diverged dataset, we switched to another dataset, SUPER90 (version 1.69), for our later stage implementation performance assessment, because we needed more sequences to estimate the position specific transition probability matrix.
SUPER90 (version 1.69) is a subset of a database called SUPERFAMILY, which was developed by Gough, et.al. [20]. The SUPERFAMILY database is also derived from SCOP. The subset we used is filtered to contain sequences with no more than 90% sequence identity. There are a total of 10501 sequences in SUPER90 database, which contains more than twice as many sequences as the ASTRAL40 dataset.

5.3 Database search heuristics

Many sequence alignment algorithms are not directly suited for large database searches due to their quadratic time complexity, which is expressed as $O(MN)$ in computational notation. Here, $M$ is the length of the query sequence and $N$ is the length of the subject. When searching databases, however, $N$ is usually interpreted as the total length of all the sequences in the database. Exhaustive comparison between the query and every sequence in the database is not realistic and certainly not even an option in large scale sequence analysis. Consequently, many heuristic approximations of the sequence alignment algorithms are employed in the real database searches. BLAST is probably one of the most popular tools used for homology sequence detection in large databases. The success of BLAST is ultimately due to the excellent performance of its heuristic approximation of the Smith-Waterman algorithm. PSI-BLAST inherits the BLAST heuristics in its database search process. In the following paragraphs, we will briefly describe the steps involved in this heuristics approach.

The first stage consists scanning in the database for matches of every one of a collection of “words”; a word is a short, fixed length stretch of residues. This word finding process does not require an exact match between query and database sequences, instead any word that can match the query with its score larger than a
predefined threshold, $T$, will be selected. No gap is involved in this step, so the number of residues of a word, called word size, is the same in the query as in the database sequences and is a parameter that users can choose.

However, the number of words is usually still too large to be used directly. So a filtering process is carried out in the second step. Only those words which have a companion within a certain window of size $A$ and both have the same offsets along the query and database sequences are identified. Graphically, this word and its companion occupy the same diagonal in a dotplot and they are less than $A$ residues away from each other. Those sequences for which such a word pair cannot be identified are discarded from further investigation. Then the middle point between these two words on the same diagonal is marked as a seed, and PSI-BLAST tries to extend the seed along the diagonal in both directions without introducing any gaps. Thus this step is called gapless extension. Such extension is stopped once the alignment score drops off from the highest score by a certain amount, $X_1$. Database sequences that have diagonals with good enough gapless extension scores are marked for final scoring.

We cannot discard the remaining sequences, because the extension without gaps is very restrictive and it is very likely that many potential homologous sequences do not have such an uninterrupted high scoring segment. So a gapped extension is resumed from where the gapless extension stops and proceeds by exploring nearby diagonals with gaps allowed. In the gapped extension process, a new dropoff threshold is adopted. Sequences that pass through this additional filtering process will be added to the list of sequences selected for final scoring.

As demonstrated by the great success of BLAST program, the heuristics described above fits very well to the Smith-Waterman type of alignment, because only the
optimal alignment path is to be identified. However, it is not clear whether this heuristics would work for the hybrid alignment which considers not only the optimal alignment, but also the suboptimal alignments. If one looks at the alignment lattice shown in Figure 5.1, there are candidate alignment paths in the upper left and lower right regions that will be ignored by the BLAST heuristics.

Figure 5.1: A schematic drawing about PSI-BLAST heuristics for extension on a seed. The gapless extension starts from the seed and extends on the diagonal to both directions. It then stops when the alignment score falls off the maximum score obtained so far by more than a certain threshold. Then the extension resumes with gaps allowed. Such extension will search other diagonals.
In Hybrid BLAST, the hybrid alignment program first developed by Bundschuh, this problem was circumvented by increasing the gapped extension score artificially by 10% to compensate for the possibility of alignment paths in the uncovered region. Although obviously this was a rather ad hoc remedy, Hybrid BLAST performed quite well compared to the original BLAST. We were not satisfied by this solution, so we designed a new heuristics for the gapped extension in Hybrid PSI-BLAST. In the new approach, the extension begins with a band of width 1 and extends forward along the diagonal. At each step of the extension, this search tries to expand the width by one unit at both edges of the band as long as the value at the border does not fall off from the current maximum value by a certain amount. One example of the resulting band is shown in Figure 5.2. This “soft-banded” search gradually increases the band size and adjusts the bandwidth to follow the dominant path on the alignment lattice. Once the alignment score (the maximum value within the band) exceeds the cutoff value for reporting, we will stop the extension and mark the sequence for further processing. Otherwise, the position with the highest forward score is identified as the end point and a backward search is executed from that point. Similarly, the backward search is also performed within a band, but the backward search has a larger dropoff threshold, which makes it easier to expand the band than that in the forward extension. One important assumption is that at least one of the end points obtained from the forward extension of the seeds comes close to the real end point and the score obtained in the extension is a good approximation of the real alignment score.

In section 6.4, some preliminary results are given to demonstrate the usefulness of this new heuristics approach for hybrid algorithm.
Figure 5.2: An example of the “soft-band” forward search heuristics. The score landscape is calculated from a full scale forward alignment. A seed is identified at (30, 14) on the alignment lattice. The thick blue line is the boundary of the gapped extension, which adjusts its position along the dominant alignment path. It also locates the correct end point at (128, 140), which is even on a different diagonal as the seed.
5.4 Finite-size correction

Many sequence alignment statistic theorems make the simplifying assumption that the sequences being aligned are of infinite length. However, the real biological sequences in the databases only contain finite number of residues. BLAST uses a finite-size correction in calculating the $E$-value of a hit in the database searches. This correction is particularly important when either sequence is short. One simple explanation for the need of the finite-size correction comes from the fact that usually it takes an alignment a certain length to achieve a certain score. So in order to obtain a high scoring path, it is less likely that an alignment will start near the edge of the alignment lattice. Consequently, this effect is also referred to as edge effect. Since the effective area in the alignment lattice is reduced because of the edge effect, the tail of the alignment score distribution falls more quickly than in the Gumbel distribution. Thus the $E$-value calculated according the following canonical equation overestimates the true $E$-value:

$$E(\Sigma) = KMNe^{-\lambda\Sigma}. \quad (5.3)$$

To make the BLAST $E$-value calculation more accurate, Altschul and Gish [51] proposed the finite-size approximation that has later been extended by Altschul, Bundschuh, Olsen, and Hwa [1] to read for a sequence pair of length $N$ and $M$, respectively:

$$E(\Sigma) = K \left[ N - \left( \frac{\lambda\Sigma}{H} + \beta \right) \right] \left[ M - \left( \frac{\lambda\Sigma}{H} + \beta \right) \right] e^{-\lambda\Sigma} \quad (5.4)$$

It is obvious that the effective alignment area is indeed reduced in the new equation and is now a function of the alignment score $\Sigma$, the relative entropy $H$ and the
offset of the scoring matrix $\beta$. If we recall that relative entropy $H$ represents the average score per position along the alignment path, then $(\frac{\lambda \Sigma}{H} + \beta)$ can be viewed as the average alignment length $\bar{l}$ for score $\Sigma$, and the total available alignment lattice is adjusted by taking away the area near the edges that cannot accommodate alignments of such score. However, this adjustment cannot be applied to all the scores, since the alignment area would be negative if the score becomes large enough, which certainly makes no sense. More importantly, the relationship between the average alignment length and the relative entropy is no longer valid when the alignment score is very large, because there will be only few alignment paths that can achieve such disproportionally large score. So in practice, the finite-size adjustment is calculated for the score $\Sigma^*$ that would result in an $E$-value of 1.

An alternative empirical formula used by Yu and Hwa [60] modifies the parameter $\lambda$:

$$E(\Sigma) = K(N-\beta)(M-\beta) \times e^{-\lambda_{\text{eff}}\Sigma}$$

(5.5)

$$\lambda_{\text{eff}} = \lambda \left\{ 1 + \frac{1}{(M-\beta)H} + \frac{1}{(N-\beta)H} \right\}$$

Analytical approaches to the edge correction problem [35, 55] are confined to alignment without gaps. Even in the absence of gaps, they only give corrections to Eq. (5.3) that are equal to $\lambda \Sigma / [(N-\beta)H]$ in the first order. Both correction formulas (Eqs. (5.4) and (5.5)) coincide up to first order term, which is equal to $\lambda \Sigma / [(N-\beta)H]$. Thus, the analytical results are not suited to distinguish one correction formula from the other even in the absence of gaps.

The equivalence of the two correction formulas up to first order term that are equal to $\lambda \Sigma / [(N-\beta)H]$ is also the reason why the existence of different formulas was not an issue for the conventional PSI-BLAST. For the default scoring system of
PSI-BLAST, i.e., the BLOSUM62 scoring matrix \[22\] with cost of $11 + k$ for a gap of length $k$ and the amino acid frequencies of Robinson and Robinson (1991) \[47\], the parameters are estimated to be $\lambda \approx 0.2670$, $K \approx 0.042$, $H \approx 0.14$, and $\beta \approx -30$ \[1\]. At a database size of $M = 10^6$ amino acids and a query size of $N = 100$ amino acids an $E$-value of one corresponds to a score of $\lambda \Sigma \approx 15$. Thus, the first order correction is $\lambda \Sigma / [(N - \beta)H] \approx 0.77$ which is sizable but still smaller than one (i.e., the correction is smaller than the leading term and higher order corrections are expected to become even smaller.)

The situation in hybrid alignment is different. For the same scoring system, the parameters are estimated as $\lambda = 1$, $K \approx 0.3$, $H \approx 0.07$, and $\beta \approx -50$. The larger value of $K$ implies that an $E$-value of one now corresponds to a score of $\lambda \Sigma \approx 17$. More importantly, due to the smaller value of the relative entropy $H$, the first order correction is $\lambda \Sigma / [(N - \beta)H] \approx 1.6 > 1$. Thus, the second order correction contributes significantly to the $E$-value. Therefore, for hybrid alignment it is important to determine which of the two formulas more appropriately describes the length dependence of the $E$-values, or if yet another formula has to be worked out. The detailed experiments are described in section 6.2, where we report the results from empirical data and our conclusion on the choice of the finite-size correction formula for hybrid alignment.

