2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems

A Parallel Crime Activity Clustering Algorithm based on Apache Spark Cloud Computing Platform

Khin Nandar Win[†,‡], Jianguo Chen[†,‡,*], Guoqing Xiao[†,‡], Yuedan Chen[†,‡], and Philippe Fournier Viger[¶]
[†]*College of Computer Science and Electronic Engineering, Hunan University, Changsha, Hunan 410082, China.*
[‡]*National Supercomputing Center in Changsha, Changsha, Hunan 410082, China.*
[¶]*School of Computer Science, Harbin Institute of Technology, Shenzhen, China.*
[*]*Jianguo Chen is the corresponding author.*
E-mail: {knandarwin, jianguochen, xiaoguoqing, chenyuedan}@hnu.edu.cn, philfv8@yahoo.com

***Abstract*—Nowadays, there are more and more criminal behaviors experiencing around the world, and crime spiking has become one of the most critical security and social issues in almost every country. It is critical to seek effective ways to discover these criminal behaviors and patterns and to carry out the prevention for the target place. In this paper, we formulate the problem of criminal behaviors and propose a Criminal Activity Clustering (CAC) algorithm. We introduce the fuzzy clustering method to detect potential criminal patterns in large-scale spatiotemporal datasets. In addition, for the improvement of the the proposed CAC algorithm performance, we implement a parallel solution for the algorithm in the Apache Spark cloud computing platform. The results of the experiment show that the proposed CAC algorithm can effectively detect accurate criminal patterns from large-scale spatiotemporal data.**

***Index Terms*—Apache Spark, criminal activity clustering, fuzzy C-Means, parallel computing, distributed computing.**

I. Introduction

In the past 20 years, different types of criminal activities have occurred more and more frequently in every corner of the world [1]. It is critical to analyze the characteristics and patterns of these criminal activities and to identify potential patterns and relationships between them [2]. Previous studies on criminal pattern mining have played a positive role in analyzing the causes and purposes of criminal actions, which helps us to effectively detect and control these crimes. In the field of criminal behavior analysis, positioning technology is usually used to locate criminal suspects and crime sites, criminal tracking and so on. In this work, we focus on the discovering of criminal patterns and the identification of the hotspot areas.

Together with the development and improvement of computer hardware, software and information technology, more and more criminal records have been recorded in time and in detail, providing rich data resources for academic and industrial crime analysis [3–6]. Various crime behavior analysis methods are proposed, including crime behavior prediction, criminal activities classification, clustering, criminal relationship network mining, etc [7, 8]. In addition, various parallel computing and distributed computing technologies are proposed to the field of data mining and crime data analysis [9–12]. Although different methods have already been proposed for criminal pattern discovering, there are not rich in attributes in their implementation in terms of attack type, and target type. They are lack of multiple attributes to find out more meaningful criminal patterns.

In this work, we focus on the criminal pattern discovering and the hotspot areas identification. We analyze the Global Terrorism DataBase (GTD) and propose a criminal activity clustering algorithm to effectively detect criminal activities. According to our experiment, the results show that our proposed algorithm can effectively detect accurate criminal patterns from large-scale spatiotemporal data. The contributions of our work are summarized as follows.

- We analyze the features of large-scale GTD dataset, formulate the problem of criminal behaviors.
- We propose a Criminal Activity Clustering (CAC) algorithm based on fuzzy clustering to detect potential criminal patterns in large-scale spatiotemporal datasets.
- We implement the CAC algorithm with parallel solution using Apache Spark for the improvement of the algorithm performance.

The remaining part of our paper is organized as follows: In section II, we discuss related work of criminal pattern discovering, clustering algorithms and the limitation of existing work. We describe the Criminal Activity Clustering (CAC) algorithm in Section III. We provide the parallel implementation for the proposed CAC system in section IV. And then, we conclude the paper in section V, and acknowledgment in section VI.

