Protein alignment HW/SW optimizations

Protein Alignment HW/SW Optimizations

G. Urgese, M. Graziano, M. Vacca, M. Awais, S. Frache and M. Zamboni
Electronics and Telecommunications Department, Politecnico di Torino, Italy

***Abstract*—Biosequence alignment recently received an amazing support from both commodity and dedicated hardware platforms. The limitless requirements of this application motivate the search for improved implementations to boost processing time and capabilities. We propose an unprecedented hardware improvement to the classic Smith-Waterman (S-W) algorithm based on a twofold approach: i) an on-the-fly *gap-open/gap-extension* selection that reduces the hardware implementation complexity; ii) a *pre-selection filter* that uses reduced amino-acid alphabets to screen out not-significant sequences and to shorten the S-W iterations on huge reference databases. We demonstrated the improvements w.r.t. a classic approach both from the point of view of algorithm efficiency and of HW performance (FPGA and ASIC post-synthesis analysis).**

I. Introduction

Biologists use alignment algorithms to investigate similarities between proteins of different species, in order to find phylogenetic or functional correlations, or proteins of the same specie, for genetic mutations studies like cancers and genetic diseases [1]. Biologists have several SW tools to perform their analysis. The major drawback of these tools is the time needed to scan the entire protein databases (DBs), because CPUs are used in a serial manner. Several optimizations have been attempted exploiting GPUs and HW accelerator [9].

In this paper we describe a new approach to perform fast local alignment search of proteins in several huge DBs. The new concept enabled by our HW accelerator is to concatenate two systolic arrays. The first, called *Word Counter Reduced Alphabet Filter (WCRA_Filter)*, takes care of selecting the significant sequences coming from DBs like Swiss-Prot [5]. It is inspired to the word concept also used in the BLAST [4] first two steps, combined to the reduced amino acids (AA) alphabet concept introduced in [8]. This filter insertion is unprecedented in literature and allows a considerable reduction of calculation. The second, called *Dynamic Gap Selector S-W (DGS_S-W)*, calculates the maximum alignment score of selected sequences using an optimized version of Smith-Waterman (S-W) algorithm we conceived. The improvement is in frequency/area performance with negligible variations in terms of alignment results.

In the following, after a short background (sec. II), we explain the architectural solutions for implementing both systolic arrays on FPGA and ASIC (sec. III) and discuss our results in terms of improved functionality and performance (sec. IV).

II. Background

Biologists usually compare the studied protein Query (Qry) with others coming from DB called Subject (Sbj) in order to get information about its function, shape and evolutionary relationship. Detailed investigations are expensive [2] and the number of Sbj in DBs is amazingly wide. A drastic reduction of sequences is then operated by skimming those that present dissimilarities that are evident even with a rough analysis. Since the protein structure could be seen as a sequence of (AA), that can be viewed as letters of an alphabet, an "alignment" between Qry and Sbj is a possible way to select the similar proteins from DBs.

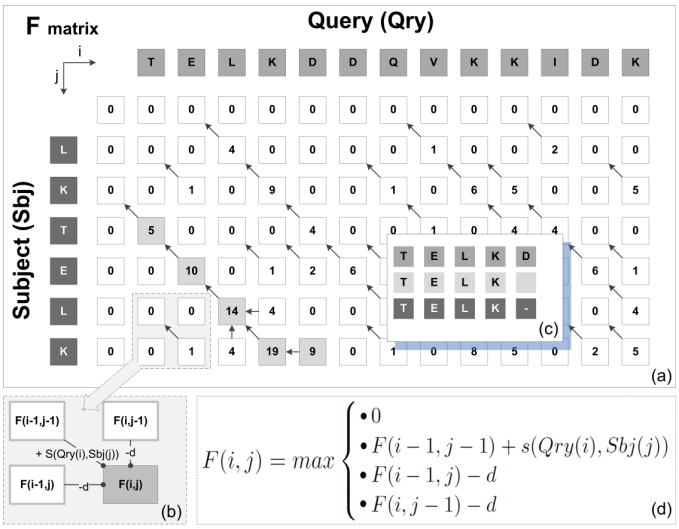

Fig. 1. (a) S-W alignment matrix: The best Local alignment is highlighted, arrows indicate cell's descendance, the number inside a cell is the alignment score. (b) Dynamic Program procedure. (c) Best local alignment. (d) S-W equation with *Linear gap* model.

