# Parallel Co-Location Pattern Mining Discovery: Constraint Neighborhood Approach

Eman M. Refaye[1], Osman Hegazy[1]
[1]Department of Information systems,
Faculty of Computers & Information,
Cairo University,
Cairo,Egypt
eman_refaye@yahoo.com,osman.hegazy@gmail.com

***ABSTRACT:*** *Spatial data mining become one of the important areas because of the rapid evolution in technology which leads in big spatial data. Co-locations pattern mining is an interesting and important issue in spatial data mining area which discovers the subsets of features whose events are frequently located together in geographic space. Spatial proximity is the important concept to determine the colocation patterns from massive data. The computation of co-location pattern discovery is very expensive with big data volume and nearby existence of neighborhoods. So there is number of spatial co-location mining algorithms have been developed to overcome these drawbacks. In this paper, a new a co-location pattern mining framework has been proposed that benefits from the power of parallel processing, in particular, the MapReduce framework to achieve higher spatial mining processing efficiency. MapReduce model have been proven to be an efficient framework solution for big data processing on clusters of commodity machines, and for big data analysis and many applications. The experimental result of the proposed framework shows scalable and efficient computational performance.*

**Keywords:** Spatial Data Mining, Co-location Pattern, Hadoop-MapReduce, Constraint Neighborhood

**Received:** 18 December 2015, Revised 28 January 2016, Accepted 4 February 2016



## 1. Introduction

Data mining in general is searching for hidden and interesting patterns that may exist in generic data. Spatial data mining in particular is discovering the interesting relationships and characteristics that may exist implicitly in spatial data [1].Spatial data mining is a new and rapidly developing area of data mining, concerned with the identification of interesting spatial patterns from data stored in spatial datasets and geographic information systems. GIS are used in various areas such as environmental impact assessment, urban planning, cartography, criminology, traffic analysis, etc. Collection of data is enabled by global positioning systems (GPS) and sensor networks, while computer storage technology enables the storage of enormous quantities of collected data. These advanced technologies are the reason for the existence of a growing number of spatial datasets. The size of spatial datasets and the complexity of dealing with spatial attributes require the use of specialized data mining techniques [2].

Spatial co-location pattern mining, is one of the important area in spatial data mining, has been researched in spatial

data mining techniques. Spatial co-location pattern describes "a set of spatial events which are frequently observed together in a spatial proximity" [3].

Also a co-location pattern determines what these co-located objects, each typically occur in geographical proximity. Identified co-location patterns are interesting and helpful for many applications such as location-based services, public health and climatology, Disease control, Transportation, Business, Social Science, Geology, and Mobile Computing [3], [4].

Co-location rules are models to find the presence of Boolean spatial features in the neighborhood of instances of other Boolean spatial features. Mining Co-location pattern is the process to identify co-location patterns from big spatial datasets with a number of Boolean features. The spatial co-location rule discovery problem looks like the association rule mining problem, but, in fact, it is very different from the association rule mining problem. These differences have been made because of the lack of transactions. It uses spatial predicate as item types. So using of co-location pattern mining enhancing the efficiency of detecting interesting patterns from the very big spatial data. Co-location patterns are discovered by using neighborhood definition and spatial joins. These definitions and algorithms will discuss the detection of co-location pattern from the large spatial datasets [5].

Big data is one of the hotspot in technological area and brings not only large amounts of data but also various data types that would not been considered [10].The evolution of location sensing, mobile computing, and scientific simulation is generating huge quantities of rich spatial data. Finding the solution that is able to translate the plentiful amount of spatial data that surrounds us into meaningful and useful information has led to the rise of spatial data mining [20]. So Spatial data mining has been popularly studied for detecting a specific association relationships between a set of spatial attributes and some of them may be non-spatial attributes. But dealing with large-scale spatial data mining isn't easy because of complex spatial data types, neighbor relationships [19].

One of the drawbacks of co-location pattern mining that is the wasting of cost of time to hold a vast number of candidate pattern sets. Also single processor's memory and CPU resources are very limited, which make an inefficient performance of co-location mining algorithms. Furthermore, because of growth of information, enterprises have to deal with growing amount of spatial data. So, the solution to this problem is parallel and distributed computing [6].

