MONIC - Modeling and Monitoring Cluster Transitions

Myra Spiliopoulou
University of Magdeburg
Magdeburg, Germany
myra@iti.cs.uni-magdeburg.de

Irene Ntoutsi
University of Piraeus
Piraeus, Greece
ntoutsi@unipi.gr

Yannis Theodoridis
University of Piraeus
Piraeus, Greece
ytheod@unipi.gr

Rene Schult
University of Magdeburg
Magdeburg, Germany
schult@iti.cs.uni-magdeburg.de

ABSTRACT

There is much recent work on detecting and tracking change in clusters, often based on the study of the spatiotemporal properties of a cluster. For the many applications where cluster change is relevant, among them customer relationship management, fraud detection and marketing, it is also necessary to provide insights about the nature of cluster change: Is a cluster corresponding to a group of customers simply disappearing or are its members migrating to other clusters? Is a new emerging cluster reflecting a new target group of customers or does it rather consist of existing customers whose preferences shift? To answer such questions, we propose the framework MONIC for modeling and tracking of cluster transitions. Our cluster transition model encompasses changes that involve more than one cluster, thus allowing for insights on cluster change in the whole clustering. Our transition tracking mechanism is not based on the topological properties of clusters, which are only available for some types of clustering, but on the contents of the underlying data stream. We present our first results on monitoring cluster transitions over the ACM digital library.

Categories and Subject Descriptors: H.2.8 [Database Management]: Database Applications – Data Mining; I.5.3 [Pattern Recognition]: Clustering

General Terms: Algorithms, Theory

Keywords: Cluster change detection, cluster transitions, data streams, clusters, temporal analysis

1. INTRODUCTION

Clusters upon the data of many real applications are affected by changes in the underlying population of customer transactions, user activities, network accesses or documents. Much research has been devoted in *adapting* the clusters to the changed population. Recently, research has expanded to encompass *tracing and understanding* of the changes themselves, as means of gaining insights on the population and supporting strategic decisions.

In Fig. 1, we depict clusters at four timepoints; they might be user profiles or topics in news. New records are marked with darker points; records older than two timepoints are removed. We can easily see the clusters at each timepoint and, also, that changes have occurred. However, finding the same cluster again, *categorizing and tracing* the changes upon it is much more challenging: "Did some clusters disappear? Or were they rather absorbed by others? When is a cluster the same and when does it mutate?" We propose MONIC for the categorization and tracing of such cluster changes.

MONIC takes as input an accummulating dataset, whose records are ageing, as is typical in data stream applications. The records are clustered at consecutive time points and their evolution is monitored. To this purpose, we propose a typification of *cluster transitions* and a transition detection algorithm. From the detected transitions we draw conclusions on the lifetime and stability of clusters.

In Section 2, we discuss relevant research. In Section 3, we introduce a cluster typification and then specify the notions of overlap and matching for clusters derived from different slots of the dataset. Section 4 contains our cluster transition model and transition detection heuristics. In Section 5 we present our first experiments. We conclude our study with a summary and outlook.

2. RELATED WORK

The change detection framework FOCUS [5] compares two datasets and computes a deviation measure between them, based on the data mining models they induce. Clusters are a special case of models, namely non-overlapping regions described through a set of attributes (structure component) and corresponding to a set of raw data (measure component). However, understanding how a cluster has evolved inside a new clustering is beyond the scope of FOCUS.

PANDA [3] proposes methods for the comparison of simple patterns and aggregation logics for the comparison of complex ones. PANDA supports cluster comparison across the time axis but concentrates on the generic and efficient realization of comparisons rather than the detection and interpretation of transitions. Finally, the "Pattern Monitor" (PAM) [2] models patterns as temporal objects and captures their evolution, focusing mainly on association rules. MONIC builds upon the insights of PANDA and PAM.

