ISSN 2349-4506
Impact Factor: 3.799

Global Journal of Engineering Science and Research Management

RECOVERING MS OFFICE PASSWORD ON PARALLEL COMPUTING SYSTEMS WITH MULTI-CORE PROCESSORS

Nguyen Minh Quy*, Nguyen Dinh Han, Do Anh Tuan, Nguyen Van Hau, Pham Quoc Hung, Nguyen Xuan Truong, Nguyen Van Quyet, Ho Khanh Lam
* Information Technology Faculty, Hung Yen University of Technology and Education, Vietnam

DOI: 10.5281/zenodo.3345111

KEYWORDS: Parallel processing, Password recovery, Performance analysis, Multi-cores programming.

ABSTRACT

Recovering MS Office password using the Brute-Force method with multi-core CPUs and GPU support has been studied in several works, and obtained some success. In these works, the combination of multi-core CPUs and computer clusters in the computing process is essential in order to increase the processing speed. In this paper, we propose a method for recovering Microsoft Office passwords by an exhaustive method based on a parallel computing system with multiple compute nodes and also take advantage of multi-core capabilities in each node. Moreover, this paper presents an assessment of the impact of communication delays on the performance of a parallel computing system.

INTRODUCTION

Nowadays, using parallel computing systems to solve the problems with high complexity and substantial computation cost have been attracted by many studies. These systems often use computational clusters with hundreds or even thousands of compute nodes for processing data, and each node has multiple CPU and GPU cores) involved in the computation process [1,2]. Analyzing the performance of parallel computing systems is a complex problem. Gropp et al. [3] divided the execution time of a parallel computing system into three parts: calculation time, communication time, and idle time while handling large datasets. Thomas Rauber and Gudula Rünger indicated that the advances of multi-core processors and cache technology could help to build parallel computing systems more efficiently [1]. Several approaches for analyzing the performance of parallel algorithms have been presented in [5] and [6]. Xian-He Sun et al. [7] proposed an approach based on the study by Amdahl in [5] for analyzing the performance analysis of multi-core systems. In this paper, we proposed a method of analyzing performance for parallel computing systems by improving the approach developed by Xian-He Sun et al. in [7]. We conduct experiments on a real multi-core system with two compute nodes for solving the password recovering problem.

Recovering MS Office password is a particular problem which can be applied a parallel computing mechanism to improve the performance of a password recovery system. There have been several studies focused on solving this problem on multiple CPU cores and GPU cores [8, 9, and 10]. In this paper, we proposed an approach for recovering password of MS Office 2010 by combining multi-core CPUs in a compute node and multiple compute nodes in a cluster with unlimited scalability to create a two-level parallel computing system. The system does not require to compute nodes to have the same hardware settings. It can be a network of personal computers and servers which are connected to each other via a Wifi or LAN/WAN network. Besides, we also discuss the problem of communication delay, which affects the performance of the system. We found that the acceleration level of a parallel computing system is not always completely linear with the number of nodes.

The rest of the paper is organized as follows. Section 2 presents our approach for analyzing the performance of a parallel computing system. Section 3 presents the recovering MS Office password problem, a design and implementation of a parallel computing system for solving password recovering problem. In Section 4, we evaluate our approach with the analysis of the experimental results and discussions. Finally, we conclude the paper in Section 5.

PERFORMANCE ANALYSIS OF PARALLEL COMPUTING SYSTEM

Analyzing the performance of parallel computing systems or parallel algorithms can be calculated by using Equation 1 as the following:

$$Sp = \frac{T_s}{T_p} \qquad (1)$$

Where T_s is the execution time of performing S programs sequentially; T_p is the execution time of performing S programs in a parallel manner using *p* processors [6]. By applying the method proposed by Amdahl in [1,6], we have:

$$Sp = \frac{l}{f+\frac{1-f}{p}} \qquad (2)$$

Where *f* is the ratio of the components which are performed in a sequential manner, (1-*f*) indicates the ratio of the components which are performed in a parallel manner with *p* processors. Based on the studies in John L. Gustafson [6] and Xian-He Sun [7], we propose an equation for calculating the execution time as the following:

$$T_p = t_s + t_p + t_c \qquad (3)$$

Where t_s is the execution time of performing components sequentially; t_p is the execution time of performing components in a parallel manner; t_c is the delay time of communication.