The alignment of two proteins is a bioinformatics procedure in which Qry and Sbj are compared and aligned depending on how the AA that compose them are arranged. That alignment is performed AA by AA and allows to identify regions that may have identical or similar functional, structural or phylogenetic relationships. Better alignment means more similar AA order measured through a proper scoring model. This model should take into account "biological events" such as *mutations* (substitution, deletion and insertion) and *exact match*. To support the comparison *Match* or *Substitution*, *Substitutional matrices* are used, that are based on the frequency with which an AA has been replaced by another one during evolution [2].
Differently, for representing the *Insertion* and the *Deletion* of protein regions, a *gap model* is needed. For example if we align the sequences *seq1*=$\{ABCDE\}$ and *seq2*=$\{ABE\}$ we will find that a *deletion*, represented by "-", is presented $\{ABCDE \Leftrightarrow AB\text{-}\text{-}E\}$. A proper processing algorithm can give an alignment score that accounts for these "biological events".

One of the most widely used is the S-W [3], suitable to be implemented using Dynamic Programming (*Fig. 1.b*). The S-W exhaustively computes the best alignment score of Qry and Sbj subsequences (i.e best local alignment). The S-W algorithm is based on a score matrix *F*, where Qry(x) and Sbj(y) are disposed on the matrix axes (*Fig. 1.a*). First row *F(i,0)* and first column *F(0,j)* are initialized to '0'.
The S-W calculates the score of each cell recursively using the equation in *Fig. 1.d*. The best score *F(i,j)* of an alignment will be the max among four (i–iv) possible values: i) Term '0' means that it is better to end (or not start) a local alignment instead of extending one with negative score; ii) *F(i-1,j-1)+s(Qry(i),Sbj(j))* represents an alignment between Qry(i) and Sbj(j), where *s(Qry(i),Sbj(j))* is the Substitution matrix score; iii) if Qry(i) is aligned to a gap, a deletion occurs and the *F(i-1,j)-d* will be chosen; iv) if there is an insertion, i.e Sbj(j) is aligned to a gap the *F(i,j-1)-d* relation is the maximum score. The calculation of the similarity score matrix begins from the top-left cell and ends in the bottom-right. During each score calculation, the actual cell stores a pointer to the father cell ($\nwarrow \leftarrow \uparrow$ in *Fig. 1.a*). Once the matrix is filled, the Trace-back procedure starts: 1) find the maximum score cell in the matrix; 2) starting from there, in reverse mode, find the path of scores that led to this max value (following the pointers). In this way the best local alignment is found.

The algorithm described here is based on the *Linear gap* model. This model has only one parameter *d*, which is a penalty per unit length of gap. The alignment with more gaps is discouraged than the one with few gaps (the overall penalty for one large gap is the same as for many small gaps). The standard cost associated with a *Linear gap* of length *g* is given by $\{\gamma_{lin}(g) = -g * d\}$.
There exists another gap model called *Affine* based on the consideration that biological sequences are more likely to have a single large gap, rather than many small gaps [6]. It is then more likely to have one big gap of length 15, due to a single insertion or deletion event, than to have 15 small gaps of length 1. This penalty model uses a *gap open* penalty *d*, and a *gap extension* penalty *e*. The cost associated with a gap of length *g* is given by $\{\gamma_{aff}(g) = -d - (g-1) * e\}$. The *Affine gap* is a more realistic model that stresses opening of gaps instead of their lengths. To compute the alignment with the affine gap model, other two score matrices have to be calculated [2].

Since the S-W Space and Time complexity are O(MN) (where M and N are Qry and Sbj lengths) and given that protein DBs are growing exponentially, the speed with which a DB can be scanned is low. To overcome this drawback, several S-W and BLAST HW implementation on different platforms such CPU [10], FPGA [11] [12] [13] , ASIC [14] (Nano ASIC [15]) and recently on GPU too (see [9] for a comparison) have been developed, both for research and commercial purposes.

III. Proposed Optimizations

We have designed an HW accelerator to rapidly scan the entire DB with the aim of isolating the most similar sequences. The resulting small list of similar sequences are afterwards processed using the specific available analysis tools [10] by biologists. We chose to perform only the central part of the S-W alignment which is the most expensive, without taking care of Trace-back part. This choice is made because in the sequence selection phase it may be sufficient to get the maximum alignment value instead of the complete alignment.