In spatiotemporal clustering, a cluster is a "densification" in a time-invariant trajectory. Yang et al. detect "formation" and "dissipation" events upon clusters of spatial scientific data [11]. Aggarwal models clusters with kernel functions and changes as kernel density changes at each spatial

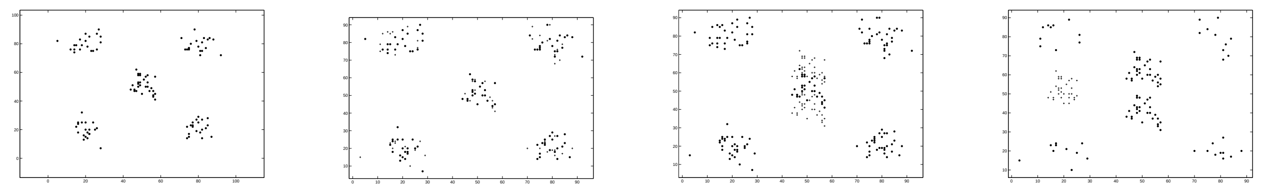


Figure 1: Data records at four timepoints

location [1]; he considers different types of change, with emphasis on computing change "velocity" and finding the locations with the highest velocity. Such methods assume that the trajectory does not change. Thus, they cannot be used if the feature space changes, e.g. in text stream mining, where features are usually frequent words. Further, hierarchical clusterers cannot be coupled with such a method, as they use ultra-metrics, i.e. a dataset defines its own trajectory.

Kalnis et al propose a special type of cluster change, the "moving cluster", whose contents may change while its density function remains the same during its lifetime [6]. They find moving clusters by tracing common data records between clusters of consecutive timepoints. MONIC is more general, since it encompasses several cluster transition types, allows for the "ageing" of old objects and does not require that the density function of a moving cluster is invariant.

Cluster change detection is also relevant for "topic evolution" in text streams, as dealt with in [8, 7], where a "topic" is a cluster label, consisting of the dominant words inside the cluster. In [8], the emphasis is on adapting the clusters, while in [7] a topic evolution graph is built and used to trace topic transitions, i.e. changes in the cluster labels rather than the cluster themselves. These methods are applicable whenever a human-understandable cluster label can be extracted and traced. Cluster labeling is not feasible for all applications though. For this reason, MONIC detects cluster transitions rather than cluster label transitions.

3. CLUSTER MODEL IN MONIC

MONIC models and traces cluster transitions upon data that are collected and clustered at timepoints $t_1, \ldots, t_n$. A "data ageing" function may assign lower weights to old records. The feature space may also change. MONIC assumes re-clustering rather than cluster adaptation at each timepoint, so that both changes in existing clusters and new clusters can be monitored. Moreover, transitions can be detected even when the underlying feature space changes, i.e. when cluster adaptation is not possible. To do so, we first specify the notion of "same" cluster or rather cluster "match" across the time axis.

3.1 Clusterings upon Ageing Data

We observe clustering as a partitioning of the dataset into homogeneous groups. We separate the cluster construction algorithm from the data ageing function that specifies the weights of the records processed by the algorithm.

Definition 1. *A "clustering" ζ is a partitioning of dataset D into partitions/clusters $X_1, \ldots, X_k$ such that (a) $\forall u \neq w$: $X_u \cap X_w = \emptyset$, (b) $\cup_{u=1}^{k} X_u = D$ and (c) some optimization criterion is satisfied, e.g. the members of each cluster are more similar to each other than to other data records.*

This is a set-theoretic definition of clusters. More elaborate definitions are possible for clusters over a metric space, whose topology can be used to specify proximity of objects. As mentioned in section 2, MONIC is intended for arbitrary clustering methods. We therefore observe clusters as sets.

Def. 1 assumes a complete partitioning of the dataset. Clusterers that ignore outliers are indirectly covered by assuming a preprocessing step that removes outliers.

Definition 2. *Let $t_1 \ldots, t_n$ be the sequence of timepoints under observation and let $D_i, i = 2 \ldots, n$ be the set of data records accumulated from t_{i-1} until t_i, while D_1 is the initial dataset, so that $D_i \cap D_j = \emptyset$ for $i \neq j$. A "data ageing function" assigns a weight $age(x, t_i) \in [0, 1]$ to data record x at t_i for each $x \in \cup_{l=1}^{i} D_l$ and for each t_i.*

The weights assigned by the ageing function determine the impact of a record upon clustering $\zeta_i \equiv \zeta_i(\cup_{l=1}^{i} D_l, age, t_i)$. This function covers sliding windows (the weights of records outside the window are zero) but also more elaborate schemes.

