2017

An Out-of-Core GPU based dimensionality reduction algorithm for Big Mass Spectrometry Data and its application in bottom-up Proteomics

Muaaz Awan
WMU, muaazgul.awan@wmich.edu

Fahad Saeed
Western Michigan University, fahadsaeed11@gmail.com

An Out-of-Core GPU based dimensionality reduction algorithm for Big Mass Spectrometry Data and its application in bottom-up Proteomics

Muaaz Gul Awan[2] and Fahad Saeed [1,2]
[1]*Department of Electrical and Computer Engineering,*
[2]*Department of Computer Science, Western Michigan University, MI, USA.*

Abstract

Modern high resolution Mass Spectrometry instruments can generate millions of spectra in a single systems biology experiment. Each spectrum consists of thousands of peaks but only a small number of peaks actively contribute to deduction of peptides. Therefore, pre-processing of MS data to detect noisy and non-useful peaks are an active area of research. Most of the sequential noise reducing algorithms are impractical to use as a pre-processing step due to high time-complexity. In this paper, we present a GPU based dimensionality-reduction algorithm, called G-MSR, for MS2 spectra. Our proposed algorithm uses novel data structures which optimize the memory and computational operations inside GPU. These novel data structures include *Binary Spectra* and *Quantized Indexed Spectra (QIS)*. The former helps in communicating essential information between CPU and GPU using minimum amount of data while latter enables us to store and process complex 3-D data structure into a 1-D array structure while maintaining the integrity of MS data. Our proposed algorithm also takes into account the limited memory of GPUs and switches between *in-core* and *out-of-core* modes based upon the size of input data. G-MSR achieves a peak speed-up of 386x over its sequential counterpart and is shown to process over a million spectra in just 32 seconds. The code for this algorithm is available as a GPL open-source at GitHub at the following link: https://github.com/pcdslab/G-MSR.

GPU; Big Data; Data Reduction; Mass Spectrometry; Proteomics; Denoising;

1 Introduction and Background

With the advent of high resolution and more sensitive mass spectrometers, MS based proteomics has become a go-to method for systems biology research. It has found its applications in detection, treatment and determination of phenotypes of cancer [1], protein sequencing and quantization [2], profiling of exosomes [3], study of toxicology [4] [5] and in evolutionary biology [6].

For the above mentioned uses of MS based proteomics, protein sequencing and quantization is the core step. This process involves breaking down a protein into peptides and separating them based on their masses followed by fragmentation and quantization in a mass spectrometer [7]. The resultant spectra contain mass-to-charge ratios of these fragments along with their corresponding intensities which are referred to as *peaks*. These MS2 spectra can be large in number where each spectrum can have up to 4000 peaks [8].

For peptide sequencing MS2 spectra can then be processed through two types of peptide sequencing algorithms: i) denovo algorithms or ii) database search algorithms. Both algorithms have to sift through large number of combinatorial possibilities to deduce a peptide for a given spectra. In our previous studies we [8] [9] have shown that 90% of the peaks in a given spectra are not helpful in peptide deduction. However, classifying peaks as useful/non-useful *before* peptide deduction is a difficult problem which has led to the development of complex pre-processing algorithms [10][8]. Pre-processing of MS data has been studied under three major categories i.e. clustering [11], noise reduction [12] and quality assessment [13]. Algorithms from all categories have a common goal of assisting in peptide deduction by improving the quality of peptide spectral matches using standard peptide deduction algorithms. Several existing algorithms [10][14][12] are able to isolate and remove about 60% to 70% of the useless peaks which results in speeding up the process of peptide deduction. However, many of these compute intensive algorithms require more time for pre-processing than actual processing of peptide deduction; defeating the purpose of their design. A detailed review of these algorithms can be found in [8].