### 5.5 Sequence weighting

It is well known that there are certain proteins that are easier to sequence, or that have been the focus of intensive research in the past. Therefore databases tend to be biased toward such proteins, either containing many copies of the proteins in various
mutations, or a large number of such proteins across many species, or even various engineered versions. For example, hemoglobins have been studied extensively for decades and the globin proteins have more entries in databases nowadays than most other protein families. This situation poses particular problems for iterative search schemes such as PSI-BLAST, which try to build and refine protein family models from a list of proteins found in the database. Because of this bias in the databases, well represented members can flood this list and skew the resulting model toward its characteristics.

This problem is addressed by sequence weighting techniques. Each sequence in the list is assigned a weight, which aims to balance the representation of that protein family. Usually, proteins that are very close to each other and are also present in large numbers will be assigned a smaller weight than proteins that are more dissimilar and fewer in number. PSI-BLAST adopts a weighting scheme similar to the position-based weighting method proposed by Henikoff [23] in consideration of the simplicity and efficiency of the method. This method relies on the multiple alignment $M$ built by the Smith-Waterman algorithm. A two dimensional array is used to record the position and type of the letters appearing in the optimal alignment of every training sequence. For each position $C$, PSI-BLAST constructs a multiple alignment block $M_C$, which is reduced from the raw multiple alignment $M$ by only including set of sequences $R$ that contribute residues to this position and columns of $M$ in which all the sequences of $R$ are represented. Inside $M_C$, the number of different residues, $r_j$ that occur at every column, $j$, is counted, as well as the number of times, $s_{j,a}$, that any given residue $a$ has occurred in that column. The individual weight assigned to a sequence with residue $a$ in column $j$ is computed as $w_j = 1/(r_j \cdot s_{j,a})$ and the weight
for the whole sequence at position $C$ is the sum of $w_j$ over all the columns in $M_C$.

Interestingly, $r_j$ is actually the normalization factor of $1/s_{j,a}$ over all the sequences in set $R$, i.e., $r_j = \sum_R 1/s_{j,a}$. This property is later used in our generalized version of the position specific weighting scheme.

For Hybrid PSI-BLAST, however, the situation is different, since hybrid alignment will produce a probabilistic alignment which contains information from all possible alignments, including the optimal alignment. So the final multiple alignment also has a probabilistic representation. The single value entry in the aforementioned two dimensional multiple alignment is now replaced by a vector of probabilities, indicating how likely a residue would occur in the multiple alignment. The normal position-based weighting scheme cannot be used directly. We then propose a generalized position-based weighting method for hybrid alignment as follows:

Suppose, we have the probability $f_{j,a,i}$ representing for each training sequence $X_i$ the occurrence frequency of any amino acid $a$, not just the amino acid contained in the optimal alignment. A three dimensional floating point matrix is created to hold these values calculated from the forward-backward algorithm as described in Appendix A. The probability of having an alignment at position $j$ for sequence $X_i$ is:

$$Pr(Alignment_j, X_i|\Omega) = \sum_a f_{j,a,i}$$

The contribution to residue $a$ at position $j$ by sequence $X_i$ is then the fraction of all occurrences of $a$ in the training sequence set:

$$C_{j,a,i} = \frac{f_{j,a,i}}{\sum_i f_{j,a,i}}$$
Table 5.1: Example of position specific weighting (PSW) and generalized position specific weighting (GPSW).

The contribution to position $j$ by sequence $X_i$, which is a generalization of $1/s_{j,a}$, can be calculated as the sum of the contributions from all the residues:

$$S_{j,i} = \sum_a C_{j,a,i}$$

The different number of residues $r_j$ defined in Henikoff’s position-based method can be generalized to: $r_j = \sum_i S_{j,i}$. The weight for sequence $X_i$ will become:

$$W_i = \sum_j \left( \frac{S_{j,i}}{r_j} \cdot Pr(Alignment_j, X_i|\Omega) \right)$$

For example, if we have such a multiple alignment shown below in Table 5.1 in both the Smith-Waterman type multiple alignment and hybrid type probabilistic multiple alignment:

We can notice that both weighting schemes conclude that the ‘F’ in the second column of the second sequence carries more information than that of the ‘D’ and the
'K' in the third sequence combined, because the 'F' is relatively rare than the 'D' and
the 'K'. Consequently, the second sequence is assigned a larger weight than the third
sequence. The first and forth sequence are equivalent in information importance, thus
being assigned equal weights by both schemes. However, generalized position-based
weighting gives the biggest weight to the second sequence, where normal position-
based weighting favors the first and the forth sequences. The reason is that generalized
position-based weighting considers that the 'F' is so unique with three 'Y's, comparing
to the equal appearance probabilities of 'V' and 'Q' in the third column or 'S' and
'G' in the last column.

5.6 Model building and regularization

To a large extent, the quality of the sequence models that are built by an homology
database search tool determines the performance in terms of the coverage of the tool.
If the models are too specific, only the closest homologous sequences will be identified.
On the contrary, the tool will run into the risk of falsely classifying unrelated sequences
as homologs if the models are too general. So building correct sequence models that
can accurately describe the properties of the whole family of the query sequence is
a delicate art. Currently, this is still the focus of intensive research. Nonetheless,
many ways to build reasonably good models have been proposed. The most common
approach among them is the counting-based model building method.

If we recall the construction of the famous PAM and BLOSUM matrices, we will
find that the first step is to build a multiple sequence alignment of related sequences.
Then the number of each possible substitution amid the multiple sequence alignment
is counted, and the frequency of each possible substitution is calculated. The result-
ing matrix gives us an empirical estimate of the propensity of an amino acid to be
substituted by another during the evolutionary history. Such matrix then can be used
as an objective function to assess the likelihood of certain sequences to be homologous
to each other.

PSI-BLAST basically follows this same strategy to build its models. It first collects
a list of sequences that are thought to be relatives of the query, then it tries to build a
multiple sequence alignment from the optimal pair-wise alignment between the query
and each sequence in the list. Instead of integrating the counts over all the query
positions, it calculates the frequencies of the substitutions for each position and build a
position-specific substitution scoring matrix. There are many considerations involved
in this seemingly simple process. One important issue is the sequence weighting
problem, which has been addressed in section 5.5. Another more subtle issue is the
question of how to count the unseen substitutions. Clearly, during the database
search process, not all the possible members of the family in the database will be
identified, let alone those sequences that for a reason or another were not included
in the database. Consequently, it is very likely that some of the substitutions will
not be represented in the multiple sequence alignment and thus will not be counted
toward the frequency of the substitution.

To circumvent the problem, the so-called pseudocount approach is often adopted.
The main idea is to add counts to the substitution events no matter whether they
are actually observed in the multiple sequence alignment or not. How to add these
counts and how many should be added depends on the what people believe a priori.
In PSI-BLAST, the pseudocount is added in the form of a mixture of the observed
frequencies and the frequencies embedded in the BLOSUM matrix, which implies that its developers assume those unseen substitutions took place according to the probability entailed in the BLOSUM matrix. Many other tools use a more complicated Dirichlet mixture instead of the simple pseudocount method to allow finer tuning of the a priori knowledge.

In our hybrid implementation of PSI-BLAST, we also had to address those issues. Since we replaced the Smith-Waterman algorithm in PSI-BLAST with the hybrid algorithm, we no longer have the optimal pair-wise sequence alignments nor the multiple sequence alignment built from them. On the other hand, hybrid alignment provides more information than just the optimal pair-wise sequence alignments. It does not only tell us whether a certain amino acid would appear in some position, but also enables us to calculate the probability of seeing such amino acid at this position. The approach we used to build the model at each iteration without the multiple sequence alignment is detailed in the next two sections.

### 5.6.1 Emission probabilities estimation

In PSI-BLAST, the underlying Smith-Waterman algorithm can produce the optimal pair-wise sequence alignment between the query and each of the selected sequences, which is then used to build a multiple sequence alignment by using the query as the template. At any query position $j$, the number of occurrence of each amino acid $N_a$ is counted and the observed frequency $f_a$ is calculated. This frequency is then mixed with the BLOSUM matrix $s_{a,b}$ to form the pseudocount $g_a$:

$$f_a = \frac{N_a}{\sum_a N_a}$$

$$g_a = \sum_b \frac{f_b}{p_b} q_{a,b} \quad (5.6)$$
where the $q_{a,b}$ are the target frequencies implicit in the BLOSUM matrix and $q_{a,b} = e^{\lambda s_{a,b}}$.

The final target frequency is then computed as follows:

$$q_a = \frac{\alpha f_a + \beta g_a}{\alpha + \beta}$$  

(5.7)

According to Karlin and Altschul’s theory, the new substitution score matrix is calculated as proper scaled log-likelihood: $s_a = \frac{1}{\lambda} \log \frac{q_a}{p_a}$.

For hybrid algorithm, we only need to calculate the target frequency $q_a$, which is also called the emission probability. However, unlike the Smith-Waterman algorithm, hybrid algorithm does not provide us the optimal pair-wise alignment between query and a subject sequence, thus we do not have an integer count $N_a$ for the occurrence of any amino acid, but rather a vector of 20 probabilities $v_{x,a}$ for aligning amino acids in subject sequence $x$ to the query at position $j$. The formula that can be used to compute $v_{x,a}$ have been detailed in section 3.5. Then by adding up the probability for amino acid $a$ at query position $j$ over all the subject sequences, we obtain the observed frequency $f_a$.

$$f_a = \frac{\sum_x v_{x,a}}{\sum_{a,x} v_{x,a}}$$  

(5.8)

Using the same pseudocount formulas as PSI-BLAST does in Eq. (5.6), we can finally derive the emission probability $q_a$ for the hybrid algorithm.

In the optimal pair-wise alignment, gaps in the query sequence are ignored by PSI-BLAST in constructing the multiple alignment. Gaps that appear in the subject sequences are counted, but distributed as counts of amino acids according to the their background probabilities. The assumption is that amino acids are inserted
randomly. We followed the same assumption and distributed the gap probability to the frequencies of amino acids by their background probabilities.

Although the quality of the emission probabilities derived from hybrid alignment is probably inferior to the models constructed by dedicated multiple sequence alignment, we did not attempt to try additional measures to polish the emission probabilities, because our primary focus is to investigate the sensitivity of hybrid algorithm with the adoption of position-specific gap costs. Having a mechanism to obtain reasonably good emission probabilities provides us a starting point to pursue the main goal of this thesis.

5.6.2 Transition probabilities estimation

The traditional approach to modeling the position-specific indel events is similar to modeling the substitution events, i.e. constructing a multiple sequence alignment and counting the transitions. Facing the same problem as in calculating the emission probability matrix, we can no longer count the transitions from hybrid alignment, since there is no conventional multiple sequence alignment available. However, due to the probabilistic nature of hybrid alignment, we are able to calculate the probability of each transition at every position using the forward-backward algorithm. The formulas are outlined in Appendix A.

