Biological sequence comparison on a parallel computer

Andrew Lyall

Ph.D. Department of Molecular Biology, University of Edinburgh. 1988

ABSTRACT

The late 1970s technological revolution in molecular biology has made it possible to generate DNA sequence and derived protein sequence very rapidly. The international databases that contain these data are already so large that the exhaustive comparison of novel sequences with them is non-trivial. Future developments, including automation of sequence generation and the human genome project, will greatly exacerbate this problem.

Exhaustive sequence comparison algorithms that use the dynamic programming method have been implemented on the ICL DAP. This is a highly parallel computer, the architecture of which is particularly well suited to the efficient execution of such programmes. These implementations achieve impressive cost/performance ratios. Significantly they are more cost effective than implementations of inferior inexhaustive algorithms generally available for serial computers.

The utility of these programmes has been greatly enhanced through the use of empirically derived similarity tables. These tables are however widely regarded as being sub-optimal. The power of the DAP programmes will facilitate the task of collecting the data necessary to update these tables and broaden their scope. This work is in progress.

The programmes take advantage of the DAP architecture to collect a large number of alignments for each search. This feature has made possible the development of a new and rigorous method of assessing the significance of biological sequence alignments.

Five sets of alignments, each generated by exhaustively comparing a novel sequence with a database, are presented. Biological functions and evolutionary relationships suggested by these alignments are discussed.

It is suggested that, in the future, sequence comparison programmes running on parallel computers will become an essential part of experimental Molecular Biology.

Remember, then, that scientific thought is the guide of action; that the truth at which it arrives is not that which we can ideally contemplate without error, but that which we may act upon without fear; and you cannot fail to see that scientific thought is not an accompaniment or condition of human progress, but human progress itself.

William Clifford.

The Common Sense of the Exact Sciences.

CONTENTS

FIGURES

CHAPTER ONE

Molecular Biology

The name Molecular Biology seems first to have been used by Warren Weaver in 1938 [Rees & Sternberg, 1984]. In his report for that year as Natural Sciences Director of the Rockefeller Foundation he writes

> "...in those borderline areas in which physics and chemistry merge with biology, gradually there is coming into being a new brand of science - Molecular Biology - which is beginning to uncover many secrets concerning the ultimate units of the living cell."

Although this may have been the first recognition of the existence of Molecular Biology as a distinct science, the philosophy that underlies it had been understood by at least a few for very much longer.

Possibly the earliest statement of this idea is that made some three centuries ago by Robert Hooke in "Micrographia", his famous book describing observations made with an early microscope. In a philosophical introduction to the work he says

> "...that knowing what is the form of Inanimate or Mineral bodies, we shall be the better able to proceed in our next Enquiry after the forms of Vegetative bodies; and last of all of Animate ones, that seeming to be the highest step of natural knowledge that the mind of man is capable of."

It is this idea, that a detailed understanding of life may be arrived at using the methods of chemistry and physics applied to biological molecules, that provides the motivation for molecular biology.

In 1931 workers using X-ray diffraction to study the structure of natural fibres observed that the diffraction patterns generated were similar to those from crystalline and fibrous specimens of inanimate origin [Astbury & Street, 1931]. Figure 1.1 is a reproduction of some of their X-ray diffraction photographs which must be amongst the earliest data of molecular biology.

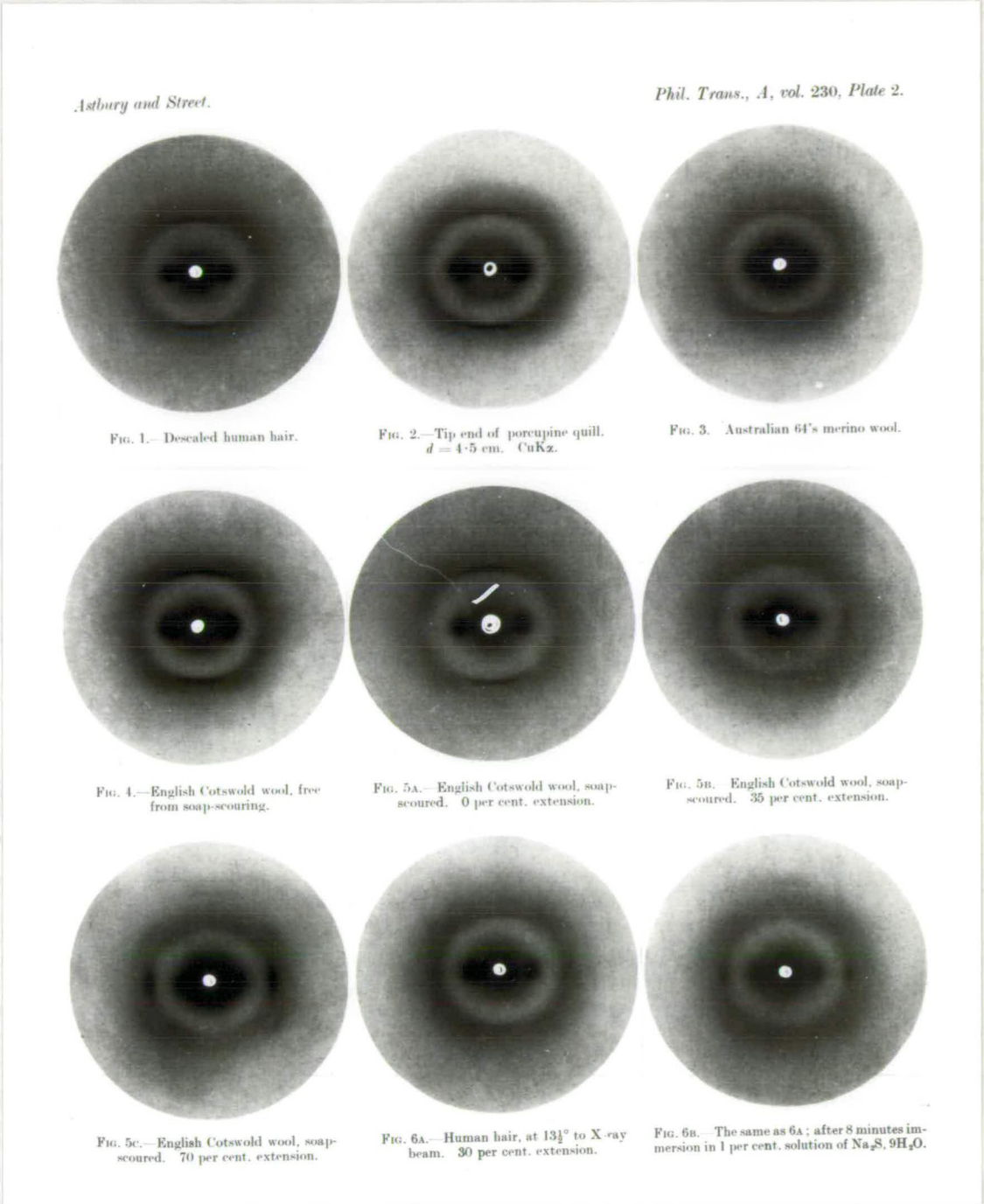


Figure 1.1

X-ray photographs from the early days of Molecular Biology [Astbury & Street, 1931]. Taken in the 1930s, they are of a variety of natural fibres. The order present in the photographs is indicative of order within the biological molecules that make up the fibres.

The order present within the diffraction patterns is clearly visible and is a consequence of periodicities within the structure of the fibre. These photographs constitute a clear demonstration that the biological molecules that make up each particular fibre all have the same structure and are all in the same orientation.

Some twenty years after this early crystallographic work, Astbury, one of the principal investigators, commenced his 1950 Harvey Lecture "Adventures in Molecular Biology" with a description of what he felt molecular biology to be. He said that the name molecular biology

> "...implies not so much a technique as an approach, an approach from the viewpoint of the so-called basic sciences with the leading idea of searching below the large-scale manifestations of classical biology for the corresponding molecular plan."

In describing the subject area of Molecular Biology he went on to say that it was concerned with

> "...the forms of biological molecules, and with the evolution, exploitation and ramifications of those forms in the ascent to higher and higher levels of organization."

and also that it

> "...is predominantly three-dimensional and structural; which does not mean, however, that it is merely a refinement of morphology. It must of necessity enquire at the same time into genesis and function."

Although molecular biology has expanded greatly since these early days, and new technologies have shifted the emphasis away from direct structure determination, this basic philosophy remains.

The Molecules of Life

Biological molecules fall naturally into five major classes; Nucleic Acids, Proteins, Carbohydrates, Fats and a fifth class comprising Vitamins and Co-enzymes. Two of these classes of molecules, nucleic acids and proteins, are linear directional co-polymers; they are made up of long unbranched chains of similar monomers whose order or sequence determines the properties of the molecule. It is the study of these two classes of molecule, the determination of the sequence of their monomers, their

arrangement in three-dimensional space, their interactions, and how, through those interactions, they mediate the processes of life that is the domain of modern molecular biology.

Nucleic Acids

Nucleic acids are of two types, deoxyribonucleic acid (DNA) and ribonucleic acid (RNA). They are both copolymers of four monomers that differ very slightly between DNA and RNA. These monomers are called nucleotides. The nucleotides consist of an invariant part which is a five-carbon sugar-phosphate, and a variable part which is selected from one of four different organic bases. These purine and pyrimidine bases carry the information whilst the sugar-phosphate groups are structural. For this reason, because the base component of the nucleotide is perceived as being more important than the sugar-phosphate, the terms nucleotide and base are used interchangeably to describe the individual nucleotides.

DNA is the Genetic Material. The four deoxyribonucleotides of which it is composed are represented by the letters A, T, G and C. The information stored in the order of the nucleotides that make up DNA is the information that specifies all living things. In its DNA an organism carries information that is general to life, particular to its species and unique to it as an individual. It is as DNA that this information is passed between the generations.

DNA occurs as a double helix with the sense strand, that is the strand from which the information is read, stored along side its complementary strand; each base in the sense strand exactly specifying the corresponding base in the complementary strand and *vice versa* according to the base pairing rules. The helix is plectonemic, that is to say the two strands cannot be separated without twisting. The complementary relationship extends to the topology of the strands which have directionality and are in opposite orientations. This structure is the famous double helix of the 1960's popular press and the story of its discovery is one of the most widely known in all of science. The base-pairing rules derive from the size and shape of the nucleotides which are such that for a DNA double helix to form, A will only pair with T and G will only pair with C. This complementary relationship is utilized by life to replicate the DNA and also to read the information it contains. Figure 1.2 shows the chemical structure of the nucleotides and the

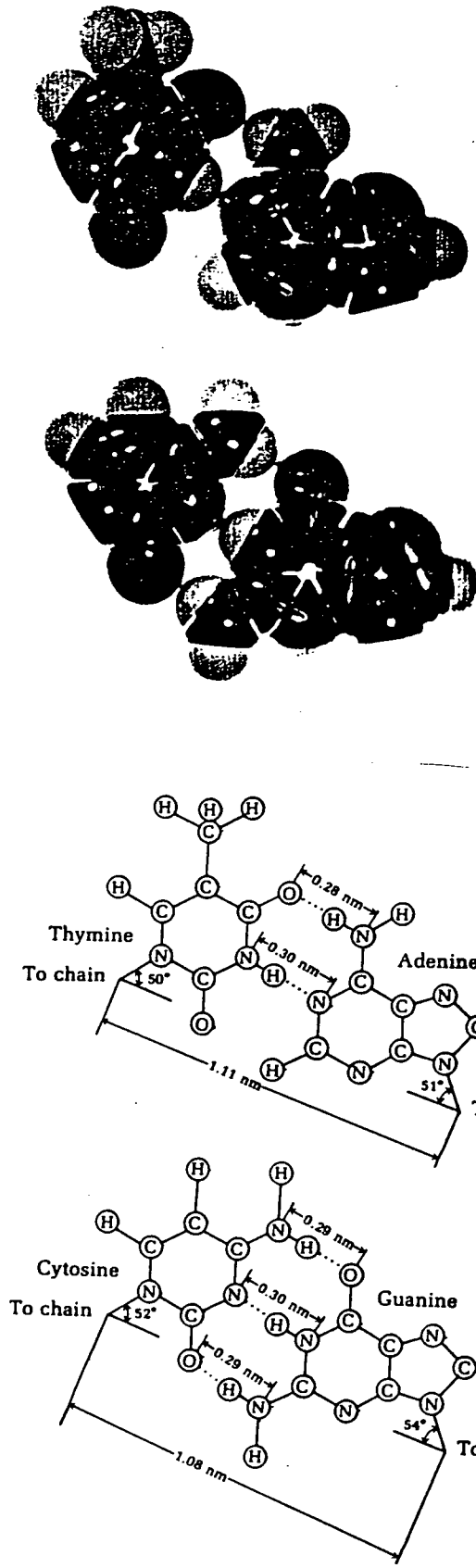


Figure 1.2

The structure of the bases that form the variable part of the nucleotides of which the DNA double helix is composed. The most important aspect of the helix is the specificity of the pairing of the bases. Hydrogen bonding and steric restrictions mean that A can only pair with T and G can only pair with C [After Lehninger, 1970].

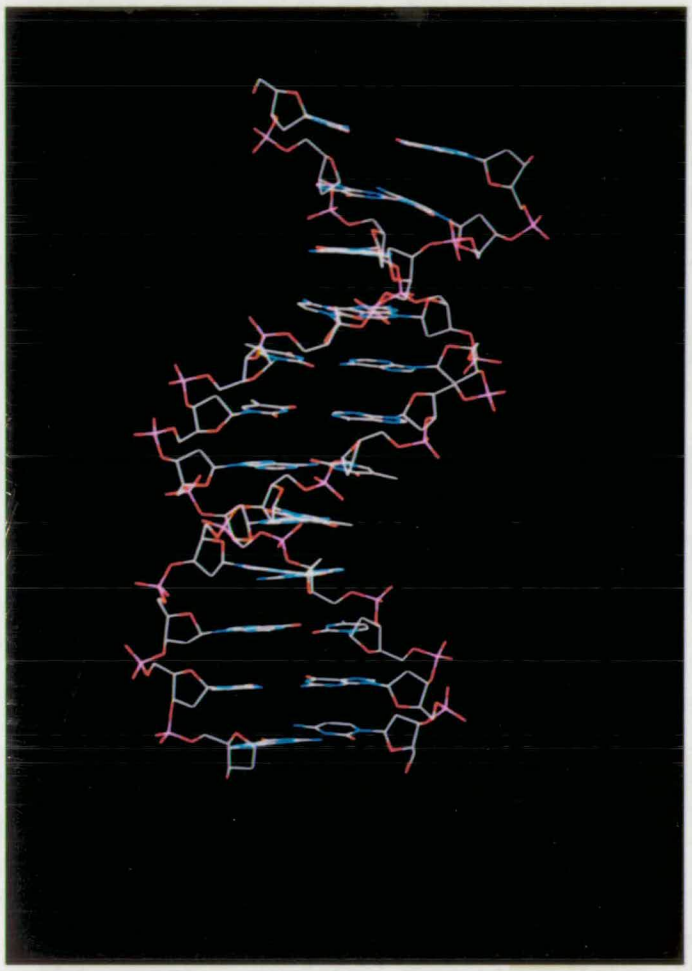


Figure 1.3

A piece of double stranded DNA containing 12 base pairs [Drew *et al.*, 1981] modelled on an Evans and Sutherland PS330 vector graphics system running FRODO [Jones *et al.*, 1978]. The DNA double helix is approximately 20 Ångstrøms across and makes one complete turn approximately every 34 Ångstrøms. The atoms are coloured by type; C - white, N - blue, O - red, P - purple. Hydrogens are not shown. The picture was taken with a 35 mm lens at F11, using 35 mm ASA 200 colour print film exposed for 2 seconds. Within FRODO the contrast was set to 0.5 and the intensity to 1.0.

base-pairing rules. Figure 1.3 is a photograph of a small piece of the DNA double helix modelled on a computer graphics system.

The size of DNA molecules covers a wide range. The entire genome of many organisms consists of just one extremely long DNA molecule, perhaps several million bases long, whilst DNA molecules prepared for experimental purposes may only be a few hundred bases long. Figure 1.4 lists the sizes, in kilobases, of genomes from a range of organisms.

Name	Type	Size (kb)
SV40*	Virus	5.1
øX174*	Phage	5.4
lambda*	Phage	49
Epstein-Barr virus	Virus	170
T2*	Phage	180
Escherichia coli	Bacteria	4,200
*Saccharomyces cerevisiae**	Yeast	13,000
Dictyostelium discoideum	Fungus	54,000
Caernorhabditis elegans	Nematode	80,000
Drosophila melanogaster	Fruit fly	140,000
Bombix mori	Silkworm	300,000
Strongylocentrolus purpuratus	Sea urchin	860,000
Mus musculus	Mouse	2,700,000
Rattus norvegicus	Rat	3,000,000
Xenopus laevis	Frog	3,100,000
Homo sapiens	Man	3,300,000
*Protopterus aethiopicus**	Lungfish	102,000,000

Figure 1.4

The sizes of the genomes of a number of organisms in kilobases (a kilobase is 1000 bases). *[Kornberg, 1974], others [Lewin, 1980].

Figure 1.5 shows two electron micrographs of a human chromosome, the cellular organelle that contains DNA. In the upper picture the chromosome has been disrupted to release its DNA which can be seen as the enormously long thread filling most of the picture. The lower picture shows a similar chromosome that has not been treated in this way for comparison. In this picture the DNA is tightly packed so as to form much of the structure of the chromosome.

DNA sequence is recorded as long linear lists of letters. By convention these are recorded in an order which is referred to as being in the

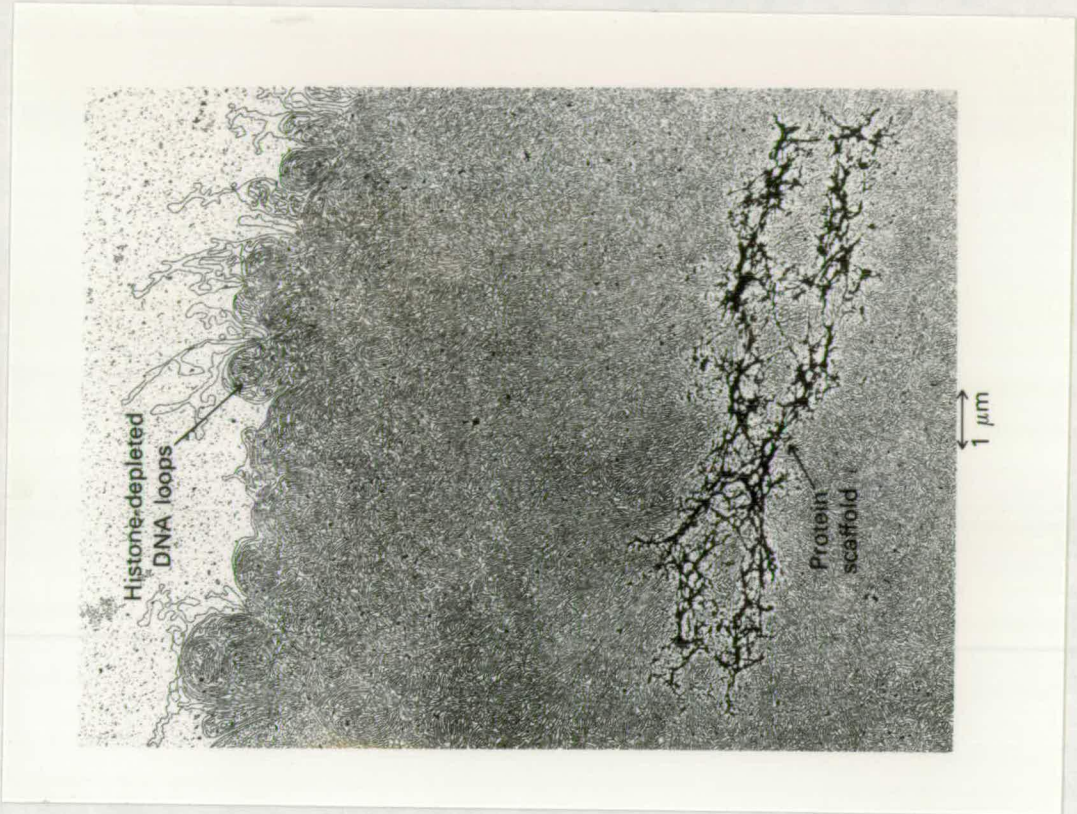
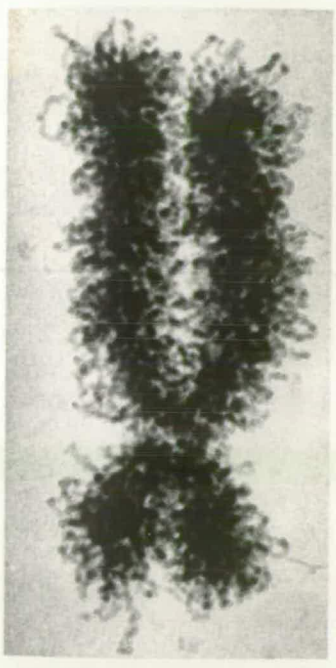


Figure 1.5.

(Above) An electron micrograph of human chromosome number 23 [Paulson & Laemmli, 1977] at metaphase. The chromosome is the cellular organelle that contains the DNA. Each chromosome is thought to consist of a single extremely long DNA molecule. The chromosome has been treated to remove attached protein (histones) and has been osmotically shocked to release the DNA. The DNA can be seen as the immensely long thread filling most of the picture. The vague outline of the familiar X-shape of the chromosome can also be seen. The whorls and patterns within the strand are thought to be indicative of the way in which the DNA is super-coiled in the construction of the chromosome. (Right) A similar chromosome [Du Praw, 1970] that has not been so treated.



-1985 (incorporated)
ssociated retrovirus (arv-2; proviral), complete genome
ed immune deficiency syndrome; complete genome; envelope protein; polyprotein; reverse transcriptase.
ssociated retrovirus, Viridae; Ss-rna enveloped viruses; Retroviridae.
(bases 1-9737)
z-pescador R., Power M.D., Barr P.J., Steimer K.S., Stempien M.M., Brown-shimer S.L., Gee W.W., Renard A.,
oh A., Levy J.A., Dina D., Luciw P.A.;
ptide sequence and expression of an aids-associated retrovirus (arv-2)"; Science 227:484-492(1985).

	From	To	Description
	791	2299	Gag polyprotein
	791	1180	P12 protein
	1193	1885	P25 protein
	1928	2296	P16 protein
	6233	8800	Envelope protein precursor (env)
T	332	332	G in 7a,9b; a in 8a
T	340	340	G in 7a,9b; a in 8a
T	4233	4233	A in 8a; g in 8b
T	4677	4677	T in 8a; c in 8b
T	6215	6215	A in 7d; g in 7a,9b

ed immune deficiency syndrome (aids) is caused by a retrovirus known by four different names, probably
enting four different strains: human t-cell leukemia virus-iii (htlv-iii), aids-associated retrovirus type
), aids virus, and lymphadenopathy-associated virus (lav). it is still unclear with which type of virus it
y associated. the long terminal repeat (ltr) of arv-2 is terminated by an inverted 3 bp repeat ('ctg' and '
rom htlv-i and htlv-ii have a 2 bp inverted repeat. immediately downstream (bases 637-653) from the 5' ltr
uence which is complementary to lys-trna. the rna poly-a site is only tentatively assigned. there are three
g frames not shown in the sites table. the putative pol region extends from base 2092 to 5103. orf-1 (bases
6) probably does not encode a functional peptide. orf-2 (bases 8802 to 9434) is unique in that it extends w
he 3' ltr. it is in a similar position to those in htlv-i and htlv-ii, although there is no apparent nucleo
acid sequence homology. the gag gene encodes a putative p12 peptide, as well as p25 and p16. the pol gene h
gy with htlv-i, rsv and mulv. the env gene probably starts at base 6233, but may start at one of two additi
codons near the 5' end (at base 6299 or 6305). a striking feature of the env gene is that its nh2-terminus
erlap the cooh-terminus of the pol gene. this may set arv apart from other retroviruses.

ce 9737 BP; 3445 A; 1738 C; 2377 G; 2177 U;
IARV2 Length: 9737 28-JUL-1987 16:37 Check: 2542 ..

UGGAAGGGC UAAUUUGGUC CCAAAGAAGA CAAGAGAUCC UUGAUCUGUG GAUCUACCAC ACACAAGGCU ACUUCCCUGA UUGGCAGAAU UACACACCA
GCCAGGGAU CAGAUAUCCA CUGACCUUUG GAUGGUGCUU CAAGCUAGUA CCAGUUGAGC CAGAGAAGGU AGAAGAGGCC AAUGAAGGAG AGAACAACA
UUGUUACAC CCUAUGAGCC UGCAUGGGAU GGAGGACGCG GAGAAAGAAG UGUUAGUGUG GAGGUUUGAC AGCAAACUAG CAUUUCAUCA CAUGGCCCG
AGCUGCAUC CGGAGUACUA CAAAGACUGC UGACAUCGAG CUUUCUACAA GGGACUUUCC GCUGGGGACU UUCCAGGGAG GCGUGGCCUG GGCGGGACU
GGAGUGGCG UCCCUCAGAU GCUGCAUAUA AGCAGCUGCU UUUUGCCUGU ACUGGGUCUC UCUGGUUAGA CCAGAUCUGA GCCUGGGAGC UCUCUGGCU
CUAGGGAAC CCACUGCUUA AGCCUCAAUA AAGCUUGCCU UGAGUGCUUC AAGUAGUGUG UGCCCGUCUG UUGUGUGACU CUGGUAACUA GAGAUCCCU
GACCCUUUU AGUCAGUGUG GAAAAAUCUC UAGCAGUGGC GCCCGAACAG GGACGCGAAA GCGAAAGUAG AACCAGAGGA GCUCUCUCGA CGCAGGACU
GCUUGCUGA AGCGCGCACA GCAAGAGGCG AGGGGCGGCG ACUGGUGAGU ACGCCAAUUU UUGACUAGCG GAGGCUAGAA GGAGAGAGAG AUGGGUGCG
AGCGUCGGU AUUAAGCGGG GGAGAAUUAG AUAAAUGGGA AAAAAUUCGG UUAAGGCCAG GGGGAAAGAA AAAAUAUAAG UUAAAACAUA UAGUAUGGG
AGCAGGGAG CUAGAACGAU UCGCAGUCAA UCCUGGCCUG UUAGAAACAU CAGAAGGCUG CAGACAAAUA UUGGGACAGC UACAGCCAUC CCUUCAGAC
GAUCAGAAG AACUUAGAUC AUUAUAUAAU ACAGUAGCAA CCCUCUAUUG UGUACAUCAA AGGAUAGAUG UAAAAGACAC CAAGGAAGCU UUAGAGAAG
AGAGGAAGA GCAAAACAAA AGUAAGAAAA AGGCACAGCA AGCAGCAGCU GCAGCUGGCA CAGGAAACAG CAGCCAGGUC AGCCAAAAUU ACCCUAUAG
CAGAACCUA CAGGGGCAAA UGGUACAUCA GGCCAUAUCA CCUAGAACUU UAAAUGCAUG GGUAAAAGUA GUAGAAGAAA AGGCUUUCAG CCCAGAAGU
UACCCAUGU UUUCAGCAUU AUCAGAAGGA GCCACCCCAC AAGAUUUAAA CACCAUGCUA AACACAGUGG GGGGACAUCA AGCAGCCAUG CAAAUGUUA
AGAGACUAU CAAUGAGGAA GCUGCAGAAU GGGAUAGAGU GCAUCCAGUG CAUGCAGGGC CUAUUGCACC AGGCCAAAUG AGAGAACCAA GGGGAAGUG
AUAGCAGGA ACUACUAGUA CCCUUCAGGA ACAAAUAGGA UGGAUGACAA AUAAUCCACC UAUCCCAGUA GGAGAAAUCU AUAAAAGAUG GAUAAUCCU
GAUUAAAUA AAAUAGUAAG AAUGUAUAGC CCUACCAGCA UUCUGGACAU AAGACAAGGA CCAAAGGAAC CCUUUAGAGA UUAUGUAGAC CGGUUCUAU
AACUCUAAG AGCCGAACAA GCUUCACAGG AUGUAAAAAA UUGGAUGACA GAAACCUUGU UGGUCCAAAA UGCAAACCCA GAUUGUAAGA CUAUUUUAA
GCAUUGGGA CCAGCAGCUA CACUAGAAGA AAUGAUGACA GCAUGUCAGG GAGUGGGGGG ACCCGGCCAU AAAGCAAGAG UUUUGGCUGA AGCCAUGAG
AAGUAACAA AUCCAGCUAA CAUAAUGAUG CAGAGAGGCA AUUUUAGGAA CCAAAGAAAG ACUGUUAAGU GUUUCAAUUG UGGCAAAGAA GGGCACAUA
CAAAAAUUG CAGGGCCCCU AGGAAAAAGG GCUGUUGGAG AUGUGGAAGG GAAGGACACC AAAUGAAAGA UUGCACUGAG AGACAGGCUA AUUUUUUAG
AAGAUCUGG CCUUCCUACA AGGGAAGGCC AGGGAAUUUU CUUCAGAGCA GACCAGAGCC AACAGCCCCA CCAGAAGAGA GCUUCAGGUU UGGGGAGGA
AAACAACUC CCUCUCAGAA GCAGGAGCCG AUAGACAAGG AACUGUAUCC UUUAACUUCC CUCAGAUCAC UCUUUGGCAA CGACCCCUCG UCACAAUAA
AUAGGGGGG CAACUAAAGG AAGCUCUAUU AGAUACAGGA GCAGAUGAUA CAGUAUUAGA AGAAAUGAAU UUGCCAGGAA AAUGGAAACC AAAAAUGAU
GGGGAAUUG GAGGUUUUAU CAAAGUAAGA CAGUACGAUC AGAUACCUGU AGAAAUCUGU GGACAUAAAG CUAUAGGUAC AGUAUUAGUA GGACCUACA
UGUCAACAU AAUUGGAAGA AAUCUGUUGA CUCAGAUUGG UUGUACUUUA AAUUUCCCCA UUAGUCCUAU UGAAACUGUA CCAGUAAAAU UAAAGCCAG
AUGGAUGGC CCAAAAGUUA AGCAAUGGCC AUUGACAGAA GAAAAAAUAA AAGCAUUAGU AGAGAUAUGU ACAGAAAUGG AAAAGGAAGG GAAAAUUUC
AAAUUGGGC CUGAAAAUCC AUACAAUACU CCAGUAUUUG CUAUAAAGAA AAAAGACAGU ACUAAAUGGA GAAAACUAGU AGAUUUCAGA GAACUUAAU
AAGAACUCA AGACUUCUGG GAAGUUCAGU UAGGAAUACC ACACCCCGCA GGGUUAAAAA AGAAAAAAUC AGUAACAGUA UUGGAUGUGG GUGAUGCAU
UUUUCAGUU CCCUUAGAUA AAGACUUUAG AAAGUAUACU GCAUUUACCA UACCUAGUAU AAACAAUGAG ACACCAGGGA UUAGAUAUCA GUACAAUGU
UGCCACAGG GAUGGAAAGG AUCACCAGCA AUAUUCCAAA GUAGCAUGAC AAAAAUCUUA GAGCCUUUUA GAAAACAGAA UCCAGACAUA GUUAUCUAU
AUACAUGGA UGAUUUGUAU GUAGGAUCUG ACUUAGAAAU AGGGCAGCAU AGAACAAAAA UAGAGGAACU GAGACAGCAU CUGUUGAGGU GGGGAUUUA
CACACCAGAC AAAAAACAUC AGAAAGAACC UCCAUUCCUU UGGAUGGGUU AUGAACUCCA UCCUGAUAAA UGGACAGUAC AGCUAUUGC GCUGCCAGA
AAGACAGCU UUAUGUAAA
CCUUAGAGG AACCAGUAC
UGAAGUAUAU AUUUAAAAA
CUGAAAACAG AGCAUAGUA
AUGGGGAAA CUGAGUGGG
UUUGUCAAU AGCUAAUAG
AGACUAAAU UUACAAGCA
UCAUCUAGC AGAGUGAAU
GAGUUAGUC UGAACAAGU
AUAAAUUAG UGGAGAGCA
GGCUAGUGA GACAAGUAG
UGUAGUCCA AGCAGAAGU
AUUCCAGCAG AGCAAUUUC
CAGUACUAC AAUCUAUGA
AAUGAAUUA UAAAAGAAA
GGGGGAUUG AAAAUUCAA
AUUUUCGGGU UUAUUACAGG GACAACAAAG AUCCCCUUUG GAAAGGACCA GCAAAGCUUC UCUGGAAAGG UGAAGGGGCA GUAGUAAUAC AAGAUAAUA
UGACAUAAAA GUAGUGCCAA GAAGAAAAGC AAAAAAUCAUU AGGGAUUAUG GAAAACAGAU GGCAGGUGAU GAUUGUGUGG CAAGUAGACA GGAUGAGGA
UAGAACAUGG AAAAGUUUAG UAAAACACCA UAUGUAUAUU UCAAAGAAAG CUAAAGGAUG GUUUUAUAGA CAUCACUAUG AAAGUACUCA UCCAAGAGU
AGUUCAGAAG UACACAUCCC CCUAGGGGAU GCUAAAUUGG UAAUAACAAC AUAUUGGGGU CUGCAUACAG GAGAAAGAGA AUGGCAUUUG GGCCAGGGA
UCGCCAUAGA AUGGAGGAAA AAGAAAUAUA GCACACAAGU AGACCCUGGC CUAGCAGACC AACUAAUUCA UCUGCAUUAU UUUGAUUGUU UUUCAGAAU
UGCUAUAAAA AAUGCCAUAU UAGGAUAUAG AGUUAGUCCU AGGUGUGAAU AUCAAGCAGG ACAUAACAAG GUAGGAUCUC UACAAUACUU GGCACUAGC
GCAUUAAUAA CACCAAAAAA GACAAAGCCA CCUUUGCCUA GUGUUAAGAA ACUGACAGAG GAUAGAUGGA ACAAGCCCCA GAAGACCAAG GGCCACAGA
GAGCCAUAC AAUGAAUGGA CACUAGAGCU UUUAGAGGAG CUUAAGAGAG AAGCUGUUAG ACAUUUUCCU AGGCCAUGGC UCCAUAGCUU AGGACAAUA

Figure 1.6

Part of the entry from release 12 of the EMBL nucleic acid sequence data base [EMBL, 1988] that records the complete genome of the human virus ARV-2. Such sequences and their ancillary information are accessed using the keys present in the first two columns of the records. Although the information is stored in a format that is human-readable it would normally be accessed automatically. Most of the 9737 nucleotides and much of the text are not shown.

5' to 3' direction. This is because it corresponds to the order in which the 5' and 3' carbon atoms of the deoxyribose monomers occur in the DNA backbone. Only one of the two complementary strands is recorded. Should it be necessary the sequence of the complementary strand can be generated using the base pairing rules. Figure 1.6 is the symbolic representation of some of the DNA that makes up the genome of a virus.

RNA is chemically very similar to DNA. The four ribonucleotides of which it is comprised are represented by the letters A, U, G and C. The small chemical difference between the deoxyribonucleotides that make up DNA and the ribonucleotides that make up RNA have the effect of making RNA much less stable. The reduced chemical stability of RNA is reflected in its roles in life which often make use of this property.

RNA is involved in the process of expressing, as protein, the information present in the DNA. Many RNAs have a structural role and a few have a catalytic role. It has been suggested that RNA was the original molecule of life, pre-dating both DNA and protein [Orgel, 1968]. As far as its primary sequence is concerned RNA can usually be considered in the same way as DNA. Some virus genomes are made of RNA rather than DNA.

Proteins

Proteins are directed co-polymers of twenty different amino acids. In this context amino acid is used rather loosely as one of them, proline, is an imino acid. As with the four nucleotides that make up nucleic acids, the twenty amino acids of proteins contain a common portion which allows them to polymerize and a unique portion which gives them their chemical identity. The widely differing physical and chemical properties of proteins are conferred on them by the different physical and chemical properties of the amino acids of which they are comprised. Figure 1.7 shows the chemical structures of the amino acid side chains. Proteins range in size from about a hundred to less than a thousand amino acids in length. Sometimes several different proteins will be synthesized as a single molecule and subsequently processed to produce the individual proteins. Such polyproteins often occur in the databases as single sequences which can be as much as 5000 residues long. Proteins are the expression of much of the information stored in DNA. Virtually all the chemical reactions of life are catalysed by proteins; protein catalysts are called enzymes. Proteins also provide much of the structure of

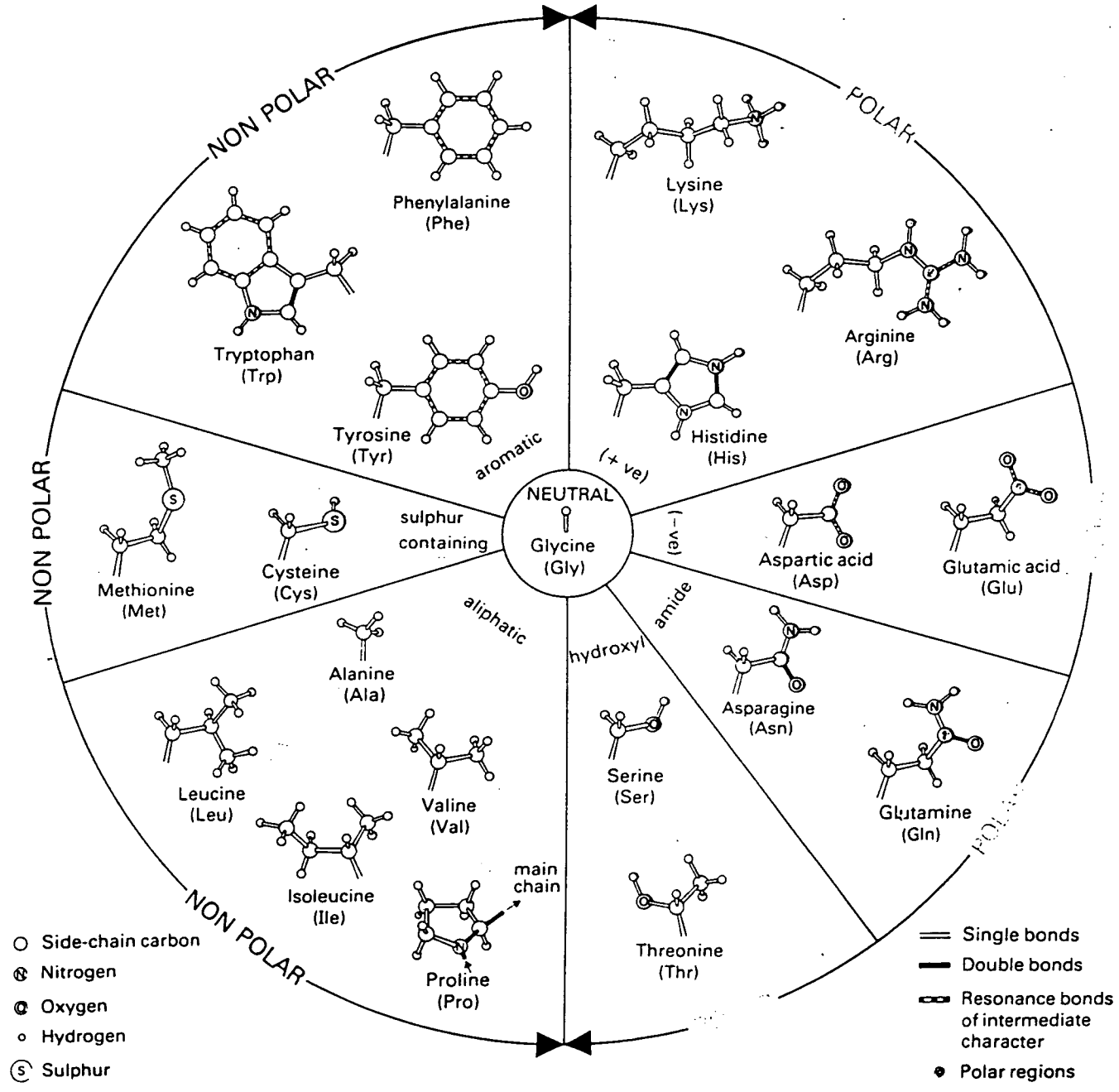


Figure 1.7

The structural formulae of the 19 amino acids and 1 imino acid commonly found in proteins. With the exception of proline, they have the generic formulae NH_2-CHR-COOH. They have been grouped according to the chemical nature of the R group. It is the variety of the chemical properties of the R groups or side chains that is responsible for the huge number of different proteins. Only the R groups have been drawn. [After Rees & Sternburg, 1984].

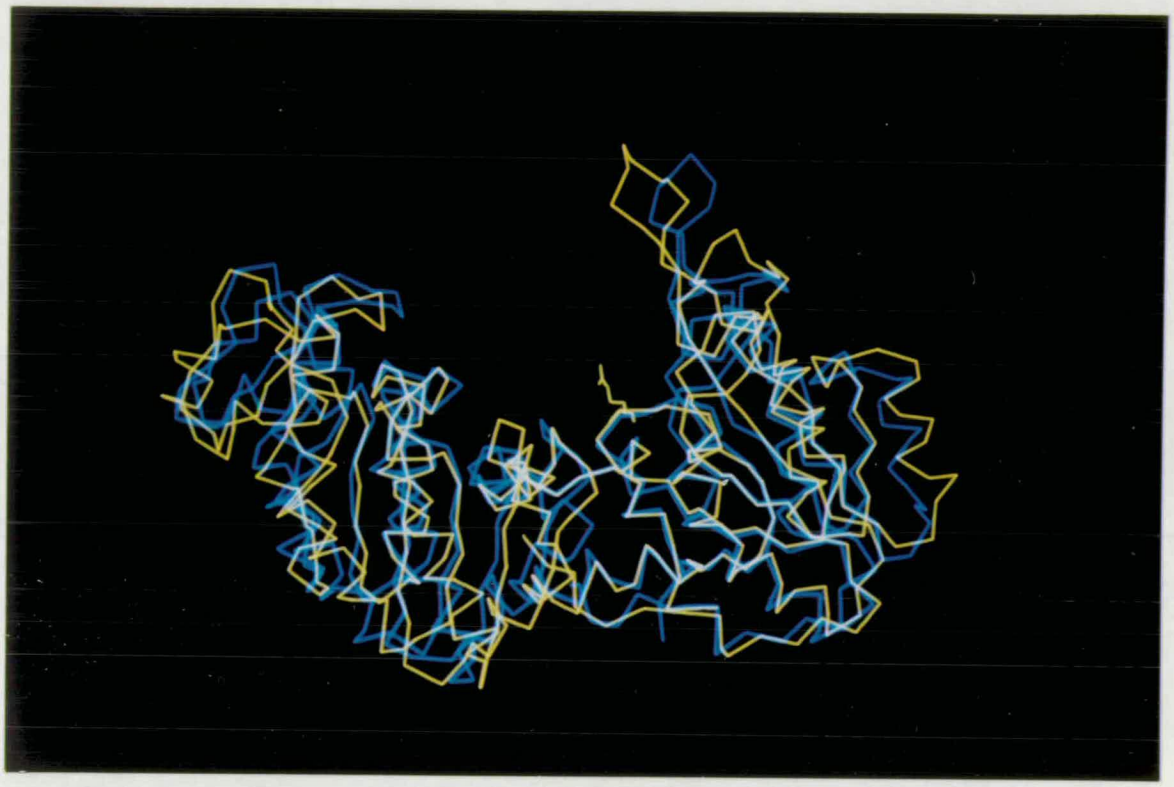


Figure 1.8

The glycolytic enzyme Phophoglycerate kinase displayed on an Evans and Sutherland PS330 vector graphics system running FRODO [Jones *et al.*, 1978]. The length of the long axis of the molecule is approximately 70 Ångstrøms. The structure in yellow is the enzyme from yeast [Watson *et al.*, 1982] and in blue from horse [Banks *et al.*, 1979]. The white regions are where the two structures overlap. It can be seen that the two structures are very similar. This similarity is mirrored by the kinetic and chemical properties of the two enzymes which are also extremely similar. With the exception of Arginine 168 of the yeast structure, which is involved in a site directed mutagenesis experiment described in Chapter 5, none of the side-chains are shown. The picture was taken with a 35 mm lens at F11, using 35 mm ASA 200 colour print film exposed for 2 seconds. Within FRODO the contrast was set to 0.5 and the intensity to 1.0.

living things.

Figure 1.8 is a photograph of a single molecule of the enzyme phosphoglycerate kinase modelled on a computer graphics system and figure 1.9 is the symbolic representation of the 417 amino acids of which it is comprised. By convention protein sequence is recorded with the amino acids running from the N-terminal to the C-terminal end of the protein. This is the order in which proteins are synthesized by biological systems and corresponds to the 5' to 3' direction in which nucleic acid sequence data is conventionally stored.

The relationship between nucleic acids and proteins implied above is exact and known; it is called the genetic code and is illustrated in figure 1.10.

Within the living cell, molecular machinery copies portions of the information present in the permanent and stable DNA of the chromosome into smaller complementary and ephemeral molecules of RNA; this process is called transcription and obeys the base pairing rules. Some of these RNAs (messenger RNA or mRNA) are then used as templates for protein synthesis; this process is called translation and is performed in accordance with the genetic code by a subcellular body or organelle called the ribosome. These proteins then fold to a unique three-dimensional structure probably using only the information present in their sequence. Such proteins along with various RNAs control and mediate these and all other processes of life. The flow of information in life from DNA, through RNA to protein is illustrated in figure 1.11.

Sequencing

One of the major tasks of modern experimental molecular biology is that of working out the order of the bases in nucleic acids and the order of the amino acids in proteins. These activities are referred to as nucleic acid sequencing and protein sequencing respectively.

The extraordinarily elegant chemical and enzymological techniques and the ingenious strategies that have been devised to perform sequencing are the result of a great deal of work by many scientists over the last 25 years. The various milestones in the field have been recognized by Nobel prizes.

```
Phosphoglycerate kinase (EC 2.7.2.3) - Human and horse

C;Species: Homo sapiens (man); Equus caballus (domestic horse)

R;Michelson, A.M., Markham, A.F., and Orkin, S.H. Proc. Nat. Acad. Sci. USA 80, 472-476,
1983 (Human liver, sequence translated from the mRNA sequence)
A;The initiator Met is not shown.

R;Huang, I.-Y., Welch, C.D., and Yoshida, A.J. Biol. Chem. 255, 6412-6420, 1980 (Human
erythrocytes, complete sequence with experimental details)
A;This sequence differs from that shown in having an additional residue, Lys, following 38
Arg, in lacking residue 417, and in the amidation states of residues 52, 109, 275, 299,
336, and 385.

R;Merrett, M. J. Biol. Chem. 256, 10293-10305, 1981 (Horse, complete sequence with
experimental details)
A;This sequence differs from that shown in having 148-Thr, 294-His, 317-Thr, 328-Ala, and
416-Val; in lacking residue 417; in the transposition of residues 68-69; and in the
amidation states of residues 12, 78, 109, and 267.

R;Banks, R.D., Blake, C.C.F., Evans, P.R., Haser, R., Rice, D.W., Hardy, G.W., Merrett, M.,
and Phillips, A.W. Nature 279, 773-777, 1979 (Horse muscle, X-ray crystallography, 2.5
angstroms)
A;The amino end is acetylated.
A;The structure consists of two discrete, globular domains that are joined by residues 186
189 and correspond to the amino- and carboxyl-terminal halves of the sequence, except that
residues 405-416 form a helix associated with the amino-terminal domain.
A;Residues thought to be involved in ADP-ATP binding are Glu-343 to ribose and Lys-219 to
the alpha-phosphate group. The adenine ring is in a slot bounded by residues 212-214, 236
238, and 339-341.


   NBRF:KIHUG  Length: 417  April 21, 1988  18:20  Check: 2600  ..

     1  SLSNKLTLDK LDVKGKRVVM RVDFNVPMKN NQITNNQRIK AAVPSIKFCL
    51  DNGAKSVVLM SHLGRPDGVP MPDKYSLEPV AVELKSLLGK DVLFLKDCVG
   101  PEVEKACANP AAGSVILLEN LRFHVEEEGK GKDASGNKVK AEPAKIEAFR
   151  ASLSKLGDVY VNDAFGTAHR AHSSMVGVNL PQKAGGFLMK KELNYFAKAL
   201  ESPERPFLAI LGGAKVADKI QLINNMLDKV NEMIIGGGMA FTFLKVLNNM
   251  EIGTSLFDEE GAKIVKDLMS KAEKNGVKIT LPVDFVTADK FDENAKTGQA
   301  TVASGIPAGW MGLDCGPESS KKYAEAVTRA KQIVWNGPVG VFEWEAFARG
   351  TKALMDEVVK ATSRGCITII GGGDTATCCA KWNTEDKVSH VSTGGGASLE
   401  LLEGKVLPGV DALSNIL
```

Figure 1.9

The entry for human and horse phophoglycerate kinase from the NBRF protein sequence database release 12 [NBRF, 1988]. It was retrieved using FETCH from the UWGCG suite [Devereau *et al.*, 1984] and is in UWGCG format.

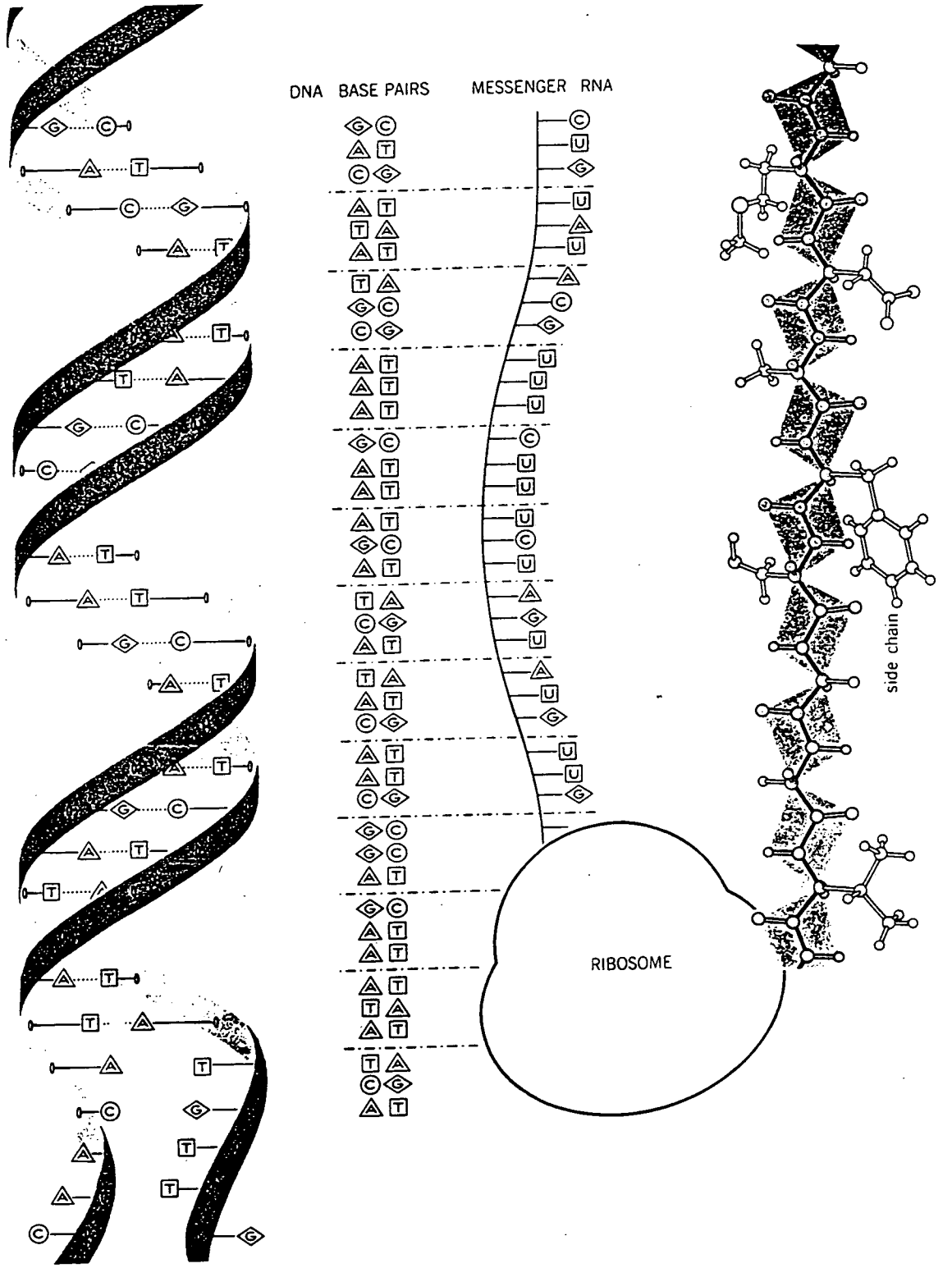


Figure 1.11

The flow of information in life. The information stored in the sequence of bases of the DNA double helix (left) is transcribed using the base pairing rules. The mRNA so produced (centre) is translated using the genetic code to produced a polypeptide chain (right) which folds, probably using only the information contained in its sequence, to produce a protein [After Dickerson & Geis, 1969].