MapReduce offers a simple programming model for parallel data analysis. It is one of the most popular system built based on these ideas is Google's MapReduce and its open-source implementation, Hadoop. It simplifies parallel data processing by abstracting the details of data partitioning, node communication and fault tolerance [7].

This paper is organized as follows: Section 2; presents background of co-location pattern mining algorithms, MapReduce model and Hadoop framework. Section 3; related work is discussed. Section 4; presents the proposed framework. Section 5; shows performance evaluation. Conclusion is presented in section 6.

## 2. Background

### 2.1 Spatial Data Mining Approaches

Spatial data mining is a rising exploration field devoted to the advancement and utilization of novel computational procedures for the examination of big spatial datasets. It envelops methods for finding valuable spatial associations and patterns that are not stored in spatial datasets. Generally these procedures need to manage complex features with spatial data properties. The properties and relationships that have been contained in spatial data are different from transactional data. For instance, transactional data are stored in discrete space of numeric and categorical data rather than spatial data which are stored in continuous space. Transactions in transactional data are independent of each other unlike the spatial data share a variety of spatial relationships among each other.

The complexity of spatial data type and implicit relationships among spatial objects makes the process of discovering spatial patterns from spatial data is more difficult compared to the process discovering patterns from traditional data. Different approaches have been developed for knowledge discovery from spatial data such as spatial classification, spatial association rule mining and spatial clustering [8].

**Spatial Classification.** Spatial data spatial classification is known as that attributes can be grouped with respect to their values

into categories also the attribute values of objects of neighbors may also be related to the membership of objects so that they have to be considered as well Assigning an object to a class from a given set of classes based on the attribute values of the object is the main objective of classification [8].

**Spatial Association:** Spatial identifying the regularities between the items in the large transactional datasets is known as association rule mining. Confidence and support are measures that indicate the strength of the frequency of the association rule [8], [20].

**Spatial Clustering:** The task of collecting the objects of a datasets into meaningful detectable subclasses that is clusters is known as clustering so that the members of the same cluster are as similar as possible whereas the members of different clusters differ as much as possible from each other. Clustering algorithms can be categorized into five types they are model-based clustering methods, partitioning algorithms, hierarchical algorithms, density based clustering and grid-based methods [8].

**2.2 Co-location Pattern Mining**

There is a similarity between spatial co-location pattern mining and association mining. A rule of the form "A→B" is a spatial association rule, where A and B are sets of predicates and some of which are spatial ones. For big datasets many relationships may exist but some may occur rarely or may not hold in most cases [3].

A set of instances S, a set of spatial features F, and a spatial neighbor relationship R over S. Spatial neighbor relationship R may be one of the following categories: directional, topological and distance. R could be distance relationships (e.g. Euclidean distance metric) and topological relationships (e.g. linked, intersection), and mixed relationships (e.g. the shortest distance of two points on a map) [9].

A co-location $C = \{A, B, C, \ldots..\}$ is a subset of spatial features $C \subseteq F$ whose instance objects are frequently observed in a nearby area according. A co-location instance $I \subseteq S$ of a co-location C is defined as a set of objects which includes all features types in C and forms a clique under the neighbor relationship *R*. i.e., {*A*.2, *B*.4, *C*.2} is a co-location instance of {*A*, *B*, *C*}. The prevalence of co-locations is often measured by participation index [10], [11].

The participation index $PI(C)$ of a co-location $C = \{f_1, f_2, \ldots.. f_k\}$ is defined as: $PI(C) = min\ e \in C\{\text{PR(C, } fi\}$, where $1 < i < k$, and $PR(C, f_i)$is the participation ratio of event type fi in the co-location C that is the fraction of objects of event fi in the neighborhood of instances of co-location $C\text{-}\{f_i\}$,

$$PR(C,f_i)=\frac{\text{number of distinct objects of fi in instances of C}}{\text{number of objects of fi}}$$

The prevalence measure indicates wherever an event in C is observed, with a probability of at least PI(C), all other events in C can be observed in its neighborhood. If the participation index of an event set is greater than a user-specified minimum prevalence threshold min prev., the event set is called a co-location or co-located event set [10].