The main improvements presented in our design with respect to previous works are: (I) The design of an optimized HW accelerator that performs S-W algorithm *DGS_S-W*. The optimization is represented by the possibility to dynamically choose, in case the *affine gap* model is used, between *gap open* (*d*) or *gap extension* (*e*) value. This, in opposition to the classical solution of saving all the gaps history in each Processing Element (PE). This optimization saves HW complexity and execution time with almost negligible costs in terms of algorithm efficiency. (II) The design of a fast pre-filter *WCRA_Filter*, based on Word concept usage [4] and Reduced AA Alphabets [8], that discards the great part of the not related sequences present into the DB. This choice has not precedents in literature and we show the evident advantages in terms of total processing time and processing efficiency.
We chose to design, both *DGS_S-W* and *WCRA_Filter*, using a systolic array architecture. This because for this kind of problems it is the more suitable architecture to be implemented on FPGA or ASIC.

I) S-W optimization *DGS_S-W*. Using a systolic array architecture for the S-W, the score matrix (*Fig. 1.a*) is filled from the top-left cell, that in our reference system is *F(1,1)*, to the bottom-right cell *F(M,N)*. The matrix is diagonally covered: in the 1st time the 1st PE calculates *F(1,1)*, in the 2nd time the 1st PE will calculate the value *F(1,2)* and the 2nd PE the value *F(2,1)*, and so on up to *F(M,N)*.
In the systolic array there will be a n^o of PE equal to the n^o of Qry AA. *Fig. 2.a* shows the architecture of the S-W systolic array. The information between the PEs (big arrows in *Fig. 2.a*) consist of: configuration signals, computed alignments scores (*F matrix* values) and AA coming from the Sbj. The AA coming from Qry and Sbj are converted in numbers, from 1 to 23 and shared on a 5-bit bus. The 0 code is used to reset the score computation registers between a Sbj and another. We have simplified the algorithm removing the two gap matrices and adding the possibility of a dynamic choice between the *gap open* or the *gap extension*. This variation means that the gap history is not saved, leading to a variation in the best alignment (that we demonstrate to be negligible). Our single PE is composed of three elements: (I) The *pe_config*, used in the configuration phase for loading the PE. (II) The *Subject_id Register* used to store the identification Sbj code. In the end of the entire database scanning we will have a list of maximum alignment score associated to the corresponds Subject_id. (III) The *pe_calc* is the PE engine used in the max score computation phase. It is used to compute the maximum alignment score and to produce the *F(i,j)* value needed by next PEs computations.
To implement the S-W using the Classic implementation of *Affine gap* model (previous works), 3 score matrices are