The transition profiles built from the forward-backward algorithm were focused to follow the general patterns of the transition profiles of SUPER90 models, which are constructed by a more dedicated model building tool, HMMER. However, hybrid models tend to have smoother peaks than SUPER90 models do (Figure 5.3). One possible reason is that the human experts who construct the multiple alignments for
Figure 5.3: A example of the $M \rightarrow M$ probabilities in a model (of sequence d1kzfa_4) from SUPER90 v1.69 dataset and the corresponding model built by the forward-backward algorithm. There is an abundance of structures in the SUPER90 model as demonstrated by the peaks. The peaks derived from the forward-backward alignment are fewer in number and not as deep as those in SUPER90 model.

SUPER90 models are more likely to align gaps together instead of spreading them over several positions.

To address the problem of incomplete sequence samples for model building, we adopted the same pseudocount strategy we used in calculating the emission probability matrix. For example, the equation below is used to compute the new transition
probability for $M \rightarrow M$ transitions:

$$\eta' = \frac{\alpha \eta + \beta \eta_0}{\alpha + \beta}$$

where $\eta$ is the value obtained from the forward-backward algorithm and $\eta'$ is the new transition probability.

As for the transition pseudocount $\eta_0$, we have at least two choices. One option is to use the transition probabilities derived from the gap opening and extension costs specified in the command line arguments. Thus we just assume that no preference is given as to where the indel events occur; it turns out that the resulting transition profiles are quite smooth and possess few pronounced structures. An alternative option is to use the transition probabilities from the previous round as the pseudocounts, which will reinforce the signal and produce transitions with larger peaks. However, the risk is that noise could also be amplified and potentially yield wrong peaks. We tried both approaches and find that the latter option gives better results as shown in Figure 5.4.

Transition probabilities are generally much harder to model than emission probabilities. First, the indel events are generally fewer than substitutions, which may lead to less data for deriving the model parameters. Second, the rules that govern the occurrence of gaps are not fully understood and thus difficult to express mathematically. The above pseudocount approach is still not entirely satisfactory. Since there is currently no well established procedure for modeling gaps, we experimented with many postprocessing techniques to further refine the transition probabilities to see
Performance on different pseudocount choices

In calculating new transition probability matrix

Figure 5.4: Using transition probabilities from the previous round as the pseudocount $\eta_0$ works better than using the non-position specific values given in the command line. For this experiment, the ASTRAL40 dataset was used.
which worked best with the hybrid alignment. The methods we have experimented with are described below, and the corresponding results are described in Chapter 6:

**Two-level gap costs**

We first tried a two-level gap opening cost scheme. Basically, we set the target transition probability of \( M \rightarrow M \) to a low value to make it easy to open gaps for the positions whose raw transition probability \( M \rightarrow M \) was below a certain threshold, say 0.88, otherwise this transition was set to a high value to discourage the opening of gaps. To reduce the degree of freedoms, we set \( M \rightarrow I \) and \( M \rightarrow D \) transitions to be equally likely. For example, if the raw transition probability for \( M_j \rightarrow M_{j+1} \) is less than 0.88, we will set \( P(M_j \rightarrow M_{j+1}) \) to 0.4, and \( P(M_j \rightarrow I_j) \) and \( P(M_j \rightarrow D_{j+1}) \) both to 0.3. If the raw transition probability is no less than 0.88, they are set to 0.95, 0.025 and 0.025, respectively. This is a very simple and crude scheme of position-specific gap penalties, but it nevertheless provides us some flexibility in modeling indel events.

**Capping the Transitions**

We noticed that the \( M \rightarrow M \) probabilities of the majority of the model positions computed by the hybrid alignment are close to 1, which essentially makes it very hard to open a gap at those positions. However, we expect that more positions will be opened up for indel events when sequences diverge away from their ancestors. In conjunction with the consideration of the limited sample size, we have to keep the gap initialization cost at a reasonable level in order to capture those events. So we then experimented with a variation of the two-level gap opening cost approach. Instead of setting a fixed value for higher the \( M \rightarrow M \) transition probabilities, we established a
cap to prevent it from going up beyond a certain threshold. For instance, we set the cap to 0.92, and the final transition probability of $M \rightarrow M$ will never be allowed to exceed this value. By doing so, we tried to avoid creating too conservative models.

Enhancing the peaks of the transition probability profile

As we mentioned earlier, the transition profiles obtained from the forward-backward algorithm tend to be smoother than the corresponding SUPER90 transition profiles. For example, there is a region of medium to low $M \rightarrow M$ transition probabilities spread over multiple positions for hybrid, where in the corresponding SUPERFAMILY model there is just a single position with a much lower $M \rightarrow M$ transition probability inside the same region, or a much narrower spread of low transition probability positions. We thought this was an undesirable outcome of the hybrid alignment, since such profile may be too general and encourage the alignment of non-members of the family. So we tried to make those peaks more pronounced by first squaring the $M \rightarrow M$ probability and then renormalize it with the $M \rightarrow I$ and $M \rightarrow D$ probabilities. The purpose was to emphasize the distinctive features of the profile in order to provide more discerning power to the model.

Utilizing the information from emission probability matrix

Studies have shown that other sources of information, such as secondary structures, tertiary structures and sequence annotations can be used to improve the quality of sequence models. Since these information provides help in locating conserved regions and identifying domain boundaries. Unfortunately, we do not have such information available for hybrid alignment inside PSI-BLAST, and we also did not want to increase the complexity of the code to incorporate these information in the current
implementation of Hybrid PSI-BLAST. However, we were still able to get some hints about the conservedness of the positions in the model by examining the values in the emission probability matrix. If some positions are well conserved, the corresponding entries in the emission probability matrix will have high probability values for a few amino acids at those positions. So if we can measure the similarity between the distribution of the emission probabilities and that of the background probabilities, we may derive a quantitative judgment of the conservedness of those positions. Usually, the more similar these two residue distributions are, the less likely the corresponding position is conserved. Consequently, we should not set a high gap costs for such positions. Analysis of the models provided by SUPER90 v1.69 have demonstrated the validity of this observation as shown in Figure 5.5. Here the quantitative measure of the similarity between the emission probabilities and the background probabilities is given by the average relative entropy $\bar{H}$ over a window of certain size. The reason that we calculated the relative entropy over a window is because we are interested in regions that are well conserved where gaps are less likely to occur. The relative entropy $H_j$ at individual model position $j$ is calculated by the following equation:

$$H_j = \sum_a q_{j,a} \cdot \log\frac{q_{j,a}}{p_a}$$

where $q_{j,a}$ is the target frequency of amino acid $a$ at model position $j$ and $p_a$ is the background probability for amino acid $a$. The vertical axis is the transition probability for $M \rightarrow M$. The graph clearly shows that these two quantities have a positive correlation for positions with high relative entropy, which tend to have a high $M \rightarrow M$ probability as well. But for positions with low relative entropies, there is a wide distribution of possible $M \rightarrow M$ probabilities.
Figure 5.5: Relationship between average relative entropy $\bar{H}$ and $M \rightarrow M$ probability. As shown in the graph, these two quantities have a positive correlation for positions with high relative entropy, which tend to have high $M \rightarrow M$ probabilities as well. But for positions with low relative entropies, there is a wide distribution of possible $M \rightarrow M$ probabilities. (Plot produced with the help from Nicholas Chia.)
6.1 Compatibility studies

In our first implementation stage, we tested the compatibility of hybrid algorithm as an $E$-value computation engine in a well established iterative search framework, namely PSI-BLAST. This assessment was crucial to determine the extent to which the hybrid algorithm can leverage the investments that went into building the current crop of high performance bioinformatics tools. In principle, the hybrid alignment can offer crucial additional features to PSI-BLAST such as position-specific gap costs. However, the heuristics built into a high performance tool like PSI-BLAST have been extensively optimized for its native, Smith-Waterman based, alignment algorithm over the course of several years. Thus it was not clear what would be the effects of replacing the current algorithm for $E$-value calculation with the hybrid algorithm, even without adding extra features. For example, we found that the edge effect correction of $E$-values follows different laws between the two versions of PSI-BLAST, pointing to at least one area in which the two algorithms (and their underlying statistics) interact differently with the rest of the code. This topic has been discussed in more detail
in section 5.4 and the results are explained in section 6.2. Here, we present the performance comparison results from this study.

As described in section 5.2, the ASTRAL40 (version 1.59) dataset was use as a “gold standard” and we performed two different sensitivity assessments. First, we used each of the sequences in the gold standard database as queries. For each query we searched the gold standard database with the Hybrid and the NCBI version of PSI-BLAST. We ran both PSI-BLAST versions for several iterations until they converged. From the resulting lists of hits with their $E$-values we calculated two values for each $E$-value cutoff, namely the errors per query and the coverage, i.e., the number of true hits with an $E$-value smaller than the cutoff divided by the total number of true hits in the database (88,171 in this experiment). The plot of errors per query versus coverage as the $E$-value cutoff is varied is what is commonly known as the ROC curve and it represents the relationship between the sensitivity and selectivity of a program.

In this test we observed a large increase in computational run time when using the hybrid algorithm. The total computer time required for the assessment of the Hybrid PSI-BLAST was about ten times higher than for the original PSI-BLAST. However, this was an artifact of the unrealistically small database size in this test. The Hybrid PSI-BLAST requires some query-dependent parameters like the relative entropy $H$ to be calculated during the startup phase. For a small database this startup phase dominates the computational effort. For large databases, the computational effort for the startup phase is not so important any more and the run times for the Hybrid PSI-BLAST and NCBI PSI-BLAST become comparable.

Since the hybrid algorithm treats gaps differently from the Smith-Waterman algorithm underlying the NCBI PSI-BLAST, it was not immediately clear if the cost
Figure 6.1: Comparison of Hybrid PSI-BLAST performance for different gap costs. The curves show the trade-off between the errors per query and the coverage for Hybrid PSI-BLAST on a “gold standard” database using different gap costs. While all curves are relatively close together, a cost of $11 + 1 \cdot k$ for a gap of length $k$ seems to lead to the best performance.
of $11 + 1 \cdot k$ for a gap of length $k$ that has been determined to be optimal for the original PSI-BLAST was also good for the hybrid version. Thus, we first compared different values of the gap initiation and extension cost for the hybrid version of PSI-BLAST given as command line parameters. Modifying the gap costs affects the gap distribution in the model built in the first iteration, and therefore exposes potential differences in the gap bias of the two algorithms over the following iterations. The result of the measurements was the family of curves shown in Figure 6.1. Examination of these curves shows that the hybrid version of PSI-BLAST is relatively robust with respect to the gap costs. However, among all these substantially similar curves, the default value of $11/1$ for NCBI PSI-BLAST seems also optimal for the Hybrid version, suggesting no differences in gap bias.