In our previous work, we introduced MS-REDUCE, a pre-processing algorithm [8]. It is shown to bring down pre-processing time from days to minutes. To the best of our knowledge, MS-REDUCE is the fastest known sequential noise/data reduction algorithm for MS2 spectra. But even with MS-REDUCE the pre-processing time forms a significant portion of proteomics pipeline. This calls for the introduction of many-core devices such as GPUs to solve this problem. GPUs with their thousands of cores have the capability of performing thousands of calculations concurrently thus speeding up the process manifolds. However, the performance of most GPU-based algorithms is limited by the memory-related bottlenecks [15]. Typical memory bottlenecks in GPU include large transfer times[16], limited in-core memory [15] and the toll incurred by the irregular memory accesses when large data structures are used [17][18].

In this paper we present a GPU based dimensionality reduction algorithm for MS2 spectra and we call it G-MSR. We introduce two novel data structures i.e. *Binary Spectra* and *Quantized Indexed Spectra (QIS)* which solves the memory-bottleneck problems and achieves large speed up over MS-REDUCE. G-MSR is effectively an out-of-core algorithm since it is capable of processing datasets larger than the size of GPU's in-core memory. Using the novel data structures and careful parallel design we were able to achieve a peak speed-up of 386x over MS-REDUCE without loss of any accuracy.

1.1 Overview of MS-REDUCE

The algorithm first introduced in [8] operates in three steps: 1) Spectral Classification, 2) Quantization and 3) Weighted Sampling stage. Given a spectrum s and a reduction factor $0 < R \leq 1$, MS-REDUCE outputs a reduced spectrum s' such that the size of reduced spectrum is approximately $R * |s|$. The core step of MS-REDUCE is the weighted sampling step, in which a reduced spectrum s' is constructed such that the following total peak equation is satisfied.

$$\sum_{i=1}^{n} (\frac{x_i}{100} * q_i) = p^{'}$$

here x_i is the sampling weight for i-th quantum, q_i is the number of peaks in quantum i, $p^{'}$ represents total peaks in reduced spectrum and n is the number of quanta for given spectrum. Details of remaining steps can be found in [8]. Flow of MS-REDUCE algorithm can be observed in Fig. 2 A).

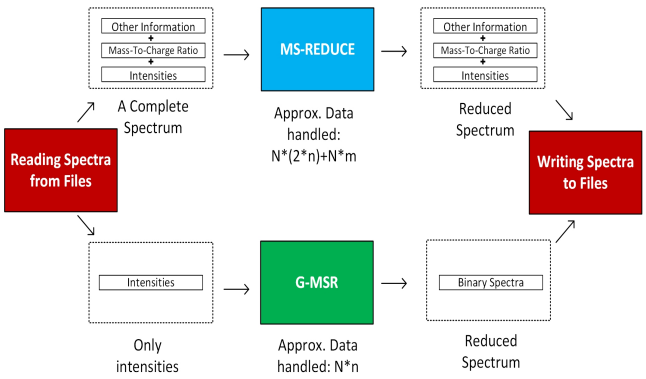

Figure 1: Here N denotes the number of spectra, n the size of largest spectrum and m the size of other information. Transferring only intensities to GPU for processing can conserve more than 50% of scarce in-core memory. G-MSR outputs the newly introduced Binary Spectra(defined in section: 2.1.2).

1.2 CUDA Programing Model

With the introduction of CUDA programing environment GPUs have become easier to program and have offered a new platform in the realm of parallel computing [17] [19]. A GPU houses several Streaming Multiprocessors (SM) with each SM containing multiple CUDA cores. In modern GPUs with compute capability 3.0 and later, each SM can have up to 192 CUDA cores. Using CUDA programing environment the intricacies of hardware architecture are hidden from a programmer; instead she can focus on parallelizing the problem. GPUs employ SIMT model which combines the usual SIMD with multiple threading. Threads in CUDA environment are organized in a two level hierarchy of grids and thread blocks. A grid consists of multiple blocks and each block contains several threads. Each thread within a block has a unique thread and a block id. Threads within a block can communicate via a shared memory and local thread synchronizations. Inter block communication is performed through global memory and block synchronizations. The global memory is 100x slower than the shared memory but the smaller size of shared memory limits its utilization to a user controlled cache [20].