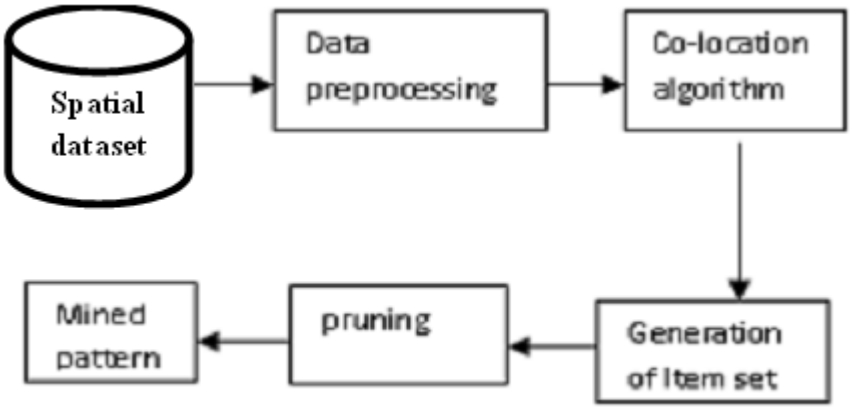


Figure 1.General architecture of co-location mining algorithm [12].

### 2.3 Hadoop Framework

Hadoop is open source software that runs on a cluster of machines. Hadoop supplies both distributed processing and distributed storage for very large data sets [13].It consists of two models: Hadoop Distributed File System (HDFS) which is the distributed storage model which designed after Google File System (GFS) and Map-educe; the programming model. Now it supports additional models and systems such as: HBase; a distributed column-oriented database, Hive; a data warehouse system, Avro; a data serialization system, Chukwa; a data collection system, ZooKeeper; ahigh-performance coordination service for distributed application, and Pig; a high level data-flow language[14].

HDFS - Hadoop Distributed File System HDFS, gives the programmer unlimited storage, HDFS implementation is modeled after GFS, Google Distributed File system .However; here are additional advantages of HDFS.

• **Horizontal Scalability:** Thousands of servers holding petabytes of data. When you need even more storage, you don't switch to more expensive solutions, but add servers instead.

• **Commodity Hardware:** HDFS is designed with relatively cheap commodity hardware in mind. HDFS is self-healing and replicating.

• **Fault Tolerance:** Every member of the Hadoop knows how to deal with hardware failures. If you have 10 thousand servers, then you will see one server fail every day, on average. HDFS foresees that by replicating the data, by default three times, on different data node servers. Thus, if one data node fails, the other two can be used to restore the third one in a different place [13].

| Job type | Job Function | Skills |
|---|---|---|
| Hadoop Developer | Develops MapReduce jobs, Designs data warehouses | Java,Scripting,Linux |
| Hadoop admin | Manages Hadoop cluster, designs data pipelines | Linux administration, Network management, |
| Data Scientist | Data mining and figuring out hidden knowledge in data | Experience in managing large cluster of machines |
| Business Analyst | Analyzes data | Math, data mining algorithms<br>Pig, Hive, SQL superman, familiarity with other BI tools |

Table 1. Simulation Parameters Hadoop Roles [13]

Hadoop-MapReduce is a software framework used for writing applications which process vast amounts of data (multi-terabyte data-sets) in-parallel on large clusters (thousands of nodes) of commodity hardware in a reliable, fault-tolerant manner [15].

### 2.4 MapReduce Model

MapReduce is a programming model for expressing distributed computations on massive amounts of data and an execution framework for large-scale data processing on clusters of commodity servers. MapReduce simplifies parallel processing by abstracting away the complexities involved in working with distributed systems, such as computational parallelization, work distribution, and dealing with unreliable hardware and software. The MapReduce model abstraction is presented in Figure 2. A MapReduce job is executed in two main phases of user defined data transformation functions, namely, map and reduce. When a job is launched, the input data is split into physical blocks and distributed among nodes in the cluster. Such division and distribution of data is called sharding, and each part is called a shard. Each block in the shard is viewed as a list of key- value pairs. In the ûrst phase, the key-value pairs are processed by a mapper, and are provided individually to the map function. The

