Hardware Accelerated Novel Protein Identification

Anish Alex[1], Jonathan Rose[1], Ruth Isserlin-Weinberger[2], and Christopher Hogue[2]

[1] Department of Electrical and Computer Engineering, University of Toronto, 10 King's College Road, Toronto, Ontario, M5S 3G4, Canada
{anish,jayar}@eecg.toronto.edu

[2] Department of Biochemistry, University of Toronto, University of Toronto, 1 King's College Circle, Toronto, Ontario, Canada M5S 1A4
{isserlin,hogue}@mshri.on.ca

Abstract. The proteins in living organisms perform almost every significant function that governs life. A protein's functionality depends upon its physical structure, which depends on its constituent sequence of amino acids as specified by its gene of origin. Advances in mass spectrometry have helped to determine unknown protein sequences but the process is very slow. We review a method of de-novo (novel) protein sequencing that requires a fast search of the genome. In this paper, we present the design of a hardware system that performs this search in a very fast, cost-effective manner. This hardware solution is more than 30 times faster than a similar search in software on a single modern PC, and up to 40 times more cost effective than a computer cluster capable of the same performance. The hardware is FPGA-based to reduce the cost and allow modification of the algorithm, both key requirements of practical protein analysis tools.

1 Introduction

Proteins and their interactions regulate the majority of processes in the human body. From mechanical support in skin and bones to enzymatic functions, the operation of the human body can be characterized as a complex set of protein interactions. Despite the efforts of scientists, many proteins and their functions have yet to be discovered. The wealth of information that lies in these unknown proteins may well be the key to uncovering the mysteries that govern life. The subject of this paper is the use of digital hardware to aid in a specific technique used to discover new proteins.

Proteins are composed of long chains of molecules known as amino acids, and the order of these amino acids is known as the sequence of a protein [2]. Protein sequencing - the process of identifying the sequence of a given protein - is a means of establishing the protein's identity, from which its functionality can be inferred. Advances in technology over the past two decades introduced the concept of protein sequencing by mass spectrometry [3]. A mass spectrometer (MS) is a device that takes a biological or chemical sample as input and measures the masses of the constituent particles of the sample. This mass information is used to identify the molecules in the sample. Protein samples to be identified are broken down into smaller subunits known as peptides and fed into an MS for identification. For novel proteins, the common approach is to identify each peptide, combine this information and thus determine the complete protein sequence. However, the sequence of novel proteins can be obtained from the information contained in genes which act to create proteins [2]. In effect, a complete genome can be interpreted as a complete protein database. For this technique to be useful in MS experiments however, very high speed searches of the genome are required, which are not feasible on general purpose processors due to the limited memory bandwidth. In such applications such as high speed searches, which involve identical repeated operations, custom hardware implementations are an ideal solution.

2 Background

A basic understanding of genetics and protein synthesis is required to comprehend the protein identification methods described in this paper. To this end, we provide a brief overview of protein synthesis and common methods of protein identification in use today.

2.1 Protein Synthesis

Within an organism, proteins are synthesized from the DNA template stored in the cell. DNA is contained in the *genes* of organisms where it is stored as a chain of nucleic acid molecules which consist of Adenine, Thymine, Cytosine and Guanine (A,T,C,G). These genes are translated into a chain of amino acids by a series of biological mechanisms. Thus if the set of all genes of the organism – its *genome* – is known, the set of all proteins it can create – its *proteome* – can be inferred. An example of protein synthesis is shown in Fig 1.

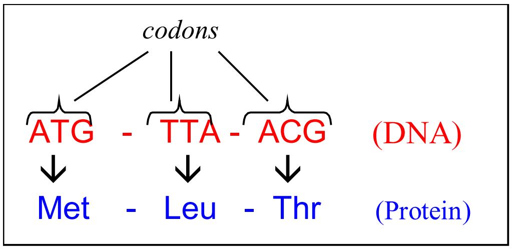

Fig. 1. Protein Synthesized from gene

Note that the DNA in the gene is grouped into sets of 3 DNA molecules, or *codons*. Each of these codons is then translated into an amino acid resulting in the protein chain. The rules for the translation from codons to amino acid are well known, and it is therefore easy to translate a codon string to its corresponding amino acids and vice versa [2]. However, it is difficult to reliably predict the starting point of a gene. The three DNA molecules in a codon, imply 3 different possibilities (known as *reading frames*). Further, every DNA strand is coupled with a complementary DNA strand that may also encode proteins resulting in a total of 6 reading frames.