The final result of this direct comparison between the Hybrid and NCBI version of PSI-BLAST using the gap cost $11/1$ is shown in Figure 6.2. The curves show that the sensitivity versus selectivity tradeoff of the two versions is quite comparable. Hybrid PSI-BLAST is slightly better than the NCBI PSI-BLAST up to a level of coverage of about 15%, and then incurs slightly more errors than the NCBI PSI-BLAST. The two curves are qualitatively similar, which suggests that their differences reflect the untuned performance of Hybrid PSI-BLAST.

In the second sensitivity assessment we aimed at comparing the two algorithms in a more realistic setting. Instead of searching the very small gold standard database alone, we augmented the gold standard database with the non-redundant protein database, nr, from NCBI. Searching this much larger database allows better sequence models to be built and is closer to a typical applications of a tool like PSI-BLAST. The sequences from the gold standard database were marked so that they could be
Figure 6.2: Comparison of the NCBI and the Hybrid version of PSI-BLAST. The curves show the trade-off between the errors per query and the coverage for NCBI and Hybrid PSI-BLAST on a “gold standard” database. For small coverages Hybrid PSI-BLAST is slightly superior while for high coverages the NCBI PSI-BLAST performs better.
identified in the program output. Sequences in the nonredundant database longer than 10 kilobases were trimmed to 10 kilobases because the protein sequence formatting program 'formatdb' associated with PSI-BLAST 2.0 could not handle such long sequences. The newly combined dataset was called ASTRAL40NRtrim. Since an exhaustive test using all sequences from the gold standard database as queries would be too time consuming, we randomly selected 100 queries from the gold standard database and searched against ASTRAL40NRtrim. By selecting very high $E$-value thresholds for output of sequences we ensured that enough of the sequences from the gold standard databases were included in the hit lists. In typical applications of PSI-BLAST, the number of iterations is restricted to a relatively small number since a failure to converge quickly is usually a sign of the model being polluted by foreign sequences, in which case more iterations actually worsen the quality of the model. In order to get an idea of the influence of the allowed maximal number of iterations we chose a limit of 5 and 6 for both algorithms and compared the results. For all other parameters their respective default values were used.

As expected, the run time of the two algorithms applied to this database of realistic length is very comparable, with the Hybrid PSI-BLAST taking roughly 25% longer than the original PSI-BLAST. This confirms our hypothesis that the apparently large difference in run time between the Hybrid and the original PSI-BLAST seen in the test on the smaller database was due to the startup phase of the Hybrid PSI-BLAST. The sensitivity was assessed by the same type of ROC curves representing the tradeoff between errors per query and coverage as before. In calculating the errors per query and the coverage all hits from the non redundant database were ignored since their homologies are not known. Only hits from the gold standard database were evaluated.
Figure 6.3: Comparison of the NCBI and the Hybrid version of PSI-BLAST on a large database ASTRAL40NRtrim. The curves show the trade-off between the errors per query and the coverage for NCBI and Hybrid PSI-BLAST for those sequence pairs the homology of which is known from structural considerations. For small coverages Hybrid PSI-BLAST is slightly inferior while for high coverages the two algorithms perform nearly identically.
The results for the two algorithms for the different limits on the number of iterations are shown in Figure 6.3. We found that the Hybrid PSI-BLAST seems to have a stronger dependency on the limit on the number of iterations than the original PSI-BLAST. In general the Hybrid PSI-BLAST is inferior at small coverage. Note, however, that the region of coverage and errors per query in which the Hybrid PSI-BLAST was found to be superior on the smaller database cannot be probed in this test due to the smaller number of queries which limit the errors per query to a minimum of 0.01. At higher coverage the sensitivity of the two algorithms becomes nearly indistinguishable if the number of iterations is limited to five.

In conclusion of Phase I, we established that the hybrid alignment algorithm can be successfully used within PSI-BLAST with only modest changes to the original code. Through direct comparison, and in optimizing one parameter for the hybrid algorithm within the whole framework of PSI-BLAST, namely the gap costs, we found that the Hybrid version of PSI-BLAST and the original version of PSI-BLAST are comparable in their performance.

6.2 Finite-size correction

We addressed the problem of finite-size correction empirically by aligning sequences from a database derived from SCOP [39, 15] which is part of the ASTRAL compendium [30, 11] (http://astral.stanford.edu/, release ASTRAL SCOP 1.59). This database contains only sequences with less than 40% pairwise sequence identity. We use every sequence from the database as a query for a hybrid alignment search of the whole database. This yields a list of hits for each query and their respective E-values calculated by formula Eq. (5.4) or (5.5). Following the approach of
BLAST and PSI-BLAST instead of evaluating Eqs. (5.4) and (5.5) for each hit, we use

\[ E(\Sigma) = K A_{\text{eff}} e^{-\lambda \Sigma} \]  

(6.1)

where \( A_{\text{eff}} \) is the effective search space. It is determined once for each query as

\[ A_{\text{eff}} = \frac{e^{\lambda \Sigma^*}}{K} \]  

(6.2)

with \( \Sigma^* \) given by \( E(\Sigma^*) = 1 \) according to Eq. (5.4) or (5.5). In this framework the difference between Eqs. (5.4) and (5.5) translates into a different value of \( \Sigma^* \), i.e., in a different value of the effective search space \( A_{\text{eff}} \).

Since the database is derived from the structural SCOP database, hits can be identified as true homologs if they belong to the same superfamily in SCOP and as non-homologs if not. Thus, for each \( E \)-value cutoff the number of errors per query can be calculated as the number of non-homologs with an \( E \)-value lower than the given cutoff divided by the total number of queries in the dataset, which is 4,383 in this case. If the calculation of \( E \)-values is correct, the number of errors per query is identical to the \( E \)-value cutoff.

Figure 6.4 shows the relationship between the errors per query and the \( E \)-value cutoff for the PSI-BLAST default scoring system and for the BLOSUM62 scoring matrix with a cost of \( 9 + 2k \) for a gap of length \( k \). The latter has a relative entropy of \( H \approx 0.15 \) and thus the contribution of the higher order terms should be less dramatic than for the PSI-BLAST default scoring system. Each of the two graphs shows the plots of the \( E \)-value versus the errors per query for hybrid alignments with both length correction formulas and of the original BLAST 2.0. The (ideal) identity is shown as the dashed line. In both cases it can be seen that the \( E \)-values obtained
Figure 6.4: Comparison of two formulas for edge effect correction. Both graphs show the dependence between the $E$-value cutoff and the number of errors per query, i.e., the number of non-homologous sequence pairs with an $E$-value lower than the cutoff divided by the total number of sequences in the database. The dotted line corresponds to hybrid alignment with $E$-values calculated according to Eq. (5.4) while the solid line corresponds to hybrid alignment with $E$-values calculated according to Eq. (5.5). The dash-dotted line is the result of BLAST 2.0 and the dashed line is the identity corresponding to an ideal algorithm. Both graphs show data for the ASTRAL40 database and the BLOSUM62 scoring matrix. In (a) the cost of a gap of length $k$ is $11 + k$ while in (b) it is $9 + 2k$. In both cases BLAST 2.0 and Eq. (5.5) yield good estimates of the $E$-value while Eq. (5.4) is clearly inferior for hybrid alignment.
by Eq. (5.5) are very close to ideal. The $E$-values of BLAST 2.0 are similarly good while the $E$-values for the hybrid algorithm calculated according to Eq. (5.4) are too small. As expected based on the differences in the relative entropies the effect is much stronger for the BLOSUM62/11/1 scoring system than for the BLOSUM62/9/2 scoring system. We conclude that for the hybrid alignment algorithm Eq. (5.5) provides better estimates of the $E$-value than Eq. (5.4).

6.3 Parallelization of Hybrid PSI-BLAST

Running our experiment on the ASTRAL40NRtrim dataset for the first stage implementation of Hybrid PSI-BLAST took about 80 CPU days. To get a more manageable execution time we decided to parallelize the code using a simple MPI wrapper. The wrapper allowed us to run experiments on an 8-node cluster of dual 1 GHz Pentium III, 1 GB RAM machines interconnected with a Myrinet network. The resulting MPI program is organized as a master/workers computation, with one worker running on each node. Each worker spawns an instance of the binary on a different input query sequence, handed down by the master. The use of MPI reduces various sources of overhead, such as the distribution of the input data, and the communication between master and workers.

Figure 6.5 shows the speedup measured on a random sample of 100 sequences chosen from ASTRAL40. The speedup is defined as $S = T_s/T_p$, where the serial execution time $T_s = 58,080s$ is measured using the wrapper on a single node. The number of nodes actually used to measure each value $T_p$ is one more than shown, in other words the graph refers to the number of workers. As shown in Figure 6.6, the efficiency is larger than 80% up to 8 nodes (93% if a rogue sequence is excluded, see
Figure 6.5: Speedup of the first stage implementation of Hybrid PSI-BLAST by parallelization
Figure 6.6: Efficiency graphs of the parallelized Hybrid PSI-BLAST after implementation stage I on a sample of 100 sequences chosen at random from ASTRAL40 dataset.
below), then decreases to 52% for 16 nodes - partly due to every pair of processes sharing a node.

One of the sequences (d1qbkb) took significantly longer than the average (approximately one order of magnitude), as can be seen in Figure 6.7. The execution time of this single sequence becomes comparable to the total parallel execution time when the number of processors is sufficiently large, therefore affecting the value of \( T_p \). This effect can be seen in Figure 6.5, where the dashed lines represent the execution time for the very same sample of input sequences except d1qbkb. Obviously the simple parallelization scheme we used work reasonably well from an efficiency point of view, but it can fail in case of extreme load unbalances.

### 6.4 Database search heuristics

In section 5.3 we have described the problem that may arise during the gapped extension step of the database search heuristics, which results from the fact that such extension heuristics will ignore part of the alignment lattice and may produce an underestimated alignment score for possible true positives. Due to the underestimated score, these potential positives might be filtered out and result in less sensitive models, hence a reduced coverage of the search. We proposed a “soft-band” approach to solve the problem, and we present the evaluation of this technique here.