All sequencing is based on a "divide and conquer" approach; the DNA and protein molecules are too large to have their sequence determined directly so they are broken into smaller sections that can be sequenced. Several different sets of such fragments are generated and sequenced. The different sets of fragments can then be compared and their order within the whole molecule inferred. If sufficient fragments have been sequenced it will be possible to generate the sequence of the whole molecule.

DNA sequencing

In 1977 a technique for the fast sequencing of DNA was devised that revolutionized molecular biology [Sanger *et al.*, 1977]. It is referred to either as Sanger sequencing after its inventor or as dideoxy sequencing after the dideoxynucleotides without which it would not be possible. Before the invention of this technique, nucleic acid sequencing was extremely slow and laborious [Peattie, 1979], [Maxam & Gilbert, 1977] but as a result of this development it suddenly became possible to generate large quantities of sequence fast; up to a thousand bases of sequence per technician day in a well equipped laboratory [Anderson *et al.*, 1981].

The technique utilizes the property of complementarity to synthesize DNA that is the complement of the piece of DNA of interest using a bacterial DNA polymerase enzyme. Four *in vitro* reaction mixtures are prepared containing the DNA to be sequenced in its single stranded form, DNA polymerase and the four nucleotides that comprise DNA. One of the nucleotides is labelled (current technologies use a radioactive atom; either ^{32}P or ^{35}S) in order to make DNA synthesized in the reaction detectable. Each of the four reaction mixtures has a limitingly small amount of one of the four chain-terminating dideoxynucleotides added. These are synthetic molecules of a structure almost identical to the deoxynucleotides that make up DNA with the difference that once they have been incorporated into a growing DNA polymer they prevent further polymerization from taking place. The conditions are adjusted appropriately in order that each of the four reactions synthesizes a whole series of molecules of DNA of increasing length, terminated in all possible positions by the appropriate dideoxynucleotide. These molecules can then be separated by high resolution gel chromatography in four adjacent tracks, one for each nucleotide, and the sequence read directly from an autoradiograph of the gel. Figure 1.12 is a photograph of such a gel. The sequence of the original DNA molecule can be

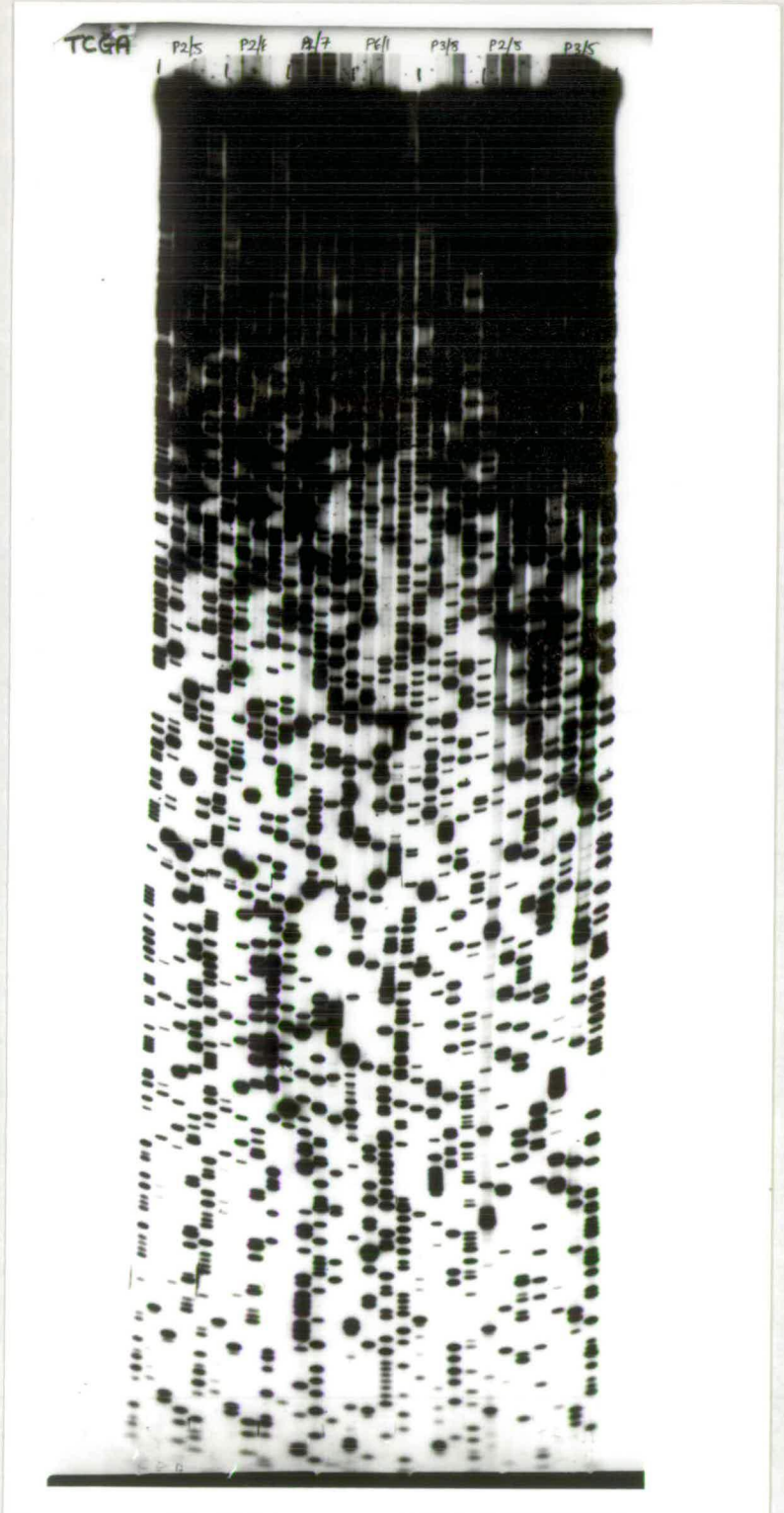


Figure 1.12

An auto-radiograph of a DNA sequencing gel. This particular gel has seven sequencing experiments on it. Each set of four tracks reports the sequence of a particular DNA molecule. The sequences are 'read' from the gel as described in the text.

generated from the sequence read from the gel using the base pairing rules.

Protein sequencing

Protein sequencing relies on a series of enzymatic and chemical techniques that cleave polypeptide chains at specific residues to produce sets of shorter fragments [Bailey, 1967]. These fragments are separated from each other and purified on the basis of size and charge using high voltage paper electrophoresis or, more recently, column chromatography. Unlike the whole protein which is too large and complex these fragments can have their sequence determined directly by chemical methods. Using very pure chemicals and a sequencing machine, fragments as long as 50 amino acids can be sequenced [Edman & Begg, 1967]. As with nucleic acid the sequence of the whole molecule is deduced by sequencing a number of sets of overlapping fragments.

In addition to determining protein sequence directly it is possible to search DNA sequence, which is far easier to generate, for regions that code for proteins and to infer the sequence of the protein from the DNA sequence. There are however many problems associated with this process.

In simple organisms, including virtually all prokaryotes and some simple eukaryotes, coding sequences are contiguous and are bounded by reasonably well defined signal sequences which can be easily recognized; protein sequences can be determined by looking for such open reading frames. In higher organisms things are not so easy. Almost certainly because their greater complexity requires more complex control mechanisms, the proteins of higher organisms are not usually co-linear with the DNA sequence from which they are derived; the coding sequences (exons) are interleaved with intervening sequences (introns) along the genome. During the course of gene expression in such an organism the introns are removed and the dispersed exons brought together to form a mRNA containing only the exons. This so-called mature mRNA is translated into the protein. The removal of the introns is called splicing. As it is still poorly understood, attempts to try to simulate splicing in order to determine the sequence of the protein from DNA sequence are likely to fail. It should be noticed that this problem is exacerbated by the consistent three to one correspondence of the genetic code. If, when trying to simulate splicing, an intron is incorrectly excised by an amount that is not an exact multiple of three, all of the

resulting hypothetical protein sequence downstream of the error will be incorrect.

An additional problem derives from the fact that the techniques for the determination of DNA sequence are prone to errors of insertion and deletion. Whilst a small number of errors of this type do not have a significant impact on the information content of DNA sequence the effect of such errors on the translated protein sequence will be disastrous. Once again, because of the consistent three to one correspondence of the genetic code, it is likely that all data downstream of an insertion or deletion error will be erroneous.

Less serious problems are posed by post-translational modifications; the collective name for the changes that are made to proteins after they have been synthesized. Such changes tend not to have the serious effects on the information content of the sequence that the problems discussed above have, however, they are usually of biological significance. Post-translational modifications vary from minor changes such as blocking of termini or phosphorylation of serine and tyrosine side-chains through glycosylation, to large scale enzymatic cleavage and ligation. Examples of this last case are virus polyproteins which are translated as one very large molecule which is then cleaved into a number of separate functional units. The current state of knowledge makes it impossible to correctly predict whether such events take place for a particular sequence.

For these reasons it is very important to know, when dealing with protein sequence, whether it was determined directly or inferred from DNA sequence. In the latter case, although it is quite likely that the sequence will be incorrect to a greater or lesser degree, there are still ways available to increase confidence in the correctness of the sequence. For example, many organisms show a skewed and highly characteristic utilization of the genetic code. In such cases, when presented with a hypothetical protein from such an organism, it is a trivial matter to test it to see if it conforms to a particular codon usage.

Future developments

At the time of writing, methods of automating sequence collection are being developed. Two distinct approaches are being followed; programming

robots to perform the current techniques [Wada, 1986] and developing completely new technologies that can be performed by a simple machine. The first approach has the advantage of using existing technologies whilst the second promises to be much cheaper [Knobeloch, 1988].

A technique that uses fluorescence detection instead of radioactivity is under development [Smith *et al.*, 1986]. The four reaction mixtures of the dideoxy technique described above are combined using chain-terminating dideoxynucleotides that have been chemically labelled with four different fluorescent dyes. This mixture is separated using an appropriate physical separation technique. The material leaving the separator is excited with a laser beam and a photodetector placed at 90° detects the four different fluorescences corresponding to the four different bases. The machine will be connected to a computer which will record the sequence directly.

As well as having many technical advantages, this technique also overcomes the limited resolution of a statically read gel; the reaction mixture can be run through completely until there are no more fragments. When such machines become generally available the rate of production of sequence data will increase dramatically and the problems addressed by this project will become even more severe.

Sequence Storage & Retrieval

A number of international organizations have come into being for the purpose of collecting sequence data and making it available to scientists at minimal cost. They distribute the data in a variety of formats. This is rather inconvenient however discussions are currently under way with a view to unifying the formats.

When compared with the traditional commercial database applications the sequence databases are not large, however, they do have very fast growth rates. Figure 1.13 shows the rate of growth the EMBL nucleic acid sequence database from its inception until the present. The agencies that collect the data are already having difficulty keeping up to date [Hamm *et al.*, 1987].

In addition to the sequence itself there is a slightly larger quantity of ancillary data. This includes information as to the source of the sequence, bibliographic information and details of the relationships and functions that

Artificial	Chloroplast	Elements	Mitochondrial	Prokaryotic	Viral/Phage	Eukaryotic	unclassified	unannotated	Total
4688	5873	17015	74422	58288	164692	260455	0	0	585433
8238	12616	18341	109008	134270	391292	440682	0	0	1114447
9594	20718	24471	124480	223892	494356	757352	0	0	1654863
10163	29495	24471	150068	293437	720041	919530	0	0	2147205
14066	51401	27180	170201	392693	834979	1362366	21607	0	2874493
41962	83832	36948	255461	625207	1188124	2326404	9654	0	4567592
46073	119852	37804	303549	763188	1344597	2997022	10553	0	5622638
52672	128219	42990	316905	879633	1446696	3475686	10239	0	6353040
68540	153786	43857	346721	1130637	1689681	4364797	15195	0	7813214
71237	465378	43857	376392	1305116	1975030	5465260	64678	0	9766948
76608	482334	43857	387809	1422327	2179141	6124513	42405	1430789	12189783
85281	500141	43857	403542	1583162	2314384	6863698	47902	1766094	13638061

Figure 1.13

The growth of the European Molecular Biology Laboratory (EMBL) Nucleic Acid Sequence database from its inception until the present day. The rows in the table correspond to successive releases of the database. The first row is for release 1 which was in June 1982. The final row is for release 12 which was in November 1987. The numbers are the total number of nucleotides contained in the database at each release. The introduction of unnanotated entries at release 11 is an undesirable stop-gap forced on the database administrators by the quantity of data that they were receiving.

have been assigned to various parts of the sequence. It may well be that as understanding increases the quantity of this kind of data will come to exceed considerably the quantity of sequence it is describing.

In view of the fact that the majority of computing activity is in the field of data processing it would not be unreasonable to assume that "off-the-shelf" systems exist that could be used to store and retrieve sequence data. Unfortunately this naïve assumption turns out to be incorrect. The models that have been developed for the handling of data in commerce are not at all suitable for sequence data. It is true that the ancillary data that accompanies the sequence would fit very well into the relational or hierarchical models of conventional data processing; however, the same cannot be said for the sequence data itself.

If the data atom is taken to be the individual sequence, then the system would have to be able to cope with fields ranging in size from a ten amino acid neuro-peptide to the 170,000 base genome of the Epstein-Barr virus; commercial database systems are unable to do this. If the individual sequence element is the atom then the system is faced with tens of millions of one-byte records; a ludicrous situation. Although it might be possible to divide the sequence up into reasonably sized pieces and store these pieces one per record, there is no biological rationale that could be used to decide how to divide the sequences and any arbitrary method that was chosen would make the retrieval programmes unnecessarily complicated.

In practice the approach that has been followed is to store the data as flat files and to generate indices to provide fast access to them. The PSQ and NAQ systems of the Protein Identification Resource [George *et al.*, 1988] and the GCG package of the University of Wisconsin [Devereux *et al.*, 1984] use this approach and are widely used. It is possible to access sequences by name, author, feature and keyword. These systems provide rudimentary query languages that include logical operators for combining the search criteria in useful ways.

CHAPTER TWO

Sequence Comparison

In essence, the aim of sequence comparison in molecular biology is to find similarities, variously called homologies, matches or alignments, between the strings of letters that represent nucleic acid and protein molecules and to assign values to them. Such similarities are used to ascribe function or evolutionary relationships to the molecules and the values assigned to them are used to compare groups of sequences.

It has been suggested [Reeck *et al.*, 1987] that the terminology for describing similarities should be formalized to avoid confusion. The suggestion is that the term homology be reserved for the purpose of ascribing evolutionary significance to a similarity and that in other circumstances similarity should be used. Although it remains to be seen if this idea will be adopted widely as a convention, it will be used here.

From a biological view point similar sequences may be similar because they share a common function or because they are of common origin. Often both of these will be true. Sequences that show similarities of the former kind are said to be related by convergent evolution whilst those that show similarities of the latter kind, but not the former, are said to be related by divergent evolution. For a particular similarity it will not always be possible to decide conclusively between these two models although, divergent evolution is thought to be much more common than convergent evolution [Gould, 1980].

The problem of detecting similarities is made difficult by the large range in the amount of similarity that is deemed significant according to circumstances. Sometimes the biologist may be looking for a perfect match to the sequence in which he is interested, that is to say he wishes to find other sequences which contain an exact, letter by letter, copy of his sequence. This

is a very easy problem. More often, he will be prepared to tolerate a degree of mismatch between his sequence and those that he considers as sufficiently similar to be of interest, that is to say, a proportion of the letters in the two sequences need not be the same. Such unmatched letters are generally referred to as mismatches. This too is an easy problem. The most usual situation and the problem that is most difficult to compute is the one where the biologist will also consider as similar to his sequence, sequences that both, have letters missing that are present in his sequence and also, have additional letters present that are not in his sequence. The letters that are missing from one sequence and are present in the other may be regarded either as insertions in the one sequence or as deletions from the other. To overcome this problem of nomenclature such letters are referred to as indels, a composite word derived by combining the first parts of insertion and deletion.

When devising methods for finding similarities between biological sequences it is important to realize that similarities that are biologically significant, that is to say that correspond to some functional or evolutionary relationship, may be less good, by whatever criterion is used to assess goodness, than ones that do not represent such a relationship and have occurred by chance. For this reason it is not satisfactory for a programme to return the single best alignment, rather it should return a number of alignments and allow the biologist to use his intuition to assess their biological significance.

It is almost always the case that an alignment chosen to represent a particular similarity is not unique. Whatever the scoring method used it is usually possible to rearrange the letters that comprise the aligned sequences to produce a different alignment of the same sequences with the same score. Thus, an alignment representing a particular similarity will be just one possible alignment chosen from many slightly different ones. Figure 2.1 demonstrates this for a highly simplified example. For similarities between biological sequences one particular alignment may well be preferred to the others because it is indicated by an observation or hypothesis. In the absence of such information it is usually considered best to make decisions that keep the alignment as compact as possible [Korn *et al.*, 1977], [Smith & Waterman, 1981].

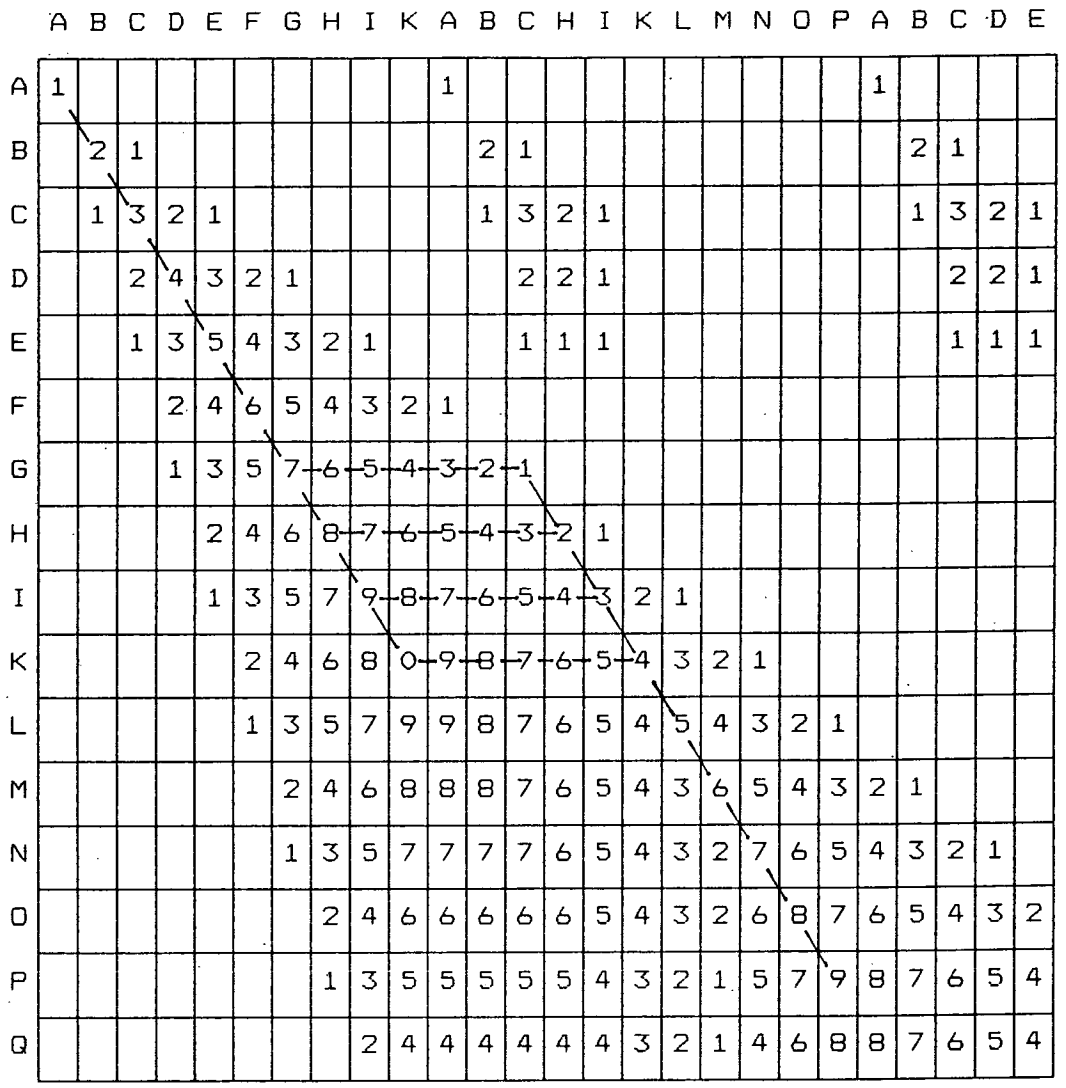


Figure 2.1

An alignment representing a similarity will usually be just one taken from a set of many equivalent ones. In the above highly simplified example there are four possible ways of aligning the side sequence against the top one corresponding to the four marked paths through the match matrix. Empty cells contain zero. The example uses the type three dynamic programming algorithm and the +1/-1 metric.

Most computer programming languages contain the necessary facilities for the representation and manipulation of the characters of the alphabet and it is normally most convenient to use these to represent and manipulate biological sequence data. DNA and RNA are represented by sequences of the letters A,C,G & T or U and proteins as sequences of the letters of the alphabet excluding B,J,O,U,X & Z. Other symbols are used to represent ambiguity. The conventions used in this project are those recommended by IUPAC-IUB Commission on Biochemical Nomenclature and are detailed in figure 2.2.

The first biological application of sequence comparison was for taxonomic purposes [Needleman & Wunsch, 1970]. A decade prior to this, it had been suggested that differences in the sequence of the same protein from different organisms might be used to establish taxonomic relationships [Crick, 1958]. In other words the differences would provide the basis for a protein taxonomy. When, in the late 1960s, sufficient protein sequence data became available to test this hypothesis it was found to be so. All proteins for which more than a few instances are known have been classified in this manner and are published in a compilation [Dayhoff *et al.*, 1978]. A particular attraction and indeed a unique feature of this method of taxonomy, is that it permits the estimation of resemblance between organisms that are exceedingly diverse [Sneath & Sokal, 1973]. For example, it is extremely difficult to imagine a characteristic that could be used to compare horse and yeast, yet the sequences of the enzyme phosphoglycerate kinase from these two organisms are identical in 274 out of 417 comparable positions; the expected number of identities for random amino acid sequences of this length would be about 29. Figure 5.1 in chapter five shows these two sequences aligned.

In the 1980s, as a result of developments in technology, large quantities of DNA sequence are now available and what was an interesting side issue has become an essential part of experimental molecular biology. It is now common practice for a molecular biologist to speculatively generate a piece of sequence and use a computer program which compares it with others in order to try to assign function by analogy. In addition many of the standard techniques of molecular biology are sequence specific; when using such a technique it is often necessary to use a computer program that can search for patterns in DNA sequence.

Code		Nucleotide	
G	=	Guanine	
A	=	Adenine	
T	=	Thymine	
C	=	Cytosine	
R	=	Purine	(A or G)
Y	=	Pyrimidine	(C or T or U)
U	=	Uracil	(only in RNA)
N	=	any	
X	=	any	

Code				Amino Acid
A	=	Ala	=	Alanine
B	=	Asx	=	Aspartic Acid or Asparagine
C	=	Cys	=	Cysteine
D	=	Asp	=	Aspartic Acid
E	=	Glu	=	Glutamic Acid
F	=	Phe	=	Phenylalanine
G	=	Gly	=	Glycine
H	=	His	=	Histidine
I	=	Ile	=	Isoleucine
K	=	Lys	=	Lysine
L	=	Leu	=	Leucine
M	=	Met	=	Methionine
N	=	Asn	=	Asparagine
P	=	Pro	=	Proline
Q	=	Gln	=	Glutamine
R	=	Arg	=	Arginine
S	=	Ser	=	Serine
T	=	Thr	=	Threonine
V	=	Val	=	Valine
W	=	Trp	=	Tryptophane
X	=	X	=	any amino acid
Y	=	Tyr	=	Tyrosine
Z	=	Glx	=	Glutamine or Glutamic Acid

Figure 2.2

The one-letter codes for nucleotides and the one-letter and three-letter codes for amino acids adopted by the Commission on Biochemical Nomenclature of the IUPAC-IUB [IUPAC-IUB, 1966] [IUPAC-IUB, 1968] and used in this project.

Algorithms for biological sequence comparison that directly manipulate the characters that represent individual sequence elements provide considerable flexibility with regard to the insertion of gaps between all the elements of both sequences in a simple manner. As this is deemed to be an important feature from a biological stand point these algorithms are regarded as preferred solutions.

In addition to such character-based algorithms there are algorithms that use small groups of letters (words) as the atomic objects. These word-based algorithms are still able to permit indels but in a much more limited way. They are used because they have much smaller requirements for computing power than the character-based algorithms. They are discussed further in chapter seven.

The character-based algorithms may be classified according to which of two major problem solving methods they use; rule-based or dynamic programming.

The Rule-Based Method

Algorithms that use the rule-based method work by taking each element in one sequence with each in the other and attempting to grow similarities from that position according a set of rules [Korn *et al.*, 1977]. Examples of the kind of rules that are used are:-

a similarity starts if :-

a pair of letters from each sequence match

a similarity propagates if :-

1) the next pair of letters match
2) the next pair does not match but the next two pairs of letters do
3) omitting a letter from the first sequence the next two pairs of letters match
4) omitting a letter from the second sequence the next two pairs of letters match

and so on.

A value for an alignment generated in this manner may be calculated

either by summing values ascribed to each rule as it is applied or by waiting until the similarity finishes and then counting the number of matches, mismatches and indels of which it is made up. As such programmes perform a great deal of backtracking the latter method is likely to require much less computation.

The Dynamic Programming Method

The Dynamic Programming method [Bellman, 1957] is a method of solving problems that have the property of consisting of very many similar sub-problems. Generally algorithms using this method will take advantage of this to reduce the amount of work needed to solve the problem. This might include such a strategy as constructing a table of partially computed results which are used to solve the final problem. Formally a problem that is to be solved by this method is posed as follows

> "To find an optimal path between two prescribed nodes in a graph with weighted edges."

The Dynamic Programming method is to observe [Michaelson, 1987]

> "...that for such a path, each stretch along it must be the solution of the same problem, but with the two end points of the stretch as the prescribed nodes. Then, if one takes a set of points which cuts the graph into two components, such that each component contains one of the end points, the optimal path must contain at least one point of the set. Each sub-path from that point to the two end-points must then be optimal. The point at which the path meets the cut-set can be found by finding the optimal paths from each point of the cut-set to the end-points and choosing the cut-point to optimize the sum of the weights of the sub-paths to the two original end-points."

The types of problem that can be effectively solved by this method are not well defined [Sedgewick, 1983]; there are many hard problems for which it does not seem to be suitable and many easy problems which are better solved by other methods. Problems for which dynamic programming is a good method are those which involve looking for a best way to do something when there are many ways of doing it; this is exactly the type of problem presented by biological sequence comparison.

The dynamic programming method applied to sequence comparison is based on the idea of a match-matrix. This is a two dimensional matrix which may be visualized by imagining the two sequences under comparison along

orthogonal edges of the matrix and allowing each of all the possible pair-wise comparisons of their elements to define a unique cell of the matrix. Values are calculated to fill these cells corresponding to alignments that end at the elements in the two sequences that define the cell. The values represent the relationship between the aligned sequences either in terms of similarity; the larger the score the greater the degree of similarity, or distance; the larger the score the more distant the relationship. In an optional second stage of the algorithm paths are traced back through the matrix to find the alignment(s) corresponding to the scores.

The distance, often called evolutionary distance, between two sequences is the sum of the total number of changes that are made to the one sequence in order to turn it into the other. The permitted changes are indels and substitutions. Substitutions correspond to the previously described mismatches. These changes have appropriate weights ascribed to them and the figure for the measure of distance is the sum of the weights for the minimum changes necessary to accomplish the inter-conversion. A value for the similarity between two sequences may be calculated in a similar manner except that instead of being neutral, exactly matched letters contribute a positive score to the value.

In order to guarantee the exhaustivity of the dynamic programming algorithms as formally described [Kruskal, 1983] the weights ascribed to these changes must comprise a metric, that is to say, they must have the following four properties.

1) Non-negative property. The distance between two sequences must be greater that or equal to zero.

2) Zero property. If two sequences are the same then the distance between them must be zero.

3) Symmetry. The distance between one sequence and another must be the same as the distance between the second sequence and the first and *vice versa*.

4) Triangle inequality. The sum of the distances between two sequences and a third sequence must be greater than or equal to the distance between the two sequences.

These four rules comprise the metric axioms and are summarized in figure 2.3.

```
Non-negative property.    d(a,b)>= 0
Zero property.            d(a,b) = 0 iff a = b
Symmetry.                 d(a,b) = d(b,a)
Triangular inequality.    d(a,b) + d(b,c) >= d(a,c)
```

Figure 2.3

The Metric Axioms.

Although similarity and distance have slightly different applications in biological sequence comparison they have been demonstrated to be equivalent under general conditions [Smith & Waterman, 1981]. The following description of the dynamic programming algorithm as used in biological sequence comparison is in terms of similarity.

The Algorithm to exhaustively generate all alignments between two sequences a and b proceeds as follows. Let the two sequences be a and b having elements a[i] and b[j] where a[1..i] and b[1..j] respectively represent the initial segments from a[1] to a[i] inclusive and b[1] to b[j] inclusive. The lengths of a and b which are the maximum values that i and j can take are represented by n and m. The values or weights for the metric are represented by w(x,y) which is the weight given to a substitution of x by y, w(x,ϕ) which is the weight given to the deletion of x and w(ϕ,y) which is the weight given to the insertion of y. The character ϕ is the null character and does not occur in the alphabet of which the sequences are comprised. In the case of biological sequences a space character is normally used. The algorithm calculates the distance d(a[1..i],b[1..j]) for successively larger values of i and j. These values may be considered as being calculated in an array like the one in figure 2.1. The value d(a[1..i],b[1..j]) chosen for a particular cell (i,j) is the minimum of the values of the formulae in figure 2.4. On the edge of the array, when i or j equal one, and access is required to values for i-1 and j-1 appropriate values must be "put in by hand". These values constitute parameters to the algorithm and affect its behaviour in important ways that will be discussed in chapter four.

```
d(a[1..i-1],b[1..j])   + w(a[i],φ)
d(a[1..i-1],b[1..j-1]) + w(a[i],b[j])
d(a[1..i]  ,b[1..j-1]) + w(φ,b[j])
```

Figure 2.4.

Possible values for d(a[1..i],b[1..j])

If two or more of the values in figure 2.4 are the same then a strategy for deciding between them must be devised. The simplest of these is to choose a diagonal step in preference to the others, because it leads to more compact alignments, and to make an arbitrary choice between vertical and horizontal steps. Other strategies are available that use additional information. These may well be appropriate to particular biological circumstances.

If it is necessary to generate alignments that represent the similarities as well as calculating values for them then a pointer must be recorded for each combination of i and j. The pointer will be chosen from the options in figure 2.5 according to the formulae chosen from those in figure 2.4. The alignments may then be generated by following the pointers through the match-matrix in an optional second stage.

```
                        (i-1,j)   or
pointer (i,j)   =       (i-1,j-1) or
                        (i,j-1)
```

Figure 2.5

Pointer equations.

As might be imagined for an algorithm that computes all the values in an (n x m) array the computational complexity is no worse than proportional to (n x m); this has been demonstrated formally [Wong & Chandra, 1976].

Advantages of the Dynamic Programming Method

When used for comparing biological sequences algorithms that use the dynamic programming method have a number of advantages over ones

that use the rule based method.

Global optimality

Dynamic programming algorithms are exhaustive, that is to say, they are guaranteed to find the best similarity (or several best similarities) out of all possible similarities. Because rule based algorithms only examine a subset of the possible similarities they cannot provide this guarantee. Although it is possible that the best of the rule based implementations will usually find the best similarity it is worrying to the scientist who obtains a negative answer not to be able to be sure that it is correct. Attempts to make the rule based implementations as exhaustive as possible contain so many rules that they are extremely slow [Brutlag, 1981].

Stable parameters.

In general the parameters supplied to an implementation of the dynamic programming method do not need to be changed between analyses of the same type with different sequences. The parameters are the weights applied to matches, mismatches and indels; in the case of mismatches the more unlikely, in evolutionary terms, a particular substitution is the more heavy the penalty for including it will be. The parameters for rule based algorithms affect the behaviour of the algorithm in rather unintuitive ways. Usually considerable tinkering is necessary to produce reasonable alignments for a particular pair of sequences.

Soft limiting parameters.

The parameters that penalize poor alignments in dynamic programming algorithms are applied progressively. For example for each successive mismatch added to an alignment a fixed penalty will be subtracted from the value of the alignment. In contrast, implementations of rule based algorithms use a sharp cutoff. For example, some implementations have a rule that says three consecutive mismatched elements are permitted but four are prohibited.

Evaluation system independent of algorithm.

Rule based algorithms are effectively evaluating on the fly according to the rules. Dynamic programming algorithms potentially find all possible alignments and then evaluate them afterwards. This means that it is possible to change the evaluation system without changing the algorithm.

Simple evaluation score.

The evaluation systems that can be used with the dynamic

programming method are biologically reasonable and provide a way of at least tentatively assessing alignments relative to each other. With the rule based method there is no analogous way of calculating an evaluation score and indeed it is often not clear to the user how one similarity is selected in preference to another.

Disadvantages of the Dynamic Programming Method

Although the dynamic method programming possesses these very great advantages when used for biological sequence comparison, in particular that of global optimality or exhaustiveness, it has not been widely used. The main reason for this is that, when searching entire databases, programmes written using this method have a very high requirement for computing power, to the extent that it is only really practicable to run them on very high performance computers which historically have not been made available to biologists. This factor on its own would not, however, seem to be sufficient to explain the neglect that this method has suffered, particularly when one considers the great advantage offered by its exhaustive nature. It is said [Kruskal, 1983] that this method is much less intuitive and more difficult to program than the rule based method and it may be that this too has been a contributory factor.

Although implementations of the dynamic programming methods are available that can align pairs of sequences, the question that all molecular biologists would like to ask as a matter of routine, namely "...is any part of my sequence like any part of any other sequence discovered so far?" is not answerable in a reasonable amount of time.

Attempts to overcome this problem have centred on the idea of reducing the size of the databases before using the dynamic programming algorithm. The apparently reasonable assumption made is that most of the sequences in the database will not show interesting similarities to the query sequence. An initial fast method is used to filter out the greater portion of the data base sequences leaving a subset to be processed by the more time consuming method. The fatal flaw in this reasoning is that the fast methods are much less discriminating in their ability to pick out alignments of biological interest. The hope of those that use them is that by casting their net sufficiently wide they will include sequences containing the interesting alignments that were not detectable in the initial phase. Unfortunately it is

not possible to demonstrate that this is the case.

The real solution to this problem is to use a computer that is powerful enough to run the exhaustive algorithm as a matter of routine. As well as answering a real need, the use of such a machine may well catalyse the development of new methods for the understanding of biological sequences.

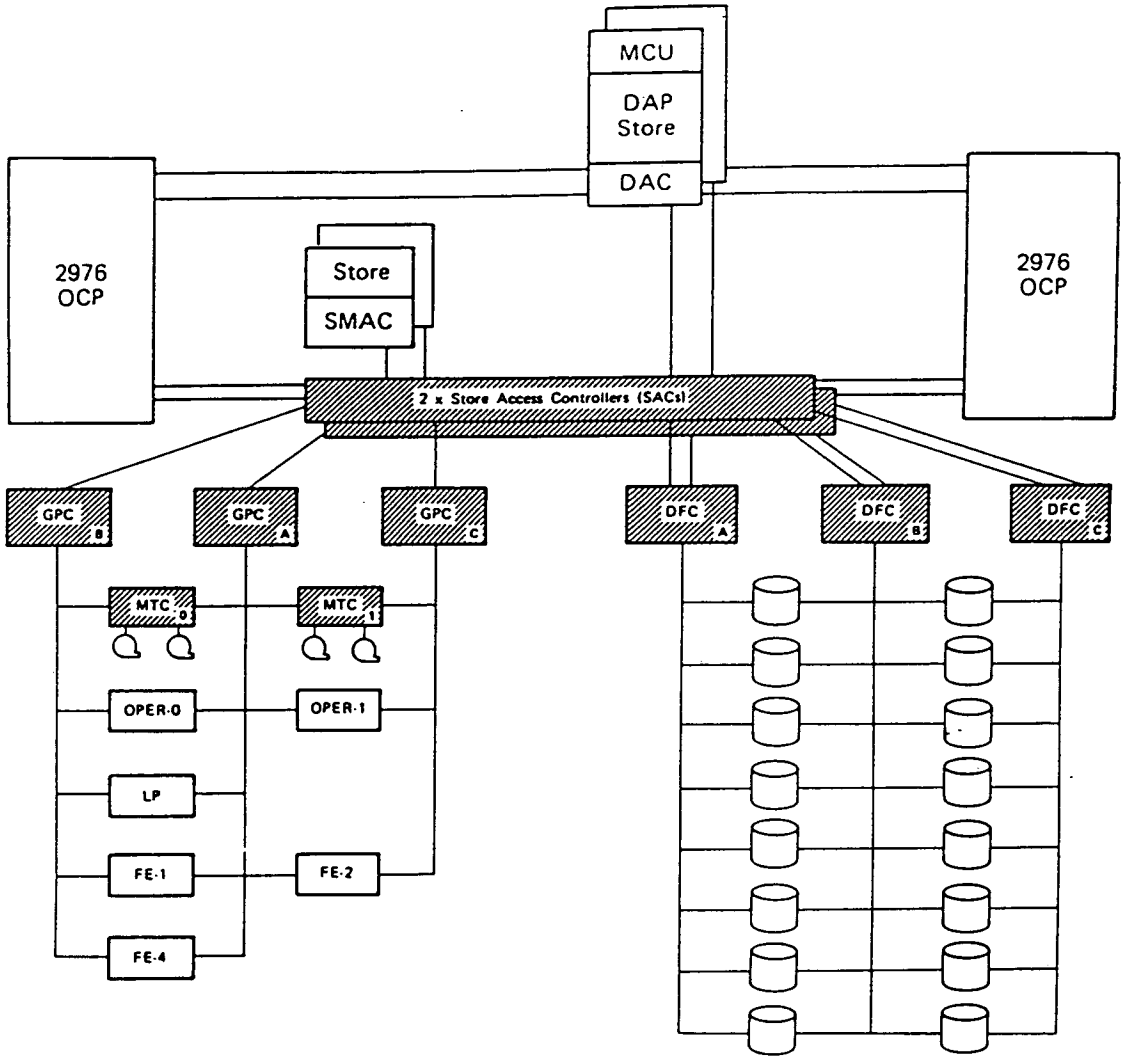
CHAPTER THREE

The Computer

A conventional, or serial, computer of the kind that is typically available to molecular biologists is not powerful enough to run exhaustive sequence comparison programmes that use the dynamic programming method. In order to achieve sufficiently high processing speeds for such programmes to be of use as a matter of routine it is necessary to use a parallel computer that can apply many processors to the same computation. The parallel computer used in this project is the International Computers Limited (ICL) 64 x 64 prototype Distributed Array Processor (DAP).

Serial computers derive their name from the fact that they perform their basic operations one at a time, that is serially. In fact when viewed at a low enough level even serial computers do certain things in parallel. For example, within a computer all the bits in a number will be operated on simultaneously, a time sharing system may well contain a small number of closely linked CPUs and the system as a whole will contain device controllers, direct memory access controllers and so on which allow processing and I/O to go on simultaneously. However, the fact remains that, the user's perception of the computer is that it is only capable of obeying one instruction at a time on one data item.

Because of inherent limitations in the technologies used to build computer hardware it was realized from early days that the serial architecture of computers imposed severe limits on the performance that they could achieve. For this reason designers and builders have sought to create computers with many processors that are able to process many data items simultaneously and communicate the results of the processing so that they co-operate in the solution of the same problem. Such computers have an architecture that is described as parallel and are often called super-computers.



Key:

OCP – Order Code Processor	GPC – General Peripheral Controller
DAP – Distributed Array Processor	MTC – Magnetic Tape Controller
DAC – DAP Access Controller	OPER – Operator Console
MCU – Master Control Unit	LP – Line Printer
SAC – Store Access Controller	FE – Front End (Communications Processor)
SMAC – Store Multiple Access Controller	
DFC – Disc File Controller	Mbyte – 1,048,576 bytes

(Shaded boxes represent control units)

Figure 3.1

A schematic representation of the main time-sharing computer system at Edinburgh University comprising two 2976 CPUs and two DAPs. At peak time it supported about 90 users. The operating system was the Edinburgh Multi Access System (EMAS). The system was decommissioned in June 1987.

Parallel computers are of two types. Single instruction stream multiple data stream (SIMD) and multiple instruction stream multiple data stream (MIMD). An SIMD machine performs the same instruction simultaneously on many different data items. An MIMD machine will be able to perform different instructions simultaneously on different data items. Both kinds of computers have been built but when this work began only SIMD machines had reached the market place.

Available SIMD computers are of two distinct types having very different architectures and properties. It is important to distinguish between them if any meaningful performance comparisons are to be made [Hockney, & Jesshope, 1981]. Vector processors, for example the CRAY-1, achieve high processing power by the use of a small number of extremely powerful processors made from expensive high-speed integrated circuits. The architecture of such machines is described as pipelined. Pipelining is a technique that implements parallelism by allowing several machine instructions to proceed simultaneously by overlapping the various atomic operations that comprise the instruction.

For example, a multiply instruction will require both its operands to be fetched from memory and placed in registers before the actual multiply is performed, subsequent to this the answer will have to be written out to memory. In situations where a large number of similar multiplications are to be performed a pipelined machine will have all these separate activities going simultaneously for many of the multiplications; a production line is probably a better analogy than a pipeline [Coulson, 1985].

Vector processors are intended to hide parallelism from the user and will run existing serial codes although advantage will be gained by considering the details of the architecture of the particular machine when writing programmes.

The true array processors or distributed array processors consist of an array of serial processors under the control of a master control processor and achieve high processing speeds through the use of a great many identical processors made from inexpensive integrated circuits. When programming such machines it is necessary for the programmer to be aware of the parallel nature of the architecture and to code the parallelism into programmes explicitly.

An important feature of the DAP is that the individual processors do not share memory. Each processor has its own private memory and cannot access the memory of the other processors. A number of other parallel computers, both SIMD and MIMD, have been designed so that the individual processors are able to access the same memory. Such machines have not been successful because of problems associated with memory contention.

Hardware

The DAP is an SIMD machine of the latter type. It is available as an additional store module for certain of the ICL 2900 series mainframes and is described by ICL as a store module which can process, in parallel, the data which it contains [ICL, 1982]. The DAPs used were a pair operated by the Edinburgh Regional Computing Centre (ERCC) and were part of a dual-processor ICL 2976 system running the Edinburgh Multi-Access System (EMAS). The DAP support software which was designed to run under ICL's own operating system VME/B was modified at the ERCC to run under EMAS [Stephens & Yarwood, 1986]. Figure 3.1 is a schematic diagram of the system at Edinburgh.

In 1987 the part of ICL that was responsible for the DAP was converted into a separate independent company called Active Memory Technologies Ltd. (AMT). AMT are now marketing a new generation of DAPs of various sizes, the smallest of which has 1024 processors. It seems likely that these machines will overcome many of the problems encountered when using the prototype DAPs.

The ICL DAP store module consists of a master control unit (MCU) and 4096 processors, or processing elements (PEs), arranged in a 64 x 64 square connected array. In addition there are connections which allow the processors to be used as a 4096 long-vector of processors. Each (PE) has three one bit registers, the Accumulator (Q), the Carry register (C) and the activity register (A), a one bit full adder and 4096 bits of local store. Taken together the local store of all the PEs makes up the 2 megabyte store module that is the DAP. Figure 3.2 emphasises the three dimensional nature of the DAP store.

The PEs are so connected, by row and column highways, that their

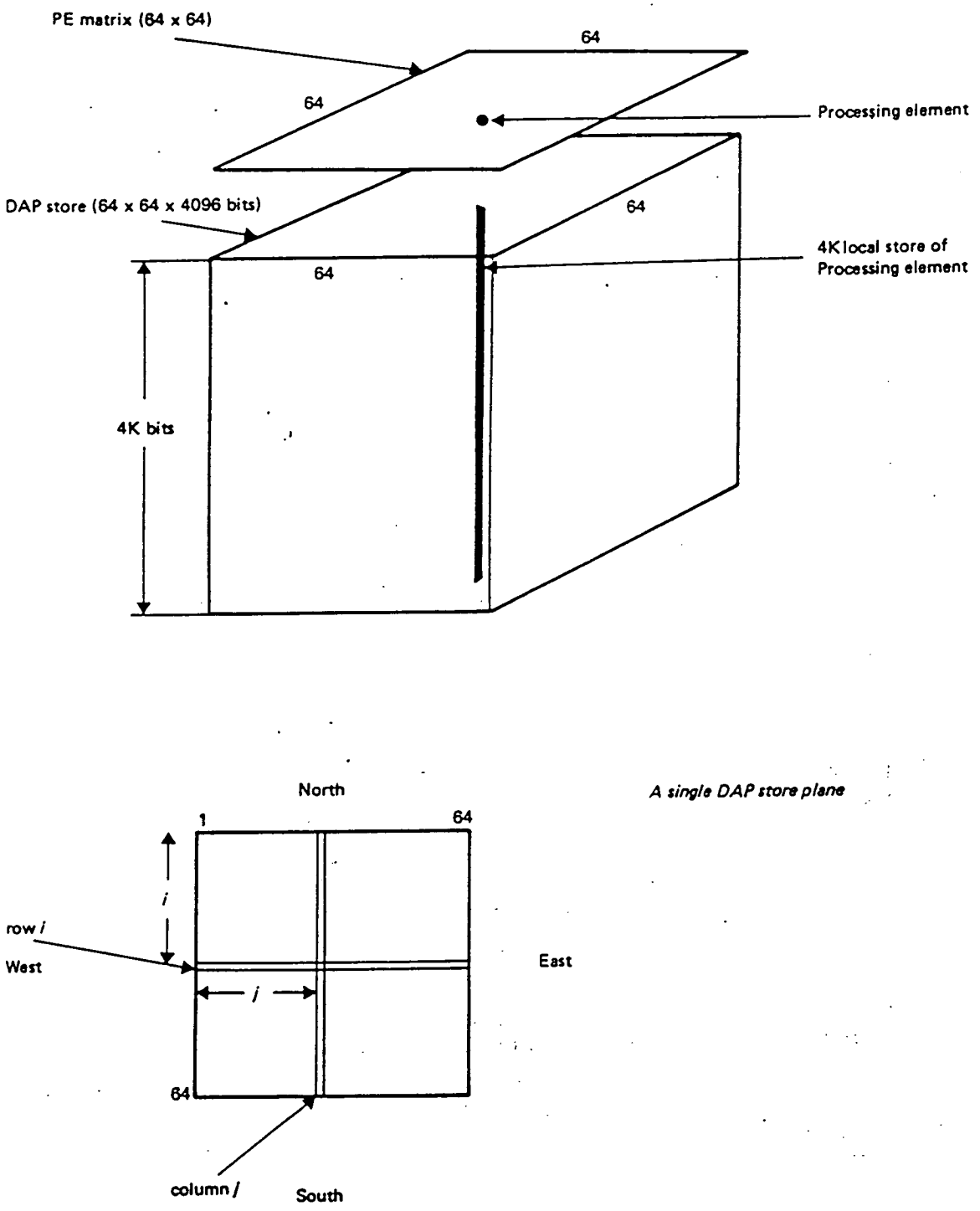


Figure 3.2

The DAP store module. Each of the 4096 PEs has 4096 bits of local store. Taken together this makes up the 2 Megabyte store module that is the DAP.

registers form register planes each containing 64 x 64 bits which may be shifted one bit per machine cycle to allow planar routing; with the plane geometry option in force bits shifted in at the edge are zero whilst with the cyclic geometry option they are the same as the bit shifted out on the opposite side.

The MCU is a powerful serial processor with 64 bit registers and various features for fast processing including an instruction buffer for fast execution of loops of up to 32 instructions. It broadcasts a stream of instructions to the PEs which optionally obey them according to the state of their A registers. It is this limited degree of local control possessed by the individual PEs that expands the parallel processing capability of the DAP into a useful form. Figure 3.3 is a schematic representation of the relationship between a single processing element and the DAP array.

Software

One of the most common criticisms of the DAP is that it is necessary for the DAP programmer to explicitly code the parallelism into his or her programmes. This is deemed undesirable because it greatly affects the way in which the programmer thinks. In fact the requirement is actually more stringent than this criticisms suggests; it is in fact essential that the programmer think very carefully about which parts of the algorithm may be executed in parallel and how this may be optimized for the DAP's hardware. For example, an existing serial programme transferred to the DAP will almost certainly run slower than it did on the serial machine. Even quite considerable changes to the code may well not produce an appreciable improvement in performance. In most cases it is necessary to return to the problem stage and design a new implementation, or sometimes even a new algorithm, bearing in mind the DAP architecture. This process is time consuming as it is necessary for the programmer to develop new programming techniques. This is not as unreasonable as it first sounds since many problems contain a very high degree of parallelism. Indeed, the algorithms that have been devised to solve such problems on serial machines are often extremely elegant methods for removing this innate parallelism in order to fit the problem onto a serial architecture. The very limited commercial success of the DAP may be in part due to this problem.

