Distributed Detection of Cancer Cells in Cellular Spike Streams

Abdul Hafeez†, M. Mustafa Rafique‡, Ali R. Butt†

†*Virginia Tech, Blacksburg, Virginia*
‡*IBM Research, Dublin, Ireland*
hafeez@vt.edu; mustafa.rafique@ie.ibm.com; butta@vt.edu

***Abstract*—Detection and identification of important biological targets such as, DNA, proteins, and diseased human cells is crucial towards early disease diagnosis and prognosis. The key to differentiate healthy cells from the diseased cells is the biophysical properties that differ significantly. Micro and nanosystems, such as solid-state micropores and nanopores, can measure and translate these properties of human cells and DNA into electrical spikes to decode useful biological insights. Nonetheless, such approaches result in large data streams that are often plagued with inherit noise and baseline wanders. Moreover, the extant detection approaches are tedious, time-consuming, and error-prone, and there is no error-resilient software that can analyze large datasets instantly. The ability to effectively process and detect biological targets in larger datasets lies in the automated and accelerated data processing strategies using state-of-the-art distributed computing systems. To this end, we propose a distributed detection framework, which collects the raw data stream on a server node that then splits/distributes the data into segments across the worker nodes. Each node reduces noise in the assigned data segment using moving-average filtering, and detects the electric spikes by comparing them against a statistical threshold (based on the mean and standard deviation of the data), in a Single Program Multiple Data (SPMD) style. Our proposed framework enables the detection of cancer cells with an accuracy of 63% in a mixture of Cancer cells, Red Blood Cells (RBCs), and White Blood Cells (WBCs), and achieves a maximum speedup of $6X$ over a single-node machine by processing 10 gigabytes of raw data using an 8-node cluster in less than a minute.**

***Keywords*-Distributed computing; automated cancer cell detection; solid-state micropores; accelerated-diagnosis;**

I. Introduction

Diseases such as cancer can be mitigated, if detected and treated at an early stage. Micro and nanoscale devices such as micropores and nanopores, enable the translocation of biological targets, such as, DNA, proteins, and human cells at its finer granularity. These devices are tiny orifices in silicon-based membranes and the output is a current signal, measured in nanoamperes. Solid-state micropore is capable of electrically measuring the biophysical properties of human cells, when a blood sample is passed through it [1]. The passage of cells via such pores results in an interesting pattern (pulse) in the baseline current, which can be measured at a very high rate, e.g., $500,000$ samples per second [2]. The *pulse* is essentially a sequence of temporal data samples that abruptly falls below and then reverts back to a normal baseline with an acceptable predefined time interval, i.e., pulse width. The pulse features such as, *width* and *amplitude* corresponds to the translocation behavior and the extent to which the pore is blocked, under a constant potential. These features are crucial in discriminating the diseased cells from healthy cells such as, identifying cancer cells in a mixture of cells [3].

The task of detecting interesting patterns is critical in many biomedical applications including ECG and MRI [4], [5], [6] in order to find useful insights towards disease diagnosis. With the advent of novel biological applications of micro and nanoscale devices [7], [8], [9], [10], we are able to detect the biological targets, such as DNA and human cells, at finer granularity. Unfortunately, the detection of biological targets with these devices have many challenges. The devices produce large amounts of raw data and the task of pattern detection is coupled with the enormity of the datasets [2], [11]. For example, the data collected from the translocation of a typical biopsy sample through a pore is about 10 GB. The state-of-the-art detection and analysis is based on the visual inspection, which is tedious, error-prone, and even an expert has to spend innumerable hours to analyze a blood sample, which is in the magnitude of gigabytes. In addition, the available softwares, such as, pClamp can only analyze a subset of the total acquired data. Furthermore, due to the highly dynamic nature of data produced by the bio-nano sensors, the useful patterns are very sparse in the data and are orders of magnitude smaller than the total acquired data. In a clinical setup, the raw data collected from many patients' biopsy samples, using multiple solid-state micropores quickly becomes too large to be handled by a single workstation. These challenges motivate the design of an automated and distributed detection technique, which can effectively acquire and process the raw data (collected from many micropores) for detecting and identifying useful patterns.

Recent trends show the efficacy of using distributed computing in genomics [12], [13], [14], [15], proteomics [16], [17], and other large-scale applications such as physics [18] and astronomy [19]. In this paper, we design and develop a novel distributed technique that distributes/splits the data

across multiple nodes and enables individual nodes to acquire and process its raw data segments to detect and identify useful pulses.

To this end, our proposed framework performs the following steps:

- Splits the collected raw data into a number of segments, one for each participating node in the system.
- For improved performance, each raw data segment is acquired by individual nodes using double buffers.
- Each node converts the raw data into integers for efficient processing, i.e., to avoid round-off errors and also reduce the total size of the acquired data.
- The integer data is then smoothed using moving-average filtering to reduce the noise in the data.
- The useful patterns in the de-noised data are detected using a threshold that is based on the mean and standard-deviation of the data.

Evaluation of our system using the datasets of real cancer cells show that our technique can detect pulses with 63% accuracy, and process 10 gigabytes of raw data on an 8-node cluster in less than a minute, a task that would rather take several hours when using the extant manual process. Our framework produces output in the form of scatter plots, which can be further used by physicians/scientists to infer useful information for disease diagnosis and useful decision-making.

The rest of the paper is organized as follows. In Section II, we discuss the design and algorithm of our distributed framework. Section III presents our experimental setup and results with rigorous performance evaluation and comparison between the distributed system and single node implementation with an increasing size of input raw data and different size of double buffers. Section IV captures the related work. Finally, we discuss success stories and limitations in Section V and conclude in Section VI.