We examined several metrics to provide a comprehensive measure on the viability of the “soft-band” approach. These metrics include a measure of the selectivity defined by the miss rate and a measure of the effectiveness defined by the average execution time per subject that passes the heuristics filtering.
Figure 6.7: Execution times for the 100 individual query sequences.
In testing the new heuristics, we used a subset of a database called SUPERFAMILY of version 1.69. The SUPERFAMILY database was developed by Gough, et al. [20], and is also derived from SCOP. The subset we used is filtered to contain sequences with no more than 90% sequence identity, which is referred here as SUPER90. There are 10501 sequences in SUPER90 database and every sequence is used as a query to search against the whole database. For each query, we kept the track of the following information:

- The number of gapped extension attempts, $N_{GEA}$
- Whether the subject sequence is selected the heuristics
- The actual alignment score, $S$, if the subject sequence ever made it to the final scoring phase
- Execution time for the gapped extension attempt for this pair, $T_{GEA}$

Then we pooled these statistics together and for each score, calculated the following quantities:

- The number of query-subject pairs that have such an alignment score, $N$
- Total number gapped extension attempts, $N_{TGEA}$
- The number of pairs successfully passed the heuristics, $N_{suc}$
- The number of pairs that should pass the heuristics but missed, $N_{miss}$
- Total execution time, $T_{tot}$
The formula below gives the miss rate, $MR$, which characterizes the loss of coverage. Normally we look for lower miss rates, if we expect better coverage.

$$MR = \frac{N_{\text{miss}}}{N}$$

(6.3)

The effectiveness is defined as the average execution time per successfully passed subject, $AET$, which can be computed by the following formula:

$$AET = \frac{T_{\text{tot}}}{N_{\text{suc}}}$$

(6.4)

In fact, a closely related measure could also be used to judge how effective the heuristic approach is, which is the average number of attempts per successfully passed subject, $AA$.

$$AA = \frac{N_{\text{TGEA}}}{N_{\text{suc}}}$$

(6.5)

By plotting the $MR$ vs. score ($S$), and $AET$ vs. score ($S$), we will get the indication of the usefulness of the “soft-band” approach. However, we consider $AET$ to be a more direct measure of the effectiveness of the approach and it works better than $AA$, since the execution time for each gapped extension attempt is not a constant.

In Figure 6.8, the selectivity of the new heuristics (Hybrid heuristics) and the old heuristics (BLAST heuristics) is shown as their miss rate at different alignment scores. The “soft-band” heuristics (Hybrid heuristics) has a much lower miss rate than the BLAST heuristics, when the alignment score is moderate. For very good alignment scores, their miss rates all approach to zero. Figure 6.9 shows that the new heuristics approach takes less time on average than the old heuristics method to identify a potential hit. These two plots clearly demonstrate the robustness of the new heuristics approach.
Miss rate of gapped extension heuristics

Figure 6.8: Miss rate vs. Alignment score plot. It is clear that the new heuristics, which uses an adaptive banded search, works better than the old heuristics approach, since the latter may potentially miss a good sequence candidate for hybrid alignment due to the limited region covered on the alignment lattice.
Figure 6.9: The plot of the effectiveness of both heuristics approaches. The effectiveness is defined as the average execution time per passed sequence (AET). It shows that the new heuristics spends less time on average than the old heuristics to identify a potential hit.
6.5 Sequence weighting

For the performance assessment of the early implementations of the hybrid alignment, such as for the first two stages, we employed the ASTRAL40 dataset, which only contains sequences with less than 40% sequence identities. So we did not need to worry much about the sequence weighting, as the sequences are not closely related. However, such dataset is sometimes not able to provide a good representative set of sequences of the protein families, which often leads to low quality models.

Considering the drawbacks of benchmarking on a small and well differentiated dataset, we switched to another dataset, SUPER90, for our later stage implementation performance assessment, because we needed more sequences to estimate the position specific transition probability matrix. The dataset bias becomes evident as the sequences in SUPER90 can share up to 90% sequence identities. Thus we tested different sequence weighting schemes, which are listed as follows:

- **Equal Weighting**: Each sequences in the training set receives the same weight when used to derived the new model, which is essentially the same as no sequence weighting at all.

- **NCBI Weighting**: A variation of the position specific weighting scheme. Sequence weights are derived from the NCBI PSI-BLAST weighting routines, which use the multiple sequence alignment constructed from pair-wise optimal sequence alignments.

- **Generalized position specific weighting (GPSW)**: A generalized position specific weighting proposed for the probabilistic version of the multiple sequence
Figure 6.10: Comparison for different sequence weighting schemes. As illustrated in the plot, the generalized position-specific weighting (GPSW) outperforms the equal weighting method, because the latter essentially does not take the database bias into account. Interestingly, the weighting scheme used by NCBI PSI-BLAST works better than the GPSW approach in the low coverage region, but worse than GPSW in the high coverage region.
alignment built from the forward-backward algorithm in Hybrid PSI-BLAST.

The details of this weighting scheme is covered in section 5.5.

The results are shown in Figure 6.10. It demonstrated that the generalized position-specific weighting (GPSW) outperforms the equal weighting method, because the latter essentially does not take the database bias into account. Interestingly, the weighting scheme used by NCBI PSI-BLAST works better than GPSW approach in the low coverage region, but worse than GPSW in the high coverage region.

NCBI PSI-BLAST utilizes a block-wise weighting approach described in section 5.5, which only considers sequences whose optimal pair-wise alignments contribute residues to a position in the multiple alignment. We also tried to implement a block-wise GPSW and tested whether it may work for Hybrid PSI-BLAST. However, the hybrid alignment does not have a definite beginning but a probability for having an alignment at each position. To defined the boundary for blocks, we used a threshold value $Y$ to decide whether one position should be considered in the alignment or not. Thus if the probability of having an alignment at a position exceeded $Y$, this position was included in the multiple alignment, otherwise it was excluded. Figure 6.11 illustrated the effect of such block-wise GPSW method, which does not change the ROC curve significantly; the block-wise GPSW with $Y = 0.5$ seems to work best overall.

6.6 Model regularization

Following the discussion in Chapter 5, we used the forward-backward algorithm described in Chapter 3 to estimate the new emission probability and transition probability matrices, which are different from the matrices that are built from optimal
Figure 6.11: Performance of the block-wise GPSW shows that such weighting does little change to the ROC curves. However, it improves the performance in the high coverage region, while slightly loses performance in the low coverage region. The block-wise GPSW with $Y = 0.5$ seems work best overall.
pair-wise alignments. These new matrices have incorporated the information from suboptimal alignments. In section 6.6.1, we describe how we tested the effectiveness of the new emission probability matrix by comparing our implementation of Hybrid PSI-BLAST from stage II with the NCBI PSI-BLAST and the Hybrid PSI-BLAST from implementation stage I. To distinguish these two versions of Hybrid PSI-BLAST, we use the notation \textbf{preHybrid PSI-BLAST} for the first stage implementation, and refer to the version from the second stage as \textbf{emHybrid PSI-BLAST}, since it extends the functionality of Hybrid PSI-BLAST by integrating suboptimal alignments into the creation of the position specific emission probability matrix. We also optimized the choice of the pseudocount constant for building the final emission probability matrix from the raw values calculated by the forward-backward algorithm.

After implementing the generalized position-based sequence weighting scheme and the routines for calculating position-specific transition probability matrix in stage III, we evaluated the new version (now named as \textbf{Hybrid PSI-BLAST}), by comparing its performance with NCBI PSI-BLAST, which is detailed in section 6.6.2.

6.6.1 Emission probabilities estimation

In assessing the performance of emHybrid PSI-BLAST, we followed the method described in section 5.2, which relies on the ASTRAL40 database derived from the SCOP classification. Because the sequences in this database share less than 40% sequence identity and are presumed to contain only weak homologies, it constitute a challenging test. The original database has 4383 sequences. However, one sequence (the representative of superfamily c.11.1, which contains only itself in that superfamily) was excluded due to the wrong classification assignment by SCOP, which is
evident as later releases of SCOP have changed its classification to even a different fold, c.10.3. The remaining 4382 sequences have a total of 88171 pairs of true homologs, indicated by their SCOP classification. All-vs-all comparisons were carried out in our performance evaluation. Coverage and error per query (EPQ) as defined in Chapter 5 were calculated as a parametric function of the cutoff $E$-value after pooling the reported hits together and ranking them by $E$-value. The tradeoff between sensitivity and selectivity of the programs is demonstrated on ROC curves drawn as EPQ vs. Coverage plot.

One consideration we had about the emHybrid PSI-BLAST is the choice of the pseudocount constant $\beta$ in Eq. (5.7). Since NCBI PSI-BLAST has undergone years of optimization, it is not clear whether the pseudocount constant used by NCBI is also optimal for hybrid alignment. Specifically we expected that we might have to add less pseudocount because the contribution of the suboptimal alignments already provides counts for amino acids that are absent in the optimal alignments. Thus, we tried different values of the pseudocount constant $\beta$ (since only the ratio $\beta/\alpha$ enters in Eq. (5.7), it is enough to vary one of these constants). We found that the default value for NCBI PSI-BLAST is also optimal for emHybrid PSI-BLAST, as shown in Figure 6.12.

Having determined the pseudocount for calculating the position-specific alignment emission probability matrix, we went ahead to evaluate our new approach by comparing the emHybrid PSI-BLAST with preHybrid PSI-BLAST and NCBI PSI-BLAST. However, the accuracy of SCOP classification and conventional application of the homology relationship implied by the SCOP classification have recently been questioned by researchers, as some of the proteins that are grouped in different superfamilies in
Figure 6.12: emHybrid PSI-BLAST performance with different choices of pseudocount constant $\beta$. The larger the constant, the more the emphasis on the prior belief of the amino acid substitution probabilities. The default value 10 used in NCBI PSI-BLAST works best for emHybrid PSI-BLAST as well.
SCOP might actually be homologs. This question becomes a real concern when comparing sequence alignment algorithms. Recall that only sequence hits from the same superfamily are counted as true homologs under the conventional evaluation process. Suppose a more sensitive sequence alignment algorithm were able to identify very weak homologies, those sequences would be counted as errors under the old evaluation scheme. To circumvent this difficulty and draw a fair conclusion, we employed another way of counting errors, which does not count as errors (but neither as true positives) the sequence hits that come from different superfamilies but are still in the same fold. Since this counting method acts on the conservative side by considering there is no sufficient information to make a judgment on those sequences, we denote such counting scheme as conservative counting. On the other hand, we refer to the conventional counting method as the straight counting, because it is a straight application of the SCOP classification.

