A Performance Study of Secure Data Mining on the Cell Processor

Hong Wang∗, Hiroyuki Takizawa∗, and Hiroaki Kobayashi†
∗Graduate School of Information Sciences, Tohoku University, Sendai 980-8579, Japan
Email: wangh@sc.isc.tohoku.ac.jp, tacky@isc.tohoku.ac.jp
†Information Synergy Center, Tohoku University, Sendai 980-8578, Japan
Email: koba@isc.tohoku.ac.jp

Abstract—This paper examines the potential of the Cell processor as a platform for secure data mining on the future volunteer computing systems. Volunteer computing platforms have the potential to provide massive computing power. However, privacy and security concerns prevent using volunteer computing for data mining of sensitive data. The Cell processor comes with a hardware security feature. The secure volunteer data mining can be achieved by using this hardware security feature. In this paper, we present a general security scheme for the volunteer computing, and a secure parallelized K-Means clustering algorithm for the Cell processor. We also evaluate the performance of the algorithm on the Cell secure system simulator. Evaluation results demonstrate a large performance overhead introduced by the decryption process of the security features. Possible optimization for the secure K-Means clustering is discussed.

I. INTRODUCTION

The world’s computing power is no longer primarily concentrated in supercomputer centers. Instead, it is distributed in hundreds of millions of personal computers and game consoles. Volunteer computing platforms have the potential to provide massive computing power. By September 2007, the most powerful volunteer computing platform - Folding@home [1] achieved more than one Petaflops computing power by connecting more than 600,000 PlayStation3. The second powerful volunteer computing platform - BOINC provided a sustained processing power of 350 Teraflops [2] by November 2006. In contrast, the fastest conventional supercomputer, BlueGene/L achieves a maximum LINPACK performance of 280.6 Teraflops [3]. However, volunteer computing cannot be applied to the sensitive data processing, due to privacy and security concerns.

Latest generation game consoles are equipped with high performance processors. Therefore, these game consoles have the potential to become ideal peers of future volunteer computing systems. The CPU of PlayStation3 is the Cell processor, which was jointly developed by a Sony, Toshiba, and IBM alliance [4], [5]. The Cell processor comes with hardware security features [6]. On top of the hardware security features, secure volunteer data mining can be achieved on the Cell based devices. We design a secure data processing method for volunteer computing on the Cell processor, and then apply it to an implementation of a typical data mining algorithm, called the K-Means clustering [7].

The rest of the paper is organized as follows. Related work of secure data mining and our target application - K-Means clustering are introduced in Section II. Section III introduces the Cell architecture, our previous work on performance evaluation of K-Means clustering on the Cell processor, and the hardware security features of the Cell processor. Section IV presents a general security scheme for the volunteer computing, and a secure parallelized K-Means clustering algorithm for the Cell processor. Section V demonstrates performance overhead that is introduced by security function, and discusses the solution for this performance issue. Section VI concludes and summarizes the paper.

II. RELATED WORK

Privacy preserving data mining [8] is a novel research direction in data mining. The main object in privacy preserving data mining is to develop algorithms for modifying the original data in some way, so that the private data and private knowledge of any participants remain private even after the mining process. The typical modification methods include: perturbation, blocking, aggregation/merging, swapping, and sampling. A number of algorithms have been designed for different data mining techniques, such as classification, association rule discovery, and clustering. These algorithms can be classified into the following three types:

- Heuristic-based: has side effect due to selective data modification or sanitization.
- Cryptography-based: multi parties conduct data mining on their private data. None of these parties is willing to disclose its own data and found knowledge. It is referred to as the Secure Multiparty Computation (SMC) problem.
- Reconstruction-based: perturbing the data and reconstructing the distribution at an aggregate level.

