International Journal of Computational Intelligence Systems
Vol. **14(1)**, 2021, *pp.* 266–281
DOI: https://doi.org/10.2991/ijcis.d.201126.002; ISSN: 1875-6891; eISSN: 1875-6883
https://www.atlantis-press.com/journals/ijcis/

Research Article

User Community Detection From Web Server Log Using Between User Similarity Metric

M. S. Bhuvaneswari*, K. Muneeswaran

Department of Computer Science and Engineering, Mepco Schlenk Engineering College, Sivakasi, Tamilnadu, India

ARTICLE INFO

Article History

Received 17 Jan 2020
Accepted 06 Nov 2020

Keywords

Session identification
Sequential pattern mining
Clustering
Community detection

ABSTRACT

Identifying users with similar interest plays a vital role in building the recommendation model. Web server log acts as a repository from which the information needed for identifying the users and sessions (pagesets) are extracted. Sparse ID list and Vertical ID list are used for identifying the closed frequent pagesets which is beneficial in terms of memory and processing. The browsing behavior of a user is identified by computing similarity among the pageset that belongs to the user. A new metric for measuring within user similarity is proposed. The novelty in this approach is, only the users having consistent behavior over the time are taken into consideration for clustering. Consistent users are then clustered by different clustering techniques such as Agglomerative, Clustering Large Applications Using RANdomized Search (CLARANS) and proposed Density-Based Community Detection (DBCD). The quality of the clusters formed by DBCD is found to do better for clustering the users. The outcomes show significant improvements in terms of quality and speed of the clustering.

1. INTRODUCTION

In order to hold the present market needs, the way in which the business firm makes decision changes day by day. Making right decision at right time is important for the enhancement of the business. The recommendation model plays an important role for making such decisions. Nowadays people prefer to use website for any kind of transactions, therefore it is inevitable to provide service in a way that it would satisfy their interest and needs. The interest of an individual can be found out by discovering hidden navigational patterns. Context aware interactive web page navigation with prediction capabilities has necessitated the web page usage mining in extracting the patterns from web server logs [1].

Pattern mining is the process of discovering interesting, useful, hidden and unexpected patterns from the dataset. Various types of patterns that can be identified include frequent item sets, sequential rules and periodic patterns. In those types of patterns, sequential patterns plays a major role in numerous real life applications such as bioinformatics, e-learning, market basket analysis, texts and webpage click-stream analysis due to the fact that data is naturally encoded as sequences of symbols.

Frequent itemset mining and association rule mining deals with identifying the set of items which occurs together in a transaction and relationship between the items [2–5]. They do not take it into consideration the order of the items in the transaction. The drawback of this methodology is that it may not be appropriate for sequence prediction type of applications where the order of items is important. Ref. [6] made use of the sequential rule pattern to explore the user behavior and foresee whether they are intruder or not.

Sequential pattern mining is a data mining task specialized for analyzing the data, to discover sequential patterns [7]. More precisely it is the process of discovering interesting subsequence from a set of sequences, where the interestingness of the subsequence can be measured in terms of various criteria such as its occurrence frequency and length. On the other hand sequential pattern mining is concerned with finding statistically relevant patterns among the data examples where the values are delivered in a sequence.

Sequential pattern mining has numerous applications in the area of medicine, DNA sequencing, Web log analysis, computational biology. Large amount of sequence data have been and continue to be collected in genomic and medical studies, security applications, business applications, etc.

In Web log analysis, the behavior of a user can be extracted from member records or log files. In this case, the sequences could be sequence of Web Pages visited by users on a website. Then a sequential pattern mining algorithm can be used to discover sequences of web pages that are often visited by users [8].

Ref. [9] proposed a Generalized Sequential Pattern Mining (GSP) Algorithm which is faster compared to the Apriori approach. It performs a level wise search where n-length sequences are generated from n − 1 length sequences. It is not appropriate for large sized database as it has to scan the database multiple times to find the support. It tries to generate sequences from already discovered

* *Corresponding author. Email: bhuvaneswari@mepcoeng.ac.in*

sequences without considering the actual sequences in database which may increase the processing time. To overcome the disadvantages of GSP, columnar database format was introduced where an IDList is maintained for each item. It is used to store the details about the transaction and the itemsets in which an item occurs. It can be used directly to compute the support for an item instead of scanning the database [10,11]. The problem with this approach is when an item appears in many transactions then the size of IDList continues to grow and IDList for a new pattern requires a join operation on the IDLists of the items in the pattern which makes it costlier.

To overcome the limitations in previous approaches, pattern-growth algorithm was introduced where the patterns in original database is scanned repeatedly for generating larger patterns instead of forming larger patterns from smaller ones [12,13]. Projected database was used to reduce the database scan. The drawback of this approach is database must be scanned repeatedly to form projected database. In this approach the time taken for constructing the itemset depends on the size of the projected database and for lengthy sequence it takes longer time. In our approach we have used the vertical representation to overcome it.

Many researches were carried for grouping users and for finding communities where each user is considered as a point in the Euclidean space [14,15]. The similarity among the users is found by computing pairwise distance between the points in the space. The network-based approaches Refs. [16–18] takes as input the graph and identifies communities based on the edges/link that exists between the pair of users. Many hierarchical clustering-based approaches were proposed for identifying communities [19–21]. Ref. [19] made use of the PrefixSpan algorithm for generating the frequent patterns. To reduce the number of patterns generated, only patterns whose length is greater than three is taken into consideration. The travelers were grouped based on the patterns generated. The problem with this approach is that if the size of the dataset is large and number of patterns with length>3 is used then the process becomes costlier in terms of time and space. The problem with hierarchical clustering is they are good at discovering strong communities but they cannot be used for finding arbitrary shaped clusters. The goal of this paper is to cluster the data points associated with the users to form communities even they are distributed in an arbitrarily shaped cluster.

The contribution of this paper is (i) extended log format is used for session identification, (ii) a density-based clustering algorithm is used for finding community among the users of arbitrary shape, (iii) closed frequent pageset (CFP) is used instead of frequent pageset (FP) thus reducing the search space, (iv) vertical representation is used for identifying the FP and the CFP to avoid the usage of projected database and scanning the database multiple times, (v) the processing time and usage of memory is reduced by considering only consistent users who repeatedly visits the similar set of pages for clustering instead of including all users. During the mining process, a non-Euclidean distance metric is used for computing the similarity between the users based on the sequence of pages visited by them. The recommendation model could use the clusters formed to generate suggestions specific to a user.

The rest of the paper is structured as follows. In Section 2, the past works done with regards to the proposed work are examined. Section 3 details the proposed work which includes the expulsion of information which is not significant, identifying unique users, grouping of records of a specific user and grouping users of similar interest. Section 4 examines the trials that are carried out and the outcomes received. Section 5 gives review and the enhancement that can be carried out.

2. RELATED WORK

Due to advent of many applications added to the digital era, Web mining has gained momentum in recently. The web log files forms the major source for Web Usage Mining. The web log files can be collected from web server, proxy server and client (browser). Log files stored in the web server contains information about the users accessing the web site. It contains information such as date and time of access, IP address of the client, page requested, browser, Operating System (OS) and other information related to the user. Proxy server acts as an intermediator between client and the web server. Caching at proxy reduces the response time and network traffic [22]. The Log files stored in proxy server contains information about several users using the same proxy server. Log files stored in client side can be used to collect data about the client and avoids the problems associated with identifying the sessions. But for collecting the data at client side requires the cooperation of the client. Client side log file can be used for identifying user navigation behavior [23,24]. In order to collect data about different clients using different proxy servers, it is best to use the Web server log files. In our approach we have used Web Server Log file for collecting data.

In a web mining work, data cleaning forms the core part of session identification [25]. It deals with extracting relevant details from the web log file by converting the unstructured data into a structured format. Whenever a page is visited several image files are also downloaded and an entry will be created in web log file. These log entries must be deleted as part of data cleaning. The cleaned data can be stored in the log table for further processing [26]. IP address field from web server log can be used for user Identification [27]. The problem with this approach is if website is accessed through proxy server then multiple users will be having the same IP address. It can be used in applications like server side caching where importance is given to the pages which are frequently visited and not to find out who visited the pages. Cookie-based user identification is proposed by Ref. [28] which requires user cooperation.

Session Identification deals with identifying the sequence of web pages visited by the user over a period of time. Two different heuristics are adopted for identifying the web sessions. In time-oriented heuristics page stay time or fixed session duration is used for identifying the sessions. In referrer-based heuristics the link between the web pages are used for identifying the sessions.

In the work proposed by Ref. [23] a time threshold of 30 minutes was used for identifying a session that belongs to a user, where they have studied the relation between the average frequency with average path length per site per visit. This session duration of 30 minutes is still being used in many research works. Many researchers have adapted different strategies for identifying the sessions in terms of duration, referrer based and its combination of these [29–32]. Ref. [33] made use of the sitemap for identifying the link between web

pages. The problem with existing approaches is they have used the common log format (CLF) for session identification. The requested url of previous entry and requested url of subsequent entry is used for finding the link between the WebPages. In our approach we have used the extended log format which includes the referrer url field for finding the link between WebPages.

Closed frequent itemset can be considered as the lossless representation of the frequent itemset. CloSpan [34] algorithm first generates candidate CFPs and then performs postpruning. If there are large numbers of CFP then more memory space is needed and time taken for checking the closure property is also high. In Ref. [35] Bi-Directional Extension (BIDE) algorithm was proposed for finding CFP without candidate generation. It makes use of the projected database for finding frequent sequences. It is suitable for dataset which sparse and low support threshold. Clasp [36] made use of the vertical database format for finding the CFP and computing the support without scanning the entire database. It may produce large number of patterns which are not in the input. Ref. [37] introduced a data structure named co-occurrence map (CM) which can be integrated with any of the vertical mining algorithms which greatly reduced the time taken for mining the closed frequent items. Clo-FAST algorithm was presented by Ref. [38] which uses sparse ID lists and vertical ID lists to compute the support of the sequential patterns quickly.