From Equation 1 and Equation 2, we have:

$$T_s = t_s + p \times t_p \quad (4)$$

So, from Equation 1, 3, and 4, we can calculate S_p by using Equation 5 as follows.

$$Sp = \frac{t_s+p\times t_p}{t_s+t_p+t_c} \qquad (5)$$

RECOVERING PASSWORD OF MICROSOFT OFFICE

Password in MS Office

Password in applications of Microsoft Office, such as MS Word or MS Excel, is used to secure the documents generated by them [15]. However, the fact is that users sometimes forget their documents' password, and they request to recover them to use the original documents. In practice, Microsoft uses the same encryption mechanism for all applications like Microsoft Word and Excel [11]. Therefore, in this paper, we focus on solving the problem of recovering password for MS Word 2010, while other applications in MS Office can be applied similarly.

Password protect of MS Office documents can be used one of four different mechanisms [11]. In which, MS Office version 2000 and 2003 used RC4 cryptography 40 bit by default [12], but MS Office version 2007 and 2010 have been used Advanced Encryption Standard (AES) with the Secure Hash Algorithm 1 (SHA-1) [13]. The speed of key generation in the AES cryptography is much slower RC4 cryptography [10,14] as performing many loops in SHA algorithm.

Recovering password in MS Office 2010

By using AES cryptography with key size 128, 192, 256 bit [12,13], it seems impossible to decrypt a password in MS Office 2010. Therefore, we use a *Brute Force* method to find out the password for the original documents. With this method, we can take advantage of the parallel computing power of multi-core CPUs and a cluster system. The data structure for storing information related to document protection of MS Office had described by Microsft in [15]. It could be extracted to calculate and generate a key with a given input string. This step can be done by using an open source in [16]. Here, we propose a method for recovering the password by using a parallel computing system with the process, as shown in Figure 1.

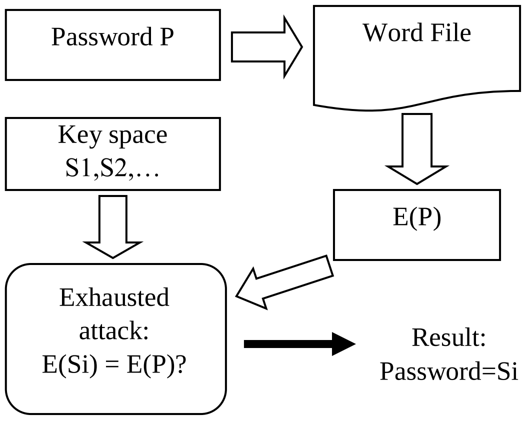

Figure 1. The procedure of recovering password in MS Word

In Figure 1, users first set a password P for a given document. Next, MS Word performs encryption the password, and we have a hash of the password, called E(P). The password decoding from E(P) is hard, and it seems impossible.

The process of using the Brute Force method on a cluster for parallel processing with multi-core CPUs is illustrated in Figure 2.