II. Distributed Detection Framework

We present the design and implementation of our framework for splitting and distributing the data stored on an NFS shared storage across multiple nodes. Each node then processes the raw data segment assigned to it in a SPMD (Single Program Multiple Data) style using the following software modules: pre-processor, smoother, detector, and post-processor. The high-level work flow of our distributed framework is shown in Figure 1. The pre-processor formats its segment of raw data and reads the data efficiently using double buffers. The smoother reduces the noise and eliminates baseline shifts. The detector detects pulses in the data based on a threshold computed from the statistics of the smoothed data. Finally, the consolidated results from all the nodes are merged and delivered for further analysis.

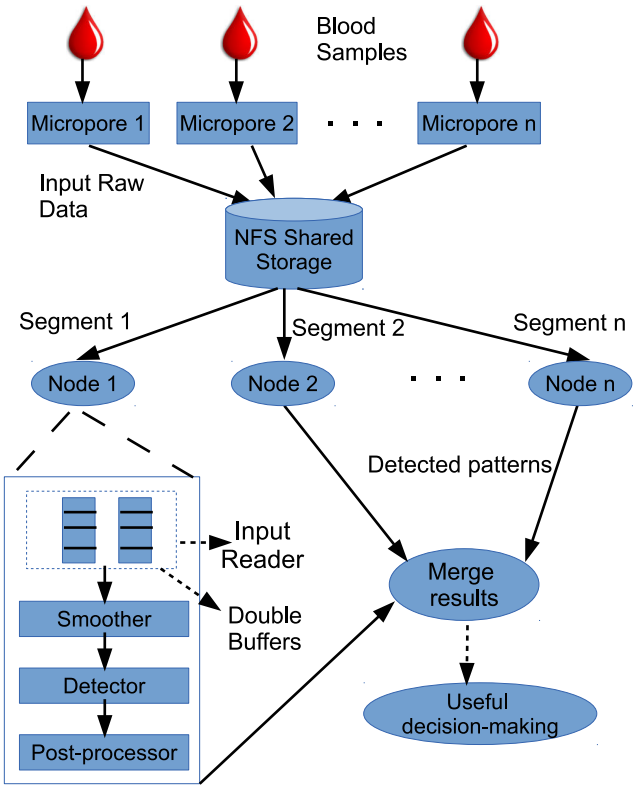


Figure 1. Distributed Detection Framework: Data can be collected from multiple bio-nano sensors on a shared storage of a high-speed server. Clusters nodes retrieve their own data segment from the shared storage and process it in parallel. Each data segment is processed by an individual node in four steps, including pre-processor, smoother, detector, and post-processor that delivers results for useful decision-making.

A. Distributed Design

The data is collected on the shared storage of an NFS server and then distributed among participating nodes such that each node gets its own data segment using the system modules as below. The pre-processor at each node retrieves its data segment from the shared NFS storage and optimizes the overall processing from two perspectives: (i) the pre-processor overlaps the data transfer with the computation using double buffering; and (ii) converts the raw data into an integer format to avoid round-off errors, as well as enable reliable and data efficient processing. The raw data segment is read from the storage device in chunks for subsequent processing. However, the time required to process a given chunk comprises of the data transfer phase from the storage and the subsequent computation phase while the data is held in the main memory of the node. Double buffering enables us to partition the main memory into two buffers, which are switched alternatively between the data transfer and the computation phase. For example, copying of the data chunk k in buffer A and the computation on chunk $k-1$ in buffer B is achieved simultaneously. This enables processing of

the data chunks in a pipeline fashion in order to improve the overall performance.

In the next step, the smoother removes noise and baseline wanders from the integer data. The cut-off frequency is an important factor used for deciding which band of frequencies need to be passed in order to only capture the interesting patterns in time-domain. However, the cut-off frequency is inversely proportional to the size of the sampling window used in moving-average filtering. Higher cut-off frequency allows higher frequencies to pass, and thus results in slighter smoothing, and vice versa. We tested the moving-average technique with different size of sampling window, i.e., 5, 10, and 20, and found 5-sample moving average to be the most effective for our problem. Different techniques have been designed to detect peaks in temporal data [6]. The detector module detects pulses against a statistical threshold that is based on the mean and standard-deviation of the smoothed data. Furthermore, the mean and standard-deviation are computed from the number of samples equal to the size of window used in moving-average (5 in our case), and then used for the detection of patterns in the smoothed data samples as given by Eq. 1:

$$Threshold = mean - 4 \times std - deviation \quad (1)$$

The value of 4 in Eq. 1 is selected from empirical knowledge. However, our framework has the capability to automatically adapt to the changing characteristics (i.e., mean and standard-deviation) of the input data. The threshold is recomputed based on the initial k samples of the buffer of size N. The computed threshold stays constant for $N - k$ data samples of a given buffer. In addition, our framework allows for experts to adjust the size of sampling window in moving average technique and the domain-specific value in Eq. 1 in order to further tailor and tune the detection according to the dynamic characteristics of the collected data.

In case of high variations in data, the buffer size can be reduced in order to allow the threshold to quickly adapt to the data and vice versa. The threshold detects the pulses along with their width and amplitude in smoothed data. The acceptable range of pulses for our problem is greater than 3 data samples, and less than 45 data samples. From the domain knowledge, we know that fewer samples (i.e., less than or equal to 3) are noisy pulses, while pulses with a width less than or equal to 45 data samples constitute a useful pulse that stems from a human cell [3]. The detected pulses along with their features are delivered to the post-processor. The post-processor stores the features of the detected pulses in a comma separated values (csv) format on the storage device, which can be later used for further analysis in the form of scatter plots.

B. Algorithm

During the initialization phase, the data is split into number of segments equal to the number of nodes, and assigned to individual nodes according to static data distribution strategy. Each node then processes its data segment through several time-steps. Note that the size of double buffers is always less than or equal to the size of an individual data segment. The duration of a time-step is directly proportional to the size of double buffers that is used. Larger buffers result in coarser time-steps and thus the given data segment can be processed within few time-steps. Conversely, smaller buffers result in fine-grained time-steps and therefore, requires large number of time-steps in order to process a given data segment.