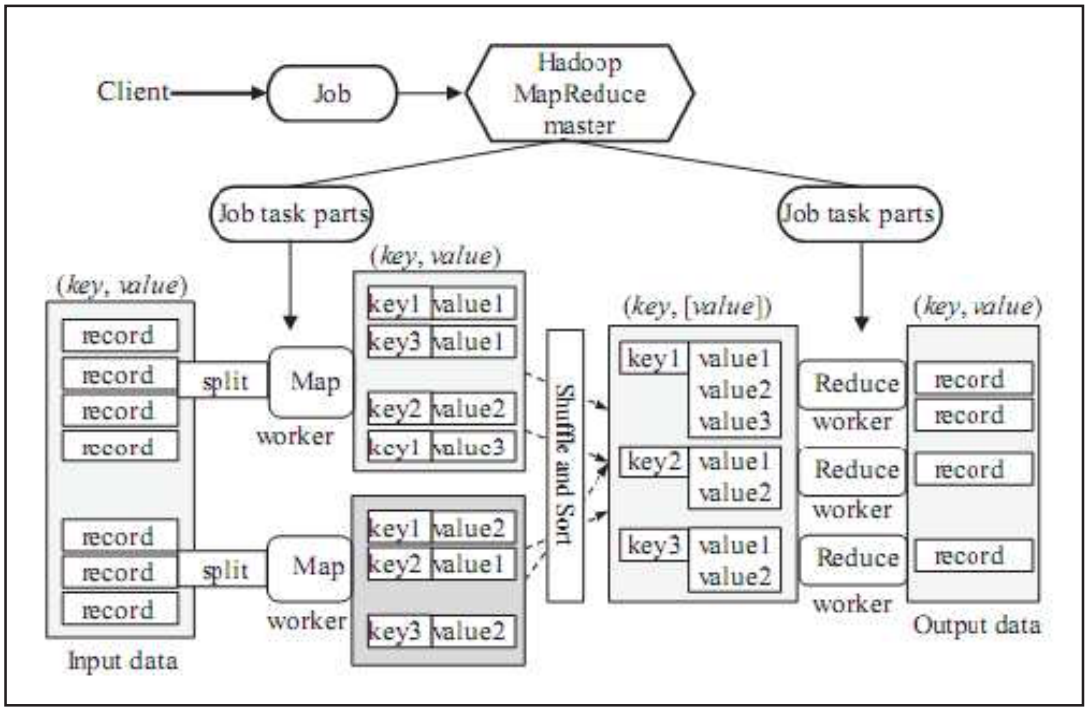


Figure 2. MapReduce program and execution models [10]

output of the map function is another set of intermediate key- value pairs [10].

A MapReduce job usually splits the input data-set into independent chunks which are processed by the map tasks in a completely parallel manner. The framework sorts the outputs of the maps, which are then input to the reduce tasks. Typically both the input and the output of the job are stored in a file-system. The framework takes care of scheduling tasks, monitoring them and re-executes the failed tasks [15]. .MapReduce takes care of distributed computing. It reads the data, usually from its storage, the Hadoop Distributed File System (HDFS), in an optimal way. However, it can read the data from other places too, including mounted local file systems, the web, and databases. It divides the computations between different computers (servers, or nodes). It is also fault-tolerant [13].

**3. Related Work**

Candidate co-location pattern instances have been discovered using several proposed approaches. In [17] the authors of this approach groups neighboring instances with a non-overlapping instance grouping constraint arbitrarily. Deferent instance sets by the order of grouping can be obtained from the disjoint grouping method [16].

Other works on spatial co-location mining have presented different approaches for identifying co-location instances. According to [18] presented these contributions such as, some uses space partitioning and non-overlap grouping scheme is used for ûnding neighboring objects for a frequent neighboring feature set. However, there are number of missed co-location instances across partition areas and incorrect results generated from the distinct space partitioning approach.

Other works on spatial co-location mining have presented different approaches for identifying co-location instances. According to [18] presented these contributions such as, some uses space partitioning and non-overlap grouping scheme is used for ûnding neighboring objects for a frequent neighboring feature set. However, There are number of missed co-location instances across partition areas and incorrect results generated from the distinct space partitioning approach.

In [3] a novel constraint neighborhood based approach had been proposed to find co-location patterns. This approach can discover different co-location patterns such as star and clique (as shown in Figure 3), including single and complex self co-locations. Based on the constraint neighborhood idea, this method neither needs to perform spatial or instance neither joins nor checks for cliques to find co-location instances.

The constraint neighbor co-location approach discovered the colocation patterns across two algorithms:

• **Algorithm1:** Starts with determining the constraint neighbors by scanningthe spatial object dataset to of each object, and then builds a set of single feature (size-1) co- location candidates.