User community detection/Clustering is one of the major applications of the data mining tasks where the likeminded users could be brought under one group. Many researches were carried out for clustering based on the usage of the FP [39,40]. Ref. [41] used the ratings of the items in the itemset for clustering the users by giving weightage for the items. Ref. [42] made use of agglomerative hierarchical clustering for clustering the sessions and Similarity Between Session (SBS) measure for finding similarity between the patterns. An initial method for setting the initial partitions and number of clusters was proposed in Ref. [43]. Ref. [44] made use of the Density Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm for clustering the sessions and sequence similarity measure for finding the similarity between patterns. Problem with this approach is the weightage for given for the set similarity and sequence similarity must be chosen appropriately as it determines the quality of clustering. Ref. [45] made use of self-organizing map (SOM) for clustering the students. But SOM is ideal for generating only low-dimensional input-space representation. A divide and agglomerative algorithm was used to find the user communities in social networks [46]. Ref. [47] proposed a framework which makes use of density-based techniques for semi supervised clustering. A recommendation model by clustering the users using agglomerative clustering was proposed by Ref. [48] in which the similarity between items preferred by the users are taken into account and not the order of the items.

3. PROPOSED WORK

User Community detection involves various steps such as i) Data Preprocessing, ii) User/ PageSet Identification, iii) CFP Identification, iv) User Community Detection/ Clustering users based on similarity. The architecture of the entire process is given in Figure 1.

In data preprocessing the unstructured data is converted into structured data to extract the needed information. Records which are

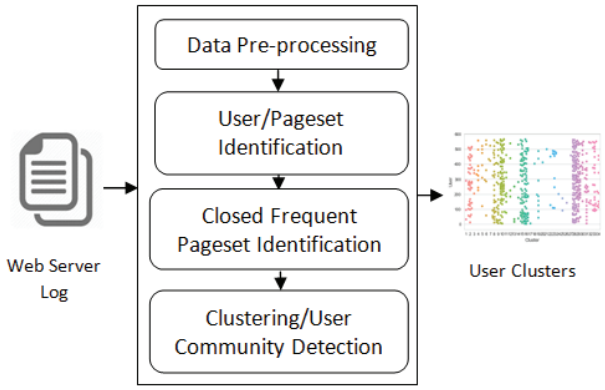


Figure 1 Block diagram of the user community detection.

not relevant to group the users such as image request, robot request and unsuccessful request are removed during preprocessing. After preprocessing the records that belong to individual users are identified based on the IP address, browser and OS. In order to identify the sequence of pages (pageset) visited by the user, requested url and referrer url of the log record is used and the sequence of pages visited by the user is grouped into multiple pagesets/sessions [49].

The identified users and the pagesets are given as input to the CFP identification process. It is used for reducing the voluminous number of redundant pagesets associated with each and every individual user. The CFPs associated with the users undergoes the clustering process. The within user similarity metric is used for checking whether a user has consistent navigation behavior or not. If the user is not consistent then the users along with their pagesets are removed from further processing. Various clustering techniques are applied on the identified consistent users and are clustered based on their CFPs. The resultant clusters are evaluated using the intrinsic quality measure. The proposed work is suitable for clustering when a user can be represented as a data point in the pattern space and the distance metric used is non-Euclidean. The advantages of using our proposed approach are i) the memory space and processing time is reduced by considering the CFPs instead of FPs for clustering, ii) importance is given to sequential order rather than frequency of the pages visited by the users while computing similarity, iii) taking into account only the users who repeatedly visits the same set of pages will improve the accuracy of recommendation.

3.1. Closed Frequent Pageset (CFP) Identification

Sessionization results in the generation of large of number of pagesets for one user. In order to reduce the number of pagesets only FPs are taken into consideration, for clustering. Pageset Support can be defined as the number of sequences which contains the pageset relative to the total number of sequences. Minimum support (msup) is defined as the minimum pageset support required for classifying a pageset as FP.

Example 3.1. FP is defined as the set of page(s) visited by the user whose support is greater than a specific threshold. Let us consider a user who visits the pages of a website in the following sequence over period of time t1 = <1, 3, 4>, t2 = <1, 2, 4, 3>, t3 = <4, 5> and t4

= <3, 2, 4>. The pages frequently visited by the user are ≪1>, <2>, <3>, <4>, <1, 3>, <1, 4>, <2, 4>, <3, 4≫ with a msup of 0.5.

Example 3.2. Maximal Frequent PageSet (MFP) is defined as the set of page(s) visited by the user which does not have any superset in the FP generated. It does not take into consideration the support (sup). For the above example the maximal frequent pages are ≪1, 3>, <1, 4>, <2, 4>, <3, 4≫. The problem with this approach is that if for an example: FP = ≪1> with sup:1.0, <2, 3> with sup:1.0, <2, 4, 3> with sup:0.5≫ then MFP = <1, <2, 4, 3≫with the absence of <2, 3>. Here more appropriate pattern would be <2, 3> instead of <2, 4, 3>.

Example 3.3. CFP is defined as the set of page(s) visited by the user which does not have any superset with the same support in the FP generated. If FP = ≪12, 16> with sup:1.0, <12, 12, 12, 16> with sup:1.0> then MFP and CFP = ≪12, 12, 12, 16≫. On the other hand, if FP = ≪12, 16> with sup:1.0, and <12, 12, 12, 16> with sup:0.75≫ then MFP = ≪12, 12, 12, 16≫ whereas CFP = ≪12, 16>, <12, 12, 12, 16≫ as both have different support.

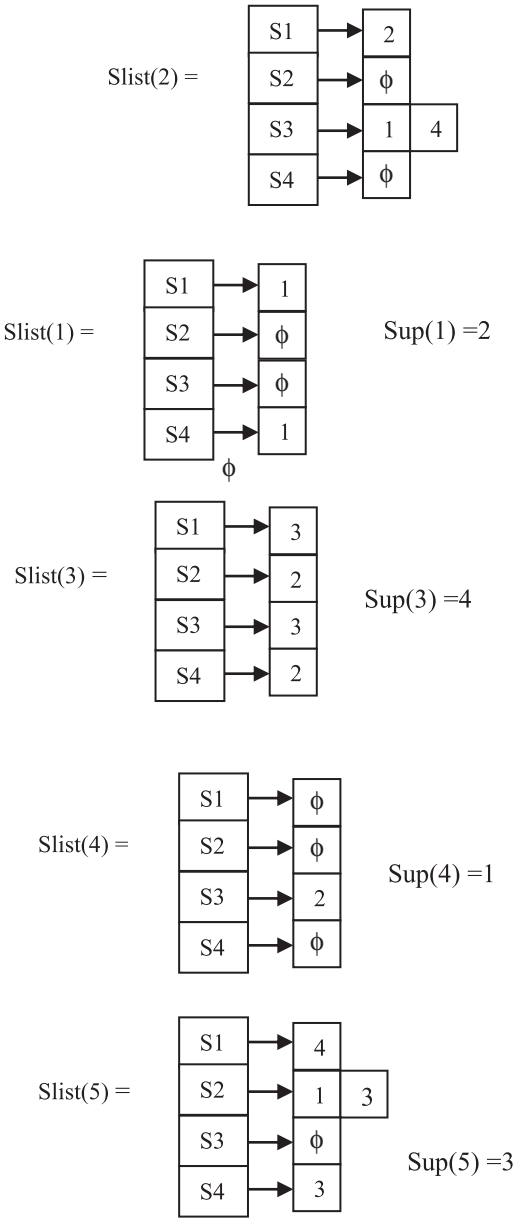
If the FP generated by PrefixSpan algorithm is used for clustering the users then each individual user will be having a large number of FP and the time taken for processing the pageset will be high [19]. To improve the time complexity, Sparse ID list (Slist) and Vertical ID list (Vlist) used in CloFAST algorithm is employed to identify the CFP. The identified CFP is used for clustering the users which is our contributory work not cited in any existing literature [38]. The supports of individual pages in Session database S are identified using the sparse ID list. Individual Pages whose support is greater than msup threshold is identified using the algorithm findClosed-Sequence and is given as input for the sequence extension process. To check whether a sequence can be extended or not, vertical ID list (Vlist) is used. Vlist can be computed from Slist. A sequence can be extended by adding the last page of the sibling to the sequence being extended and by adding the last page of the sequence to itself. The closed sequence tree data structure is used for storing the closed sequential patterns generated as given in extendNodeSequence algorithm. The algorithm is applied to each and every user to identify the CFP of that user.

Example 3.4. Sparse ID list (Slist) for a page is used for storing the position of the page in respective sessions. Consider a Session Database S which consists of a set of sessions as shown below:

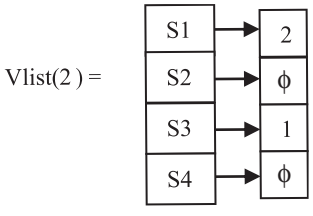
SessionID	Pages Visited			
S1	1	2	3	5
S2	5	3	5	
S3	2	4	3	2
S4	1	3	5	

Each session consists of a set of pages P. For every individual page present in S, Slist is computed which enables fast computation of support of the page. Slist for a page p(Slist (p)) is used for storing the position of the page p in respective sessions. Support for a page is computed by counting the number of nonnull list in Slist.

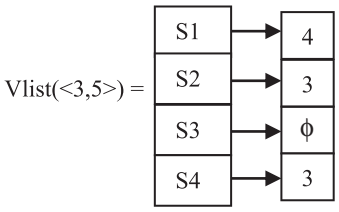
In session S1, page 2 has occurred at position 2 and in session S3 it has occurred at position 1 and 4. In sessions S2 and S4 it has not occurred so it is represented as Φ. Support of 2 is computed by counting the number of nonnull list in Slist (i.e.,) Sup(2) = 2. Similarly Slist of other pages is also computed and is shown below:



Example 3.5. Vertical ID list (Vlist (p)) is used for storing the first occurrence position of the page in respective sessions. For the sequence database given in Example 3.4 the Vlist for the page 2 can be computed as follows:

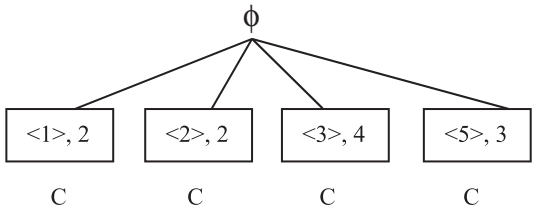


If a sequence consists of n pages Vlist can be computed by taking the first occurrence of the last page (n^{th} page) after $(n-1)^{th}$ page in the sequence. Slist of the pages can be used for computing Vlist. Slist of 3 and 5 are taken for computing the Vlist(<3, 5>). Find the first occurrence of 5 after 3 in all the sessions.