II. Related Work

The crime analysis system uses the relative important attributes to help forensic investigators determine the most dominant criminal groups members for the purpose of investigation [13, 14]. Kaza *et al.* examined the use of criminal activity networks (CANs) for analyzing the information which are from law enforcement and other sources to provide the profit of border security and transportation[13]. In [14], Vural *et al.* introduced a solution to the criminal prediction problem using Naive Bayes theory. With the development of information processing technology and satellite positioning technology, more and more spatiotemporal datasets are collected in criminal incidents [3]. Rashidi *et al.* identified elephant poaching hotspots by analyzing the differences in the poached elephants clusters in the Tsavo ecosystem [6]. Xue *et al.* used crime data from Campinas, Brazil for the investigation of the crime consolidation and the correlations among different crime types throughout the location [4].

Focusing on clustering technology on crime, various efforts were contributed using different methods [15]. Fuzzy clustering is one of the most very attractive continuing research area in the field of not only computer science, mathematics and other areas of engineering, but also all areas of optimization practices [8]. To predict computer crimes, Brown *et al.* proposed a method for discovering the preferences of computer criminals [15]. In [16, 17], Xiao *et al.* introduced a probabilistic reverse Top-k Query algorithm to report most significant objects in uncertain databases. In [18], Son *et al.* reviewed the fuzzy C-Means for big data, and summarized some novel hybrid clustering algorithms that are based on the initial selection and incremental clustering in the field of crime analysis.

To efficiently analyze and mine criminal behaviors and patterns from large-scale crime data, scientists and engineers have tried to apply advanced computing capabilities to the field of data mining and crime data analysis [10–12]. In [10], Wang *et al.* proposed the parallel crime scene analysis system depending on the computational experiments, artificial societies and parallel execution (ACP) approach. Focusing on supercomputer, Chen *et al.* designed a performance-aware model for sparse Matrix-Matrix multiplication on the Sunway TaihuLight Supercomputer [19–21]. In addition, to optimize the execution performance of parallel data mining tasks, Li *et al* proposed a scheduling precedence constrained strategy for stochastic tasks on heterogeneous cluster systems [22].

III. Criminal Activity Clustering Algorithm

A. Criminal Activity Dataset

In this work, we collect the crime activities dataset from the Global Terrorism DataBase (GTD) [23]. The GTD is the database that can be available publicly as an open-source and it includes the information of terrorist events throughout the world from the period of 1970 to 2017. The difference between the GTD and many other databases is that it consists of systematic data both domestic and international terrorist incidents which have occurred within this period. And at present, there are more than 180,000 cases in the database.

There are more than 182,438 records with 105 features in the collected GTD dataset, containing a wealth of criminal information. For example, from the features of incident location, we can obtain information about the country, region, city, longitude, and latitude of each crime activity, providing accurate data sources for clustering analysis. From the features of perpetrator information, we can get the names of the various perpetrator groups, the number of perpetrators involved, and the motives of each crime activity, and can further extract the social relationship among these perpetrator groups. The detailed features description of GTD is provided in Table I.

Based on the collected dataset, we conduct a holistic statistical analysis of annual crime records in the range of 1970-2017 in approximately 228 countries/locations where the incident occurred. The geopolitical boundaries of many countries have changed over time. For example, the attack in Bonn is recorded in West Germany(FRG) as taking place [23]. In addition, we select the top 4 countries with the highest number of crimes as the analysis project. The selected countries are Iraq, Pakistan, Afghanistan, and India, respectively. The distribution of the number of criminal activities in the top 4 countries is shown in Fig. 1.

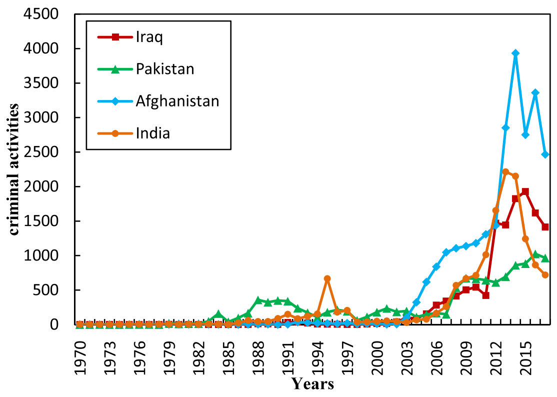


Fig. 1. Distribution of the number of crimes in the top 4 countries.