• **Algorithm 2:** It generates size-k pattern candidates based on size- (k-1) prevalent patterns by applying the level-wise approach. Also checks whether all subsets of the new candidate are prevalent. The algorithm uses the participation index that has the anti-monotonic property to measure the prevalence of new candidates [3].

Comparing the advantages of this approach, to others, this algorithm neither has to perform instance joins nor checks for cliques co-location patterns. Thus creating to an important performance gained form co-location patterns discovered of mining process. But this approach finds co-location patterns with more reads in the candidate set generation process.

Authors in [19], [10] they used partition strategy edges in the neighbor graph to divide the search space of co-location patterns. Dividing the edges of the neighbor graph is done according to this rule such that each vertex v (a spatial object) keeps the relationship edge with other vertex u when v.type < u.type. There is a total ordering of the event type was assumed.

The partition strategy is assumed to divide neighbor relations without duplicating or missing any relationships needed for co-location patterns. Also these algorithms have been implemented using the power of parallel processing, in particular, the MapReduce framework to achieve higher spatial mining processing efficiency. But these models do not discover both single self co-location patterns and complex self co-location patterns also need more computations in checks for cliques co-location instances.

In this paper a new implementation of parallel Co-location Pattern Mining algorithm based on Constraint Neighborhood Approach using Hadoop-MapReduce model to overcome the problems found in the previous algorithms.

**4. Proposed System**

In this section, we present our parallel colocation pattern mining model based on constraint neighborhood approach and MapReduce model, Which starts by Identifying spatial input dataset reordering the instances records according their types then according to ID'S. COUNT find total number of instances for each feature type Find CN for each object in the dataset according to spatial relationship between objects such that $R(o_i, o_j)$ computed according to constraint neighbor approach. Find list of constraint neighbor for each object.

Eliminate/remove object that are irrelevant. If object has no neighbors and doesn't included in a list of constraint neighbor. Finally generate colocation patterns. Our proposed system find prevalent co-location patterns and implemented across three map/reduce jobs.

**4.1 Phase 1: Preparation of Spatial Files**

Our spatial dataset obtained from more than one file. So there is need to combine these files into one file using MapReduce model. First the map job assigns each object $o_i$ to its corresponding feature type $f_i$. Then applying the reduce job for sorting these objects according to their features types and within the same type according to the objects ID'S. Then counting and saving the number of object instances per feature type for future prevalence calculation. The ordering task used for eliminating the duplications and missed instances. If we have D spatial dataset consists of $n$ objects $D = \{o_1, o_2, ..., o_n\}$ , and F set of m features $F = \{f_1, f_2, ..., f_m\}$, $(m << n)$, i.e., $f_1 < f_2 < ... < f_m$, $id_1 < id_2 < ... < id_n$ .

**4.1 Phase 2: Generate the List of Constrain Neighbors (CN) For Each Object**

In this phase the main job is to generate the list of neighbors of each object according to the definition of the constraint neighborhood approach CN by check each object with other to find the constraint neighbor list according to the following: For clique colocation: CCN $(\{o_i\})$: = $sort < (\{o_j \mid (o_i, o_j) \in R \wedge ((o_i.type < o_j.type) \vee (o_i.type = o_j.type \wedge o_i.id < o_j.id)), (j \neq i)\})$. For star colocation patterns: SCN $(\{oi\})$: = sort<$(\{o_j \mid (o_i, o_j) \in R, (j \neq i)\})$.Then builds a set of single-feature (size-1) co-location candidates.

**4.3 Phase 1: Colocation Patterns Generation**

```
1: procedure MAPPER(key,value=o)
2:F( o_i) ← o_i ;
3:emit( F(o_i), o_i) ;
4:end procedure

5:procedure REDUCER(key= , value=[ o])
6:objectSet ← o_i ;
7:sort(objectSet) ;
8:count ← sum(value) ;
9:save( objectSet, count) ;
10:emit(F( o_i), o_i ) ;
11:end procedure
```

Figure 3. A preparation of spatial files

```
1: procedure MAPPER(key=o_i, value= o_j)
2:findN( o_i);
3:emit(o_i, o_j) ;
4:end procedure
5:procedure REDUCER(key=o_i, value=o_j)
6:checkCN(o_i, o_j ) ;
7:GroupCN(o_i);
8:emit(o_i, CN);
9:end procedure
```

Figure 4. Generate the list of constrain neighbors (cn) for each object

In this phase the algorithm applies the level-wise approach to generate size-k pattern candidates from size-(k-1) prevalent patterns and checks whether all subsets of the new candidate are prevalent. The pattern instances of the new candidate are discovered. The algorithm uses the participation index that has the anti-monotonic property to measure the prevalence of new candidates.