$Slist_s(p)$ represents the position of page p in session s and $Vlist_s(p)$ represents the first occurrence position of page p in session s. $Slist_{s1}(3) = 3$ specifies that the page 3 has occurred at position 3 in session S1 and $Slist_{s1}(5) = 4$ specifies that the page 5 has occurred at position 4 in session S1. As $Slist_{s1}(3) < Slist_{s1}(5)$, place the value of $Slist_{s1}(5)$ in $Vlist_{s1}(<3, 5>)$. Position of 5 in session S2 is $Slist_{s2}(5) = [1, 3]$. $Slist_{s2}(3) = 2$ and first value in $Slist_{s2}(5) = 1$. Here 2> 1 so try to find the next position of 5 such that its position comes after page 3 called as Virtual Shifting. From $Slist_{s2}(5)$ it is found that the next occurrence position of 5 in S2 is 3 which comes after $Slist_{s2}(3)$. So place 3 in $Vlist_{s2}(<3, 5>)$. In S3, $Slist_{s3}(3) = 2$ but $Slist_{s3}(5) = \Phi$. So, place Φ in $Vlist_{s3}(<3, 5>)$. In S4, $Slist_{s4}(3) = 2$ and $Slist_{s4}(5) = 3$ where $Slist_{s4}(3) < Slist_{s4}(5)$. So place $Slist_{s4}(5)$ in $Vlist_{s4}(<3, 5>)$.

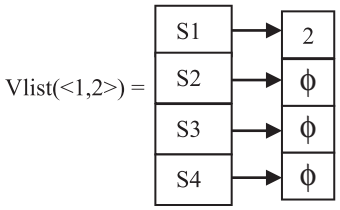
Example 3.6. Closed Sequence Enumeration Tree is used for representing the closed sequences and for enumerating new sequences from existing sequences using depth first search strategy. Vlist is used for checking whether a sequence can be extended are not. Root node is represented by Φ. All sequences of length-1 whose support is greater than msup become first level nodes of the tree. The sequence extension starts from left most nodes in a depth first manner. For the session database S given in Example 3.4, sequences <1>, <2>, <3>, <5> are taken for node extension and sequence <4> is removed from sequence extension as its support is less than msup. Here it is assumed that the msup = 2. Each node contains the sequence and its support



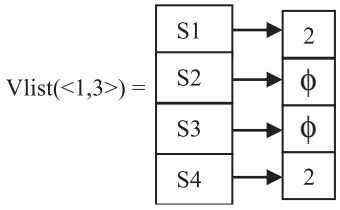
Initially all the nodes are assumed to be closed (C) and during extension they are either marked as nonclosed (NC) or unpromising (U). If a node is labeled as NC then one of its descendant is closed with same support, unpromising means the node has a supersequence with the same support in other branches of the tree. During extension if the support of the new sequence is less than msup it is not added to tree.

Node extension starts with 1. The node is extended by adding the last sequence in the siblings and by adding last sequence in itself. So possible extensions of sequence <1> are {<1, 2>, <1, 3>, <1, 5>, <1, 1>}. First <1, 2> is taken. Vlist<1, 2> is computed from Slist(1) and Slist(2) and support calculated from Vlist(1, 2).

Sup(<1, 2>) < msup, so sequence <1, 2> is not added to the tree. For next extension <1, 3> is taken. Compute Vlist(<1, 3>) using Slist(1) and Slist(3).



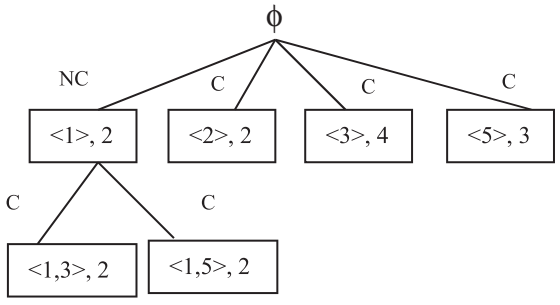
Sup(<1,2>) = 1



Sup(<1,3>) = 2

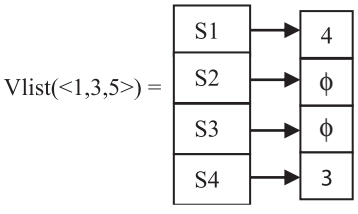
Sup(<1, 3>) is greater than msup. If a supersequence exists in other branches of the tree with same support it is not added. Otherwise it adds a node for the new sequence and checks the following conditions: If any ancestor until Φ exists with the same support, the ancestor node is marked as NC. If any sibling of the ancestor exists whose sequence is a subsequence of the new sequence with same support it is marked as unpromising and that node will not be extended further.

Since there is no supersequence for <1, 3> in the tree with same support, it is added as child of the node with sequence <1> in the tree. Ancestor of <1, 3> is <1>. The support of the ancestor <1> is same as <1, 3>, so it is marked as NC. Sibling of the ancestor <1> whose sequence is subsequence of <1, 3> is <3>. Since Sup(<3>) is not same as Sup(<1, 3>), no changes is done to <3>.



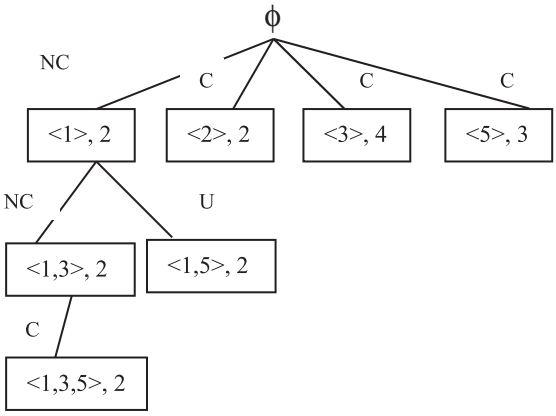
Similarly the Vlist(<1, 5>) and Vlist(<1, 1>) are computed. Sup(<1, 5>) = 2 and Sup(<1, 1>) = 0. Sequence <1, 5> is added to the tree and sequence <1, 1> is not added to the tree.

Now the first child of <1> will be extended in a depth first manner. Possible extension of the sequence <1, 3> are {<1, 3, 5>, <1, 3, 3>}. First <1, 3, 5> is taken. Vlist<1, 3, 5> is

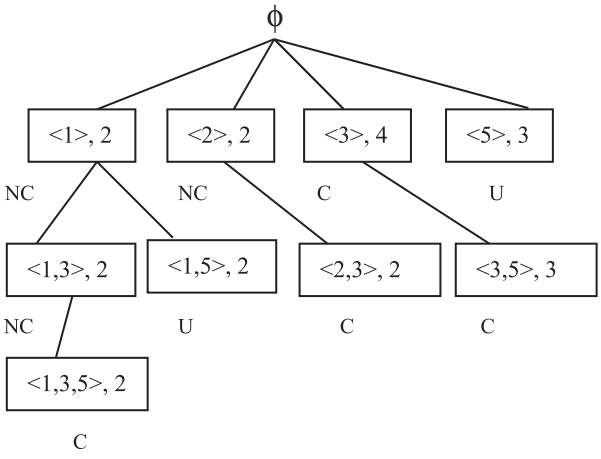


Sup(<1,3,5>) = 2

Sup(<1, 3, 5>) is greater than msup. Ancestors(<1, 3, 5>) with same support = {<1, 3>, <1>}. They are marked as NC. Subsequences(<1, 3, 5>) in the tree = {<1, 5>, <5>}. <1, 5> is having same support as <1, 3, 5>, so it is marked as unpromising and it will not be extended further. <5> is having different support, so no changes will be done.



Sup(<1, 3, 3>) is 0 so it is not added to the tree. Since <1, 3, 5> is not having any sibling it will not be further extended. It moves up one level. <1, 5> is unpromising so it is not extended. In previous level it moves to the node with sequence <2> for further extension. The process is repeated until all the children of root are extended.



The sequences in the node marked as C represents the closed sequential pattern generated. The Closed Sequential Patterns = {<1, 3, 5> Sup:2, <2, 3> Sup:2, <3, 5> Sup:3, <3> Sup:4}. The sessions belonging to individual users can be considered as a session database. For each user the closed sequential pattern mining algorithm is applied and the CFPs are generated.

3.2. User Community Detection

User community detection is the process by which users with similar interest are grouped together based on their navigation patterns. For grouping users with similar interest the similarity measures must be properly defined. As large number of users are identified, to reduce the processing time and to improve the quality of clustering we take into consideration the consistent behavior of the users.

Algorithm 1 *findClosedSequence*

Input: *S - Session Database, msup – minimum support*
Output: *CS - closed sequence tree which contains closed frequent pagesets*

```
CS.root = ϕ
for each unique sequence s ∈ S
s.constructSlist()
s.Sup = computeSupport(s.Slist)
if(s.Sup > msup)
   // create a new node with sequence s
   newNode = CS.createNode(s);
   newNode.label = "closed"
   newNode.Vlist = constructVlist(newNode.Slist)
   //add newNode as child of root in the CS tree
   CS.addChild(root, newNode);
end if
end for
for each n ∈ children(CS, root)
   extendNodeSequence(CS, n, msup)
end for
return CS
```

A user is said to have consistent behavior if he repeatedly visits the same set of pages. In order to identify the consistent user, within user similarity measure is used.

PageSet Similarity (ρ):

For any given two pageset (p_1, p_2) the PageSet Similarity (ρ) can be calculated using Eq. (1) which is simplified version of the equation given in Ref. [19].

$$\rho\left(p_1, p_2\right) = \frac{2 * \left(\left|\mu\left(p_1, p_2\right)\right|\right)}{\left|p_1\right| + \left|p_2\right|} \tag{1}$$

where,

$|p_1|$ - Number of pages in p_1 pageset

$|p_2|$ - Number of pages in p_2 pageset

$\mu\left(p_1, p_2\right)$- Common Pages in sequential order between p_1 and p_2

$\left|\mu\left(p_1, p_2\right)\right|$- number of pages in $\mu(p_1, p_2)$

The value of ρ lies between 0 and 1. A value closer to 1 indicates pageset are closer to each other otherwise not.