To assess the changes in sensitivity resulting from the incorporation of the information of the suboptimal alignments, we applied both counting approaches to evaluate the performance of preHybrid PSI-BLAST, emHybrid PSI-BLAST and NCBI PSI-BLAST. The result in Figure 6.14 shows that emHybrid PSI-BLAST is comparably or slightly more sensitive than NCBI PSI-BLAST in detecting homologous sequences in the database when straight counting is applied. When coverage is low, emHybrid PSI-BLAST makes more errors than preHybrid PSI-BLAST but still fewer than NCBI PSI-BLAST. However, emHybrid PSI-BLAST improves greatly over preHybrid PSI-BLAST when coverage is high. The picture becomes quite different when conservative
Figure 6.13: Performance comparison among NCBI PSI-BLAST, preHybrid PSI-BLAST, and emHybrid PSI-BLAST. The straight counting method is used in calculating EPQ. preHybrid PSI-BLAST works best in the low coverage region, but worse in the high coverage region. emHybrid PSI-BLAST is comparable to or slightly better than NCBI PSI-BLAST.
Figure 6.14: Performance comparison among NCBI PSI-BLAST, preHybrid PSI-BLAST, and emHybrid PSI-BLAST. The conservative counting scheme is used instead. emHybrid PSI-BLAST outperforms both NCBI PSI-BLAST and preHybrid PSI-BLAST over the entire range of coverage.
counting is applied. Figure 6.13 clearly shows that emHybrid PSI-BLAST outper-
forms both NCBI PSI-BLAST and preHybrid PSI-BLAST over the whole range of
coverage.

Figure 6.15 and Figure 6.16 highlight the difference between straight and conserva-
tive counting for NCBI PSI-BLAST and emHybrid PSI-BLAST, respectively. It can
be easily seen that the counting method makes hardly any difference for NCBI PSI-
BLAST. This implies that all false hits of NCBI PSI-BLAST are real non-homologs.
On the contrary, for emHybrid PSI-BLAST there is a noticeable disagreement between
the results for straight counting and for conservative counting. Thus, we suspect that
the performance of emHybrid PSI-BLAST is already beyond what the SCOP classi-
ification can achieve, i.e., emHybrid PSI-BLAST can find some homologs that are so
remotely related that they are classified into different superfamilies in SCOP.

To further augment this point, we collected hits with $E$-value less than 0.01 for
emHybrid PSI-BLAST that are false hits under the straight counting scheme but un-
decided according to conservative counting. We examined their relationships by re-
ferring to another popular protein classification database CATH [43], which combines
manual and automated processes to organize the proteins. There are 39 ambiguous
sequence pairs found by emHybrid PSI-BLAST that are classified as non-homologous
pairs in SCOP. In CATH, almost all of these proteins pairs (35 out of 39) are classi-
fied as homologs. The exceptions are 3 sequence pairs that are not classified at all in
CATH and one sequence pair (d1jtdb_, vs. d1k3ia3, the one with the largest $E$-value
of 0.009) that is classified as non-homologous. These 39 sequence pairs involve 31
sequences, listed in Table 6.1.

123
Evaluation discrepancies for emHybrid PSI-BLAST

Figure 6.15: Difference between two error counting methods on emHybrid PSI-BLAST. The conservative counting scheme reports less errors than the straight counting as expected. Those disputable sequences that have an $E$-value less than 0.001 are collected and examined more closely. When two other protein structure classification systems, namely CATH and DALI, have been used to test their relationship, almost all of the “errors” found by the straight counting are classified as “homologs” according to CATH and DALI.
Figure 6.16: Difference between two error counting methods on NCBI PSI-BLAST. There is hardly any change, which implies that almost all the false hits identified by the straight counting belong to even different folds. Such errors are true non-homologs.
To further investigate the possible homologies uncovered by emHybrid PSI-BLAST, we also submitted the coordinate files of the 35 pairs that are considered homologs in CATH to DaliLite, which is the structure comparison and database search engine for another protein classification system called DALI [17]. The result shows that except for two very short sequences (\(\sim 50\) amino acids in length) the Z-score for each pair is above 10. As for the root-mean-square deviation (RMSD) of \(\alpha\)-carbon atoms, 22 pairs share structures within 3Å RMSD, 11 pairs within 4Å RMSD and 2 pairs within 6Å RMSD. These findings provide a strong argument for these sequence pairs to be classified as homologs.

6.6.2 Transition probabilities estimation

Proteins have secondary structures such as \(\alpha\) helices or \(\beta\) sheets that are more conserved than other regions such as loops or coil-coil. So it is less likely for proteins to have insertions or deletions within their structural elements than between them. Position-specific gap penalties are therefore more appropriate than the simple position-independent gap costs scheme. In this section, we describe the results we have obtained from our implementation of position-specific transition probability matrix enhancements strategies.

As detailed in section 5.6.2, we have tried many ways to fine-tuned the transition probability matrix that we have derived from the forward-backward algorithm.

Enhancing the peaks of the transition probability profile

We tested on the ASTRAL40 dataset (version 1.59) the strategy described in section 5.6.2, which tries to make more pronounced peaks in the transition probability profiles created from the forward-backward alignment. Figure 6.17 shows the
Figure 6.17: Performance comparison among emHybrid PSI-BLAST, Hybrid PSI-BLAST with the initial incorporation of the position-specific transition probability matrix, and Hybrid PSI-BLAST with subsequent adjustment on the matrix. emHybrid PSI-BLAST works the best among them, but the adjusted transition profiles improve over those which only use raw values.
<table>
<thead>
<tr>
<th>SCOP NAME</th>
<th>SCOP ID</th>
<th>CATH ID</th>
<th>SCOP NAME</th>
<th>SCOP ID</th>
<th>CATH ID</th>
</tr>
</thead>
</table>
| d1hg3a.   | c.1.1.1  | 3.20.20.90 | d1a4ya.   | c.10.1.1 | 3.80.10.10 |}
| d1thfd.   | c.1.2.1  | 3.20.20.90 | d1yrsga.  | c.10.1.2 | 3.80.10.10 |}
| d1rpxa.   | c.1.2.2  | 3.20.20.90 | d1fqa2.   | c.10.1.3 | 3.80.10.10 |}
| d1dbta.   | c.1.2.3  | 3.20.20.90 | d1h6ta2.  | c.10.2.1 | 3.80.10.10 |}
| d2tpsa.   | c.1.3.1  | 3.20.20.90 | d1h6ua2.  | c.10.2.1 | 3.80.10.10 |}
| d2dora.   | c.1.4.1  | 3.20.20.90 | *d1j15a.  | c.10.2.6 | 3.80.10.10 |}
| d1ep3a.   | c.1.4.1  | 3.20.20.90 | d1deca3.  | c.10.2.2 | 3.80.10.10 |}
| d1d3ga.   | c.1.4.1  | 3.20.20.90 | d1thfd.   | g.41.3.1 | 2.20.25.10 |}
| d1gex.    | c.1.4.1  | 3.20.20.90 | d1qys.    | g.41.9.1 | 2.20.25.10 |}
| d1ltida1. | c.1.4.1  | 3.20.20.90 | d150i2.   | g.41.9.1 | 2.20.25.10 |}
| d1h7wa2.  | c.1.4.1  | 3.20.20.90 | d1en2a2.  | g.3.1.1  | N/A       |}
| d1e0a2.   | c.1.4.1  | N/A       | d1fja.    | g.3.7.3  | N/A       |}
| d1zfja1.  | c.1.5.1  | 3.20.20.90 | d1hf2a1.  | b.80.3.1 | N/A       |}
| d1ak5.1   | c.1.5.1  | 3.20.20.90 | d1ea0a1.  | b.80.4.1 | N/A       |}
| d1jr1a1.  | c.1.5.1  | 3.20.20.90 | d1j1db.   | b.69.5.2 | 2.130.10.30 |}
|           |          |           | d1k3ia3   | b.69.1.1 | 2.130.10.80 |}

Table 6.1: Ambiguous sequences that have been identified by the conservative counting method during the evaluation of the performance of emHybrid PSI-BLAST. These sequences have an $E$-value less than 0.001. The first column is the names of those sequences, the second column is the SCOP classification of each sequence. Four codes, which are separated by dot, represent *Class, Fold, Superfamily*, and *Family*, respectively. Conventionally, sequences from the same superfamily are considered homologs. Otherwise, they are non-homologs. The last column is the CATH classification, with each number representing *Class, Architecture, Topology*, and *Homologous superfamily*, respectively. In CATH, sequences are considered homologs if they are classified in the same Homologous superfamily. Discrepancies are evident for the classification on these sequences by SCOP and CATH.

comparison between the performance of Hybrid PSI-BLAST before and after this adjustment. It also demonstrated the difference between emHybrid PSI-BLAST and Hybrid PSI-BLAST with initial implementation of the position-specific gap costs.

We may notice that the position-specific gap costs, which is directly derived from the transition probability profiles calculated by the forward-backward alignment, actually deteriorate the sensitivity of Hybrid PSI-BLAST, even after the adjustment on the
raw values. However, this adjustment on the peaks nonetheless improves over the raw transition profiles, as evident in the Figure 6.17.

**Capping the gap opening cost**

Even after the adjustment of the raw values of the transition probability profile, the performance of Hybrid PSI-BLAST is still worse than emHybrid PSI-BLAST which only use non-position specific gap penalties. Careful examination of the values of the position-specific transition probability matrices showed that at many positions the \( M \rightarrow M \) transition probabilities computed by the forward-backward alignment become very high and close to the maximum value 1. A high \( M \rightarrow M \) probability that approaches 1 essentially turns off gaps, as the corresponding gap opening cost is approaching infinity. However, we should allow relatively more gaps in our model in order to find distantly related sequences. We consider that the unreasonable high gap costs that are derived from the forward-backward alignment are due to the number of transitions implied in the training sequences being insufficient. So we tried to put a cap on the possible values that \( M \rightarrow M \) transition probability can take.

Figure 6.18 demonstrated the effectiveness of this capping strategy. Before capping on the gap opening cost, Hybrid PSI-BLAST performs worse than emHybrid PSI-BLAST. However, after capping, the sensitivity of Hybrid PSI-BLAST increases and it becomes better than emHybrid PSI-BLAST in the high coverage region.