**5. Experiment and Results**

In this section, we present the results of our experimental evaluation of the proposed model constraint neighborhood based on MapReduce to mine co-location patterns. We show the effectiveness of our approach to find co-location patterns by comparing the patterns discovered to those of the constraint neighborhood approach represented in [3].

All experiments have been performed on a single machine that contains: Windows 7 64-bit, Eclipse Java EE IDE for web developers version Mars Release (4.5.0), Apache Hadoop 2.3.0 run on the stand-alone mode, ESRI geometry API, Spatial SDK Hadoop. All algorithms have been implemented in Java: JDK version 4.5.0. Experimental dataset used is spatial data about Leeds city that contains multiple features such as schools, accidents, hotels, traffic signals and so on. The dataset sample contains 6473records with distinct feature109. The following table demonstrates the performance of the proposed algorithm compared by constraint neighborhood colocation miner using different prevalence threshold.

```
1:  procedure MAPPER (key=oi, value=CN)
2:  k := 2;L1 := CN;
3:  while Lk-1 ≠ ϕ do

4:        foreach p, q ∈ Lk-1 s.t.
5:        p =[c1, ..., ck-2,ck-1],
6:        q =[c1, ..., ck-2,ck], ck∈p.CN
7:        DO
8:          c =[c1, ...ck-1,ck];
9:            if checkSubsets(c, Lk-1) then
10:             foreach I ∈ p.instances do
11:           foreach
12:                       (o ∈ I.CN) and (o.type = ck) do
13:             newI.objs := I.objs ∪{o};
14:                   newI.CN :=getCN(I.CN,o);
15:                   c.instances.add(newI);
16:               end for
17:             end for
18:           end if
19: emit(c,c.instaces);
20: procedure REDUCER(key=c, value=c.instances)
21:    if Prevalence(c) ≥ ρ then
22:    Ck.add(c);
23: end procedure
```

Figure 5. Generate colocation patterns discovery

| No. of Records 6473 and 109 Unique Features | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Prevalence | 0.01 | 0.015 | 0.020 | 0.025 | 0.030 | 0.035 | 0.040 | 0.045 | 0.050 |
| MapReduce CN Co-Location | 59.30 | 59.16 | 58.50 | 59.90 | 60.90 | 58.70 | 58.88 | 58.34 | 58.38 |
| Constraint Neighbor | 315 | 312 | 304 | 301 | 294 | 288.16 | 286.30 | 279.20 | 279 |

Table 2. Execution time vs. Prevalence threshold

## 6. Conclusion

In this paper an efficient parallel co-location pattern mining approach proposed that effectively discovers colocation patterns and self co-location patterns based on constraint neighborhood approach. Also the drawbacks in previous approaches have been enhanced by using Hadoop –MapReduce model which enable us with the parallel and distributed processing manner.

## References


[1] Sarangi, Banalata., Sahoo, Laxman. (2013). A Survey on Spatial Association Rule Mining Technique and Algorithms for Mining Spatial Data. *International Journal of Scientific & Engineering Research*, 4 (12), 1664- 1670.

[2] Lavrac, N., Jesenovec, D., Trdin, N., Kosta, N. M. (2008). Mining spatiotemporal data of traffic accidents and spatial pattern visualization. *Metodoloskizvezki*, 5 (1), 45-63.

[3] Canh,., Michael Gert. (2012). A constraint neighborhood based approach for co-location pattern mining, *Fourth International Conference on Knowledge and System Engineering*, 128-135.

[4] Yang, H., Parthasarathy, S., Mehta, S. (2005). A generalized framework for mining spatio-temporal patterns in scientiûc data. *In*: Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, *ACM*, 716–721.