For all the above three types, specified algorithms are required to handle the privacy issues for different data mining techniques. For example, numbers of specified algorithms have been proposed for cryptography-based association [9], [10], decision tree induction [11], [12], and clustering [13]. The heuristic-based [14], [15] and reconstruction-based algorithms [16]–[19] introduce side effect on the found knowledge. The cryptography-based algorithms can only apply to the SMC problems which is not suitable for volunteer computing, because of the different ownership of sensitive data.
To process sensitive data on a volunteer computing platform, privacy preserving technique is required to guarantee that the owner of a volunteer peer cannot access sensitive data and the knowledge inside the data. The security features of the Cell processor makes it possible to process sensitive data without disclose the data and the found knowledge, while no side effect is introduced. This paper explores the potential of the Cell processor for sensitive data processing, with an important data mining application: K-Means clustering [7].

K-Means clustering [7] is one of the simplest learning algorithms for data clustering. As shown in Figure 1, the procedure follows a simple and easy way to cluster a dataset into a certain number of subsets.

Let $k$ be the number of clusters. In the first step of the procedure shown in Figure 1, $k$ centroids are selected from the dataset, one for each cluster. The next step is the beginning of a loop, which initializes “tag” to 0. The second step of the loop is to associate each object of the dataset to the nearest centroid. Thirdly, $k$ new centroids are calculated. After that, a new binding has to be done between the dataset objects and the nearest new centroid. If any object’s membership changes, “tag” is set to 1. Then, the new cluster centroids are calculated. The last step in the loop is the termination test of the “tag” value. If the “tag” value is 0, the loop is terminated and post-process starts.

As a result of this loop we may notice that the $k$ centroids change their locations step by step until no more changes are done. In other words, the centroids do not move any more if none of the object’s membership changes. The algorithm iteratively reduces an objective function:

$$
\sum_{i=1}^{k} \sum_{n \in S_i} |x_n - c_i|^2
$$

(1)

where $S_i$ is a cluster, and $|x_n - c_i|^2$ is the distance between a data object $x_n$ and the cluster centroid $c_i$. The function is minimized by iterating the loop described in Figure 1.

III. THE CELL PROCESSOR

A. Cell Architecture Overview

As shown in Figure 2, the Cell processor is a single-chip multiprocessor with nine cores [4], [5]. The nine cores, main memory and I/O are connected via the Element Interconnect Bus (EIB). One of the cores, the PowerPC Processor Element (PPE) is responsible for overall control of the system. The Synergistic Processor Elements (SPE) are 128-bit SIMD cores optimized for data-rich operations. The PPE allocates computation-intensive applications to the SPEs for processing. Each SPE contains a RISC core called Synergistic Processing Unit (SPU), 256KB software-controlled Local Store (LS), and a Memory Flow Controller (MFC) that controls the DMA transfer. Each SPU can only access its own LS directly.

Data transfer between the main memory and the LS of SPEs is handled by software-controlled DMA operations. The SPU can execute instructions while the MFC processes the DMA operations concurrently. Thus, we can use double-buffered DMA transfer to overlap computation and DMA transfer.

The Cell processor’s peak single precision performance is 25.6Gflops@3.2GHz per SPE. Its peak double precision performance (1.83Gflops@3.2GHz per SPE) is not as high as single precision, but still impressive.

To examine the potential of the Cell processor for data mining, we have proposed parallelized and evaluated a parallelized K-Means algorithm [20]. As shown in Figure 3, the process is divided onto the PPE and the SPEs. The PPE is designed to be responsible for overall control of the system. The SPE’s architecture is optimized for computation-intensive applications. K-Means clustering involves massive data-parallelism, which is suitable for the SPE’s 128-bit SIMD operations. Thus, most of computation is mapped onto the SPEs to utilize their computing power. The PPE only calculates the new centroids with the local results from the SPEs.

To fully exploit the computing power of the Cell processor, we considered two challenges: the software-controlled memory hierarchy and the 128-bit SIMD ISA of the SPE. Software controlled double-buffered DMA transfer can mitigate memory latency. Thus, the following optimizations have been applied to our implementation:


- Use SIMD intrinsics in SPE thread to explore data-parallelism.
- Overlap computation and DMA transfer by double buffering to reduce DMA stalls.
- Unroll loops to provide better dual issue rates.