2 GPU Based MS-REDUCE (G-MSR)

2.1 In-Core G-MSR

We have designed G-MSR to operate only on peak intensities. This preserves PCIe bandwidth since we only transfer intensities after rounding them off to the nearest integer. This does not have any effect on the quality as shown in our subsequent quality assessment experiments. Fig. 1 shows the difference in the amount of data handled by MS-REDUCE and G-MSR. In this paper, the word spectra or spectrum will refer to the arrays of peak-intensities.

2.1.1 Sorting and Intensity Spread Calculation

The calculation of average intensity spread (I_{avg}) requires sorting of intensities which forms a bottleneck [8]. To counter this we recently proposed a array sorting algorithm called GPU-ArraySort. It is a highly scalable algorithm for sorting large number of arrays making full use of GPU's resources [17]. We refer to the GPU-ArraySort Kernel as *Kernel-1*. Next, the intensity

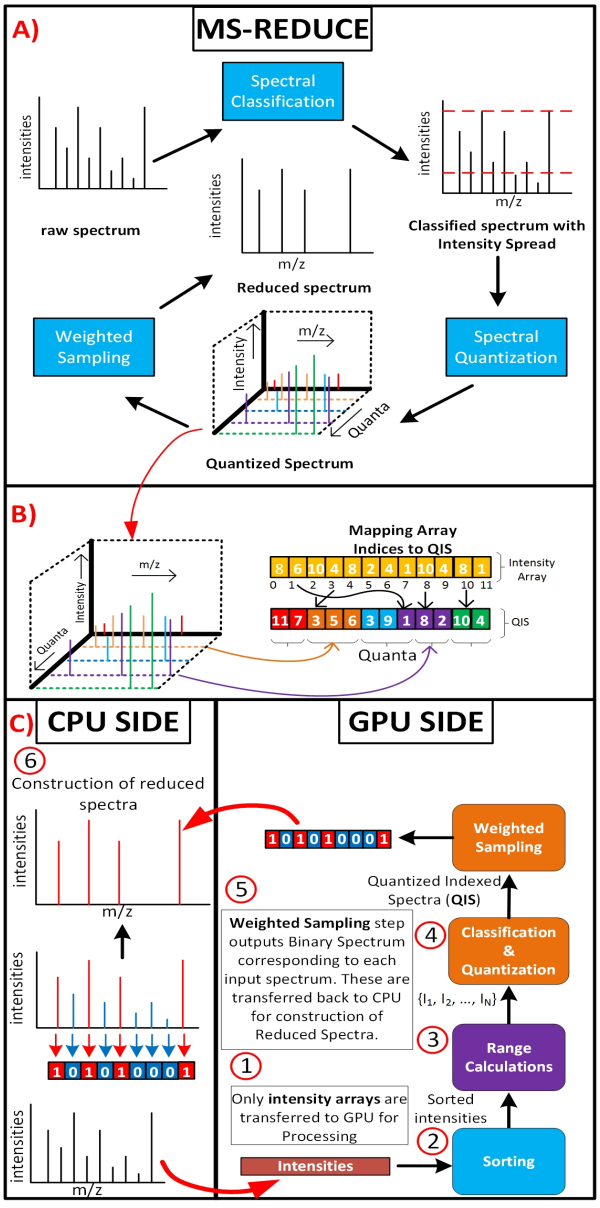

Figure 2: A) Shows the work flow of MS-REDUCE. B) Shows the construction of QIS from 3-D quantized spectrum from MS-REDUCE. C) Shows the work flow of G-MSR, blocks with same color represent processing in same kernel. A copy of actual spectra is maintained on the CPU for construction of reduced spectra.

spread kernel i.e. *Kernel-2* calculates intensity spread for each spectrum and calculates I_{avg}. Pseudo code for Kernel 2 is given in Algo. 1.

2.1.2 Spectral Classification, Quantization and Sampling

Spectral Classification, Quantization and Sampling steps are boxed together in one single kernel function; we call it *Kernel-3*. Output of this kernel is a list of Binary Spectra. Binary Spectra provides a way of communicating this information between CPU and GPU necessary for constructing reduced spectrum from Binary Spectrum, as shown in Fig. 2.