Algorithm 1 Distributed Detection Algorithm

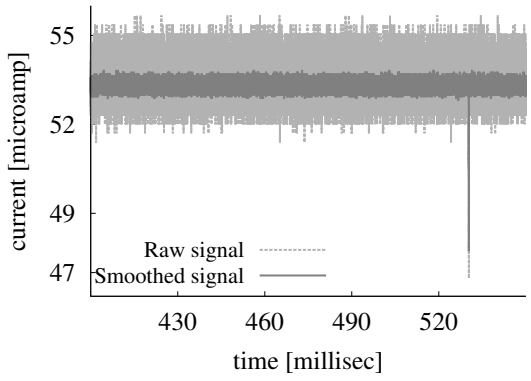
```
Initialize();
for all sample ∈ data − segment do
  DataConversion();
end for
for all sample ∈ data − segment do
  Smoother();
end for
for all sample ∈ data − segment do
  Detector(); // detects patterns
end for
for all patterns ∈ detected − patterns do
  ComputeFeatures();
end for
```

Algorithm 1 shows that during a given time-step, each node concurrently processes its assigned data segment based on the offset, as shown in the following steps:

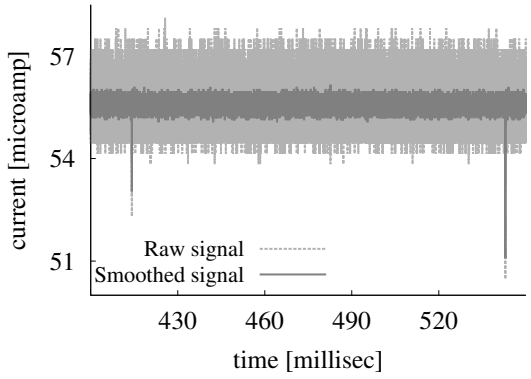
1) Initially, each node reads the assigned data segment using double buffering technique and converts the raw data into integer data.
2) The converted data is then de-noised using smoothing.
3) Patterns in the de-noised data are detected against the threshold, which is computed from the statistics of the data.
4) Compute useful features (i.e., width and amplitude) of the detected patterns.

III. Evaluation

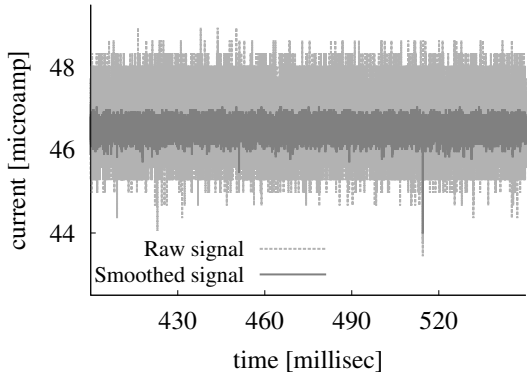
In this section, we first explain the experimental setup of our target distributed system. Then, we show the impact of different levels of smoothing and threshold on the detection of different cell types and summarize statistics of distinguishing features of the detected pulses. Next, we analyze the speedup achieved on a distributed systems in comparison to a single node for an increasing size of input data. We also examine the impact of the size of double buffers on overall execution time. Finally, we show the performance impact of using different interconnects for the nodes, such as 1 Gbps Ethernet and Infiniband.



(a) Temporal data of Cancer Cells.

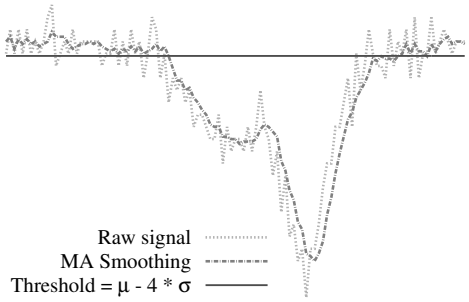


(b) Temporal data of WBCs.

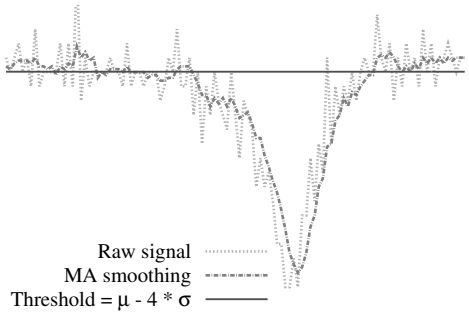


(c) Temporal data of RBCs.

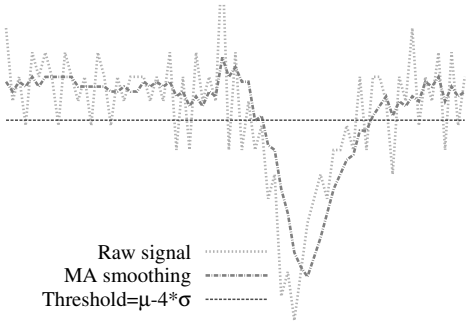
Figure 2. Noisy raw data and its smoothed version for different cell types with distinguishing pulses. Different features of the RBCs, WBCs, and Cancer cells are summarized in Table I.



(a) Typical Cancer pulse.



(b) Typical WBC pulse.



(c) Typical RBC pulse.

Figure 3. Typical pulses from each cell type and their moving-average filtering with sampling window size of 5.

A. Experimental Setup

Our experimental testbed consists of 9 server nodes, in which one node is used as master node and the remaining 8 are worker nodes. Each node is equipped with two 64-bit Intel Quad-Core Xeon processors, clocked at 2.8 GHz with an 8 GB of main memory (as reported by commands, `cat /proc/cpuinfo` and `cat /proc/meminfo`, respectively) and a 500 GB, 7200 RPM Seagate Barracuda ES.2 SATA disk. The eight cores on each node are organized as core 0, 1, 2, and 3 on processor 0, and cores 4, 5, 6, and 7 on processor 1. The operating system used is Linux CentOS 6 with kernel version 2.6.32. We implemented our framework in C using `gcc` compiler version 4.4.4.