Example 3.7. Let p_1 = <2, 3, 4, 6, 7>, p_2 = <3, 4, 7, 8> be the pagesets. $\left|p_1\right| = 5$, $\left|p_2\right| = 4$, $\mu(p_1, p_2)$ = <3, 4, 7>, $\left|\mu\left(p_1, p_2\right)\right| = 3$. Now ρ can be computed as follows:

$$\begin{aligned}\rho\left(p_1, p_2\right) &= (2 * 3) / (5 + 4)\\ &= 0.67\end{aligned}$$

Within User Similarity (γ_k):

The similarity between pageset that belongs to a user can be computed using γ_k. As proposed by Ref. [19] the within user similarity can be computed using Eq. (2).

$$\gamma_k = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n}\rho\left(p_{ik}, p_{jk}\right)}{n^2} \tag{2}$$

Algorithm 2 *extendNodeSequence*

```
Input: CS - closed sequence tree, n - node, msup -
minimum support
if(n.label= "Unpromisin g")
   return
end if
// sequence extension is trying to merge with the
   last sequence of siblings of a node
for each s ∈ sibling(CS, n)
// create a new vlist u sin g vlist of node
// and its sibling
   ns = extendSequence(n, s)
   ns.Vlist = computeVlist(n.Vlist, s.Vlist)
// compute support(Sup) of the new sequence
ns.Sup = computeSupport(ns.Vlist)
if(ns.Sup >= msup)
   newNode = CS.createNode(ns)
   newNode.label = "closed"
   // To check whether the new sequence
// can be added to the tree and for updating the
// labels of other nodes in the tree
   for each node x ∈ CS
      if(x.isAncestor(newNode) and
            x.isSubSequence(newNode) and
            x.Sup = newNode.Sup)
         x.label = "NonClosed"
      else if(x.isSubSequence(newNode) and
               x.Sup = newNode.Sup)
         x.label = "Unpromisin g"
   else if(x.isSuperSequence(newNode) and
                  x.Sup = newNode.Sup)
         newNode.label = "Unpromi sin g"
         // exit for loop
         Break
      end if
   end for
   if(newNode.label = "closed")
   CS.addChild(n, newNode)
   end if
   end if
end for
// now all childs of n will be created.
// for each child of n try to find the
// extension.It is a DFS approach
for each child ∈ children(n)
   extendNodeSequence(CS, child, msup)
end for
```

where, k = 1.. m (number of users)

n- represents the number of pageset/session of user k

p_{ik}, p_{jk} represents i^{th} and j^{th} pageset/session of user k

ρ- similarity between two pagesets

Every pageset visited by the user is compared with his own pagesets. Problem with this approach is if u1 $= \{p_1, p_2, p_3\}$ then pageset similarity is computed between the following pagesets:

(p_1p_1, p_1p_2, p_1p_3, p_2p_1, p_2p_2, p_2p_3, p_3p_1, p_3p_2, p_3p_3) Here $\rho(p_1, p_2) = \rho(p_2, p_1)$ and pageset must not be compared to itself (i.e.,) $\rho(p_1, p_1)$, as it is always 1. To reduce computational complexity and to improve accuracy, a new similarity measure is proposed as given in Eq. (3).

$$\gamma_k = \frac{\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}\rho(p_{ik}, p_{jk})}{\upsilon} \tag{3}$$

where $\upsilon = n!/(2*(n-2)!$ is the number of pageset similarity computed. By the proposed within user similarity, the pageset similarity is computed only for the following pagesets: p_1p_2, p_1p_3 and p_2p_3

The value of γ_k lies between 0 and 1. If user k repeatedly visits the same sequence of pages then the value of γ_k is closer to 1. A user is said to have a consistent behavior, if the γ_k value is above a specific threshold δ, which is set as 0.5.

Example 3.8. Let us consider a user u1 with pagesets PS = (p_1, p_2). If p_1 = <1, 2, 3>, p_2 = <4, 5> then using the Ref. [19] approach

$$\gamma_1 = \frac{\rho(p_1,p_1) + \rho(p_1,p_2) + \rho(p_2,p_2) + \rho(p_2,p_1)}{2*2} = \frac{1+0+1+0}{2*2} = 0.5$$

Though there is no relation between the pagesets p_1 and p_2 the similarity value is 0.5 which is not desirable.

Using our proposed approach,

$$\gamma_1 = \frac{\rho(p_1,p_2)}{1} = \frac{0}{1} = 0$$

The within user similarity value is 0 which specifies there is no relation between the pagesets p_1 and p_2.

Between User Similarity (β):

Once the users is having consistent behavior, they can be clustered together based on between user similarity measures. The similarity between two users can be computed using β as given in Eq. (4). The similarity measure is symmetric as $\beta(U_i, U_j) = \beta(U_j, U_i)$

$$\beta(U_i,U_j)_i = \left(\frac{\sum_{n=1}^{x}\left(avg\left(s(p_{in}), s(\varphi_j^n)\right) * \rho(p_{in}, \varphi_j^n)\right)}{\sum_{n=1}^{x} avg\left(s(p_{in}), s(\varphi_j^n)\right)}\right)$$

$$\beta(U_i,U_j)_j = \left(\frac{\sum_{n=1}^{y}\left(avg\left(s(\varphi_i^n), s(p_{jn})\right) * \rho(\varphi_i^n, p_{jn})\right)}{\sum_{n=1}^{y} avg\left(s(\varphi_i^n), s(p_{jn})\right)}\right)$$

$$\beta(U_i,U_j) = \frac{\beta(U_i,U_j)_i + \beta(U_i,U_j)_j}{2} \tag{4}$$

where,

U_i- i^{th} User

U_j- j^{th} User

$\beta(U_i, U_j)_i$- similarity between U_i and U_j w.r.t U_i

$\beta(U_i, U_j)_j$- similarity between U_i and U_j w.r.t U_j

x- number of pagesets belonging to U_i

y- number of pagesets belonging to U_j

$s(p_{in})$- support of n^{th} pageset of U_i

φ_j^n- best match pageset (which has highest similarity) in User j for the n^{th} pageset of U_i

φ_i^n- best match pageset (which has highest similarity) in User i for the n^{th} pageset of User j

avg- average of the support of the given pagesets

Best match pageset represents the pageset which has highest similarity to the pageset of a user. Let us assume that there two users U1 = ≪2, 3, 2>, <2, 5, 2≫ and U2 = ≪2, 3, 2>, <2, 5, 2≫ and they have visited same set of pages. If the between user similarity is computed without using best match pageset then the similarity value will be 0.65 < 1 [50]. Actually the similarity must be 1 as they have visited the same set of pages. To overcome this problem, best match pageset is used. For the pageset <2, 3, 2> in User1 the best match pageset in User2 is <2, 3, 2>, similarly for <2, 5, 2> in User1 the best match pageset in User2 is <2, 5, 2>. Based on this, the between user similarity value will be 1.

The value of $\beta(U_i, U_j)$ lies between 0 and 1. If the value is closer to 1 then more similar the users are. The dissimilarity matrix (DM) can be computed using 1-β. The computed DM is given as input for clustering process.

Clustering:

The popular clustering techniques are i) Hierarchical, ii) Partitioning and iii) Density-based clustering [51]. Agglomerative Hierarchical Clustering is a bottom-up approach in which it starts with individual user as a cluster and continues to merge the users until a stopping condition is reached. The stopping condition used in our approach is the optimum number of clusters. To identify the optimum number of clusters various linkage metrics such as complete, single, average, weighted are applied. Based on the dendrogram results the cut-off distance is identified. The number of clusters formed by average linkage at the specified cut-off distance is chosen as optimal number of clusters. The drawback of using hierarchical clustering is ambiguity in the stopping condition and once merging is done it cannot be undone.

To overcome it, the Clustering Large Applications Using RANdomized Search (CLARANS) partitioning clustering applied to identify the user community. The limitation in partitioning clustering is choosing the value of "k" [52]. The optimal number of clusters found by hierarchical clustering is set as the value of "k." The advantage of partitioning clustering over hierarchical clustering is it is an iterative approach and the quality of clusters formed is improved in subsequent iterations. CLARANS are lesser sensitive to outlier and it does not depend on order of comparison performed. The problem with this approach is that it identifies only spherical shaped clusters and it is difficult to find arbitrary shaped clusters. The runtime for CLARANS is very high for large datasets.

To identify user community of arbitrary shape a modified form of DBSCAN, a density-based clustering algorithm, user CommunityDetection algorithm (Density-Based Community Detection [DBCD]) is applied. It is used for identifying the community among users and findNeighbors algorithm is used for finding neighbors for a specific user. The density-based clustering algorithm does not require specifying the total number of clusters. It can be used to identify the relation among the data that are difficult to infer manually.

The time complexity is $O(m|\eta|)$ where m is the number of users and $|\eta|$ represents the number of neighbors of m. The space complexity is $O(m^2)$ as the DM must be stored in memory. The parameters required for the algorithm are minN and epsilon (ε). The term minN specifies the minimum number of neighbors required for forming a dense region. The users having minN neighbors are called core users. Core users are the core samples with high density from which the cluster can be expanded. If a user is having minN neighbors it is assumed that they have high density. The symbol ε specifies the minimum of distance between two users to consider them as neighbors. If the core users are within ε distance they are grouped into single cluster. All users directly reachable from the core user or through other core user will be added to the cluster to which the core user belongs to. The set of users accessing the website can be represented as $U = \{u_1, u_2, \ldots, u_m\}$. The web activity of a user in U can be represented by a set of sessions $S = \{s_1, s_2, .., s_l\}$ where each session consists of the sequence of pages visited by the user along with the url of the webpage $s_i = \left((p_1, url_1), (p_2, url_2), (p_3, url_3) \ldots \left(p_p, url_p\right)\right)$. The internal representation of the Users along with their pagesets/sessions is given in Figure 2. The set of users and set of sessions are internally represented by the data structures HashMap.

Figure 3 shows a graph $G = (V, E)$ where V represents a set of m vertices $V = \{u_1, u_2, u_3, \ldots, u_m\}$ and each vertex represents a user, E represents the set of edges where each edge (U_{ij}) represents a link between user u_i and u_j. If $U_{ij} = 1$ the user u_j is within ε distance from u_i where $j = 1, 2, 3..m, i \neq j$. The search for community in G starts with the first user (vertex) and identifies all neighbors of the user within ε distance. A directed edge is drawn from user u_i to u_j, if user u_i is a core user and u_j is within ε distance from u_i. The distance between users u_i and u_j can be retrieved from the distance matrix

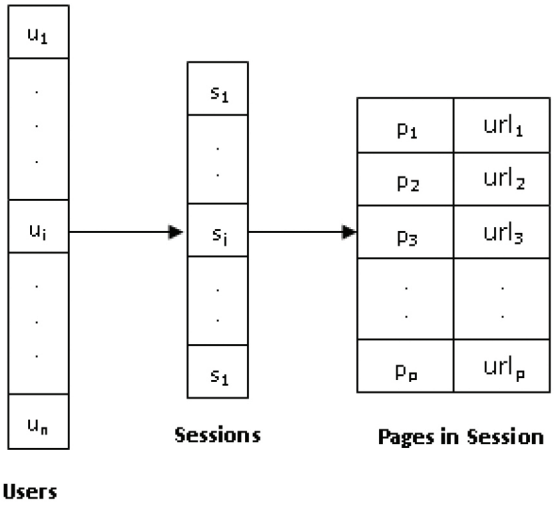


Figure 2 | Internal representation of users and sessions.