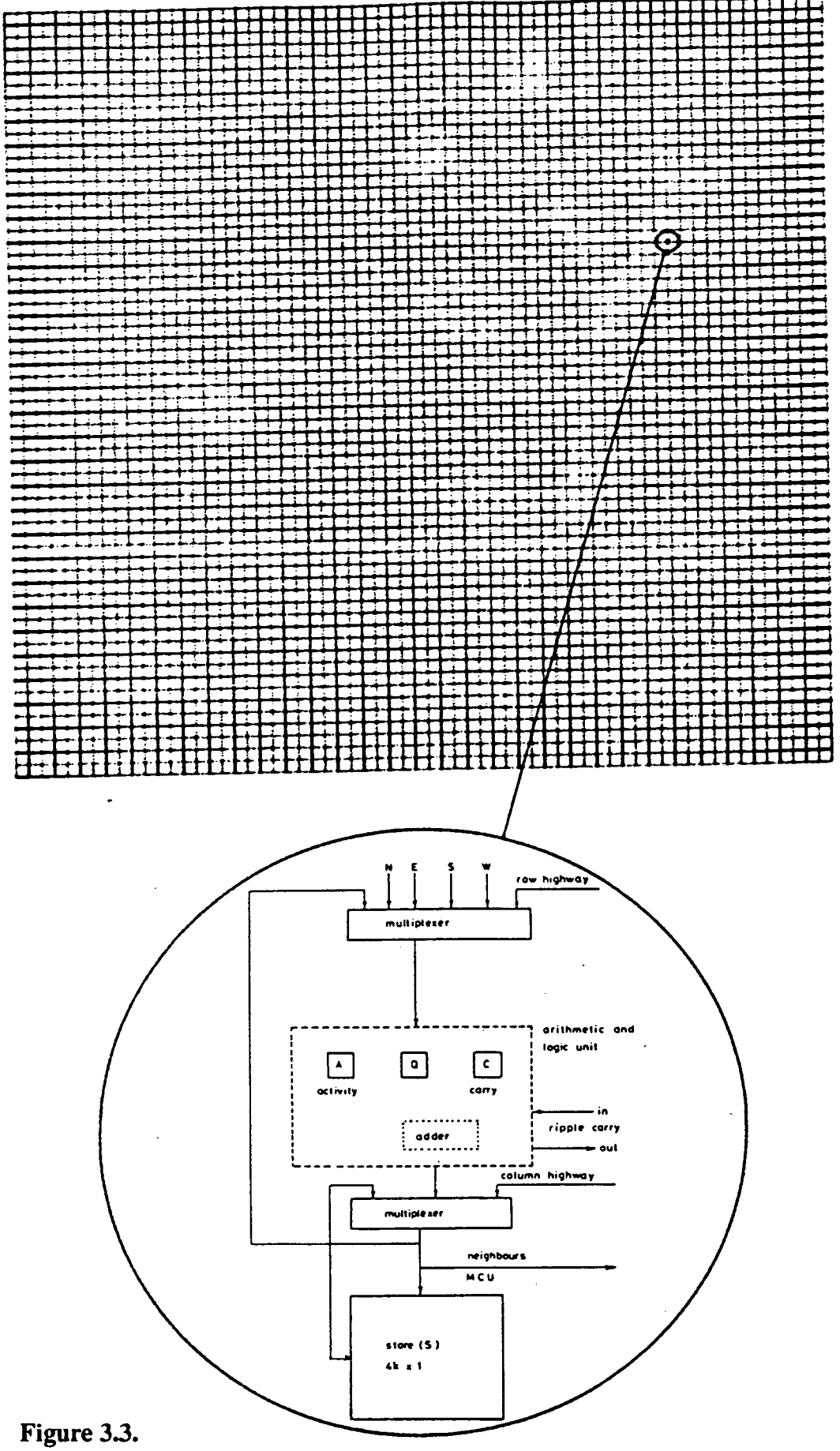


Figure 3.3.

Schematic representation of the DAP array. The DAP consists of 4096 individual processors or processing elements (PEs) arranged in a 64 x 64 square connected array. Each PE consists of a full one bit adder three registers (A,Q & C) and 4096 bits of local store.

The DAP support software comprises an assembler to assemble Array Processor Assembly Language (APAL), a compiler with runtime libraries for the high level language DAPFORTRAN and a linker (CONSOLIDATOR). All the support software runs on the host computer, the product of the development cycle being a two megabyte block of code and data which, in response to the DAPRUN command, is loaded into the block of physical memory that is the DAP before control is passed to the MCU. At the end of the programme, or if an exception occurs, control is passed back to the host. During the interval between these two events, no interaction is possible with either the programme or the data (with a limited exception discussed later). This not only precludes the use of interaction in DAP programmes, but also makes debugging extremely difficult and time-consuming; to quote from the ERCC DAP FORTRAN Summary Guide, the section entitled "Error Messages etc" [Blair-Fish, 1986],

> "...the best thing is not to make mistakes [in DAP programmes] because there is no direct way of printing intermediate numbers out from the DAP in the middle of a calculation..."

The compiler, assembler and linker all use considerable amounts of host processor time. This means that DAP programme development has a significant impact on system performance and is precluded at peak times. Similarly, execution of DAP programmes has a great impact on the rest of the system. When the DAP is not executing a programme the DAP store is used as ordinary ICL 2900 store. The host operating system expands to fill the DAP store so that when the DAPRUN command is issued a long period elapses as the operating system clears the two megabytes of physical memory that is the DAP before it can load the DAP programme block. On the ERCC dual 2900 system with about 80 users and the DAP in use as ordinary store module the time that elapses between issuing the DAPRUN command and programme execution commencing is about two minutes.

These defects in the software, in particular the lack of support for the programme development cycle, have been recognized by the manufacturer and it is hoped that future DAPs will show great improvements in this area. Indeed, recent sales literature [AMT, 1987] cites ease of programme development as a key feature of the latest generation of DAPs.

Array Processor Assembly Language (APAL).

APAL was not used much in this project as DAPFORTRAN is very flexible and efficient and there are large libraries of subroutines available. The major exception to this was the use of APAL to increase the efficiency of store mode conversion. This is discussed under the section on "musical bits".

DAPFORTRAN.

The parallelism of the DAP hardware is available to the DAPFORTRAN programmer through the concept, analogous to that of type, of mode. All variables have mode, either explicitly as a result of declaration or, through an extension of the rather unfortunate implicit mechanism of FORTRAN. The permitted modes are scalar, vector and matrix. Variables of these modes are assumed to have one, sixty four and four thousand and ninety six elements respectively. There is no restriction on the combination of mode and type. Scalar variables are stored horizontally under the PEs one per row. All the bits of a scalar variable are processed simultaneously. Vector variables are stored horizontally one per layer. All the bits of all the 64 elements of a vector variable are processed simultaneously. Matrix variables are stored vertically. All 4096 elements of a matrix variable are processed simultaneously one bit at a time. Scalar and vector variables of length less that sixty four bits are stored one per row; the remainder of store in the row is not used. A set of store mode conversion subroutines are provided to convert variables between the various modes.

The DAPFORTRAN compiler was written without the use of the hardware scalar matrix comparison instruction. This means that such comparisons take longer than the theoretical minimum time. All of the database searching programmes developed in this project do a very great deal of scalar matrix comparison and their runtimes will have been adversely affected by this problem.

DAPFORTRAN provides a library of functions which, in the style of FORTRAN, need not be declared and have specific and generic versions.

Componential Functions.

These are analogous to the standard FORTRAN functions providing such things as type and length conversion and various arithmetic and trigonometric functions with the difference that the arguments may be of any

mode and the operation is performed on all the components of the argument. An example of a componential function is SQRT which finds the square root of all the components of its argument. When SQRT is applied to a scalar, vector or matrix argument it will simultaneously process 1, 64 and 4096 elements respectively.

Aggregate functions.

The aggregate functions provide a great deal of the parallel processing functionality of DAPFORTRAN in a high level manner. A proper understanding of them is essential if efficient DAP programmes are to be written. By using them carefully it is also possible to write understandable DAP programmes [Flanders, 1982].

Mode conversion and expansion.

Matrix data are stored vertically, each value in the column below its processor, vector and scalar data are stored horizontally in a plane of data. In addition, there is a fourth mode of storage corresponding to the way in which data is stored in the host computer. A set of store mode conversion routines exist to convert data between these modes. An example of such a function that also expands its argument is MAT; it takes a single scalar argument of any type and returns a matrix of the same type. MAT was used extensively in this project; every time a letter from one sequence was compared with sequence from a database it was first expanded using MAT. The DAPFORTRAN compiler permits implicit use of MAT, that is to say the reserved word can be omitted and the conversion will still take place. This kind of feature is to be deprecated as it can lead to programme code that is difficult to understand.

Componential reordering.

These functions alter the ordering of the components of their vector or matrix argument. For example REV reverses the ordering of its vector or long vector argument.

MERGE.

This function merges two vectors or matrices using a logical mask. The three arguments are of the same mode, the third is of type logical. The function returns a value of the same mode as the first two arguments, each component of which is taken from the corresponding component of the first or second argument according to whether the corresponding component of

the third argument is .TRUE. or .FALSE..

Arithmetic reduction.

These functions perform an arithmetic operation on all the components of a vector or matrix. They are referred to as reductions because they reduce the mode of their arguments. For example SUM takes a matrix argument and returns the scalar sum of its components.

Logical reduction.

These functions perform a mode reducing logical operation on all the components of a vector or matrix. For example ALL and ANY return the scalar logical AND and OR respectively of all the components of their non-scalar argument.

Maximum and minimum values and positions.

MAXV and MINV return the scalar value corresponding to the maximum and minimum values respectively of their non-scalar real or integer argument. Similarly MAXP and MINP return a logical mask of the same mode as the argument whose components are set in the positions corresponding to the maximum and minimum values respectively.

FRST.

This function returns a logical value of the same mode as its logical vector or matrix argument with all its components false except for the one in the position corresponding to the first .TRUE. component of the argument.

Logical Integer transformations.

These functions return integer indices of logical arguments; for example ELN returns an integer scalar that is the index of the first .TRUE. component of its logical non-scalar argument.

Pattern generation.

These functions set the components of non-scalar variables to various patterns. For example ALTV sets alternating components of vectors and long-vectors to .TRUE. and .FALSE..

Shifting.

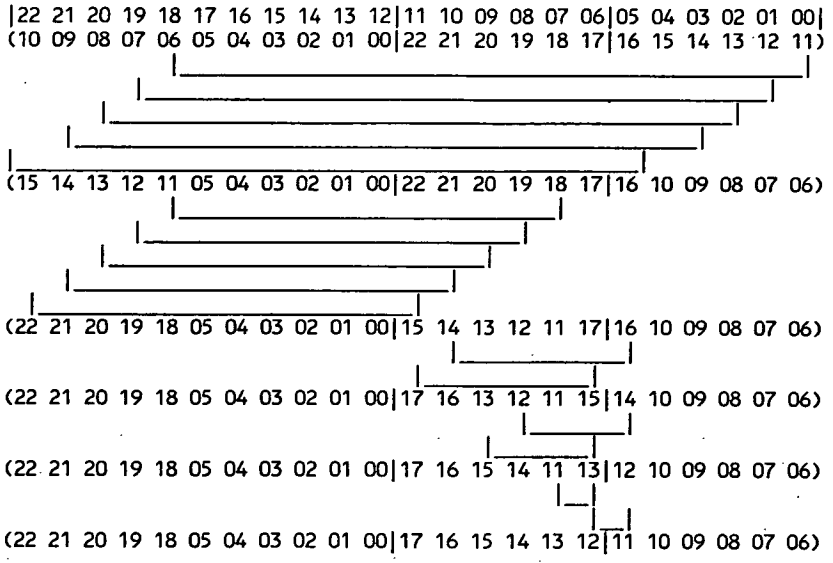
These functions perform shifts on their matrix or vector first argument by the amount specified in the second integer scalar argument. A matrix may be shifted north, south, east or west. A vector or long-vector may be shifted right or left. Geometry may be either planar or cyclic. For example SHRP shifts a long-vector rightwards with planar geometry.

Musical Bits.

Much of the effort involved in writing DAP programmes goes into designing methods of data movement to minimize the time spent in bringing elements of a data structure that need to interact close enough together within the DAP store to do so. For many applications where such movements are programmed in an *ad hoc* and inefficient manner optimal performance is not achieved. The musical bits routines, named after the children's game musical chairs [Flanders, 1980], provide a generalized method for simply programming complex data movements within all three dimensions of the DAP store. As they are written in APAL they are extremely fast. It is unfortunate that they were not available when the DAPFORTRAN compiler was written as the store mode conversion routines described above would have been much faster if they had been written using the musical bits routines.

The user interface to the musical bits routines uses the concept of a mapping vector. This is a one dimensional entity which describes how the data is mapped onto the three dimensions of the DAP store. The user designs a mapping vector to describe the original data mapping and then specifies a series of simple operations which transform it into the mapping vector corresponding to the new data mapping. The only operation available at the time of writing was the exchange, which as its name suggests exchanges bits within the mapping vector. The routine to perform the exchanges is simply called 'X' and its two arguments are the numbers of the bits within the mapping vector that are to be exchanged.

To write a subroutine to perform a particular store mode conversion it is necessary to design two mapping vectors corresponding to the arrangements of data in store before and after the conversion. These vectors are then used to create a simple APAL routine which calls the appropriate routines. The musical bits documentation includes the necessary timing information to enable the programmer to choose the most efficient way of performing the conversion. Figure 3.4 illustrates the process for the routine that was written for the sort described in Chapter 7. In this example the data have been left crinkled 32 deep after the sort and the purpose of the routine is to convert them into normal DAP mode.




```
! Uncrinkles a LV of 8 byte objects crinkled 32 deep.
! Andrew Lyall 22 June 1985
CODE U8   DAP
ENTRY_POINT
PLANES 2048
X 0,18
X 1,19
X 2,20
X 3,21
X 4,22
!
X 7,18
X 8,19
X 9,20
X 10,21
X 11,22
!
X 5,10
X 6,11
X 5,8
X 6,9
X 6,7
X 5,6
EXIT_POINT
FINISH ENDMODULE
```

Figure 3.4

The Musical Bits Routines provide a generalized method for simply programming complex data movements within all three dimensions of the DAP store. (upper) This example details the exchanges in the mapping vector necessary to convert 8 byte data crinkled 32 deep into normal DAP mode. The line at the top contains the number of the bit in the address of the data items as used by the Musical Bits Routines. The most significant bit is 00 and the least significant 22. The vertical bars divide the address bits according to the dimensions of the DAP store. (lower) The APAL code to perform the necessary exchanges. This routine was used to uncrinkle data produced by the vertical mode sorting routine described in chapter seven.

DAPSU libraries.

The Dap Support Unit at Queen Mary College London writes and maintains a library of subroutines written in DAPFORTRAN and APAL. Most of these are aimed at engineering applications and thus have not proved useful in this context.

Block Transfer System.

BTS was developed by ICL in response to demand from users. It was intended to simulate an interactive capability from within DAPFORTRAN and thus overcome what is a major limitation of the prototype DAP; the lack of any interactive facility. Unfortunately the prototype DAP project was discontinued before BTS reached the market place.

In response to user requests to provide a "BTS-like" facility, the ERCC developed two products. Dap Data Exchange (DDE) and DDE extension (DDX) [Brown, 1986]. Although these products did not provide an interactive facility, they could only be used in batch mode, they did permit exchange of data during a DAPRUN without completely unloading the DAP. Unfortunately all the data movements had to be specified before running the programme; this made it impossible to use them in situations where the data movements are only known at run time, such as external sorting. Furthermore even in cases when the data movements were known prior to run time other restrictions made them very awkward to use; DDE and DDX were thus not used extensively.

Suitability of the DAP for Sequence Comparison

The DAP possesses a number of features which make it particularly suitable for analysing biological sequence data. The most obvious of these and the original motivation for the project is the compatibility between the basic connectivity of the machine and sequence data. The DAP also has a major advantage over vector architectures which emphasise performance in terms of high precision arithmetic. The very great flexibility of the DAPs data representation with a full trade-off against speed and storage is of great advantage when, as is the case for sequences comparison, the data types require only a small number of bits. On a vector machine there is no advantage to be gained by using low precision variables to store the sequence data whereas on the DAP this provides very great advantage. At the worst,

sequence data can be stored as single bytes with even greater advantage obtained using fewer bits; proteins can be represented with 5 bit objects and nucleic acids with 3 or even 2 bits if it is not necessary to represent ambiguity. Similarly, such logical variables can be used very efficiently to store and process relationships between sequences. The very high speed of processing such logical variables allows DAP performance to equal or even exceed that of vector processors costing ten times as much on these types of problem [Collins & Coulson, 1987].

Additionally as a highly parallel SIMD machine the DAP is ideally suited to problems that can be broken down into a large number of identical sub-problems with different data; this property is the salient feature of the dynamic programming method which is used extensively in this project.

CHAPTER FOUR

The Algorithms

Biological sequence comparison problems fall naturally into three distinct classes which may be tackled by three slightly different versions of the dynamic programming algorithm described in chapter two. The three versions differ only in the metrics that they use and the values that are fed in to the edges of the array in the circumstances, described in chapter two, when i or j equal one. When both i and j equal one a value of zero is always used.

The Type One Problem

When a pair of sequences, almost always proteins, are suspected of having a common ancestry it is often desirable to align them along their entire length so as to emphasise that ancestry and to provide a numerical estimate of the evolutionary distance between them. An example of the use of such values is for the construction of phylogenic trees.

In order to force the sequences to align co-linearly, overhang on either sequence is penalized as if the overhanging residues were indels. It is this feature that defines the algorithm for the type one problem and it is achieved by moving values into the match-matrix at the edges corresponding to the row or column number minus one, multiplied by the indel penalty.

The Type Two Problem

It is often necessary to search the databases for all instances of similarity to a short oligo-nucleotide or peptide sequence. The sequence in question may be a signal or control sequence or a consensus sequence, typically less than fifty elements long. In this case it is desirable to penalize unmatched overhangs on the query sequence as indels, in order to try to force alignment along its entire length, but meaningless to do so for the database sequences. This is achieved for the query sequence as for the type one problem. In the case of the database sequences overhangs are neutral in

the affect that they have on the alignments. This is achieved by always moving a zero into the matrix along the edge defined by the database sequences.

The Type Three Problem

The type three problem is to answer the question "...is there any known sequence that has any regions that are similar to any of the regions in the query sequence?"

The algorithm to do this requires the most computation and is the most useful. It is described as the "best local similarity algorithm" [Smith & Waterman, 1981]. It is used to ascribe function and evolutionary relationships, by analogy, to sequences of unknown function and origin.

In this case, as it is regions within each sequence that are of interest, overhangs in both sequences are treated neutrally; zeros are moved in at the edge of the matrix in all locations. This, however, is not enough on its own to ensure that the algorithm will find local similarities. The problem is that any pair of sequences may be considered to align albeit trivially. However it is only regions that produce a good positive score when they are aligned that are of interest. Since the value for such a region will be reported relative to the value of the cell in which it starts, it will not be possible to determine the goodness of an alignment by its value, or to separate the good portions of the alignment from the bad portions.

In the published description of the algorithm [Sellers, 1974] this problem is overcome by never letting the value in a cell drop below zero; immediately a value becomes negative it is reset to zero and the alignment is terminated. Under these circumstances local similarities may be detected and ordered according to their absolute score.

Implementations

When programming conventional serial computers, the arrangement of data in store can often be the most critical performance issue [DEC, 1987]. If this is so when the store only has a single dimension, as is the case with virtually all serial computers, then it becomes the over-riding consideration when programming the DAP which has a multi-dimensional store; if the square connectivity of the DAP is used then the store has three dimensions

(64 x 64 x 4096 bits), if the linear connectivity is used then it has two dimensions (4096 x 4096 bits).

The critical implementation decision then is on mapping the data onto the store so as to allow as many as possible of the processors to do useful work for as much of the time as is possible. Often this will mean that a programme performs more than the theoretical minimum number of operations in order to increase the opportunities for parallel processing and thus reduce the total run time. A classic example of this is the bitonic sort [Batcher, 1968] described in chapter seven.

A more subtle problem involves balancing the relative costs of processing and data movement, referred to in the DAP literature as "routing" [Reddaway, 1984]. Although the DAP possesses row and column highways in order to increase the efficiency of routing, time spent routing can still dominate the run time of certain programmes.

The aim, when balancing processing and routing, is to arrange the data so that PEs that need to exchange information with each other are as close together as possible. Once again it may be advantageous to perform extra processing in order to reduce the over all runtime by reducing the amount of routing. It is likely that future DAPs will have very much faster routing [Reddaway, 1984] which will make this issue less dominant.

Type One

The type one problem is different from the other two problems in that a large database is not being searched. Usually only pairs or small numbers of sequences are compared. This type of analysis is not normally applied to nucleic acid sequence data.

Typical proteins range in size between about one hundred and about four hundred amino acids in length. This is inconveniently distant from the two major dimensions of the DAP; 64 if the square connectivity of the array is to be utilized and 4096 if the linear connectivity is used. This means that the two most obvious ways of mapping the problem onto the array present problems.

For sequences under 64 in length an extremely fast implementation which maps the match-matrix directly onto the array is possible, however, the

special programming necessary for the edges of the array, were the programme extended to cope with sequences of length greater than 64, is not only difficult but also likely to badly effect efficiency [Collins & Coulson, 1984]. The opposite problem arises with the long vector mapping; that if only single pairs of sequences are compared then, for most of the time at least 90% of the processors will not be working productively. In the case where the user wishes to make a number of comparisons then a single sequence may be compared with about 10 or so others simultaneously using the long vector mapping.

If optimum performance were a critical issue it would be necessary to write a number of different implementations for various different lengths of sequence. As this is not the case a reasonable compromise is to use the long vector mapping and put up with the fact that a single pair-wise comparison will take the same amount of time as a comparison of a single sequence with about ten others.

As it is distance that is of interest this implementation used a difference metric. This causes the best alignment to finish in the cell in the bottom row of the match-matrix containing the lowest value. There is sufficient room in the prototype DAP to generate the whole matrix and step back through it to generate the actual alignment.

Type Two

The type two problem involves comparing a short query sequence with a large database, ideally containing all known sequences. The major decision is that of how to map the database onto the array.

The option of using the square connectivity and taking letters in parallel from both the query and database sequences was rejected for the following reasons. A great deal of time would be spent on the special processing necessary for dealing with the cases where the database sequences run off the edge of the array. Also, since it is likely that the query sequence will usually be substantially less than 64 letters long a great many of the PEs will not be used for much of the time.

The mapping chosen used the linear connectivity. The database was taken 4096 letters at a time and compared in parallel with a single letter from the query sequence. This allows all 4096 processors to be used all the time

and minimises the amount of special processing necessary for dealing with the edges. A distance metric was used. This causes paths of interest to finish in cells corresponding to minima along the bottom line of the matrix.

The values for the cells are only generated a line at a time as they are only needed in the final row to record the maximum score achieved by each path. The whole matrix is recorded but each cell only records how it was arrived at. As this need only assume four different values (match, mismatch, vertical step & horizontal step) it can be stored in a two-bit variable which is very space efficient although slightly fiddly when it comes to printing the alignments out. The two-bit values within the layers of the matrix corresponding to steps back along the paths are left-packed within the array (an extremely efficient process) to bring all the steps in each particular path under the same PE.

A problem arose here because the paths are not all necessarily of the same length. Their length depends on the number of horizontal steps in them. As it is not possible to use the parallelism of the DAP effectively on data items that are of varying length it was necessary to ensure that the objects representing the paths were of the same length. The solution was to record all horizontal moves as a single horizontal step regardless of the number of step actually involved. This can be coped with by the programme that prints the alignments because this particular implementation of the dynamic programming algorithm always ends a series of horizontal steps at the first possible match. This feature is a side effect of the strategy for selecting between steps of equivalent value that is designed to keep the alignments as compact as possible.

Finally, now that each path is under its own PE, the paths may be sorted into order of goodness according to their maximum score and returned to the host computer where a simple programme decodes the packed paths and prints out the alignments. Usually many more paths are collected than the user requires to see. Ideally, the output programme would be made interactive so that the user could browse through the large number of alignments varying the selection criteria before deciding which ones to print.

Type Three

As described [Sellers, 1974] the algorithm for this problem is not adequate in two respects. The first is that it only returns the single best alignment between the pair of sequences under comparison. Unfortunately the regions of the sequences that align the best may do so fortuitously and not because of an evolutionary or functional relationship. It may be the case that the alignment to which biological significance or functionality is ascribed is not the best alignment that can be generated between the two sequences.

What is required is an algorithm that provides the best n alignments where n is a biologically reasonable number. These n alignments may be then examined by eye in order to ascribe biological significance.

The second problem is a pragmatic one. As described the algorithm requires an amount of computer memory proportional to the product of the lengths of the two sequences under comparison. This is not possible with the prototype DAP because of the lack of I/O and the small memory size.

Interestingly enough, it turned out that the solutions to these two problems were linked. The reason that the algorithm, as described, does not return all possible paths is not that they are not computed, rather, it is that they are computed, but in such a way that it is not possible to detect then easily. The maximum scores of the paths are not directly available, they are scattered throughout the matrix. The best alignment, and its score are collected in the optional second stage of trace-back described in chapter two.

Rather than generate the whole match-matrix within the DAP only a band of the matrix two cells wide is generated. This band is computed to sweep down the matrix, each value being generated only once during the computation. In addition to the scores other values are maintained and moved with the appropriate cell in the band. These additional values include such things as the start and stop points within the two sequences, the maximum score, the current score and other housekeeping information. These values are carried forward with the band sweeping down the match-matrix so that at the end of the sweep all the information necessary to reconstruct the alignments is available.

This reconstruction of the alignments from the details of the paths through the match-matrix was performed using a serial programme as it was

not computationally intensive enough to justify using the prototype DAP. Although not essential to this reconstruction, the maximum deviation the alignment makes from the main diagonal was collected by the DAP programme as this speeds up the serial calculation enormously. It should be noted that this is only a temporary measure as the task performed by the serial programme is likely to remain constant whilst the task performed by the DAP programme will increase in magnitude rapidly. In the near future it is likely that it will be necessary to modify the DAP programme to increase its efficiency. When this happens relatively expensive and unnecessary features like the one just described will have to go.

The programme maintains a list of the best four thousand or so alignments irrespective of which sequence in the database they are derived from. These results may be sorted according to a variety of criteria that are user selectable. Ideally, as with the type two implementation, these results should be presented to the user through the medium of an interactive browsing programme.

CHAPTER FIVE

Similarity

The description of the dynamic programming algorithm in chapter two does not enumerate the values (weights) used to penalize indels and mismatches or to reward matches, that is to say the values that comprise the metric. Early versions of the programmes written in this project used a metric that scored +1 and -1 for rewards and penalties respectively. It is not, however, necessary to be restricted to such a simple scheme since the algorithm will work with any set of values that conform to the metric axioms.

The alignments of protein sequences generated using the +1/-1 metric are very short and consist almost entirely of perfect matches. This renders them uninterpretable unless they are from comparisons between very similar sequences. This is unsatisfactory because such similarities can be detected using the inexhaustive (or heuristic) and thus very fast, algorithms. It is however apparent why this should be the case.

As proteins contain twenty different amino acids, perfect matches between dissimilar sequences will occur at low frequencies; within the vicinity of 1 in 400. Using the +1/-1 metric, where the score for an alignment is penalized by the same amount for a single mismatch as it is credited for a single match, it is thus impossible to generate alignments between dissimilar sequences. This is because the formal description of the algorithm [Sellers, 1974] requires that when the value for an alignment drops below zero it is deemed to have finished and all the various counters are reset. The obvious solution to this, namely, to change the metric so that this is not the case, does not provide a satisfactory solution. A number of different ratios of reward to penalty were tried and values in the region of twenty seemed to be the most satisfactory. Although this produced a large number of much longer alignments it was still difficult to interpret them. Using test data it was found that the interpretable alignments were masked by large number of

uninterpretable ones that appeared spurious. Fortunately there is a biological rationale for why this should be so. An understanding of this makes it possible to devise methods that favour interpretable alignments over spurious ones.

It has been known for some time that two proteins from different sources that perform the same function can show considerable differences in their sequences whilst having very similar structures. Figure 1.8 in chapter one shows the three dimensional structure of the glycolytic enzyme phosphoglycerate kinase (PGK). This was generated from X-ray crystallographic data and is displayed on a computer graphics system. The structure in blue is the enzyme from horse whilst the one in yellow is from yeast. The white lines are where the structures overlap. All the side chains, except for that of an Arginine (168 in the yeast structure and 170 in the horse), have been omitted for clarity. It can been seen that the structures are extremely similar, and indeed this is not surprising as these two proteins have virtually identical catalytic, kinetic and physical properties.

Figure 5.1 shows the sequences from these two enzymes aligned. From this it can be seen that only 272 of the 417 amino acids of yeast PGK correspond exactly to those in horse PGK; these two proteins show virtually identical properties and structure when 35% of their amino acids are not the same.

Conversely it has also been known for a long time that changes of just one of the several hundred amino-acids in a protein can result in extremely big changes to its properties. Considerable circumstantial evidence to support this belief has been available in the form of lethal point mutations. The hypothesis explaining such mutations is that they introduce a single amino-acid change into a protein which is thereby so badly affected that the organism cannot survive.

More recently direct evidence has been provided by the technique of site directed mutagenesis (SDM) [Winter & Fersht, 1984]. Using this technique a genetically-engineered PGK has been prepared from the yeast enzyme with Arginine 168 changed to a Methionine. Arginine 168 can be seen as the single side-chain projecting into the inter-domain cleft of the PGK molecule in figure 1.8. This change, of just one amino acid, resulted in a mutant enzyme whose catalytic activity was reduced one hundred fold

```
         10         20         30         40         50
*** **    ** * ***  *******     ** **** ** * **  *
SLSNKLTLDKLDVKGKRVVMRVDFNVPMKNNQITNNQRIKAAVPSIKFCL
SLSSKLSVQDLDLKDKRVFIRVDFNVPLDGKKITSNQRIVAALPTIKYVL

         60         70         80         90        100
      *** ****** *     **** *** ** ******* ** ****
DNGAKSVVLMSHLGRPDGVPMPDKYSLEPVAVELKSLLGKDVLFLKDCVG
EHHPRYVVLASHLGRPNG.ERNEKYSLAPVAKELQSLLGKDVTFLNDCVG

        110        120        130        140        150
**** *    * ********** ******      * ****       **
PEVEKACANPAAGSVILLENLRFHVEEEGKGKDASGNKVKAEPAKIEAFR
PEVEAAVKASAPGSVILLENLRYHIEEEG.SRKVDGQKVKASKEDVQKFR

        160        170        180        190        200
  ** * *** ***************  *** * ***  *** ** ***
ASLSKLGDVYVNDAFGTAHRAHSSMVGVNLPQKAGGFLMKKELNYFAKAL
HELSSLADVYINDAFGTAHRAHSSMVGFDLPQRAAGFLLEKELKYFGKAL
                         ^
        210        220        230        240        250
* * ******************* * ****   ********** *****
ESPERPFLAILGGAKVADKIQLINNMLDKVNEMIIGGGMAFTFLKVLNNM
ENPTRPFLAILGGAKVADKIQLIDNLLDKVDSIIIGGGMAFTFKKVLENT

        260        270        280        290        300
*** * **  ** **  ** **   **   *****  ** *   * *
EIGTSLFDEEGAKIVKDLMSKAEKNGVKITLPVDFVTADKFDENAKTGQA
EIGDSIFDKAGAEIVPKLMEKAKAKGVEVVLPVDFIIADAFSADANTKTV

        310        320        330        340        350
*   ****** *** **** *  *  *  ** ****** ****** ** *
TVASGIPAGWMGLDCGPESSKKYAEAVTRAKQIVWNGPVGVFEWEAFARG
TDKEGIPAGWQGLDNGPESRKLFAATVAKAKTIVWNGPPGVFEFEKFAAG

        360        370        380        390        400
**** *****    *   *********   *    ** ************
TKALMDEVVKATSRGCITIIGGGDTATCCAKWNTEDKVSHVSTGGGASLE
TKALLDEVVKSSAAGNTVIIGGGDTATVAKKYGVTDKISHVSTGGGASLE

        410
***** ****  **
LLEGKVLPGVDALSNIL
LLEGKELPGVAFLSEKK
```

Figure 5.1

The amino acid sequences of horse (upper) and yeast (lower) phosphoglycerate kinase aligned along their entire lengths. Exact matches are marked with a star. It can be seen that at 274 of the 417 comparable positions the amino acids are the same. The numbering corresponds to the horse sequence. Arginine 168 in the yeast sequence is marked with a caret.

compared with that of the wild type enzyme [Minard *et al.*, 1987]. This experiment was performed as part of a project investigating the catalytic mechanism of PGK, however, the important point in this instance is that this small change has a dramatic effect on the properties of the molecule.

These two extreme examples, the first showing more than 140 changed amino acids having very little effect on the properties of a protein and the second showing a single amino acid substitution which virtually inactivated the same protein, are a good illustration of the problem inherent in the comparison of protein sequences; unfortunately PGK is not an exception, there are good reasons to believe that this phenomenon is ubiquitous.

Simply stated, if an alignment is to indicate as accurately as possible a functional similarity, then the contribution, either positive or negative, made to the goodness of the alignment by a mismatched pair of amino acids depends both on what those two amino acids are and also on the precise molecular environment that they inhabit.

In order to improve the discrimination and sensitivity of the dynamic programming alignment algorithms a scoring method is needed that will try to take these phenomena into account. Instead of scoring separately for a match and a mismatch a single operation that can provide a score that varies according to the biological similarity of amino acids is required.

A method of devising such similarity scores has been proposed [Dayhoff *et al.*, 1978]. According to this proposal similar protein sequences are similar because they have evolved from a common ancestral sequence by divergent evolution. It should be noted that this model and the values so derived are applicable only to sequences that are related in such a manner; if the sequences are related by convergent evolution, that is to say they come from more dissimilar ancestral sequences then some other model is required. This is not as big a problem as might first appear as convergent evolution is thought to be very much rarer than divergent evolution [Gould, 1980]. Additionally, they assume that the probability of amino-acid X mutating to amino acid Y is the same as that of Y mutating to X. They justify this by saying that the frequency of a particular mutational event is related to the product of the frequency of occurrence of the amino-acids concerned and their physical and chemical similarity.

These workers model the mutational process in terms of two distinct quantities; the frequency at which each amino acid changes to each other one and the propensity for each amino acid to remain unchanged. The former quantity is referred to as the frequency of accepted point mutation. This is because it quantifies the replacement of one amino acid by another that, as a consequence of the continued existence and functionality of the protein, can be said to have been accepted by natural selection. The latter quantity they call the relative mutability although relative immutability might be a better term.

In order to calculate these quantities they collected data for some 1500 mutational events from pairs of sequences that they had aligned. They restricted the alignments to pairs of sequences that showed fewer than 15% unmatched pairs of amino acids. The reason they give for this restriction is that it makes double mutations so unlikely that it is unnecessary to consider them.

The frequencies of accepted point mutation for all pairs of amino acids were collected by counting all unmatched amino acids in these alignments. The relative mutability of a particular amino acid is proportional to the number of times it has been changed divided by the number of times it occurs. As these values come from many different alignments of different evolutionary distances and of different lengths they must be normalized to an equivalent evolutionary distance. The arbitrary distance chosen corresponds to a single mutation per hundred residues. This distance is referred to as one PAM (Point Accepted Mutation).

These two kinds of data, quantifying individual mutation frequencies and relative mutabilities, were combined to produce a single distance-dependent mutation probability matrix. This is called the one PAM mutation probability matrix. For every given pair of amino acids there is an element of this matrix that gives the probability that the first amino acid will replace the second after an evolutionary interval of one PAM. This matrix can be multiplied by itself N times to yield a matrix that predicts the amino acid replacements to be found after N PAMs of evolutionary change. It should be noted that, strictly speaking, this can only be said to be the case for a sequence that is of the same composition as the original sample.

If these values are to be used to calculate the probability that one

sequence has mutated into another then they must be modified to represent the probability (or odds) that a particular amino acid will be substituted by any other. This is done by dividing the values in the various PAM tables by the frequency of occurrence of the amino acids in the original sample; the frequency of occurrence of an amino acid represents the opportunity it had to mutate. This table can be used multiplicatively to calculate the odds for whole proteins. As it is computationally easier to add, the logarithms of the matrix elements are normally used. Figure 5.2 shows the log odds table for 100 PAMs which was used extensively in this project.

Although these results have been used by many workers, particularly in the form of the 250 PAM log odds matrix, there is a certain amount of dissatisfaction with them. Although this dissatisfaction is largely empirical it can be rationalized.

Examination of the original data used reveals that some of the values derive from an extremely small number of observations; as low as no observations at all for some pairs. Additionally more subtle problems derive from the fact that the data are being used in different circumstances from those under which they were collected. The 250 PAM table corresponds to sequences that contain only 20% of identical amino acids, whereas the sequences that the data were collected from contained 85% or more of identical amino acids. There is no evidence to support the assumption that the factors which operate over short evolutionary distances are the same as those which operate over longer ones, indeed preliminary evidence from other workers [Collins & Coulson, 1987] suggest that this is not the case. Also, the data were collected only from those proteins whose sequence was known at the time, almost entirely globular proteins. Amongst the most vociferous protesters at the inadequacy of these similarity values are those who work with membrane associated proteins.

During this project a range of similarity tables have been used based on this original data, with what were considered glaring deficiencies subjectively corrected. The one hundred PAM table was used most widely. The ability of the programmes to discriminate between interesting and uninteresting alignments derives from the use of similarity tables and the exhaustive nature of the algorithm.

In order to answer the criticisms above it would be desirable to have

	A	B	C	D	E	F	G	H	I	K	L	M	N	P	Q	R	S	T	V	W	X	Y	Z
A	6	-1	-5	-1	0	-7	1	-5	-3	-5	-5	-3	-1	1	-2	-5	2	2	0	-11	0	-6	-2
B	-1	4	-11	4	1	-11	-2	-2	-5	-2	-7	-6	1	-3	-1	-5	-2	-3	-5	-14	0	-7	-1
C	-5	-11	14	-11	-11	-10	-9	-6	-5	-11	-12	-11	-8	-6	-11	-6	-1	-5	-4	-13	0	-2	-11
D	-1	4	-11	3	5	-11	-1	-1	-6	-2	-9	-8	4	-4	1	-6	-1	-2	-6	-14	0	-9	1
E	0	1	-11	5	8	-11	-2	-2	-5	-2	-7	-6	1	-3	4	-5	-2	-3	-5	-14	0	-7	4
F	-7	-11	-10	-11	-11	12	-8	-4	0	-11	0	-2	-6	-9	-10	-7	-5	-6	-5	-2	0	6	-11
G	1	-2	-9	-1	-2	-8	8	-7	-7	-5	-8	-8	-1	-3	-5	-8	1	-3	-4	-13	0	-11	-5
H	-5	-2	-6	-1	-2	-4	-7	11	-7	-3	-5	-7	3	-2	4	1	-4	-5	-6	-7	0	-1	-2
I	-3	-5	-5	-6	-5	0	-7	-7	9	-4	2	2	-4	-6	-5	-4	-4	-1	5	-12	0	-4	-5
K	-5	-2	-11	-2	-2	-11	-5	-3	-4	8	-6	1	1	-4	-1	4	-2	-1	-6	-9	0	-10	-2
L	-5	-7	-12	-9	-7	0	-8	-5	2	-6	9	4	-6	-5	-4	-7	-7	-5	1	-7	0	-5	-7
M	-3	-6	-11	-6	-6	-2	-8	-7	2	1	4	13	-5	-6	-2	-2	-4	-2	1	-11	0	-8	-6
N	-1	1	-8	4	1	-6	-1	3	-4	1	-6	-5	7	-3	-1	-3	2	0	-5	-8	0	-3	-1
P	1	-3	-6	-4	-3	-9	-3	-2	-6	-4	-5	-6	-3	10	-1	-2	1	-1	-4	-11	0	-11	-3
Q	-2	-1	-11	1	4	-10	-5	4	-5	-1	-4	-2	-1	-1	9	1	-3	-4	-5	-11	0	-9	4
R	-5	-5	-6	-6	-5	-7	-8	1	-4	4	-7	-2	-3	-2	1	10	-2	-4	-6	1	0	-10	-5
S	2	-2	-1	-1	-2	-5	1	-4	-4	-2	-7	-4	2	1	-3	-2	6	3	-4	-4	0	-6	-3
T	2	-3	-5	-2	-3	-6	-3	-5	-1	-1	-5	-2	0	-1	-4	-4	3	7	-1	-10	0	-6	-4
V	0	-5	-4	-6	-5	-5	-4	-6	5	-6	1	1	-5	-4	-5	-6	-4	-1	8	-14	0	-6	-5
W	-11	-14	-13	-14	-14	-2	-13	-7	-12	-9	-7	-11	-8	-11	-11	1	-4	-10	-14	19	0	-2	-14
X	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Y	-6	-7	-2	-9	-7	6	-11	-1	-4	-10	-5	-8	-3	-11	-9	-10	-6	-6	-6	-2	0	13	-9
Z	-2	-1	-11	1	4	-11	-5	-2	-5	-2	-7	-6	-1	-3	4	-5	-3	-4	-5	-14	0	-9	4

Figure 5.2

The log(odds) table for 100 PAMs used extensively in this project. The table contains one cell for each possible amino acid substitution. Each cell contains the logarithm of the odds of the particular exchange taking place during the course of 100 PAMs of evolutionary time. The PAM is an arbitrary unit described in the text.

better tables. The ideal solution would be to generate new tables using the very much larger amount of data that is now available. In addition it would be desirable to collect data from sequences that are more dissimilar than the 15% maximum dissimilarity used previously and also to generate tables for particular classes of proteins. An obvious examples being those proteins that are associated with membranes. This work will require large amounts of computing resource to generate the very large numbers of alignments that will be necessary. Fortunately the efficiency of the DAP programmes mean it can be done although the task will require many hundreds of hours of DAP time. This work is currently in progress.

CHAPTER SIX

Significance

It is invariably the case that the first question a biologist will ask on seeing a pair of sequences aligned is

> "..is it [the alignment] significant?"

The correct answer to this question is

> "...the alignment is significant if it represents an evolutionary or functional relationship between the two biological molecules whose sequences are aligned."

Bearing this important point in mind it can however still be useful to have a figure that gives some idea of the likelihood of a particular alignment occurring by chance.

Because alignments can contain insertions and deletions in either sequence it is not immediately obvious how to calculate the likelihood of an alignment occurring by chance. It is a simple matter to calculate the probability of getting a certain number of identical bases in a region of known length using the lengths of the sequences and the frequency of occurrence of the letters. This calculation is quite satisfactory for modelling matches and mismatches but does not take account of insertions and deletions.

A practical approach that has been used [Goad & Kanehisa, 1982] is to perform database searches using randomly generated sequences. The results of such searches can be used to construct a standard curve. This method has been criticised [Lipman *et al.*, 1984] as it does not take account of the various inhomogeneities of real sequences. These workers have generated their random sequence by scrambling real sequences so as to preserve the various statistical properties of real sequence. These include such things as highly skewed and characteristic nearest neighbour frequencies

and various inhomogeneities of sequence composition. However, even this is not entirely satisfactory as real DNA and protein sequence have other properties which they are unable to mimic.

A theoretical approach has been taken to calculate the significance of an alignment [Clayton *et al.*, 1981]. If an alignment is considered as a series of events each of which elongates it by one base then this series has the property of being a Markov chain [Feller, 1968]; the probability of each event depends only upon the current state and not on any previous state. This means, using the base frequencies counted for the sequence in question, that it is possible to calculate the probability of each type of event extending the alignment. These values comprise a transition matrix [Feller, 1968] which may be used to calculate the significance of the alignment. This however, is computationally expensive and an approximation has been used which is said to require 10^3 less computation and produce probabilities good to 1 part in 10^4 [Clayton *et al.*, 1981].

All of these methods have two major disadvantages; they do not accurately model real sequence data and they require a considerable amount of computing resource.

The DAP implementation of the type three dynamic programming algorithm described in chapter four stores all the alignments that score higher than a very trivial threshold until the whole database has been searched. At this point the alignments are sorted and the best few presented to the user. For any database search against a single query sequence it can be assumed that the first few alignments may represent genuine relationships of biological interest whilst the remainder are fortuitous. These unrelated alignments may be used to generate a measure of significance for each alignment as follows.

The scores of a substantial number of the best alignments, currently 4096, are collected. The top scoring 3% of the alignments are removed. The remaining 97% of scores are assumed to have arisen fortuitously and are plotted against the logarithm of the number of alignments attaining that score and fitted to a straight line. The 3% of alignments that are assumed to include all significant ones are also plotted on the same graph.

Examination of the graph shows that many of the alignments in the

top 3% deviate markedly from this line and can thus be assigned a numerical significance. Figure 6.1 is graph of the alignments produced for one of the studies, that involving the human CF antigen, in chapter eight.

For each alignment, two numbers are generated. The number of alignments that would be expected to attain that score purely by chance for that particular combination of database and query, and the number that were actually detected by the programme. These two numbers can be compared and used to guide the intuition of the biologist who is assessing the alignments.

This method has very great attraction, not only because of its rigour and the very small extra amount of computation required but also because it satisfies the requirements which are intuitively felt to be necessary by biologists, in particular it uses real sequence data. All of the studies in chapter eight use this method to assess the significance of the alignments generated in them.

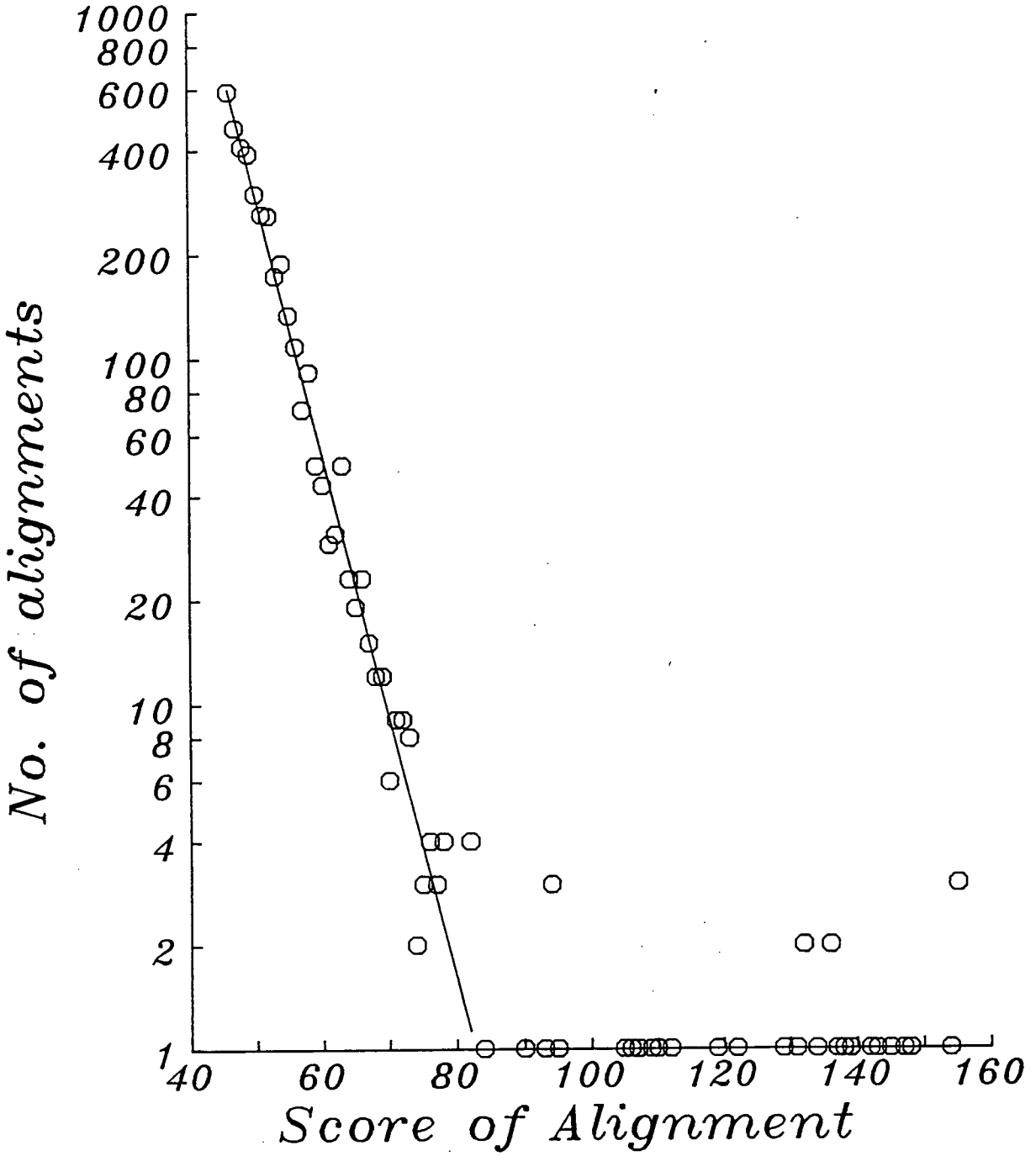


Figure 6.1

An example of the type of plot used to assess the significance of alignments detected with the authors implementation of the type three dynamic programming algorithm. These results are from the search using the CF antigen described in chapter eight. The interpretation of such plots is explained in the text although, roughly speaking, points that deviate markedly from the straight line indicate alignments that are likely to be due to a functional or evolutionary relationship. [After Collins *et al.*, 1988].

CHAPTER SEVEN

Approximate Methods

Most molecular biologists do not have access to computers like the DAP and thus cannot conveniently perform database searches using dynamic programming algorithms. In order to overcome this problem an algorithm has been devised to pre-process the sequence data bases to extract those sequences, it is hoped a small number, that have a marked similarity to the query sequence [Wilbur & Lipman, 1983]. The rationale for this approach is that it is then feasible to process this small number of entries with a program that uses the dynamic programming method on an inexpensive computer.

The algorithm relies on the fact that two similar sequences will contain short regions of exact identity. Such regions are called words. Whilst this is a reasonable assumption to make for close relationships it is not reasonable for distant ones. This is unfortunate as more trivial and faster algorithms can find the obvious relationships already; it is the distant ones that are difficult and require the power of the dynamic programming algorithms.

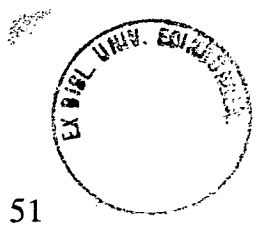
The Approximate Algorithm

The algorithm proceeds by using a user supplied word length to build a hash table from all possible words of that length in the query sequence. The locations of duplicate words are stored in a linked list pointed to by the appropriate entry in the hash table. A histogram is created with one entry for each diagonal of the match-matrix. Every word in the database is looked up in the hash table and, if it occurs, the appropriate entry in the histogram is incremented. Overlapping words only increase the counter in proportion to the amount that they increase the length of the perfect match by. The database sequences are filtered by examining the histogram and choosing those sequences that have clusters of diagonals that score highly.

In addition to failing to detect similarities that do not contain a substantial number of common words the algorithm also has the unfortunate property of favouring diagonals which contain a large number of widely separated perfect matches over those which contain a smaller number of closely spaced ones. This latter effect is a problem because it is likely that the small number of closely spaced matching words are due to a biological feature whereas the widely separated ones may not be. Most astounding of all is the enormous amount of unnecessary work such programmes perform. As the whole database is searched every time the programmes are used, their run times are proportional to the size of the database. If, instead of converting the query sequence into a dictionary, the database were stored as a dictionary and the words of the query looked up in that then the runtime would be proportional to the length of the query sequence (normally some 3 to 4 orders of magnitude shorter!). It is true that in this case there is the overhead of constructing and storing the database dictionary, however the databases are not distributed very often and there is no reason why the dictionary should not be constructed at a central site and distributed in a similar manner to the databases.

Many molecular biologists currently use implementations of the approximate algorithm as a matter of routine on mini-computers [Devereux *et al.*, 1984], [Lipman & Pearson, 1985]. Even these programmes require significant amounts of CPU time. This is unfortunate as it discourages repeated use of the programmes whilst varying the parameters, something that is essential if maximum benefit is to be obtained from them. This is necessary because the particular alignments that are detected by the approximate algorithm are very dependent on the values that the user supplies for the programme parameters. In this respect the critical parameters are the wordsize and the number of diagonals of the match-matrix over which integration takes place.

If such users had access to versions of the databases stored as dictionaries they would be able to perform database searches in seconds rather than minutes or hours. An obvious use of the DAP is to use it to generate such dictionaries by sorting.



Sorting

Sorting is a very major activity in computing. It is fundamental to data processing and is also used considerably in scientific computing. A great deal of work has gone into the development of efficient sorting algorithms for machines with serial architectures.