3.2 Cluster Matching

Consider a cluster $X \in \zeta_i$ discovered at a timepoint t_i. A "cluster transition" is a change seen upon this cluster at a later timepoint t_j. To detect such a transition, we must first find the cluster X in the clustering ζ_j of t_j – if X is still existent. We first define the (non-symmetric) "overlap" between clusters and then the (best) match for a cluster.

Definition 3 (Cluster overlap). *Let ζ_i, ζ_j $(i < j)$ be the clusterings at t_i, t_j, and let $X \in \zeta_i, Y \in \zeta_j$ be two clusters. The "overlap of X to Y" is the normalized sum of weights of the records in their set intersection:*

$$overlap(X, Y) = \frac{\sum_{a \in X \cap Y} age(a, t_j)}{\sum_{x \in X} age(x, t_j)}$$

Definition 4 (Cluster match). *Let X be a cluster in the clustering ζ_i at t_i and Y be a cluster in ζ_j at $t_j > t_i$. Further, let $\tau \equiv \tau_{match} \in [0.5, 1]$ be a threshold value. Y is "a match for X in ζ_j subject to τ", i.e. $Y = match_\tau(X, \zeta_j)$ if and only if $Y \in \zeta_j$ is the cluster with the maximum overlap for X and the overlap of X to Y is at least τ:*

$$overlap(X, Y) = \max_{U \in \zeta_j}\{overlap(X, U)\} \geq \tau$$

If there is no such $Y \in \zeta_j$, then $match_\tau(X, \zeta_j) = \emptyset$.

By Def. 4, ζ_j can contain at most one match for each cluster in ζ_i, although the same cluster in ζ_j can be the match of more than one clusters in ζ_i. We restrict the threshold τ to the interval $[0.5, 1]$ to stress that a cluster is a match only if it contains at least half of the pivot cluster (e.g. half of its members, if the members are weighted equally). A tie can then occur only for $\tau = 0.5$. Different tie breakers can be used, choosing e.g. the Y that has the maximum reverse overlap $overlap(Y, X)$ or the Y that is closest to X in size.

4. CLUSTER TRANSITIONS IN MONIC

In MONIC, a *cluster transition* at a given timepoint is a change experienced by a cluster that has been discovered at an earlier timepoint. Such a transition may concern the content and form of the cluster, i.e. be "internal" to it, or it may concern its relationship to the rest of the clustering, i.e. be an "external transition". We define these types of transitions and introduce heuristics that trace them.

4.1 Detection of External Transitions

The "external transitions" of cluster $X \in \zeta_i$ with respect to ζ_j at $t_j > t_i$ are defined in Table 1. A cluster $X \in \zeta_i$ survives in ζ_j if (a) there is a match for it in ζ_j subject to τ and (b) this match does not cover any further cluster of ζ_i. If the match covers at least one further cluster in ζ_i, then X has been absorbed. If no match exists, then a split may have occurred: The contents of X are in more than one clusters of ζ_j. Then, the overlaps must be no less than τ_{split} (obviously: $\tau_{split} < \tau$), to prevent degenerate cases. Moreover, all those clusters together must form a match for X. If none of these cases occur, then X has disappeared.

All but the last transitions in Table 1 refer to changes of a *given cluster*. Emerging clusters are detected after tracing all external transitions for each cluster in ζ_i: They are the clusters in ζ_j that are not the result of external transitions.

In Fig. 2 we present our transition detector. For the clusters in clustering ζ_i of t_i (ζ_i in the Figure), it detects their external transitions on the clustering $\zeta\_j \equiv \zeta_j$ of $t_j > t_i$.

For each cluster $X \in \zeta_i$, the detector performs some initializations and then computes the overlap of X to each cluster of ζ_j (line 5). In line 7, the best match for X is selected, according to some tie breaking criterion as discussed after Def. 4. So, each cluster in ζ_i has at most one survival candidate. If X has none, clusters overlapping with it for more than τ_{split} are found (lines 10–12). If neither exist, then X is marked as disappeared (lines 15–16).

Split detection involves building a list of split candidate clusters (line 11), which must form a match for cluster X when taken together (cf. Table 1). The operation of "taking the clusters together" (line 12) is currently the set union of the records, i.e. without considering their weights. However, weights are still considered in the overlap test performed at line 18. If the test succeeds, cluster X is marked as split (line 20), otherwise it is marked as disappeared (line 22).