Definition 1 *Given a spectrum* $s_i = \{p_1, p_2, p_3, \ldots, p_n\}$ *a Binary Spectrum* B_i *for the corresponding reduced spectrum* s'_i *is defined as,* $B_i = \{e_j = 1 | p_j \in s'_i\} \cup \{e_j = 0 | p_j \notin s'_i\}$.

As shown in Fig. 2 A), quantization step yields a complex 3-D data structure. Maintaining and accessing 3-D data structures inside GPU's memory can be very inefficient. To tackle this problem we introduce a novel data structure, Quantized Indexed Spectrum (QIS) to represent a quantized spectrum in a simple 1-D array. QIS helps avoiding irregular memory accesses and uses 50% less data, resulting in efficient, in-core memory usage. Fig. 2 B) shows the construction of QIS from intensity array. QIS can be formally defined as:

Definition 2 *Given a spectrum* s_i *of size* n*, if after quantization,* s_i *has* m *quanta, then its QIS is given by* $Q_i = \{l_1, l_2, l_3, \ldots, l_n\}$*. Here* l_x *represents a peak index. Starting and ending offsets of quanta are stored in a separate array* $p = \{st_1, e_1, st_2, e_2, \ldots, st_m, e_m\}$ *where* st_y *is the offset where quantum* y *starts and* e_y *is where quantaum* y *ends.*

The *Sampling* step involves determining the sampling weights for satisfying the peak equation. Peak-indices in QIS are sampled across all quanta based on these sampling weights. From the sampled-indices a Binary Spectrum is constructed using the array $O = \{0_1, 0_2, 0_3, \ldots\ 0_k\}$ of size same as s_i. And placing 1s at each index sampled before. Fig. 2 B) shows the process of constructing reduced spectra from Binary Spectra on CPU side.

2.2 Out-of-Core G-MSR

The out-of-core algorithm estimates the amount of memory required to process a given dataset. Depending on the total in-core memory of given GPU it then divides the dataset into parts and processes them in passes. However, because of data dependencies processing has to be performed in phases.

2.2.1 Phase 1

After moving the first chunk of spectra in the GPU memory, Kernel-1 is launched followed by Kernel-2 to calculate I_{avg}. The array of I_i for each chunk is stored on CPU temporarily. Then the process is repeated for remaining chunks, till I_i for all spectra and their sum is available on CPU. I_{avg} value for complete dataset is calculated on CPU and copied back.

2.2.2 Phase 2

In phase 2, each chunk of spectra along with its corresponding I_i array is copied to GPU and Kernel-3 is launched. The process repeats till all the chunks have been processed. Fig. 3 shows the flow of out-of-core execution.

Algorithm 1 Per thread Pseudo code for Kernel 2

Require: An array A of intensities and array S containing spectrum sizes. Subscript i wherever used, represents unique spectrum
Ensure: Array B containing binary spectrum of A
  for each thread i **do**
    $R_i \leftarrow$ SUM(Max10Peaks) - SUM(Min10Peaks)
    $R_{avg} \leftarrow$ Average(R)
  end for

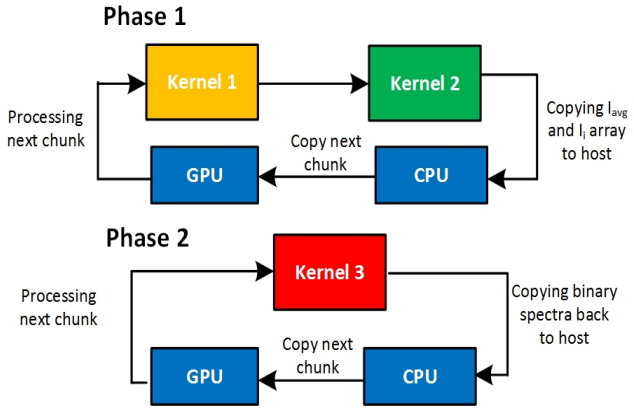

Figure 3: The figure showing the flow for out-of-core processing, phase 1 and phase 2 have been shown separately.