We have evaluated this algorithm using both a Cell processor of PlayStation3 and the Cell full system simulator. Evaluation results have proven the effectiveness of our optimizations for the Cell processor. The results have also indicated that K-Means clustering is a highly scalable application on the Cell processor. For the fine tuned code, 5.68x and 5.92x speedup can be achieved using six SPEs for single precision and double precision, respectively. Furthermore, in terms of both performance and power efficiency, the Cell processor greatly outperformed two other commodity processors (Athlon64 3400+ and PowerPC G4 1.67GHz). The Cell processor has been proved to be an ideal computing platform for data mining algorithms with massive data-parallelism.

B. Cell Security Features

The core of Cell’s security features is the isolation mode of SPE [6]. As shown in Figure 4, by isolating a SPE, its LS is locked for its own use. A small area of the LS is left open for communication purposes. External execution path control of the SPE is also disabled. Thus, the only possible external action for an isolated SPE is “cancel.” When a isolated SPE is cancelled, all the data in the LS and SPE are erased before external access is enabled. The basic functionality provided by the current security SDK includes: “decrypt in” function and “encrypt out” function. The first function decrypts encrypted data and puts the decrypted plaintext data in the LS. The second one does the work on the opposite direction.

By running a computing task on an isolated SPE, any access from outside of this SPE to this SPE’s LS is prohibited. Thus, to preserve the privacy of data, data can be encrypted by its owner. No one can decrypt the data without the right key. Only the computing task running on the isolated SPE can decrypt and access the data with the embedded key.

The correctness of the Cell processors security features, key hierarchy and API of the security features are out of scope of this paper. The detail of these topics can be found in [6], [21].

IV. SECURE K-MEANS CLUSTERING FOR VOLUNTEER COMPUTING

Besides the performance, the Cell processor is also compelling because of the hardware security features. These features enable it for secure data processing. Here, we present a method to insure the security of data processing by volunteer computing.

A. Problem Definition

On the volunteer computing platform, volunteer peers process the tasks dispatched from a server. Volunteer peers do not own the data inside the task that is dispatched to them. Therefore, any unauthorized access to the plaintext data of computing tasks should be disabled on the volunteer peers to guarantee the data security. The problem here is how to insure that the only possible access to the plaintext data is inside an isolated SPE, which runs the authorized computing task.
B. Data Encryption

Our security method relies on the isolation mode of SPE and its cryptography method. The only two places that plaintext data exists are the data owner’s server and volunteer peer’s isolated SPE.

As shown in Figure 5, before dispatching to the volunteer peer, plaintext data are encrypted by the data owner. The decryption requires a proper key that is embedded in the computing task and authorized by the owner. This task is authorized to run on the isolated SPE. Thus, this key is not accessible from anywhere else. Therefore, the encrypted data cannot be decrypted on any volunteer peers other than an isolated SPE running an authorized computing task. By insuring that the blue area in Figure 5 is only accessible for the data owner and the computing task authorized by the data owner, the data security is guaranteed.

C. Secure K-Means Clustering on the Cell Processor

Applying the above security method to K-Means clustering, secure volunteer data clustering can be achieved. The overall process flow is the similar to the algorithm we proposed in [20], with extra data decryption while loading data from main memory.

The main difference for clustering process on plaintext data and encrypted data is shown in Figure 6. While plaintext data is transferred from main memory to SPE’s LS requires only MFC to execute DMA commands, encrypted data decryption occupies both MFC and SPU. The SPU handles all the decryption computation. Thus, unlike the non-secure K-Means clustering on the Cell processor, the secure K-Means clustering cannot take advantage of the double buffering optimization. The effect on performance for the lack of double buffering will be discussed in the next section. Other modifications on the previous work are related to the isolation mode activation and other SDK specified functions [21].

V. PERFORMANCE EVALUATION AND DISCUSSION

This section first presents the performance evaluation results of both the secure K-Means clustering algorithm and the non-secure K-Means clustering algorithm on the Cell on the secure system simulator. We discuss the performance overhead for security. Then, a possible solution is proposed.