Naïve algorithms, for example the insertion sort, have a complexity of N^2 making them unsuitable for anything other than very small data sets, their only virtue being ease of implementation. The most efficient algorithms have a complexity of $N\log_2 N$; the goal of the algorithm designer being to minimize the value of the constant by which this is multiplied.

Heapsort [Williams, 1964] is widely used as it is a true *in situ* method, has very good behaviour in that it is also $N\log_2 N$ for the worst case order of input data and is relatively easy to implement. For very large problems where the ultimate performance is required the algorithm with fastest average performance is Quicksort [Hoare 1961]. Quicksort is a partition-exchange sort; by pair-wise exchange of elements the original dataset is partitioned into two subsets which are sorted independently by the same method. This process could be continued until the subsets contain two elements, at which point they can be trivially ordered. However when the subsets get down to a certain size it becomes quicker to sort them by an insertion sort. The exact size at which this happens is machine dependant. Further drawbacks of Quicksort are that it requires an additional 2 $\log_2 N$ locations of storage to store the subsets in and that its worst case behaviour is extremely bad indeed; of order N^2. This behaviour occurs when the data already contains some degree of order with correctly sorted data being the worst possible case. When implementing quicksort it is necessary to guard against this eventuality. Recommended methods include sampling the data prior to sorting and using a random number generator to select the data [Knuth, 1973].

Parallel sorting algorithms

A much smaller, though not inconsiderable, amount of work has been done on the development of sorting algorithms for parallel computers. Many of these algorithms are not suitable for the DAP because they assume architectural features that it does not possess such as independent addressing

within the PEs [Baudet & Stevenson, 1978]. The preferred algorithm for this application is the bitonic algorithm [Batcher, 1968]. The so-called odd/even algorithm of the same author can achieve slightly better performance under certain circumstances but is much more difficult to program [Reddaway & Flanders, 1982]. So, rather like the Heapsort/Quicksort choice for serial sorting Bitonic sorting is the choice on the DAP unless extremely large data sets are to be sorted. Like Quicksort, bitonic sorting relies on a divide and conquer approach analogous to partition-exchange. However, whereas Quicksort sorts the independent subsets one at a time the bitonic sorting algorithm takes advantage of the independence of the subsets and the parallel nature of the hardware on which it is running to sort many of the subsets simultaneously.

A bitonic sequence is one which when considered cyclically has one ascending portion and one descending portion. The critical property of bitonic sequences used by the algorithm is as follows. If a bitonic sequence, Ai where i = 1,2,..n, is split into two sequences such that:-

```
A1i = min (Ai,Ai+x/2) where 1 < i < x/2 and
A2i = max (Ai,Ai+x/2) where 1 < i < x/2
```

then A1 and A2 are also bitonic sequences and all the elements in A2 are greater than or equal to all the elements in A1.

If this process of splitting is applied to a bitonic sequence of 2^N elements and then again to the two sub-sequences and so on until there are 2^N sequences each containing one element it can be seen that each element will be greater than or equal to all elements in preceding sequences. An arbitrary set of 2^N items may be converted in to a bitonic sequence by the following method.

The sequence of 2^N numbers may be considered as 2^{N-1} sequences of 2 numbers each of which is trivially bitonic. If these bitonic sequences are now ordered so that the first and alternate sequences are in increasing order and the second and alternate ones are in decreasing order then each pair of the 2^{N-1} sequences will be a bitonic sequence of four elements. The number of sequences has been halved and the number of elements each contains has been doubled. This process is repeated until there is a single bitonic

sequence of 2^N elements which may be ordered as described previously. Figure 7.1 illustrates the whole process for a sequence of eight elements.

The basic operation of the Bitonic sort is a comparison-exchange in which a pair of elements is compared and conditional exchanged depending on the result of the comparison. If one is sorting N elements then at each stage of the algorithm there are N/2 such comparison exchanges. For a complete sort there are $\frac{1}{4}N\log_2N^2$ of these [Flanders, 1982]. This is about $\frac{1}{4}\log_2N$ more than the number of equivalent operations in a serial sort.

When implementing a bitonic sort on DAPs the aim is to arrange it so that all the PEs are active and performing distinct comparison exchanges for as much of the time as possible, avoiding duplication of comparison-exchanges. An additional problem with the current generation of DAPs is that routing is a relatively slow process and can dominate execution time if care is not taken.

The QMC libraries described in chapter three contain sorting routines for individual vectors and long vectors. The routine for sorting a single long vector could be used to sort a large number of layers, however this is extremely inefficient as it entails a great deal of repetition. A sorting programme written using the long vector sort routine proved so unsatisfactory that it was decided to write a new routine.

As with all DAP programming the arrangement of data in store is the most important consideration. Data is usually considered as being stored in the DAP store in horizontal mode. This is the mode in which the Host store mode conversion routines expect the data they convert to be in. If the data is represented by a 3-dimensional array then if it is horizontal mode the first index is the fastest running.

Data stored in vertical mode is often referred to as being crinkled. If the data is represented by a 3-dimensional array then if it is in vertical mode the third index runs fastest, the first index second fastest and the second index slowest.

A sorting routine was written that left the data in horizontal mode. Experience with this routine indicated that a significant proportion of the time was spent in routing. A second routine was written that left the data in

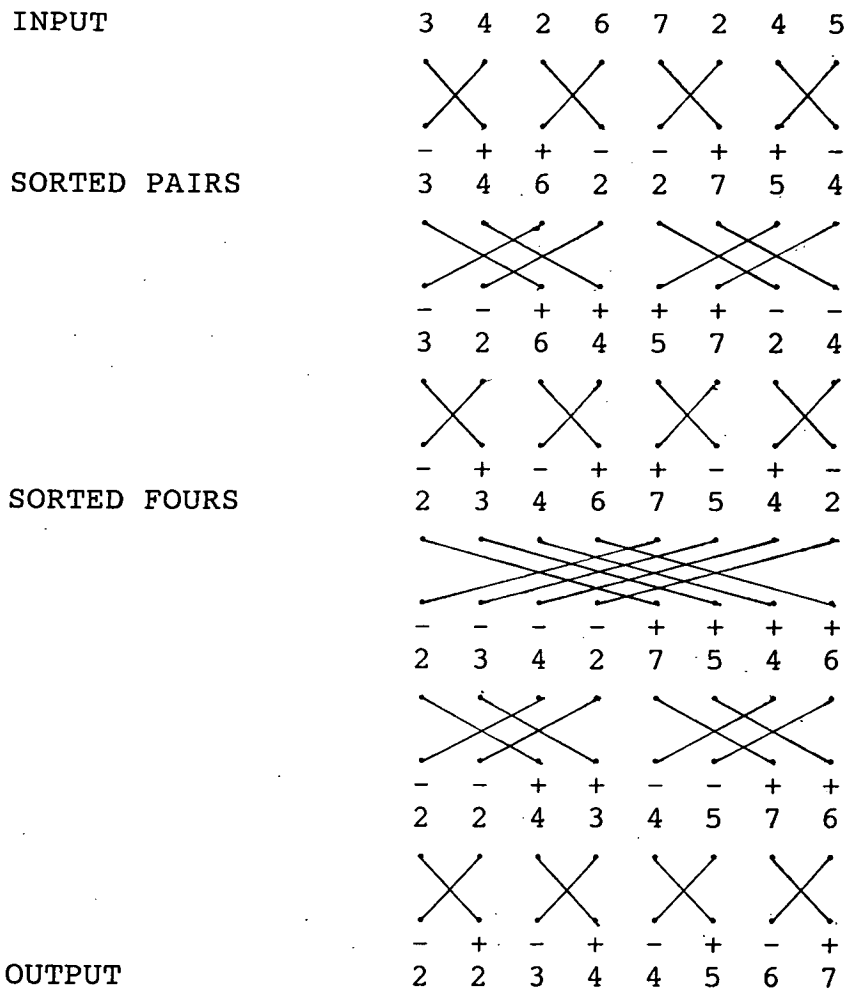


Figure 7.1

Batchers Bitonic sort for eight items. At each step data is moved as indicated by the diagonal lines and compared with the data already there. + & - indicate whether the larger or smaller element is selected.

vertical mode and thus reduced the amount of routing. An accompanying routine was written using the musical bits routines to convert the data from the crinkled mode in which it is left by the vertical mode sorting routine directly into the host store mode. These two routines combined ran in less than one third of the time of the horizontal mode routine on its own.

It is possible, using the musical bits routines, to vary the mapping vector during the sort with the aim of reducing routing even further. Theoretical timings for such a routine have been calculated [Reddaway & Flanders, 1982]. It was decided that the extra effort involved in coding a variable mapping vector sort was not worth while. Timings for these four methods of sorting on the DAP are given in figure 7.2.

Method	Time/secs
QMC Longvector sort	>20
DAPFORTRAN Horizontal mode	6.5
DAPFORTRAN Vertical/uncrinkle	2
APAL variable mapping vector	<0.5

Figure 7.2

DAP CPU time to sort 1,048,576 (1M) of 12 byte data on the ICL 64 x 64 prototype DAP by the various methods discussed in the text.

External sorting.

When the amount of data to be sorted exceeds the size of the main store the data is divided into blocks of that size, these blocks are sorted separately and then merged in a number of phases. Each phase combines successively fewer blocks until a single sorted block is obtained.

In the following description it is assumed that two blocks of fixed length records, both in ascending order, are to be merged into a single block also in ascending order. Records are processed from each block in fixed length sub-blocks each containing N records.

1) The first sub-block is taken from either block and moved to a merge area, M.

2) The next sub-block is chosen by comparing the first element in the first sub-block in both blocks. The one with the smaller first element is chosen.

3) These two sub-blocks are merged with the bitonic merge.

4) The lower half of the merge area is output.

steps 2, 3 & 4 are repeated until both blocks are exhausted.

As the two sub-blocks in the merge area are both sorted they may be taken together as a bitonic sequence and sorted as already described. An m-way merge combines m sorted blocks into a single sorted block. When m is a power of 2, an m-way merge may be performed by a complete binary tree of two-way merges. The first level performs m/2 two-way merges to produce m/2 blocks, the next level uses these as inputs to m/4 two-way merges to produce m/4 output blocks and so on until, finally, a single sorted block is produced. When m is not a power of two, branches of the tree may be omitted as required.

For an m-way merge a block of N records is output after each stage of $\log_2 m$ merges. The work per merge is proportional to $N\log_2 N$. From this it can be seen that the bitonic merge works most efficiently with the smallest sub-block size concomitant with fully parallel operation of the array [Flanders, 1982].

This however assumes that transfers from disk to the DAP present no problems. As explained previously, there is no direct interface between the DAP and the 2900 disk packs. Data has to be read from the disk to the 2900 memory and then moved into the DAP. Furthermore, using the standard versions of the DAP support software input-output is not possible; data is built into the DAP program block which is then moved into the DAP, the program is executed and the block moved out to the host. Due to the fact that the host is also running a time sharing operating system with up to one hundred users, at peak times it takes between one and two minutes to get a contiguous area of virtual memory the size of the DAP, build the program block and load it into the DAP. In an attempt to alleviate these problems the ERCC produced an enhancement to the support software called DDX [Brown, 1986]. This provides a limited form of I/O that has to be exactly

specified before running the program. Obviously this is not suitable for merging as the I/O necessary for merging is data dependent and thus not known until run time.

The solution to these problems was to write a serial program on the host which scattered the data based on the first few characters of the sort key into blocks that were guaranteed to be no larger that the one megabyte of keys, this being the maximum that the two megabyte DAP store could handle bearing in mind that memory was also needed for the half megabyte of tags and the program. This strategy was feasible because the composition of the protein databases is known and relatively constant over the short timescale of this project. As their order relative to each other is known at run time, these blocks could then be presented to DDX in the correct order and sorted in the DAP in batch mode (DDX does not work interactively). This rather unfortunate compromise actually produced an extremely fast sort/merge program. In the future DAPs will have direct access to disk and these problems will cease to be an issue.

Applications of the Sequence Dictionaries

Using the sorting and merging routines described above the NBRF protein database was sorted into a dictionary based on a key of five amino acids. This number was chosen because, with the current size of the database, it produced a set of keys the majority of which were unique.

Because it is not necessary to store the keys; the tags can be binary searched and the key worked out by looking in the data base at the appropriate position, the dictionary only takes up five times the amount of space that the database takes up. For each position in the database there is the letter itself which is one byte and a four byte integer for the tag. The databases would have to get very large indeed before it becomes necessary to move to eight byte tags.

This dictionary was transferred to a serial machine and a number of applications that used it were developed by other workers. These are briefly described below.

Fast database searches.

A serial programme that used the dictionary to mimic the

approximate algorithm described above ran in a few seconds, making it between two and three orders of magnitude faster. It should be noted that the relative costs of CPU power and secondary storage mean that this approach represents a very acceptable trade off.

The programme collected all database sequences containing more than a certain number of words in common with the query sequence and then assessed them using three different methods. The first and simplest method counted the total number of common words, the larger the number of words the better the similarity. The second method assessed the density of matching words in order to favour short alignments containing closely clustered matches above longer ones containing the same number of matches but more widely spaced. This method was used because it was felt that the shorter alignments were more likely to be indicative of biological similarities. The third method used a set of rules to link sets of matching words that, due to their proximity are likely to be part of the same similarity.

Experience with this programme revealed that patterns of length five were insufficiently common, not finding a large enough number of similar sequences to be useful. On the other hand sequences of length three matched too frequently and increased the amount of processing unacceptably. The programme was most satisfactory, and exceptionally fast, when using tetra-peptides. Elapsed time was further reduced by creating a two tier index to the dictionary to reduce paging; on average a single word could be recovered with just one or two page throws. In this production version the programme (PRELATE) was deemed to fulfil the requirements of a database searching programme [Collins & Coulson, 1987].

Database comparison using non-overlapping words.

A considerable number of individuals, organisation and national bodies have established collections of sequence data. All of these databases overlap to some degree although usually not very considerably. As all these databases are being continuously updated on an almost daily basis it would be highly desirable to be able to compare them very quickly and remove identical and almost identical sequences to produce a single database without duplications. Dictionaries provide a very fast way of doing this.

One of the databases is converted into a dictionary. The other

database is taken one sequence at a time and converted into non-overlapping words. A sequence of n letters would produce n/m words of length m. These words are then looked up in the dictionary and if more than a certain very high percentage of them occur in the same sequence in the dictionary then the query sequence is discarded. If this is not the case it is added to the data base from which the dictionary was made. Because of the high degree of similarity that is being sought it is acceptable to use non-overlapping words which gives very high speed. On an ICL 2976 running EMAS a single sequence of about 100 letters could be compared with the approximately one million letters of the NBRF database in less than half a second of CPU time. A particularly useful feature of the performance of this programme is that it shows its best behaviour when the sequence being tested is not present in the database. This makes the process of merging dissimilar databases exceptionally fast.

Further developments.

When used for detecting similarity, the fast techniques discussed in this chapter depend on the fact that the similar sequences contain short perfect matches. This need not be the case. The danger when using such a method is that a similarity that is indicative of a functional or evolutionary relationship may go undetected because it does not contain sufficient short perfect matches.

The dictionary method is currently being extended in an effort to overcome these problems. The approach used is to generate additional dictionaries based on alternative orderings of the letters present in the words. The aim is to permit the detection of discontinuous or partially mismatched similarities at similar efficiency to the exactly matched ones.

For example, to extend the method to cope with single indels in tetra-peptides it is necessary to construct three additional dictionaries. If one imagines looking up the word ABCDE then these dictionaries would be sorted on orders corresponding to ACDEB, ABDEC and ABCED. It is not necessary to arrange for the first letter (A) to be omitted as these cases will be taken into account when the words starting with the second letter (B) are looked up. Initial results with these dictionaries [Collins & Coulson, 1987] indicate that the increase in sensitivity is very considerable.

This technique possess great potential for enhancement. For example, by designing the reordered sub-strings appropriately it should be possible to exploit the short-range orderings present in protein sequence. These are caused by structural features and motifs common to all proteins such as α-helices and β-sheets. The reordered sub-strings would be designed using the knowledge that members of the various classes of amino-acids occur with periodicities attributable to these structures. The possibilities afforded by exceptionally fast searches based on this kind of feature are very great.

Although this method showed extremely promising preliminary results, it was not pursued as far as would have been desirable as hardware limitations were still a problem. Even the 12-fold increase in the size of the database necessary for the three additional dictionaries described above caused problems on the computer system being used.

This need only cause concern in the short term. In the medium term it seems likely that optical technology disk drives will make it realistic to consider expanding the databases 100 or even 1000 fold. When this happens it may well be that this method will supersede all others as the preferred way of searching sequence databases for similarities.

CHAPTER EIGHT

Alignments

This chapter contains five sequence comparison studies that were performed using the DAP implementation of the type three dynamic programming algorithm developed during this project. All five examples involved comparing a query sequence of unknown function with a database containing between five and six thousand protein sequences. The database was based on the most recent version of the National Biomedical Research Foundation (NBRF) Protein Identification Resource (PIR) protein sequence database with a small number of additional sequences obtained from other sources. The alignments for examples one and two were generated by the author and the conclusions were drawn in discussions with those workers who provided the sequences. The alignments for examples three, four and five were generated by other workers using the author's programme with appropriate guidance.

Each study involved a small number of runs of the programme, usually just a single run, each run taking between half and one hour of CPU time. As time on the DAP is charged at one hundred pounds per CPU hour this represents an extremely cost-effective approach. Each run produced hundreds or even thousands of alignments, however only the highest scoring ones are shown in the listings, although the scores of them all are summarized in the accompanying tables. These tables also include various administrative data produced by the programme at run time. These data differ between the tables because the studies were performed at different stages during the development of the programme. The tables and listings are in Appendix A and their format is explained in a key which precedes them. The listings are numbered from 1 to 5 and the individual alignments are also numbered. So, for example, alignment number 14 in listing number 2 is referred to as alignment number 2.14. In all five cases the alignments have been ordered by their absolute score.

MerC Gene From Plasmid R100

MerC is a predicted gene product from the mercury resistance determinant of the bacterial plasmid R100. R100 shows considerable similarity to the transposon Tn*501* which also confers mercury resistance, however, Tn*501* does not posses an equivalent gene product to *merC* [Summers, 1986]. For this reason, it appears that *merC* might not be essential for mercury resistance in the R100 system. As the *merC* gene product has recently been identified on gels [Ni'Bhriain & Foster, 1986] demonstrating the existence of a protein corresponding to the open reading frame, it seemed sensible to try to assign function by analogy. As a database search with an inexhaustive programme had not revealed any interpretable alignments the *merC* gene product was an ideal candidate for evaluating the exhaustive method.

Alignment number 1.1 is with the *merT* gene product from Tn*501*. It has been proposed [Misra *et al.*, 1984] [Misra *et al.*, 1985] that the *merT* protein contains three membrane spanning regions. Furthermore, the proposed mechanism for mercury detoxification [Brown, 1985] requires that the *merT* protein sits in and spans the inner membrane. The region of the *merT* protein that takes part in alignment number 1.1 includes the first transmembrane helix, a surface bend and the second transmembrane helix. This strongly suggests that the *merC* gene product is associated with a membrane in a similar manner to that of *merT*.

Alignment number 1.4 is with the same region of the *merT* protein from R100 which has the same function and properties as the *merT* protein from Tn*501*.

Similarity is also shown to other membrane proteins. Alignment number 1.2 is with pBR322 tetracycline resistance protein which is a membrane-associated protein that acts to exclude tetracycline from the cell [Pedan, 1983]. Alignment number 1.3 is with the M chain of *Rhodopseudomonas sphaeroides* reaction centre. The reaction centre is a membrane-bound complex that mediates the initial photochemical event in the electron transfer process of photosynthesis [Williams *et al.*, 1983]. Alignment number 1.16 is with the c-chain of the non-enzymic components of the membrane associated ATPase complex [Gay & Walker, 1981]. Further examples of these proteins from other species occur throughout listing 1.

The conclusions to be drawn from these alignments are as follows. It seems likely that the *merC* gene product is a membrane protein. It also seems likely that *merC* bears a relationship, either evolutionary or functional, to *merT*. It is, for example, possible that the pair of cysteine residues at positions 23 and 26 are involved in binding Hg^{2+} in the same manner as has been proposed for the two pairs, the first at positions 24 and 25, and the second as positions 76 and 82, present in the *merT* gene product [Brown, 1985].

Although these alignments appear to show a convincing and believable functional relationship, their expectation values, calculated as described in chapter six, are not much above the noise level. For this reason it is necessary to be careful when interpreting them. They are, however, sufficiently high scoring for it to be worthwhile to test the relationships that they suggest by experiment. It is hoped that this will be done in the near future.

Streptomyces coelicolor gylR

The glycerol utilization (*gyl*) operon of *Streptomyces coelicolor* A3(2) is currently being characterized; *gylA* codes for glycerol kinase, *gylB* for glycerol-3-phosphate dehydrogenase and *gylX* for a product of unknown function. In addition a 0.9 kilobase transcription unit, containing an open reading frame for a 27.6 kilodalton protein, has been identified and since circumstantial evidence exists for its role in regulation of the *gyl* operon it has been tentatively assigned the name *gylR*.

The evidence suggesting that *gylR* has a regulatory role is as follows. Firstly, transcription of *gylR* is specifically induced by glycerol-3-phosphate, the product of the first step of the glycerol utilization pathway [Seno & Chater, 1983]. This suggests a positively-acting role in glycerol catabolism for the *gylR* gene product [Smith & Chater, 1988]. It should be noted that, in view of the ready availability of glycerol and glycerol-phosphate esters in the organisms normal habitat, this observation is consistent with the demand theory of gene regulation [Savageau, 1977]. Secondly, two glycerol non-utilizing mutants that are complemented by DNA from the *gylR* region have been identified [Smith, 1986]. One of these has been partially sequenced and found to contain a mis-sense mutation [Smith, 1988].

The first thing to notice is that none of the alignments with expectation values that might be considered as significant are interpretable in a biologically interesting manner.

The most promising alignment is number 2.14. This is with *E. coli* hypothetical protein E-152. This sequence was submitted to the database as an open reading frame of unknown function [Nakamura *et al.*, 1981]. Since then the gene has been identified as a regulatory element in asparagine metabolism and named *asnC* [de Wind *et al.*, 1985]. The protein, AsnC, has been identified and found to be an autogenously regulated activator of asparagine synthetase A transcription [Kolling & Lother, 1985]. The alignment includes the region of AsnC that has been postulated as binding to DNA. Furthermore, residues 27 to 46 of the gylR gene product fulfil the criteria for a classical helix-turn-helix DNA-binding structure [Pabo & Sauer, 1984] and are aligned against the region of AsnC that also fulfils these criteria. This relationship is illustrated in figure 8.1.

Alignment number 2.21 is with glycerol-3-phosphate acyl transferase from *E. coli*. This enzyme catalyses the committed step of phospholipid synthesis; the acylation of glycerol-3-phosphate with a fatty acyl-CoA.

Unfortunately, little is known about which residues of glycerol-3-phosphate acyltransferase are involved in the glycerol-3-phosphate binding site, however, the observation that the enzyme is inactivated by phenylglyoxal and butanedione suggest that arginine residues may be in or near the active site [Green & Bell, 1984]. As it happens, the region of the acyltransferase that takes part in the alignment includes 6 arginine residues. It is thus conceivable that this is the region of glycerol-3-phosphate acyltransferase that binds glycerol-3-phosphate and that it shows similarity to the gylR protein in the region of it that also binds glycerol-3-phosphate. As these proteins are of rather different origin and function it is possible that this similarity is the result of convergent evolution.

Alignments number 2.10 & 2.33 also include this region of the *gylR* protein. They are with exactly corresponding regions of glyceraldehyde-3-phosphate dehydrogenase (GAPDH) from *Bacillus stearothermophilus* and *Thermus aquaticus* respectively.

Glyceraldehyde-3-phosphate and glycerol-3-phosphate have similar

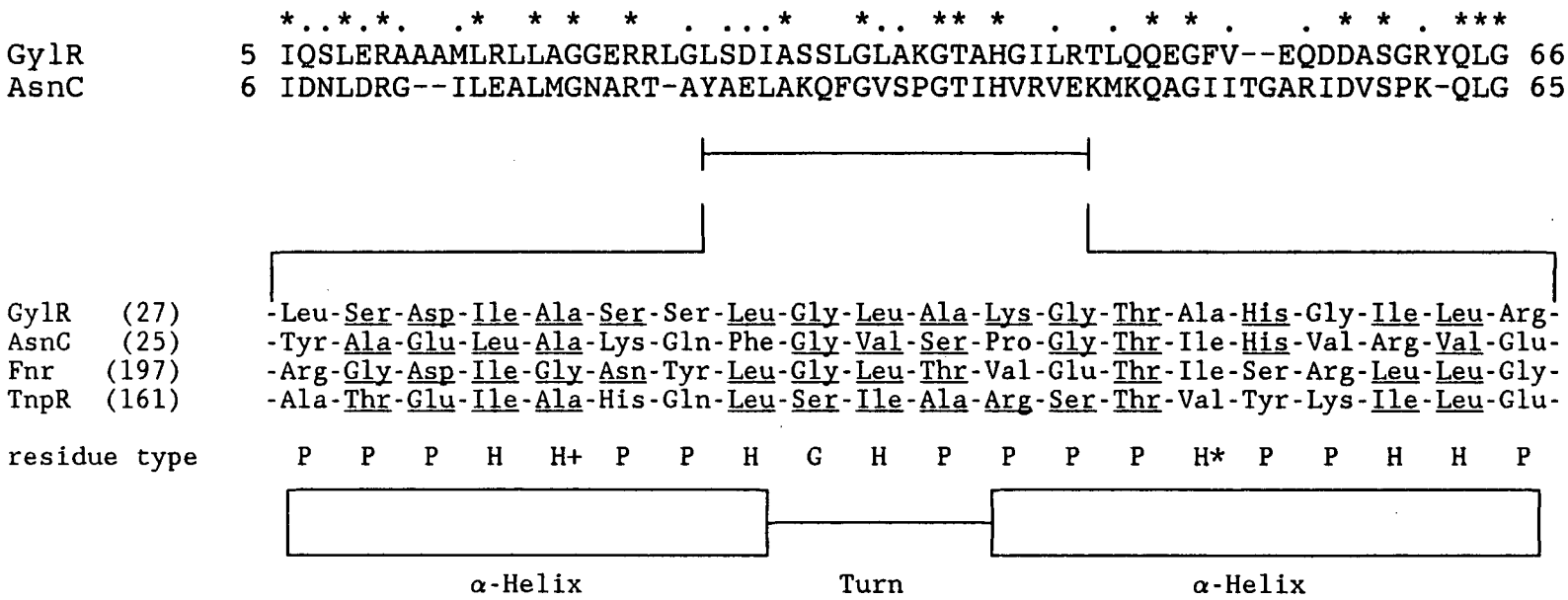


Figure 8.1

(upper) Alignment of the amino acid sequences of *Streptomyces coelicolor* GylR and *E. coli* AsnC using the 100 PAM similarity table. Identical residues are marked with a star and pairs of residues that score positively are marked with a full stop. The postulated DNA binding domain of AsnC is indicated with a bar. This alignment was generated using the authors programme and has been reported elsewhere [Smith & Chater, 1988]. (lower) Similarities between GylR and the so-called helix-turn-helix DNA binding domain from some other *E. coli* DNA binding proteins from a published collection [Dodd & Egan, 1987] generated using the FASTP programme [Lipman & Pearson, 1985] against a set of known regulatory proteins. Underlined residues score positively against GylR at 100 PAMs. The consensus residue types [Pabo & Sauer, 1984] are indicated with the following code: G - predominantly glycine; H - usually nonpolar; H+ - as H, alanine favoured; H* - as H, usually valine or isoleucine; P - usually polar.

structures. Apart from carbon 1, which is part of an aldehyde in glyceraldehyde-3-phosphate and an alcohol in glycerol-3-phosphate, they are identical. For this reason it would not be surprising to find that their binding sites show sequence similarities. Indeed it is known that glycerol-3-phosphate acyltransferase is inhibited by glyceraldehyde-3-phosphate and that the kinetics of inhibition most closely resemble simple competitive inhibition with respect to glycerol-3-phosphate [Green & Bell, 1984]. In essence this means that glycerol-3-phosphate and glyceraldehyde-3-phosphate bind to glycerol-3-phosphate acyltransferase in the same place. Thus it seems possible that it is the region of GAPDH that binds glyceraldehyde-3-phosphate and the region of the gylR protein that binds glycerol-3-phosphate that are aligned in these two alignments.

Unfortunately this hypothesis has to be rejected for a close examination of the structure of *Bacillus stearothermophilus* GAPDH, which has been determined to a resolution of 1.8 Ångstrøms [Skarzynski *et al.*, 1987] reveals that this region is not sufficiently near to the active site for it to be involved in binding glyceraldehyde-3-phosphate.

Another alignment that may be of interest is 2.25 which is with *E. coli* NusA (also called L-factor). NusA binds directly to the core of *E. coli* DNA-dependent RNA polymerase. It participates in the antitermination reaction mediated by bacteriophage lambda N gene protein to prevent premature termination of transcription initiated at the early lambda promoters and has been demonstrated to be necessary for the *in vitro* synthesis of β-galactosidase [Ishii *et al.*, 1984]. The hypothesis suggested by this alignment is that, just as NusA interacts with *E. coli* RNA polymerase in order to exert control over transcription, so the *gylR* protein interacts with the analogous polymerase in *Streptomyces coelicolor* and that this alignment is between the regions in each protein that do this.

Most tantalizing of all are the two extensive alignments numbers 2.42 and 2.59 which are both with the same protein, Gene 430 protein from bacteriophage pf3. This is an hypothetical sequence deduced from an open reading frame and unfortunately nothing is known concerning its function. The two regions corresponding to these two alignments are transposed between the two proteins so that the larger region, corresponding to alignment number 42, is C-terminal in P430 and N-terminal in GylR and the

```
Alignment number 42.
          ** *  **   **  * . *. *..* *..    * *  **  .* *     ....**  ..    * * *  **  ..  *  *  .  .*.  . *.*..  .*    .. *  .  *  *  **.*  *.*.
P430   204  RLGLNWGGALSLGNWSAVTAGDLSVAAGSSIGFGFLSNTLSLDGLFTAMENEGNGR VVSRPTL L T  LD RQ SASVL RGTELPYQQSAGDGAT SVAFKHAALSLEVKPVISPDNSI VIEV   320
Gy1R    16  RL LA GGERRLGL SDI ASSLGLAKGTA  HGILR TLQQEG FVE QDDASGRYQLGAELLRLGTTYLDVHELRARALVWTDDLA R SSGESVHLGVLHQQGVL I VHHVFRPDDSRQVLEI   129

Alignment number 59.
          * *.  .* **  .* * * .**    *  . .*  *  .  .  *. *. .* .*  . *. .*     *. | *. **
P430    80  LLSSMVGDVLVITAMDQVLNSERKADDLRTFR RDLFNANDIERRVINIVHA S ASEVVSLFKESFMSLDAP   149
Gy1R   135  LHSTALGKVL  SAYDPVAHSEALEADRKAFTDRTVCEPDSFEH VLDITRARGYAADVEETW EGIASIAAP   203
```

Relationship between the similar regions of P430 (upper) and Gy1R (lower).

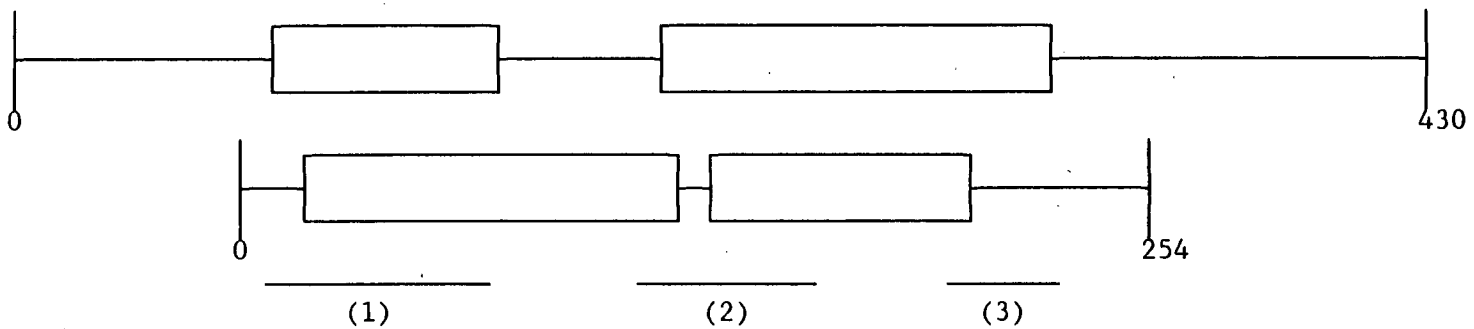


Figure 8.2

The relationship between *Streptomyces coelicolor* GylR [Smith & Chater, 1988] and Gene 430 protein from Bacteriophage Pf3 [Luiten *et al.*, 1985]. The alignments were generated using the 100 PAM similarity table. Identical residues are marked with a star and residues that contribute positively with a full stop. The two similar regions occur in opposite orientations in the two proteins. The large number of indels present in these alignments and the fact they do not posses expectations above the noise level suggest that they may well be fortuitous however since nothing is known concerning the function of P430 it is not possible to judge them from a biological stand-point. The numbered bars mark the alignments between GylR and (1) the AsnC helix-turn-helix DNA binding motif, (2) Glycerol-3-phosphate acyltransferase and (3) *nus*A protein. It is proposed that the region of GylR corresponding to (1) is involved in binding to the *gyl* operator, to (2) is the Glycerol-3-phosphate induction binding site and to (3) is the region that interacts with the DNA-dependent RNA polymerase to control transcription. Although all three of these hypotheses fit with the proposed biological function of GylR none of them have been tested experimentally.

smaller one is N-terminal in P430 and C-terminal in GylR. This relationship is illustrated in figure 8.2.

If these two regions are of common origin then it is likely that they correspond to distinct domains and that a domain order reversal has occurred during the course of evolution. Such a phenomenon has been postulated for a ribosomal protein (known variously as L7, L12 & L12e) [Liljas & Thirup, 1986] between organisms at least as distantly related. In this case eukaryotes and archaebacteria show one arrangement whilst the eubacteria show the other.

In summary, the original idea, that GylR positively regulates the *Streptomyces coelicolor* glycerol utilization operon, gains considerable support from this set of alignments. In order to achieve its regulatory effect it may well bind to DNA. This could take place in the region that aligned with the helix-turn-helix motif of AsnC. Furthermore it could bind the inducer molecule in the region that aligned with the (hypothetical) glycerol-3-phosphate binding site of glycerol-3-phosphate acyltransferase. Finally it could interact with the DNA-dependant RNA polymerase in the region that aligned against NusA protein. The fact that all three of these regions are well spaced out in the GylR sequence adds further support to the ideas as with this arrangement it would be possible for the three functions to occupy separate domains. These relationships are illustrated in Figure 8.2. It is hoped that at least some of these hypotheses will be tested by experiment.

This set of alignments is particularly interesting and was included because it is a good example of the problem inherent in biological sequence comparison. The problem, discussed in chapter six, is that of assessing the significance of alignments. Although these alignments appear to show convincing and believable functional relationships, their expectation values, calculated as described in chapter six, are below the noise level. Great caution must be exercised when dealing with low scoring alignments like these.

Drosophila melanogaster YP3

The three *Drosophila melanogaster* yolk proteins (YP1, YP2 & YP3) are studied as models for developmental regulation. They occur only in the fat body and ovarian follicle cells of adult females and are temporally, tissue-

specifically and sex-specifically regulated [Bownes & Hames, 1978], [Brennan *et al.*, 1982]. They are assumed to play a nutritional role in embryogenesis and until this study no other function had been proposed for them. The sequences of YP1 and YP2 have been known for sometime and when optimally aligned show 53% identity [Hung & Wensink, 1983]. YP3 has recently had its sequence inferred from a *Sal*I-*Hin*dIII genomic DNA fragment [Garabedian *et al.*, 1987].

Only the first three alignments in listing 3 have expectations indicating that it is exceedingly unlikely that they occurred by chance.

Alignments 3.1 and 3.2 are with YP1 and YP2. These similarities are extremely marked and had already been reported. As they align along their entire length they have been omitted from the listing. Alignment 3.3 is with porcine triacyl glycerol lipase. This observation had not previously been reported.

The substrate binding site of the lipase has been studied [Guidoni *et al.*, 1981] and spans the region consisting of residues 147 to 156. Serine 152 is believed to be involved in lipid-water interface recognition and its hydroxyl group is almost certainly an absolute requirement for the lipase activity. From the alignment it can be seen that YP3 does not possess an equivalent serine residue to serine 152 of the lipase. The analogous position in YP3, residue 264, is occupied by a glycine which, not possessing a hydroxyl group, could not substitute functionally for the serine. YP3 does have a serine at position 266, however, it seems unlikely that this one could substitute functionally in the active site, as tolerances in active sites are always extremely small and it is in the wrong position. This view is supported by the absence of a serine in the equivalent positions in YP1 and YP2, namely positions 257 and 258 respectively. It thus seems unlikely that any of the YPs will possess lipase activity. These relationships, more extensive alignments between all the YPs and the lipase and a consensus sequence are illustrated in figure 8.3.

Why then should the YPs show similarity to a lipase? This question has been addressed and the results of the enquiry reported elsewhere [Bownes *et al.*, 1988]. They may be summarized as follows.

The YPs do not possess a lipase activity. They do however possess a

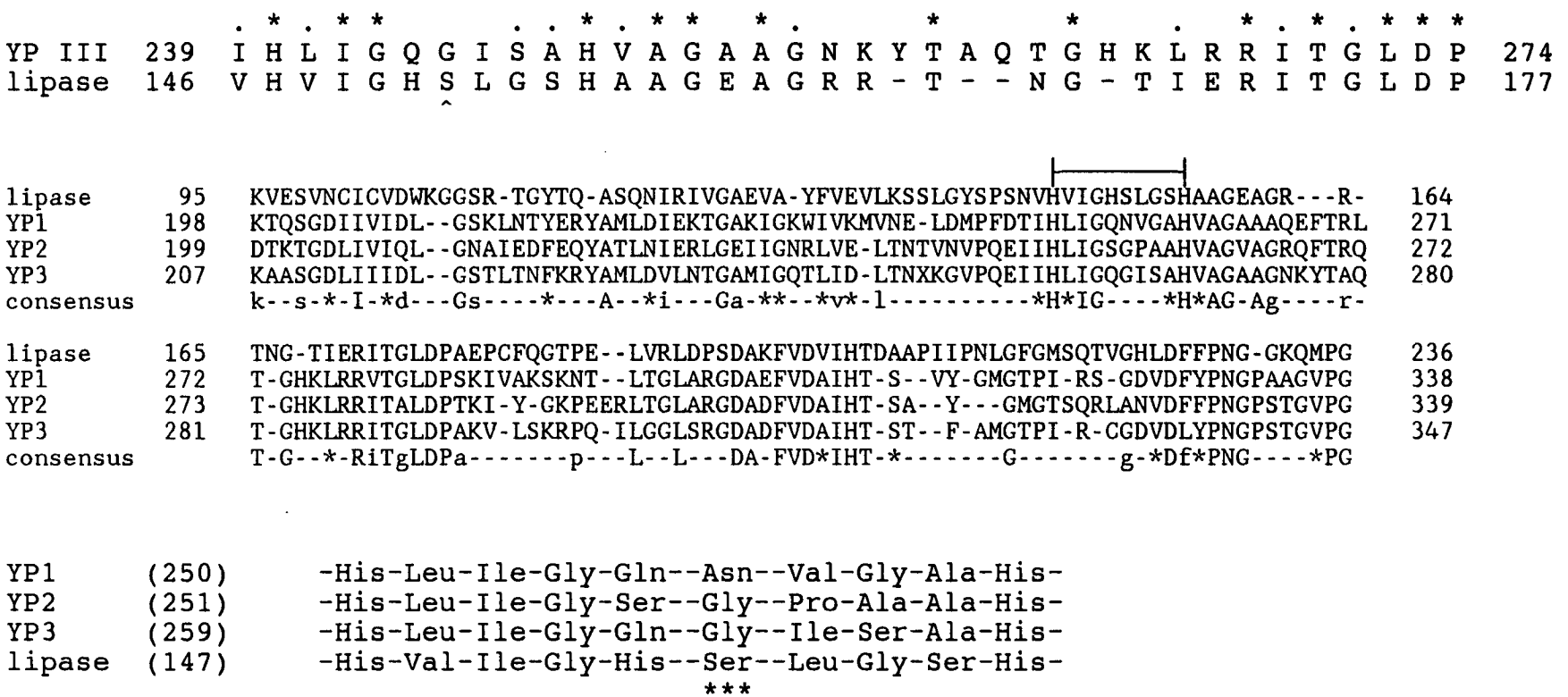


Figure 8.3

(top) The similarity between *Drosophila melanogaster* YP3 and Porcine Triacylglycerol lipase using the 100 PAM similarity table. Identical residues are marked with a star and those that contribute positively to the score with a full stop. The alignment was generated using the authors programme and has been reported elsewhere [Bownes *et al.*, 1988]. (middle) More extensive alignments between the lipase and all three YPs generated using the GAP programme from the UWGCG package [Devereax *et al.*, 1984]. In the consensus line an upper case letter indicates that all four sequences have that residue at that position, a lower case letter indicates that two of the YPs and the lipase have that residue at that position and a star indicates a position where all pairwise comparisons yield a positive score using the 100 PAM table. The bar marks the position of the active site in the lipase. (bottom) The active site region of the lipase with the catalytically important serine indicated with stars.

triacyl glycerol binding activity, which functions to bind apolar conjugates of ecdysteroids. Ecdysteroids are insect moulting hormones and it is proposed that proteolytic degradation of the YPs times the release of the bound ecdysteroid and hence cuticle development. It would appear that this fundamental process has been highly conserved during evolution since in locusts ecdysteroid polar conjugates are bound to vitellin [Lagueux *et al.*, 1981] and are released, as free ecdysteroid, in peaks that coincide with cuticle secretion [Lui, 1984].

This study is a good demonstration of the power of sequence comparison when it is used in conjunction with experimental work. To quote the final paragraph of the publication reporting the experimental work [Bownes *et al.*, 1988]

> "...an unexpected sequence similarity between the YPs and a lipase led us to devise and successfully test a model in which YPs play an important role in regulating embryogenesis; this mechanism seems to have been conserved in many insects and arthropods."

Human Cystic Fibrosis Antigen CFAg

Cystic fibrosis is an autosomal recessive disease in humans. The biochemical basis of the disease is not known although elevated levels of a serum protein, referred to as the Cystic Fibrosis Antigen (CFAg), has been described in both homozygotes and obligate heterozygotes [Bullock et al., 1982]. The sequence of CFAg has recently been inferred from a cDNA clone [Dorin et al., 1987]. It was decided to attempt to ascribe function to CFAg by analogy.

The first seven alignments in listing 4 have exceptionally small expectations of occurring by chance and it is almost inconceivable that they do not represent a functional or evolutionary relationship. These scores are illustrated graphically in figure 6.1.

Of these first seven alignments 4.1, 4.3 and 4.4 are with the homologous a and b subunits of human and bovine S100 calcium binding protein from the brain. Alignments 4.2, 4.4 and 4.6 are with mammalian intestinal calcium binding proteins. These proteins all belong to a class of proteins that have been postulated to be major transducers of biological calcium signals [Van Eldrick *et al.*, 1982]. In addition to this many of the

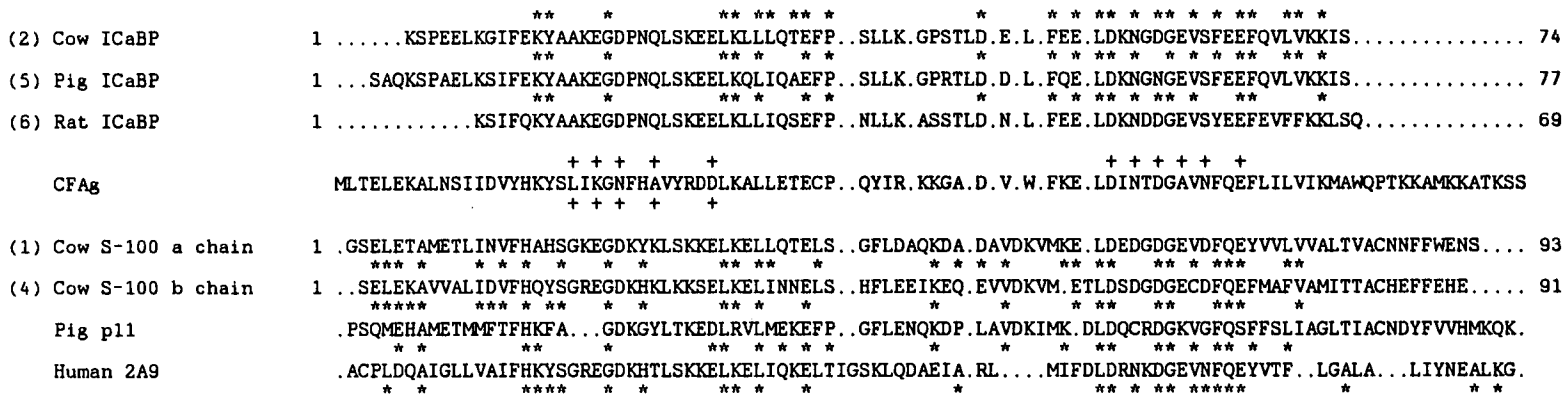


Figure 8.4

Alignment of the predicted amino acid sequence of human cystic fibrosis antigen (CFAg) with amino acid sequences from three intestinal calcium binding proteins (ICaBPs) [Fullmer & Wasserman, 1981] and the brain associated calcium binding proteins S-100 α and S-100 β [Isobe & Okuyama, 1981]. Also included are two sequences, not present in version 12 of the NBRF database, that have recently been demonstrated to be similar to the S100 proteins. They are porcine P11, a regulatory subunit of the complex that is the major cellular target for tyrosine kinase [Gerke & Weber, 1985] and 2A9 a hypothetical protein sequence deduced from a human cell-cycle specific cDNA clone [Calabretta *et al.*, 1986]. Residues that are identical to the equivalent ones in CFAg are marked with a star. Residues in CFAg analogous to the residues that bind calcium in the other proteins are marked with plus signs; the group to the right of the figure mark residues that are ligands in the EF-hand region of the ICaBPs, the group to the left mark residues that are ligands in the type two calcium binding region present in both the ICaBPs and the S100 proteins. These alignments were generated by the authors programme and have been reported elsewhere [Dorin *et al.*, 1987].

subsequent alignments are to other proteins that bind calcium, including calmodulins and troponins. Figure 8.4 shows these alignments and also indicates residues involved in calcium binding.

These observations prompted the investigators concerned to perform equilibrium binding studies using $^{45}Ca^{2+}$ in competition with unlabelled Ca^{2+} which showed that each molecule of CF antigen binds two calcium ions with a similar affinity to that shown by S100 protein [Bock & Haywood, 1987]. This experiment indicates that CFAg, like the ICaBPs, contains one of each of the two types of calcium binding site.

How this observation, that CFAg binds calcium, fits in to current understanding of CF pathology is not immediately obvious. It is however an important clue, and taken together with the observation that β-adrenergic stimulation of chloride channel activity observed for normal cells is absent in CF cells [Welsh & Liedtke, 1986] suggests that the basic defect will be in a component of the cAMP, phosphotidyl inositol or other related signal transducing pathway [Dorin *et al.*, 1987].

Escherichia coli ftsA

FtsA codes for a cell-cycle protein from *E. coli*, the sequence of which has been determined recently [Robinson *et al.*, 1984]. Although nothing is known of its function at a molecular level it has been demonstrated that synthesis of the *ftsA* protein during the 10 to 15 minute period immediately prior to cell division is an absolute requirement for cell division is to take place successfully [Donachie *et al*, 1974].

An initial comparison between the database and the sequence of the *ftsA* protein revealed a striking similarity with CDC28, a cell-cycle protein from the budding yeast *Saccharomyces cerevisiae.*

The sequence of a similar protein, CDC2 from the fission yeast *Schizosaccharomyces pombe*, which was not present in the version of the database used, was obtained and it was also found to align with *ftsA*. The alignment between these two proteins and the *ftsA* protein is shown in figure 8.5.

Both these yeast proteins have had kinase activities demonstrated *in*

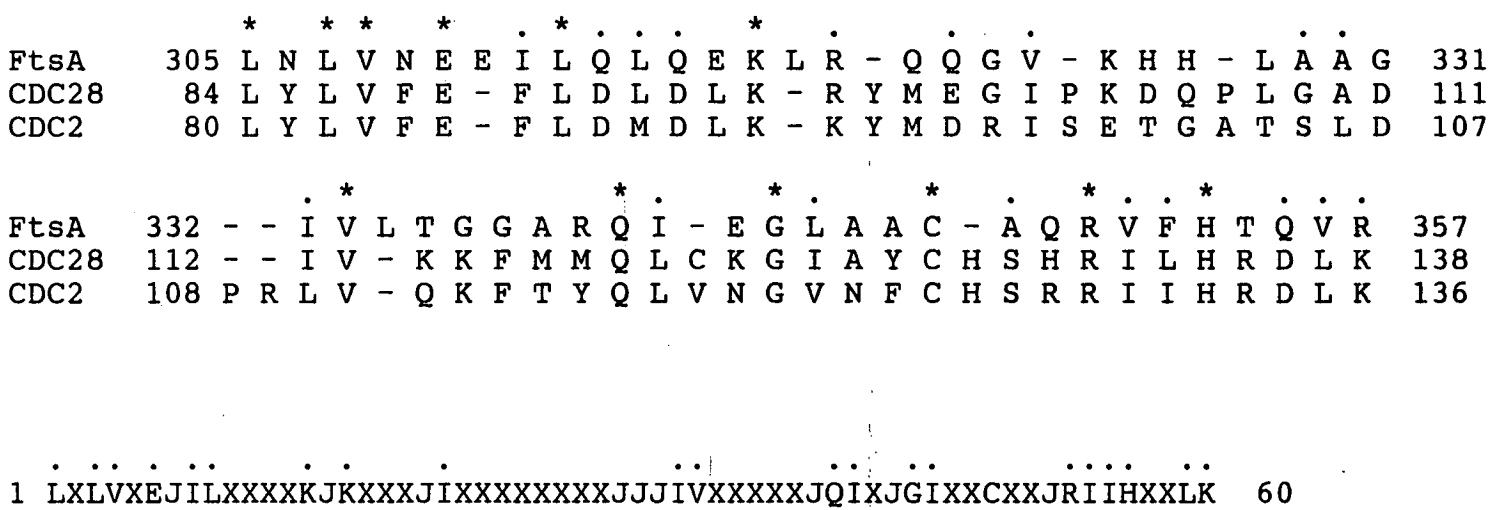


Figure 8.5

(top) Alignment of cell-cycle proteins from three genetically diverse organisms using the 100 PAM similarity table; FtsA of *E. coli* [Robinson *et al.*, 1984], CDC28 protein of *Saccharomyces cerevisiae* [Lorincs *et al.*, 1984] and CDC2 of *Schizosaccharomyces pombe* [Hindley *et al.*, 1984]. Locations where all the residues are identical are marked with a star. Locations where all pairwise comparisons of residues would contribute positively to the score of all pairwise alignments are marked with a full stop. The alignment between *fts*A protein and CDC28 was generated using the authors programme and has been reported elsewhere [Robinson *et al.*,1987]. (bottom) The modified sequence of *fts*A protein used for the conserved element search described in the text.

vitro [Reed *et al.*, 1985], [Simanis & Nurse, 1986] however as the regions of the yeast proteins that align with the *ftsA* protein do not include the consensus sequences for the ATP-binding and phosphorylation sites it is not reasonable to propose kinase activity for the *ftsA* protein on the evidence of these alignments alone.