As it can be seen from Fig. 1, before 2003, the number of crimes in each country was at a very low level. This may be because of the lack of the information technology and it leads to the incomplete records. However, since 2003, the number of crimes in each country has increased rapidly every year, with Afghanistan being the most visible. For example, from 2003 to 2017, the number of crimes in Afghanistan rise rapidly from 102 to 2,466 per year, additionally, in Iraq from 100 to 1414, and crimes rate in Pakistan from 196 to 966, while the number of crimes in India increases from 29 to 719.

TABLE I
Feature description of the GTD dataset.

No.	Category	Features
1	Date	Year, month, day.
2	Location	Country, city, latitude, longitude.
3	Incident	Inclusion criteria, related incidents.
4	Attack	Attack type, successful attack, suicide attack.
5	Target	Target type, name of entity.
6	Perpetrator	Perpetrator group name, number of perpetrators, motive.
7	Weapon	Weapon type.
8	Consequences	Fatalities total number, Injuries total number.

In this paper, we intend to use data mining and machine learning technology to analyze the crime datasets in the spatiotemporal format to discover potential crime patterns and make an effective prediction. To achieve the desired goals, the main challenges we encountered and the technical issues that need to be addressed are summarized as follow.

(1) Large-scale spatiotemporal datasets have the characteristics of large data volume, high-dimensionality, and noisy data. It is critical to design effective data mining algorithms to accurately detect various types of criminal behaviors from these datasets. And it is important to effectively classify, cluster behaviors and identify the potential crime patterns.

(2) According to the criminal behavioral classification/clustering results, it is needed to evaluate the crime rate of various crimes and predict crime hotspots from temporal and spatial dimensions, respectively. In addition, we provide reliable data analysis results of criminal activities that can be used in governments and social publicly.

(3) In a large-scale crime data environment, it is an essential issue to use high-performance and cloud computing resources for the performance improvement of the crime data analysis algorithm and provide timely results.

B. *Criminal Activity Clustering (CAC) Algorithm*

We propose a Criminal Activity Clustering (CAC) algorithm based on fuzzy clustering to detect potential criminal patterns in large-scale spatiotemporal datasets. Firstly, we introduce Fuzzy C-Means clustering algorithm to cluster criminal activities from the crime type, spatial, and temporal aspects, respectively.

We select N records with M important features from the GTD dataset as the training dataset X, namely, $X = \{x_1, ..., x_N\}$. The M features of the training dataset are selected from Table I, such as GTD ID and date (i.e., year, month, and day), incident location (i.e., country, region, province/state, city, vicinity, latitude, and longitude), and attack information (i.e., attack type, successful attack, and suicide attack).

We assume that there are C potential clusters in the training dataset X, and each record $x_i (i = 1, 2, ..., N)$ is termed as a data point (sample) in the clustering process. The FCM algorithm divides the N samples $x_i (i = 1, 2, ..., N)$ into C fuzzy groups, and finds the cluster center of each group. Therefore, the value function of the non-similarity index is minimized. The main difference between FCM algorithm and traditional clustering algorithms is that fuzzy partitioning is used in FCM. In this case, every given data point uses a membership value of $[0, 1]$ for determining the extent to which it belongs to each group. In correspondence with the introduction of fuzzy partitioning, the membership matrix U allows elements u_{ij} with values between 0 and 1. Note that after adding the normalization rule, the sum of the memberships of a training dataset is equal to 1, as calculated in Eq. (1):

$$\sum_{i=1}^{C} u_{i,j} = 1, \forall j = 1, ..., N. \tag{1}$$

In this way, we initialize the membership matrix U with a value between 0 and 1 randomly, so that it satisfies the constraints in Eq. (1). Based on the membership matrix, we need to calculate the potential C crime activities clusters. And then, it is easy to generalize the value function (or objective function) of FCM in the form of Eq. (2):

$$J(U, C) = \sum_{i=1}^{C} J_i = \sum_{i=1}^{C} \sum_{j=1}^{N} u_{ij}^{\gamma} d_{ij}^2, \tag{2}$$

where u_{ij} is in the range of 0 and 1, c_i in C is the cluster center of fuzzy group I, d_{ij} is the Euclidean distance between the i-th cluster center and the j-th data point, and γ is the weighted index. Then, we construct a new objective function to match the necessary condition of Eq. (2) and get the minimum value. The new objective function is defined as:

$$\begin{aligned} \overline{J}(U, C, \lambda) &= J(U, C) + \sum_{j=1}^{N} \left(\sum_{i=1}^{C} u_{ij} - 1 \right) \\ &= \sum_{i=1}^{C} \sum_{j=1}^{N} u_{ij}^{\gamma} d_{ij}^2 + \sum_{j=1}^{N} \left(\sum_{i=1}^{C} u_{ij} - 1 \right), \end{aligned} \tag{3}$$

where $\lambda_j (1 \leq j \leq N)$ is the N constrained Lagrangian multipliers in Eq. (1). The necessary conditions for deriving all input parameters to minimize Eq. (2) are defined as:

$$c_j = \frac{\sum_{j=1}^{N} u_{ij}^{\gamma} x_j}{\sum_{j=1}^{N} u_{ij}^{\gamma}}, \tag{4}$$

and

$$u_{ij} = \frac{1}{\sum_{i'=1}^{C} \left(\frac{d_{ij}}{d_{i'j}} \right)^{2/(\gamma - 1)}}. \tag{5}$$

Based on the initialized membership matrix U and candidate cluster centers C, we calculate the value function according to Eq. (2). If the value of $J(U, C)$ is less than a certain threshold θ, or if its change from the last value function value is less than a certain threshold ϑ, then the criminal activity clustering process stops. In addition, we continue to update the membership matrix U using Eq. (5) and additionally update the crime cluster centers C.

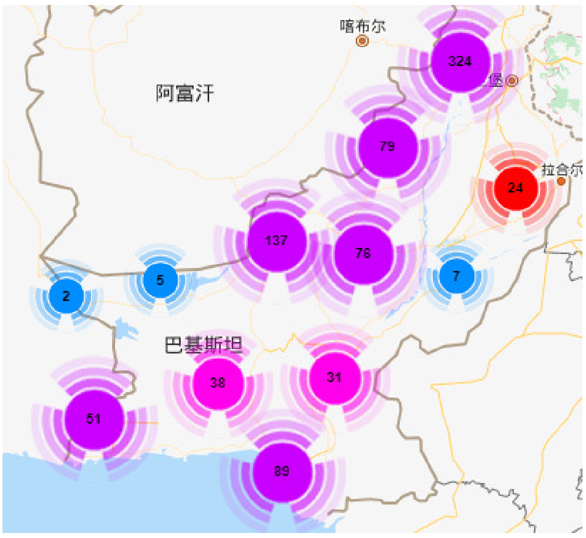
The CAC algorithm detailed steps are presented in Algorithm III.1.

IV. Parallel Implementation of CAC System

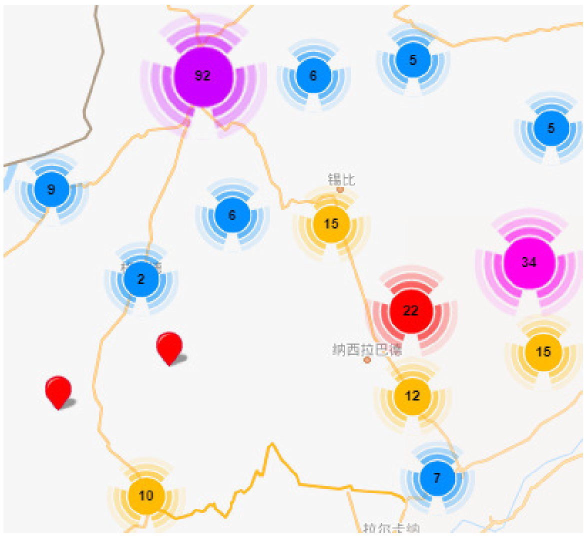
A. *Parallel Computing Architecture*

Apache Spark platform is a great open-source project that provides cloud computing services, and the founders of this project come from developers, organizations, and individuals around the world [24]. Spark is an efficient cloud computing platform that ideal for large-scale data processing and distributed and parallel data analysis. The main components of the Spark platform include Yarn, Tachyon, HDFS, Spark Core, machine learning library (Mllib), graph processing library (GraphX), SQL database access library (SQL Shark), and streaming computing module (DStream). The technical architecture of Spark is shown in Fig. 3.