A. Evaluation Environment

In this section, all the evaluations of the secure K-Means clustering are done on top of the Cell secure system simulator. It is due to the fact that the hardware security features are not accessible on the physical hardwares with the current Cell security SDK. We evaluate the algorithms in the cycle-accurate mode to gather and compare performance statistics. For simplicity, we choose to model a 3.2GHz Cell with 8 SPEs and 25.6GB/s of memory bandwidth. The Cell processor in IBM QS20 Cell Blade Server has the same configuration. The evaluations of the non-secure K-Means clustering are done on top of the same simulator, while isolation mode is not enabled.

B. Performance Statistics of SPE Threads

Similar to the evaluation in our previous work, the dataset to be clustered for evaluation has 16384 objects; the number of coordinates is 64. Each buffer stores up to 1024 elements, either single precision floating point or double precision floating point. Table II shows a dynamic timing analysis of both the secure K-Means clustering and the non-secure K-Means clustering. The detail process statistics of the different process stage is also gathered, including data transfer cycles and buffer processing cycles. In our previous work, we presented the performance in terms of “Gflops.” Because of the massive computation cost for data decryption and lack of support for security features on the hardwares, it is not appropriate to discuss the performance of the secure K-Means clustering using “Gflops.” The performance is discussed in terms of the performance statistics. Table I lists the specification of evaluation environment.

To study the overhead for security features, we gather SPE performance statistics of both the secure K-Means clustering and the non-secure K-Means clustering. The detail process statistics of the different process stage is also gathered, including data transfer cycles and buffer processing cycles. In our previous work, we presented the performance in terms of “Gflops.” Because of the massive computation cost for data decryption and lack of support for security features on the hardwares, it is not appropriate to discuss the performance of the secure K-Means clustering using “Gflops.” The performance is discussed in terms of the performance statistics. Table I lists the specification of evaluation environment.

To study the overhead for security features, we gather SPE performance statistics of both the secure K-Means clustering and the non-secure K-Means clustering. The detail process statistics of the different process stage is also gathered, including data transfer cycles and buffer processing cycles. In our previous work, we presented the performance in terms of “Gflops.” Because of the massive computation cost for data decryption and lack of support for security features on the hardwares, it is not appropriate to discuss the performance of the secure K-Means clustering using “Gflops.” The performance is discussed in terms of the performance statistics. Table I lists the specification of evaluation environment.

B. Performance Statistics of SPE Threads

Similar to the evaluation in our previous work, the dataset to be clustered for evaluation has 16384 objects; the number of coordinates is 64. Each buffer stores up to 1024 elements, either single precision floating point or double precision floating point. Table II shows a dynamic timing analysis of both the secure K-Means clustering and the non-secure K-Means clustering on the dataset.

For both single precision and double precision, the buffer transfer cycles for the secure K-Means clustering is 30.7x of the corresponding value for the non-secure K-Means clustering. The reason is the massive computation cost for data buffer decryption. While plaintext data transfer from main memory to SPE’s LS requires MFC to execute DMA commands, encrypted data decryption occupies both MFC for DMA transfer and SPU for data decryption. Moreover, the
buffer transfer cycles for the secure K-Means clustering are 5.57x (single precision) and 1.67x (double precision) of the buffer clustering cycles, which means the decryption process takes the most of cycles during the process. While the buffer clustering cycles for both two algorithms are almost the same, the huge overhead for decryption makes the non-secure K-Means clustering outperforming the secure algorithm by 5.56x (single precision) and 2.53x (double precision), in terms of total buffer processing cycles.

Furthermore, since encrypted data decryption occupies both MFC for DMA transfer and SPU for data decryption, the double buffering optimization is not applicable for the secure K-Means clustering algorithm. The evaluation of previous work [20] shows that double buffering provides about 17% (single precision) and 10% (double precision) improvement for the non-secure K-Means clustering algorithm. The secure K-Means clustering algorithm cannot benefit from this optimization.

Due to the large computation cost for decryption and lack of double buffering, the overall processing speed of the secure K-Means clustering are about 10.4% (single precision) and 32.5% (double precision) of the original non-secure algorithm.