In order to try to get a clue as to the function of the common residues, the region of *ftsA* protein that aligned with CDC28 was taken and the letters that did not contribute positively to the score of the alignment replaced with Xs and Js. The metric was modified so that these letters, which do not represent any amino acid, score zero when mismatched. In addition, in order to simulate the gaps between the conserved elements, it was arranged that a space occurring opposite a J would not be penalized as an indel whereas one occurring opposite an X would be. This modified sequence, shown in figure 8.6 was used in a repeat of the original search and it is the alignments produced by this second search that are shown in listing 5.

From these alignments it can be seen that, once again, even using only the conserved elements, no proteins align significantly with the *fts*A protein apart from CDC28; alignment 5.1. Although this is disappointing as it does not provide any clues as to the function of the protein at a molecular level, what it does indicate is that the similarity is very significant indeed. Had there been any other proteins showing even a very faint similarity to the conserved regions, either due to a functional or evolutionary relationship or fortuitously, then it is likely that this search would have detected them.

The alignments between FtsA and the CDCs suggest two hypotheses [Robinson *et al.*, 1987]. As these three organisms are genetically extremely diverse, it may be that these sequence similarities are the result of convergent evolution. The alternative hypothesis, that this domain arose before the emergence of eukaryotes, suggests a very severe functional constraint on the further evolution of the sequence.

Although in the absence of experimental evidence it is never possible to be certain that an alignment corresponds to a functional or evolutionary relationship the extremely low value for the expectation of these alignments and the failure to pick up any other similar sequences in the conserved element search very strongly suggest that they are not due to chance similarities.

CHAPTER NINE

Conclusions

The advantages gained by using exhaustive algorithms for sequence comparison to assign function and evolutionary relationship by analogy is not disputed. The rationale for not using the exhaustive methods is always given as being economic. The development of inexpensive parallel architecture computers must now challenge this view.

Machine	Agency	Cost/CPU hour	Mips	Cost/Mip
VAX 750	ERCC	£40	1	£40
ICL DAP	ERCC	£100	20	£5
CRAY-1	RAL	£500	160	£3

Figure 9.1

Cost of commercially available computer time in 1985.

Figure 9.1 shows the commercial cost of time on a VAX 750, a DAP and a CRAY-1 from the Edinburgh Regional Computer Centre (ERCC) and the Rutherford Appleton Laboratory (RAL). The mips figures (Million instruction per second) often used as an indication of relative performance, and thus quoted here, do not take into account the different architectures of the machines and do not give a comparison of performance on the sequence comparison problems. The programmes written in this project for the ICL DAP achieved a performance at least as good as some similar programmes written for the CRAY-1 [Smith *et al.*, 1985]. This is because the CRAY-1 architecture has been optimized for programmes that perform a lot of high precision arithmetic, such as one finds in engineering applications, and cannot achieve the same performance with sequence data. In the case of the DAP, sequence comparison programmes map onto the hardware extremely

efficiently and provide performance far in excess of that which the 20 mips figure might suggest. For these problems the DAP provides as least as much power as the CRAY-1 at substantially less cost. The DAP is also very much more cost effective than a VAX.

The surprising conclusion must be that, not only are exhaustive methods demonstrably better biologically but, with appropriate implementations on appropriate hardware, they are also cheaper than existing inexhaustive methods.

Searching the Nucleic Acid Databases

There is no intrinsic reason why the programmes written in this project and used to search the protein sequence databases cannot be used with nucleic acid sequence. There are, however, a number of pragmatic reasons associated with using nucleic acid sequence, which have caused the programmes to be much less successful than when they were used with protein sequence.

Specific to this project were the problems with the prototype DAP described in chapter three; it had only two megabytes of store and no direct connection to disks. At the time of writing, the protein sequence database, the query sequence and the programme all fitted into two megabytes. This could be loaded into the DAP in a single operation in response to the DAPRUN command. This was not the case when using nucleic acid sequence. The EMBL nucleic acid sequence database contained some ten million bases and a search against it required several runs of the DDX system. In practice this was always problematic.

There are also problems inherent in the comparison of nucleic acid sequence that do not pertain to the comparison of protein sequence. The twenty character alphabet used to represent proteins results in a low noise level; on average fortuitous matches will occur at a frequency in the region of 5%. In the case of nucleic acids where only four different letters are used the frequency of fortuitous matches is very high; in the region of 25%. This makes it much more difficult to distinguish distant similarities from background noise. There is also a problem concerning the degree to which indels may be tolerated in similarities that are deemed to be significant. With protein sequence there is good reason to suspect alignments that have indels

scattered evenly through them, whereas those that have their indels clustered are much more likely to be indicative of a real similarity. This is because proteins are constructed from a number of distinct secondary structural features, principally α-helices and β-sheets, alignment against which must be considered as being very sensitive to indels. Conversely, the unstructured regions and loops by which these features are connected may be considered as being tolerant of large numbers of indels when they are involved in alignments. Thus, considerable success has been achieved with protein sequences using metrics designed to penalize indels quite severely. It should be noted that, as explained in chapter two, the soft limiting nature of parameters to the dynamic programming algorithm ensures that this does not prohibit large indels altogether. However, in cases where this is felt to be a problem, the soft limiting effect can be enhanced by reducing the size of the penalty as the size of the indel increases.

For these reasons, the best way to deal with DNA that codes for protein is to translate it into the corresponding protein sequence and search against the protein sequence data base using a PAM table. All five of the studies in chapter eight are of this type and it is fortunate that a great deal of the nucleic acid sequence that is of interest to contemporary molecular biologists codes for proteins.

Similar structural considerations make it likely that these criteria also apply to nucleic acid sequences that code for structural RNAs and to regions of DNA that interact in a specific manner with proteins. Other nucleic acid sequence, in the case of a higher eukaryote like man the vast majority of the genome, is of unknown function and its behaviour with respect to indels can only be guessed at.

For all these reasons, working with protein sequence has proved to be much more rewarding. In the future it is hoped that larger computers and an improved understanding of appropriate ways of dealing with nucleic acid sequence will yield similar successes.

The Human Genome Project

During the course of the latter part of this project there has been considerable discussion in scientific circles concerning the idea of sequencing the entire human genome. Figure 9.2 is a reproduction of a poster

IntelliGenetics, Inc., and BIONET
present a Symposium

Advances Toward Sequencing the Human Genome

Speakers

Dr. Ronald Davis
Stanford University School of Medicine
Methods for Physically Mapping the Human Chromosome

Dr. Helen Donis-Keller
Collaborative Research, Inc.
Mapping the Human Genome with Genetic Markers: A Progress Report

Dr. Lloyd Smith
California Institute of Technology
Automated DNA Sequencing Using Fluorescence Detection

Professor Paul Berg of the Stanford University School of Medicine will lead a panel discussion on the scientific and management issues of the sequencing project.

July 21, 1987
9:00 a.m.
Fairchild Auditorium
Stanford University School of Medicine
Stanford, California

Lunch will follow

A User Group meeting will take place in the afternoon at IntelliGenetics headquarters, 700 East El Camino Real, Mountain View, California

Registration: $30; $15 for students
for details, call Mary Valente at (415) 962-7356

Limited Seating Preregister by July 14, 1987

Figure 9.2.

Reproduction of a poster advertising a meeting to discuss the practicalities involved in the Human Genome Project, the grand design of which is to sequence all 3,500,000,000 nucleotides of the human genome. By the end of the 1980s it is likely that this project, possibly the most far-reaching the human race has ever embarked upon, will have started in earnest.

announcing a recent scientific meeting to discuss this idea.

Although there is some scepticism concerning the feasibility of such a project and considerable opposition to funding it, it appears that the human genome project is going to happen. The National Academy of Sciences USA has convened a committee under its Basic Biology Board to consider the project and a report of a recent meeting of this committee contained the following [Lewin, 1987]

> "It is clearly no longer a question of whether the [human genome] project ought to be done, but of how fast it will be done."

The Human Genome Project, the grand design of which is to sequence all of the approximately three and a half thousand million bases that make up the human genome, must surely be one of the greatest scientific endeavours that the human race has ever embarked upon. It is only necessary to understand that the huge impact of modern molecular biology has been based on the sequence of less than one percent of the human genome, to be awestruck at the consequences of this great undertaking. The Human Genome Project is the beginning of the much heralded biological revolution.

There is, however, one very large caveat to this extravagant view. That the project is possible at all is without doubt due to the remarkable technologies for sequencing that were developed in the late 1970s. What is also without doubt is that for the sequence to be of any great value there must be concomitant advances in the technologies for handling, comparing and understanding it.

It is surely the case that dynamic programming algorithms running on highly parallel architecture computers will provide an essential tool for performing such analyses.

Recovery of Textual Material

The sequence databases would be quite useless without the textual material, described in chapter one, that accompanies them. This textual material contains limited details of the known biological function of the molecule represented by the sequence. Other relevant biological information and references to the literature are also included.

Although attempts have been made to structure this data, a lot of it is in the form of natural language and as there is still no agreement on a common format for those data that are structured, recovery of it is problematic.

A lot of the information specific to individual residues and regions of the sequences is stored in what are known as feature tables. These have a relatively regular and stable structure and the programmes written in this project would be greatly enhanced if they recovered this information and displayed it alongside the alignments. An example of the kind of residue-specific information that could be usefully treated in this manner is demonstrated by the case of alignment number 3.3; the alignment of *Drosophila melanogaster* YP3 with porcine triacylglycerol lipase. In this case the important biological information, was the position of the active site region of the lipase and in particular that of the catalytically critical residue; serine 252. In fact this information was not present in the database and had to be recovered from the literature. However, it can be seen that had it been present in the feature table for the lipase entry, and had it been recovered and displayed alongside the sequence, a great deal of time and effort would have been saved.

For the programmes written in this project additional information recovered and displayed for a particular alignment was restricted to the single line title that accompanies each database entry. As these titles are descriptive, they often proved to be enough. For example, in the case of the cystic fibrosis antigen, the alignments in against which are in listing 4, all the titles of the sequences involved in the interesting alignments included references to calcium binding and it was possible to start devising hypotheses without further recourse to the literature.

The contrasting case is demonstrated by alignment number 2.14, between the *gylR* protein and *E. coli* hypothetical protein E-152. In this case the only additional information present in the database was a reference to the literature. In fact a great deal more was known about E-152 and a time consuming literature search was required to find it out.

For the future, it is important that as much biological information as possible is included in the database entries, and that alignment programmes recover this information and present it to the user along with the alignment.

The Matrix of Biological Knowledge

The difficulty of recovering all the information associated with a particular piece of sequence extracted from a database provides a useful paradigm for a far larger problem believed to present in all modern biological sciences.

It is believed that the very large amount of data being produced by modern molecular biology and the very disparate and non-numerical nature of that data presents a new and unique problem. What is feared, is that the difficulties of accessing and comprehending the data will soon become so great that further progress in understanding will be inhibited and eventually cease all together.

The proposed solution to this problem involves the setting up of an international system consisting both of people and also the most modern computers and networks to connect them that will permit working scientists to gain rapid access to information that is essential to them whilst being outside of their direct area of expertise. The information to be contained in such a system has been named the Matrix of Biological Knowledge.

The Matrix of Biological Knowledge Workshop held at St. Johns College, Santa Fe, New Mexico in July and August 1987 under the auspices of the National Institutes for Health, USA (NIH) was the first scientific meeting held to address this problem. To quote from the report of this workshop [Morowitz, 1987]

> "We seem to be at a point in the history of biology where new generalizations and higher order biological laws are being approached but may be obscured by the simple mass of data."

A "working description", from the same report defines the matrix of Biological Knowledge as

> "...the complete database of published experiments, structured by the laws, empirical generalizations, and physical foundations of biology and connected by all the interspecific transfers of information."

The important point to note is that the matrix includes the results of any analyses that the data may have been subjected to as well as the data itself. An example of this relevant to sequence comparison would be a

computer programme that performed sequence database searches and then updated the databases with pointers between all significant similarities after each search was performed. It should be noted that it would probably need the intervention of a human expert during the assessment of similarity stage and that the computer system might well have to take precautions to protect the database from unsound judgements on the part of the human expert.

The development of such a large system is not without pitfalls. Modern molecular biology is a very recent innovation and consequently there has been very little time for an understanding of it to percolate through into other sciences. Furthermore it is very badly served by the media. Insufficient biological knowledge amongst the builders and designers could lead to a disaster.

There is unfortunately a precedent for this eventuality in the MOLGEN project [Friedland *et al.*, 1982], [Bach *et al.*, 1982]. This project came out of a university artificial intelligence department and had as its aim the development of expert systems for use in experimental design. It was intended to pioneer the practical application of such techniques as "knowledge engineering" and "knowledge representation". Molecular biology was chosen as a "test bed" because it was believed that it represented a "small stable body of knowledge" that could readily be incorporated into such a system.

Unfortunately, from a biological stand-point the project was an unmitigated disaster and it is likely that this was due to the lack of biological knowledge amongst the system builders. The software developed by the project was less useful and less efficient than equivalent systems developed using conventional software engineering techniques and third generation languages. The project has been closed down and the software is no longer used.

It is hoped that those involved in developing solutions for the matrix of biological knowledge problem will be aware of the dangers inherent in this type of grandiose project. Fortunately it appears that on this occasion they are proceeding more cautiously. For, with regard to the problem of lack of specialized knowledge among those who will be building the systems the report says

> "The development of the matrix [of biological knowledge] and the extraction of biological generalizations from it are going to require a new kind of scientist, a person familiar enough with the subject being studied to read the literature critically, yet expert enough in information science to be innovative in developing methods of classification and search."

Furthermore, it is a relief to see that the closing sections of the report maintain a degree of scepticisms as the following indicates.

> "The matrix of biological knowledge should by further investigated as a potential tool in biomedical research under the aegis of the NIH [National Institute of Health, USA]. The concept must be sharpened and tested as to its utility."

What ever solutions are developed to handle the matrix of biological knowledge, there can be no doubt at all that sequence comparison will be a major component. Further more, the vast quantity of sequence data that will shortly become available will ensure that biological sequence comparison on parallel computers will be an essential part of the future of all biological science.....

ACKNOWLEDGEMENT

I would like to thank my supervisors, Dr. J.F. Collins of the Department of Molecular Biology and Prof. S. Michaelson of the Department of Computer Science, both at Edinburgh University, for their help and encouragement throughout the project. I would also like to thank my adviser Dr. A.F.W. Coulson of the Department of Molecular Biology, Edinburgh University for many useful discussions.

I would like to thank my industrial supervisor Dr. S.F. Reddaway, and Drs. D. Hunt and P.M. Flanders all three of whom are members of the DAP team at The Systems Strategy Centre, International Computers Ltd., Stevenage, for their help in understanding and programming the DAP.

I gratefully acknowledge the receipt of a Science and Engineering Research Council CASE award with International Computers Ltd. (ICL).

I would like to thank Drs. N.L. Brown and C.P. Smith of the Biochemistry Department, Bristol University for the sequences, prior to publication, of *mer*C and *gyl*R respectively. I would like to thank Drs. M. Bownes and A.C. Robinson of the Department of Molecular Biology, Edinburgh University, for the sequences, prior to publication, of YP3 and *fts*A respectively. I would like to thank Dr. V. van Heyningen of the Medical Research Council Clinical and Population Cytogenetics Unit, The Western General Hospital, Edinburgh, for the sequence of CFAg, prior to publication.

DECLARATION

The following declaration is made in accordance with paragraph 2.4.15 of the University of Edinburgh regulations governing the submission of theses.

I declare that this thesis has been composed by me and that all the work described in it is my own unless clearly indicated otherwise.

Andrew Lyall
June 1988.
Bristol.

BIBLIOGRAPHY

Aho, A. & Corasink, M.J. (1975) "Efficient String Matching: An Aid to Bibliographic Search." Communications of the A.C.M., Vol. 18, No 6, pp. 333.

AMT (1987) "The AMT Distributed Array Processor 510." Active Memory Technology sales literature.

Anderson, S. (1981) "Shotgun DNA sequencing using cloned DNAse-I generated fragments" Nucl. Acids Res., Vol. 9, pp. 3015-3027.

Astbury W.T. & Street, A. (1931) "X-ray Studies of the Structure of Hair, Wool and Related Fibres." Phil. Trans. Roy. Soc. A. Vol. 75, pp. 230.

Bach, R., Friedland, P., Brutlag, D.L. & Kedes, L. (1984) "MAXAMIZE. A DNA sequencing strategy adviser." Nucl. Acids Res., Vol. 10, No. 1, pp. 295-322.

Bailey, J.L. (1967) "Techniques in Protein Chemistry" 2nd Edition, American Elsevier, New York.

Banks, R.D., Blake, C.C.F., Evans, P.R., Haser, R., Rice, D.W., Hardy, G.W., Merrett, M. and Phillips A.W. (1979) "Sequence, structure and activity of Phosphoglycerate kinase: A possible hinge-bending enzyme." Nature (London), Vol. 279. pp. 773.

Batcher K.E. (1968) "Sorting Networks and their Applications." Proc. AFIPS, SJCC. AFIPS Press, Montvale, N.J. Vol. 32, pp. 307-314.

Baudet, G. & Stevenson, D. (1978) "Optimal Sorting Algorithms for Parallel Computers." IEEE Transactions on Computers. Vol. C-27, No. 1.

Bellman, R. (1957) "Dynamic programming." Pub. Princeton: Princeton University Press.

Bownes, M. & Hames, B.D. (1978) "Analysis of yolk proteins in *Drosophila melanogaster.*" FEBS Lett., Vol. 96, pp. 327-330.

Bownes, M., Shirras, A., Blair, M. Collins, J. & Coulson, A. (1987) "Evidence that insect embryogenesis is regulated by ecdysteroids released from yolk proteins." P.N.A.S., Vol. 85, pp. 1554-1557.

Boyer, R.S. & Moore J.S. (1977), "A Fast String Searching Algorithm." Communications of the A.C.M., Vol. 20, No. 10, pp. 762-772

Brennan, M.D., Weiner, A.J., Goralski, T.J. & Mahowald, A.P. (1982) "The follicle cells are a major site of vitellogenin synthesis in *Drosophila melanogaster*." Develop. Biol., Vol. 89, pp 225-236.

Brock, D.J.H. & Haywood C. (1987), "The CF antigen binds two calcium ions with a similar affinity to S100 protein." Unpublished Data, M.R.C. Human Genetics Unit, Western General Hospital, Edinburgh.

Brown, N.L. (1985) "Bacterial resistance to mercury - *reductio ad absurdum*?" T.I.B.S., Vol. 10, No. 10, pp. 400-403.

Brown, W.B. (1986) "Integrating Distributed Array Processing into EMAS 2900." Software Practice and Experience., Vol. 16, pp. 517-529.

Brutlag, D.L., Clayton, C.J., Friedland, P. & Kedes, L. "SEQ: a nucleotide analysis and recombination system."(1982) Nucl.Acids Res., Vol. 10, pp. 279.

Bullock, S., Haywood, C., Manson, J.C., Brock, D.J.H. & Raeburn, J.A. (1982) "Elevated levels of Cf Ag in homozygotes and obligate heterozygotes." Clin. Genet., Vol 21, pp. 336-341.

Calabretta, B., Battini, R., Kaczmarek, L., de Riel, J.K., & Baserga, R. (1986) "Molecular Cloning of the cDNA for a Growth Factor-inducible Gene with Strong Homology to S-100, a Calcium-binding Protein." J. Biol. Chem., Vol. 261, pp. 12628-12632.

Clayton, C.J., Friedland, P. & Kedes, L., Brutlag, D.L. (1981) "SEQ - sequence analysis system Manual." Pub. Stanford Univ., Calif., USA.

Collins, J.F. & Coulson, A.F.W (1984) "Applications of parallel processing algorithms for DNA sequence analysis." Nucl. Acids. Res., Vol. 12, No. 1, pp. 181-192.

Collins, J.F. & Coulson, A.F.W. (1987) "Molecular Sequence Comparison and Alignment." In Nucleic Acid and Protein Sequence Analysis: A Practical Approach. Ed. Bishop, M. & Rawlings, C., I.R.L Press, Oxford.

Coulson, A.F.W. & Collins, J.F. (1986) "Biological Sequence Analysis: Advanced Architecture Computing Requirements." Report prepared under contract to Directorate XII of the Commission of the EEC.

Crick, F.H.C. (1958) "On protein synthesis." The Biological Replication of Macromolecules. XIIth Symposium of the Society for Experimental Biology. pp. 138-163., Pub. Cambridge University Press.

Date, C.J. (1980) "An Introduction to Data Processing." 4th Edition, Pub. Addison Wesley.

Dayhoff, M.O, Schwartz, R.M & Orcutt, B.C. (1978) "Atlas of Protein Sequence and Structure." Pub. NBRF, Washington.

de Wind, N., de Jong, M., Merjer, M. & Stuitje. D.R. (1985) "Site-directed mutagenesis of *Escherichia coli* chromosome near *oriC*: identification and characterization of *asnC*, a regulatory element in *E. coli* asparagine metabolism." N.A.R., Vol. 13, No. 24, pp. 8797.

DEC (1987) "VAX-11 Macro Assembler Language Programming." Digital Education Services Manual, EY-L089E-HO E001.

Devereux, J., Haeberli, P. & Smithies, O. (1984) "A comprehensive set of sequence analysis programs for the VAX." Nucl. Acids Res., Vol. 12, pp. 387-395.

Dickerson, R.E. & Geis, I. (1969) "The structure and action of proteins." Benjamin/Cummings Publishing Company.

Dodd, I.B. & Egan, J.B. (1987) "Systematic Method for the detection of Potential λ Cro-like DNA-binding Regions in Proteins." J. Mol. Biol., Vol. 194, pp. 557-564.

Donachie, W.D., Begg, K.J., Lutkenhaus, J.F., Salmond, G.P.C., Martinez-Salas, E. and Vincente, M. (1974) "Role of the *fts*A Gene Product in control of *Escherichia coli* cell division." Vol. 140, pp. 388-394.

Dorin, J.R., Novak, M., Hill, R.E., Brock, D.J.H., Secher, D.S. & van Heyningen, V. (1987) "A clue to the basic defect in cystic fibrosis from cloning the CF antigen gene." Nature, Vol. 326, pp. 614-616.

Drew, H.R., Wing, R.M., Takano, T., Broka, C., Tanaka, S., Itakura, K. & Dickerson, R.E. "Stucture of a B-DNA dodecamer: Conformation and dynamics." (1981) Proc. Nat. Acad. Sci. USA., Vol. 78, pp. 2179.

Du Praw, E.J. "DNA and Chromosomes" (1970) Pub. Holt Rinehart and Winston Inc.

Dumas, J.P. & Ninio, J. (1982) "Efficient algorithms for folding and comparing nucleic acid sequences." Nucl. Acids Res., Vol. 10, pp. 197-206.

Edman, P. & Begg, G. (1967) "A protein Sequenator." Eur. J. Biochem., Vol. 1. pp. 80-91.

Friedland, P., Kedes, L., Brutlag, D.L., Iwasaki, Y, & Bach, R. (1982) "GENESIS. A knowledge-based genetic engineering simulation system for representation of genetic data and experiment planning." Nucl. Acids Res., Vol. 10., No. 1, p. 323

Feller, W. (1968) "An Introduction to Probability Theory and its Applications." Pub. John Wiley.

Flanders, P.M., (1982) "Languages and Techniques for Parallel Array Processing." Ph.D. Thesis. Queen Mary College, University of London.

Fullmer, C.S. & Wasserman, R.H. (1981) "The Amino Acid Sequence of Bovine Intestinal Calcium Binding Protein." J. Biol. Chem., Vol. 256, pp. 5669-5674.

Garabedian, M.J., Shirras, A.D., Bownes, M. & Wensink, P. (1987) "The nucleotide sequence of the gene coding for Drosophila melanogaster yolk protein 3." Gene, Vol. 55, pp. 1-8.

Gay, N.J. & Walker, J.E. (1981) "The atp operon: nucleotide sequence of the promoter and the genes for the membrane proteins, and the ∂ subunit of *Escherichia coli* ATP-sythetase." Nucl. Acids Res., Vol. 9, pp. 3919-3926.

George, D.G., Barker, W.C. & Hunt, L. (198) "The Protein Identification Resource. (PIR)" Nucl. Acids Res., Vol. 14, pp. 11-15.

Gerke, V. & Weber, K. (1985) "The regulatory chain in the p36-kd substrate complex of viral tyrosine specific protein kinase is related in sequence to the S100 protein of glial cells." EMBO J., Vol. 4, pp. 2917-2920.

Gingeras, T.R. & Roberts, R.J. (1980) Science Vol. 209, pp. 1322-1328.

Glenney, J.R. & Tack, B.F. (1985) "Amino-terminal sequence of p36 and associated p10: Identification of the site of tyrosine phosphorylation and homology with S100." Proc. Nat. Acad. Sci. USA, Vol. 82, pp. 7884-7888.

Goad, W.B. & Kanehisa, M.I. (1982) "Pattern Recognition in Nucleic Acid Sequences I. A general method for finding Local Homologies and Symmetries." Nucl. Acids Res. Vol. 10, No 1 pp. 247-263.

Goad, W.B. & Kanehisa, M.I. (1982) "Pattern Recognition in Nucleic Acid Sequences II. An efficient method for finding Locally Stable Secondary Structure." Nucl. Acids Res. Vol. 10, No 1 pp. 264-278.

Gostik, R.W. (1979) "Software and algorithms for the Distributed-Array Processors." I.C.L. Technical J. May, pp. 116.

Gould, S.J. (1980) "Ever Since Darwin. Reflections in Natural History." Pelican Books.

Green, P.R. & Bell, R.M. (1984) "The Triose-phosphate site of homogenous reconstituted *sn*-Glycerol-3-phosphate acyltransferase of *Escherichia coli*." Biochimica et Biophysica Acta, Vol. 795, pp. 348-355.

Guidoni, A., Benkarka, F., De Caro, J. & Roveny, M. (1981) Biochem. Byophys. Acta. Vol. 660. pp148-150.

Hoare, C.A.R. (1961) "Quicksort." Communications of the A.C.M., Vol 4, No. 7 pp. 321

Hockney, R.W. & Jesshope C.R. (1981) "Parallel Computers: Architecture, pragramming & algorithms" Pub. BRISTOL: Adam Hilger.

Hung, M.C. & Wensink, P.C. (1983) "Sequence and structure conservation in yolk proteins and their genes." J. Mol. Biol., Vol. 164, pp. 481-492.

ICL (1982) "The ICL Distibuted Array Processor (DAP)." I.C.L. Sales Literature.

Intelligenetics. (1983) "Quest Manual." Intelligenetics Corp. USA.

Isobe, T & Okuyama, T. (1981) "The amino acid sequence of the α Subunit in Bovine S100a Protein." Eur. J. Biochem. Vol. 116, pp. 79-86

Ishii, S., Ihara, M., Maekawa, T., Nakamura, Y., Uchida, H. & Imamoto, F. (1984) "The nucleotide sequence of the cloned *nus*A gene and its flanking region of *Escherichia coli*." Nucl. Acids Res., Vol. 12, No. 7, pp. 3333-3342.

IUPAC-IUB (1966) Commission on Biochemical Nomenclature, "Abbreviated Designations of Amino Acid Derivatives and Peptides, Tentative Rules", J. Biol. Chem., Vol. 241, pp. 2491-2495.

IUPAC-IUB (1968) Commission on Biochemical Nomenclature, "A one-letter Notation for Amino Acid Sequences, Tentative Rules", J. Biol. Chem., Vol 243, pp. 3557-3559.

Jones, T.A. (1978) "A Graphics Model Building and Refinement System for Macromolecules." J. Appl. Cryst., Vol. 11, pp. 268-272.

Kabsch, W. & Sander, S. (1983) "Dictionary of Protein Secondary Stucture: Pattern Recognition of Hydrogen-Bonded and Geometrical Features." Biopolymers Vol. 22, pp. 2577-2637.

Knobeloch, D.W., Hildebrand, C.E., Moyzis, R.K., Longmire, J.L., Sirotkin, K.M. & Beugelsdijk, T.J. (1988) "Robotics in the Human Genome Project." BIOTECHNOLOGY, Vol. 5, pp. 1284-1287

Knuth D.E. (1973) "Sorting and Searching." The Art of Computer Programming. Vol 3, Pub. Addison-Wesley. Reading Mass.

Knuth, D.E., Morris, J.H. & Pratt, V.R. (1977) "Fast Pattern Matching in Strings." S.I.A.M J. Computing, Vol. 6, No. 2, pp. 323-350.

Kolling, R & Lother, H. (1985) "AsnC: an Autogenously Regulated Activator of Asparagine Synthetase A transcription in *Escherichia Coli*." J.Bact., Vol. 164, No.1, pp. 310-315

Korn, Queen & Wegman (1977) Proc. Nat. Acad. Sci. USA, Vol. 74, No. 10 pp.4401

Kornberg, A. (1974) "DNA synthesis." Freeman. p. 16.

Kretsinger, R.H. (1980) "Crystallographic Studies of Calmodulin and Homologs." Ann. NY Acad. Sci., Vol. 356, pp. 14-19.

Krustal, J.B. (1983) "An Overview of Sequence Comparison." Time Warps, String Edits and Macromolecules: The Theory and Practise of Sequence Comparison. Ed. Sankoff, D. & Krustal, J.B., Pub. Addison-Wesley, Chap 1, pp. 1-45.

Lagueux, M., Harry, P. & Hoffman, J.A. "Ecdysteroids are bound to vitellin in newly laid eggs of locusta." Mol. Cell Endocrinol., Vol. 24, PP.325-338.

Lehninger, A.L. (1975) "Biochemistry. The Molecular Basis of Cell Structure and Function." Second Edition. Worth Publishers Inc.

Lewin, B. (1980) "Gene Expression 2. Eucaryotic chromosomes". Second Edition. John Wiley & Sons. p. 962.

Lewin, R. (1987) "National Academy Looks at Human Genome Project, Sees Progress." Vol.234, No.4790, pp. 747-748.

Liljas, A. & Thirup, S. (1986) "Evolutionary aspects of Ribosome-factor interactions." Chemica Scripta, Vol. 26B, pp. 109-119.

Lipman, D.J. & Pearson, W.R. (1985) "Rapid and Sensitive Protein Similarity Searches. (FASTP)" Science. Vol. 227. pp. 1435-1440.

Lipman, D.J., Wilbur, W.J., Smith, T.F. & Waterman, M.S. (1984) N.A.R., Vol. 1, No. 1A, pp. 215.

Luiten, R.G.M., Putterman, D.G., Schoenmakers, J.G.G., Konongs, R.N.H. & Day, L.A. (1985) "Nucleotide Sequence of the Genome of Pf3, an Inc-1 Plasmid-Specific Filamentous Bacteriophage of *Pseudomonas aeruginosa*." J. Virol., Vol. 56, No. 1, pp. 268-276.

Lyall, A. (1982) "Expert system like tools for research in molecular biology." MSc Thesis, Dept. of Computing Science, Imperial College of Science and Technology, London.

Martinez, H.M. (1983) "An efficient method for finding repeats in molecular sequences." Nucl. Acids Res., Vol. 11, p. 4629.

Maxam, A.M. & Gilbert, W. (1977) "A new method for sequencing DNA." Proc. Nat. Acad. Sci. Vol. 74, pp. 560-564.

Michaelson, S. (1987) "The Dynamic Programming Method." Personal Communication, Department of Computing Science, Edinburgh University.

Minard, P., Walker, P., Bowen, D., Davies, G., Littlechild, J., Hall, L. & Watson, H.C. (1987) "The use of site-directed mutagenesis to study the glycolytic enzyme phosphoglycerate kinase." Protein Engineering, Vol. 1, pp. 260

Misra, T.K., Brown, N.L., Fritzinger, D.C., Pridmore, R.D., Barnes, W.B., Haberstroh, L., & Silver, S. (1984) "Mercuric ion-resistance operons of plasmid R100 and transposon Tn*501*: The beginning of the operon including the regulatory region and the first two structural genes." Proc. Nat. Acad. Sci. USA, Vol. 81, pp. 5975-5979.

Misra, T.K., Brown, N.L., Haberstrohl, L., Schmidt, A., Goddette, D. & Silver, S. (1985) "Mercuric reductase structural genes from plasmid R100 and transposon Tn*501*: Functional domains of the enzyme." Gene. Vol. 34, pp. 253-262.

Nakamura, M., Yamada, M., Hirota, Y., Sugimoto, K. Oka, A. & Takanami, M. (1981) "Nucleotide sequence of the *asn*A gene coding for asparagine synthetase of *E.coli* K-12." Nucl. Acids Res., Vol 9, No. 18, pp. 4669-4676.

Morowitz, H.J. (1985) "Models for Biomedical Research: A New Perspective." Report of Committee under the Auspices of the NIH. National Academy of Sciences Press, Washington.

Needleman, S.B. & Wunsch, C.D. (1970) "A General Method Applicable to the Search for Similarities in the Amino Acid Sequence of Two Proteins." J. Mol. Biol. Vol. 48 pp. 443.

Ni'Bhriain, N. & Foster, T.J. (1986) "Polypeptides specified by the mercuric resistance (*mer*) operon of plasmid R100." Gene, Vol. 42, pp. 323-330.

Niall, H., (1977) "Automated Methods for protein sequencing." The proteins. Ed. Neurath, H. & Hill, R.L., Pub. Accademic Press. Vol 3, pp. 179-238.

Orgel, L.E. (1968) "Evolution of the Genetic Apparatus" J. Mol. Biol., Vol. 38, pp. 381-393.

Pabo, C.O. & Sauer, R.T. (1984) "Protein-DNA Recognition." Ann. Rev. Biochem., Vol. 53, pp 293-321.

Palca, J. (1987) "Interest in the human genome project reaches new heights." Nature, Vol. 325, p. 651.

Paulson, J.R. & Laemmli, U.K. (1977) "The structure of histone depleted metaphase chromosomes." Cell. Vol. 12, pp. 817-828.

Peattie, D.A. (1979) P.N.A.S., "Direct chemical methods for sequencing RNA." Vol. 76, pp. 1760-1764

Pedan, K.W.C. (1983) "Revised sequence of the tetracycline-resistance gene of PBR322." Gene. Vol. 22, pp. 277-280.

Reddaway S.F. & Flanders P.M. (1982) "Sorting on DAP." The ICL Technical J., Vol 4.

Reddaway S.F. (1984) "Distributed Array Processor, Architecture and Performance." NATO Advanced Studies Institute Series. Vol. F7, High Speed Computation, Ed. Kowalik, J.S., Spinger Verlag.

Reeck, G.R., de Haen, C., Teller, D.C., Doolittle, R.F., Fitch, W.M., Dickerson, R.E., Chambon, P., McLachlan, A.D., Margoliash, E., Jukes, T.H. & Zuckerkandl, E. (1987) "Homology in Proteins and Nucleic Acids: A Terminology Muddle and a Way out of It." Cell, Vol. 50, pp. 667.

Rees, A.R. & Sternberg, M.J.E. (1984) "From Cells to Atoms. An illustrated introduction to Molecular Biology." Blackwell Scientific Publications.

Robinson, A.C., Collins, J.F. & Donachie, W.D. (1987) "Prokaryotic and Eukaryotic Cell-cycle Proteins" Nature, Vol. 328, pp 766.

Robinson, A.C., Kenan, D.J., Hatfull, G.F., Sullivan, N.F., Spiegelberg, R. & Donachie, W.D. (1984) "DNA Sequence and Transcriptional Organisation of Essential Cell Division Genes *ftsQ* and *ftsA* of *Escherichia coli*: Evidence for Overlapping Transcriptional Units." J. Bact., Vol. 160, pp. 546-555.

Sanger, F., Nicklen, S. & Coulson, A.R. (1977) "DNA sequencing with chain-terminating inhibitors." Proc. Nat. Acad. Sci. USA, Vol. 74, pp. 5463-5467.

Savageau, M.A. (1977) "Design of molecular control mechanisms and the demand for gene expression." Proc. Nat. Acad. Sci. USA, Vol. 74, pp 5647-5651.

Sedgewick, R. (1983) "Algorithms." Pub. Addison Wesley.

Sellers, P.H. (1974) "On the Theory and Computation of Evolutionary Distances." SIAM J. Appl. Math. Vol. 26, pp. 787-793.

Seno, E.T & Chater, K.F. (1983) "Glycerol Catabolic Enzymes and Their Regulation in Wild-type and Mutant Strains of *Streptomyces coelicolor* A3(2)." J. Gen. Microbiol. Vol. 129, pp 1403-1413.

Skarzynski, T., Moody, P.C.E. & Wonacott, A.J. (1987) "Structure of holo-Glyceraldehyde-3-phosphate Dehydrogenase from *Bacillus stearothermophilus* at 1.8 Å Resolution." J. Mol. Biol., Vol. 193, pp. 171-187.

Shaw, D.J., Rice, D.W. & Guest, J.R. (1983) "Homology between CAP and Fnr, a Regulator of Anaerobic Respiration in *Escherichia coli*." J. Mol. Biol., Vol 166, pp. 241-247.

Simoncsits, A., Brownlee, G.G., Brown, R.S. Rubin, J.R. & Guilley, H. (1977) "A new rapid gel sequencing method for RNA." Nature. Vol 269 pp. 833-836

Smith, A.J.H. (1979) "The use of exonuclease III for preparing single stranded DNA for use as a template in the chain terminator sequencing method." N.A.R. Vol. 6, No. 3, pp. 831.

Smith, C.P. & Chater, K.F. (1988) "Structure and Regulation of Controlling Sequences for the *Streptomyces coelicolor* Glycerol Operon." J.M.B. In Press.

Smith, C.P. (1986) "Molecular Biology of the glycerol utilization operon of *Streptomyces coelicolor* A3(2)." Ph.D. Thesis, University of East Anglia, Norwich, U.K.

Smith, C.P. (1988) "A glycerol non-utilizing mutant of *Streptomyces coelicolor gylR* contains a mis-sense mutation." Personal Communication, Department of Biochemistry, University of Bristol, U.K.

Smith, L.M., Sanders, J.Z., Kaiser, R.J., Hughs, P., Dodd, C., Connell, C.R., Heiner, C., Kent, S.B.H. & Hood, L.E. (1986) "Fluorescence Detection in Automated DNA Sequence Analysis." Vol. 321, pp. 674.

Smith, T.F. & Waterman, M.S. (1981) "Comparison of Biosequences." Advances in Applied Mathematics. Vol. 2, pp. 482-489

Smith, T.F., Waterman, M.S. & Burks, C. (1985) "The statistical distribution of nucleic acid similarities." N.A.R., Vol. 13, No. 2, pp. 645-656.

Sneath, P.H.A. & Sokal, R.R. (1973) "Numerical Taxonomy: the principles and practice of numerical classification." 2nd edition, Pub. San Francisco. W.H. Freeman.

Stephens, P.D. & Yarwood J.K. (1986) "Providing Multi-User Access to Distributed Array Processors." Software Practice and Experience. Vol. 16 pp. 531-539.

Stryer, L. (1981) "Biochemistry" Second Edition. W.H. Freeman and Company, San Francisco.

Summers, A.O. (1986) "Organization, expression and evolution of Genes for Mercury Resistance." Ann. Rev. Microbiol., Vol. 40, pp. 607-634

Van Eldrick, L.J., Zendegui, J.G., Marshak, D.R. & Watterson, D.M. (1982) "A class of proteins that transduce biological calcium signals." Int. Rev. Cytol., Vol. 77, pp. 1-61.

Wada, A. (1986) "Japanese super DNA sequencer project." University of Tokyo Report.

Wagner, R.A. & Fischer, M.J. (1974) "The string-to-string correction problem." J. Ass. Comput. Mach. Vol. 21, pp. 168-173.

Walker, J.E., Carne, A.F., Runswick, M.J., Bridgen, J. & Harris, J.I. (1980) "D-Glyceraldehyde-3-Phosphate Dehydrogenase. Complete Amino-Acid Sequence of the Enzyme from *Bacillus stearothermophilus*." Eur. J. Biochem. Vol. 108, pp. 549-565.

Watson, H.C., Walker, N.P.C., Shaw, P.J., Bryant, T.N., Wendell, P.L., Fothergill, L.A., Perkins, R.E., Conroy, S.C., Dobson, M.J., Tuite, M.F., Kingsman, A.J. and Kingsman, S.M. "Sequence and structure of Phosphoglycerate kinase." (1982) EMBO J., Vol. 1. pp.1635

Welsh, M.J. & Liedtke, C.M. (1986) "Chloride and potassium channels in cystic fibrosis airway epithelia." Nature, Vol. 322, pp. 467-470.

Wigely, D.B., Lyall, A., Hart, K.W. & Holbrook, J.J. (1987) "The greater strength of Arginine Carboxylate over Lysine Carboxylate ion pairs. Implications for the design of novel enzymes and drugs." Biochem. Biophys. Res. Comm., Vol. 149, No. 3., pp. 927-929

Wilbur, W.J. & Lipman, D.J. (1983) "Rapid similarity searches of nucleic acid and protein data banks." Proc. Nat. Acad. Sci. USA. Vol. 80, pp. 726-730.

Williams, J.C., Steiner, L.A., Ogden, R.C., Simon, M.I., & Feher, G. (1983) "Primary structure of the M subunit of the reaction centre from *Rhodopseudomonas sphaeroides.*" P.N.A.S., Vol. 80, pp. 6505-6509.

Williams, J.W.J., (1964) "The heapsort algorithm." Communications of the A.C.M., Vol. 7, pp. 347-348.

Winter, G., & Fersht A.R. (1984) "Engineering enzymes." Trends in Biotechnology, Vol. 2, No. 5, pp. 115-119.

Wong, C.K. & Chandre, A.K. (1976) "Bounds for the string editing problem" Communications of the A.C.M, Vol 23, pp. 13-16.

APPENDIX A

Listings

This appendix contains five listings of the alignments that are referred to in the discussion in chapter eight. Each listing is preceded by a table summarizing all the results for that particular programme run. The entire section is preceded by a key. In all the alignments the upper sequence is from the database being searched and the lower is from the query sequence.

Key to the interpretation of listings

Item.	Meaning.
>	This precedes the identifier from the NBRF protein database. The identifier uniquely identifies the data base entry. It is followed by a short title from the database.
No.	The number of the alignment in the listing.
Score.	The sum of all the values for the pair-wise comparisons present in the alignment according to the similarity figure used.
EXPECTED.	The number of alignments of this score that would be expected to occur by chance calculated in the way described in the text.
PREDICTED	same as EXPECTED.
OBSERVED	Number of results found for each score
Quality.	The score as a percentage of the maximum possible score for the portion of query sequence in the alignment.
Hits.	The number of perfect matches in the alignment.
MISMATCH.	The number of mismatches in the alignment.
INDEL.	The number of letters that are aligned opposite a space. These may be considered as insertions in one sequence or deletions in the other and are referred to as indels.
'*'	A star above a pair of letters indicates that they are the same.
'.'	A dot above a pair of letters indicates that they contribute positively to the score but are not the same. Pairs of letters with which this is used will vary according to the similarity table used.
' '	A space above a pair of letters indicated that they do not contribute or contribute negatively to the score. Pairs of letters with which this is used will vary according to the similarity table used.

Summary of Results for MerC from plasmid R100.

Score	Predicted	Observed	Score	Predicted	Observed
46	512.7	542	69	12.3	10
47	436.0	448	70	10.4	14
48	370.7	399	71	8.9	7
49	315.3	309	72	7.5	12
50	268.1	230	73	6.4	8
51	228.0	198	74	5.4	9
52	193.8	205	75	4.6	7
53	164.8	139	76	3.9	12
54	140.2	168	77	3.3	5
55	119.2	136	78	2.8	4
56	101.3	124	79	2.4	1
57	86.1	89	80	2.0	4
58	73.2	57	81	1.7	4
59	62.3	71	82	1.4	3
60	52.9	46	83	1.2	1
61	45.0	75	84	1.0	3
62	38.3	32	85	0.9	1
63	32.5	33	86	0.7	3
64	27.7	31	87	0.6	4
65	23.5	17	91	0.3	2
66	20.0	20	94	0.2	2
67	17.0	16	95	0.1	1
68	14.4	8	97	0.1	2

Administrative data from program run.

```
V11P100 - 100 pams
indel 10
MERC USED AS QUERY SEQUENCE, LENGTH   140 START  1 END
140
NBRF11 and new seqeunces appended; 25-3-87 15:40pm
  Proteins 4612 Residues 1066790
 MCDAP_NW3OP VERSION OF 08/07/86
 TOTAL RESULTS 3512
 THRESHOLD VALUE SET INITIALLY WAS   10
 FINAL THRESHOLD VALUE IS   46
 GEN. PENALTY ADDED TO NEGATIVE SCORES FROM FIGURE: 0
 MAXIMUM SCORE   97

A 13.6; SIGA 0.3 B -0.1621; SIGB 0.005 CHI2 1.4 HIGH 73
```