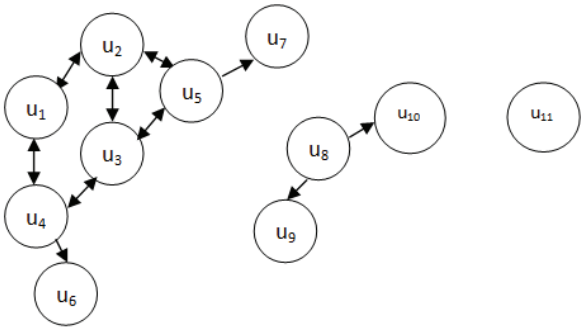


Figure 3 Graph model for the web user community.

$M_{m\times m}$. where $d_{u_i u_j}$ represents the distance between users u_i and u_j.

$$M_{m\times m} = \begin{bmatrix} d_{u_1u_1} & d_{u_1u_2} & d_{u_1u_3} & \cdots & d_{u_1u_m} \\ d_{u_2u_1} & d_{u_2u_2} & d_{u_2u_3} & \cdots & d_{u_2u_m} \\ d_{u_3u_1} & d_{u_3u_2} & d_{u_3u_3} & \cdots & d_{u_3u_m} \\ . & . & . & . & \\ d_{u_mu_1} & d_{u_mu_2} & d_{u_mu_3} & \cdots & d_{u_mu_m} \end{bmatrix}$$

If minN = 3 then users u_1 to u_5 and u_8 are core users as they have atleast 3 neighbors including themselves within ε distance. Users u_6, u_7, u_9, u_{10} and u_{11} are noncore users. As the users u_1 to u_5 are reachable from one another they form a single cluster. Users u_6 and u_7 are reachable from u_1 through other core users so they also belong to cluster of u_1. Similarly users u_8, u_9 and u_{10} form a single cluster with u_8 as core user. As u_{11} is not reachable from other users it is an outlier.

The set of communities formed by the userCommunityDetection algorithm can be represented as $C = \{C_1, C_2, C_3, \ldots C_c\}$, where each community $C_i = [u_1, u_2, u_3, \ldots, u_n]$ is represented by a bitvector. A value of 1 in the bitvector represents the user belongs to the community and 0 represents the user does not belong to the community. For instance, if $C_1 = [1, 0, 1]$ then user u_1, u_3 belongs to cluster C_1 and u_2 belongs to another community.

The quality of clusters formed by different clustering techniques is validated using silhouette coefficient (SC), an intrinsic validity measure which is one of popular methods to assess the quality of the clustering. Since the ground truth labels are not known only the intrinsic clustering index (SC, Calinski-Harabaz Index, Davies Bouldin index) can be used for evaluating the quality of clusters formed. Calinski-Harabaz Index is suitable only if the clusters are spherical and the clustering elements lie in the Euclidean space. Davies Bouldin index involves computation of centroid and is applicable for objects in Euclidean space.

As our work related is non-Euclidean space, we have used SC for cluster evaluation which was found suitable. The scientific contribution of our proposed work lies in the use of within user similarity which reduces the number of users and CFP which reduces the number of pagesets within the user. As number of users and pagesets are reduced, time taken for the clustering is reduced. Since network-based approaches are suitable for applications whose topology is known we have experimented with other clustering approaches such as partitioning-based method (CLARANS), hierarchical clustering method (agglomerative) and DBCD and their results are evaluated and discussed in Section 4.

Algorithm 3 *userCommunityDetection*

Input: *US: HashMap<User, List<Session≫,*
ε- radius, δ- threshold,
minN- minimum number of neighbors,
Output: *UL - HashMap<User, clusterLabel>*
c = 0
for each user u in US
// if γ < δ user is removed from US
if γ (u) < δ
US.remove(u)
end if
end for
for each user u in US
for each user v in US
// β(u, v) is the between user similarity
compute β(u, v) using Eq. (4)
// d(u, v) is the distance between user u and
// user v
$d(u, v) = 1 - \beta(u, v)$
end for
end for
for each user u in US
if(is_set(label(u)) then
continue
end if
$\eta = findNeighbors(US, u, \varepsilon, d)$
//if number of neighbors < min N
// label as noise
if $|\eta| <$ *min N then*
label(u) = noise
continue
end if
c = c + 1
label(u) = c
UL.add(u, label(u))
$N = \eta$
for each neighbor n in N
// if user already clustered
// move to next user
if (is_set(label(n)) and label(n)!=noise) then
continue
end if
$\eta = findNeighbors(US, u, \varepsilon, d)$
label(n) = c
UL.add(n, label(n))
// reachable from core user
if $|\eta| <$ *min N then*
continue
end if
$N = N \cup \eta$
end for
end for
for each user u in US
if(label(u) = noise)
UL.add(u, label(u))
end if
end for

Algorithm 4 *findNeighbors*

Input: *US: HashMap<User, List<Session»,*
m - m^{th} user, ε- radius,
d - dissimilarity matrix
Output: *η- neighbors for the user m*
$\eta = \phi$
for each user u in US
if d(m, u) <= ε then
$\eta = \eta \cup u$
end if
end for
return η

4. RESULTS AND DISCUSSION

The web log entries are acquired from the institution where the authors are working over a period of 3 months. The logs are of webserver extended log format. A total of 1402 pages are in the website. The website contains information about the departments, courses offered, curriculum etc., Initially the web server log was having 538308 records. The number of records after data preprocessing is 230005. After preprocessing the number of unique users is identified based on their IP address, OS and browser. From the preprocessed data 31517 unique users are identified. The pagesets produced by pageset identification process is given as input to the CFP identification process.

4.1. Closed Frequent Pageset (CFP Identification

The classical algorithms for CFP mining such as CloSpan [34], ClaSP [36], CM-ClaSP [37] and CloFAST [38] are implemented and the time taken for computing the CFPs are identified. The runtime of the algorithms by varying the support is given in Figure 4. It is evident from the graph that the runtime of CloFAST algorithm is less compared to other algorithms by the usage of Sparse ID list and vertical ID list.

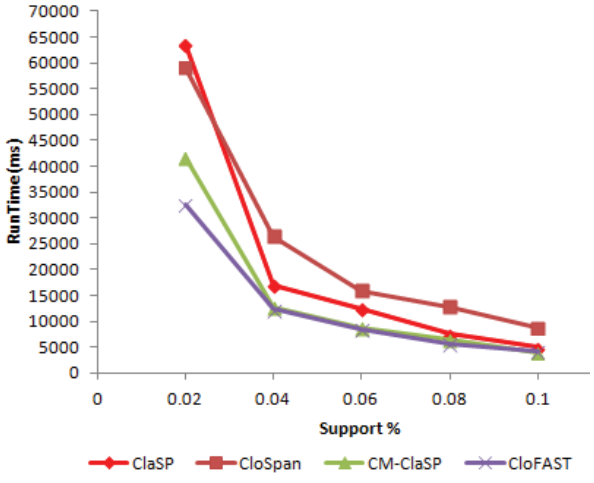


Figure 4 Running time in milliseconds by varying support.

Based on the result, CloFast algorithm is applied for identifying the CFP. The comparison between the PrefixSpan algorithm, used for FP and CloFast algorithm is given in Table 1.

By taking into consideration all the users without computing within user similarity, the total number of frequent patterns is reduced by 64.57% and the time taken for constructing the DM is reduced by 81.59% and memory usage is reduced by 21.21% by using the CloFast sequential pattern mining algorithm. Considering only the users whose within user similarity is greater than 0.5, the total number of frequent patterns is reduced by 68.31% and the time taken for constructing the DM is reduced by 87.85% and memory usage is reduced by 43.61% by using the closed sequential pattern mining algorithm.

4.2. Clustering

The 137 consistent users along with the CFPs identified by CloFast algorithm is given as input to the clustering process. The consistent users can be considered as the data point in the pattern space.

4.2.1. Agglomerative hierarchical clustering

To determine the optimal number of clusters, agglomerative hierarchical clustering is applied on the FPs. Since the dissimilarity between two pagesets is non-Euclidean, distance metrics such as Single, Complex, Average and Weighted are applied on the FPs. The number of clusters formed at different cut-off distance in the range of 0.1 to 1.0 is shown in Figure 5.

The cut-off distance at which majority of the linkage methods coincides is selected for identifying the total number of clusters. The cut-off distance for CFPs generated by CloFast is set as 0.8. Figure 6 shows the number of clusters formed by different linkage methods by setting optimal cut-off distance.

To identify the linkage method suitable for clustering, SC is used. It is an intrinsic method which is used for evaluating the quality of clusters formed. The value of SC lies between −1 to 1. The value closer to 1 indicates that the clusters are compact and are well separated from other clusters. The SC denoted as s(u) for a single user u can be computed using Eq. (5).

$$s(u) = \frac{b(u) - a(u)}{\max(a(u), b(u))} \tag{5}$$

where, a(u) - compactness of the cluster to which user u belongs to.

b(u) - degree to which the user u is separated from other cluster

The SC for various linkage methods is shown in Table 2.

Based on the internal index, Average Linkage method is used for clustering the pageset. The dendrogram obtained by using average linkage agglomerative hierarchical clustering is shown in Figure 7. The horizontal axis represents the number of points in a node within parenthesis or index of the user (without parenthesis) and vertical axis represents the distance between the clusters containing the users.

The optimal number of clusters identified by agglomerative hierarchical clustering algorithm by using average linkage distance metric is 23, which can be set as the value for k.

Table 1 | PrefixSpan versus CloFast.