The double precision buffer clustering cycles is 3.33x of the one for single precision. It is because of the lack of optimization for double precision instructions. A double precision instructions takes 13 cycles on SPU. Unlike the single precision instructions, the double precision instructions are partially pipelined (the last 7 cycles). Furthermore, no other instructions are dual-issued with double-precision instructions [22]. Because of the relatively large buffer clustering cycles for double precision data, the performance overhead for double precision is less significant than the one for single precision.

### C. Optimization Discussion

The security method for volunteer computing introduces a huge performance overhead on the K-Means clustering algorithm. While the hardware security features make the Cell processor compelling, the performance overhead becomes an issue for applying security features to the volunteer computing. Because the huge performance overhead is mainly introduced by the decryption computation, we discuss the possible optimization on this issue.

Whenever an encrypted data is transferred to the LS of a SPE, it needs to be decrypted into plaintext data. Due to the size limit of the LS, it cannot store all the decrypted data. Thus, encrypted data has to be decrypted for each iteration of the K-Means algorithm.

If the dataset is small enough (e.g., object data is 128KB for each SPE), the decrypted data can be stored in the LS until the clustering process ends. In this way, the encrypted data is only decrypted once. Therefore, the performance can be speeded up by reducing cycles for extra decryption.

The above method is only applicable for small datasets. To process a large dataset, extra optimization is required. MicroCluster [23] is a technique that summarizes data before the final data processing. It first clusters all the data objects into number of micro clusters. Each micro cluster presents a number of data objects. Then, the required data processes (e.g., clustering, classification) are applied to the micro clusters. Because the number of micro clusters is much smaller than the number of objects, the processing speed can be improved.

To speed up the secure K-Means clustering on the Cell processor, we can apply the MicroCluster technique. By clustering the original objects to a proper number of micro clusters, the data of micro clusters can be stored in LS during the clustering process. Therefore, the overhead for decryption is reduced.

This optimization method can improve the processing speed of the secure K-Means clustering algorithm. However, the MicroCluster technique also leads to accuracy issues. The effect of this optimization on the accuracy and processing speed will be studied in our future work.

### VI. CONCLUSION AND FUTURE WORK

Modern generation game consoles are equipped with high performance processors and internet-ready. Therefore, these
game consoles have the potential to become ideal peers of future volunteer computing systems. By September 2007, the most powerful volunteer computing platform - Folding@home [1] achieved more than one Petaflops computing power by connecting more than 600,000 PlayStation3. However, volunteer computing cannot be applied to the sensitive data processing, due to privacy and security concerns. The Cell processor inside PlayStation3 is compelling for its high performance and the hardware security features. These advantages make the secure high performance volunteer data mining possible.

In this paper, we have briefly introduced the related work in privacy preserving data mining and our previous work of parallelized K-Means clustering algorithm for the Cell processor. Then, we have presented a secure K-Means clustering algorithm for the Cell processor. The security method can be applied to other sensitive data processing for the volunteer computing. We have evaluated the performance of the secure K-Means clustering on the Cell secure system simulator. The performance statistics has been compared with the corresponding statistics for the non-secure K-Means clustering on the same simulator. Results demonstrated a huge performance overhead due to the data decryption. We presented a possible solution to address this performance issue.

Because of the PlayStation3, millions of Cell processors will be connected to the Internet. The high performance, the low power consumption, the large number of peers in the future, and the security features make Cell a promising platform for future secure volunteer data mining systems.

While the security feature leads to huge performance overhead on the Cell processor, a specific optimization is required. For the future work, we will design and evaluate a data summarizing method to reduce the computation cost for data decryption. A general secure volunteer data processing framework will also be designed on top of the Cell security SDK and the BOINC platform.

ACKNOWLEDGMENT

The authors would like to thank IBM Japan for providing Cell security SDK and related documents for evaluation. This research was partially supported by Grant-in-Aid for Scientific Research on Priority Areas #18049003 and Strategic Information and Communications R&D Promotion Program (SCOPE-S) #061102002.

REFERENCES