**Two-level gap costs**

The ASTRAL40 dataset (version 1.59) was used as previously described to test different choices for the two-level gap costs scheme. In this experiment, we first calculated the raw transition probability matrix from the forward-backward algorithm,
Figure 6.18: Performance comparison among emHybrid PSI-BLAST, Hybrid PSI-BLAST with the adjustment on the raw position-specific transition probability matrix, and Hybrid PSI-BLAST with additional capping on the adjusted gap opening costs. emHybrid PSI-BLAST still works the best when coverage is low. However, the limitation that prevents the gap opening costs from going up too high improves the performance of Hybrid PSI-BLAST and makes it even better than emHybrid PSI-BLAST at high coverages.
then set the $M \rightarrow M$ transition probability to a high level (high gap opening cost) if the raw value of the $M \rightarrow M$ probability was larger than a predefined threshold. Otherwise, it was set to a lower level (low gap opening cost). In Table 6.2, the first column shows the gap costs (gap opening/gap extension) choices for the lower bound and the third column for the upper bound. The second column are the threshold values that were used to set the actual $M \rightarrow M$ transition probabilities. The ROC values calculated at different $EPQ$ are also included in the table.

We can conclude that the choice of the upper bound of the gap opening cost is more sensitive than the choice of the lower bound. For example, when we fixed the upper bound for gap opening cost at 11 and 13, respectively, the variation among ROC values due to the changes in the lower bound of the gap opening cost and the

<table>
<thead>
<tr>
<th>Low level</th>
<th>Threshold</th>
<th>High level</th>
<th>ROC</th>
</tr>
</thead>
<tbody>
<tr>
<td>11/1</td>
<td>N/A</td>
<td>11/1</td>
<td>EPQ = 1</td>
</tr>
<tr>
<td>4/1</td>
<td>0.8</td>
<td>11/1</td>
<td>2.081E-01</td>
</tr>
<tr>
<td>8/1</td>
<td>0.8</td>
<td>11/1</td>
<td>2.067E-01</td>
</tr>
<tr>
<td>4/1</td>
<td>0.7</td>
<td>11/1</td>
<td>2.060E-01</td>
</tr>
<tr>
<td>8/1</td>
<td>0.7</td>
<td>11/1</td>
<td>2.049E-01</td>
</tr>
<tr>
<td>4/1</td>
<td>0.5</td>
<td>11/1</td>
<td>2.067E-01</td>
</tr>
<tr>
<td>4/1</td>
<td>0.7</td>
<td>15/1</td>
<td>2.170E-01</td>
</tr>
<tr>
<td>4/1</td>
<td>0.7</td>
<td>13/1</td>
<td>2.205E-01</td>
</tr>
<tr>
<td>8/1</td>
<td>0.7</td>
<td>13/1</td>
<td>2.218E-01</td>
</tr>
<tr>
<td>4/1</td>
<td>0.8</td>
<td>13/1</td>
<td>2.221E-01</td>
</tr>
<tr>
<td>4/1</td>
<td>0.6</td>
<td>13/1</td>
<td>2.215E-01</td>
</tr>
</tbody>
</table>

Table 6.2: The ROC values for different choices of two-level gap opening gap cost scheme. It shows that the choice of the upper bound of the gap opening cost is more important, because the variation of ROC is about 1% due to changes in the lower bound of the gap opening cost and threshold, while it increases to 10% at $EPQ = 1$ if we change the upper bound instead. The upper bound of the gap open cost at 13 works the best among all the choices.
threshold is about 1%. However, the variation increases to 10% if we change the upper bound of the gap opening cost. The choice of 13 as the upper bound for gap opening cost works best overall among the choices tested here.

Utilizing the information from emission probability matrix

As we were considering if there is any other information that we can take into account for calculating the position-specific transition probability profiles, we noticed that gaps are less likely to occur in well conserved motifs, such as regular secondary elements or functional sites. However, such information is not available to the model building routines in Hybrid PSI-BLAST, where only the sequences themselves are present. Interestingly, the emission probability matrix can be used to estimate how conserved a position is by comparing the emission probability distribution of amino acids at this position with the background amino acid distribution. Since a well conserved position only contains a few types of amino acids, the more similar these two distributions are, the less conserved the position will be. We calculated the average relative entropy $\bar{H}$ as described in section 5.6.2 over a window of size 5 for each position, then use $\bar{H}$ as an approximation of the conservedness of the position to adjust the $M \rightarrow M$ transition probability for the same position.

Since the SUPER90 dataset contains an HMM with an emission probability matrix and a transition probability matrix for each sequence, we were able to collect the average relative entropy $\bar{H}$ and the $M \rightarrow M$ transition probability for every position across all the models. Figure 5.5 in section 5.6.2 displays the relationship between these two quantities.
Given this observation, we adjusted the $M \rightarrow M$ probability according to the following formula:

$$\Pr^{\text{adj}}(M \rightarrow M) = \begin{cases} 
0.5 \cdot [1 + \Pr(M \rightarrow M)] & \text{if } \tilde{H} > h \\
\Pr(M \rightarrow M) & \text{if } \tilde{H} \leq h 
\end{cases}$$

which moves up the $M \rightarrow M$ probability half of the distance to its maximum value of 1 when the average relative entropy $\tilde{H}$ is bigger than some threshold $h$, because we expect fewer gaps at well conserved regions.

We tested this idea on the SUPER90 dataset and Figure 6.19 shows the performance of Hybrid PSI-BLAST under different $h$ choices. It works best when $h$ is set to 0.9.

Following the same reasoning for putting an upper bound on the largest $M \rightarrow M$ probability, we also experimented with capping together with the adjustment on gap opening costs based on $\tilde{H}$. So in addition to the increase of the $M \rightarrow M$ probability at positions where its $\tilde{H}$ is larger than the threshold $h$, we also limited the largest possible value for $M \rightarrow M$ probability at positions with relative entropy smaller than $h$. The effect of this additional measure can be seen from Figure 6.20. The results indicate that capping the $M \rightarrow M$ for positions with low relative entropy generally helps at the high coverage region. But if the cap is set too low (0.9 in this case), the performance deteriorates in the low coverage region. The cap is most effective at 0.93 (corresponding to a gap opening cost of 13), which is coincident with the data from the capping experiment where the optimal performance is also achieved at a cap of 13 for gap opening penalty.

Figure 6.21 summarizes the results of our project by comparing the performance of different versions of PSI-BLAST. It demonstrated that our implementation of hybrid alignment in the second stage, emHybrid PSI-BLAST works slightly better than
Performance comparison

Different thresholds for $h$

Figure 6.19: Performance of Hybrid PSI-BLAST with different values of threshold $h$. The choice of 0.9 works best among all the tested values.
Figure 6.20: Performance of Hybrid PSI-BLAST with different values for the cap on $M \rightarrow M$ probability. The threshold $h$ is set to 0.9. The performance is the best when $M \rightarrow M$ is capped at 0.93, which roughly corresponding to a gap opening cost of 13. This result is in consistent with the data from the capping experiment, which also achieves the optimal performance with a cap of 13.
the original PSI-BLAST, NCBI PSI-BLAST. Our subsequence implementation of
the position-specific gap costs (Hybrid PSI-BLAST) improves the performance of
emHybrid PSI-BLAST significantly except for a small portion of the high coverage
region, where the EPQ is close to 1. The sensitivity of Hybrid PSI-BLAST is further
enhanced by coupling with the adjustment of the gap opening penalty based on the
conservedness of the positions which is defined as the average relative entropy within
a fixed window centered at those points.
Figure 6.21: Performance of different versions of PSI-BLAST: emHybrid PSI-BLAST is the version produced in stage II, which incorporates the information in the suboptimal alignments into the building of new models; Hybrid PSI-BLAST is the stage III version, which is built upon emHybrid PSI-BLAST and is enhanced with regularized position specific gap penalties.
CHAPTER 7

CONCLUSIONS AND FUTURE DIRECTIONS

In this dissertation, we describe our investigation on how to enhance the sensitivity of hybrid alignment for homology detection in database searches. We used PSI-BLAST as the framework for our implementation of hybrid alignment. As described in Chapter 4, we proceeded in three stages, each with its own specific aims. The development process begins with simple modifications of the PSI-BLAST package, and gradually adds more features of hybrid alignment.

Chapter 5 describes many important research problems we have faced during the whole process. For example, we addressed the issue of choosing appropriate dataset for evaluating the performance of our implementations in section 5.2. We proposed a new gapped extension heuristics method for hybrid alignment in section 5.3. Two formulas have been examined in section 5.4 for finite-size correction while assessing the statistical significance of short sequences. Section 5.5 describes a generalized position-specific weighting scheme for building more accurate sequence models. A large effort was devoted to the construction of the position-specific emission probability matrix and transition probability matrix. Section 5.6 depicts the methods we used for building the sequence model and some of the post-processing techniques we tried to improve the quality of the transition probability matrix.
Chapter 6 shows the results of the measurements we used to quantify the effect of the techniques described in Chapter 5. The first result reported in section 6.1 is the compatibility of the hybrid alignment and PSI-BLAST package. We showed that the hybrid alignment algorithm can be readily used in PSI-BLAST for scoring the alignment and assessing the significance of sequence hits, without loss of sensitivity and efficiency. Section 6.2 evaluated the two finite-size correction formula proposed by Altschul and Hwa and found that Hwa’s formula works better for hybrid alignment. We also studied the parallelization of Hybrid PSI-BLAST using a simple master/slave scheme. The speedup measured on a random sample of 100 queries increases almost linearly with respect to the number of processes. This result demonstrated the suitability for future applications of parallelized Hybrid PSI-BLAST.

We demonstrated in section 6.4 the effectiveness of the new gapped extension heuristics proposed for the hybrid alignment. We examined the miss rate and average execution time per selected sequence and found out that this new heuristics misses fewer potential hits and takes less time to select them on average than the old heuristics approach. Section 6.5 reports the results on the experiments with different sequence weighting methods. The results showed improved sensitivity for Hybrid PSI-BLAST with our generalized position specific weighting scheme, coupled with the blocking technique used in NCBI PSI-BLAST.

In section 6.6.1 we show that the information from suboptimal alignment, otherwise ignored in PSI-BLAST, already improves the sensitivity of PSI-BLAST. More interestingly, we found a set of sequences on which our tool disagrees with the classification given by SCOP. Careful examination points to a possible misclassification in SCOP. Cross-referencing with two other methods of protein structure classification,
CATH and DALI, supports this view. We further enhanced the power of PSI-BLAST for detecting remote homology in database searches by improving the quality of transition probabilities. In section 6.6.2 we describe several additional techniques that can improve the quality of the estimated position-specific transition probability matrix, including a pseudocount approach analog to the one used for estimating emission probability matrix and a method aimed at increasing the signal to noise ratio by enhancing the peaks of the transition probability profiles. The restriction on the maximum gap opening costs also helps to build more sensitive models, which is illustrated in section 6.6.2 as well.