Metric	Without γ_k [Number of Users:578]		With γ_k [Number of Users:137]	
	PrefixSpan [19]	CloFast (Proposed)	PrefixSpan (Proposed)	CloFast (Proposed)
Number of frequent patterns	2255	**799**	**609**	**193**
Time taken for constructing dissimilarity matrix (in ms)	68652	**12637**	**5012**	**609**
Memory Usage (MB)	128.86	**101.53**	**132.77**	**74.87**

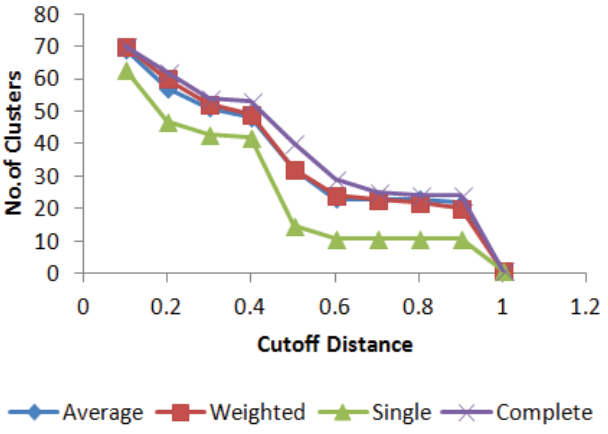


Figure 5 | Number of clusters obtained by varying cut-off distance.

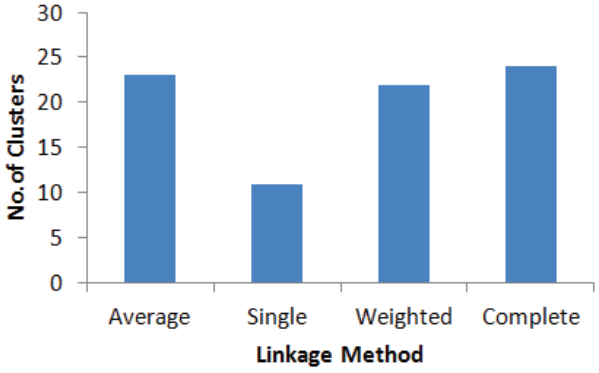


Figure 6 | Number of clusters obtained using linkage methods.

Table 2 | Silhouette coefficient for different linkage methods.

S. No	Linkage Method	Silhouette Coefficient
1	Average	**0.6124**
2	Weighted	0.5821
3	Single	0.1832
4	Complete	0.6035

4.2.2. Clustering Large Applications Using RANdomized Search

It is a medoid-based clustering algorithm, where a graph structure is used for finding the optimal k medoids. Each vertex represents a set of k-medoids. Each vertex has k(n-k) neighbors, where n represents number of users; k represents the number of medoids. Two nodes

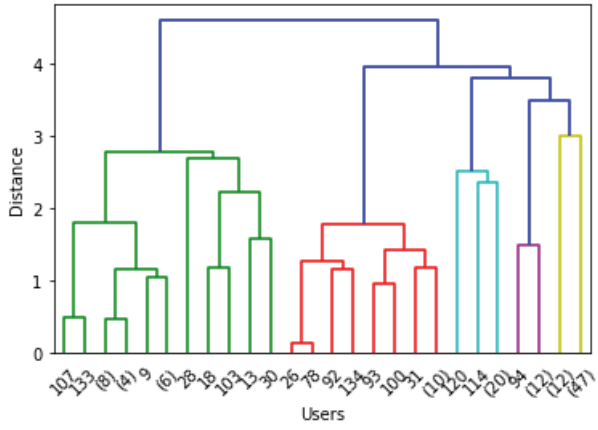


Figure 7 | Agglomerative clustering of users.

Table 3 | Silhouette coefficient for varying number of maxNeighbor.

S. No	maxNeighbor	Silhouette Coefficient
1	100	0.5238
2	150	0.5326
3	200	0.5460
4	250	**0.5924**

are said to be neighbors if they differ by only one user. The objective is to identify the vertex with minimum cost.

The optimal number of clusters identified by hierarchical clustering is used as the value of k. Two important parameters to be set in CLARANS are maxNeighbors and numlocal. The term *maxNeighbors* represents the number of neighbors inspected and *numlocal* represents the number of local minima found. The quality of clusters formed by setting numlocal as 2 and by varying the maxNeighbor is tabulated in Table 3. The silhouette plot of the consistent users clustered using CLARANS is given in Figure 8 which shows the density and separation of the clusters formed. The red line represents the average silhouette score.

4.2.3. Density-Based Community Detection

The parameters required for DBCD are minN and epsilon (ε). minN is set as 3, so that a user has at least 3 neighbors within ε distance to make it suitable for recommendation. The quality of clusters formed by setting minN = 3 and varying the value of ε is tabulated in Table 4.

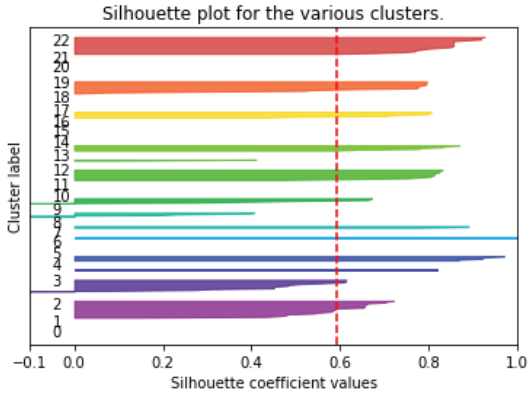


Figure 8 | Clusters formed using Clustering Large Applications Using RANdomized Search (CLARANS).

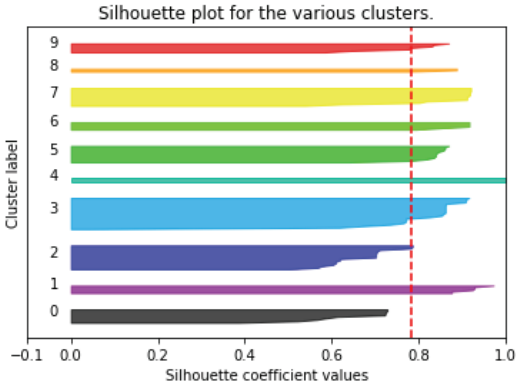


Figure 9 | Clusters formed using Density-Based Community Detection (DBCD).

Table 4 | Silhouette coefficient for different values of ε using closed frequent pageset and within user similarity (Number of users: 137).

S. No	ε	Number of Clusters	Number of Users Clustered	Silhouette Coefficient
1	0.1	11	75	0.8733
2	0.2	9	93	0.8040
3	0.3	10	99	**0.7830**
4	0.4	10	101	0.7471
5	0.5	3	123	0.2192

If ε value is too small then large part of the users are not clustered and are considered as outliers. If the value chosen is too large then clusters will be merged together and the quality of clusters formed will not be good. So the value of 0.3 is chosen for clustering the users.

The silhouette plot of the consistent users clustered using DBCD is given in Figure 9. It shows the density and separation of the clusters formed. The red line represents the average silhouette score. The cluster number −1 represents the outliers. Out of 137 users 98 users are clustered and the remaining are the outliers. It is interesting to learn web usage behavior by probing into the history of the user's sessions, which enables recommendation of the next page to be popped. Recent literatures reveal that many of the researchers focus on only FPs and clustering has been completed to predict the web navigation behavior of the unknown users.

4.2.4. Performance Evaluation

The results of clustering the users having FP and CFP using Hierarchical agglomerative, CLARANS and DBCD algorithms by taking into consideration within user similarity (137 users) and without within user similarity (578 Users) is shown in Figures 10–15. Based on the SC it is found that the quality of the clusters formed without using within user similarity is poorer compared to clustering users using within user similarity measure. Within user similarity enables to identify the consistent behavior of the users. Consistent behavior shows the interest of the users on the pages visited. Clustering users who are not having consistent behavior cannot be used for applications like recommendation as it does not reflect the actual interest of the users on the pages visited.

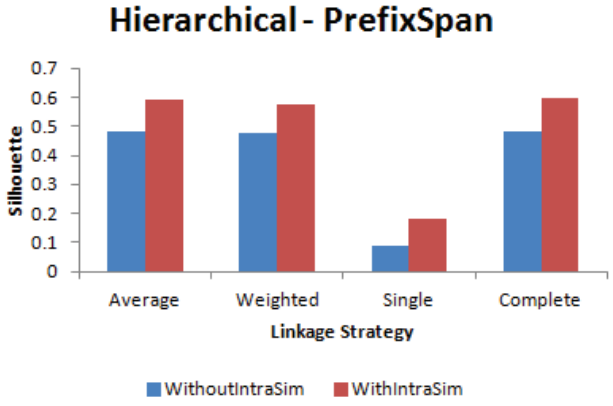


Figure 10 | Hierarchical clustering using frequent pagesets (FP).

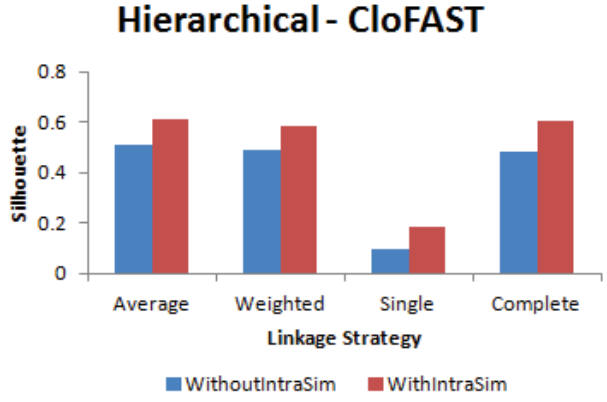


Figure 11 | Hierarchical clustering using closed frequent pagesets (CFP).

An analysis based on clustering the users having FP and CFP, selected using within user similarity is performed and the results are shown in Figures 16–18. From the analysis based on the SC it is found that the quality of the clusters formed using CFP is good compared to FP. There is only a marginal difference in the quality of the clusters formed, but in terms of memory usage and time taken for clustering, CFP is advantageous compared to FP.

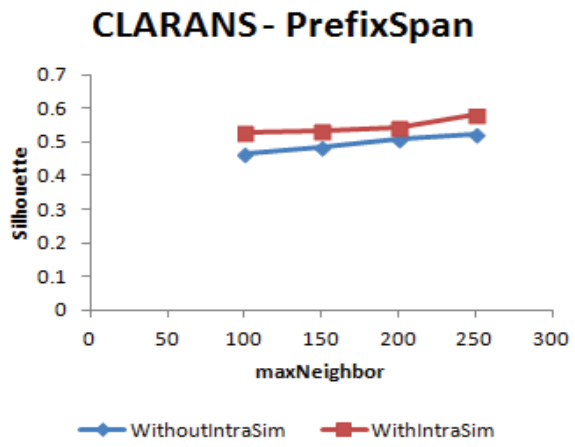


Figure 12 | Clustering Large Applications Using RANdomized Search (CLARANS) clustering using frequent pagesets (FP).