2.2 Protein Identification

Prior to analysis by a mass spectrometer, proteins are digested or broken down into smaller subunits known as *peptides*. Digestion is performed by enzymes (such as trypsin) that cleave the protein at specific amino acid points. An example of the tryptic cleavage of a protein is shown in Figure 2.

MAVRAPCOKLHNWF
Original protein in sample

MAVR APCOK LHNWF
After digestion – 3 smaller tryptic peptides

Fig. 2. Trypsin Digestion of Proteins

Trypsin always cleaves the protein after the amino acids Argnine (R) and Lysine (K) with a few special exceptions. Mass spectrometric analysis of the peptides is then performed as follows:

1. **Selection:** An MS measures the masses of the tryptic peptides, and creates a *list of masses*. An operator then selects an individual peptide by mass from this list for further analysis.
2. **Identification:** The selected peptide is fragmented and analyzed by a second MS; this is followed by a complex computation that produces the sequence of the selected peptide.
3. **Repeat Cycle:** After a short delay (approx 1 sec.), another peptide is selected from the list and Step 2 is repeated. This is done for each peptide on the list.

The peptide sequences from individual peptides are grouped together and ordered to obtain the full sequence of the protein. With a few hundred peptides in a sample, a great deal of the delay in the MS process comes from having to repeat the identification process for each peptide. It is possible to use a single peptide sequence as a query to a protein database, which will return the protein that contains the query peptide as its substring. However, this technique only applies to known proteins, whose sequences exist in databases. For de-novo sequencing, i.e. the identification of novel proteins, a variant of this technique is used which relies on the core concept described in the previous section - *every protein is synthesized from the template of DNA stored in the genes*. This implies that given the knowledge of all the DNA of an organism (its *genome*) the sequence of all its proteins can be determined. In effect, the genome of an organism is translated into its *proteome* – the set of all its proteins. This proteome can then be used as a protein database, as described above to identify novel protein sequences.

3 Inferring Proteins from Genes

The concept of using genetic information to infer protein sequences is not new. Several researchers have proposed and implemented the concept in software. We employ an approach similar to that employed by Choudary et al [1]. The steps of the method are illustrated in Figure 3.
The process begins with step a) in Figure 3, which begins immediately after the second step of the MS analysis procedure described earlier. After one peptide has been identified (sequenced), it is reverse translated to the codons from which it could have originated. Thus a set of DNA strands that could have coded the peptide is now available. In step b) these DNA strands are used as queries to a genome database – in effect we are searching for the genes that may have coded the sequenced peptide. As indicated in b) it is likely that there will be multiple genes that match the set of queries. To uniquely resolve the true match, each gene must be individually considered. In step c) in Figure 3, each gene is translated to its protein equivalent as shown in Figure 1, and these translated proteins are then cleaved as shown in step d). Note that the translated protein is also cleaved at the K and R amino acids. The mass of each of the translated tryptic peptides is then calculated and compared to the list of masses produced by the first step of MS analysis. This process essentially compares the translated protein from the database against the protein actually detected by the MS. If a sufficient match is found, the protein sequence has been identified and no further work need be done. If not, the next matching gene is translated to its protein equivalent and the process is repeated.
The most obvious advantage of this approach is that the overall sequencing time will be greatly reduced. In the example in Figure 3, only a single peptide was required to obtain the full protein sequence. Note from step 3 of the MS analysis process that there is a 1 sec. (approx) delay before each peptide on the mass list is analyzed. If the algorithm described above is to be useful it must be performed within this delay. If not, an expensive downtime will be incurred as the MS instruments are stalled while the operator determines the next peptide to be analyzed. Note that several attempts at hardware sequencing have been implemented in the past [8] but we believe that this is the first published design targeting real time sequencing. Further, the work presented here is the only published design that uses the unannotated MS mass information to rank its outputs.