Algorithm 2 Per thread Pseudo code for Kernel 3

Require: An array A of intensities, Average Intensity Spread R and array of Intensity Spread R
Ensure: Array B containing binary spectrum of A
  for each thread i **do**
    $C_i \leftarrow$ getClass(A_i, R_i)
    $Q_i \leftarrow$ getTotalQuanta(C_i)
    for each $a_j \in A_i$ **do**
      $a_j \leftarrow$ getQuantum(a_j)
    end for
    $q_j \in qS \leftarrow$ quantaSizes(A_i)
    $B \leftarrow 0$
    for each $q_j \in qS$ **do**
      $q_j \leftarrow$ getSampleRate(q_j)
      B[getSample(q_j)] $\leftarrow 1$
    end for
  end for

3 Time Complexity Analysis

GPU-ArraySort uses insertion sort for sorting small sized bins. If p CUDA threads are used for sorting a spectrum of size n, its time complexity would be $O(\frac{n^2}{p})$, we can extend this for N spectra and B CUDA blocks to get $O(\frac{N*n^2}{B*p})$ for the sorting step. Intensity Spread calculation and Classification steps are constant time processes and take $O(N)$ for N spectra, while using a total of B threads across all CUDA blocks, both of these steps can be performed in $O(\frac{N}{B})$. The Quantization step takes $O(n)$ time for a spectrum of size n. For N spectra this would take $O(N * n)$, and when using B threads across all blocks it becomes $O(\frac{N*n}{B})$. The Random Sampling step has an execution time of $O(s * n)$ where s is the rate of sampling and its value varies from 0 to 1. While using a total of B threads across all blocks and for N spectra it becomes $O(\frac{s*N}{B})$. Combining all of the above time complexities and replacing $l = p * (2 + n + n * s)$, we get:

$$O(\frac{N * (n^2 + l)}{B * p}) \tag{1}$$

4 Performance Evaluation

We evaluated the performance of G-MSR from two different aspects. First was a time based analysis to determine the amount of speed-up obtained over the serial version. Second was a quality assessment experiment to see how the number of high quality peptide spectral matches (PSMs) varied when treating the spectra with G-MSR. We also evaluate the performance of G-MSR while using it as a tool for reductive proteomics for high resolution instruments.

4.1 Data Generation

For all the experiments, we made use of the thirteen datasets we used before in [8] and [11]. Naming conventions for all thirteen data sets are same as in [8].

4.2 Experiment Setup

For all the experiments we made use of a Linux server running Ubuntu Operating System, version 14.01. The server houses two Intel Xeon E5-2620 Processors, clocked at 2.40 GHz with a total RAM of 48 GBs. The system has an NVIDIA Tesla K-40c GPU with a total of 2880 CUDA Cores and 12 GBs of RAM. CUDA version 7.5 and GCC version 4.8.4 were used for compilation.

4.3 Scalability and Time Analysis

For this experiment, we used the appended UPS2 dataset which had over a million spectra. Timing experiments were performed with increasing datasizes to cover the out-of-core cases. Results for scalability studies can be observed in Fig. 5. In Fig. 5, when N is small and is almost equal to B we get huge speed ups. But as the number of spectra increase and the number of concurrent threads reach their limit we observe a decrease in speed up in accordance with Eq. 1. Also term l in Eq. 1 increases when reduction factor is increased. This explains higher speed ups for lower reduction factors.

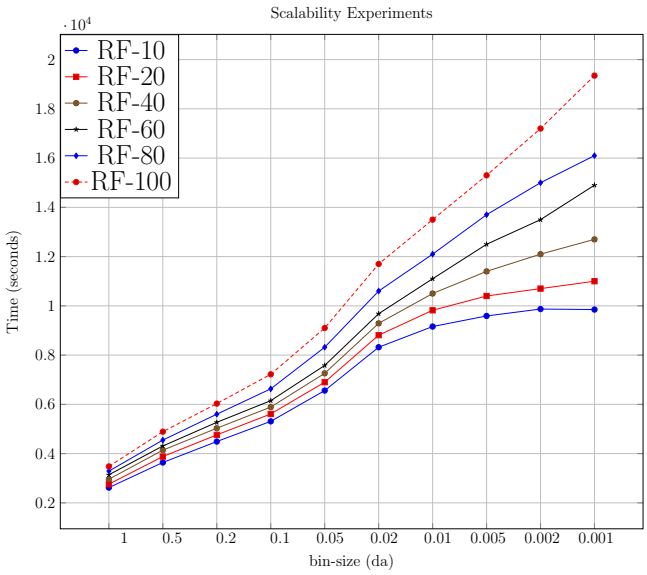

Figure 4: Timing plots of peptide deduction process using Tide with hiXcorr algorithm. Here RF is the reduction factor. An increasing RF makes the process more scalable.