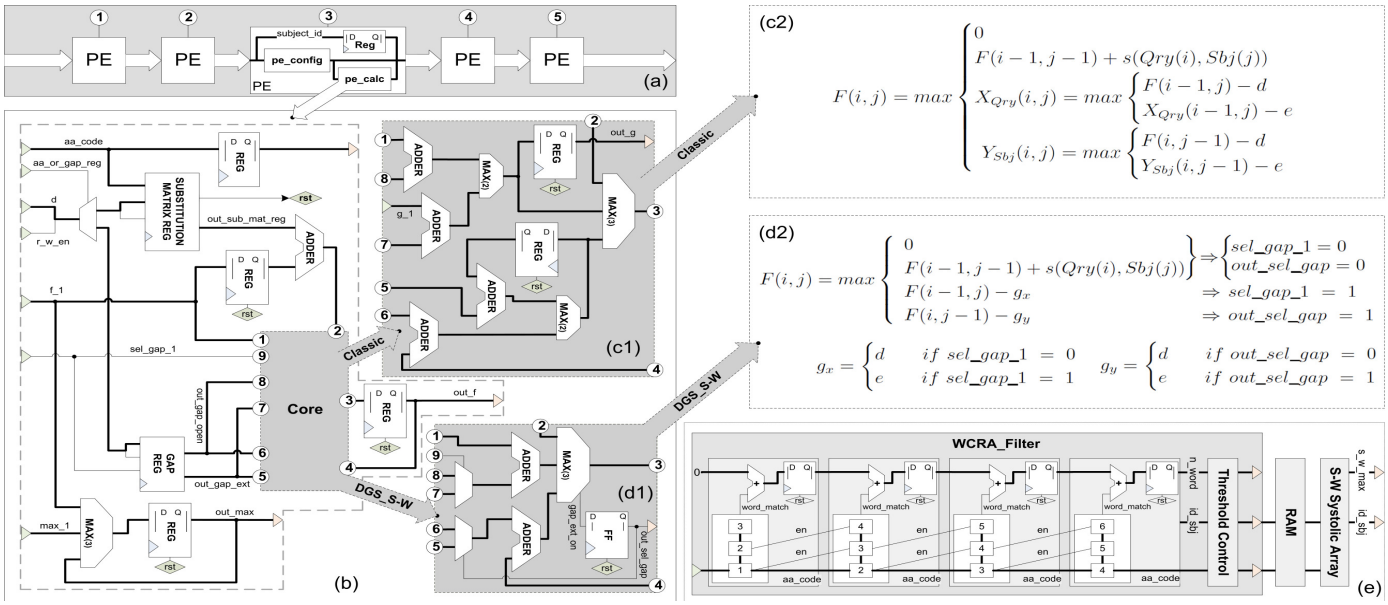

Fig. 2. (a) S-W systolic array. (b) Architecture of S-W PE engine. (c1) Classic S-W Core implementation. (d1) *DGS_S-W* Core (optimized). (c2) Classic S-W equations. (d2) *DGS_S-W* S-W equations. (e) Complete connection between *WCRA_Filter* architecture and S-W systolic array.

needed (*Fig. 2.c1*). This in architectural terms means more HW resources into the PE compared to our *DGS_S-W* (*Fig. 2.d1*). Our optimization in maths terms means a change from the eq. in *Fig. 2.c2* to the eq. in *Fig. 2.d2*. In the *pe_calc* part shared by both the Core implementations (*Fig. 2.b*), *SUBSTITUTION MATRIX REG* stores the column of substitution matrix associated to the Qry AA. The *GAP REG* saves gap values. The *MAX (3)* is used to find the maximum value of an alignment, saved and propagated through the entire array.

II) WCRA_Filter. Combining the idea of reduced AA alphabet with the word concept, we designed a filter that could be inserted before the S-W array. This is made to rapidly throw away sequences not related to the Qry using the power of parallelization. Are called "words" (*W*) small AA sequences of size *w* coming from Qry, i.e. if *seq*={ABCDE} and *w*=3 there will be *W1*={ABC} *W2*={BCD} *W3*={CDE}.
The main features of our *WCRA_Filter* must be: very fast, small area occupation and selectivity comparable to the S-W. To obtain this type of filter we include in it informations on the correlation that exists between the various AA, using reduced alphabets. The concept of reduced alphabet introduced by [7] and evolved by [8] is used in our filter in order to increase the size of the words to be selected with optimal sensitivity/selectivity trade-off. The alphabet reductions are made taking into account the Substitutional matrices or can be calculated directly from the frequency with which an AA replaces another into the sequences from the DB. *WCRA_Filter* (*Fig. 2.e*) is essentially a word counter, that is able to sense not only the perfect word match but also the similar words. This ability is given from the reduced AA alphabet usage.
The filter is designed in order to be highly parametrisable, the variable parameters are: (I) word size, (II) n^o of words, (III) Alphabet reduction Style, (IV) Alphabet reduction Size.

IV. Results

In this section we will show both functional and performance analyses, in order to validate our architectures. We design a S-W and a filter in the form of systolic array architectures using VHDL language. Many different comparative analyses have been carried out using 4 Qry of various length (30, 90, 150, 205) chosen from the "human hexokinase 1" regions. A DB of 5818 sequences (318 related and 5500 random), coming from Swiss-Prot [5], is scanned and the Classic S-W algorithm (with both gap models) is used as reference to be compared with the *DGS_S-W* and the *WCRA_Filter*.

As first test we propose the difference between the Classic S-W and *DGS_S-W* architecture using the *Affine gap* model (*gap open* = 10, *gap extension* = 1) . The usage of a *Linear gap* model gives the same identical results in the two architectures. In the *Fig. 3* it is evident that, with the *DGS_S-W* architecture, the majority of sequences get the same alignment score values. With the increasing of the Qry length the possibility to align the sequences through different ways increases. For this motivation the possibility that a longer Qry alignment differs, using the optimized architecture, is higher than the short Qry.