The key requirement of this approach is the ability to search the genome database and make comparisons to the mass list at high speeds. The Human genome contains 3.3 billion nucleic acids, and a search time of 1 second requires enormous throughput. Fortunately this kind of search is highly parallelizable in both software and hardware. Applications of this nature are good candidates for custom hardware implementation, thus the goal in this work is to design a hardware system that meets the requirements of the sequencing algorithm as described above. A number of hardware based genome search techniques have been developed over the years. Many of these, such as implementations of the Smith-Waterman algorithm are geared towards calculating the edit distance between two stringss. Other commercial devices such as DeCypher [8] are designed to perform BLAST and HMM searches which are not designed to use MS data to help identify true matches.

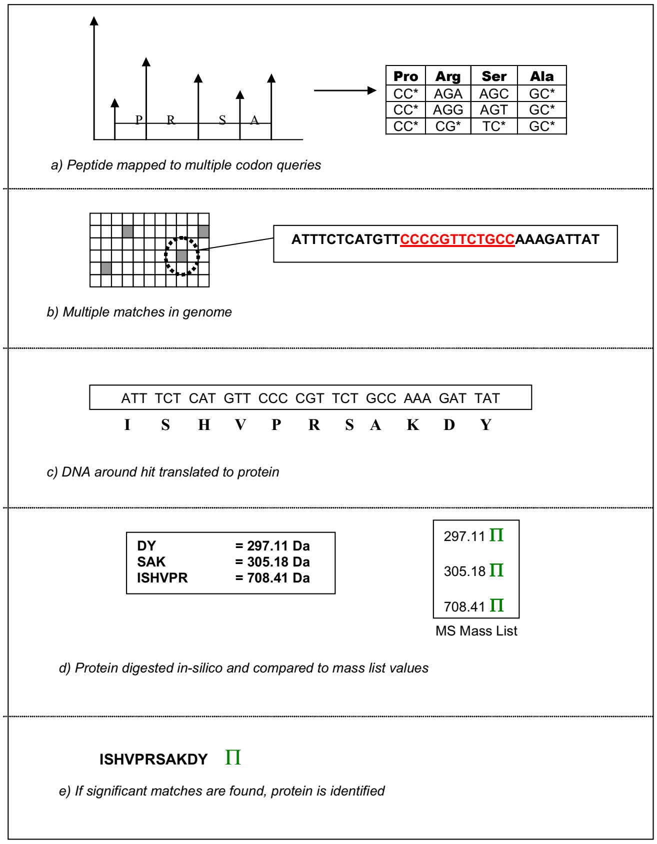
P
R
S
A
Pro Arg Ser Ala
CC* AGA AGC GC*
CC* AGG AGT GC*
CC* CG* TC* GC*
a) Peptide mapped to multiple codon queries
ATTTCTCATGTTCCCCGTTCTGCCAAAGATTAT
b) Multiple matches in genome
ATT TCT CAT GTT CCC CGT TCT GCC AAA GAT TAT
I S H V P R S A K D Y
c) DNA around hit translated to protein
DY = 297.11 Da
SAK = 305.18 Da
ISHVPR = 708.41 Da
297.11
305.18
708.41
MS Mass List
d) Protein digested in-silico and compared to mass list values
ISHVPRSAKDY
e) If significant matches are found, protein is identified

Fig. 3. Algorithm Outline

4 Design

The goal of the algorithm described in the previous section is to search through the genome in less than 1s. In addition a mass calculator is required to translate the genome to its tryptic peptide equivalents and calculate the peptide masses. Finally, a scoring system capable of comparing these calculated peptide masses to the masses measured by the MS is required.

The design takes three primary inputs, namely:

1. A peptide query from the MS, which is a string of 10 amino acids or less,
2. A genome database,
3. A list of peptide masses detected by the MS.

The design produces a set of outputs for a given peptide query:

1. A set of gene locations within the genome, which can code the input peptide query,

2. A set of scores for each potential protein gene location identified in the search. The scores rank the genes based on the likelihood that they coded the protein in the sample.