Apache Spark platform supports various distributed computing, parallel computing, and iterative computer mechanisms, including powerful memory storage, flexible cloud file storage,



(a) Clustering in country-level



(b) Clustering in city-level

Fig. 2. Criminal activities clustering in multi-level spatial dimension.

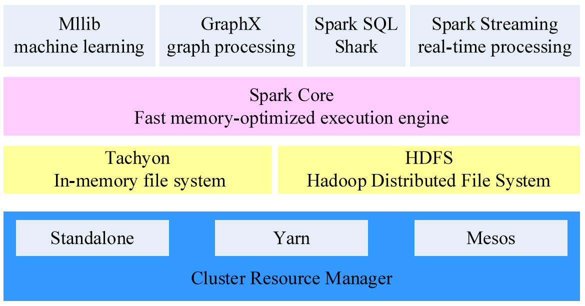


Fig. 3. Technical architecture of the Apache Spark platform.

Algorithm III.1 CAC algorithm.

```
Input:
    $X$: The crime activity dataset;
    $C$: Number of crime activity clusters;
    $\gamma$: The weighted index;
    $U$: The initialized membership matrix;
    $\vartheta$: The threshold of the difference for the target functions
    between two iterations.
Output:
    $U$: The membership matrix;
    $CAC$: The crime activity clusters.
 1: get the shape of dataset $N, M \leftarrow shape(X)$;
 2: initialize the membership matrix $U \leftarrow$ initWithFuzzyMat$(N, C)$;
 3: calculate initialized cluster centers $CAC \leftarrow$ calCentWithFuzzyMat$(X, U, \gamma)$;
 4: create a list of target function value $tfs \leftarrow [\,]$;
 5: while True do
 6:     tf1 $\leftarrow$ calTargetFunc$(X, U, CAC, C, \gamma)$;
 7:     if this is the first round then
 8:         append $tf1$ to $tfs$;
 9:         update membership matrix $U \leftarrow$ calFuzzyMatWithCent$(X, CAC, \gamma)$;
10:         update cluster centers $CAC \leftarrow$ calCentWithFuzzyMat$(X, U, \gamma)$;
11:     else if $|(tfs[len(tfs) - 1] - tf1| > \vartheta$ then
12:         append $tf1$ to $tfs$;
13:         update membership matrix $U \leftarrow$ calFuzzyMatWithCent$(X, CAC, \gamma)$;
14:     else
15:         break;
16:     end if
17: end while
18: return $U, CAC$.
```

efficient fault recovery, and real-time streaming processing. The core programming model of Spark is the Resilient Distributed Datasets (RDDs), which represent a collection of distributed datasets, and allows different distributed computers to perform parallel access operations on the same RDD object. Apache Spark platform provides a variety of programming interfaces, from which users and programmers can easily create RDD objects and perform various operations on these RDD objects.

B. Executing CAC Algorithm on Apache Spark

In this work, we construct an Apache Spark cloud cluster at the National Supercomputing Center in Changsha [25]. The CAC algorithm with the parallel module of CAC is deployed on the Apache Spark cluster. The Spark cluster is constructed with one driver computer, one system master computer, and five worker computers. The physical execution process of the CAC algorithm on the Spark cluster is illustrated in Fig. 4.

We firstly need to start-up the Spark cluster and initialize the execution environments and related operating parameters.

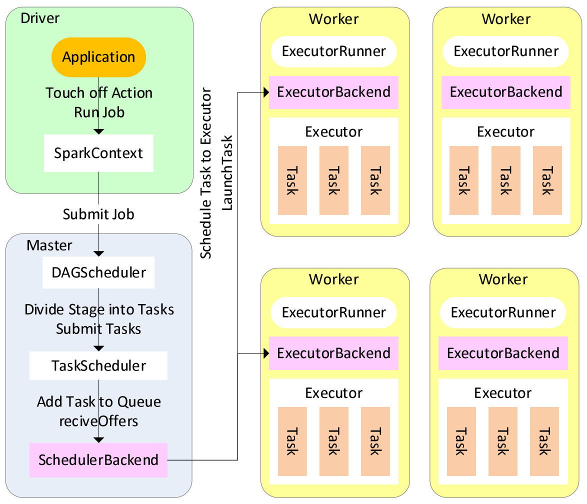


Fig. 4. Physical execution process of CAC algorithm on Spark.