Datasets: The biological raw datasets are collected from the translocation of a typical biopsy sample via a micropore. The sample consists of Red Blood Cells (RBCs), White

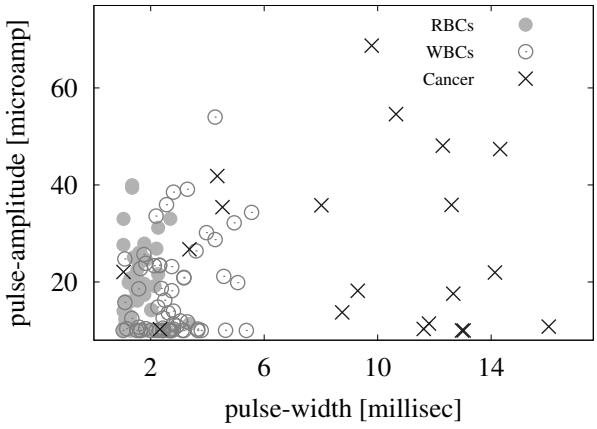


Figure 4. Scatter plot of different types of cells and their features detected from a mixture of cell types.

Pulse Features	RBCs	WBCs	Cancer Cells
Pulse Amplitude (microamp)	1.6 ± 5	3.14 ± 1	8.3 ± 4
Pulse Width (msec.)	15.0 ± 90	27.8 ± 31	30.1 ± 25

Table I
SUMMARY STATISTICS OF PULSE FEATURES FROM RED BLOOD CELLS (RBCS), WHITE BLOOD CELLS (WBCS), AND CANCER CELLS.

Raw Input Data			Integer Data	Double Buffers
Num. Profiles	Samples (Billion)	Size (GB)	Size (GB)	Max. Size (GB)
90	0.36	10	1.44	0.72
180	0.72	20	2.88	1.44
270	1.08	30	4.32	2.16
540	2.16	60	8.64	2.16

Table II
THE MAXIMUM SIZE OF DOUBLE BUFFERS THAT CAN BE USED IN COMPARISON TO THE INTEGER DATA, I.E., CONVERTED FROM RAW INPUT DATA.

Sampling Type	RBCs	WBCs	Cancer Cells
Raw signal	± 465.9	± 469.1	± 492.1
5-sample smoothing	± 152.7	± 188.9	± 181.0
10-sample smoothing	± 93.2	± 143.3	± 135.9
20-sample smoothing	± 65.5	± 124.8	± 116.2

Table III
SUMMARY OF THE VARIATION REDUCED IN THE RAW DATA DUE TO DIFFERENT LEVELS OF SMOOTHING.

Blood Cells (WBCs), and Cancer cells. A typical profile of cells contains around 4 million samples recorded over a period of 2.2 microseconds. The overall data collected from a sample consists of 90 profiles (i.e., 360 million samples = 10 GB of raw data), as shown in Table II. Each sample has a resolution of 2 bytes and can measure up to $65,535$ nanoamperes.

Micropore assembly: The biopsy sample is translocated through a Micropore of 12 micrometer radius and 200 nm long. The calibration of sampling frequency is an important factor to achieve the maximum throughput out of the pore. Decreasing sampling frequency results in a stable baseline with less noise, but cannot capture the useful translocation events at finer granularity. Conversely, higher sampling frequency results in the noisy data, which can suppress some of the translocation events. The optimal sampling frequency found for the micropore is 0.4 MHz.

B. Detection of Target Corpuscles

The level of threshold and the extent of smoothing are the two important parameters in order to effectively detect useful pulses. Changing the level of smoothing or threshold affect the shape and count of the detected pulses, respectively.

Impact of Smoothing: Smoothing helps to eliminate the noise and baseline shifts, however, it also reduces information available in raw data, i.e., it affects shape of pulses, as shown in Figure 2. Larger smoothing reduces variations in the baseline to a greater extent, but at the cost of significantly changing the pulse shape. Smoothing with a sample size of 5 results in better pulse shape as compared to a sample size of 10 and 20, which results in higher reduction of noise and deterioration of pulses. For clarity, we have only shown smoothing achieved by 5-sample moving average in Figure 3. The results also demonstrate that cancer pulses are larger than other cell types and retain their shapes even after smoothing, thus making them amenable to such proposed automated detection.

Impact of Threshold: Different thresholds result in different count of the detected pulses. Threshold closer to the baseline (e.g., $Threshold = mean - 3 \times std - deviation$) results in the detection of noisy pulses (false positives) in addition to the useful pulses. However, threshold away from the baseline (e.g., $Threshold = mean - 5 \times std - deviation$) results in miss-detection of useful pulses that are smaller in size (false negatives). Smoothing with 5-sample moving average followed by a $Threshold = mean - 4 \times std - deviation$ is found optimal for our datasets, as captured in Figure 3.

Detected Pulses: Figure 4 shows the scatter plots of the detected pulses from all the three different types of data. The plots show the width and amplitude of the detected pulses along x-axis and y-axis, respectively. The reason for horizontal spread observed in the plot is due to smoothing that actually reduces the amplitude of the pulse by eliminating vertical fluctuations in the pulse, however, this increases the horizontal width of the pulses. We also observe overlap among different types of pulses. This is because some

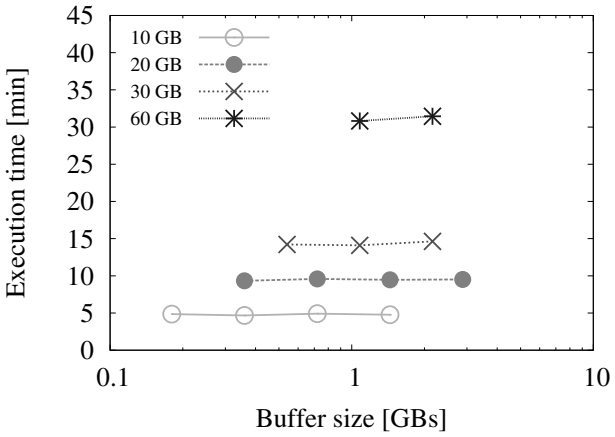


Figure 5. Scalability of our framework on a single node.