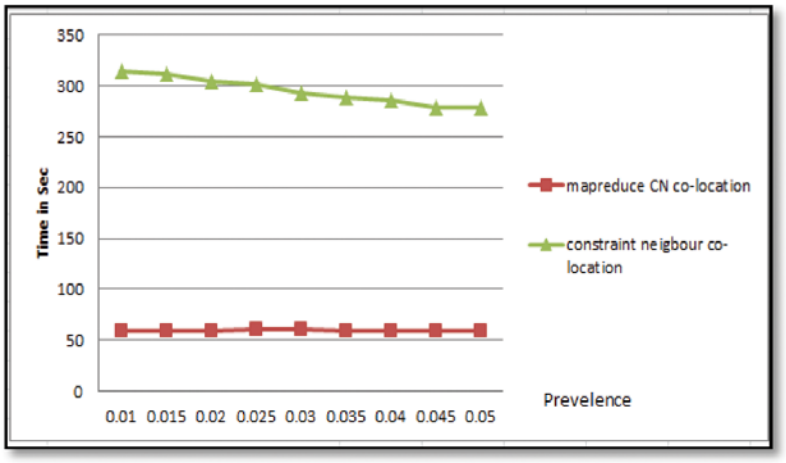


Figure 6. Performance evaluation of the proposed colocation miner

[5] Sundarama, Venkatesan Meenakshi., Nagavelub, Arunkumarth., Paneerc, Prabhavathy . (2012). Discovering Co-location Patterns from Spatial Domain using a Delaunay Approach. *International Conference On Modelling Optimization and Computing* 38, 2832–2845.

[6] Satija, Sonali., Nath, Rajender. (2015). Performance Improvement of Apriori Algorithm Using Hadoop. *International Journal of Advanced Research in Computer Science and Software Engineering*, 5 (6) June, 765-768.

[7] Ofalvi, Gy" oz" o Gid´. (2007). Spatio–Temporal Data Mining for Location–Based Services. Ph.D. Thesis, Faculties of Engineering, Science and Medicine at Aalborg University, Denmark.

[8] Kiran Kumar, G., Premchand, P., Venu Gopal, T. (2012). Mining Of Spatial Co-location Pattern from Spatial Datasets. *International Journal of Computer Applications*, 42 (21) 25-30.

[9] Wang, Lizhen., Baoa, Yuzhen., Lub, Zhongyu. (2009). Efficient Discovery of Spatial Co-Location Patterns Using the iCPI-tree. *The Open Information Systems Journal*, 3, 69-80.

[10] Yoo, Jin Soung., Boulware, Douglas., Kimmey, David (2014). A Parallel Spatial Co-location Mining Algorithm Based on MapReduce, *IEEE International Congress on Big Data, Anchorage, AK*, 25 – 31.

[11] Shekhar, S.. Huang, Y. (2001). Co-location Rules Mining: A Summary of Results. *In*: Proceedings of International Symposium on Spatio and Temporal Database, 236–256.

[12] Rushirajsinh, L., Zala., Brijesh, B., Mehta., Mahipalsinh, R. Zala. (2014). A Survey on Spatial Co-location Patterns Discovery from Spatial Datasets. *International Journal of Computer Trends and Technology (IJCTT)*, 7 (3) 137-142.

[13] Hadoop, https://github.com/hadoop-luminated /hadoop-book

[14] Yahya., Othman., Hegazy, Osman., Ezat, Ehab. (2012). An Efficient Implementation Of Apriori Algorithm Based On Hadoop-Mapreduce Model. *International Journal of Reviews in Computing*, 12, 59-67.

[15] MapReduce Tutorial, https://hadoop.apache.org/ docs/r1.0.4/mapred_tutorial.pdf

[16] Yoo, Jin Soung., Shekhar, Shashi. (2004). A Partial Join Approach for Mining Co-location Patterns. GIS 04, ACM, Washington, DC, USA, 12–13 November, 2004.

[17] Morimoto, Y. (2001). Mining Frequent Neighboring Class Sets in Spatial Databases. *In:* Proceedings ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2001.

[18] Yoo, Jin Soung., Shekhar, Shashi., Celik, Mete. (2005). A Join-less Approach for Co-location Pattern Mining: A Summary of Results. 5th IEEE International Conference on Data Mining.

[19] Jin SoungYoo., Douglas Boulware. (2013). A Framework of Spatial Co-Location Mining on MapReduce. I*EEE International Conference on Big Data*, 44.

[20] S. Park., J. S. Yoo. (2014). Spatial Association Mining for Focal Events in Cloud Computing. *In*: Proceedings of the International Conference on Advances in Big Data Analytics, 157-164.