An overview of the units and their interconnection is shown in Figure 4.

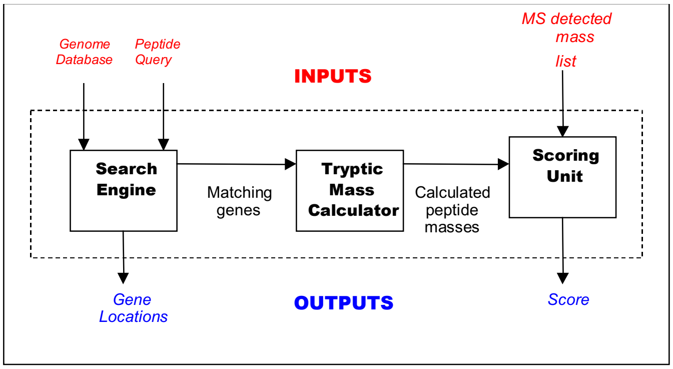

Fig. 4. Device Architecture

The search engine identifies all locations in the genome that can code the peptide query while the tryptic mass calculator translates these gene locations into their protein equivalents. The scoring unit then compares the peptides in the translated proteins to the peptides detected by the MS and provides a ranking for each gene location based on how well it matches the masses detected by the MS.

4.1 Search Engine

The first key component of our hardware is the search engine, which returns the location of every gene that can synthesize a query peptide. Many FPGA based text search techniques have been designed over the years, particularly for genome databases [8] and network security systems [6]. Most network security systems however require a new circuit to be synthesized for different sets of rules (i.e. different query strings) while genomic search engines are optimized for edit distance calculations. We require a far simpler circuit that serves merely to identify the presence of a query string in the genome.

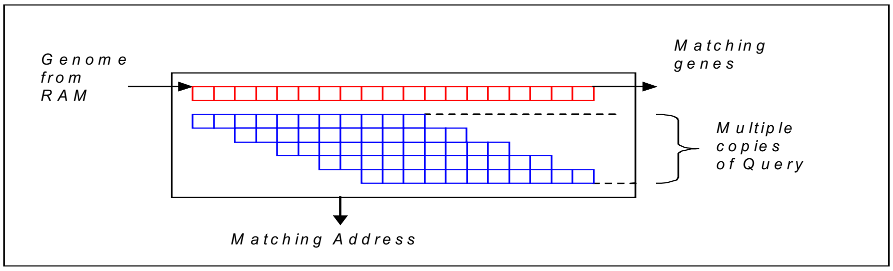

Fig. 5. Search Engine

Thus our search is simply a linear traversal through memory in which each address is read, and its contents are compared with the query. Figure 5 shows that multiple copies of the query are initialized in the hardware to simultaneously perform a parallel comparison with every position in the RAM word. If a match is detected, the corresponding memory address is sent to the user, who can then identify the coding gene at this location. Figure 5 illustrates a simplified view of the search engine, and additional hardware to implement wildcarded searches [7] has been removed from the diagram for brevity. This search looks at every nucleic acid position in the genome individually, thus no additional consideration of reading frames is required. Note further, that the matching genes are passed to the next unit, the tryptic mass calculator.

4.2 Tryptic Mass Calculator

As described, there may be several matching genes and it remains to determine which of these is the true coding gene. To this end, each gene must be translated to its protein equivalent to determine whether its constituent peptides have been detected by the mass spectrometer. To perform the translation of genes, the matching genes from the search engine are sent to the tryptic mass calculator, which interprets the DNA data from the genome as a set of codons or equivalently, a set of amino acids. It further accumulates the amino acid masses and identifies the tryptic digestion sites (the amino acids K and R) to produce a list of tryptic peptide masses. An example of the translation and calculation process is shown in Figure 6. It must be noted that the calculator interprets the gene as stored in RAM and also as the complement of the stored sequence - thus a calculator produces two reading frames of peptide masses for every gene sequence. To cover all possibilities, three calculator units are instantiated to translate all six frames of information simultaneously.