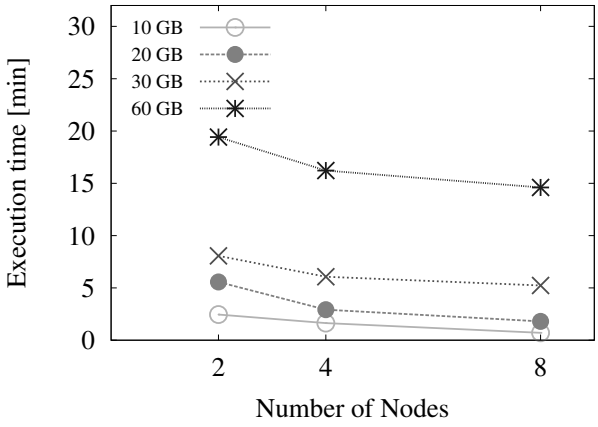


Figure 6. Scalability of our distributed framework on cluster nodes.

clustered smaller pulses of cancer cells closely resemble the larger pulses of WBCs due to their similar widths and amplitudes. The average of the translocation time and the amplitude of the detected pulses show that these features are different for each cell type and that they differ significantly in their size and stiffness i.e., the extent to which they block the pore when the pass through it, as shown in Table I.

Pulse Statistics: Increasing smoothing decreases variation in the data, as shown in Table III. Greater smoothing (e.g., 20-sample moving-average) results in higher suppression of noise in the original signal (i.e., the standard deviation for RBCs is reduced from ± 465.9 nanoamperes to ± 65.5 nanoamperes), as compared to 10-sample and 5-sample moving average, which results in fairly small reduction in noise. As show in Table I, Cancer pulses have greater pulse amplitude and width than other cell types. Furthermore, the p-value <0.01 for the translocation time shows that the cell types significantly differ from each other. In contrast, p-value >0.01 for pulse amplitude shows that the cell types do not differ significantly.

C. Performance and Scalability

Next, we discuss the performance of automated pulse detection on a single node versus multiple nodes on a distributed system. We show the scalability achieved over multiple nodes. In addition, we also discuss the impact of changing the size of double buffers on overall execution time.

An Increasing Input Raw Data: The input data processed by our distributed framework ranges from 10 GB to 20 GB, 30 GB, and 60 GB. These datasets correspond to the raw data collected from typical blood sample(s) when translocated through a solid-state pore. Furthermore, when a 60 GB raw data is converted to integer data, it reduces to about 8.64 GB, slightly larger than the memory footprint of a cluster node, as shown in Table II. The time taken to process these different datasets on a single node versus multiple distributed nodes is shown in Figure 5 and Figure 6, respectively.

Performance on a Single Node: We show the scalability achieved with a single node on our target cluster machine. The 10 GB raw data is processed for different size of double buffers (i.e., 180 MB, 360 MB, 720 MB, and 1.44 GB), 20 GB with a double buffer size of 360 MB, 720 MB, 1.44 GB, and 2.88 GB, and so on, as captured in Figure 5. The maximum size of double buffers is kept less than the total size of the integer data in order to make sure that we process the given data in chunks and do not overflow the memory footprint of the machine. We observe linear scalability for an increasing size of input data. Additionally, we do not see significant change while changing the size of double buffers. However, we see slight degradation in performance, when we increase the size of double buffers i.e., in case of 30 GB and 60 GB.

Impact of Double Buffers: The execution time with respect to different double buffer sizes for an increasing size of input raw data is shown in Figure 5. We do not see significant difference in the execution time for different double buffers. This shows that the results are scalable with varying size of double buffers. However, slight overhead is incurred in case of large buffer size, such as when using the 2.16 GB double buffers to process 60 GB of raw data, as captured in Figure 5.

Speedup Achieved on Multiple Nodes: Execution of our distributed detection system for 2, 4, and 8 nodes is shown in Figure 6. We observe the performance when the number of nodes is increased for a given dataset. However, we do not see significant performance improvement when the size of double buffers is larger, due to the overhead incurred in the critical path. Additionally, we are able to process 60 GB of

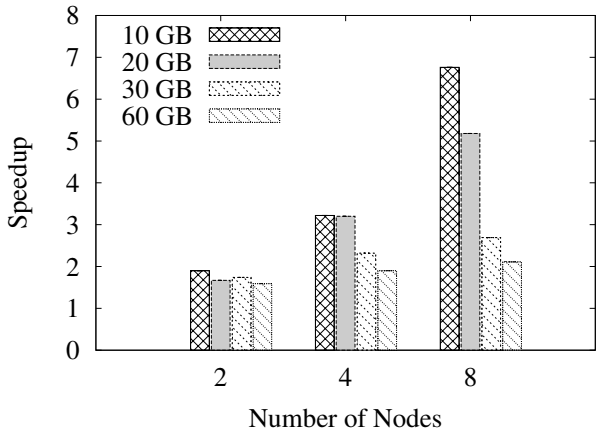


Figure 7. Speedup for an increasing size of input raw data w.r.t. different number of nodes.