4.4 Quality Assessment

We used the same method of quality assessment as discussed in [8]. For our experiments we set the FDR value of interest to 5% i.e. any PSM having FDR value below 5% is an acceptable match, we call them effective matches. Fig. 6 shows percentage of effective matches with varying reduction factors for both algorithms. G-MSR and MS-REDUCE gave almost same percentages of effective matches.

4.5 Reductive Proteomics for high resolution instruments

In high resolution mass spectrometery data, number of bins created can be quite large which can lead to large processing times for all sorts of peptide deduction algorithms [21].

Pre-processing of spectra with G-MSR will reduce the size of spectrum and hence the processing time for peptide deduction. We performed peptide deduction for UPS2 dataset after preprocessing it with G-MSR at different reduction factors. For this experiment Tide [22] integrated with hiXcorr [21] was used for peptide deduction. Fig. 4 shows that with smaller reduction factors the performance of peptide deduction algorithm becomes more scalable even with increasing resolution.

5 Discussion and Conclusion

Modern day multi- and many-core devices are changing the way scientists tackle computationally complex problems. Theoretically intractable problems with complex approximate solutions are now being solved in acceptable times using modern high performance computing devices. In the field of bioinformatics datasets may range from a few gigabytes to several terabytes. These large datasets not only provide computational bottlenecks but an added problem of data management inside the limited in-core memories of these devices. In this paper we presented a GPU based data reduction algorithm for MS2 spectra called G-MSR. We introduced two novel

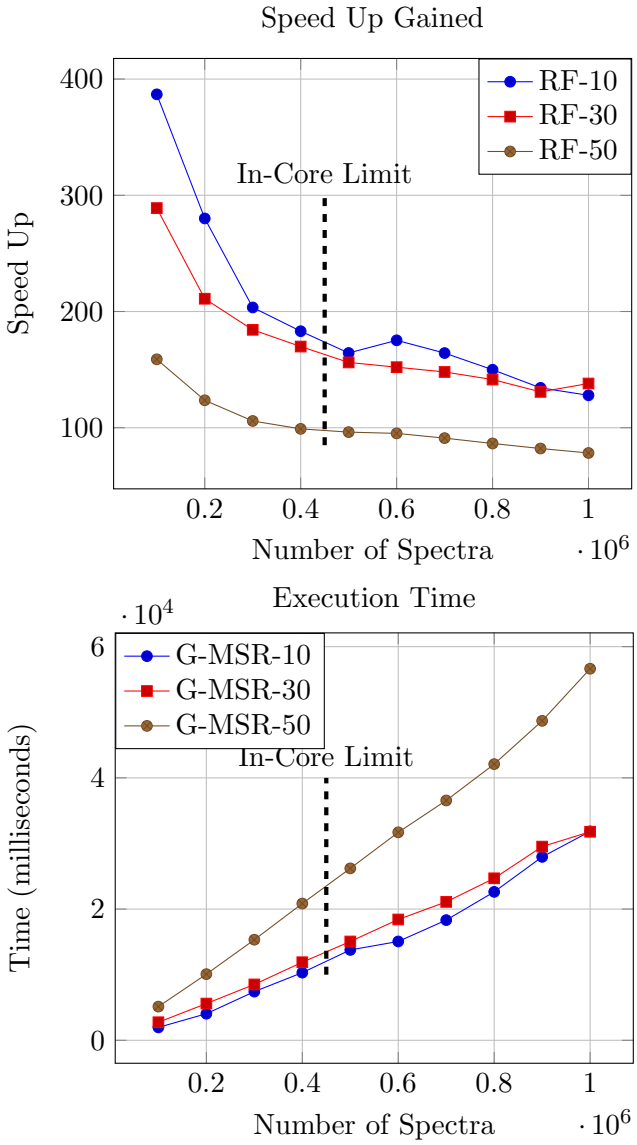

Figure 5: Top figure shows the speed up achieved by G-MSR over MS-REDUCE and the bottom figure shows the execution times for G-MSR operating at reduction factors of 10, 30 and 50. The vertical line represents the point where in-core memory is filled.