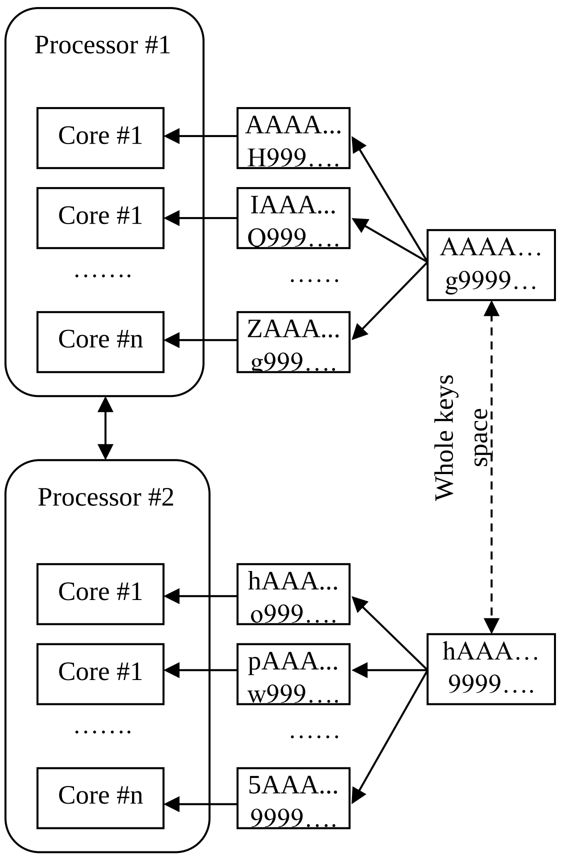

Figure 2. Parallel processing using multi-core CPUs and a cluster with 2 compute nodes

In practice, a password often consists of lowercase characters, uppercase characters, numbers, and special characters. However, According to the habits of most users, only uppercase and lowercase letters are used (52 characters), and it can also contain numbers (10 characters) sometimes. Therefore, the key-space of a given string with length k can be $62^k+62^{k-1}+\ldots+62^1$.

Table 1. The key-space with various password length

Password with k length	Key-space
1	62
2	3.906
3	242.234
4	15.018.570
5	931.151.402
6	57.731.386.986

Thus, using a single machine for checking all possible of the key-spaces above and comparing with E(P) might only solve with small *k* such as with the length of 5 or 6. However, by taking advance of multi-core CPUs and the support of GPU on a distributed system with multiple connected machines, the speed of finding key could be reduced hundred times or even a thousand times depending on the size of the cluster. In that case, the system can find the password with the length k being equal to 9 or 10 characters.

Recovering password using a parallel computing system with multi-core CPUs

The basic idea of our approach is to use multi-core CPUs (e.g., 4 cores) and perform finding the password on a cluster in a parallel manner. To do this, our system performs two main jobs:

1) *Assigning tasks for each computer in the cluster*
2) *Assigning tasks for each core of every machine in the cluster*

For the first job, we implement a program, called *Master*, running on a machine in the cluster calculate and assign tasks to other machines, called *Slave*s, in the cluster. The *Master* communicates with *Slaves* through UDP (User Data Protocol) protocol.

For the second job, once *Slave(s)* is assigned the main task with a given key-space, then it separates the main task into multiple small tasks, each small task is assigned to a CPU core to find out the original password. Deploying multiple tasks on multi-core CPUs in a machine can be easily implemented by using multi-threading programming technique, each thread is assigned an individual task by the operating system, as shown in Figure 2.

During the process of comparing strings to find out the original password, *Slaves* often send meta information (e.g., the number of passwords have been checked) to *Master* via UDP protocol in order to show that to the users. This process will stop if a thread found the original password or there is no password could found by all threads after finishing the Brute Force algorithm with a given key-space.

EVALUATION AND DISCUSSION

Evaluation Settings

In the password recovering problem above, *Slaves* often send packets to *Master* and threads in each slave has an interrupt time due to updating the information from global variables. These cause of a substantial communication delay (t_c), which depends on the frequency of sending packets from *Slaves* to *Master.*

To evaluate the acceleration of using a parallel computing system with multiple compute nodes, we use two compute nodes (*Slaves*) and another node works as a *Master*, where each *Slave* uses CPU Intel Core i3 4x3.3GHz with 2GB RAM and the *Master* uses CPU Intel Core M380 Core i3 4x2.53GHz with 8GB RAM. The nodes in our cluster are connected via a LAN network with bandwidth 100Mps. The length of a packet from *Slaves* to *Master* are varied in order to evaluate the communication delay cost. Besides, the frequency of transferring packets also can be adjusted by using a timer.