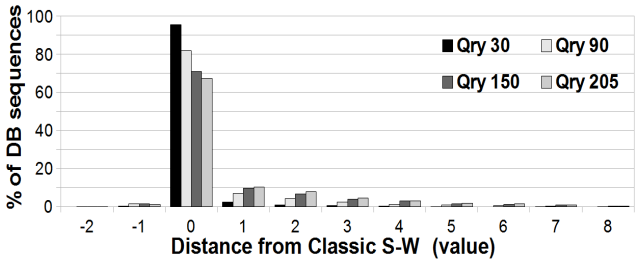

Fig. 3. Functional comparison: *DGS_S-W* and Classic S-W *Affine gap* model.

Regarding the functional *WCRA_Filter* validation, we propose a procedure that highlights the *efficiency* and the *errors*. The *efficiency* is calculated taking into account how many uncorrelated sequences are eliminated without the complete alignment. Instead, with the *errors* calculation, we evaluate how many significant sequences were lost from the filter.
We chose a threshold alignment *S-W_score* of 60 to classify a sequence as related to the Qry. This value represents an alignment of 10 AA (average BLOSUM62 matrix value of AA exact match is 6). If the threshold is too low, the possibility of getting alignments due a chance increases. For our experiment we use the two reduced AA alphabets proposed in [8] with two reduction sizes: 13 and 15.

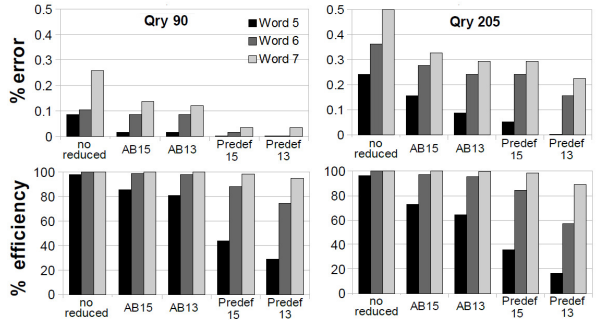

Fig. 4. *WCRA_Filter* **error** and **efficiency**. Results compared with S-W *Affine gap* model parametrised for: Qry lenght, Word size, Alphabet type [8] and Reduction dimension (no reduced, AB15, AB13, Predef15, Predef13)

From *Fig. 4* the error increases with the Qry's and word's length. Instead the efficiency of *WCRA_Filter* decreases with the Qry length. If the alphabet is reduced, the *error* and the *efficiency* are decreasing. Using different alphabets the trade-off between error and efficiency can change a lot, one of the best choice seems to be the Predefined alphabet [8] reduced from 20 to 15. An important improvement is notable from the *WCRA_Filter* usage without reduced alphabet compared with the reduced ones. This filter gives less *error* if compared to a S-W with *Linear gap* model. Moreover it is important to take into account that the *error* values showed in these graph are very low. As a final comment, it is important to underline that also BLAST [4], that is the heuristic methods widely used in biosequence analysis, produces a little error of the same order of magnitude of the one we get.

We synthesized all the architectures on a Xilinx Virtex5 XC5VLX330T-3 with the ISE 14.1 Design Suite, using FPGA design flow. Moreover we synthesize with Synopsys Design Compiler (technology used 90 nm) to get an idea of performance which could be achieved using the ASIC design flow [16],[17],[18]. *Tab. I* shows that it is more convenient, in both technologies, the *DGS_S-W* architecture in terms of frequency and area consumption. *DGS_S-W* reaches 47.7 GCUP on FPGA. On FPGA the *WCRA_Filter* area occupation is the half while the maximum frequency is almost doubled compared to standard S-W. For the ASIC target, the *WCRA_Filter* area occupation is one third, while the frequency is three times bigger. It is interesting to note that the filter can achieve very high frequency: if used before the S-W computation, it is expected to enable an important time saving.

TABLE I
PERFORMANCES OF CLASSIC S-W, *DGS_S-W* AND *WCRA_Filter* ACHIEVED BY SYNTHESIS ESTIMATION ON FPGA AND ASIC.