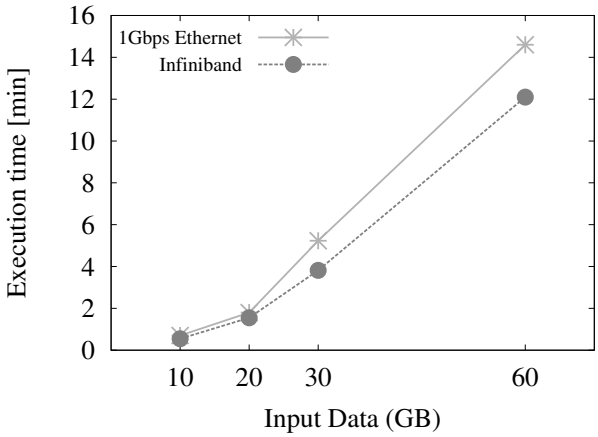


Figure 8. Comparison of Execution time on 1 Gbps Ethernet vs. Infiniband.

raw data, collected from six biopsy samples, using 8 nodes in less than 15 minutes.

Overhead in Speedup: The overhead in speedup is smaller in the case of fewer nodes as compared to the large number of nodes as shown in Figure 7. In case of 2 nodes, the overhead in speedup is $5\%, 16.2\%, 14.2\%$, and 20.1% for an input data of 10 GB, 20 GB, 30 GB, and 60 GB, respectively. While in case of 4 nodes, we observe an overhead of $20\%, 22\%, 42\%$, and 53% for an input of 10 GB, 20 GB, 30 GB, and 60 GB, respectively. However, the parallelization of 10 GB, 20 GB, 30 GB, and 60 GB, results in $16\%, 35.2\%, 66\%$, and 78% overhead, respectively. We observe that the overhead increases with the increase in the size of input data.

Impact of Interconnects: We observe a large difference between the underlying interconnects, i.e., 1 Gbps Ethernet and Infiniband. The performance increase achieved from the Infiniband is 23.1% compared to 1 Gbps Ethernet, as shown in Figure 8.

Selection of Parameters: In case of moving-average filtering, a size of 5 for the sampling window is found optimal for the signal-to-noise ratio in our datasets. Based on the empirical knowledge, the value of k=4 is found suitable in order to subtract k times standard-deviation from the mean of the data and thus, compute the threshold, as shown in Eq. 1. Finally, the increase in size of double buffers is scalable with an increasing size of input data, as far as, their size (double buffers) do not overflow the memory footprint.

IV. Related Work

In this section, we present existing pattern detection approaches and a brief background about solid-state micropores that are closely related to our work.

Grid Computing: Grid computing has been used to leverage large-scale scientific applications, including genomics [12], [13], [14], [15], proteomics [16], [17], physics [18], and astronomy [19] etc. Furthermore, distributed computing has been used to enable real-time response from an online collection of large-scale ECG data [20], simulations of diffusion MRIs [21], and cardiology [22]. In this work, we aim to show the use and scalability of distributed systems in processing large amount of raw data generated by micropore-based experimental setup, for disease diagnosis.

Pattern Detection Techniques: Recent trends in medical research and modern clinical setups show an increase in recording important physiological signals, which substantially involves the detection of patterns (peaks/troughs) in the target signals. Numerous pattern-detection algorithms have been developed [23], [24], [25], [26], [27], however, these are domain specific. Efforts have been made to build a generalized mathematical model [28] for peak detection algorithms. Nonetheless, such model suffers from a large number of false alarms and cannot be used in a particular domain, unless properly tailored and tuned. In contrast, our goal is to develop such a pattern detection technique that is quicker and amenable for online monitoring and prognosis.

Solid-State Micropores: These are tiny orifices in silicon-based membranes used to measure the passage of human cells through them in the form of electrical pulses [1]. Corpuscles passing via orifices blocks the micropore and results in a translocation event in the output current. The strength of the event is determined from the degree to which the pore is blocked by a target corpuscle. These events are registered as pulses and their features depict patterns specific to the human cell type [3], [2]. Biomechanical properties of the diseased cells, such as tumor cells, are known to be different than the normal cells [3], [29], [30], [31]. Furthermore, the diseased cells are more elastic than

the other cell types [32], [33], [34], [35] and the recorded pulses significantly differ in the case of tumor cells [3]. The downside of these devices is that the detection and analysis is all done manually. Therefore, it is important to automate and accelerate the detection and identification of different biological targets in the high-throughput raw data generated by micropores. Our work benefits from improvements in micropore technology, and these works are complementary to ours.

V. Discussion

In this paper, we present a distributed detection approach that can acquire and process raw data collected from bio-nano sensors at a very high speed. In our experiments, we show that the designed framework can process data at a maximum throughput of 36 MB/sec. In other words, the framework has the ability to support the collection of raw data and its online processing from about 18 micropores simultaneously (each calibrated at a sampling rate of 2 MB/sec). In order to further accelerate the detection process, we need to incorporate fine-grained parallelism at the chunk level with the use of accelerator such as graphics processing units (GPUs). Such faster platforms will demand an increased sampling rate from the micropore-based experimental setup. Nevertheless, the sampling rate depends on many factors including the speed and size of the biological targets that are translocated via micropores.

The accuracy achieved with our detection technique is about 63%, mainly because of the highly dynamic nature of data generated from bio-nano sensors, and secondly, due to the moving-average filtering and subsequent threshold-based detection that is suitable for an online monitoring and clinical setup to make quicker decisions. On the other hand, such a technique is not quite robust against noise and more sophisticated approaches such as machine learning techniques are needed. Such approaches, while computationally expensive, are more error-resilient and can differentiate between the noise and information (actual patterns) with an increased accuracy. While developing more robust and error-resilient detection algorithms, the need for an increased accuracy also stresses advancements on bio-nanotechnology to detect biological targets with an increased signal-to-noise ratio, while sensing them at finer granularity.

Our distributed computing framework splits the overall data on a shared storage among nodes, and enables each node to retrieve the assigned data chunk with its offset. Chances are rare for useful patterns to span across the boundaries of the split data segments and become a false alarm. The reason is that the patterns are very sparse in the data and the size of patterns is orders of magnitude smaller than a data segment (few bytes vs. gigabytes).