The program of the CAC algorithm is submitted from the development environment to the Spark driver computer. After receiving the program of the CAC algorithm, the driver computer calls the RDD data management component and parses the tasks in the program code. Then, the RDD data management component analyzes the logical dependencies and data dependencies between the tasks, and generates corresponding RDD objects. When computing tasks that contains an *action* function is triggered, the Spark platform will generate a new execution job with a new Spark context. The job will be submitted to the Spark master computer immediately. The detail steps of the parallel CAC algorithm are described in Algorithm IV.1.

Algorithm IV.1 Parallel process of the CAC algorithm.

```
Input:
    $Path_X$: the path of crime activity dataset $X$;
    $C$: Number of crime activity clusters;
    $\gamma$: The weighted index;
    $U$: The initialized membership matrix.
Output:
    $RDD_C$: the RDD object of crime activity clusters.
1: $conf \leftarrow$ new SparkConf("CAC", "SparkMaster");
2: $sc \leftarrow$ new SparkContext($conf$);
3: $RDD_X \leftarrow sc$.textFile($Path_X$);
4: $RDD_C \leftarrow sc$.parallelize($RDD_X$).map
5:     $U \leftarrow$ calculate membership matrix ($RDD_X$);
6:     $CAC$ update cluster centers ($RDD_X, U, \gamma, \vartheta$);
7:     $RDD_C$ calculate crime clusters;
8: $RDD_C$.reduceBykey();
9: return $RDD_C$.
```

On the master computer, a DAG scheduling component is activated to generate a task DAG graph for the newly arrived execution job. In this step, the component divides the entire job into multiple stages based on the dependencies of tasks in the DAG graph. Further, it divides the tasks in each phase, and marks parallel tasks in parallel with non-parallel. All computing tasks are then submitted to a task scheduler, which is responsible for assigning these tasks to different distributed worker nodes in the Spark cluster.

Each worker node is configured with high-performance computing units such as multiple threads, multi-core CPU, or GPUs that can efficiently work in parallel. There are one or more ExecutorBackend (EB) processes on each worker node, each of them contains an executor object, and the executor object has a thread pool, and each thread starts a task. In this way, the computing tasks of the CAC algorithm are executed in parallel in different working nodes, thereby improving the performance of the entire CAC algorithm. Benefit from the Spark cloud computing platform and the designed parallel algorithms, the CAC algorithm can efficiently process large-scale crime datasets and output the analysis results in real time.

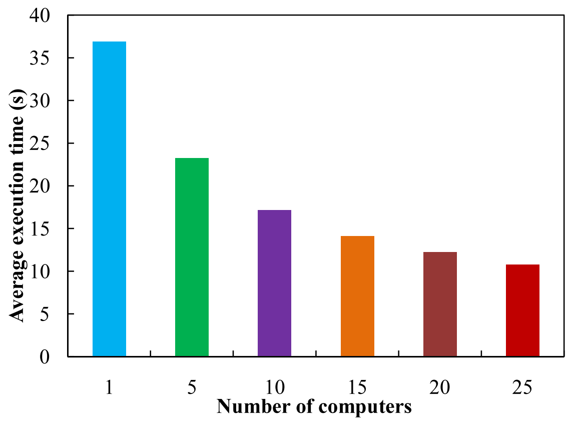
C. Performance Evaluation of Parallel Algorithms

Experiments are conducted for the evaluation of the proposed CAC algorithm performance. Two groups of crime datasets with different number of samples are used in the experiments. In these experiments, the number of computers is set in the range of 1 to 25 for the Spark cloud environment. For each case, the execution time of the parallel CAC process is recorded and make comparison for each records. The experimental results are presented in Fig. 5.