FPGA	**Freq (Mhz)**	**n° of Slices**	**n° of LUT/FF**	**MCUPS**	**Word Lenght**	**n° PE max**
Classic S-W	143	199	392/289	37414		261
DGS_S-W	167	182	265/268	47731		285
WCRA_Filter	281	84	108/156	173072	5	617
	280	99	123/178	146902	6	524
	281	99	136/202	147481	7	524
ASIC	**Freq (Mhz)**	**Cell area**	**Power (mW)**	**MCUPS**	**Word Lenght**	**n° PE max (@ FPGA)**
Classic S-W	375	12366	1.541	97753		261
DGS_S-W	377	10032	1.368	107547		285
WCRA_Filter	1163	4733	4.078	717442	5	617
	1163	5350	4.616	609302	6	524
	1163	5895	5.095	609302	7	524

V. CONCLUSION

Our work contributes to the HW acceleration in the biosequence analysis scenario in terms of improved processing capabilities. Two are the main key points: (I) the reduction of the S-W architecture complexity of each processing-element of S-W systolic array (*DGS_S-W*); (II) the introduction of *WCRA_Filter* that discards the majority of the uncorrelated sequences skipping any alignment computation. We demonstrated the efficiency in terms of algorithm functionality and of HW performance on both FPGA and ASIC platforms.

REFERENCES

[1] Alberts et al. "Biologia molecolare della cellula, 4 ed.", Zanichelli, 2004.
[2] Durbin et al. "Biological sequence analysis: Probabilistic models of proteins and nucleic acids", Cambridge Univ. Press, 1998.
[3] Smith and Waterman "Identification of Common Molecular Subsequences", Journal of Molecular Biology (JMB), 147, 195-197, 1981.
[4] Altschul et al. "Gapped BLAST and PSI-BLAST: a new generation of protein database search programs", Nucl Acids Res, 25, 3389-3402, 1997.
[5] Bairoch et al. "Swiss-Prot: juggling between evolution and stability", Briefings in Bioinformatics, 5, 39-55, 2004.
[6] Vingron and Waterman "Sequence alignment and penalty choice. Review of concepts, case studies and implications", J.M.B., 235(1), 1-12, 1994.
[7] Murphy et al. "Simplified amino acid alphabets for protein fold recognition and implications for folding", Protein Eng., 13(3), 149-152, 2000.
[8] Roytberg et al. "On Subset Seeds for Protein Alignment", IEEE/ACM T.C.B.B., 6(3), 483-494, 2009.
[9] Hasan L. and Al-Ars Z. "An Overview of Hardware-Based Acceleration of Biological Sequence Alignment", Computational Biology and Applied Bioinformatics, H. S. Lopes and L. M. Cruz (Ed.), 2012.
[10] "BLAST Help", NCBI, http://www.ncbi.nlm.nih.gov/books/NBK1762/.
[11] Zhang et al. "Implementation of the Smith-Waterman algorithm on a reconfigurable supercomputing platform", HPRCTA '07, 39-48, 2007.
[12] Arpith et al. "Mercury BLASTP: Accelerating Protein Sequence Alignment", ACM Trans Reconfigurable Tech. Syst., 1(9), 2009.
[13] Benkrid et al. "A Highly Parameterized and Efficient FPGA-Based Skeleton for Pairwise Biological Sequence Alignment", VLSI Systems, IEEE Transactions on, 17(4), 561-570, 2009.
[14] Halim et al. "Design and Analysis of 8-bit Smith Waterman based DNA Sequence Alignment Accelerator's Core on ASIC Design Flow", EMS '10, 126-131, 2010.
[15] Frache et al. "Nanofabric power analysis: Biosequence alignment case study", NANOARCH '11, 91-98, 2011.
[16] A. Pulimeno et al. "Udsm trends comparison: From technology roadmap to ultrasparc niagara2", *IEEE TVLSI*, 20(7),2012.
[17] S. Papaharalabos et al. "On optimal and near-optimal turbo decoding using generalized max* operator", IEEE Comm. Letters, 13(7), 2009.
[18] M. Martina et al., "Low-complexity, efficient 9/7 wavelet filters VLSI implementation", IEEE T. on Circ. and Sys. II: Exp.Br., 53(11), 2006.