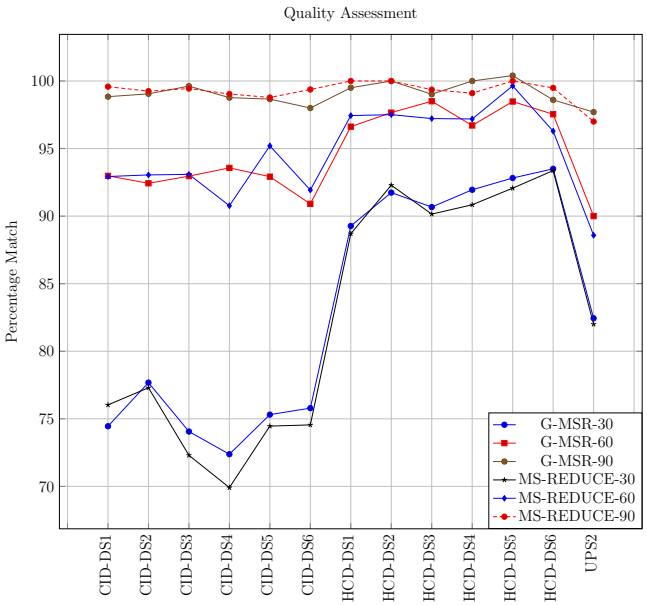

Figure 6: Quality assessment plots of MS-REDUCE and G-MSR. Numerical values in legend are the reduction factors. X-axis is the data set labels and Y-axis is the PSM percentage with FDR greater than 5%.

data structures called Quantized Indexed Spectrum (QIS) and Binary Spectrum which helped minimize the use of GPU memory and making memory accesses more localized and thus enabled the processing of large data sets in a single pass with maximum efficiency. Binary spectra allowed us to process spectra by just copying the required information on to the GPU while storing the actual spectra on CPU Memory.

Our proposed strategy is capable of processing datasets which exceed the memory limits of a GPU using an out-of-core technique. Our experiments show a peak speed-up of $386x$ as compared to the serial version of the algorithm. And finally we introduce G-MSR to be used as a pre-processing step in reductive proteomics for high resolution instruments for high throughput processing.

6 Acknowledgements

This research was supported by the National Institute Of General Medical Sciences (NIGMS) of the National Institutes of Health (NIH) under Award Number R15GM120820. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health. This work was supported in part by National Science Foundation grants NSF CRII CCF-1464268 and NSF CAREER ACI-1651724. We would also like to acknowledge the donation of a K-40c Tesla GPU from NVIDIA which was used for all GPU based experiments performed in this paper.

References

[1] H. M. Verheul, "Mass spectrometry-based proteomics: from cancer biology to protein biomarkers, drug targets, and clinical applications." American Society of Clinical Oncol-

ogy, 2014.

[2] J. K. Eng, A. L. McCormack, and J. R. Yates, "An approach to correlate tandem mass spectral data of peptides with amino acid sequences in a protein database," *Journal of the American Society for Mass Spectrometry*, vol. 5, no. 11, 1994.

[3] T. Pisitkun, R.-F. Shen, and M. Knepper, "Identification and proteomic profiling of exosomes in human urine," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 101, no. 36, 2004.

[4] L. K, "Toxicological screening and quantitation using liquid chromatography/time-of-flight mass spectrometry," *Journal of Forensic Science and Criminology*, vol. 1, no. 1, 2013.