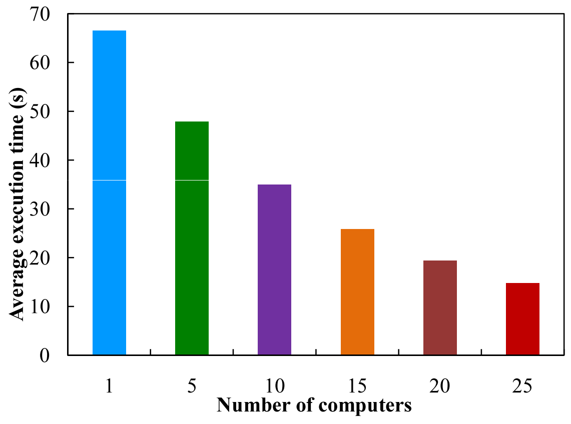
We can see clearly from Fig. 5 that the average execution time of the entire CAC algorithm significantly reduces in accordance with the increase of computing nodes. For example, using 900,000 samples, 36.87(s) is required if we run CAC algorithm as the serial version on the single machine. In contrast, the parallel version of the CAC algorithm uses less time in the Spark cluster. The execution time of CAC is 23.22(s) using 5 nodes, 17.18(s) using 10 nodes, 14.09(s) using 15 nodes, 12.26(s) using 20 nodes, and 10.79(s) using 25 nodes. As shown in Fig. 5(b), when the scale of samples is up to 180,000, the performance of the CAC algorithm is even more significant. For example, 66.54(s) is required for the serial version of the CAC algorithm runs on a single machine. In contrast, the execution time of parallel CAC is 47.90(s) using 5 nodes, 34.97(s) using 10 nodes, 25.88(s) using 15 nodes, 19.41(s) using 20 nodes, and 14.75(s) using 25 nodes. With taking the advantage of parallel optimization task, the CAC algorithm achieves significant strengths over the serial version in terms of performance.

V. Conclusions

This paper proposed a Parallel Crime Activity Clustering algorithm using fuzzy clustering algorithm to discover criminal behaviors. We introduced the FCM clustering algorithm to effectively detect potential criminal patterns from large-scale criminal activity datasets in a spatiotemporal format. We further focused on the proposed CAC algorithm performance



(a) Samples = 900,000



(b) Samples = 1,800,000

Fig. 5. Average execution time of CAC algorithm.

and implemented a parallel resolution of CAC algorithm on the Apache Spark cloud computing platform, which can significantly accelerate the process of crime data mining. Extensive experiments show that the results of the proposed CAC algorithm can efficiently detect accurate criminal patterns.

Acknowledgment

The authors would like to thank the anonymous reviewers for their valuable and helpful comments on improving the manuscript. The research was partially supported by the Program of National Natural Science Foundation of China (Grant Nos. 61806077, 61661146006, 61860206011), the Program of Hunan Provincial Innovation Foundation for Postgraduate (Grant No. CX2018B230), the International Postdoctoral Exchange Fellowship Program of China Postdoctoral Council (Grant Nos. OCPC2017032, OCPC20180024), and the Fellowship Program of China Scholarship Council.

References

[1] C. Phua, K. Smith-Miles, V. Lee, and R. Gayler, "Resilient identity crime detection," *IEEE Transactions on Knowledge and Data Engineering*, vol. 24, no. 3, pp. 533–546, 2012.

[2] J. Chen, K. Li, Z. Tang, S. Yu, and K. Li, "A parallel random forest algorithm for big data in spark cloud computing environment," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 4, pp. 919–933, April 2017.

[3] H. Wang, H. Yao, D. Kifer, C. Graif, and Z. Li, "Non-stationary model for crime rate inference using modern urban data," *IEEE Transactions on Big Data*, vol. 30, 2018.

[4] Y. Xue and D. E. Brown, "Spatial analysis with preference specification of latent decision makers for criminal event prediction," *Decision Support Systems*, vol. 41, no. 3, pp. 560–573, March 2006.

[5] J. L. Toole, N. Eagle, and J. B. Plotkin, "Spatiotemporal correlations in criminal offense records," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 4, p. 38, July 2011.