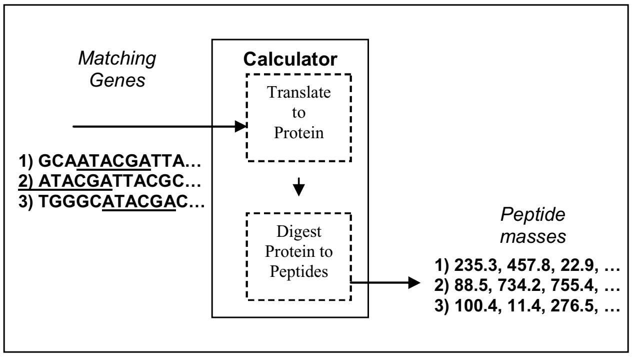

Fig. 6. Tryptic Mass Calculator

Each matching gene in Figure 6 is translated to a list of tryptic masses. It remains to identify whether these calculated peptide masses were detected by the MS. The scoring unit detailed in the following section performs this task.

4.3 Scoring Unit

Once each potential coding gene is translated to a tryptic mass list, it must be compared to the list of detected masses from the MS. The gene showing the closest match can then be identified as the true coding gene for the protein sample in the MS.
The first step of the scoring process is to store the MS mass list to chip. The values are stored using data associative techniques similar to those used in Content Addressable Memory (CAMs). A subset of the most significant bits of the mass value is used to divide the masses into specific ranges as illustrated in Figure 7. The ADDR_BITS most significant bits of the address are used as an address to store the MS measured-mass.

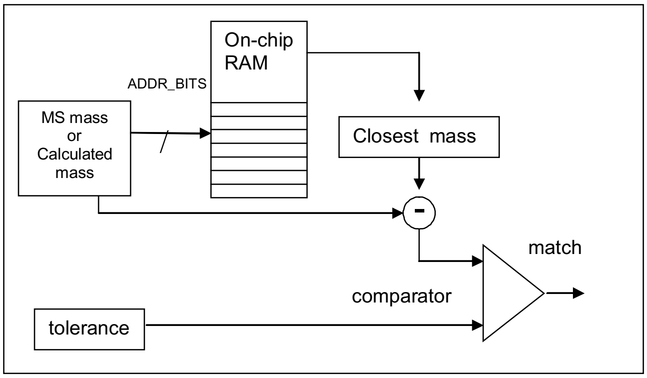

Fig. 7. Data Associative Mass Storage and Matching

Upon device initialization, each of the masses from the MS is stored in the on-chip RAM. The calculated mass is then used as an address to retrieve its closest matching mass from RAM. If the difference between these values meets a user specified tolerance, a match is signaled. It must be noted that the matches alone do not determine the final score. The final step of the scoring process once again divides masses into ranges in a manner similar to that depicted in Fig 7. In effect a histogram of masses is recorded in hardware. This histogram records the number of matches in each of the mass ranges and uses this information to calculate the final score. The score is calculated based on the MOWSE algorithm, which attempts to rank matches based on their significance. The interested reader can find the details of the MOWSE algorithm and our specific implementation in [5] and [6] respectively.

5 Implementation Performance and Costs

Variants of the design described above have been implemented in software. In this section we compare the software approach of Choudary et al [1] to our hardware. A slightly simplified version of the hardware design described has been successfully implemented and tested on the University of Toronto's Transmogrifier 3A platform [7][10].

The software scoring algorithm against which we compare our design is MASCOT [4], which is based on the MOWSE scoring algorithm. We choose the work in [1] as our software reference, as it shows the closest match to our hardware. Since the technique of real time genome scanning has generally been deemed infeasible, there have been comparatively few software implementations of search engines that use MS data for scoring. The operations in [1] were performed on a 600 MHZ Pentium III PC, resulting in search and score times of 3.5 minutes (210 s) per query. A significant portion of this time is spent searching through the genome for matches. We scale these values to current processor speeds, by assuming a linear increase in speed based on processor clock speed, which is optimistic. Based on this assumption, we state that the software can complete the task in 52.5 seconds on a 2.4 GHz processor. This scaling is unlikely, as memory bandwidth does not scale with processor speed, but this optimistic assumption presents the ideal performance of this algorithm in software. Table 1 shows what is required to achieve performance that is comparable to the hardware.