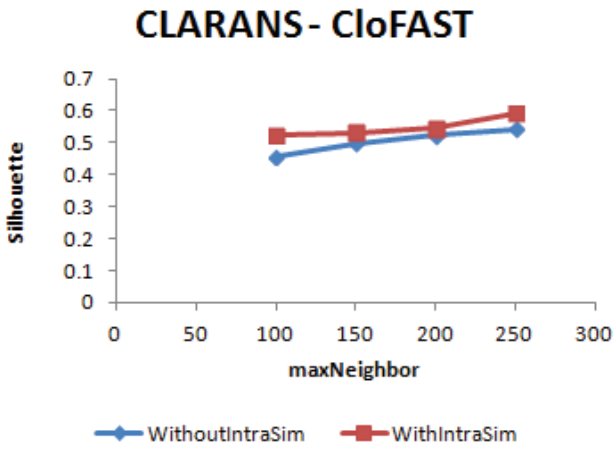


Figure 13 | Clustering Large Applications Using RANdomized Search (CLARANS) clustering using closed frequent pagesets (CFP).

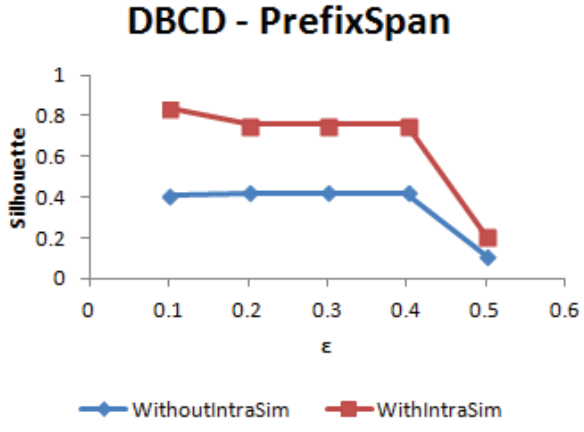


Figure 14 | Density-Based Community Detection (DBCD) clustering using frequent pagesets (FP).

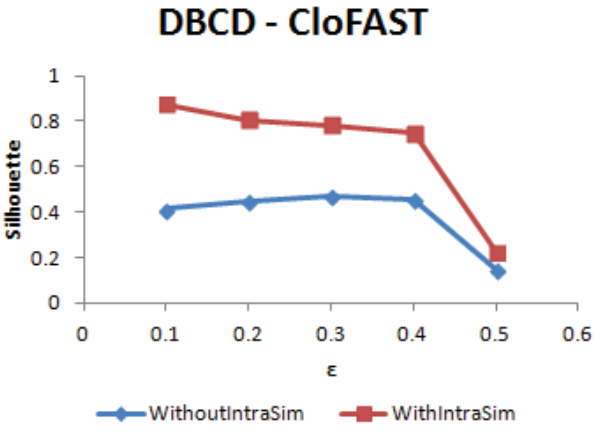


Figure 15 | Density-Based Community Detection (DBCD) clustering using closed frequent pagesets (CFP).

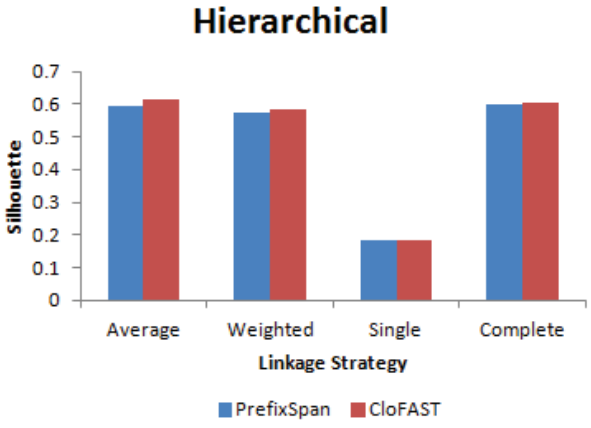


Figure 16 | Hierarchical clustering of users.

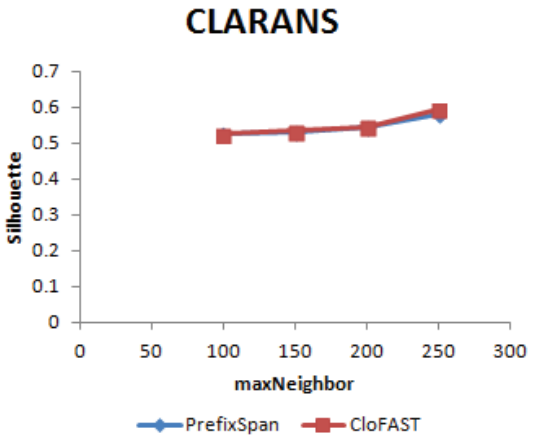


Figure 17 | Clustering Large Applications Using RANdomized Search (CLARANS) clustering of users.

The proposed work follows three different approaches: (i) Without considering the consistent navigation of the users and using the CFPs, (ii) Considering the consistent behavior of the users and using the FPs, (iii) Considering the consistent behavior of the users

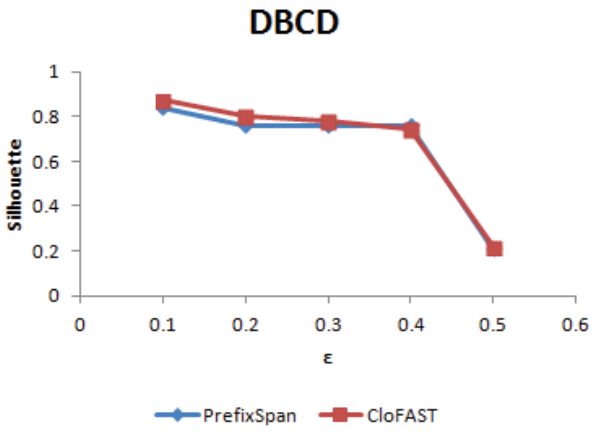


Figure 18 | Density-Based Community Detection (DBCD) clustering of users.

and using the CFPs. We observed that all our approaches are providing improved results in terms of metrics such as reduced number of frequent patterns, minimized time taken and memory usage.

5. CONCLUSION AND FUTURE WORK

Our proposed work started with the collection of web pages visited by the users from web log of the web server and the data are preprocessed. The web server log with extended log format is used for our experimental purposes. The user identification is done and the pagesets for the users are separated. By filtering process the FPs are extracted using the support of the pageset. The pagesets which are not having the superset with the same support are selected for the individual user. The pageset similarity is found within users using our proposed within user similarity metric for identifying the users with consistent navigation behavior. The clustering of the consistent users is done using the between user similarity. Each user is represented by a node in a graph and the link between nodes represents the closeness of the users. The pageset associated with the session for the users are internally represented by Hash Map data structure. Most of the existing works have used the FP with prefix span algorithm leading to clustering. However, the CFP approaches have not been found for clustering the web pageset, which is the novelty of the proposed work. The consistent users are subjected to the clustering techniques such as agglomerative, CLARANS and DBCD. The results observed by DBCD method shows promising one using the clustering metric SC. The application of our work will be more suitable for recommendation model. Any new user can be categorized into an appropriate group based on the similarity among the pagesets and the top "n" pages can be recommended to the user. Some of the applications for the proposed work will be to get recommendation in online shopping, e-commerce applications such as customization, web personalization, provision of discount for items based on interest of users and user profiling.

The usefulness of consistent user pruning could be experimentally evaluated by performing the following experiment: Implement the recommendation system and provide the results to the users of the pages. Then the number of clicks on that recommended pages could be measured and the performance can be compared with different recommendation algorithms such as those based on consistent users, FPs and random recommendations.

The proposed methodology can be applied on bigger data provided the system with high configuration and Graphics Processing Unit (GPU) for parallel processing. The work can also be extended to reduce the time taken to cluster the users preferably by exploring the parallel processing concepts. The drawback of our approach is, as the clustering process is iterative and when a new data point comes it has to be clustered once again to reflect the changes which are a time consuming process. As the clustering process is nondeterministic polynomial in nature, it is possible to compare the proposed work with other methods in an appropriate manner.

CONFLICTS OF INTEREST

The authors wish to confirm that there are no known conflicts of interest associated with this publication . The authors also confirm that the manuscript has been read and approved by all named authors and that there are no other persons who satisfied the criteria for authorship but are not listed.

AUTHORS' CONTRIBUTIONS

All authors of this research paper have directly participated in the planning, execution, analysis of this study and approved the final version.

Funding Statement

This research has no significant financial support that could have influenced its outcome.

ACKNOWLEDGMENTS

The authors wish to express their sincere thanks to the Management of Mepco Schlenk Engineering College,Sivakasi(Autonomous), Tamilnadu, India for their support in carrying out this research work.The authors also thank the reviewers for their valuable suggestions in improving the quality of the research article.