The last technique described in Chapter 6 turned out to be very effective in improving the performance of Hybrid PSI-BLAST. We used the information about the conservedness of the positions, which is derived from the emission probability matrix, to adjust the gap opening costs.

In conclusion, the comparison among different version of the PSI-BLAST showed that our implementation of hybrid alignment with the advantage in statistical significance assessment, model building, and adoption of the position-specific gap penalties, yields a significant improvement in finding distantly related sequences.

More work can be done to make Hybrid PSI-BLAST an even better sequence database search tool. First, we only investigated the estimation of the gap opening cost, which we consider more important than the gap extension cost. A more thorough study on the quality of the gap extension penalty would be desirable. Second, the quality of the emission probability matrix can be improved, as such matrix when built by dedicated tools is still more sensitive than emission probability matrix constructed from the hybrid alignment. Third, iterative search tools that rely on the model refined
through iterations, such as PSI-BLAST, often run into the problem of including non-member sequence in the model building phase. Such “polluted” models often corrupt the $E$-value calculation, leading to unreliable results and skewed performance. So more studies are needed to find ways to exclude those contaminations in the model construction process. Finally, the parameters tested in this dissertation only cover a small range of their possible values. More extensive investigation on the optimization of those parameters would be desirable.
APPENDIX A

CONSTRUCTION OF NEW MODEL

In Chapter 3, we have described some examples to show that new emission probability and transition probability matrices can be derived from the forward-backward algorithm. Here we are giving a detailed account on the equations and their probabilistic interpretations.

Given a set of \( N \) sequences, \( X = \{X_1, X_2, ..., X_N\} \), sequence \( X_k \) consists of \( (x_{k,1}, x_{k,2}, ..., x_{k,n_k}) \). The letters are drawn from a finite alphabet set \( \Theta \). The set \( X \) is used as training sequence set to estimate the new emission and transition probability matrices, for each sequence \( X_k \), the old model \( \Omega \) can be used to derive the forward alignment score matrix, \( f_{i,j}^S \), and the backward alignment score matrix, \( b_{i,j}^S \), by the forward-backward algorithm. \( S \) is one of the three possible states, \( S \in \{M, I, D\} \).

The maximum forward alignment score can be identified as \( Z_{s_E,m_E}^S \), which is achieved at position \( (s_E, m_E) \) on the alignment lattice. This point is referred to as the end of the alignment between sequence \( X_k \) and the model \( \Omega \). However, there is no notion of the beginning of an alignment in hybrid algorithm, since there always exists an alignment path from point \( (0, 0) \), no matter how small the likelihood is. Instead, hybrid alignment will calculate the probability of having an alignment at any given position. If a normal beginning of the hybrid alignment is needed in some situations,
such as printing the alignment, the position for which the probability of having an alignment exceeds a predefined threshold can be marked as the start of the alignment. We will not consider such situation here, as it is not relevant to the discussions.

The forward score, $f^S_{i,j}$, can be interpreted as the total weights of the alignment paths that end in $S$ state at position $(i,j)$. On the other hand, $b^S_{i,j}$ denotes the total weights of alignment paths that run between the point $(i,j)$ and the end point $(s_E, m_E)$. $Z_{s_E, m_E}$ is the total weights of the alignment paths that end at position $(s_E, m_E)$. When we multiply the forward score, say $f^M_{i,j}$, and the backward score, say $b^M_{i,j}$, we will get the sum of weights of all alignment paths that go through the match state $M$ at position $(i,j)$. Thus the probability of observing $M_{i,j}$ being visited in all possible alignment paths is $f^M_{i,j} \cdot b^M_{i,j} / Z_{s_E, m_E}$. If we sum this quantity over the sequence positions indexed by $i$, we will get the probability of visiting the $M$ state at the $j$th model position $Pr(M_j, X_k|\Omega)$.

The following equations are then used to calculate the probabilities of alignment between sequence $X_k$ and the model $\Omega$ that visits match state ($M$) and delete state ($D$) at $j$th model position, respectively:

$$Pr(M_j, X_k|\Omega) = \sum_i f^M_{i,j} \cdot b^M_{i,j} / Z_{s_E, m_E}$$  \hspace{1cm} (A.1)

$$Pr(D_j, X_k|\Omega) = \sum_i f^D_{i,j} \cdot b^D_{i,j} / Z_{s_E, m_E}$$  \hspace{1cm} (A.2)

The same logic does not apply to the insert state, as we will see in a discussion below.

To compute the observed probability for any amino acid $a$ at position $j$, we can use the following equation:

$$Pr(a, M_j, X_k|\Omega) = \sum_{\{i | x_{k,n_i} = a\}} f^M_{i,j} \cdot b^M_{i,j} / Z_{s_E, m_E}$$  \hspace{1cm} (A.3)
To calculate the probability of having an alignment at any model position \( j \), we just need to sum the probabilities of an alignment visiting the match state and the delete state, since the insert state can only be reached through the corresponding match state. Thus:

\[
Pr(Alignment_j, X_k|\Omega) = Pr(M_j, X_k|\Omega) + Pr(D_j, X_k|\Omega)
\] (A.4)

Similarly, the probabilities of observing the transitions from \( M_j \) can be computed as follows:

\[
Pr(M_j \rightarrow M_{j+1}, X_k|\Omega) = \sum_i f^{M}_{i,j} \cdot \eta_j \cdot \omega_j(x_i) \cdot b^M_{i+1,j+1} / Z_{s_E, m_E}
\] (A.5)

\[
Pr(M_j \rightarrow I_j, X_k|\Omega) = \sum_i f^{M}_{i,j} \cdot \mu_{I_j} \cdot b^I_{i,j+1} / Z_{s_E, m_E}
\] (A.6)

\[
Pr(M_j \rightarrow D_{j+1}, X_k|\Omega) = \sum_i f^{M}_{i,j} \cdot \mu_{D_j} \cdot b^D_{i,j+1} / Z_{s_E, m_E}
\] (A.7)

The probabilities of observing the transitions from \( D_j \) is calculated below:

\[
Pr(D_j \rightarrow M_{j+1}, X_k|\Omega) = \sum_i f^D_{i,j} \cdot \eta_j \cdot \mu_{I_j} \cdot b^M_{i+1,j+1} / Z_{s_E, m_E}
\] (A.8)

\[
Pr(D_j \rightarrow D_{j+1}, X_k|\Omega) = \sum_i f^D_{i,j} \cdot \nu_{D_j} \cdot b^D_{i,j+1} / Z_{s_E, m_E}
\] (A.9)

The probability of observing the transition from \( I_j \) to \( M_{j+1} \) is:

\[
Pr(I_j \rightarrow M_{j+1}, X_k|\Omega) = \sum_i f^I_{i,j} \cdot \eta_j \cdot \mu_{I_j} \cdot b^M_{i+1,j+1} / Z_{s_E, m_E}
\] (A.10)

The reason that the insert state should be dealt with differently than the match and delete states for calculating the observed \( I_j \rightarrow I_j \) probability is that the \( I_j \rightarrow I_j \) transition loops over one model position \( j \). If we still sum over the index \( i \) across the sequence positions as we do in above equations, we will include the weights from some
alignment paths repeatedly in the calculation. The result is no long interpretable as probability; instead, it can be viewed as the average insertion length at that model position. On the other hand, we can still compute the gap opening and gap extension probabilities for insertion events if we model the gap extension as a binomial process.

Suppose we have a gap opening probability $\mu_j$ and a gap extension probability $\nu_j$ for insertion at model position $j$. Then the probability to have an insertion of length $l$ can be calculated as: $Pr(l) = \mu_j \cdot \nu_j^{(l-1)}$. Thus the average insertion length:

$$\bar{l} = \sum_{l=1}^{l=\infty} Pr(l) \cdot l$$

$$= \frac{\mu_j}{(1 - \nu_j)^2}$$

$$= \frac{\sum_i f_{\text{m}}^l \cdot \nu_j^l \cdot \text{b}_{i,j+1}^l}{Z_{sE,mE}}$$

(A.11)

Note that $\mu_j$ can be calculated from $Pr(M \rightarrow I, X|\Omega)$ as in Eq. (A.6), therefore $\nu_j$ can be derived from Eq. (A.11).

Since we do not allow $D_j \rightarrow I_j$ or $I_j \rightarrow D_{j+1}$ transitions in our implementation of hybrid alignment in PSI-BLAST, we will not give the corresponding equation here. However, it should be trivial to derive those equations, in case they are permitted.

Once the probabilities of observing different amino acids and transitions for every positions in one training sequence have been computed, the counts of the occurrence of those residues and transitions can be derived for the whole training set.

Thus the number of occurrences of any give amino acid $a$ at model position $j$, and the total number of amino acids that appear at position $j$ are given by the following equations:

$$N_{j,a} = \sum_k Pr(a, M_j, X_k|\Omega)$$

(A.12)

$$N_j = \sum_a N_{j,a}$$

(A.13)
The observed frequency for amino acid \( a \) at position \( j \) is:

\[
    f_{j,a} = \frac{N_{j,a}}{N_j}
\]

As for the counts of the transitions at position \( j \), the equations described below can be used:

\[
    Tr(M_j \rightarrow M_{j+1}) = \sum_k Pr(M_j \rightarrow M_{j+1}, X_k|\Omega) \quad (A.14)
\]

\[
    Tr(M_j \rightarrow I_j) = \sum_k Pr(M_j \rightarrow I_j, X_k|\Omega) \quad (A.15)
\]

\[
    Tr(M_j \rightarrow D_{j+1}) = \sum_k Pr(M_j \rightarrow D_{j+1}, X_k|\Omega) \quad (A.16)
\]

\[
    Tr(I_j \rightarrow M_{j+1}) = \sum_k Pr(I_j \rightarrow M_{j+1}, X_k|\Omega) \quad (A.17)
\]

\[
    Tr(D_j \rightarrow M_{j+1}) = \sum_k Pr(D_j \rightarrow M_{j+1}, X_k|\Omega) \quad (A.18)
\]

\[
    Tr(D_j \rightarrow D_{j+1}) = \sum_k Pr(D_j \rightarrow D_{j+1}, X_k|\Omega) \quad (A.19)
\]
BIBLIOGRAPHY


148


149