Table 1. Total Cost of Processor-based System

Number of CPUs	Scan time (s)	Cost (USD)
1	52.5	$1,962
32	1.6	$31,392
64	0.8	$62,784
512	0.1	$502,272

The original target for the hardware implementation was the Transmogrifier 3A, but as noted above, the design had to be simplified (i.e. lower bit-widths) to fit the design into the onboard FPGAs. To better reflect the capabilitets of modern FPGAs, the system was redesigned to target an Altera Stratix EP1S20 for the search engine and three Stratix EP1S40 FPGAs to implement[1] multiple parallel calculator and scoring units to maximize the processing throughput. The calculator and scoring units operate at a maximum frequency of 75 MHz, thus limiting the system to 2G char/sec. Table 2 below shows the costs of building a full system capable of performing the operations described here.

Table 2. Cost of Hardware Search and Score System

Scan Time (s)	Cost of RAM (USD)	Cost of PCB (USD)	Cost of FPGAs (USD)	Purchase Price [Full] [2] (USD)	Purchase Price [Search] (USD)
1.6	$344	$131	$6,950	$11,137	$1,530
0.8	$689	$262	$13,900	$25,426	$1,530
0.1	$5,512	$2,100	$111,200	$225,469	$12,087

[1] The units were implemented using Altera's Quartus II 3.0

[2] The Purchase Price columns include the cost of a PCB for each of the systems with an additional 50% margin.

The last two columns in Table 2 show the price of the full system (as described above) and the genome search engine as a standalone unit. We divide the systems in this manner as there are myriad applications that only require the search capabilities without the scoring described above. As a standalone search engine, the hardware is capable of out performing the software by a factor of 40 in terms of cost. With advances in mass spectrometry and the rapid progression of genetic and proteomic research, it is clear that custom hardware is a far more practical processing solution.

6 Conclusion

In this work we have studied the design of a hardware system designed to accelerate MS/MS based de-novo protein sequencing. The objective has been to study the feasibility of a custom hardware implementation of a real time protein-sequencing algorithm. The results of this work show that hardware implementations of certain key features of the sequencing system result in performance gains up to 30 times as compared to a modern processor. In addition the cost of a custom hardware solution ranges from 2 to 40 times less than that of a processor cluster capable of similar performance. With such obvious advantages, it is clear that a custom hardware implementation of this algorithm is the better choice for this protein sequencing technique.

References

[1] Choudary, Joyti S., Blackstock, Walter P., M.Creasy, D. Cottrell, John S. "*Interrogating the human genome using uninterpreted mass spectrometry data*", Proteomics, 1, pp. 651-667, 2001
[2] Lesk, Arthur M., *Introduction to Bioinformatics* . Oxford press, NY, 2002, pp.6-7
[3] McLuckey S.A. and Wells J.M., *"Mass Analysis at the Advent of the 21st Century*", Chem Rev. 101 (2), 2001, pp. 571-606
[4] MASCOT, http://www.matrixscience.com/cgi/
[5] Pappin, D.J.C., Hojrup, P. and Bleasby, A.J., "*Rapid identification of proteins by peptide mass fingerprinting*". Curr Biol, 1993, 3(6), pp 327-32
[6] Ioannis Sourdis, Dionisios N. Pnevmatikatos: Fast, Large-Scale String Match for a 10Gbps FPGA-Based Network Intrusion Detection System. FPL 2003: 880-889
[7] Alex, Anish, "*Hardware Accelerated Protein Identification*", Master's Thesis, University of Toronto, 2003, http://www.eecg.toronto.edu/~jayar/pubs/theses/ Alex/AnishAlex.pdf
[8] Decypher ®, TimeLogic Corporation, http://www.timelogic.com/products.html
[9] Lewis D., Betz V., Jefferson D., Lee A., Lane C., Leventis P., Marquardt S., McClintock C., Pedersen B., Powell G., Reddy S., Wysocki C., Cliff R., and Rose J., *"The Stratix Routing and Logic Architecture"* FPGA '03, pp. 15-20, February 2003.
[10] Transmogrifier 3A, University of Toronto, http://www.eecg.toronto.edu/~tm3