[6] P. Rashidi, T. Wang, A. Skidmore, A. Vrieling, and P. Omondi, "Spatial and spatiotemporal clustering methods for detecting elephant poaching hotspots," *Ecological Modelling*, vol. 297, no. 10, pp. 180–186, February 2015.

[7] S. Jeyanthi, N. U. Maheswari, and R. Venkatesh, "An efficient automatic overlapped fingerprint identification and recognition using anfis classifier," *International Journal of Fuzzy Systems*, vol. 18, no. 3, pp. 478–491, June 2016.

[8] C. Li, H. Zhao, and Z. Xu, "Kernel c-means clustering algorithms for hesitant fuzzy information in decision making," *International Journal of Fuzzy Systems*, vol. 20, no. 1, pp. 141–154, January 2018.

[9] K. Li, J. Mei, and K. Li, "A fund-constrained investment scheme for profit maximization in cloud computing," *IEEE Transactions on Services Computing*, vol. 11, no. 6, pp. 893–907, December 2018.

[10] S. Wang, X. Wang, P. Ye, Y. Yuan, S. Liu, and F.-Y. Wang, "Parallel crime scene analysis based on acp approach," *IEEE Transactions on Computational Social Systems*, vol. 5, no. 1, pp. 244–255, 2018.

[11] K. Li, W. Yang, and K. Li, "Performance analysis and optimization for spmv on gpu using probabilistic modeling," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 1, pp. 196–205, January 2015.

[12] C. Liu, K. Li, C. Xu, and K. Li, "Strategy configurations of multiple users competition for cloud service reservation," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 2, pp. 508–520, February 2016.

[13] S. Kaza, J. Xu, B. Marshall, and H. Chen, "Topological analysis of criminal activity networks: Enhancing transportation security," *IEEE Transactions on Intelligent Transportation Systems*, vol. 10, no. 1, pp. 83–91, 2009.

[14] M. G. Mehmet Sait Vural, "Criminal prediction using naive bayes theory," *Neural Computing and Applications*, vol. 28, no. 9, pp. 2581–2592, September 2017.

[15] D. Brown and L. Gunderson, "Using clustering to discover the preferences of computer criminals," *IEEE*

Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans, vol. 31, no. 4, pp. 311–318, 2001.

[16] G. Xiao, K. Li, and K. Li, “Reporting l most influential objects in uncertain databases based on probabilistic reverse top-k queries,” *Information Sciences*, vol. 405, pp. 207–226, 2017.

[17] G. Xiao, K. Li, X. Zhou, and K. Li, “Efficient monochromatic and bichromatic probabilistic reverse top-k query processing for uncertain big data,” *Journal of Computer and System Sciences*, vol. 89, pp. 92–113, 2017.

[18] L. H. Son and N. D. Tien, “Tune up fuzzy c-means for big data: Some novel hybrid clustering algorithms based on initial selection and incremental clustering,” *International Journal of Fuzzy Systems*, vol. 19, no. 5, pp. 1585–1602, October 2017.

[19] Y. Chen, K. Li, W. Yang, G. Xiao, X. Xie, and T. Li, “Performance-aware model for sparse matrix-matrix multiplication on the sunway taihulight supercomputer,” *IEEE Transactions on Parallel and Distributred Systems*, vol. 29, no. 99, 2018.

[20] Y. Chen, G. Xiao, and W. Yang, “Optimizing partitioned csr-based spgemm on the sunway taihulightt,” *Neural Computing & Applications*, 2019.

[21] Y. Chen, K. Li, W. Yang, G. Xiao, X. Xie, and T. Li, “Performance-aware model for sparse matrix-matrix multiplication on the sunway taihulight supercomputer,” *IEEE transactions on parallel and distributed systems*, 2018.

[22] K. Li, X. Tang, B. Veeravalli, and K. Li, “Scheduling precedence constrained stochastic tasks on heterogeneous cluster systems,” *IEEE Transactions on Computers*, vol. 64, no. 1, pp. 191–204, January 2015.

[23] U. of Maryland, “Global terrorism database (gtd),” http://www.start.umd.edu/gtd.

[24] Apache, “Spark,” http://spark-project.org.

[25] H. University, “National supercomputing centre in changsha,” http://nscc.hnu.edu.cn.