REFERENCES

[1] F.M. Facca, P.L. Lanzi, Mining interesting knowledge from weblogs: a survey, Data Knowl. Eng. 53 (2005), 225–241.
[2] N. Aryabarzan, B. Minaei-bidgoli, M. Teshnehlab, negFIN: an efficient algorithm for fast mining frequent itemsets, Expert Syst. Appl. 105 (2018), 129–143.
[3] Z. Deng, S. Lv, Fast mining frequent itemsets using Nodesets, Expert Syst. Appl. 41 (2014), 4505–4512.
[4] I. Feddaoui, F. Felhi, J. Akaichi, EXTRACT: new extraction algorithm of association rules from frequent itemsets, in 2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), San Francisco, CA, USA, 2016, pp. 752–756.
[5] M.J. Zaki, Scalable algorithms for association mining, IEEE Transactions on Knowledge and Data Engineering, 12 (2000), 372–390.
[6] Y.W.T. Pramono, Suhardi, Anomaly-based intrusion detection and prevention system on website usage using rule-growth sequential pattern analysis, Case study: Statistics of Indonesia

(BPS) website, in IEEE International Conference of Advanced Informatics: Concept, Theory and Application (ICAICTA), Bandung, Indonesia, 2014, pp. 203–208.
[7] J.W. Han, J. Pei, X.F. Yan, From sequential pattern mining to structured pattern mining: a pattern-growth approach, J. Comput. Sci. Technol. 19 (2004), 257–279.
[8] M. Salehi, I. Nakhai Kamalabadi, M.B. Ghaznavi Ghoushchi, Personalized recommendation of learning material using sequential pattern mining and attribute based collaborative filtering, Educ. Inf. Technol. 19 (2014), 713–735.
[9] R. Srikant, R. Agrawal, Mining quantitative association rules in large relational tables, Proceedings of the 1996 ACM SIGMOD international conference on Management of data (SIGMOD '96), Association for Computing Machinery, New York, NY, USA, (1996), 1–12.
[10] M.J. Zaki, SPADE: an efficient algorithm for mining frequent sequences, Machine Learning, 42 (2001), 31–60.
[11] J. Ayres, T. Yiu, J. Gehrke, J. Flannick, Sequential pattern mining using a bitmap representation, in The Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD '02), ACM, Edmonton, Canada, 2002.
[12] Q. Chen, M.-C. Hsu, J. Pei, J. Wang, B. Mortazavi-Asl, J. Han, H. Pinto, Mining sequential patterns by pattern-growth: the PrefixSpan approach, IEEE Trans. Knowl. Data Eng. 16 (2004), 1424–1440.
[13] Y. Chen, M. Chiang, M. Ko, Discovering time-interval sequential patterns in sequence databases, Expert Syst. Appl. 25 (2003), 343–354.
[14] P. Pons, M. Latapy, Computing communities in large networks using random walks, J. Graph Algorithms Appl. 10 (2006), 191–218.
[15] A. Biswas, B. Biswas, Investigating community structure in perspective of ego network, Expert Syst. Appl. 42 (2015), 6913–6934.
[16] L. Duan, Y. Liu, W.N. Street, H. Lu, Utilizing advances in correlation analysis for community structure detection, Expert Syst. Appl. 84 (2017), 74–91.
[17] Y. Xin, J. Yang, Z. Xie, A semantic overlapping community detection algorithm based on field sampling, Expert Syst. Appl. 42 (2015), 366–375.
[18] B. Cai, H. Wang, H. Zheng, H.U.I. Wang, Evaluation repeated random walks in community detection of social networks, in Ninth International Conference on Machine Learning and Cybernetics, Qingdao, China, 2010, pp. 11–14.
[19] Z. Shou, X. Di, Similarity analysis of frequent sequential activity pattern mining, Transport. Res. Part C Emerg. Technol. 96 (2018), 122–143.
[20] P. De Meo, E. Ferrara, G. Fiumara, A. Provetti, Journal of Computer and System Sciences Mixing local and global information for community detection in large networks, J. Comput. Syst. Sci. 80 (2014), 72–87.
[21] C. Remy, B. Rym, L. Matthieu, Tracking Bitcoin users activity using community detection on a network of weak signals, in: C. Cherifi, H. Cherifi, M. Karsai, M. Musolesi (Eds.), Commplex Networks & Their Applications, VI, COMPLEX NETWORKS 2017, Studies in Computational Intelligence Springer, Cham, 689 (2018), pp. 166–177.
[22] E. Cohen, B. Krishnamurthy, J. Rexford, Improving end-to-end performance of the Web using server volumes and proxy filters, in Proceedings of the ACM SIGCOMM'98 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM'98), Association for Computing Machinery, New York, NY, USA, 1998, pp. 241–253.
[23] L.D. Catledge, J.E. Pitkow, Characterizing browsing strategies in the world-wide web, Comput. Netw. ISDN Syst. 27 (1995), 1065–1073.
[24] K.D. Fenstermacher, M. Ginsburg, Mining client-side activity for personalization, Proceedings Fourth IEEE International Workshop on Advanced Issues of E-Commerce and Web-Based Information Systems (WECWIS 2002), Newport Beach, CA, USA, 2002, pp. 205–212.
[25] R. Cooley, B. Mobasher, J. Srivastava, Data preparation for mining World Wide Web browsing patterns, Knowl. Inf. Syst. 1 (1999), 5–32.
[26] R.S. Larsen, J. Andersen, J. Skyt, A.H. Jensen, A. Giversen, T.B. Pedersen, Analyzing clickstreams using subsessions, in Proceedings of the 3rd ACM International Workshop on Data Warehousing and OLAP(DOLAP '00), Association for Computing Machinery, New York, NY, USA, 2000, pp. 25–32.
[27] M. Gery, H. Haddad, (2004). Evaluation of web usage mining approaches for user's next request prediction, in Proceedings of the 5th ACM International Workshop on Web Information and Data Management (WIDM '03), Association for Computing Machinery, New York, NY, USA, 2003, pp. 74–81.
[28] R. Ivancsy, S. Juhász, Analysis of web user identification methods, World Acad. Sci. Eng. Technol. 2 (2007), 338–345.
[29] M. Chen, A.S. LaPaugh, J.P. Singh, Predicting category accesses for a user in a structured information space, in Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '02), Association for Computing Machinery, New York, NY, USA, 2002, pp. 65–72.
[30] A. Hemapriya, M.S. Bhuvaneswari, K. Muneeswaran, Reconstruction of user sessions from Web server log, in Proceedings of the International Conference on Computing and Communications Technologies (ICCCT 2015), Chennai, India, 2015, pp. 37–42.
[31] M.-S. Chen, J.S. Park, P.S. Yu, Data mining for path traversal patterns in a web environment, in IEEE Proceedings of the 16th International Conference on Distributed Computing Systems, Hong Kong, 1996, pp. 385–392.
[32] Z. Peng, M.S. Zhao, Session identification algorithm for web log mining, in 2010 IEEE International Conference on Management and Service Science (MASS 2010), Wuhan, China, 2010, pp. 1–4.
[33] M. Munk, J. Kapusta, P. Švec, Data preprocessing evaluation for web log mining: reconstruction of activities of a web visitor, Procedia Comput. Sci. 1 (2010), 2273–2280.
[34] X. Yan, J. Han, R. Afshar, CloSpan: mining closed sequential patterns in large datasets, in Proceedings of the 2003 SIAM International Conference on Data Mining, San Francisco, CA, USA, 2003, pp. 166–177.
[35] J. Wang, J. Han, C. Li, Frequent closed sequence mining without candidate maintenance, IEEE Transaction on Knowledge and Data Engineering, 19 (2007), pp. 1042–1056.
[36] A. Gomariz, M. Campos, R. Marin, B. Goethals, ClaSP: an efficient algorithm for mining frequent closed sequences, in Advances in Knowledge Discovery and Data Mining: 17th Pacific-Asia Conference (PAKDD 2013), Lecture Notes in Computer Science, Berlin, Heidelberg, 2013, pp. 50–61.
[37] P. Fournier-Viger, A. Gomariz, M. Campos, R. Thomas, Fast vertical mining of sequential patterns using co-occurrence

information, Advances in Knowledge Discovery and Data Mining: 18th Pacific-Asia Conference (PAKDD 2014), Springer, Cham, 2014, pp. 40–52.

[38] F. Fumarola, P.F. Lanotte, M. Ceci, D. Malerba, CloFAST: closed sequential pattern mining using sparse and vertical id-lists, Knowl. Inf. Syst. 48 (2016), 429–463.

[39] E.-H. Han, G. Karypis, V. Kumar, B. Mobasher, Clustering based on association rule hypergraphs, Machine Learning, 1997, pp. 9–13. http://www-users.cs.umn.edu/~kumar/papers/cluster-hyper.ps

[40] T. Morzy, M. Wojciechowski, M. Zakrzewicz, Web users clustering, in Proceeding of the 15th International Symposium on Computer and Information Sciences, Istanbul, Turkey, 2000, pp. 374–382. https://www.researchgate.net/profile/Tadeusz_Morzy/publication/228611562_Web_users_clustering/links/02bfe50e40a22f0cd4000000.pdf%0Ahttp://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.139.3503&rep=rep1&type=pdf

[41] L. Yanxiang, G. Deke, C. Fei, C. Honghui, User-based clustering with top-N recommendation on Cold-Start problem, in Proceedings of the 2013 Third International Conference on Intelligent System Design and Engineering Applications (ISDEA 2013), Hong Kong, 2013, pp. 1585–1589.

[42] D.S. Anupama, S.D. Gowda, Clustering of web user sessions to maintain occurrence of sequence in navigation pattern, Procedia Comput. Sci. 58 (2015), 558–564.

[43] L. Bai, X. Cheng, J. Liang, Y. Guo, Fast graph clustering with a new description model for community detection, Inf. Sci. 388–389 (2017), 37–47.

[44] P. Kumar, Sequence clustering approach for clustering web user session, Int. J. Bus. Inf. Syst. 28 (2018), 67–78.

[45] N.B. Ahmad, U.F. Alias, N. Mohamad, N. Yusof, Map clustering for student browsing behaviour analysis, Procedia Comput. Sci. 163 (2019), 550–559.

[46] Z. Liu, Y. Ma, A divide and agglomerate algorithm for community detection in social networks, Inf. Sci. 482 (2019), 321–333.

[47] J. Castro Gertrudes, A. Zimek, J. Sander, R.J.G.B. Campello, A unified view of density-based methods for semi-supervised clustering and classification, Data Mining Knowl. Discov. 33 (2019), 1894–1952.

[48] E. Ntoutsi, K. Stefanidis, K. Nørvåg, H.P. Kriegel, fast group recommendations by applying user clustering, in: Atzeni P., Cheung D., Ram S. (Eds.), Conceptual modeling. ER 2012, lecture notes in computer science, Springer, Berlin, Heidelberg, 7532 (2012), pp. 126–140.

[49] T. Arce, P.E. Román, J. Velásquez, V. Parada, Identifying web sessions with simulated annealing, Expert Syst. Appl. 41 (2014), 1593–1600.

[50] J.J.C. Ying, E.H.C. Lu, W.C. Lee, T.C. Weng, V.S. Tseng, Mining user similarity from semantic trajectories, in Proceedings of the 2nd ACM SIGSPATIAL International Workshop on Location Based Social Networks, LBSN 2010 Association for Computing Machinery, New York, NY, USA, 2010, pp.19–26.

[51] M. Ester, H.-P. Kriegel, S. Jorg, X. Xu, A density-based clustering algorithms for discovering clusters, KDD-96 Proc. 96 (1996), 226–231.

[52] R.T. Ng, J. Han, CLARANS: a method for clustering objects for, IEEE Trans. Knowl. Data Eng. 14 (2002), 1003–1016.