Listing One. MerC from Plasmid R100.

```
>P1;QQPSHT   Hypothetical protein merT (transposon Tn501) - Pseudomonas aeruginosa
No. 1.1 Score = 97  Quality = 23.775 HITS = 36 Mis. = 8 INDELS = 10 EXPECTED NO.  0.13E+00
     **.*..... .*  *  *  * * .*..* ** *. *  *  .**.  *..**
  13 TGGLAAILAS AC  CLGPLVLIALGFSGAWIGNLAVLEPYRPIFIGVALVALF        63
  11 TGALGSVVSAMGCAACF P AL A SFGAA IG LGFLSQYEGLFIS RLLPLF        57

>P1;YTEC32   Tetracycline resistance protein - Escherichia coli plasmid pBR322
No. 1.2 Score = 97  Quality = 23.716 HITS = 33 Mis. = 11 INDELS = 8 EXPECTED NO.  0.13E+00
     **.*.* . ** *    **. .*..*   * **** * **. .. * *.*
  48 LLALYALMQFLCAPVLGALSDRFGRRPVLLASLLGATIDYAIMATTPVLWIL        99
  53 LLPLFAALAFL A   NALG WFSHRQ WLRSLLG MIGPAIVFAATV WLL        96

>P1;WNRFMS   Reaction centre protein M chain - Rhodopseudomonas sphaeroides
No. 1.3 Score = 95  Quality = 27.457 HITS = 24 Mis. = 9 INDELS = 5 EXPECTED NO.  0.18E+00
     ****.. * * .. ** .*   .*.. .*   .* * *
  40 LGWFGNAQ LGPIYLGSLGVLSLFSGLMWFFTIGIWFW        76
  67 LGWFSHRQWLRSL LGMIGPAIVFAATVWL  LGNW W       100

>P1;QQEBHT   Hypothetical protein merT - Shigella flexneri plasmid R100
No. 1.4 Score = 94  Quality = 23.039 HITS = 34 Mis. = 12 INDELS = 6 EXPECTED NO.  0.21E+00
     .*.*.....  * ..    **. *..* ** *  *  *  *** . *..**
  13 AGGLAAILASTCCLGPLVLVALG FSGAWIGNLTVLEPYRPLFIGAALVALF        63
  11 TGALGSVVSAMGC AA CFPALASFGAA IG LGFLSQYEGLFI SRLLPLF        57

>P1;QQADB5   Hypothetical protein C-119 - Adenovirus 5
No. 1.5 Score = 94  Quality = 34.815 HITS = 16 Mis. = 11 INDELS = 4 EXPECTED NO.  0.21E+00
     * . *   *  ** **  .  **.* * .*
  46 PQVSAFVNN WDNLGMWWFSIALMFVCLIIM        75
  85 PAI VFAATVW LLGNWW TANLMYVGLALM       112

>P1;VCLJA2   env polyprotein precursor - AIDS virus ARV-2 (AIDS-associated retrovirus)
No. 1.6 Score = 91  Quality = 19.078 HITS = 30 Mis. = 14 INDELS = 10 EXPECTED NO.  0.34E+00
     ** *. ** ** .*.. .. . .   * *  **.  * * .   *. . ** *
 765 FSYRR LRDLL LIAARTVEILGHRGWEALKYWWSL LQYWI QELKNSAVS W       813
  70 FSHRQWLRSLLGMIGP AI VFAATVW LLGNWWTANLMY VGLALMIG VSIW       118

>P1;QQBE50   Hypothetical BNLF1 protein - Epstein-Barr virus (strain B95-8)
No. 1.7 Score = 91  Quality = 13.481 HITS = 39 Mis. = 28 INDELS = 10 EXPECTED NO.  0.34E+00
     *** . ** * .   .* . * *   .   * * .  *.     *.    .. *  ****   .*. . * * ... **
 111 LFIFGCLLVL GIWIYLLEML WRLGATIWQLLAFFLAFFLDLILLIIALYLQQNWWTLLVDLLWL L LFLAILIW  183
  48 LFI SRLLPLFAALAFLANALGW FSHRQW LRSL LGMIGPAIVFAATVWLLGNWWT  ANLMYVGLALMIGVSIW  118

>P1;VCVWEK   env polyprotein - AKV murine leukemia virus
No. 1.8 Score = 87  Quality = 18.432 HITS = 28 Mis. = 18 INDELS = 7 EXPECTED NO.  0.66E+00
     ** *. *.**   * ** *  . *. *. .      .. *   **.   *. .*
 229 ATSWVTGHWWGLRL YVSGHDPGLIFGIRL KITDSGPRVPIGPNPV LSDRR       278
  91 ATVWLLGNWWTANLMYV GL A LMIGVSIWDFVSPAHR RCGPDGCELPAKR       139

>F1;UIHU     Thyroglobulin precursor - Human (fragment)
No. 1.9 Score = 87  Quality = 25.072 HITS = 27 Mis. = 12 INDELS = 3 EXPECTED NO.  0.66E+00
     *...***  * *   .* ..* *.. ** *   * ** .*..
 547 PTVGSFGFEINLQ ENQ NALKFLASLLELPEFLLFLQHAIS       586
  28 PALASFGAAIGLGFLSQYEGL FISRLLPLFAALAFLANALG        68

>P1;CBMS     Cytochrome b - Mouse mitochondrion (SGC1)
No. 1.10 Score = 87  Quality = 17.864 HITS = 32 Mis. = 15 INDELS = 8 EXPECTED NO.  0.66E+00
     *. ** *** .*  *..  * * ..   *  ...  . .** . .*  *. .* *
 115 IGVLL LFAVMATAFMGYVLPW GQMSFWGATVITNLLSAIPYIGTT LV EWIW       165
  50 ISRLLPLFAALA  FLANALGWFSHRQ WLRSLLGMIGPAIVFAATVWLLGNW W       100

>P1;QQSABT   Hypothetical protein B-295 - Staphylococcus aureus plasmid pT181
No. 1.11 Score = 87 Quality = 9.265 HITS = 57 MISMATCHES = 42 INDELS = 8 EXPECTED NO. 0.66
       * .* ..* ** **.*    *  .   *  . **. ...*  .*.  ..  .. . *.*
 109 GRLVQGVGSAA FPSLIMVVVARNITRKKQGKAFGFIGSIVALGEGLGPSIGGIIAHYIHWSYL  171
  15 GSVVSAMGCAACFPALASFGAAIGLGFLSQYEGL FISRLLPLFAALA FLANALGWFSHRQWL   88

         .* **   .* *   * . *.  * * *  **. ** ..**
  172 L ILPMITIVTIPFLIKVMVPGKS TKNTLDIVGIVLM SISI 211
   87 RSLLGMIG PAIVFAATVWLLGNWWTAN LMYVGLALMIGVSI 117
```

```
>P1;LBRRB    Light-harvesting protein, beta chain - Rhodospirillum rubrum
No. 1.12 Score = 86  Quality = 27.564 HITS = 18 Mis. = 13 INDELS = 2 EXPECTED NO.  0.78E+00
      ..  .* * .*  *.  ** *. * *   * *.
  17  EFHKIFTSSILVFFGVAAF AHLLVWIW RPWV        47
  44  QYEGLFISRLLPLFAALAFLANALGWFSHRQWL        76

>P1;SAVLVD   Probable major surface antigen precursor - Hepatitis B virus (two subtypes)
No. 1.13 Score = 86  Quality = 19.816 HITS = 27 Mis. = 17 INDELS = 5 EXPECTED NO.  0.78E+00
      .*. .. ** *. *  * * *  ** ***       *  ..**** . *
 324  IPIPSSWAF AKYLWEWASVRFSWL SLLVPFVQWFVGLSPTVWLSAIW       370
  54  LPLFAALAFLANAL GWFSHR QWLRSLLGMIGPAIV FAATVWLLGNW        99

>F1;FOMVS    p15E protein - Simian sarcoma virus (fragment)
No. 1.14 Score = 86  Quality = 46.486 HITS = 12 Mis. = 6 INDELS = 1 EXPECTED NO.  0.78E+00
      ***.  *  .**. * **
  12  GWFNSSPWFTTLLSTIAGP        30
  68  GWFSHRQWLRSLLGMI GP        85

>F1;PHECGM   Maltodextrin phosphorylase (EC 2.4.1.1) - Escherichia coli (fragment)
No. 1.15 Score = 85  Quality = 26.646 HITS = 28 Mis. = 5 INDELS = 8 EXPECTED NO.  0.92E+00
      ***.  ...*  ****  ..*. * .* * * *. ** ***
 110  ALGN  GGLGRLAACFLDSMATVGQSATGYG LNYQY GLF       146
  13  ALGSVVSAMGC AACF PALASFG AAIGLGFLS QYEGLF        49

>P1;LWECA    ATPase (EC 3.6.1.34), lipid-binding protein (c chain) - Escherichia coli
No. 1.16 Score = 84  Quality = 20.388 HITS = 29 Mis. = 17 INDELS = 4 EXPECTED NO.  0.10E+01
      *. **   .**. *****.* *.  . **      *.**  .  *.  .*
  11  MA AAVMMGLAAIGAAIGIGILGGKFLEGAARQPDLIPLLRTQFFIVMGL        59
  21  MGCAACFPALASFGAAIGLGFLS QY EGLFISR LLPLFAALAFLANAL        67

>P1;QXASM4   Hypothetical protein 4 - Aspergillus amstelodami mitochondrion (SGC3)
No. 1.17 Score = 84  Quality = 30.000 HITS = 25 Mis. = 7 INDELS = 3 EXPECTED NO.  0.10E+01
      .*.. *... * *.... .*. .  *. ** . **
 379  SLGNSGTPLTLNFIGEFMSLYGVFERM PLLGVLA       412
  29  ALASFGAAIGLGFLSQYEGLF I SRLLPLFAALA        61

>P1;QQBE1    Probable membrane antigen p140 - Epstein-Barr virus (strain B95-8)
No. 1.18 Score = 84  Quality = 40.000 HITS = 17 Mis. = 6 INDELS = 1 EXPECTED NO.  0.10E+01
      **. *  * **. .  .*..***.
 288  LFMRRQHPGLFPFVNAIASSLGWY       311
  48  LFISRLLP LFAALAFLANALGWF        70

>P1;PWFF6    ATPase (EC 3.6.1.34), protein 6 - Fruit fly mitochondrion (SGC4)
No. 1.19 Score = 83  Quality = 23.714 HITS = 22 Mis. = 11 INDELS = 5 EXPECTED NO.  0.12E+01
      .. * ..* .**    .* * *   * .** ** *.*
   2  MTNLFSVFDPLAIFNFSLNWLS T FL GLL MI PSI        34
  50  ISRLLPLFAALAFLANALGWFSHRQWLRSLLGMIGPAI        87

>P1;O4BOC2   Cytochrome P450(C21), steroid 21-hydroxylase, liver - Bovine
No. 1.20 Score = 82  Quality = 23.631 HITS = 28 Mis. = 10 INDELS = 6 EXPECTED NO.  0.14E+01
      .*** ** .* . **  *.. * **.  ** *  * . *. ..*
 416  SALA FGCGARVCLGESLARLE LFVV LLRLLQAFTLLPPPVG       456
  28  PALASFG AA IGLG FLSQYEGLFISRLLPLFAALAFLANALG        68

>P1;OTHU3 Cytochrome c oxidase (EC 1.9.3.1), polypeptide III - Human mitochondrion (SGC1)
No. 1.21 Score = 82  Quality = 29.078 HITS = 23 Mis. = 4 INDELS = 6 EXPECTED NO.  0.14E+01
      ** .** * * ...*. ** * . . .** *.*
  17  PLTGALSALLMTSGLAMWF HFHSM TLL MLG        46
  55  PLFAAL AFL ANALG WFSHRQWLRSLLGMIG        84

>P1;LWBOA    ATPase (EC 3.6.1.34), lipid-binding protein - Bovine
No. 1.22 Score = 82  Quality = 16.335 HITS = 40 Mis. = 13 INDELS = 6 EXPECTED NO.  0.14E+01
      .*.  ...* *.  ..... **. * .*. ..  .*   .* . .* *.* *..*.* *
   9  IGAGAATVGVAGSGAGIGTVFGSLI IGY ARNPSL KQQLFS YAILGFALSEAMGLF        63
  14  LGSVVSAMGCAACFPALAS FGAAIGLGFLSQYEGLFISRLLPLFAALAF LANALGWF        70

>F1;QXAS4M   Hypothetical protein 4 - Aspergillus nidulans mitochondrion (fragment) (SGC3)
No. 1.23 Score = 81  Quality = 28.929 HITS = 25 Mis. = 7 INDELS = 3 EXPECTED NO.  0.17E+01
      **.. *... * *.... .*. .  *. *. . **
 112  ALGNSGTPLTLNFIGEFMSLYGVFERM PILGVLA       145
  29  ALASFGAAIGLGFLSQYEGLF I SRLLPLFAALA        61
```

```
>P1;ZPECL    Lipoprotein signal peptidase (EC 3.4.22.-) - Escherichia coli
No. 1.24 Score = 81  Quality = 15.056 HITS = 35 Mis. = 16 INDELS = 6 EXPECTED NO.  0.17E+01
      ** .***.. **  .* *  . .. ** ... *  .   ..  * .*  .. **.**
  56  AAFSFLADSGGW  QR WFFAGIA IGISVILAVMMYR SKA TQKLNNIAYALIIG       106
  58  AALAFLANALGWFSHRQWLRSLLGMIGPAIVFAATVWLLGNWWTANLMYVGLALMIG       114

>P1;IMBPBL   rexB gene protein - Bacteriophage lambda
No. 1.25 Score = 81  Quality = 26.045 HITS = 25 Mis. = 7 INDELS = 6 EXPECTED NO.  0.17E+01
      *.. **.*   ***  ** * *  *. .* . ***.**
  31  PGV SFSAHRDGLGATLSSYAGTMIA IL I AALTFL        64
  28  PALASFGAAI GLG FLSQYEGLFISRLLPLFAALAFL        63

>P1;PHRBG    Glycogen phosphorylase (EC 2.4.1.1) - Rabbit
No. 1.26 Score = 81  Quality = 22.192 HITS = 31 Mis. = 7 INDELS = 9 EXPECTED NO.  0.17E+01
      * . .* **.  ...*  ****  ..*. * ** * * . .** *.*
 125  IEEDAG LGN  GGLGRLAACFLDSMATLGLAAYGYG I RYEFGIF       166
   7  IADKTGALGSVVSAMGC AACF PALASFG AAIGLGFLSQYE GLF        49

>P1;LWZMA    ATPase (EC 3.6.1.34), lipid-binding protein - Maize mitochondrion
No. 1.27 Score = 80 Quality = 18.52 HITS = 33 Mis. = 15 INDELS = 6 EXPECTED NO. 1. 0.20E+01
      ..*.*.*. .   *  *.**  *** ** *** *..  . * . ...   .  *
  23  GIGNVLSSSIHSVARNPSLAKQSFGYAI LGFALTEAIASF APMMAFLISFVF        74
  13  ALGSVVSA MGCAACFPALA  SFGAAIGLGF LSQYEGLFISRLLPLFAALAF        62

>P1;VCVWGF   env polyprotein - Feline leukemia virus (strain Gardner-Arnstein)
No. 28 Score = 80  Quality = 38.462 HITS = 14 Mis. = 7 INDELS = 1 EXPECTED NO. 1. 0.20E+01
      ***.   *  .*.. * ** ..
 595  GWFNKSPWFTTLISSIMGPLLI       616
  68  GWFSHRQWLRSLLGMI GPAIV        88

>F1;QXRT6M   Hypothetical protein 6 - Rat mitochondrion (fragment) (SGC1)
No. 29 Score = 80  Quality = 23.810 HITS = 25 Mis. = 10 INDELS = 7 EXPECTED NO. 1. 0.20E+01
      *. * .**  ** .*   *. **.. .**  **  *  *. .
  28  GGFGLIVS  GCIGCLMVLG FGGSFLGLMVFLI YLGGMLV        65
  12  GALGSVVSAMGCAACFPALASFGAA IGL GFLSQY EGLFI        50

>P1;OTBO3 Cytochrome c oxidase (EC 1.9.3.1), polypeptide III - Bovine mitochondrion (SGC1)
No. 30 Score = 80  Quality = 28.369 HITS = 21 Mis. = 6 INDELS = 6 EXPECTED NO. 1. 0.20E+01
      ** .** * * ...*  ** *   . .** ***
  17  PLTGALSALLMTSGLTMWF HFNSM TLL MIG        46
  55  PLFAAL AFL ANALG WFSHRQWLRSLLGMIG        84

>P1;MFIV1    Matrix (M1) protein - Influenza B virus (strain B/Lee/40)
No. 31 Score = 79  Quality = 25.649 HITS = 22 Mis. = 11 INDELS = 2 EXPECTED NO. 1. 0.24E+01
      *..  * * .** *  ... *  .  ..**..* *
  32  FGGKEFDLDSALEWIKNKRCLTDIQKALIGASICF        66
  57  FAALAF LANALGWFSHRQWLRSLL GMIGPAIVF        89

>P1;SYECR    Threonine synthase (EC 4.2.99.2) - Escherichia coli
No. 32 Score = 78  Quality = 30.11 HITS = 20 Mis. = 8 INDELS = 2 EXPECTED NO. 1. 0.28E+01
      **. *.  * *.  .*** .. ** *. *.
  14  SFAQAVTQG LGKNQGLFFPHDLPEFS LT        41
  32  SFGAAIGLGFLSQYEGLFISRLLPLFAALA        61

>P1;TVFVR Kinase-related transforming protein (src) (EC 2.7.1.-) - Rous sarcoma virus (str
No. 33 Score = 78  Quality = 21.972 HITS = 22 Mis. = 10 INDELS = 5 EXPECTED NO. 1. 0.28E+01
      **.  .*   *. ** * ...* .  *.  **. * *
 255  LAKD AWEIPRESLR LEAKLGQG CFGE VWM GTW       286
  63  LANALGWFSHRQWLRSLLGMIGPAIVFAATVWLLGNW        99
```

Summary of Results for *Streptomyces coelicolor gylR*.

Score	Predicted	Observed	Score	Predicted	Observed
49	615.1	608	69	21.82	18
50	520.6	510	70	18.46	20
51	440.5	442	71	15.63	22
52	372.8	352	72	13.22	10
53	315.5	306	73	11.19	11
54	267.0	229	74	9.4	14
55	225.9	235	75	8.0	8
56	191.2	204	76	6.7	10
57	161.8	171	77	5.7	1
58	136.9	155	78	4.8	9
59	115.9	121	79	4.1	6
60	98.04	97	80	3.4	2
61	82.96	73	81	2.9	4
62	70.21	76	83	2.1	2
63	59.41	66	84	1.7	2
64	50.28	56	85	1.5	2
65	42.55	42	86	1.2	3
66	36.00	43	88	0.91	1
67	30.47	21	89	0.77	1
68	25.78	27	90	0.65	1
			96	0.24	1

Administrative data from program run.

```
 SOURCE.LYALL
    0 READ; FIRST LAYER SET TO    1
    0 READ; LAST LAYER SET TO  275
    1 TO  254; LENGTH  254
100 Pam
Indel 10

 SOURCE.LYALL          USED AS QUERY SEQUENCE, LENGTH
254
USING NBRF PROTEIN SEQUENCE DATABASE VERSION 12
  Proteins 4750 Residues 1108197
 EDINBURGH DAP VERSION OF 08/07/86
 TOTAL RESULTS 3982
 THRESHOLD VALUE SET INITIALLY WAS   10
 FINAL THRESHOLD VALUE IS   49
 MAXIMUM SCORE   96

A 14.6; SIGA 0.3 B -0.2; SIGB 0.005 CHI2 0.6 HIGH 72
```

Listing Two. *Streptomyces coelicolor gylR.*

```
>P1;RKAIL7 Ribulose bisphosphate carboxylase (EC 4.1.1.39) large chain precursor - Anabaena
No. 2.1 Score = 96  Quality = 21.381 HITS = 32 Mis. = 19 INDELS = 4 EXPECTED NO.  0.24E+00
        * *. *** .**.. . . . . *   .* . * .  * .* .**** . * *
    313  RVLAKALRL SGGDH IHTGTVVGKLEGERGITMGFV DLLRENYVEQDKSRGIY       364
     10  RAAAM LRLLAGGERRLGLSDIASSLGLAKGTAHGILRTLQQEGFVEQDDASGRY        63

>P1;ATSY1    Actin 1 - Soybean
No. 2.2 Score = 90  Quality = 29.605 HITS = 28 Mis. = 7 INDELS = 2 EXPECTED NO.  0.65E+00
        *...* .*  .. .....* .**. *.  *..*. **
    316  MSKEISALAPSSMKIKVVAPSERKFGVW IGGSI LA       350
      1  MARNIQSLERAAAMLRLLAGGERRLGLSDIASSLGLA        37

>F1;FWSYG2   Glycinin, A2B1a chain - Soybean (fragment)
No. 2.3 Score = 89  Quality = 25.284 HITS = 24 Mis. = 13 INDELS = 3 EXPECTED NO.  0.77E+00
        . * .****  . .   .** *  * *.. ***** . *
    111  IYALNGRALV  Q VVNCNGERVFDGELQEGGVLIVPQNF       147
     79  VHELRARALVWTDDLARSSGESVHLGVLHQQGVLIVHHVF       118

>P1;HGMQP    Hemoglobin gamma chain - Pig-tailed macaque
No. 2.4 Score = 88  Quality = 33.333 HITS = 24 Mis. = 6 INDELS = 3 EXPECTED NO.  0.91E+00
        ** * * .. ....*. .*.  .*..*...  *
    113  VLAI RFGKEFTPEVQASWQKMVAGVASALSSR       144
    177  VLDITR ARGYAADVEETWEG IASIAAPIHDR       207

>P1;HGMQJ    Hemoglobin gamma chain - Japanese macaque
No. 2.5 Score = 86  Quality = 32.576 HITS = 23 Mis. = 8 INDELS = 1 EXPECTED NO.  0.12E+01
        ** *  .. ....*. .*.  .*..*...  *
    113  VLAIHFGKEFTPEVQASWQKMVAGVASALSSR       144
    177  VLDITRARGYAADVEETWEG IASIAAPIHDR       207

>P1;HGMQR    Hemoglobin gamma chain - Rhesus macaque
No. 2.6 Score = 86  Quality = 32.576 HITS = 23 Mis. = 8 INDELS = 1 EXPECTED NO.  0.12E+01
        ** *  .. ....*. .*.  .*..*...  *
    113  VLAIHFGKEFTPEVQASWQKMVAGVASALSSR       144
    177  VLDITRARGYAADVEETWEG IASIAAPIHDR       207

>P1;HGBAY    Hemoglobin gamma chain - Yellow baboon
No. 2.7 Score = 86  Quality = 32.576 HITS = 23 Mis. = 8 INDELS = 1 EXPECTED NO.  0.12E+01
        ** *  .. ....*. .*.  .*..*...  *
    113  VLAIHFGKEFTPEVQASWQKMVAGVASALSSR       144
    177  VLDITRARGYAADVEETWEG IASIAAPIHDR       207

>P1;W1WLE    Probable E1 protein - Papillomavirus (type 1a)
No. 2.8 Score = 85  Quality = 16.634 HITS = 37 Mis. = 17 INDELS = 6 EXPECTED NO.  0.15E+01
        .** * *   .. . ....   * ** .* *. * . ...*. *..  .* * .**  **
    130  DET ENIDE STQVDQQQKEHTGEVGAAG VNIL KASNIRAALLSRFKDTA GVSFTDL       184
    191  EETWEGIASIAAPIHDRRRMPVGAVGITGAVERLCREGELRPELVAAVRDCARAVS RDL       249

>P1;RKNTL Ribulose bisphosphate carboxylase (EC 4.1.1.39) large chain precursor - Common t
No. 2.9 Score = 85  Quality = 18.931 HITS = 31 Mis. = 20 INDELS = 4 EXPECTED NO.  0.15E+01
        * *. **. .**.. . . . . *   .  . * .  * .. ***** . * *
    312  RVLAKALRM SGGDHIHS GTVVGKLEGERDITLGFV DLLRDDFVEQDRSRGIY       363
     10  RAAAM LRLLAGGERRLGLSDIASSLGLAKGTAHGILRTLQQEGFVEQDDASGRY        63

>P1;DEBSGF Glyceraldehyde 3-phosphate dehydrogenase (EC 1.2.1.12) - Bacillus stearothermoph
No. 2.10 Score = 84  Quality = 21.000 HITS = 32 Mis. = 11 INDELS = 6 EXPECTED NO.  0.17E+01
        * ..***.   . .. *. *.  * .. .*. .*  **.* *.  *.*.
     12  IGRNVFRAALKNPDI EVVAVNDLTNADGLAHLLK YDSV HGR LDAE        56
    113  IVHHVFRPD DSRQVLEIGAMQPL HSTALGKVLSAYDPVAHSEALEAD       159

>P1;BBMS     Complement factor B (in EC 3.4.21.47) - Mouse (fragment)
No. 2.11 Score = 84  Quality = 25.926 HITS = 25 Mis. = 8 INDELS = 7 EXPECTED NO.  0.17E+01
       .* * * . ** * *. * *. .**. . * . *..  **
   418  VHKRSRFIQVG V ISWGVVD VCRD QRRQQPVPSYARD       453
   204  IHDRRR MPVGAVGIT GAVERLCREGELRPELVAA VRD       240
```

```
>P1;HGHUA    Hemoglobin gamma chains - Human and chimpanzee
No. 2.12 Score = 83  Quality = 31.439 HITS = 23 Mis. = 8 INDELS = 1 EXPECTED NO.  0.21E+01
        ** *  .. ....*. .*.  ....*...  *
   113  VLAIHFGKEFTPEVQASWQKMVTAVASALSSR       144
   177  VLDITRARGYAADVEETWEG IASIAAPIHDR       207

>P1;YNEC     Cyanate hydrolase (EC 3.5.5.3) - Escherichia coli
No. 2.13 Score = 83  Quality = 27.303 HITS = 26 Mis. = 8 INDELS = 3 EXPECTED NO.  0.21E+01
        . ***. *. * *.* * . . . *. ..**.. ***
     6  INRNIR LDLADAIL L SKAKKDLSFAEIANGTGLA        39
     1  MARNIQSLERAAAMLRLLAGGERRLGLSDIASSLGLA        37

>P1;QQECE1   Hypothetical protein E-152 - Escherichia coli
No. 2.14 Score = 81  Quality = 15.698 HITS = 35 Mis. = 23 INDELS = 6 EXPECTED NO.  0.29E+01
        *..*.*.  .*  * *  *  . ...*   *.. ** *  .  . * * .   . * * . ***
     6  IDNLDRG  ILEALMGNART AYAELAKQFGVSPGTIHVRVEKMKQAGIITGARIDVSPK QLG        65
     5  IQSLERAAAMLRLLAGGERRLGLSDIASSLGLAKGTAHGILRTLQQEGFV  EQDDASGRYQLG        66

>P1;WMAD52   Late L1 52K protein - Adenovirus 2
No. 2.15 Score = 81  Quality = 31.641 HITS = 21 Mis. = 6 INDELS = 3 EXPECTED NO.  0.29E+01
        *** *  *.**  **.* ..**  .*. *.
    55  RARHYL DLEEG EGLARLGAPSPERHPRV        82
   182  RARGYAADVEETWEGIASIAAPIHDRR RM       210

>P1;ATZM1    Actin - Maize
No. 2.16 Score = 81  Quality = 15.084 HITS = 36 Mis. = 22 INDELS = 8 EXPECTED NO.  0.29E+01
        * * **  . . .* * * *.  * *  .... ..   ** * * ** . **  * *   .* **.
   207  EIVRDI KEKLAYVALDYEQELE TAKSSSSV EKSYEMPDGQV ITIGSERFRCPEVLFQPSLVG       268
   175  EHVLDITRAR GYAA DVEETWEGIASIAAPIHDRR RMPVGAVGITGAVERL CREGELRPELVA       236

>P1;GNFV1R   pol polyprotein - Rous sarcoma virus
No. 2.17 Score = 81  Quality = 20.611 HITS = 30 Mis. = 14 INDELS = 7 EXPECTED NO.  0.29E+01
        ***  **  ...  .. **.. ** . .*    * . * . *  .**.**.
   188  ASSHDGLE AAGEEVISTLERAGFTISPDKVQ R EPGVQYLGYKLGSTYV       235
    31  ASSL GLAKGTAHGILRTLQQEGF VEQDDASGRYQLGAELL  RLGTTYL        77

>P1;FWSYGB   Glycinin, B1a chain - Soybean
No. 2.18 Score = 80  Quality = 22.727 HITS = 23 Mis. = 14 INDELS = 3 EXPECTED NO.  0.34E+01
        .  * .****  . .   .** * . * *..  **** . *
    72  IYALNGRALV  Q VVNCNGERVFDGELQEGDVLIVPQNF       108
    79  VHELRARALVWTDDLARSSGESVHLGVLHQQGVLIVHHVF       118

>P1;AFMWB    Allophycocyanin beta chain - Mastigocladus laminosus
No. 2.19 Score = 80  Quality = 28.470 HITS = 18 Mis. = 13 INDELS = 3 EXPECTED NO.  0.34E+01
        ..  * *.* *  * * .*  .*    *** **
     9  INSSDVQGKY LDTAALEKL KSYFSTGELRVRA        40
    54  VEQDDASGRYQLG AELLRLGTTYLDVHELRARA        86

>P1;AFAIB    Allophycocyanin beta chain - Anabaena variabilis
No. 2.20 Score = 79  Quality = 28.114 HITS = 18 Mis. = 13 INDELS = 3 EXPECTED NO.  0.41E+01
        ..  * *.* *  * * .*  .*    *** **
     9  INSADVQGKY LDTAALEKL KAYFSTGELRVRA        40
    54  VEQDDASGRYQLG AELLRLGTTYLDVHELRARA        86

>P1;XUECAG   Glycerol-3-phosphate acyltransferase (EC 2.3.1.15) - Escherichia coli
No. 2.21 Score = 79  Quality = 17.634 HITS = 33 Mis. = 18 INDELS = 6 EXPECTED NO.  0.41E+01
        *.  *  **  *.*  **  . *.  *.   .*** .*  .  .** . .. ..* .*
   707  HI NPAHSRTLQLLAAGARETLQRYAITFWLLSA NPSINRGTLEKESRTVAQRLSV       761
   116  HVFRPDDSR  QVLEIGAMQPLHSTALG KVLSAYDPVAHSEALEADRKAFTDR TV       168

>F1;WMAD65   Late L1 52K protein - Adenovirus 5 (fragment)
No. 2.22 Score = 79  Quality = 35.426 HITS = 18 Mis. = 6 INDELS = 2 EXPECTED NO.  0.41E+01
        *** *  *.**  **.* ..**  .*
    55  RARHYL DLEEG EGLARLGAPSPER        78
   182  RARGYAADVEETWEGIASIAAPIHDR       207

>P1;P3ADA2   Peripentonal hexon-associated protein (IIIA) - Adenovirus 2
No. 2.23 Score = 79  Quality = 36.916 HITS = 15 Mis. = 8 INDELS = 3 EXPECTED NO.  0.41E+01
        *. * *.*** **    .   **  **
   545  HD RQRGLVWEDD DSADDSSVLDLG       568
    80  HELRARALVWTDDLARSSGESV HLG       104
```

```
>P1;P3ADA2   Peripentonal hexon-associated protein (IIIA) - Adenovirus 2
No. 2.24 Score = 79  Quality = 22.899 HITS = 27 Mis. = 11 INDELS = 6 EXPECTED NO.  0.41E+01
        .***  .*   *..  *.* . * * *  * *.* * . *.*..
   241  PFTDSGSVS RDTYLGHLLTLYREAIG QAHVDEHTFQEITSVS       282
   162  AFTDR TVCEPDSF EHVLDITR ARGYAADVEE TWEGIASIA       201

>P1;FJEC     nusA protein - Escherichia coli
No. 2.25 Score = 79  Quality = 30.502 HITS = 22 Mis. = 8 INDELS = 3 EXPECTED NO.  0.41E+01
        .. * .. **   .*.*  **** **. ** *.
   227  SRSGFSAKIAVKTNDKRIDPVGACVGMRGARVQ       259
   193  TWEGIAS IAAPIHDRRRMPVGA VGITGA VE       222

>P1;TFCHE    Transferrin precursor - Chicken
No. 2.26 Score = 78  Quality = 14.338 HITS = 40 Mis. = 19 INDELS = 6 EXPECTED NO.  0.48E+01
        *. .  * . *. ..*   *  .  ... ** .  *..*....   *  *.**.* .. .* .**
    93  LKPIAAEIY EHTEGSTTSY YAVAVVKKGTEF TVNDLQGKTSCHTG LGRSAGWNIPIGTLLH       153
    45  LRTLQQEGFVEQDDASG RYQLGAELLRLGTTYLDVHELRARALVWTDDLARSSGESVHLG VLH       107

>P1;FPHU     Alpha-fetoprotein precursor - Human
No. 2.27 Score = 78  Quality = 40.625 HITS = 15 Mis. = 8 INDELS = 0 EXPECTED NO.  0.48E+01
        .**.  ** * ** .* .. . *
   143  EPVTSCEAYEEDRETFMNKFIYE       165
   148  DPVAHSEALEADRKAFTDRTVCE       170

>P1;FOMV1M   gag polyprotein - Moloney murine leukemia virus
No. 2.28 Score = 78  Quality = 22.609 HITS = 23 Mis. = 14 INDELS = 2 EXPECTED NO.  0.48E+01
        * .. .  * ..  *  *  ..*** *.  . ** .*.*
   238  WKNNNPSFSEDPGKLTAL IESVLITHQPTW DDCQQLL       274
    89  WTDDLARSSGESVHLGVLHQQGVLIVHHVFRPDDSRQVL       127

>P1;HAXM     Hemoglobin alpha chain - Axolotl
No. 2.29 Score = 78  Quality = 21.429 HITS = 29 Mis. = 9 INDELS = 6 EXPECTED NO.  0.48E+01
        * . **  .. *. *.  .*** .  **..*  ..*** . ***
    41  HTYFPD  KD LNEGSF ALHSHG KKVMGALSNAVAHIDDLEA        79
   116  HVFRPDDSRQVLEIGAMQPLHSTALGKVLSAY DPVAHSEALEA       158

>P1;XJEC     Orotate phosphoribosyltransferase (EC 2.4.2.10) - Escherichia coli
No. 2.30 Score = 78  Quality = 16.561 HITS = 36 Mis. = 15 INDELS = 6 EXPECTED NO.  0.48E+01
        *. **. .* *  *. . *..*  *  .  .*. .**  .*. ....*. .* . *
   102  AKDHGEGGNLVGSALQGR VMLVDDVITAGTARESMEI  IQA NGATLAGLLISLD       154
    94  ARSSGESVHL G VLHQQGVLIVHHVFRPDDSRQVLEIGAMQPLHSTALGKVLSAYD       148

>P1;IQECDB   dnaB protein - Escherichia coli
No. 2.31 Score = 78  Quality = 9.432 HITS = 60 Mis. = 31 INDELS = 13 EXPECTED NO.  0.48E+01
      *.*  ***. . ..*   **.* *. .  .... . .**  **...  .  . *. * *.  *.
 354  DNR TLEIAEISRSL  KALAKELNVPVVALSQLNRSLEQRADKRP VNSDLRESGSIEQDADLIMF    426
 122  DSRQVLEIGAM QPLHSTALGKVLS AYDPVAH SEALE  ADRKAFTDRTVCEPDSFEHVLDITRA    173

      *  .*  *  . . .**  *  ..*  *.*.* .*
 427  I YRDEVYHENSD LKGIAEIIIGKQRNGPIGTVRLT 451
 174  RGYAADV EETWEGIASIAAPIHDRRRM PVGAVGIT 218

>P1;TQECT    Transposase (transposon Tn3) - Escherichia coli
No. 2.32 Score = 78  Quality = 22.674 HITS = 25 Mis. = 10 INDELS = 4 EXPECTED NO.  0.48E+01
      *  * * *.. *..* *. ** . * .** *. . ...
 830  RKIVLQW DEMIRTAG SLKLGKV QASVL VRSLLKSE       864
  83  RARALVWTDDLARSSGESVHLGVLHQQGVLIVHHVFRPD       121

>P1;DETWG3   Glyceraldehyde 3-phosphate dehydrogenase (EC 2.2.1.12) - Thermus aquaticus
No. 2.33 Score = 78  Quality = 22.350 HITS = 25 Mis. = 13 INDELS = 3 EXPECTED NO.  0.48E+01
      * ..***   ** * *.. .  * .  .*. .*  **.. *
  11  IGRQVFRILHSRGV EVALINDLTNDKTLAHLLK YDSIYH        49
 113  IVHHVFRPDDSRQVLEIGAMQPL HSTALGKVLSAYDPVAH       152
```

********** alignments 34 to 41 omitted **********

```
>P1;Z4BP33   Gene 430 protein - Bacteriophage Pf3
No. 2.42 Layer  237     Score=  76  Quality    7.795
HITS=   69 MISMATCHES=   35 INDELS=   23    EXPECTED NO.   0.678096E+01
        ** *  **   **  * . *. *..* *..   * *  **  .* *   .....**  ..
    204  RLGLNWGGALSLGNWSAVTAGDLSVAAGSSIGFGFLSNTLSLDGLFTAMENEGNGR VVS       262
     16  RL LA GGERRLGL SDI ASSLGLAKGTA  HGILR TLQQEG FVE QDDASGRYQLG       66

           * * *  ** ..  *  * .  .*. . *.*..   .*   .. * . *  *  **.
    263  RPTL L T  LD RQ SASVL RGTELPYQQSAGDGAT SVAFKHAALSLEVKPVISPDN       314
     67  AELLRLGTTYLDVHELRARALVWTDDLA R SSGESVHLGVLHQQGVL I VHHVFRPDD       122

         *  *.*.
    315  SI VIEV       320
    123  SRQVLEI       129
```

********** alignments 43 to 58 omitted **********

```
>P1;Z4BP33   Gene 430 protein - Bacteriophage Pf3
No. 2.59 Layer  237     Score=  74  Quality   12.781
HITS=   42 MISMATCHES=   24 INDELS=    7    EXPECTED NO.   0.946901E+01
        * *. .* **  .* * * .**    * ..*  * . ...  *. *..* .* . *..*
     80  LLSSMVGDVLVITAMDQVLNSERKADDLRTFR RDLFNANDIERRVINIVHA S ASEVV       136
    135  LHSTALGKVL  SAYDPVAHSEALEADRKAFTDRTVCEPDSFEH VLDITRARGYAADVE       191

            *.  *. **
    137  SLFKESFMSLDAP       149
    192  ETW EGIASIAAP       203
```

Summary of Results for *Drosophila melanogaster* YP3.

Score	Predicted	Observed	Score	Predicted	Observed
52	530.7	556	75	14.3	14
53	453.6	454	76	12.2	19
54	387.7	404	77	10.4	6
55	331.3	379	78	8.9	10
56	283.2	290	79	7.6	3
57	242.0	295	80	6.5	9
58	206.9	210	81	5.5	3
59	176.8	143	82	4.7	5
60	151.1	123	83	4.0	5
61	129.1	130	84	3.4	2
62	110.4	115	85	2.9	1
63	94.3	91	86	2.5	4
64	80.6	87	87	2.1	1
65	68.9	67	88	1.8	1
66	58.8	61	89	1.5	1
67	50.3	37	90	1.3	2
68	43.0	54	91	1.1	2
69	36.7	27	94	0.7	1
70	31.4	30	95	0.6	2
71	26.8	22	96	0.5	1
72	22.9	30	98	0.3	1
73	19.6	21	183	0.6E-06	1
74	16.7	13	1742	0.0E+00	1
			1935	0.0E+00	1

Administrative data from program run.

```
NBRF12FIN              LOADED SOURCE.YP3
    0 READ; FIRST LAYER SET TO    1
    0 READ; LAST LAYER SET TO  271
    1 TO  420; LENGTH  420
Pam 100  Indel 10
SOURCE.YP3 USED AS QUERY SEQUENCE, LENGTH   420
USING NBRF PROTEIN SEQUENCE DATABASE VERSION 12
  Proteins 4750 Residues 1103446
 EDINBURGH DAP VERSION OF 08/07/86
 TOTAL RESULTS 3735
 THRESHOLD VALUE SET INITIALLY WAS   10
 FINAL THRESHOLD VALUE IS   52
 MAXIMUM SCORE 1935

A 14.4; SIGA 0.3 B -0.2; SIGB 0.005 CHI2 0.8 HIGH 76
```

Listing Three. *Drosophila melanogaster* YP3.

```
>P1;VJFF1    Vitellogenin I precursor - Fruit fly
No. 3.1 Score = 1935  Quality = 53.9 HITS = 333 Mis. = 82 INDELS = 16 EXPECTED NO. 0.00E+00
  . . .  aligned along entire length, not displayed  . . .

>P1;VJFF2    Vitellogenin II precursor - Fruit fly
No. 3.2 Score = 1742  Quality = 48.5 HITS = 324 Mis. = 90 INDELS = 19 EXPECTED NO. 0.00E+00
   . . .  aligned along entire length, not displayed  . . .

>P1;LIPG     Triacylglycerol lipase (EC 3.1.1.3) - Pig
No. 3.3 Score = 183 Quality = 17 HITS = 80 MISMATCHES = 36 INDELS = 20 EXPECTED NO. 0.6E-06
        .*.**.....* ** ** . *   *  . ********      *. * *  .** *** *** ..   . .*
    146  VHVIGHSLGSHAAGEAGRR T  NG TIERITGLDPAEPCFQGTPE LVRLDPSDAKFVDVIHTDAAPIIP 211
    239  IHLIGQGISAHVAGAAGNKYTAQTGHKLRRITGLDPAKV LSKRPQILGGLSRGDADFVDAIHT ST F A 306

             . .  * .* .*** .  .** .. *  ...       .... *. *.*  *. * *. **
    212  NLGFGMSQTVGHLDFFPNG GKQMPGCQKNI  LSQIVD I DGIWEGT RDF  VACNHLRSYK 268
    307   MGTPI R CGDVDLYPNGPSTGVPGSENVIEAVARATRYFAESVRPGSERNFPAVPANSLKQYK 367

>P1;VHVUNH   Probable nucleoprotein - Snowshoe hare bunyavirus
No. 3.4 Score = 98  Quality = 15.170 HITS = 47 Mis. = 27 INDELS = 7 EXPECTED NO.  0.38E+00
     ** * *** .* .*    *  . .*  ... ...** .*. .  *.*   * .* ...* *..  .   .*  . .* *.
 60  KANPKFGEWQVEVVNNHFPGNRNNPINSDDLTIHRL SGYLARWVLEQYKENEDESRRELIKTTIIN PIAESNGVRWDSG  138
113  KAQPGFGEDEVTIVLTGLP  KTSP  AQQKAMRRLIQAYVQKYNLQQLQKNAQEQQQQL KSSDYDYTSSEEAADQWKSA  188

>P1;OBWT2    Cytochrome c oxidase (EC 1.9.3.1), polypeptide II - Wheat mitochondrion
No. 3.5 Score = 96  Quality = 27.507 HITS = 29 Mis. = 12 INDELS = 6 EXPECTED NO.  0.52E+00
        .***  *.   .* *   *... * .* *... * *  * * .**.*
    202  AVPGRSNLTSISVQREGVYYGQCSEIR GTNHAFTPIVVEAVTLKDY       247
    325  GVPGSENVIE AVARATRYFAE S VRPGSERNF P AVPANSLKQY       366

>P1;SYECAT   Alanyl-tRNA synthetase (EC 6.1.1.7) - Escherichia coli
No. 3.6 Score = 95  Quality = 11.980 HITS = 56 Mis. = 29 INDELS = 13 EXPECTED NO.  0.61E+00
        *.. * * . ****  . . ....*.*.    *   .* *.. .*..   .  * .*.* *.   . * *.
    736  LERTRQLEKELQQLK  E Q AAAQESANLSSKAIDVNGVKLLVSELSGVEPKMLR TMVDDLKN QL GSTI   801
    157  LQKNAQ EQQ QQLKSSDYDYTSSEEAADQWKSAKAASG DLIIIDLGSTLTNFKRYAMLDVLNTGAMIGQTL   226

         * *.   * .  . **. *.*  *.
    802   IVLAT VV EGKVSLIA GVSKDVT       823
    227   IDLTNKGVPQEIIHLIGQGISAHVA       251

>P1;O4RBPC   Cytochrome P450, phenobarbital-inducible, liver (version 2) - Rabbit
No. 3.7 Score = 95  Quality = 17.336 HITS = 38 Mis. = 23 INDELS = 6 EXPECTED NO.  0.61E+00
        *.*  *  *   *.**   ** *..* .   *  .*  . *....*.   *.** *  .. * ** *
    420  LKRNEGFMPFS LGKR ICLGEGIARTELFLFFTTILQN FSIASPVPPEDIDLTPRESGVGNVPPS       483
    266  LRRITGLDPAKVLSKRPQILG GLSRGDA DFVDAIHTSTFAMGTPIRCGDVDLYPNGPSTG VPGS       329

>P1;AJECNA   Asparagine synthetase (EC 6.3.1.1) - Escherichia coli
No. 3.8 Score = 94  Quality = 22.651 HITS = 32 Mis. = 11 INDELS = 7 EXPECTED NO.  0.72E+00
        * *..** . ..** **  *.  **. . * ... * ..      .**.*.
    221  WSTPSELGH AGLNGDILVWNPVLED AFELSSMGI RV DADTLKHQLA       266
     33  WLTATELENVPSLN DI TWER LENQPLEQGAKVIEKIYHVGQIKHDLT        79

>P1;VHVULV   Nucleoprotein N - La Crosse bunyavirus
No. 3.9 Score = 91  Quality = 18.919 HITS = 34 Mis. = 21 INDELS = 5 EXPECTED NO.  0.11E+01
        ** * *** .* ..    *  . .* .    ...** .*. .  *.*   * .* *..*
     60  KANPKFGEWQVEVINNHFPGNRNNPIGNNDLTIHRL SGYLARWVLDQYNENDDESQHEL       118
    113  KAQPGFGEDEVTIVLTGLP  KTSP AQQK AMRRLIQAYVQKYNLQQLQKNAQEQQQQL       168

>P1;R5EC19   50S ribosomal protein L19 - Escherichia coli
No. 3.10 Score = 91  Quality = 25.490 HITS = 29 Mis. = 10 INDELS = 3 EXPECTED NO.  0.11E+01
          ...*  . .  *.*.*. *** *. . .* **.. ..  ...
      1  SNIIKQLEQ EQMKQDV PSFRPGDTVEVKVWVVEGSKKRLQ        40
     62  AKVIEKIYHVGQIKHDLTPSFVPSPS NVPVWIIKSNGQKVE       102

>P1;HIBPC7   Internal virion protein C - Bacteriophage T7
No. 3.11 Score = 90  Quality = 23.256 HITS = 30 Mis. = 13 INDELS = 3 EXPECTED NO.  0.13E+01
        ..... * . ** ... * * .  . .* . .  .*... * ..*.
    341  PDEQMTPQREWLISAQ EQVQNQMN AWTKAQAKALDDSMKSMNKL       384
     24  SNDRLKPTK WLTATELENVPSLNDITWERLENQPLEQGAKVIEKI        68
```

```
>F1;MWKW1    Myosin heavy chain I - Caenorhabditis elegans (fragment)
No. 3.12 Score = 90  Quality = 8.341 HITS = 69 Mis. = 51 INDELS = 12 EXPECTED NO.  0.13E+01
       *...  ***..*   ** *  .   *   .  ..**    ...   ... * . * . .   * * ...  *.
  1127  LDELN ERLDEQNKQLEIQQDNNKKKDSEIIKFRRDLDEKNMANEDQMAM IRRKNNDQIS ALTNTLDALQKSK  1198
    45  LNDITWERLENQ  PLE QGAKVIEKIYHVGQIKHDLTPSFVPSPSNVPVWIIKSNGQKVECKLNNYVET AKAQ   106

        .  * *  ..   *   ..  .*  ..*. *  . * *..*  .** . .** .*.
  1199  AKI EKEKGVLQKELDDINAQVDQETKSRVEQERLAKQYEIQVAELQQKVDEQSRQI 1256
   107  PGFGEDEVTIVLTGLPKTSP AQQKAMRRLIQAYVQK YNLQ  QLQKNAQEQQQQL  169

>P1;OKBO1R   cAMP-dependent protein kinase (EC 2.7.1.37) type I regulatory chain - Bovine
No. 3.13 Score = 89  Quality = 35.458 HITS = 21 Mis. = 7 INDELS = 1 EXPECTED NO.  0.15E+01
        *.. .....*   . **** *.* * *..
     5  TASEEERSLRE CELYVQKHNIQALLKDS        32
   133  TSPAQQKAMRRLIQAYVQKYNLQQLQKNA       161

>P1;ANRT     Angiotensinogen precursor - Rat
No. 3.14 Score = 88  Quality = 31.655 HITS = 24 Mis. = 7 INDELS = 2 EXPECTED NO.  0.18E+01
        .**** ..*..* .*. *  .* . .. *. **
    65  AKTSPVDEKTLRDKLVLA TEKLEAEDRQRAAQ        96
   131  PKTSPAQQKAMR RLIQAYVQKYNLQQLQKNAQ       162

>P1;FOMVGS   gag polyprotein - Simian sarcoma virus
No. 3.15 Score = 87  Quality = 23.200 HITS = 29 Mis. = 11 INDELS = 5 EXPECTED NO.  0.21E+01
        *..  *.   **   ..*   * ..** .. .* *.**.* ..*.
   376  GQQAAVATAFTG Q SAPDIKKKLQRL EG LQDYSLQDLVREAE       416
   119  GEDE VTIVLTGLPKTSPAQQKAMRRLIQAYVQKYNLQQLQKNAQ       162

>P1;FOMVMD   gag polyprotein - Feline sarcoma virus (strain McDonough)
No. 3.16 Score = 86  Quality = 18.105 HITS = 27 Mis. = 24 INDELS = 4 EXPECTED NO.  0.25E+01
        **.*  * * *  .* . * *    **.  . *  .*.  *  * *  * .. .
   111  KWITLCEAEWV MMN VGWPREGTPPLDNTSQVEKRIFAPGPHGHPDQVP YITT       162
    32  KWLTATELENVPSLNDITWERLENQPLEQGAKVIEKIYHVGQIKH DLTPSFVPS        85

>P1;VWHU     von Willebrand factor precursor - Human
No. 3.17 Score = 86  Quality = 19.501 HITS = 36 Mis. = 13 INDELS = 10 EXPECTED NO.  0.25E+01
        ** ***   ** .* ***   *. . .**..  .  .*   .*.*. . .**  * **.
   720  LGVSTLGP KRNSMVLDVAFVLEGSDKIGEADFN RSKEFMEEVIQRMDVGQD SIHVT       775
   200  LG STLTNFKRYAM LDV   LNTGAMIGQTLIDLTNKGVPQEIIH L IGQGISAHVA       251

>P1;OBZM2 Cytochrome c oxidase (EC 1.9.3.1), polypeptide II - Maize mitochondrion and rice
No. 3.18 Score = 86  Quality = 24.642 HITS = 27 Mis. = 15 INDELS = 4 EXPECTED NO.  0.25E+01
        .***  *.   .* *   *... *   *... * *  * * .**.*
   202  AVPGRSNLTSISVQREGVYYGQCSEICGTNHAFTPIVVEAVTLKDY       247
   325  GVPGSENVIE AVARATRYFAE SVRPGSERNF P AVPANSLKQY       366

>P1;W2WLEB   Probable E2 protein - Bovine papillomavirus (type 1)
No. 3.19 Score = 86  Quality = 30.935 HITS = 23 Mis. = 10 INDELS = 4 EXPECTED NO.  0.25E+01
        *. **  . * ** * .*.* . * ..    **.*.*
    82  LSKTEFGDEPWSLLDTSWDRYMSEPKRCFKKGARVVE       118
    34  LTATELENVP SLNDITWER L E NQPLEQGAKVIE        66

>P1;DJECI    DNA-directed DNA polymerase I (EC 2.7.7.7) - Escherichia coli
No. 3.20 Score = 85  Quality = 12.537 HITS = 56 Mis. = 16 INDELS = 16 EXPECTED NO. 0.29E+01
        .... . * ...*    .. ** ***: * **  ..  * *. ...* .. *.* ***.   *. . **
   150  TLINTM TNTILGPE  EVVNKYGVPPELIIDFLALMGDSSDNIPGVPGVGEK TAQA L LQGLGGL   212
   211  AMLDVLNTGAMIGQTLIDLTNK GVPQE IIHLIG QGISA HVAG AA GNKYTAQTGHKLRRITGL   272

        * . . .  *. ..*** **
   213  D T LYAE PEKIAGLSFRG   228
   273  DPAKVLSKRPQILGGLS RG   291

>P1;MMEBAD   Outer membrane protein A precursor - Shigella dysenteriae
No. 3.21 Score = 84  Quality = 32.061 HITS = 21 Mis. = 11 INDELS = 1 EXPECTED NO.  0.34E+01
        .*...* * .**.* . *   * * *  *.* .
   283  AQSVVDYLISKGIPADKISARGMGESNPVTGNT       315
   223  GQTLID LTNKGVPQEIIHLIGQGISAHVAGAA       254
```

```
>P1;MMECA    Outer membrane protein A precursor - Escherichia coli
No. 3.22 Score = 84  Quality = 32.061 HITS = 21 Mis. = 11 INDELS = 1 EXPECTED NO.  0.34E+01
         .*...* * .**.* . *   * * *  *.* .
    278  AQSVVDYLISKGIPADKISARGMGESNPVTGNT       310
    223  GQTLID LTNKGVPQEIIHLIGQGISAHVAGAA       254

>F1;WKFF     Neurogenic repetitive locus protein - Fruit fly (fragment)
No. 3.23 Score = 83  Quality = 24.412 HITS = 19 Mis. = 19 INDELS = 1 EXPECTED NO.  0.40E+01
         .   ** **   ..  *   *    ** *   *.*****
     28  MQTISPQQQQQQQQQQQQQHQQ QQQQQQQQQQQQQQQL        65
    130  LPKTSPAQQKAMRRLIQAYVQKYNLQQLQKNAQEQQQQL       168

>P1;MMEBAT   Outer membrane protein A precursor - Salmonella typhimurium
No. 3.24 Score = 83  Quality = 31.679 HITS = 21 Mis. = 11 INDELS = 1 EXPECTED NO.  0.40E+01
         .*...* * .**.* . *   * * *  *.* .
    282  AQSVVDYLISKGIPSDKISARGMGESNPVTGNT       314
    223  GQTLID LTNKGVPQEIIHLIGQGISAHVAGAA       254

>P1;GNFV1R   pol polyprotein - Rous sarcoma virus
No. 3.25 Score = 83 Quality = 11.942 HITS = 51 Mis. = 23 INDELS = 14 EXPECTED NO.  0.40E+01
   .* ** .* *..  . .. . * .*  . . ** *  * ** .* *..*  ..   *  *. ** * **  .. . **..**.  *
22 DQWPLPEG KLVA LTQLVE K ELQLGHIEPSLSCWNTPVFVIRKASGSYRLLHDLRA VN AKLVP FGAVQQGAPVLSALPRGWP
55 NQ PLEQGAKVIEKIYHVGQIKHDLTPSFV PSPS  NVPVWII KSNGQ KVECKLNNYVETAKAQPGFGE DEVTIVLTGLPKTSP

>P1;SYBYMT   Methionyl-tRNA synthetase (EC 6.1.1.10) - Baker's yeast
No. 3.26 Score = 83  Quality = 25.305 HITS = 28 Mis. = 10 INDELS = 4 EXPECTED NO.  0.40E+01
         .... . ** *. *  .. .. .  **.  **** .*. .*.
    526  FSKSRGVGVFGN NAQDSGISPSVWRYYLASVRPESSDSHFS       566
    318  YPNGPSTGVPGSENVIEA VARAT RYFAESVRP GSERNFP       356

>P1;QQPSHC   Hypothetical protein merC (transposon Tn501) - Pseudomonas aeruginosa
No. 3.27 Score = 83  Quality = 37.727 HITS = 18 Mis. = 8 INDELS = 1 EXPECTED NO.  0.40E+01
         ***.   *.. . *  * . ****..*
     20  LLASSQWVPSSSRSSSAISSRLKPSRW        46
      8  LLATCLLVAAHA SKDASNDRLKPTKW        33

>P1;VCLJH3   env polyprotein precursor - AIDS virus HTLV-III (T-cell leukemia virus, BH10)
No. 3.28 Score = 82  Quality = 25.309 HITS = 25 Mis. = 13 INDELS = 5 EXPECTED NO.  0.47E+01
         * *  *. . .*. *.. *** *  * ** *  .  .* * **
    812  VSLL NATAIAVAEGTDRVIEVVQGAYRAIRHIPRRIRQGLER       853
    315  VDLYPNGPSTGVP GSENVIEAV  A RATRYFAESVRPGSER       353
```

Summary of Results for Human CF antigen.