[5] L. A. Leuthold, J.-F. Mandscheff, M. Fathi, C. Giroud, M. Augsburger, E. Varesio, and G. Hopfgartner, "Desorption electrospray ionization mass spectrometry: direct toxicological screening and analysis of illicit ecstasy tablets," *Rapid Communications in Mass Spectrometry*, vol. 20, no. 2, pp. 103–110, 2006.

[6] B. Zhao, T. Pisitkun, J. D. Hoffert, M. A. Knepper, and F. Saeed, "Cphos: A program to calculate and visualize evolutionarily conserved functional phosphorylation sites," *Proteomics*, vol. 12, no. 22, pp. 3299–3303, 2012.

[7] R. Aebersold and M. Mann, "Mass spectrometry-based proteomics," *Nature*, vol. 422, no. 6928, pp. 198–207, 2003.

[8] M. G. Awan and F. Saeed, "MS-REDUCE: An ultrafast technique for reduction of big mass spectrometry data for high-throughput processing," *Bioinformatics*, p. btw023, 2016.

[9] ——, "On the sampling of big mass spectrometry data," in *The International Society for Computers and Their Applications (ISCA)*, 2015.

[10] N. Mujezinovic, G. Schneider, M. Wildpaner, K. Mechtler, and F. Eisenhaber, "Reducing the haystack to find the needle: improved protein identification after fast elimination of non-interpretable peptide ms/ms spectra and noise reduction," *BMC Genomics*, vol. 11, 2010.

[11] F. Saeed, J. D. Hoffert, and M. A. Knepper, "Cams-rs: Clustering algorithm for large-scale mass spectrometry data using restricted search space and intelligent random sampling," *IEEE/ACM transactions on computational biology and bioinformatics*, vol. 11, no. 1, pp. 128–141, 2013.

[12] J. Ding, J. Shi, G. G. Poirier, and F. X. Wu, "A novel approach to denoising ion trap tandem mass spectra," *Proteome Science*, vol. 7, no. 9, March 2009.

[13] M. Bern, D. Goldberg, W. H. McDonald, and J. R. Y. IIII, "Automatic quality assessment of peptide tandem mass spectra," *Bioinformatics*, vol. 20, 2004.

[14] N. Mujezinovic, G. Raidl, J. R. A. Hutchins, J.-M. Peters, K. Mechtler, and F. Eisenhaber, "Cleaning of raw peptide ms/ms spectra: Improved protein identification following deconvolution of multiply charged peaks, isotope clusters, and removal of background noise," *Proteome Science*, vol. 6, 2006.

[15] S. Chao, J. R. Green, and J. C. Smith, "Evaluation of a gpgpu-based de novo peptide sequencing algorithm," *Journal of Medical and Biological Engineering*, vol. 34, no. 5, 2014,.

[16] Y. Fujii, T. Azumi, N. Nishio, S. Kato, and M. Edahiro, "Data transfer matters for gpu computing," in *Parallel and Distributed Systems (ICPADS), 2013 International Conference on*. IEEE, 2013, pp. 275–282.

[17] M. G. Awan and F. Saeed, "Gpu-arraysort: A parallel, in-place algorithm for sorting large number of arrays," in *Parallel Processing Workshops (ICPPW), 2016 45th International Conference on Parallel Processing*. IEEE, 2016, pp. 78–87.

[18] D. Cederman and P. Tsigas, "Gpu-quicksort: A practical quicksort algorithm for graphics processors," *Journal of Experimental Algorithmics (JEA)*, vol. 14, p. 4, 2009.

[19] J. Sander and E. Kandrot, *CUDA by Example*, 2010.

[20] Nvidia. (2016) CUDA Toolkit Documentation v7.5. [Online]. Available: http://docs.nvidia.com/cuda/index.html#axzz42Wi4k0Qc

[21] H. Kim, H. Jo, H. Park, and E. Paek, "Hixcorr: a portable high-speed xcorr engine for high-resolution tandem mass spectrometry," *Bioinformatics*, p. btv490, 2015.

[22] B. J. Diament and W. S. Noble, "Faster sequest searching for peptide identification from tandem mass spectra," *Journal of Proteome Research*, vol. 10, no. 9, pp. 3871–3879, July 2011.