In addition to splitting the data into segments across nodes, we use double buffers within each node to overlap I/O with the computation, while enable processing of large datasets in chunks. Furthermore, to address large-scale processing of data, which is greater than the memory footprint, requires careful consideration of the memory usage. In order to achieve this, the breakdown for the size of double buffers is the physical memory limit. In our experiments, we are able to process 60 GB of raw data by keeping the aggregated size of double buffers to a maximum of 4.26 GB (2.13 GB for an individual buffer). However, increase in the size of buffers results in an increase in the critical path of the overall computation, and become effective until after reading the very first chunk of a data segment. In addition to that, buffers larger than the size of memory footprint can either crash due to segmentation faults, or results in paging, if virtual memory is enabled. Conversely, very small buffers result in too much swapping and I/O that leads to an increased communication rather than the computation itself. Therefore, we need an optimal size of double buffers that is selected based on the total size of the input data in order to read large amount of raw data efficiently. Our experiments show that as far the size of double buffers is less than the size of the integer version of input data and within the memory footprint, delivers an acceptable performance.

In our distributed setup, the splitting of data segments across nodes and the assignment of functional nodes is determined statically. In our future work, we aim to determine the status of nodes on fly and subsequently, utilize the node only if it is alive in order to achieve fault tolerance. Furthermore, we have assumed an even distribution of workload across the participating nodes. This is possible in our framework, since the interesting patterns are very sparse in the datasets and therefore, a node will barely have larger number of patterns than its neighbors. Nonetheless, with the advent of high-performance sensors with high-quality of raw data (i.e., enhanced information with respect to the noise and baseline shifts), we will need to design dynamic load balancing techniques. One such strategy is to throttle the amount of data fed to a node on fly and assign data to an another node that is idle or relatively less-burdened.

VI. Conclusions

We design a distributed framework for the automated and accelerated detection of cancer cells in temporal cellular spike streams collected from solid-state micropores. Our framework splits the raw data (collected in shared storage) among the worker nodes. For improved performance, each node acquires the data segment assigned to it, using double buffers, and processes it in multiple time-steps. The total number of time-steps and the duration of an individual time-step depends on the size of the assigned data segment and the size of double buffers. In each time-step, a node performs data conversion, noise removal, detection of pulses against the threshold, and finally, merging the detected pulses from the individual nodes.

In summary, our framework can support instant data processing from multiple micropores in an online setup, where many biopsy samples need to be collected, processed, and analyzed quickly for useful decision making. The framework has the ability to distribute the data across the nodes, making it an efficient tool for faster data processing.

ACKNOWLEDGEMENT

The authors would like to thank Dr. Samir M. Iqbal for useful discussions on the data analysis. The chip fabrication was carried out at the Nanotechnology Research and Education Center in the University of Texas, Arlington and the University of Illinois at Urbana-Champaign's Micro and Nanotechnology Laboratory. This work was sponsored in part by the NSF under Grants CNS-1016793 and CCF-0746832.

REFERENCES

[1] S. M. Iqbal, D. Akin, and R. Bashir, "Solid-state nanopore channels with dna selectivity," *Nature nanotechnology*, vol. 2, no. 4, pp. 243–248, 2007.

[2] A. Hafeez, W. Asghar, M. M. Rafique, S. M. Iqbal, and A. R. Butt, "Gpu-based real-time detection and analysis of biological targets using solid-state nanopores," *Medical & biological engineering & computing*, vol. 50, no. 6, pp. 605–615, 2012.

[3] W. Asghar, Y. Wan, A. Ilyas, R. Bachoo, Y.-t. Kim, and S. M. Iqbal, "Electrical fingerprinting, 3d profiling and detection of tumor cells with solid-state micropores," *Lab on a Chip*, vol. 12, no. 13, pp. 2345–2352, 2012.

[4] J. Alvarez-Ramirez, E. Rodriguez, and J. Carlos Echeverría, "Detrending fluctuation analysis based on moving average filtering," *Physica A: Statistical Mechanics and its Applications*, vol. 354, pp. 199–219, 2005.

[5] M. S. Lewicki, "A review of methods for spike sorting: the detection and classification of neural action potentials," *Network: Computation in Neural Systems*, vol. 9, no. 4, p. R53–R78, 1998.

[6] G. Palshikar *et al.*, "Simple algorithms for peak detection in time-series," in *Proc. 1st Int. Conf. Advanced Data Analysis, Business Analytics and Intelligence*, 2009.

[7] S. M. Iqbal and R. Bashir, "Nanoelectronic-based detection for biology and medicine," *Springer Handbook of Automation*, pp. 1433–1449, 2009.

[8] D.-H. Kim, N. Lu, R. Ma, Y.-S. Kim, R.-H. Kim, S. Wang, J. Wu, S. M. Won, H. Tao, A. Islam *et al.*, "Epidermal electronics," *Science*, vol. 333, no. 6044, pp. 838–843, 2011.

[9] D. Deamer, "Nanopore analysis of nucleic acids bound to exonucleases and polymerases," *Annual review of biophysics*, vol. 39, pp. 79–90, 2010.

[10] W. Asghar, M. Islam, A. S. Wadajkar, Y. Wan, A. Ilyas, K. T. Nguyen, and S. M. Iqbal, "Plga micro-and nanoparticles loaded into gelatin scaffold for controlled drug release," *Nanotechnology, IEEE Transactions on*, vol. 11, no. 3, pp. 546–553, 2012.

[11] H. Wang, W. Wang, J. Yang, and P. S. Yu, "Clustering by pattern similarity in large data sets," in *Proceedings of the 2002 ACM SIGMOD international conference on Management of data*. ACM, 2002, pp. 394–405.

[12] K. Krampis, T. Booth, B. Chapman, B. Tiwari, M. Bicak, D. Field, and K. E. Nelson, "Cloud biolinux: pre-configured and on-demand bioinformatics computing for the genomics community," *BMC bioinformatics*, vol. 13, no. 1, p. 42, 2012.