Score	Predicted	Observed	Score	Predicted	Observed
50	158.6	149	76	1.1	1
51	131.2	125	77	0.94	1
52	108.6	108	78	0.78	1
53	89.83	97	79	0.65	4
54	74.32	79	80	0.53	1
55	61.49	68	81	0.44	1
56	50.87	42	82	0.36	2
57	42.09	30	83	0.30	1
58	34.82	35	84	0.25	1
59	28.81	32	85	0.20	6
60	23.83	33	87	0.14	2
61	19.72	25	89	0.97E-01	2
62	16.31	16	90	0.80E-01	1
63	13.50	18	92	0.55E-01	2
64	11.17	10	93	0.45E-01	1
65	9.237	8	96	0.25E-01	2
66	7.642	1	97	0.21E-01	1
67	6.323	10	98	0.17E-01	8
68	5.231	10	119	0.33E-03	1
69	4.328	7	154	0.43E-06	1
70	3.580	2	173	0.11E-07	1
71	2.962	11	177	0.55E-08	1
72	2.451	4	179	0.38E-08	1
73	2.027	3	184	0.14E-08	1
75	1.388	2	195	0.18E-09	1

Administrative data from program run.

```
100 Pams
Indel 10

 CFIB         USED AS QUERY SEQUENCE, LENGTH     94 START
1 END     94
NBRF11 and new seqeunces appended; 25-3-87 15:40pm
  Proteins 4612 Residues 1066790
 MCDAP_NW3OP VERSION OF 08/07/86
 TOTAL RESULTS 3538
 THRESHOLD VALUE SET INITIALLY WAS    10
 FINAL THRESHOLD VALUE IS    43
 GEN. PENALTY ADDED TO NEGATIVE SCORES FROM FIGURE: 0
 MAXIMUM SCORE  195

A 14.5; SIGA 0.7 B -0.2; SIGB 0.01 CHI2 5.5 HIGH 69
```

Listing Four. Human cystic fibrosis antigen.

```
>P1;BCBOIA   S-100 protein, alpha chain - Bovine
No. 4.1 Score = 195  Quality = 30.139 HITS = 51 Mis. = 22 INDELS = 4 EXPECTED No. 0.18E-09
       .*** *....*.*.*  *   *.   . . .** **.** . ..   * ** *    **** . ** *.***...**
    2  SELETAMETLINVFHAHSGKEGDKYKLSKKELKELLQTELSGFLDAQKDADAVDKVMKELDEDGDGEVDFQEYVVLV   78
    3  TELEKALNSIIDVYHKYSLIKGNFHAVYRDDLKKLLETECPQYI RKKGAD V W FKELDINTDGAVNFQEFLILV   75

>P1;KLBOI    Calcium-binding protein, intestinal - Bovine
No. 4.2 Score = 184  Quality = 28.264 HITS = 48 Mis. = 25 INDELS = 1 EXPECTED No. 0.14E-08
        *. . .  .. **.   *. . . . ...** **.** *  ..  .. *  * *** * ***.*.** .** *..
    1  KSPEELKGIFEKYAAKEGDPNQLSKEELKLLLQTEFPSLLKGPSTLDELFEELDKNGDGEVSFEEFQVLVKKIS  74
    7  KALNSIIDVYHKYSLIKGNFHAVYRDDLKKLLETECPQYIRKKGA DVWFKELDINTDGAVNFQEFLILVIKMA  79

>P1;BCHUIB   S-100 protein, beta chain - Human
No. 4.3 Score = 179  Quality = 26.208 HITS = 52 Mis. = 25 INDELS = 4 EXPECTED No. 0.38E-08
      .*****. ..***.* **   *. * . . .** *.. * ....   *  .*    .* ** . **  .****.  * ...
     1 SELEKAMVALIDVFHQYSGREGDKHKLKKSELKELINNELSHFLEEIKEQEVVDKVMETLDNDGDGECDFQEFMAFVAMVT 81
     3 TELEKALNSIIDVYHKYSLIKGNFHAVYRDDLKKLLETECPQYIRK KGADV  WFKE LDINTDGAVNFQEFLILVIKMA 79

>P1;BCBOIB   S-100 protein, beta chain - Bovine
No. 4.4 Score = 177  Quality = 25.915 HITS = 52 Mis. = 25 INDELS = 4 EXPECTED No. 0.55E-08
      .*****. ..***.* **   *. * . . .** *.. * ....   *  .*    .* ** . **  .****.  * ...
   1  SELEKAVVALIDVFHQYSGREGDKHKLKKSELKELINNELSHFLEEIKEQEVVDKVMETLDSDGDGECDFQEFMAFVAMIT  81
   3  TELEKALNSIIDVYHKYSLIKGNFHAVYRDDLKKLLETECPQYIRK KGADV  WFKE LDINTDGAVNFQEFLILVIKMA  79

>P1;KLPGI    Calcium-binding protein, intestinal - Pig
No. 4.5  Score = 173  Quality = 26.252 HITS = 47 Mis. = 27 INDELS = 1 EXPECTED No. 0.11E-07
        .*.   . .. **.   *. . . ...** *...* *  ..  .  *  * *** * .**.*.** .** *..
   3  QKSPAELKSIFEKYAAKEGDPNQLSKEELKQLIQAEFPSLLKGPRTLDDLFQELDKNGNGEVSFEEFQVLVKKIS   77
   6  EKALNSIIDVYHKYSLIKGNFHAVYRDDLKKLLETECPQYIR KKGADVWFKELDINTDGAVNFQEFLILVIKMA   79

>F1;KLRTI    Calcium-binding protein, intestinal - Rat (fragment)
No. 4.6 Score = 154  Quality = 26.146 HITS = 43 Mis. = 22 INDELS = 1 EXPECTED No. 0.43E-06
        ...**.   *. . . ...** *...* *  ..  .. *  * *** * ** *...** .   *..
     3  IFQKYAAKEGDPNQLSKEELKLLIQSEFPNLLKASSTLDNLFEELDKNDDGEVSYEEFEVFFKKLS        68
    15  VYHKYSLIKGNFHAVYRDDLKKLLETECPQYIRKKGA DVWFKELDINTDGAVNFQEFLILVIKMA        79

>P1;LUPG10   Calpactin I light chain - Pig
No. 4.7 Score = 119  Quality = 18.393 HITS = 47 Mis. = 23 INDELS = 7 EXPECTED No. 0.33E-03
        ...* *.....  .**..  ** . .  ..**. *.* * * .. . * .  *   *.**   ** * ** *  *.
  2  SQMEHAMETMMFTFHKFAGDKG YLT  KEDLRVLMEKEFPGFLENQKDPLAVDKIMKDLDQCRDGKVGFQSFFSLI   75
  3  TELEKALNSIIDVYHKYSLIKGNFHAVYRDDLKKLLETECPQYIRK KGA DV WF KELDINTDGAVNFQEFLILV   75

>P1;A23758   Calmodulin - Chlamydomonas reinhardtii
No. 4.8 Score = 98  Quality = 51.309 HITS = 18 Mis. = 4 INDELS = 1 EXPECTED No. 0.17E-01
        *.* . .*...* ***.*.  **
    57  EVDADGNGTIDFPEFLMLMARKM        79
    57  ELDINTDGAVNFQEFLILVI KM        78

>P1;TPRBCW   Troponin C, slow skeletal and cardiac muscles - Rabbit
No. 4.9 Score = 98  Quality = 51.309 HITS = 19 Mis. = 3 INDELS = 1 EXPECTED No. 0.17E-01
        *.* .  *.*.*.***..... *
    63  EVDEDGSGTVDFDEFLVMMVRCM        85
    57  ELDINTDGAVNFQEFLILVIK M        78

>P1;TPPGCS   Troponin C, skeletal muscle - Pig
No. 4.10 Score = 98  Quality = 51.309 HITS = 19 Mis. = 3 INDELS = 1 EXPECTED No. 0.17E-01
        *.* .  *...*.***..... *
    61  EVDEDGSGTIDFEEFLVMMVRQM        83
    57  ELDINTDGAVNFQEFLILVIK M        78

>P1;TPCHCS   Troponin C, skeletal muscle - Chicken
No. 4.11 Score = 98  Quality = 51.309 HITS = 19 Mis. = 3 INDELS = 1 EXPECTED No. 0.17E-01
        *.* .  *...*.***..... *
    64  EVDEDGSGTIDFEEFLVMMVRQM        86
    57  ELDINTDGAVNFQEFLILVIK M        78
```

```
>P1;TPHUCS   Troponin C, skeletal muscle - Human
No. 4.12 Score = 98  Quality = 51.309 HITS = 19 Mis. = 3 INDELS = 1 EXPECTED No. 0.17E-01
         *.* .  *...*.***..... *
     61  EVDEDGSGTIDFEEFLVMMVRQM        83
     57  ELDINTDGAVNFQEFLILVIK M        78

>P1;TPBOCC   Troponin C, cardiac muscle - Bovine
No. 4.13 Score = 98  Quality = 51.309 HITS = 19 Mis. = 3 INDELS = 1 EXPECTED No. 0.17E-01
         *.* .  *.*.*.***..... *
     63  EVDEDGSGTVDFDEFLVMMVRCM        85
     57  ELDINTDGAVNFQEFLILVIK M        78

>P1;TPFGCS   Troponin C, skeletal muscle - Edible frog
No. 4.14 Score = 98  Quality = 51.309 HITS = 19 Mis. = 3 INDELS = 1 EXPECTED No. 0.17E-01
         *.* .  *...*.***..... *
     64  EVDEDGSGTIDFEEFLVMMVRQM        86
     57  ELDINTDGAVNFQEFLILVIK M        78

>P1;TPRBCS   Troponin C, skeletal muscle - Rabbit
No. 4.15 Score = 98  Quality = 51.309 HITS = 19 Mis. = 3 INDELS = 1 EXPECTED No. 0.17E-01
         *.* .  *...*.***..... *
     61  EVDEDGSGTIDFEEFLVMMVRQM        83
     57  ELDINTDGAVNFQEFLILVIK M        78

>P1;MCTE     Calmodulin - Tetrahymena pyriformis
No. 4.16 Score = 97  Quality = 48.744 HITS = 18 Mis. = 5 INDELS = 1 EXPECTED No. 0.21E-01
         .*.* . **...* *** *.  **
     53  NEVDADGDGTIDFPEFLSLMARKM        76
     56  KELDINTDGAVNFQEFLILVI KM        78

>P1;PVSNBB   Parvalbumin beta - Boa constrictor
No. 4.17 Score = 96  Quality = 23.415 HITS = 28 Mis. = 18 INDELS = 2 EXPECTED No. 0.25E-01
         *.*** *.  .  .. .  * .    ** * . ** .  .**..** *
     61  DELKKFLQNFDGKARDLTDKETAEFLKEGDTDGDGKIGVEEFVVLVTK       108
     32  DDLKKLLET EC PQYIRKKGADVWFKELDINTDGAVNFQEFLILVIK        77

>P1;MCSW     Calmodulin - Scallop, sea-anemone, and sea-pansy
No. 4.18 Score = 96  Quality = 48.241 HITS = 18 Mis. = 5 INDELS = 1 EXPECTED No. 0.25E-01
         .*.* . **...* *** ..  **
     53  NEVDADGDGTIDFPEFLTMMARKM        76
     56  KELDINTDGAVNFQEFL ILVIKM        78

>P1;A23759   Calmodulin - Spinach
No. 4.19 Score = 93  Quality = 46.734 HITS = 18 Mis. = 5 INDELS = 1 EXPECTED No. 0.45E-01
         .*.* . .*...* *** *.  **
     53  NEVDADGNGTIDFPEFLNLMARKM        76
     56  KELDINTDGAVNFQEFLILVI KM        78

>P1;MCEE     Calmodulin - Electric eel
No. 4.20 Score = 92  Quality = 46.231 HITS = 18 Mis. = 5 INDELS = 1 EXPECTED No. 0.55E-01
         .*.* . .*...* *** ..  **
     53  NEVDADGNGTIDFPEFLTMMAKKM        76
     56  KELDINTDGAVNFQEFL ILVIKM        78

>P1;MCHU     Calmodulin - Human, rabbit, bovine, rat, and chicken
No. 4.21 Score = 92  Quality = 46.231 HITS = 18 Mis. = 5 INDELS = 1 EXPECTED No. 0.55E-01
         .*.* . .*...* *** ..  **
     53  NEVDADGNGTIDFPEFLTMMARKM        76
     56  KELDINTDGAVNFQEFL ILVIKM        78

>P1;GNWE2C   Genome polyprotein B - Cowpea mosaic virus
No. 4.22 Score = 90  Quality = 14.196 HITS = 40 Mis. = 24 INDELS = 13 EXPECTED No. 0.80E-01
       .*  *.*  .*   * *  *. * **   *  *.. *. *  **  **.  .  .* ***    *. ** .* **
  100  LYKHYALFISNL  VTRT LRFKELLLF CKQQFLEKMQASIVWAPELEQYLQVEGDAVA QGVSQLLYKMVTWVPT  171
   15  VYHKYSLIKGNFHAVYRDDLK K LLETEC PQYIRKKGAD VWFKELD  INTDG AVNFQEFLILVIKM AWQPT   83

>P1;MCSW     Calmodulin - Scallop, sea-anemone, and sea-pansy
No. 4.23 Score = 89  Quality = 52.663 HITS = 16 Mis. = 4 INDELS = 0 EXPECTED No. 0.97E-01
         .* **. ** **..**. ..
    126  READIDGDGQVNYEEFVTMM       145
     56  KELDINTDGAVNFQEFLILV        75
```

```
>P1;MCDO     Calmodulin - Dictyostelium discoideum
No. 4.24 Score = 89  Quality = 44.724 HITS = 17 Mis. = 6 INDELS = 1 EXPECTED No. 0.97E-01
         .*.* . .* ..* *** ..  **
     55  NEVDADGNGNIDFPEFLTMMARKM        78
     56  KELDINTDGAVNFQEFL ILVIKM        78

>P1;TPRBCW   Troponin C, slow skeletal and cardiac muscles - Rabbit
No. 4.25 Score = 87  Quality = 23.706 HITS = 24 Mis. = 16 INDELS = 1 EXPECTED No. 0.14E+00
         *.** .*..   . *     .  *. * * ** ....***
    115  DELKIMLQAT GETITEDDIEELMKDGDKNNDGRIDYDEFL       154
     32  DDLKKLLETECPQYIRKKGADVWFKELDINTDGAVNFQEFL        72

>P1;MCCHM    Calmodulin, striated muscle - Chicken
No. 4.26 Score = 87  Quality = 45.550 HITS = 16 Mis. = 6 INDELS = 1 EXPECTED No. 0.14E+00
         *.* .  *...* *** *.  **
     54  EVDADGSGTIDFPEFLSLMARKM         76
     57  ELDINTDGAVNFQEFLILVI KM         78

>P1;A23759   Calmodulin - Spinach
No. 4.27 Score = 85  Quality = 41.463 HITS = 18 Mis. = 4 INDELS = 2 EXPECTED No. 0.20E+00
         .* *.. ** .*..** . *. **
    126  READVDGDGQINYEEF VKVM MA        147
     56  KELDINTDGAVNFQEFLILVIKMA         79

>P1;KLSWM    Calcium-binding protein, muscle - Yesso scallop
No. 4.28 Score = 85  Quality = 52.795 HITS = 14 Mis. = 4 INDELS = 0 EXPECTED No. 0.20E+00
         .*.* **.* ** .. ..
     10  IWYKSLDVNHDGIISIEN         27
     53  VWFKELDINTDGAVNFQE         70

>P1;MCHU     Calmodulin - Human, rabbit, bovine, rat, and chicken
No. 4.29 Score = 85  Quality = 59.441 HITS = 14 Mis. = 3 INDELS = 0 EXPECTED No. 0.20E+00
         .* **. ** **..**.
    126  READIDGDGQVNYEEFV        142
     56  KELDINTDGAVNFQEFL         72

>P1;MCDO     Calmodulin - Dictyostelium discoideum
No. 4.30 Score = 85  Quality = 47.753 HITS = 17 Mis. = 4 INDELS = 0 EXPECTED No. 0.20E+00
         .* *.. ** **..**. ..*
    128  READLDGDGQVNYDEFVKMMI        148
     56  KELDINTDGAVNFQEFLILVI         76
```

Summary of Results for *Escherichia coli fts*A.

Score	Predicted	Observed	Score	Predicted	Observed
42	648.5	573	59	30.3	29
43	541.6	502	60	25.3	29
44	452.3	428	61	21.1	15
45	377.8	334	62	17.6	14
46	315.5	288	63	14.7	10
47	263.5	258	64	12.3	12
48	220.0	224	65	10.2	6
49	183.8	200	66	8.5	8
50	153.5	172	67	7.1	13
51	128.2	155	68	5.9	5
52	107.0	118	69	5.0	4
53	89.4	91	70	4.1	4
54	74.6	83	73	2.4	4
55	62.3	80	74	2.0	6
56	52.0	52	76	1.4	1
57	43.4	46	77	1.1	1
58	36.3	38	81	0.6	1
			172	0.4E-07	1

Administrative data from program run.

```
V8P100 - 100 pams
Indel -10
PATTERN     USED AS QUERY SEQUENCE, LENGTH    60 START
1 END    60
 NBRF Version  8.00 Proteins 3557 Residues  809386
 MCDAP_NW3OP VERSION OF 08/07/86
 TOTAL RESULTS 3805
 THRESHOLD VALUE SET INITIALLY WAS   10
 FINAL THRESHOLD VALUE IS   42
 GEN. PENALTY ADDED TO NEGATIVE SCORES FROM FIGURE: 0
 MAXIMUM SCORE  172

A 14.0; SIGA 0.3 B -0.18; SIGB 0.005 CHI2 0.4 HIGH 62
```

Listing Five. *Escherichia coli ftsA* protein.

```
>P1;TVBY8    Cell division control protein 28 - Baker's yeast
No. 5.1 Score = 172  Quality = 86.432 HITS = 21 Mis. = 34 INDELS = 5 EXPECTED NO.  0.43E-07
        * ** *  *    * .    *           **      *.  **  *   **.*  **
     84  LYLVFE FLDLDLK RYMEGIPKDQPLGAD  IVKKFMM QLCKGIAYCHSHRILHRDLK       138
      1  LXLVXEJILXXXXKJKXXXJIXXXXXXXXJJJIVXXXXXJQIXJGIXXCXXJRIIHXXLK        60

>P1;QXHU4M   Hypothetical protein 4 - Human mitochondrion (SGC1)
No. 5.2 Score = 81  Quality = 81.000 HITS = 10 Mis. = 29 INDELS = 2 EXPECTED NO.  0.57E+00
        *   * *   *          *.      **    .  *
     14  LTWLSK KHMIWINTTTHSLIIS IIPLLFFNQINNNLFSC        52
      9  LXXXXKJKXXXJIXXXXXXXXJJJIVXXXXXJQIXJGIXXC        49

>P1;RDSODF   Dihydrofolate reductase (EC 1.5.1.3) - Streptococcus faecium
No. 5.3 Score = 77  Quality = 40.314 HITS = 14 Mis. = 42 INDELS = 3 EXPECTED NO.  0.11E+01
        * *   *.     *    .          .     .*   *      ** .  *
     53  LSLPYRHIIVLTTQ KDFK VEKNAEVLHSIDELLAYAK DIPEDIYVSGGSRIFQALL       108
      1  LXLVXEJILXXXXKJKXXXJIXXXXXXXXJJJIVXXXXXJQIXJGIXXCXXJRIIHXXL        59

>P1;VGNZA2   Fusion glycoprotein - Respiratory syncytial virus (strain A2)
No. 5.4 Score = 76  Quality = 46.061 HITS = 13 Mis. = 35 INDELS = 7 EXPECTED NO.  0.14E+01
        * .*      *    *          *.       * *.   *   *     *
    511  E LLHNVNAGKSTTNIMITTIIIV   IIVILLS LIAVGLLLYCKA RSTPVTL       559
      6  EJILXXXXKJKXXXJIXXXXXXXXJJJIVXXXXXJQIXJGIXX CXXJRIIHXXL        59

>P1;SYEC     Thymidylate synthase (EC 2.1.1.45) - Escherichia coli
No. 5.5 Score = 74  Quality = 38.743 HITS = 13 Mis. = 42 INDELS = 5 EXPECTED NO.  0.20E+01
        * *.   .*      .                .      *   * ***  *
      5  LELMQK VLDEGTQKNDRTGTGTLSIFGHQMRFNLQDGF PLV TTKRCHL RSIIHELL        60
      1  LXLVXEJILXXXXKJKXXXJIXXXXXXXXJJJIVXXXXXJQIXJGIXXCXXJR IIHXXL        59

>P1;P3IV34   P3 protein - Influenza A virus (2 strains)
No. 5.6 Score = 74  Quality = 44.848 HITS = 14 Mis. = 33 INDELS = 6 EXPECTED NO.  0.20E+01
        **    . .    * .          *        .  .   *    . ..  *.
    373  ILRKATR RLIQLIVSGRDEQS   IAEAIIVAMVF SQEDCMI KAVRGDLN       419
      8  ILXXXXKJKXXXJIXXXXXXXXJJJIVXXXXXJQIXJGIXXCXXJRIIHXXLK        60

>P1;UZADP7   Terminal protein - Adenovirus 7
No. 5.7 Score = 74  Quality = 42.775 HITS = 14 Mis. = 36 INDELS = 7 EXPECTED NO.  0.20E+01
        * * **      .    *        .       *    *      *.. * *
    552  VQE ILRQAAV NDTE IDSVELSFRFK LTGPVAFTQRR QIQDVNR RVVAHASL       602
      4  VXEJILXXXXKJKXXXJIXXXXXXXXJJJIVXXXXXJQIXJGIXXCXXJRII HXXL        59

>P1;P3IV33   P3 protein - Influenza A virus (strain A/WSN/33)
No. 5.8 Score = 74  Quality = 44.848 HITS = 14 Mis. = 33 INDELS = 6 EXPECTED NO.  0.20E+01
        **    . .    *          *        .  .   *    . ..  *.
    373  ILRKATR RLIQLIVSGRDEQS   IAEAIIVAMVF SQEDCMI KAVRGDLN       419
      8  ILXXXXKJKXXXJIXXXXXXXXJJJIVXXXXXJQIXJGIXXCXXJRIIHXXLK        60

>P1;MWKW     Myosin heavy chain - Nematode
No. 5.9 Score = 74  Quality = 50.340 HITS = 12 Mis. = 30 INDELS = 5 EXPECTED NO.  0.20E+01
        * *   *          *       *.   *       .*.  **
     30  K KNVW IPDPEEGYLAGEITATKGD QVT  IVTAREMSVIQVTLK        71
     14  KJKXXXJIXXXXXXXXJJJIVXXXXXJQIXJGIXXCXXJRIIHXXLK        60

>P1;ODBY1 Cytochrome c oxidase (EC 1.9.3.1), polypeptide I - Baker's yeast mitochondrion (
No. 5.10 Score = 74  Quality = 54.412 HITS = 12 Mis. = 30 INDELS = 7 EXPECTED NO.  0.20E+01
        . **    . *    *           *       .   **      **
    402  Q ILGLNYNEKLAQ IQFWLIFIGA  NVIFFPM HFL GING MPRRI       443
      6  EJILXXXXKJKXXXJIXXXXXXXXJJJIVXXXXXJQIXJGIXXCXXJRI        54

>P1;VCVWM1 env polyprotein - Mink cell focus-forming murine leukemia viruses (2 clones)
No. 5.11 Score = 73 Quality = 47.712 HITS = 13 Mis. = 34 INDELS = 6 EXPECTED NO. 0.24E+01
        * *.         . .   *          .*      *.   * *   *
    593  LLLILL FGPWILN RLVQFIKDRISVVQA  LVLTQQYHQLK TIGDCKS R       639
      1  LXLVXEJILXXXXKJKXXXJIXXXXXXXXJJJIVXXXXXJQIXJGIXXCXXJR        53
```

```
>P1;R3ZM4    Ribosomal protein S4 - Maize chloroplast
No. 5.12 Score = 73  Quality = 42.690 HITS = 16 Mis. = 33 INDELS = 7 EXPECTED NO. 0.24E+01
         * . . **    .      *          *.      .*  .   *   * **
     87  LEMRLDNILFRLGM  AST IPGARQLVNHRHILVNGRIVDIP SFR CKP RDII       136
      1  LXLVXEJILXXXXKJKXXXJIXXXXXXXXJJJIVXXXXXJQIXJGIXXCXXJR II       55

>F1;DENCED NAD-specific glutamate dehydrogenase (EC 1.4.1.2) - Neurospora crassa (fragments
No. 5.13 Score = 73  Quality = 42.690 HITS = 16 Mis. = 33 INDELS = 7 EXPECTED NO.  0.24E+01
         * .. . . *   * .   *          .       ..   *     *..
    224  LEVISDRMFLAKATK NTKQ IYQDIIQVA   VSRHGPVIEVF DIEGSEEMRLV       273
      1  LXLVXEJI LXXXXKJKXXXJIXXXXXXXXJJJIVXXXXXJQIXJGIXXCXXJRII       55

>P1;VCVWEK   env polyprotein - AKV murine leukemia virus
No. 5.14 Score = 73  Quality = 47.712 HITS = 13 Mis. = 34 INDELS = 6 EXPECTED NO.  0.24E+01
         * *.        . .   *          .*      *.   * *   *
    622  LLLILL FGPCILN RLVQFIKDRISVVQA  LVLTQQYHQLK TIEDCKS R       668
      1  LXLVXEJILXXXXKJKXXXJIXXXXXXXXJJJIVXXXXXJQIXJGIXXCXXJR       53

>P1;PWECA    ATPase (EC 3.6.1.34) alpha chain - Escherichia coli
No. 5.15 Score = 70  Quality = 48.276 HITS = 16 Mis. = 28 INDELS = 7 EXPECTED NO.  0.41E+01
         * ..    . .    .        .*       .. *.   *   **.
     51  E MISLPGN RYAIALNLERDSVGA  VVMGPYA DLAEGMKVKCTG RIL        95
      6  EJILXXXXKJKXXXJIXXXXXXXXJJJIVXXXXXJQIXJGIXX CXXJRII        55

>P1;ISRBT    Triosephosphate isomerase (EC 5.3.1.1) - Rabbit
No. 5.16 Score = 70  Quality = 58.824 HITS = 12 Mis. = 28 INDELS = 3 EXPECTED NO.  0.41E+01
         . *    *          .      .* * * *   ...
     52  RQKLDPKIAVAAQNCYK  VTNGAFTGEISPGMIKDCGATWVV        92
     14  KJKXXXJIXXXXXXXXJJJIVXXXXXJQIXJG IXXCXXJRII        55

>P1;ISHUT    Triosephosphate isomerase (EC 5.3.1.1) - Human
No. 5.17 Score = 70  Quality = 58.824 HITS = 12 Mis. = 28 INDELS = 3 EXPECTED NO.  0.41E+01
         . *    *          .      .* * * *   ...
     52  RQKLDPKIAVAAQNCYK  VTNGAFTGEISPGMIKDCGATWVV        92
     14  KJKXXXJIXXXXXXXXJJJIVXXXXXJQIXJG IXXCXXJRII        55

>P1;IMBPBL   rexB gene_protein - Bacteriophage lambda
No. 5.18 Score = 70  Quality = 52.239 HITS = 12 Mis. = 33 INDELS = 2 EXPECTED NO.  0.41E+01
         *.          . .   *          **      ..  ..  *
     57  LIAALTFLIGSRTRRLAKIREYGYMTSVV IVYALSFVELG ALFFC       101
      3  LVXEJILXXXXKJKXXXJIXXXXXXXXJJJIVXXXXXJQIXJGIXXC        49

>P1;VGXR1S   Glycoprotein VP7 - Simian 11 rotavirus
No. 5.19 Score = 69  Quality = 53.488 HITS = 13 Mis. = 27 INDELS = 6 EXPECTED NO.  0.50E+01
         . *.   **    . .    *          *.      *   **
     16  IILLNY ILKSLTRIMDCI IYRLLFIIV   ILSPFLRAQNY GI        55
      1  LXLVXEJILXXXXKJKXXXJIXXXXXXXXJJJIVXXXXXJQIXJGI        46

>P1;WMFM9    90K protein - Alfalfa mosaic virus
No. 5.20 Score = 69  Quality = 45.098 HITS = 13 Mis. = 37 INDELS = 3 EXPECTED NO.  0.50E+01
         * .* * *.      *    .          *.      *    .     *
    237  LDIV E IIPDVSPTKPYEAVISGNDWMTLGRIIPTTPVPTIR DVFFSGLSR       286
      1  LXLVXEJILXXXXKJKXXXJIXXXXXXXXJJJIVXXXXXJQIXJGIXXCXXJR        53

>P1;HIBP14   Internal protein I - Bacteriophage T4
No. 5.21 Score = 69  Quality = 62.162 HITS = 12 Mis. = 26 INDELS = 5 EXPECTED NO.  0.50E+01
         * * ..    * *    .          ..      .* *
      7  LTSE VIKANGKRKQGP MKAKHISAAH  LISLVDGEEIK G        44
      3  LVXEJILXXXXKJKXXXJIXXXXXXXXJJJIVXXXXXJQIXJG        45

>P1;ACRYA1   Acetylcholine receptor protein, alpha chain precursor - Electric rays
No. 5.22 Score = 69  Quality = 36.316 HITS = 17 Mis. = 32 INDELS = 9 EXPECTED NO.  0.50E+01
         ** . .*  *     .          *      *.  ..      .*.   ..
     31  LVAN LLENYNK VIRP VEHHTHFVD   ITVGLQLIQLI SVDE VN QIVETNVR        79
      3  LVXEJILXXXXKJKXXXJIXXXXXXXXJJJIVXXXXXJQIXJGIXXCXXJRIIHXXLK        60

>P1;ANHU     Angiotensinogen precursor - Human
No. 5.23 Score = 68  Quality = 52.713 HITS = 12 Mis. = 28 INDELS = 6 EXPECTED NO.  0.59E+01
         * **    *    * .    .          .      .. *.
     88  LVLVAA KLDTEDKLRAAM VGMLANFLGF  RIYGMHS ELW GV       127
      1  LXLVXEJILXXXXKJKXXXJIXXXXXXXXJJJIVXXXXXJQIXJGI        46
```

```
>P1;TVCHLV  Virus-induced, kinase-related transforming protein (gag-env-erbB) - Chicken and
No. 5.24 Score = 68 Quality = 35.602 HITS = 17 Mis. = 36 INDELS = 6 EXPECTED NO.  0.59E+01
         . *. . ..          *            .      **  *.      *..*  *
    270  VQLITQ LMPYGCL  LDY IREHKDNIGSQ YLLNWCV QIAKGMNYLEERRLVHRDL       322
      1  LXLVXEJILXXXXKJKXXXJIXXXXXXXXJJJIVXXXXXJQIXJGIXXCXXJRIIHXXL        59

>P1;YNEC     Cyanate hydrolase (EC 3.5.5.3) - Escherichia coli
No. 5.25 Score = 68  Quality = 35.602 HITS = 14 Mis. = 40 INDELS = 5 EXPECTED NO.  0.59E+01
         * *    **    * *                 *      *    ..      *..   *
     12  LDLADA ILLSKAK KDLSFAEIANGTGLAEAFVTAALLGQ Q ALPA DAARLVGAKL        65
      1  LXLVXEJILXXXXKJKXXXJIXXXXXXXXJJJIVXXXXXJQIXJGIXXCXXJRIIHXXL        59

>P1;TVYUH    Kinase-related transforming protein (erbB) - Avian erythroblastosis virus
No. 5.26 Score = 68  Quality = 35.602 HITS = 17 Mis. = 36 INDELS = 6 EXPECTED NO.  0.59E+01
         . *. . ..          *            .      **  *.      *..*  *
    206  VQLITQ LMPYGCL  LDY IREHKDNIGSQ YLLNWCV QIAKGMNYLEERRLVHRDL       258
      1  LXLVXEJILXXXXKJKXXXJIXXXXXXXXJJJIVXXXXXJQIXJGIXXCXXJRIIHXXL        59

>P1;EDBEIC   Immediate-early protein - Cytomegalovirus
No. 5.27 Score = 68  Quality = 35.789 HITS = 14 Mis. = 39 INDELS = 5 EXPECTED NO.  0.59E+01
         * * .     *       .            ..      **    .      **    **
     54  LFPE LAEESLK TFER VTEDCNENPEKDVLAELVK QIKVRVDMVRH RIKEHMLK       106
      3  LVXEJILXXXXKJKXXXJIXXXXXXXXJJJIVXXXXXJQIXJGIXXCXXJRIIHXXLK       60

>P1;QQBE22   Probable membrane antigen gp220 - Epstein-Barr virus (strain B95-8)
No. 5.28 Score = 67  Quality = 46.853 HITS = 11 Mis. = 35 INDELS = 3 EXPECTED NO.  0.71E+01
         *    * *    * *    .            *      ..  *   *
     96  LLGAGELALTMRSK KLPINVTTGEEQQVSLESVDVYFQ DVF GTMWC       141
      1  LXLVXEJILXXXXKJKXXXJIXXXXXXXXJJJIVXXXXXJQIXJGIXXC        49

>P1;GNVWLV   pol polyprotein - AIDS virus LAV-1a (lymphadenopathy-associated virus)
No. 5.29 Score = 67 Quality = 38.728 HITS = 14 Mis. = 31 INDELS = 11 EXPECTED NO.  0.71E+01
         * *  .    . *    *            *      **   *  *   . *   *
     88  VLE EMSLPGRWKPKM IGGIGGFIK    VRQYD  QIL  IEICGH KAIGTVL       132
      4  VXEJILXXXXKJKXXXJIXXXXXXXXJJJIVXXXXXJQIXJGIXXCXXJRIIHXXL        59

>P1;VCLJLV env polyprotein precursor - AIDS virus LAV-1a (lymphadenopathy-associated virus)
No. 5.30 Score = 67  Quality = 39.181 HITS = 12 Mis. = 37 INDELS = 6 EXPECTED NO.  0.71E+01
         *    *              *            .      *.  **  *   ..*
    560  LLRAIE AQQHLLQLTVWG IKQLQARILA  VERYLKDQQLL GIWGCSG KLI       608
      1  LXLVXEJILXXXXKJKXXXJIXXXXXXXXJJJIVXXXXXJQIXJGIXXCXXJRII        55

>P1;VCLJVL   env polyprotein precursor - AIDS virus LV (lymphadenopathy virus)
No. 5.31 Score = 67  Quality = 39.181 HITS = 12 Mis. = 37 INDELS = 6 EXPECTED NO.  0.71E+01
         *    *              *            .      *.  **  *   ..*
    555  LLRAIE AQQHLLQLTVWG IKQLQARILA  VERYLKDQQLL GIWGCSG KLI       603
      1  LXLVXEJILXXXXKJKXXXJIXXXXXXXXJJJIVXXXXXJQIXJGIXXCXXJRII        55

>P1;VCLJA2   env polyprotein precursor - AIDS virus ARV-2 (AIDS-associated retrovirus)
No. 5.32 Score = 67  Quality = 39.181 HITS = 12 Mis. = 37 INDELS = 6 EXPECTED NO.  0.71E+01
         *    *       -      *            .      *.  **  *   ..*
    554  LLRAIE AQQHLLQLTVWG IKQLQARVLA  VERYLRDQQLL GIWGCSG KLI       602
      1  LXLVXEJILXXXXKJKXXXJIXXXXXXXXJJJIVXXXXXJQIXJGIXXCXXJRII        55

>P1;QXASBI   Hypothetical cobA intron protein - Aspergillus nidulans mitochondrion (SGC3)
No. 5.33 Score = 67  Quality = 35.263 HITS = 12 Mis. = 38 INDELS = 9 EXPECTED NO.  0.71E+01
         * *  *      *    *            *      *   *      **     **
    262  LTYE  LGIELSIKDVQLIYKIKKILG   IGIVSFR KIN EIEMVAL RIRDKNHLK       312
      3  LVXEJILXXXXKJKXXXJIXXXXXXXXJJJIVXXXXXJQIXJGIXXCXXJRIIHXX LK        60

>P1;RDHUB5   NADH-cytochrome b5 reductase (EC 1.6.2.2) - Human
No. 5.34 Score = 67  Quality = 40.606 HITS = 15 Mis. = 33 INDELS = 6 EXPECTED NO.  0.71E+01
         .*    * *    *           *.       *  *       ..*.  .*
    122  LLVYQGKGKFA  IRPDKKSNP   IIRTVKSVGMIAGGTGITPMLQVIRAIMK       170
      8  ILXXXXKJKXXXJIXXXXXXXXJJJIVXXXXXJ QIXJGIXXCXXJRIIHXXLK        60

>P1;GNVWVL   pol polyprotein - AIDS virus LV (lymphadenopathy virus)
No. 5.35 Score = 67 Quality = 38.728 HITS = 14 Mis. = 31 INDELS = 11 EXPECTED NO.  0.71E+01
         * *  .    . *    *            *      **   *  *   . *   *
     97  VLE EMSLPGRWKPKM IGGIGGFIK    VRQYD  QIL  IEICGH KAIGTVL       141
      4  VXEJILXXXXKJKXXXJIXXXXXXXXJJJIVXXXXXJQIXJGIXXCXXJRIIHXXL        59
```

```
>P1;GNVWH3   pol polyprotein - AIDS virus HTLV-III (T-cell leukemia virus, BH10)
No. 5.36 Score = 67 Quality = 38.728 HITS = 14 Mis. = 31 INDELS = 11 EXPECTED NO.  0.71E+01
         * *  .   . *   *          *     **   *  *   . *   *
    100  VLE EMSLPGRWKPKM IGGIGGFIK    VRQYD  QIL  IEICGH KAIGTVL       144
      4  VXEJILXXXXKJKXXXJIXXXXXXXXJJJIVXXXXXJQIXJGIXXCXXJRIIHXXL        59

>P1;GNVWA2   pol polyprotein - AIDS virus ARV-2 (AIDS-associated retrovirus)
No. 5.37 Score = 67 Quality = 38.728 HITS = 14 Mis. = 31 INDELS = 11 EXPECTED NO.  0.71E+01
         * *  .   * *   *          *     **   .  *   . *   *
     88  VLE EMNLPGKWKPKM IGGIGGFIK    VRQYD  QIP  VEICGH KAIGTVL       132
      4  VXEJILXXXXKJKXXXJIXXXXXXXXJJJIVXXXXXJQIXJGIXXCXXJRIIHXXL        59

>P1;QQBE21   Probable membrane antigen gp350 - Epstein-Barr virus (strain B95-8)
No. 5.38 Score = 67 Quality = 46.853 HITS = 11 Mis. = 35 INDELS = 3 EXPECTED NO.  0.71E+01
         *   * *   * *   .          *     ..  *   *
     96  LLGAGELALTMRSK KLPINVTTGEEQQVSLESVDVYFQ DVF GTMWC       141
      1  LXLVXEJILXXXXKJKXXXJIXXXXXXXXJJJIVXXXXXJQIXJGIXXC        49

>F1;UZADP2   Terminal protein precursor - Adenovirus 2 (fragment)
No. 5.39 Score = 67 Quality = 60.360 HITS = 11 Mis. = 29 INDELS = 4 EXPECTED NO.  0.71E+01
         * * **    .    .         .*     **   *
    580  VQE ILRQAAVNDTEIDSVELSFRFKLTGPVVFTQRR QIQ EI       620
      4  VXEJILXXXX KJKXXXJIXXXXXXXXJJJIVXXXXXJQIXJGI        46

>P1;VCLJH3 env polyprotein precursor - AIDS virus HTLV-III (human T-cell leukemia virus, BH
No. 5.40 Score = 67  Quality = 39.181 HITS = 12 Mis. = 37 INDELS = 6 EXPECTED NO.  0.71E+01
         *   *            *          .      *. **  *   ..*
    555  LLRAIE AQQHLLQLTVWG IKQLQARILA  VERYLKDQQLL GIWGCSG KLI       603
      1  LXLVXEJILXXXXKJKXXXJIXXXXXXXXJJJIVXXXXXJQIXJGIXXCXXJRII        55
```

APPENDIX B

Publications

The following publications have been included in accordance with paragraph 7.7 of the University of Edinburgh regulations governing the submission of thesis. Permission to do so has been obtained from the co-authors and publishers.

PARALLEL COMPUTING 85
M. Feilmeier, G. Joubert and U. Schendel (eds.)
Elsevier Science Publishers B.V. (North-Holland)

IMPLEMENTATION OF INEXACT STRING MATCHING ALGORITHMS ON THE I. C. L. DAP

A. Lyall, C. Hill*, J.F. Collins and A.F.W. Coulson

Department of Molecular Biology and Department of Computer Science*,
University of Edinburgh,
Edinburgh, EH9 3JR
Scotland, U.K.

Searches for similarities amongst biosequences commonly make use of the Needleman-Wunsch-Sellers algorithms which are known to produce optimal alignments. We describe implementations of the N-W-S algorithms for the I. C. L. DAP and show that the multiprocessor and interconnection architecture of the machine allow inexact string-matching algorithms to be computed with high efficiency.

INTRODUCTION

Analysis and comparison of sequences of proteins and nucleic acids is an important problem for molecular biologists (Sankoff and Kruskal, 1983). While many routine analyses can be done on small workstations or main frame computers, there is a growing need for analytical tools which can handle queries with respect to entire collections of sequence data. The databases are increasing rapidly in size, though they are small by database standards - ca. 650,000 amino-acid residues in protein sequences, ca. 4×10^6 bases in the nucleic acid collections.

One key feature of both these biological polymers is that they are linear, and directed (i.e., two sequences can only be compared in one orientation). An early report of the application of the I.C.L. 64*64 distributed array processor (DAP) confirmed that multiprocessor array systems showed promise for these analyses, and the relatively high speed of 1-bit logical operations (compared to integer or real operations) means that the most powerful features of the DAP can be harnessed for these applications (Collins and Coulson, 1984).

This paper reports on the implementation of inexact string-matching algorithms on the DAP. These algorithms have been summarised by Sellers (1980), and implementations of some of them are found in most packages used for sequence analysis in molecular biology.

We consider three main variations of the inexact string-matching problem:

i) To find the alignment of two sequences such that the number of changes needed to interconvert one to the other is minimal, and to find this 'distance' which, if a simple metric scoring all changes (mismatches, insertions or deletions) with the same weight is used (the Levenshtein metric), can be considered the 'evolutionary distance' between them. Other metrics can be used, provided that the triangle inequality holds.

ii) To find the best location of one test string within a (potentially much longer) string. This problem resembles the first, but there is no penalty attached to unpaired characters in the main string overhanging the test sequence.

iii) To find those regions in the test string which show the greatest similarity to regions in the main string. This problem is of immense importance to the understanding and detection of features

shared by proteins or nucleic acids with biological properties in common.

PROBLEM 1

The algorithm for finding the alignment of two sequences A and B which reflects their 'evolutionary distance' can be stated:
Consider two strings A and B, of lengths m and n respectively:
There is a distance matrix, d(m,n), which can be completed such that each cell (i,j) contains the minimum number of changes to convert the substring A(1-i) into the substring B(1-j). The cell (m,n) therefore will contain the minimum 'evolutionary distance' between the two strings. The value entered into (i,j) is

$$\min \begin{cases} d(i-1,j-1) & \text{if } A(i) \text{ matches } B(j) \\ d(i-1,j-1) + & \text{mismatch penalty} \\ d(i-1,j) + & \text{insertion penalty} \\ d(i,j-1) + & \text{insertion penalty} \end{cases}$$

The aligment can be reconstructed by retracing legitimate steps back from (m,n) to (0,0). (A dummy first row and column is normally added to cope with penalties associated with terminal overhanging regions.) We confine ourselves to the use of the Levenshtein metric in the following discussion, for simplicity.

		A	B	C	A	C	B	A	C	A
	0	1	2	3	4	5	6	7	8	9
A	1	0	1	2	3	4	5	6	7	8
C	2	1	1	1	2	3	4	5	6	7
B	3	2	1	2	2	3	3	4	5	6
A	4	3	2	2	2	3	4	3	4	5
B	5	4	3	3	3	3	3	4	4	5
C	6	5	4	3	4	3	4	4	4	5
B	7	6	5	4	4	4	3	4	5	5
C	8	7	6	5	5	4	4	4	4	5
A	9	8	7	6	5	5	5	4	5	4

Table 1. Distance Matrix, showing path of possible optimal alignments.

Three implementations of this algorithm on the DAP have been investigated.

Mapping I.

As the information needed to calculate d(i,j) includes d(i-1,j-1), d(i-1,j) and d(i,j-1), it is possible to calculate the values of all elements along the minor diagonal simultaneously, up to the number of processors available (4096 for the 64*64 DAP). It is not necessary to record all the values of d calculated, other than those needed to calculate each set of diagonal elements: however, to enable the alignment to be found without recalculating permissible steps in a traceback procedure, the logical states representing each legitimate step can be stored in sufficient numbers to recover the history of paths that lie within 275 steps of the main diagonal. This has proved perfectly adequate for all alignments tested, which, while they may contain many insertions and deletions, stray from the main diagonal by only the difference at any step between the number of horizontal and vertical steps taken. This tends to be a much smaller number than the number of horizontal or vertical steps themselves.

The alignment of two 4096 long strings can be accomplished in about 7 seconds DAP time: we have no meaningful comparison for speed-up available as problems of this scale are beyond the scope of most serial implementations of this algorithm.

There are three comments to be made: first, the performance of the program is basically O(m+n), as that represents the number of diagonals whose values must be

calculated: second, the performance is data-independent: and, third, the maximum degree of parallelism involved is the lesser of m or n, provided it does not exceed 4096, and the average parallelism is mn/(m+n), or, at best, half the maximum number of processors available. These limitations led to the consideration of other mappings of this problem onto the DAP.

Mapping II.

The second method studied was to assign the d matrix row by row. This is convenient because a comparison can be carried out with a single character from the test sequence broadcast to all processors simultaneously. The first stage is to set d(i,j) to the lowest value given by either a vertical step (including the penalty for an insertion) or by a permissible diagonal step (corresponding to either a match or a mismatch), using the correct values for the (i-1)st row already calculated. The values of d in this row will be correct unless a horizontal step can lead to an improved value. These corrections are made by manipulating a set of logical masks in the following manner:

i) Identify all places where a horizontal step can improve the current score by testing whether d(i,j)>d(i,j-1) + insertion penalty:
ii) Identify all places where the current value will need correction if its lower neighbour's value is improved:
iii) Use a recursive doubling process to extend upwards all positions marked in test i) through any contiguous regions marked in test ii). A maximum of 12 stages can propagate the logical mask through 4096 positions.
iv) Use the mask to allow the correction of the current scores by one penalty.

As the values calculated for the previous row are already consistent, the errors in values set by the vertical or diagonal move cannot exceed the single penalty, and the new row is now correctly set. The history of the permitted path steps are stored in logical form under each processor, which restricts the length of the test sequence that may be used. On the other hand, the performance is O(n), the length of the test sequence, which is a considerable improvement.
In unbalanced problems, this mapping is clearly advantageous to Mapping I. On the other hand, it is still insensitive to the data involved. It is also clear that much of the computation is assigning values to d in positions that cannot possibly lie on the best path, so that the improvement in parallelism is not all translated into improved performance characteristics.

Mapping III.

A further mapping was investigated: by calculating the values of the d matrix in parallel along the main diagonal. This is conveniently done by matching all characters in the two strings, and then setting cell (i,i) to the sum of the number of mismatches in processors 1 to (i). This gives the correct values only if no insertion or deletion steps occur in the optimal pathway. However, if m>n, the value of d(n,n) + (m-n), gives an estimate for the current path which limits the diagonals in d which need to be examined to check whether a better path can be found. A path which enters a diagonal j from the main diagonal must have a minimum distance of 2j, even if all characters on this diagonal match. By setting additional diagonals as described, and then improving the values in adjacent diagonals where a vertical or horizontal step is able to connect better aligned regions, the value of the best alignment can be progressively refined, until it reaches a value better than the minimum value that can be reached by examining diagonals even further from the main diagonal. At this point, the algorithm terminates with the best value for the alignment.
Values have only been computed for d in regions that are potentially on the optimum alignment; in general, the performance is complex. However, the program is sensitive to the data; in the best case of a perfect matching pair of sequences, the algorithm terminates after the first cycle of execution (i.e. O(1)). This mapping may be useful for screening for close similarities between two sequences.

PROBLEM 2.

The location of the region in which a test sequence may be aligned within a larger sequence differs in two ways from Problem 1:

i) there is no penalty attached to regions of the larger sequence which overhang the test sequence; and

ii) all locations that produce the locally best alignments should be recorded, even if they are not all of equal merit.

In practice, this problem is posed when the longer sequence is not confined to a single sequence, of length equal or less than the number of processors available, but where it takes the form of all the collected sequences in the database. The entire set of protein sequences (650,000 characters) can be loaded and treated as a single object for this alignment problem. The best alignments are therefore of considerable biological significance, since they are produced from an exhaustive searching process.

		A	B	C	A	C	B	A	C	A
	0	0	0	0	0	0	0	0	0	0
A	1	0	1	1	0	1	1	0	1	0
C	2	1	1	1	1	0	1	1	0	1
B	3	2	1	2	2	1	0	1	1	1
C	4	3	2	1	2	2	1	1	1	2

Table 2. Distance Matrix, showing 2 families of optimal alignments arising at different locations.

For this problem, Mapping II has obvious advantages. The entire first row of the distance matrix is set to 0 (Table 2), and row-by-row processing continues till on the last cycle, the values represent the scores of all the legitimate paths that can exit on the final row. The number of paths that have to be traced back is first reduced by using minima or plateau detecting tests; by collecting and comparing the start points, paths can be further restricted to a single one from each start point. The selected set of paths are then traced back through the histories that have been set on the forward pass, and all logical values not associated with a desired path are pruned out. In addition, all horizontal runs are truncated to a single bit; this allows all paths to be represented by the same number of bits, which can be packed inside the DAP into larger objects, each of which on return to the host level computer, can be used to drive a printer output routine to produce, in their correct alignment, a pair of characters from either sequence, or one character and a gap on the alternate sequence. This compact method of reporting the best paths (additionally sorted in the DAP for best scores) has been a significant improvement over any serial routine available to us. This problem also benefits from the fact that, in making the corrections for possible horizontal steps, the recursion used to generate the mask used can always be stopped after $\log_2$ (i) stages on the ith row, and is thus shorter than this part of the program for Mapping II, Problem I.

PROBLEM 3.

The third problem is significantly different, but we can apply the previous lessons to show that Mapping II, row-by-row processing, is the obvious choice. However, we are now producing a similarity matrix, and the regions of interest can start and terminate anywhere inside it. The scoring credits matches with a positive score, and all errors in alignment with a negative penalty; cells whose value falls below zero are reset to zero (see Table 3). This problem assumes such a size when it is desired to use the database as one of the strings, that storage of path histories has to be abandoned. Instead, a history of the starting positions, the maximum score reached, the location of the maximum score and the current score are passed forward at each cycle. If two alignments could give the same score at one location, then the history of the aligment with the better maximum score is kept.

		A	B	C	A	C	B	A	C	A
	0	0	0	0	0	0	0	0	0	0
A	0	1	0	0	1	0	0	1	0	1
C	0	0	0	1	0	2	1	0	2	1
B	0	0	1	0	0	1	3	2	1	1
A	0	1	0	0	1	0	2	4	3	2
B	0	0	2	1	0	0	1	3	3	2
C	0	0	1	3	2	1	0	2	4	3
A	0	1	0	2	4	3	2	1	3	5
C	0	0	0	1	3	5	4	3	2	4
A	0	1	0	1	2	2	2	3	1	3

Table 3. Similarity Matrix, showing regions of greatest similarity detected.

It is necessary to report intermediate results to temporary storage, for those paths which have improved their maximum score. This store can hold 4096 results and, should additional results be acquired, the class of results with the least score is overwritten. Typically, the best alignments of regions in a test string of 100 characters can be recovered from the 650,000 protein database in ca. 70 DAP secs. and often turn out to be a relatively small number above the initial threshold value of interest. This has considerable potential in the field of pattern detection and characterisation for both proteins and nucleic acids. Character matching can also be extended to degenerate matching schemes.

DISCUSSION

The specific features of proteins and nucleic acids which made sequence analysis so fascinating, are that these polymers can be represented by one-dimensional strings of characters which carry the information, if correctly interpreted, which determines their chemical and ultimately their biological properties. Whatever system of analysis is used in the future, it is clear that a multi-processor exhibiting one-dimensional connectivity has considerable advantages over single-processor systems, as we have shown in the implementation of the inexact string-matching algorithms on the DAP.

While sequence analysis algorithms are all likely to have sufficient inherent parallelism to benefit from a multi-processor such as the DAP, programming to exploit that parallelism should include a variety of implementations, since, as we show here, the run-time characteristics can vary widely. For each problem, choice of the best implementation will depend on the data to be processed in the manner we have demonstrated. Each mapping has specific advantages under certain conditions.

ACKNOWLEDGEMENTS

We should like to thank Professor S. Michaelson and Dr. M. Jerrum (Department of Computer Science, Edinburgh University) for their help and advice. We should also like to thank Dr. S. Reddaway and Dr. D. Hunt (I.C.L.) for their insights into the DAP.

REFERENCES

(1) Sankoff, D. and Kruskal, J.B. (eds.) "Time Warps, String Edits and Macromolecules: The Theory and Practice of Sequence Comparison" (Addison-Wesley Publ. Co., Inc., Reading, Mass., 1983).

(2) Collins, J.F. and Coulson, A.F.W. "Applications of Parallel Processing Algorithms for DNA Sequence Analysis". Nucleic Acids Research, 1984, 12, 181-192.