Experimental Scenarios
In our experiments, we perform a recovering password with a length of 4 characters.

Experiment 1:
- Number of compute node: 1
- Number of cores/node: [1,4]

Experiment 2:
- Number of compute node: 2
- Number of cores/node: [1,4]
- Length of the packet from *Slaves* to *Master*: ~10 bytes.
- Interval of sending the packet from *Slaves* to *Master*: 1000ms.

Experiment 3: Increasing the length of the packet from *Slaves* to *Master* with 100 times compared to the one in Experiment 2.
- Length of the packet from *Slaves* to *Master*: 1000 bytes.
- Other settings are the same as Experiment 2

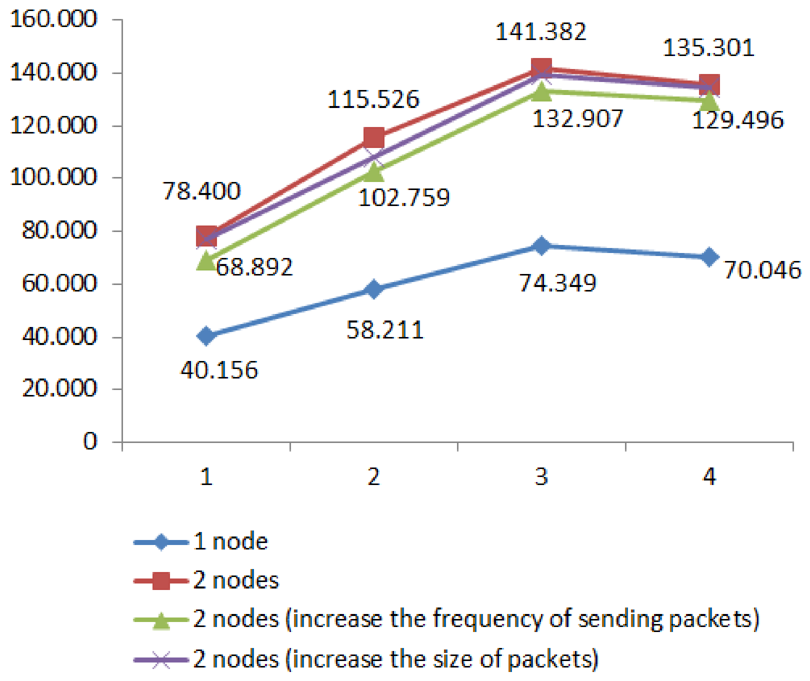

Figure 3. Experimental results

Experiment 4: Increasing the frequency of sending the packet from *Slaves* to *Master* with 20 times faster compared with the one in Experiment 2.
- Interval of sending the packet from *Slaves* to *Master*: 50ms.
- Other settings are the same as Experiment 2

Analysis and Discussion
The result in Experiment 1 indicated that the computational speed increases if we increase the number of cores of the machine, which is used to recover the password. However, this acceleration tends to slow down. It is reasonable because as the number of cores increases, the communication delay (tc) between these cores is also

increased internally due to the need for additional processing such as synchronization, resource allocation, and operating system processing.

In Experiment 2, the speed also increases with the number of nodes (Slave) involved in the calculation when we employ parallel computing on a cluster. However, if we consider the total number of cores, the performance, in this case, will be less than the one in Experiment 1 because the communication delay of the network is much larger than the communication delay between the cores in the same CPU.

In Experiment 3, if the packet size is increased by 100 times compared to Experiment 2, the performance of the system decreases. This proves that packet size also increases communication latency and consequently reduces performance. This result is also consistent with Equation 5 in Section 2.