[13] L. D. Stein *et al.*, "The case for cloud computing in genome informatics," *Genome Biol*, vol. 11, no. 5, p. 207, 2010.

[14] M. Baker, "Next-generation sequencing: adjusting to data overload," *Nature Methods*, vol. 7, no. 7, pp. 495–499, 2010.

[15] R. C. Taylor, "An overview of the hadoop/mapreduce/hbase framework and its current applications in bioinformatics," *BMC bioinformatics*, vol. 11, no. Suppl 12, p. S1, 2010.

[16] S. P. Brown and S. W. Muchmore, "High-throughput calculation of protein-ligand binding affinities: Modification and adaptation of the mm-pbsa protocol to enterprise grid computing," *Journal of chemical information and modeling*, vol. 46, no. 3, pp. 999–1005, 2006.

[17] A.-A. Tantar, N. Melab, E.-G. Talbi, B. Parent, and D. Horvath, "A parallel hybrid genetic algorithm for protein structure prediction on the computational grid," *Future Generation Computer Systems*, vol. 23, no. 3, pp. 398–409, 2007.

[18] E. Deelman, C. Kesselman, G. Mehta, L. Meshkat, L. Pearlman, K. Blackburn, P. Ehrens, A. Lazzarini, R. Williams, and S. Koranda, "Griphyn and ligo, building a virtual data grid for gravitational wave scientists," in *High Performance Distributed Computing, 2002. HPDC-11 2002. Proceedings. 11th IEEE International Symposium on*. IEEE, 2002, pp. 225–234.

[19] G. B. Berriman, E. Deelman, J. C. Good, J. C. Jacob, D. S. Katz, C. Kesselman, A. C. Laity, T. A. Prince, G. Singh, and M.-H. Su, "Montage: a grid-enabled engine for delivering custom science-grade mosaics on demand," in *Astronomical Telescopes and Instrumentation*. International Society for Optics and Photonics, 2004, pp. 221–232.

[20] S. Pandey, W. Voorsluys, S. Niu, A. Khandoker, and R. Buyya, "An autonomic cloud environment for hosting ecg data analysis services," *Future Generation Computer Systems*, vol. 28, no. 1, pp. 147–154, 2012.

[21] J.-R. Li, D. Calhoun, C. Poupon, D. Le Bihan *et al.*, "Effcient cartesian grid simulation tool for diffusion mri," 2012.

[22] J. F. G. Eijo, M. Risk, F. P. Castrillo, C. S. Ortega, M. B. Fernandez, A. P. Diaz, M. R. del Solar, and R. R. Pollan, "Cardiogrid: a framework for the analysis of cardiological signals in grid computing," in *Journal of Physics: Conference Series*, vol. 313, no. 1. IOP Publishing, 2011, p. 012010.

[23] C. Yang, Z. He, and W. Yu, "Comparison of public peak detection algorithms for maldi mass spectrometry data analysis," *BMC bioinformatics*, vol. 10, no. 1, p. 4, 2009.

[24] P. Du, W. A. Kibbe, and S. M. Lin, "Improved peak detection in mass spectrum by incorporating continuous wavelet transform-based pattern matching," *Bioinformatics*, vol. 22, no. 17, pp. 2059–2065, 2006.

[25] I. Azzini, R. DellAnna, F. C. F. Demichelis, A. Sboner, and E. B. A. Malossini, "Simple methods for peak detection in time series microarray data." *Proc. CAMDA04 (Critical Assessment of Microarray Data)*, 2004.

[26] J. Kleinberg, "Bursty and hierarchical structure in streams," *Data Mining and Knowledge Discovery*, vol. 7, no. 4, pp. 373–397, 2003.

[27] J. Pan and W. J. Tompkins, "A real-time qrs detection algorithm," *Biomedical Engineering, IEEE Transactions on*, no. 3, pp. 230–236, 1985.

[28] B. S. Todd and D. C. Andrews, "The identification of peaks in physiological signals," *Computers and biomedical research*, vol. 32, no. 4, pp. 322–335, 1999.

[29] G. Y. Lee and C. T. Lim, "Biomechanics approaches to studying human diseases," *Trends in biotechnology*, vol. 25, no. 3, pp. 111–118, 2007.

[30] M. Brandao, A. Fontes, M. Barjas-Castro, L. Barbosa, F. F. Costa, C. Cesar, and S. Saad, "Optical tweezers for measuring red blood cell elasticity: application to the study of drug response in sickle cell disease," *European journal of haematology*, vol. 70, no. 4, pp. 207–211, 2003.

[31] E. A. Evans and P. L. La Celle, "Intrinsic material properties of the erythrocyte membrane indicated by mechanical analysis of deformation," *Blood*, vol. 45, no. 1, pp. 29–43, 1975.

[32] S. E. Cross, Y.-S. Jin, J. Rao, and J. K. Gimzewski, "Nanomechanical analysis of cells from cancer patients," *Nature nanotechnology*, vol. 2, no. 12, pp. 780–783, 2007.

[33] M. Lekka, P. Laidler, D. Gil, J. Lekki, Z. Stachura, and A. Hrynkiewicz, "Elasticity of normal and cancerous human bladder cells studied by scanning force microscopy," *European Biophysics Journal*, vol. 28, no. 4, pp. 312–316, 1999.

[34] G. Vona, A. Sabile, M. Louha, V. Sitruk, S. Romana, K. Schütze, F. Capron, D. Franco, M. Pazzagli, M. Vekemans *et al.*, "Isolation by size of epithelial tumor cells: a new method for the immunomorphological and molecular characterization of circulating tumor cells," *The American journal of pathology*, vol. 156, no. 1, pp. 57–63, 2000.

[35] L. Zabaglo, M. G. Ormerod, M. Parton, A. Ring, I. E. Smith, and M. Dowsett, "Cell filtration-laser scanning cytometry for the characterisation of circulating breast cancer cells," *Cytometry Part A*, vol. 55, no. 2, pp. 102–108, 2003.