(3) Sellers, P.H. "The Theory and Computation of Evolutionary Distances: Pattern Recognition". Journal of Algorithms, 1, 1980, 358-373.

Protein and nucleic acid sequence database searching: a suitable case for parallel processing

A. F. W. COULSON*, J. F. COLLINS AND A. LYALL
Department of Molecular Biology, University of Edinburgh, Edinburgh EH9 3JR

Sequence analysis of protein and nucleic acid databases by exhaustive string-matching algorithms is effectively implemented on large processor-array machines, such as the I.C.L. DAP. An improved method of assessing the significance of the best alignments for proteins is described. Examples involving the cystic fibrosis antigen and* Drosophila *vitellogenins illustrate the usefulness of this approach.

Received June 1987

1. INTRODUCTION

Molecular biology has been revolutionised by the development of fast sequencing techniques for nucleic acids. The rate of acquisition of protein sequence data has correspondingly accelerated, and molecular biological research now depends heavily on gene cloning, sequencing and the translation of open reading frames (which code for possible proteins using the triplet genetic code). This has led to the urgent need for adequate comparative sequence analysis, to promote the efficient use of other research resources.

Proteins and nucleic acids (the genetic material) are linear polymers whose sequences may be represented by character strings, with a 20-letter alphabet for proteins (denoting the individual amino-acid residues), and a 4-letter alphabet for nucleic acids (denoting the individual bases in the DNA or RNA polymers). The international database collections of sequences are prime resources for molecular biological research. These databases are currently small; the protein database has *c.* 1 000 000 characters of sequence, and the genetic database has *c.* 10 000 000 bases of sequence information, but already the task of searching them has led to the development of a number of approximate methods for making comparisons. However, the application of the exhaustive inexact string-matching algorithms, reviewed by Sellers,[1] has been beyond the capacity of many workstations and mainframe computers. The situation will deteriorate further as the databases are growing exponentially, doubling in size every two years or less.

We report here our experience using the I.C.L. 64 × 64 Distributed Array Processor (DAP)[2] for exhaustive database searching. DAP programs for inexact string-matching have been developed by Lyall *et al.*;[3] of these the most valuable, especially for the case of novel proteins, has implemented the 'Best Local Similarity' algorithm of Smith and Waterman.[4]

It is common for the sequence of part or the whole of a protein to be determined before its function is known. Prediction of function from the analysis of secondary (i.e. local folding along the chain) and tertiary (i.e. assembly of folded regions into a stable structure) structures cannot yet be achieved, and the most profitable approach has been to find analogies with or within the sequences of known proteins.[5]

* Principal author, to whom correspondence should be addressed.

As the databases grow larger, the number and the scores of alignments with unrelated protein subsequences increase. It is therefore an important issue to determine the significance of the best alignments found, and we describe here a method which is applicable precisely because the whole database has been searched.

2. ALGORITHM AND DAP IMPLEMENTATION

The 'Best Local Similarity' algorithm[4] is related to other algorithms for sequence comparison and alignment (see Ref. 1), and uses dynamic programming techniques to track the best paths through a match matrix. Each path represents an alignment of the whole or part of the two sequences being compared. At each point in the matrix, the best path is determined by the best cumulative score of paths already running, such that

$$Score(i,j) = MAX(Score(i-1,j-1) + Sim(aa_i,aa_j),\ Score(i-1,j) + gap\ penalty,\ Score(i,j-1) + gap\ penalty)$$

where $Sim(aa_i,aa_j)$ is the similarity score for amino-acids aa_i in one sequence and aa_j in the other.

The cumulative scoring is justified in the Dayhoff[6] analysis by the use of a log(odds) table, the odds being those that a particular pair of residues is found in a significant alignment rather than in a random selection of two residues from the whole population of residues. The figures were derived by Dayhoff from the 71 families of aligned proteins then available. She also described how to produce a series of log(odds) tables corresponding to different evolutionary spans, referred to as the PAM tables (1 PAM corresponds to the appearance of 1 substituted amino-acid residue in a pair of related proteins, per 100 residues aligned). The gap penalty is set to limit the proportion of gaps in the alignments reported to a (subjectively) appropriate level, and to maintain the triangle inequality. Paths are allowed to start at any location inside the match matrix from zero, and are tracked until the score declines to 0 or less, or competing paths block further path extension. Cells scoring less than 0 are reset to 0 before the computation is extended. The best local alignments are found from the maximum path scores, and tracked back through the matrix to their origin.

The DAP host sets up a 2Mbyte DAP core image, and the results are returned as data blocks to the host after

the DAP program has terminated. As the sequence alignments are of unknown size at the outset, the program was designed to store the essential details of all runs in a fixed format within the DAP. Implementation with the complete match matrix in main memory is impossible, and for the purposes of coding the algorithm in the DAP, store limitations require that all results be acquired in a single forward pass from data corresponding to a small part of the match matrix, rather than from a double pass through the whole match matrix.

Two rows of the match matrix are used, each 4096 elements long (the DAP long vector length), together with two sets of path data, representing the previous and the current row and path data. The current row is updated; assignments of the score are carried out in parallel by matrix operations for score extensions with diagonal or vertical steps in the paths. The scoring for paths best extended horizontally can then be determined, using recursive doubling combined with logical masks to detect whether further improvements have been made at each cycle. A maximum of 12 iterations completely exhausts all the horizontal path extensions in each segment of the comparison matrix. Paths with an improved maximum score which exceeds a threshold value are reported into the results registers. The results are processed:

(i) by overwriting any existing inferior path details starting from the same coordinates;
(ii) or by writing the result into a free location;
(iii) or, if there are no free locations, by
  (*a*) discarding all path details from the lowest class currently stored, marking these locations as available and incrementing the threshold to the score of the discarded class;
  (*b*) if the path score to be stored exceeds the new threshold, returning to (ii).

The current paths and details are then written into the 'previous' row and details registers. As the database is considered in 4096-long segments, details of paths leaving at the end of each row are stored, to be made available at the beginning of each row in the next segment of the database. Paths can therefore be tracked wherever they may occur within the match matrix for the whole comparison process.

An advantage of this strategy is that the results accumulated are guaranteed to be the best available by this algorithm, and can be sorted within the DAP; a key is returned to allow host generation of the alignments in order of diminishing score, under user control. In essence, any run can be reconstructed if the coordinates of the start and stop positions are known. However, serial alignment programs calculate possible path states at many locations never included in any path; in the DAP, therefore, the maximum deviations above and below a diagonal path from the start of each path being traced have been added to the set of path details. This provides the host with additional information defining the narrowest band within which each alignment must lie. In the cases of highly related sequences with few gaps, there is a major saving in time in generating the alignments.

The DAP search of version 11 of the NBRF (National Biomedical Research Foundation) protein database, containing 1066790 amino-acids, takes *c.* 1.8 DAP second per residue in the query sequence.

3. SIGNIFICANCE

The assessment of the significance of total or part alignments between genes or proteins has usually be approached by asking whether the query sequer produces significantly better alignments than sequenc derived by randomly reordering the query sequenc Two points arise here; the time to search a database significant even on the DAP, and the investment of mc CPU time to discover the statistical behaviour of randc sequences each time is not attractive. Secondly, t database is not a collection of sets of randomly order characters; in general, proteins share characteris structure features in the natural folded state (for instanc the alpha-helix, beta sheet and various types of turn) a these are reflected in short-range ordering within t protein sequence. Therefore, real proteins are likely contain regions of better local similarity with each oth than with random rearrangements of the same residu

The DAP program returns data for the 4096 be alignments, which form in most searches the upper e of a much larger distribution of scores. The database highly diverse, and no single family of related proteins represented much more than 100 times. Hence t majority of the best results will be of alignments betwe regions of the query sequence and proteins in t database which have no close connection; in oth words, these are alignments representing the noise-lev in comparisons with a large collection of unrelat proteins. If we can determine the underlying shape of th distribution, we can predict the frequency of occurren of an alignment of any score arising from unrelat proteins, and so establish the likelihood that a particular alignment belongs to this class or not.

We can regard the alignments reported by the 'Be Local Homology' algorithm as a series of aligned pa which may be scored positively or negatively, a unmatched residues in either strand, where gaps ha been introduced (under penalty). Each reported alig ment starts with a positive score, and terminates whe the cumulative score reaches a maximum.

The analysis of significance does not require knowled of the complete distribution of scores. All that is need is a model for the expected value of the ratio of t number of alignments scoring $(n+1)$, to the number alignments scoring (n). The most important route which an alignment could improve from a score of n $n+1$ (or beyond) is from the position at which t current maximum score n was reached. The probabili that, within a region of the match matrix through whi the path can be extended with net loss of x in score, the is a matching region from which a net gain of $(x+1)$ c subsequently be obtained, must be independent or near independent of the current maximum score, once this h exceeded a low value. This implies that the distributi of path scores will decline exponentially, and this indeed found experimentally to be the case.

The lower-scoring 98% of the recorded alignmer were therefore analysed by fitting the best line to t log(no. of alignments) *v.* score with excellent resul providing parameters to estimate the expected frequen of any scoring alignment, as well as standard deviation f the distribution about the line. The high-scoring outlyi alignments can be tested for their significance by seei how well they conform to this distribution; especial

how many standard deviations they are above the expected frequency, thus expressing the likelihood of any alignment occurring with unrelated proteins.

However, it must be emphasised that any alignment can potentially provide the molecular biologist with useful information, and between 50 and 200 are normally collected for display.

4. PATTERN DETECTION AND SEARCHING

Additional processing of the results can provide further useful displays: for example, when a query sequence is related to a number of sequences in the database, the alignments can be accumulated, or 'learnt', so that it is possible to display a large number of alignments with respect to the query sequence. This is a sensitive method of finding conserved residues of short-sequence features, which are difficult to detect in individual alignments. The learning process can be guided by different criteria, to allow the disclosure of patterns relating different types of sequence within the same set of alignments. Each search for patterns can then be reinforced by re-searching the database with the pattern detected, to establish and refine its ability to discriminate sequences fitting the pattern from the bulk of unrelated proteins.

The ability to detect patterns can be extended by using a more general method of describing a pattern. In principle, a pattern search could be carried out with specific values for all matching possibilities at each position. However, we have provided simple general extensions which have been of considerable value, defining four types of character:

(i) normal characters, matched using the similarity table values, and attracting a gap penalty if unmatched in an alignment;

(ii) residues which attract scores from the similarity table used, but which must be matched with a positive score in any reported alignment;

(iii) residues which may be matched with any residue, with zero score, but which cannot be unpaired without attracting the gap penalty; and

(iv) residues which may match any character without preference, with zero score, and which may be omitted without penalty.

This has provided a flexible and versatile pattern-detection and searching tool. A specific advantage that distinguishes this mode of pattern detection from the 'regular expression' pattern-searching programs is that, in addition to exact fits to the specified pattern, other near-fits are scored and can be reported, including those with a wider range of character substitution or spacing than envisaged in the specification of the pattern. That is, it is possible to discover unexpected ways in which the pattern is variable, without explicit definition of these alternatives.

Such searches are providing interesting results with complex polyproteins deduced from viral gene sequences, where the ability to detect conserved features may help define regions of importance in the normal function of the polyprotein in the viral lifecycle, helping to define these features for further research.

For example, a pattern for a zinc-ion binding site has been proposed[8] to be $CX_{2-4}CX_{2-15}$(C or H)X_{2-4}(C or H), where C stands for the amino-acid cysteine, H for the amino-acid histidine, and X stands for an unspeci amino-acid. The database can be searched with pattern in *c.* 60 DAP seconds and the best match regions readily listed to verify this hypothesis.

5. EXAMPLES

5.1 Cystic-fibrosis associated antigen

A gene cloned by Dorin *et al.*[9] coded for a protein fo at elevated levels in the serum of cystic fibrosis patie and carriers. The gene was translated into the pro sequence, and the database search found that there w significant homologies with calcium-binding prote using the 250 PAM similarity table. The search repeated with a variety of PAM tables, and the maxim significance was found with the 80 PAM table. The se results is shown in Fig. 1. The best alignment (Fig. 2) an expected frequency of 1.7×10^{-10} (44 S.D.s ab expectation), and clearly indicated that the alignm belonged to a different class from those forming the b of the reported results, and which arise from biologic unrelated proteins.

Twenty out of the next 21 alignments were v proteins all known to bind calcium ions, and this would have indicated the same property in the cy fibrosis antigen, even in the absence of the prot giving very high-scoring alignments.

5.2 Vitellogenins in *Drosophila melanogaster*

Garabedian *et al.*[11] have shown that the sequences three storage proteins (YP 1, YP 2 and YP 3) found the eggs of the fruit fly share a long region of hig conserved sequence. A database search revealed that

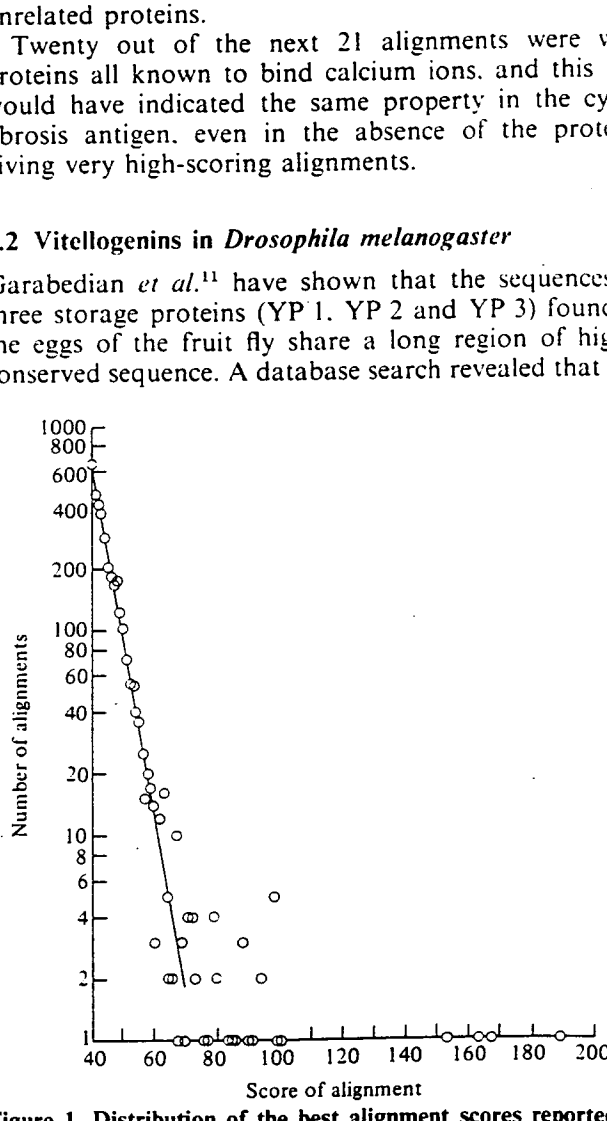


Figure 1. Distribution of the best alignment scores reporte the DAP protein database searching program, using the cy fibrosis-associated antigen sequence, and the similarity t calculated for 80 PAMs.[6] The line of best fit to the lower 9 of the results is shown. The highest-scoring alignment woul expected at a frequency of 1.7×10^{-10}, 44 S.D.s from expected value.

```
     .*** *. ..*.*.* *  *. .  . .** **.**  ..  * ** *  **** . ** *.***. .**
2'   SELETAMETLINVFHAHSGKEGDKYK LSKKELKELLQTELSGFLDAQKDADAVDKVMKELDEDGDGEVDFQEYVVLV  78
3    TELEKALNSIIDVYHKYSLIKGN FHAVYRDDLKKLLETECPQYI RKKGAD V W FKELDINTDGAVNFQEFLILV  75
```

Figure 2. Alignment of the amino-acid sequences of the cystic fibrosis antigen (lower sequence) and the bovine s-100a alpha protein chain (upper sequence), in the highest-scoring alignment reported by the 'Best Local Homology' algorithm. The one-letter notation for amino-acids is that recommended by the IUPAC-IUB Commission on Biochemical Nomenclature (the *Biochemical Journal*, 113, 1 (1969). Numbers at the ends of the sequence segments indicate the position within the entire protein chain of the aligned residues. Identical pairs of residues are starred; pairs scoring positively but non-identical are dotted; all others attract penalties.

```
     .*.**.....* ** ** . *  *  . ********      *. * *  .** *** *** ..
146  VHVIGHSLGSHAAGEAGRR T  NG TIERITGLDPAEPCFQGTPE LVRLDPSDAKFVDVIHTDAAP  208
239  IHLIGQGISAHVAGAAGNKYTAQTGHKLRRITGLDPAKV LSKRPQILGGLSRGDADFVDAIHT ST   303

     . .*   . .  * .* .*** .  .** .. *  ...      .... *. *.* *. * *. **
209  IIPNLGFGMSQTVGHLDFFPNG GKQMPGCQKNI  LSQIVD I DGIWEGT RDF  VACNHLRSYK  268
304  F A MGTPI R CGDVDLYPNGPSTGVPGSENVIEAVARATRYFAESVRPGSERNFPAVPANSLKQYK  367
```

Figure 3. The best scoring alignment found between the vitellogenin YP 3 from *Drosophila melanogaster* (lower sequence), and pig lipase (upper sequence). Expected frequency for an alignment with this score: 1.17×10^{-7}; 53 S.D.s above expectation.

conserved region otherwise aligned best with part of a sequence from a pig digestive lipase, and that this similarity was highly significant (expected frequency 1.17×10^{-7}; 53 S.D.s above the predicted expectation) (Fig. 3). However, one residue thought to play a role in the catalytic activity of the lipase was not matched in the alignment (Fig. 4). The vitellogenins, in fact, do not show lipase activity. It was postulated that the similarity found is related to the ability of these proteins to bind lipids or lipid-like materials. Subsequent tests have shown that the vitellogenins strongly bind a natural lipid-like derivative of the insect hormone ecdysone, which is involved in the control of embryonic development. The embryo breaks down vitellogenin at the stage when the hormone is known to be released, and it now appears that these proteins may have an important role in regulating embryonic development.

```
.*.**.....* ** ** . *
VHVIGHSLGSHAAGEAGRR T
IHLIGQGISAHVAGAAGNKYT
```

Figure 4. Alignment of vitellogenin YP 3 (lower sequence) and pig lipase (upper sequence), in the region of the active serine (S) residue in lipase. The serine is aligned against a glycine (G) residue, which would not be expected to substitute functionally for the serine.

6. DISCUSSION

The nature of protein and nucleic acid sequences makes them immediately suitable for processing by network-connected arrays of processors. The efficiency of the DAP 1-bit processors is particularly high, since much of the arithmetic can use 1- to 2-byte variables, and there is a large component of logical operations. Data movements are predominantly long vector shifts, which are slightly more complex than simple vector shifts. The published account of sequence comparison on a Cray-1 machine[10] shows that the DAP can match this more powerful machine (and at a fraction of the cost).

As the database grows, the biologist is as interested in increasing the variety of known sequences as in providing new examples of known protein types. Increasingly, more and more of the database for proteins is likely to be hypothetical proteins, inferred from gene sequences, whose physical and chemical properties and biological role have not been observed.

For this reason, results which have strong statistical significance are only part of the value the biologist can draw from these searches. If the query sequence is related, distantly, to a database sequence which is unique, that fact may be enough to generate a biological hypothesis which can then be tested further. The value to the biologist of having a complete and exhaustive search carried out is much more, therefore, than finding the single best alignment; valuable information may be gained from the presence of groups of related alignments and even from a single alignment of low statistical significance. This fact differentiates the biological database search problem from more conventional database searching problems.

It will be important to maintain some facility which can fulfil this search role in the near future, while the sizes and rate of increase in the databases can still be handled. When the promised improvements in sequencing technology are implemented, and gene sequences can be accumulated at 1 000 000 bases per day, a new crisis will have to be faced if this information is going to be of a significant use to the biological community.

Acknowledgements

We should like to thank Professor S. Michaelson (Department of Computer Science) for his help and advice; Dr S. Reddaway and Dr D. Hunt (I.C.L.) for their insights into the DAP; Carolyn Bucholtz (C.S.I.R.O., Sydney) for assistance. A.L. gratefully acknowledges an SERC CASE award with I.C.L.

REFERENCES

1. P. H. Sellers. The theory and computation of evolutionary distances: pattern recognition. *Journal of Algorithms* **1**, 359–373 (1980).
2. P. Flanders, D. J. Hunt, S. Reddaway and D. Parkinson. Efficient high speed computing with the Distributed Array Processor. In *High Speed Computer and Algorithm Organisation*, edited D. J. Kuck, D. H. Lawrie and A. H. Sameh, pp. 113–120. Academic Press, London (1978).
3. A. Lyall, C. Hill, J. F. Collins, and A. F. W. Coulson. Implementation of Inexact String Matching Algorithms on the I.C.L. DAP. In *Parallel Computing '85*, edited M. Feilmeier, G. Joubert and U. Schendel, pp. 235–240. North-Holland, Amsterdam (1986).
4. T. F. Smith and M. S. Waterman. Identification of common molecular subsequences. *Journal of Molecular Biology* **147**, 195–197 (1981).
5. J. F. Collins and A. F. W. Coulson. Molecular sequence comparison and alignment. In *Nucleic Acid and Protein Sequence Analysis: A Practical Approach* edited M. Bishop & C. Rawlings, pp. 323–358. I.R.L. Press, Oxford (1987).
6. M. O. Dayoff, R. M. Schwartz and B. C. Orcutt. A mo of evolutionary change in proteins. In *Atlas of Pro Sequence and Structure*, vol. 5, supplement 3, edited M. Dayhoff, pp. 345–352. N.B.R.F., Washington (1978).
7. D. J. Lipman and W. R. Pearson. Rapid and sensi protein similarity searching. *Science* **227**, 1435–1 (1985).
8. J. M. Berg. Potential metal-binding domains in nuc acid binding proteins. *Science* **232**, 485–487 (1986).
9. J. R. Dorin, M. Novak, R. E. Hill, D. J. H. Brock, D Secher and V. van Heyningen. A clue to the basic defec cystic fibrosis from cloning the CF antigen gene. *Na* **326**, 614–617 (1987).
10. T. F. Smith, M. S. Waterman and C. Burks. The statist distribution of nucleic acid similarities. *Nucleic A Research* **13**, 645–565 (1985).
11. M. J. Garabedian, A. D. Shirras, M. Bownes and P. W sink. The nucleotide sequence of the gene coding *Drosophila melanogaster* yolk protein 3. *Gene*, in press

Vol. 149, No. 3, 1987 BIOCHEMICAL AND BIOPHYSICAL RESEARCH COMMUNICATIONS
December 31, 1987 Pages 927-929

THE GREATER STRENGTH OF ARGININE:CARBOXYLATE OVER LYSINE CARBOXYLATE ION PAIRS

IMPLICATIONS FOR THE DESIGN OF NOVEL ENZYMES AND DRUGS

Dale B. Wigley, Andrew Lyall, Keith W. Hart, and J. John Holbrook

Department of Biochemistry, University of Bristol Medical School,

Bristol, BS8 1TD, U.K.

Received October 22, 1987

The rational design of enzyme catalysts for chiral chemistry and of drugs which bind to proteins would be facilitated if rules for the recognition of one partner by the other could be formulated. This communication suggests and tests one generalization: arginine forms a tighter ion pair with a carboxylate group than does lysine and is always used for ion-pairs which are not broken during turnover in naturally-occurring enzymes.

In constructing tailor-made hydroxyacid dehydrogenases this laboratory (1) starts with a catalytic pathway carried by a large and thermostable protein framework and designs onto that a new substrate-binding site. In the preliminary design work we compared the energy of bonds between a substrate analogue (oxamate : $H_2N-CO-COO^-$) and two versions of the B. stearothermophilus lactate dehydrogenase; one with arginine and the other with lysine at the carboxylate binding-position 171. The ion pair with arginine was 20 kJ mol^{-1} stronger than the corresponding lysine-carboxylate ion pair (2). The reasons we suggested for the stronger bond with arginine (the extra hydrogen bond from the bifurcated interaction and a greater hydration potential) were quite general and not specific to the particular protein or substrate-analogue. We now examine an hypothesis extending from that general argument - that ion pairs between small ligand-carboxylates in Nature always involve arginine rather than lysine. We do not examine the corollary - that a rationally designed drug aimed at a protein-carboxylate should contain a guanidinium or similar bifurcated base.

The Brookhaven Protein Database (crystal structures) and the National Biomedical Research Foundation Database (protein sequences) were searched for

0006-291X/87 $1.50

proteins whose three dimensional structures included a small ligand with a carboxylate group. The six proteins found are shown in the Table. The bond between the protein and carboxylate is always with arginine when the protein-carboxylate bond is not transformed during the enzyme reaction. In contrast, when the ligand carboxylate interaction must be altered during the enzyme reaction, the bond is never with arginine.

The distinction can be well illustrated with citrate synthase. Citrate contains three carboxylates. Two of these remain attached to the protein during catalysis and make bonds with arginines. The third, which is derived from the thio-ester in acetyl-CoA is bound to the enzyme by a catalytic histidine.

The empirical observation can be rationalized. In a simple-binding reaction, where bond strength is presumably a selection factor, arginine-carboxylate ion pairs are evolved. Where a bond must be made and broken during the catalytic reaction, a weaker bond (never arginine) will give lower activation energies and thus higher catalytic rates. Indirect support for the generalization is the observation (Table 1) that arginines which act as binding residues are always conserved in the sequences of other members of the protein family whose three dimensional structures are unknown.

The generalization would be expected to hold well with small ligands: as the size of the ligand increases and the 20 kJ mol^{-1} advantage from one strong ion pair becomes small in comparison to the total binding energy it might break down (there are as yet no known exceptions). Both for this reason, and because there are many factors other than bond strength (regulation, folding, protein processing etc) which might determine the nature of an ion pair, we have not sought to extend this rule to protein-protein ion pairs. The choice of an arginine to recognize the second carboxylate of malate ($^{-}OOC-HCOH-CH_2-COO^{-}$), when a lactate ($^{-}OOC-HCOH-CH_3$) dehydrogenase framework was successfully reconstructed as a new and regulated malate dehydrogenase (1), was planned with a knowledge of the generalization.

TABLE 1- arginine:carboxylate ion pairs

Enzyme	Residue	Conserved Sequences	References
L-lactate dehydrogenase	Arg-171	10/10	(3)
Carboxypeptidase A	Arg-145	2/2	(4)
Carboxypeptidase B	Arg-145	2/2	(5)
Phosphoglycerate mutase	Arg-59	1/1	(6,7)
Aspartate aminotransferase (complex with 2-oxoglutarate)	Arg-292 Arg-386	9/9	(8)
Citrate synthase (complex with oxaloacetate)	Arg-401 Arg-421	3/3	(9)

Amino acid sequences were obtained from the National Biomedical Research Foundation's Protein Identification Resource, release 12, or as otherwise referenced. Crystal structures were obtained from the Brookhaven Protein Database tapes held at the Synchrotron Radiation Source, Daresbury, U.K. Sequence alignments were performed using the dynamic programming algorithm (10) as implemented in (11). Other sequence amnipulations were performed using the University of Wisconsin Genetic Computing Group package (12).

ACKNOWLEDGMENT

This work was supported by SERC through a studentship (DBW) and its Molecular Recognition Initiative.

REFERENCES

1. Clarke, A.R., Smith, C.J., Hart, K.W., Wilks, H.M., Chia, W.N., Lee, T.V., Birktoft, J.J., Banaszak, L.J., Barstow, D.A., Atkinson, T. and Holbrook, J.J. (1987) Biochem. Biophys. Research Commun. (1987) in press.
2. Hart, K.W., Clarke, A.R., Wigley, D.B., Chia, W.N., Barstow, D.A., Atkinson, T. and Holbrook, J.J. (1987) Biochem. Biophys. Research Commun. 146 346-53
3. Grau, U.M., Trommer, W.E., and Rossmann, M.G. (1981) J. Mol. Biol. 151, 289-370
4. Rees, D.C., Levis, M. and Lipscomb, W.N. (1983) J. Mol. Biol. 168 367-87.
5. Schmid, M.F. and Herriott, J.R. (1976) J. Mol. Biol. 103 175-90.
6. Campbell, J.W., Watson, H.C. and Hodgson, G.I. (1974) Nature 250 301-3.
7. Fothergill, L.A and Harkins, R.N. Proc. R. Soc. Lond. (1982) B 215, 19-44
8. Harutyunyan, E.G., Malashkevich, V.N., Tersyan, S.S., Kochkin, V.M., Torchinsky, Yu. M. and Braunstein, A.E. (1982) FEBS Lett. 138 113-6.
9. Wiegand, G., Remington, S., Diesenhofer, J. and Huber, R. J.(1984) Mol. Biol. 174 205-19.
10. Sellers, P.H. (1974) SIAM J. Appl. Math. 26, 787.
11. Lyall, A., Hill, C., Coulson, A.F. and Collins, J.F. (1986) in Parallel Computing '85, Elsevier, Holland.
12. Devereux, J., Haeberli, A. and Smithies, O. (1984) Nucleic Acids Res. 12, 387-90.

CABIOS

Vol. 4 no. 1 1988
Pages 67 – 71

The significance of protein sequence similarities

J.F.Collins*, A.F.W.Coulson and A.Lyall

Abstract

A general method of assessing the significance of scored best local alignments, particularly suited to protein sequence comparisons, is described. The method establishes the parameters describing the distribution of the best results from any search program, provided that the set is sufficiently large and the majority of the alignments arise from unrelated sequences. The expected frequency of occurrence of any score can then be calculated, together with the number of standard deviations above expectation. These provide sensible measures of significance without additional search operations. However the biological significance of any alignment or set of alignments does not solely depend on the improbability of the alignment, but on all relevant factors known to the biologist.

Introduction

The importance of detecting protein sequence similarities that convey biological insight to the molecular biologist can hardly be exaggerated. The international databases are now substantial in size. The flow of sequence information has accelerated rapidly in the last few years and will accelerate again with the advent of automatic sequencing devices. As many comparison methods report a set of best results for similarity, it becomes an important issue whether any particular alignment has significance. We describe here a simple and powerful method of analysing these results which provides quantitative values for the significance of alignments in cases where large numbers of comparisons have been reported between a query sequence and some reference set of sequences. The method can be applied whatever method is used to locate and score similar (sub)sequences. We generate the sets of best alignments for this study with an implementation of the 'Best Local Similarity' algorithm of Smith and Waterman (1981) on the I.C.L. 64 × 64 Distributed Array Processor (DAP; Flanders *et al.*, 1978), as previously described (Coulson *et al.*, 1987).

The basis of the method is a description of the form of the frequency distribution of alignment scores between 'unrelated' proteins. It is not practicable to predict this form analytically, because of the variable length of alignments and the fact that they may contain insertions or deletions. In addition, the form of the distribution may be different for each query sequence and scoring scheme (and database). The significance of an alignment is most simply expressed as the probability that a given score would be reached in the comparison of the query sequence with a collection of unrelated proteins the size of the database being searched.

We illustrate the method with examples found using proglucagon as a query sequence and a range of PAM tables (Dayhoff *et al.*, 1978). These tables are effective tools for the location of the alignments with the greatest probability of being significant. The tables can be generated for different evolutionary distances (the number of PAMs or accepted point mutations found in an alignment of 100 residues) by repeatedly applying the primitive matrix of change probabilities defined for 1 PAM, to an initial population of amino acids. While this model for evolutionary changes may not apply exactly to any real case, the range from 20–250 PAMs provides a variety of sets of useful similarity scores. Other tables, e.g. based on physical or chemical parameters, can be used, provided they are expressed suitably for the local homology algorithm with a scale that provides a zero or negative expected score for random alignments.

It must be noted, however, that the significance of alignments lies in the information they convey to the biologist and not in their improbability. Alignments conveying information about probable features related to structure or function may be important, even though their scores are not outstanding and there is therefore a need to make available quite extensive lists of alignments for perusal and detailed analysis.

Algorithm

We are essentially trying to answer the question: What is the probability that a similarity as strong as that observed in some specific case will occur between unrelated proteins? Since in any large and non-specialized database, there is a wide variety of protein types, a single family of proteins can only provide a limited number of alignments. If the number of results considerably exceeds the number of alignments possible from a set of related proteins, we can use the distribution of the bulk of the results to assess the significance of the high-scoring alignments, since the majority of the lower scoring alignments must be with proteins we would regard as unrelated. We want to predict the behaviour of the extreme high-scoring end of the

Department of Molecular Biology, University of Edinburgh, King's Buildings, Mayfield Road, Edinburgh, EH9 3JR, UK

*To whom reprints requests should be sent

distribution, using the observed upper end results, without including results which may have arisen from related proteins. For the NBRF database, no family of proteins is represented much more than 100 times, and we can assume that the lower 97% of the reported 4096 results are alignments with unrelated proteins and can be used to derive the behaviour of the remainder of the set.

We demonstrate here that the upper end of the distribution, when only unrelated sequences are involved, should decay exponentially. Suppose we have an alignment, represented by a path through the match matrix, which has reached a score S. The alignment could be improved if, from any point reached with penalties X, it is possible, by permissible moves to find a continuation which can raise the score by more than X. The greater the loss of score, X, the less likely a net improvement becomes; the probability of improving an alignment is therefore determined largely by the chance of an improvement starting relatively close to the current end. This chance does not depend on the current value of S. If the probability of improvement is independent of S then the population of alignments scoring more than S will be a constant fraction of the number scoring S. That is, we have described a process which predicts exponentially declining populations of alignments as the score increases.

This expectation has been validated experimentally; Figur 1 shows a typical example. To assess the significance of th high-scoring alignments, we fit the best straight line to the log of the numbers of observed alignments, against the score. Or ly classes containing the lower 97% of the alignments, arisin from unrelated proteins, are used. The parameters obtaine allow us to predict the expected frequency with which an specific score class should be observed. A search of the databas with a query sequence derived by randomly shuffling a rea protein sequence should generate a distribution of results whic conforms throughout to the linear model. Figure 2 shows tha this expectation is also borne out.

The expected frequency is the expectation that an alignmen with a specified score occurs by chance in comparisons wit a reference set of sequences as large as the database used, con taining only unrelated sequences. This measure of significanc is convenient and readily appreciated; it has the disadvantag that the expected frequency depends on the size of the databas used. The larger this is, the more comparisons are made wit unrelated proteins and the higher will be the scores of alignmen reached by a chance similarity. This problem could be ove come by 'normalizing' the results to a standard-sized databas (e.g. 1 000 000 residues), but a sounder alternative is to ex press the frequency of the score in terms of standard devia tions from the fitted straight line; as the fits of the experiment

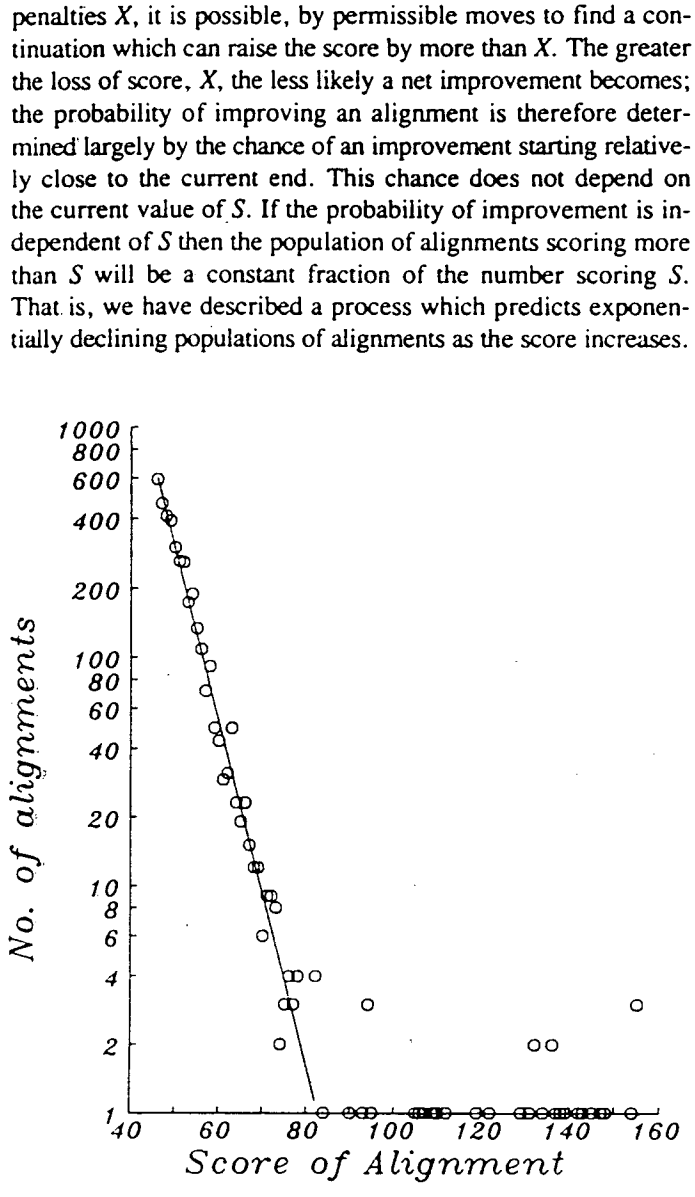


Fig. 1. Plot of the number of reported alignments achieving a given score (logarithmic scale) against the score value, for a search of the protein database with bovine proglucagon as the query sequence, using the 100 PAM similarity table. The straight line is fitted to the lowest 97% of the reported part of the distribution. The top 49 results represent unequivocally related sequences.

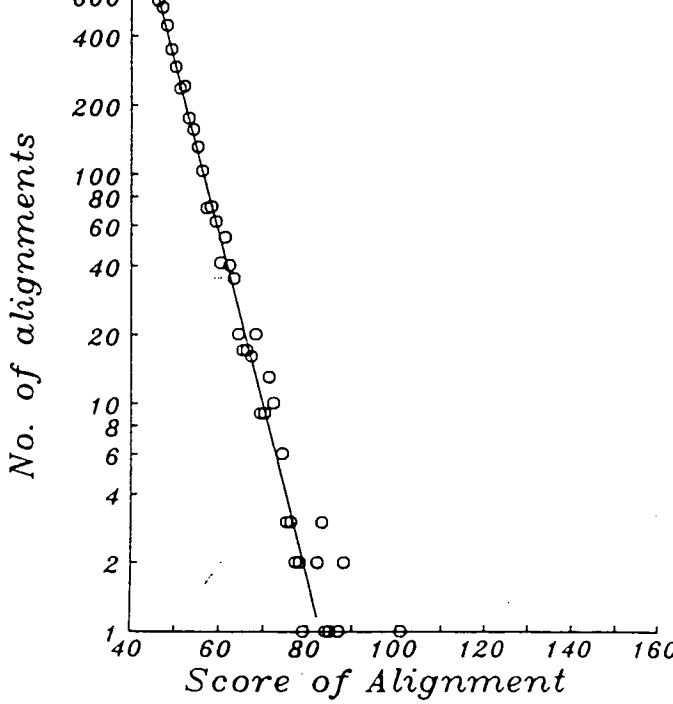


Fig. 2. As Figure 1, except that the query sequence was randomly shuffled before the search was performed. The parameters of the fitted straight line are almost identical to those in Figure 1.

points to a straight line are good, the standard deviations (s.d.) are small.

When the expected frequency falls below 1, the alignments merit attention, though it may not be possible to interpret them positively. We use a value of 0.01 to indicate alignments which need detailed examination. Many cases of significant alignments have now been demonstrated, with expectations that are orders of magnitude < 1, and up to 60 s.d.s. above expectation (e.g., Robinson *et al.*, 1987).

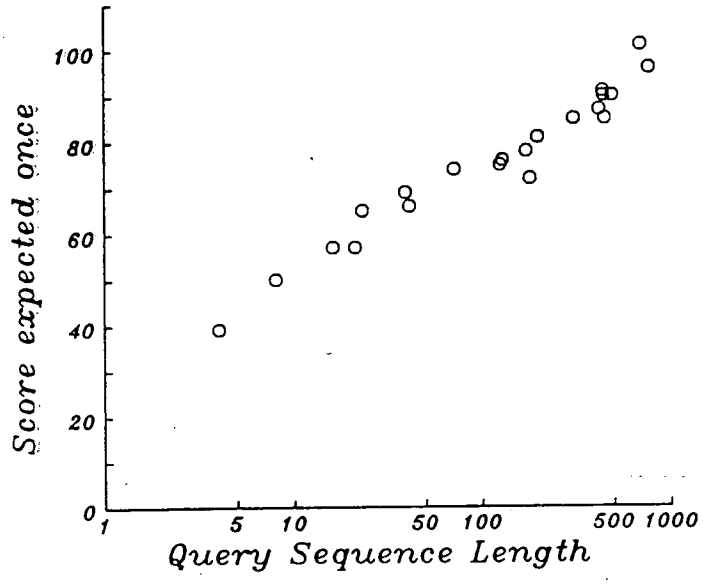


Fig. 3. Plot of the score expected once against the query sequence length (logarithmic scale) for 21 typical searches, using real proteins as query sequences, and searching the protein database with the 100 PAM similarity table.

Implementation

The alignment algorithms are particularly well-suited to parallel architecture computers (Lyall *et al.*, 1986) and details of the best local homology implementation on the I.C.L. DAP have been published (Coulson *et al.*, 1987). The program can exhaustively compare a 'query' sequence with every member of a protein sequence database and find all the locally-similar subsequence pairs, taking account of the possibility of any insertion/deletion at any position in either sequence.

The DAP returns ordered details of the alignments with the highest scores. More than one alignment can be reported from each comparison between a pair of sequences. The number of locally best alignments is huge and the top 4096 represent the tip of this much larger distribution. The results include only complete sets of results at each score and all scoring classes reported can therefore be used in the subsequent analysis.

We have made use of the similarity tables described by Dayhoff (1978), over the range 10–250 PAM's (accepted point mutations per 100 residues aligned). The indel penalty has been set to maintain the triangle inequality and to ensure that the proportion of gaps introduced into 'good' alignments is not excessive. The program requires about 1.8 s of DAP c.p.u. time per residue of the query sequence to complete a search of Release 11 (1 066 790 amino acid residues) of the NBRF database (George *et al.*, 1986).

The score of the alignment expected once, in the search with the NBRF Version 11 database and the 100 PAM similarity scoring table varies in proportion to the logarithm of the query length (Figure 3). An indication of what scores are significant could be obtained from the length alone, using this relationship.

```
20 PAM Table  Hits = 19  Mismatches = 8  Indels = 0  Expected No.= 0.565402x10^-5
          **  .****** * **.   * .*. **
      52  RHSQGTFTSDYSKYLDSRRAQDFVQWL          78
      77  RHAEGTFTSDVSSYLEGQAAKEFIAWL         103

120 PAM Table  Hits = 42  Mismatches = 15  Indels = 4  Expected No.= 0.252531x10^-6
          ...  **..****** * **... * .*. **.. . *    .  .  .*. * **.*.*.
      48  NED KRHSQGTFTSDYSKYLDSRRAQDFVQWLMNTK RNK NNIAKRHDEFER HAEGTFT        104
      72  HDEFERHAEGTFTSDVSSYLEGQAAKEFIAWLVKGRGRRDFPEEVNIVEELRRRHADGSFS       132

160 PAM Table  Hits = 58  Mismatches = 22  Indels = 4  Expected No.= 0.198303x10^-11
          ...  **..****** * **... * .*. **.. . *    .  .  .*. * **.*.*. ...  *.. *...** **.  .
      48  NED KRHSQGTFTSDYSKYLDSRRAQDFVQWLMNTK RNK NNIAKRHDEFER HAEGTFTSDVSSYLEGQAAKEFIAWLVKGR       127
      72  HDEFERHAEGTFTSDVSSYLEGQAAKEFIAWLVKGRGRRDFPEEVNIVEELRRRHADGSFSDEMNYVLDSLATRDFINWLLQTK      155

250 PAM Table  Hits = 59  Mismatches = 21  Indels = 3  Expected No.= 0.479972x10^-6
          .*  **..****** * **... *..*. **.. . *    .  .  .*. * **.*.*. ...  *.. *...** **.. .
      48  NEDKRHSQGTFTSDYSKYLDSRRAQDFVQWLMNTK RNK NNIAKRHDEFER HAEGTFTSDVSSYLEGQAAKEFIAWLVKGR       127
      73  DEFERHAEGTFTSDVSSYLEGQAAKEFIAWLVKGRGRRDFPEEVNIVEELRRRHADGSFSDEMNYVLDSLATRDFINWLLQTK      155
```

Fig. 4. Alignments and expected frequencies reported for the first internal repeat of the bovine proglucagon sequence, using similarity tables from 20–250 PAM's *indicates a matched pair of residues, . a pair with a positive score in the current table. A gap in the sequence indicates a postulated indel. Searches were also performed at 40, 60, 80, 100, 140, 180 and 200 PAM's, with similar results to those shown.

Discussion

We illustrate several features of this method with the protein proglucagon, without its signal peptide. This protein belongs to a large group of related proteins in the NBRF Protein database. The results of the DAP search show that the vast majority of the leading alignments were with proteins in the same family and that the internally repeated sequence segments gave additional alignments in the top category. The results with different PAM tables illustrate why the complexity of searching is often underestimated. The alignment between the gastrointestinal peptide from pig and the glucagon region in proglucagon is reported virtually unchanged between 20 PAMs to 250 PAMs. The expected frequency ('EF') of this alignment is least at 60 PAMs, at 2.6×10^{-6}; by 250 PAMs the expected frequency has risen to 1.3, giving no indication of the importance of the homology indicated. A similar situation arises with secretin, another active peptide. The most significant result, also at 60 PAMs, had an expected frequency of 1.1×10^{-3}. This alignment is also reported from 40–200 PAMs (at 20 PAMs it is truncated somewhat). Above 60 PAMs the expected frequency rises; by 160 PAMs it is 0.18, by 200 PAMs 1.9, and by 250 PAMs it has risen so much that it is not reported among the top 50 alignments, which include many new alignments between proglucagon and probably unrelated proteins. Significant alignments do not all behave the same way; the alignment between proglucagon and the human vasoactive intestinal peptide is most significant at 140 PAMs, with an EF of 5×10^{-8}.

Another phenomenon encountered with the Best Local Homology algorithm of Smith and Waterman is the sensitivity of paths to the scoring scheme used. As the scoring scheme is changed, alignments may lengthen or shorten, and in extreme cases may fragment into distinct parts, with lower scores and less chance of being reported. This is seen in the behaviour of the first alignment between two repeated regions in proglucagon; at low PAMs an alignment of 27 residues is found, with EF about 3×10^{-6}; by 100 PAMs an alignment of 42 residues is reported and at 160 PAMs the alignment jumps to 80 residues length, and EF 1.9×10^{-12} (Figure 4).

It is expected theoretically that the most sensitive scoring scheme in any particular case will be provided by the PAM table appropriate to the overall evolutionary distance in the alignment being sought, and these examples show this to be the case. Since the evolutionary distance is not generally known in advance, searches must be repeated with a range of PAM tables if significant similarities are not to be missed.

The ability to carry out database searches for similarities to known proteins, using the exhaustive matching algorithm for finding the best local homology, has proved a fruitful method of extending or guiding molecular biological research (Dorin *et al.*, 1987; Robinson *et al.*, 1987). We have described a method of assessment of the scores of alignments based only on the results of the initial search operation. An alternative to assess significance by reference to a further set of alignmer produced by unrelated sequences—for example, scrambled ve sions of the test sequence. As these sequences give the sar results in our analysis, they add little information to that alrea obtained.

The final test lies in the interpretation of the results by t biologist and it is clear that score alone does not determine t biological importance of an alignment. Thus, the cystic fibros antigen finds many alignments which are not individually signi cant, but which are all related to calcium-binding sites in pr teins (Dorin *et al.*, 1987). In the case of proglucagon, mar of the minor alignments confirm the presence of repeated su sequences in the whole family of glucagon precursors.

In practical terms, weak alignments may be better detecte if the query sequence is presented in a series of small segment perhaps 100 or less residues long, so reducing the backgrour of alignments arising from regions not included in the specif region under study. Extremely short sequences cannot be e: pected to produce alignments significant by their score alon other factors (e.g. the nature of the proteins in which matchir sequences are found) must be used in interpreting the outpu

We reiterate the general nature of this method; a comple database search is not required, merely the ability to accumula a large number of scores from alignments which represent th top of the distribution of such alignments. We hav demonstrated, for instance, that the method works effectivel with a much smaller search (for example, with 20 000 amin acid residues in the reference sequence set), and the parameter still fit well to the model of exponential decay. Pattern sea ches (see Coulson *et al.*, 1987) produce similar parameters a searches with the same number of identified amino acid res dues. Variable pattern spacings, with degenerate matchin possibilities or with variable spacings, produce distribution wit lower decay constants; i.e. a higher relative score must b achieved to be significant than with less flexible searches.

The parameters describing the upper portion of the distribu tion of results can also be applied to subsequent individual com parisons, since notionally each additional sequence could hav been added to the original data searched without affecting th value of the parameters derived from the initial analysis.

Acknowledgements

A.L. gratefully acknowledges the award of an SERC CASE award with I.C.L

References

Coulson,A.F.W., Collins,J.F. and Lyall,A. (1987) Protein and nucleic aci sequence database searching: a suitable case for parallel processing. *Compute J.*, 30, 420–424.

Dayhoff,M.O., Schwartz,R.M. and Orcutt,B.C. (1978) In Dayhoff,M.O. (ed.) *Atlas of Protein Sequence and Structure*. N.B.R.F., Washington, Vol. 5 supplement 3, pp. 345–353.

Dorin,J.R., Novak,M., Hill,R.E., Brock,D.J.H., Secher,D.S. and van Heyn-

ingen,V. (1987) A clue to the basic defect in cystic fibrosis from cloning the CF antigen gene. *Nature*, **326**, 614–617.
Flanders,P., Hunt,D.J., Reddaway,S. and Parkinson,D. (1978) Efficient high speed computing with the Distributed Array Processor. In Kuck,D.J., Lawrie,D.H. and Sameh,A.H. (eds), *High Speed Computer and Algorithm Organisation*. Academic Press, London, pp. 113–120.
George,D.G., Barker,W.C. and Hunt,L. (1986) The protein identification resource (PIR). *Nucleic Acids Res.*, **14**, 11–15.
Lyall,A., Hill,C., Collins,J.F. and Coulson,A.F.W. (1986) Implementation of inexact string matching algorithms on the I.C.L. DAP. In Feilmeier,M., Joubert,G. and Schendel,U. (eds), *Parallel Computing 85*. Elsevier, Amsterdam; pp. 235–240.
Robinson,A., Collins,J.F. and Donachie,W.D. (1987) Primary structure homology between prokaryotic and eukaryotic cell cycle proteins. *Nature*, **328**, 766.
Smith,T.F. and Waterman,M.S. (1981) Identification of common molecular subsequences. *J. Mol. Biol.*, **147**, 195–197.

Received on August 9, 1987; accepted on November 17, 1987

ADDENDUM

TIMING DATA FOR THE TYPE THREE ALGORITHM.

Since the completion of this project the type three dynamic programming algorithm for protein sequence comparison has been implemented on a number of different computers. The following table compares the performance of these implementations with the program written in the project.

Ref.	Machine	PMEs/s	Cost of machine $	PMEs/$/s
[1]	ICL prototype DAP	600,000	N/A	N/A
[2]	AMT production DAP	7,000,000	150,000	46
[3]	Sun 3/50	72,000	5,000	14
[4]	Sun 3/280	139,000	10,000	14
[5]	Single processor Cray XMP	1,100,000	10,000,000	0.9
[6]	32K Connection Machine CM2	25,000,000	3,000,000	8.3

[1] This project.

[2] Collins, J.F. (1988) Personal communication.

[3-6] Lander, E. (1988) 'Study of Protein Sequence Comparison Metrics on the Connection Machine CM-2' To appear in Proceedings of Supercomputing '88. Vol. 2