In Experiment 4, if the packet size is set the same as in Experiment 2, but increases the frequency of sending the packet from Slave to Master (to increase the traffic), the performance will also be reduced. This demonstrates that the communication delay has a significant impact on the performance of the system, especially when data exchange is via low-speed transmission.

In short, from the results shown in Figure 3, we found that when the number of processing cores increases, the performance of the entire system will increase linearly, but when we use all the number of cores of the CPUs, the performance will decrease.

CONCLUSION

Recovering passwords in MS Word, as well as some other systems with passwords which are not too long and complex, can be accomplished by a Brute Force method using a parallel computing system with multiple compute nodes and multi-core CPUs on each node. In fact, we can adjust the exhausting process by some smart mechanisms, such as a heuristic algorithm or dictionary, to minimize computation time. Through extensive experiments in the paper, we found that the performance of parallel computing systems depends on some factors including the number of nodes, the number of CPU cores, and the communication delay especially when the traffic increases. In order to increase the performance of the system by minimizing communication delay, it is possible to reduce traffic between processes and between compute nodes, such as minimize packet size or frequency of sending packets.

Another finding is that when selecting the number of CPU cores involved in parallel computing, it is essential to select the appropriate number of cores to achieve the highest level of performance, choosing all the cores to participate in the calculation could not have the best performance.

REFERENCES

1. Thomas Rauber and Gudula Rünger. “Parallel Programming: for Multicore and Cluster Systems”. Springer, 2013, ISBN 978-3-642-37801-0. 516 pages.
2. Hwang K. and Xu Z. “Scalable Parallel Computing: Technology, Architecture, and Programming”. McGraw-Hill, NY, 1998, ISBN-10: 0070317984. 832 pages.
3. Gropp, W. et al. “The Sourcebook of Parallel Computing. The Morgan Kaufmann Series in Computer Architecture and Design”, 2002, ISBN-13: 978-1558608719. 842 pages.
4. Michael J. Quinn. “Parallel programming in C with MPI and OpenMP”. McGraw - Hill, 2004, ISBN-10: 0072822562. 544 pages.
5. Amdahl, G.M. “Validity of the single-processor approach to achieving large scale computing capabilities”. Iin: Proc. Am. Federation of Information Processing Societies Conf., AFIPS Press, 1967, pp. 483-485.
6. Gustafson J.L. “Reevaluating Amdahl’s Law. Communications of the ACM”, Volume 31 Issue 5, 1988, Pages 532-53.
7. Xian-He Sun, Yong Chen. “Reevaluating Amdahl's law in the multicore era. J. Parallel Distrib. Comput”. 70 (2010) 183-188.

8. Phan Duc Dung et al. "Application of CUDA in recovering zip file's password". Proceedings of the student conference of scientific research, Ha Noi University of Sicence and Technology 2009-2010, ISBN: 978-604-911-001-6.
9. Apostal, D, "Password recovery using MPI and CUDA", 19th International Conference on High Performance Computing, ISBN 978-1-4673-2370-3, 2012.
10. Hu, G., Ma, J., & Huang, B. (2009, December). "Password recovery for RAR files using CUDA. In Dependable, Autonomic and Secure Computing", 2009. DASC'09. Eighth IEEE International Conference on (pp. 486-490).
11. http://msdn.microsoft.com/en-us/library/dd910529(v=office.12).aspx
12. http://msdn.microsoft.com/en-us/library/dd922354(v=office.12).aspx
13. Apostal, D., Foerster, K., Chatterjee, A., & Desell, T. (2012, December). "Password recovery using MPI and CUDA. In High Performance Computing (HiPC)", 2012 19th International Conference on (pp. 1-9). IEEE.
14. Allam Mousa and Ahmad Hamad, "Evaluation of the RC4 Algorithm for Data Encryption", International Journal of Computer Science and Applications, Vol. 3, No. 2, June 2006.
15. http://msdn.microsoft.com/en-us/library/office/cc313071(v=office.12).asp
16. http://offcrypto.codeplex.com/