The cases of absorption and survival are initially treated together: ζ_i clusters and their survival candidates are added to a list of absorptions and survivals (line 24). When all ζ_i clusters are processed, this list is completed (line 26). Then, for each ζ_j cluster Y, the detector extracts from this list all ζ_i clusters for which Y is a survival candidate (line 28). If this sublist contains more than one clusters, then these have been absorbed by Y: They are marked as such (lines 30–31) and removed from the original list (line 32). Otherwise, the single member of the sublist is a cluster X that has survived as Y (line 35). Again, the original list is updated (line 36).

Several improvements of this base algorithm are possible. First, instead of computing the overlap for each pair of clusters (line 5), MONIC computes the matrix $\mathcal{M}$ of the overlap values and retrieves the appropriate cell $Mcell$, whenever the overlap of two clusters is needed. Furthermore, split detections (lines 12, 18) can be performed more effectively, if one observes that two clusters in ζ_j cannot have common members. Then, the split test can be computed from the

```
1  FOR $X \in \zeta\_i$
2    splitCandidates = splitUnion = $\emptyset$;
3    survivalCandidate = NULL;
4    FOR $Y \in \zeta\_j$
5      Mcell = overlap(X,Y);
6      IF Mcell $\geq \tau$ THEN
7        IF g(X,Y) > g(X,survivalCandidate)
8          survivalCandidate = Y;
9        ENDIF
10     ELSEIF Mcell $\geq \tau\_split$ THEN
11       splitCandidates += Y;
12       splitUnion = splitUnion $\cup$ Y ;
13     ENDIF
14   ENDFOR
15   IF survivalCandidate == NULL OR splitCandidates == $\emptyset$
16     THEN deadList += X;                          // $X \rightarrow \odot$
17   ELSEIF splitCandidates $\neq \emptyset$ THEN
18     IF overlap(X,splitUnion) $\geq \tau$ THEN
19       FOR $Y$ $\in$ splitCandidates
20         splitList += (X,Y);
21       ENDFOR                      // $X \xrightarrow{\subsetneq} splitCandidates$
22     ELSE deadList += X;                          // $X \rightarrow \odot$
23     ENDIF
24   ELSE Absorptions_Survivals += (X,survivalCandidate);
25   ENDIF
26 ENDFOR
27 FOR Y $\in \zeta\_j$
28   absorptionCandidates = makeList(Absorptions_Survivals,Y);
29   IF cardinality(absorptionCandidates) > 1 THEN
30     FOR X $\in$ absorptionCandidates
31       absorbtionList += (X,Y);                   // $X \xrightarrow{\subsetneq} Y$
32       Absorptions_Survivals -= (X,Y);
33     ENDFOR
34   ELSEIF absorptionCandidates == X THEN
35     survivalList += (X,Y);                       // $X \rightarrow Y$
36     Absorptions_Survivals -= (X,Y);
37   ENDIF
38 ENDFOR
```

Figure 2: Detector of external transitions

individual intersection values in $\mathcal{M}$ because:

$$\sum_{a \in X \cap (\cup_{u=1}^{p} Y_u)} age(a, t_j) = \sum_{u=1}^{p} \sum_{a \in X \cap Y_u} age(a, t_j)$$

Thus, the complexity is $\mathcal{O}(K^2)$ for $K = \max\{|\zeta_i|, |\zeta_j|\}$, once the matrix $\mathcal{M}$ is computed.

4.2 Detection of Internal Transitions

Survived clusters may undergo internal changes. In Table 2, we have grouped the internal transitions as changes in size, in compactness and location. The transitions inside a group are mutually exclusive, but transitions of different groups can be combined. For example, a cluster $X \in \zeta_i$ matched by $Y \in \zeta_j$ can become larger and more compact.

Size transition indicators compare the datasets of X and Y, taking the t_i weights of the records in X into account. Compactness transitions cannot be traced by observing the data records directly, so we resort to studying derivative values over the data distribution. The indicator appearing in Table 2 is the standard deviation: If it has decreased by more than some small value δ, then the cluster has become more compact; if it has increased by more than δ, the cluster has

Transition	*Notation*	*Indicator*
the cluster survives	$X \rightarrow Y$	$Y = match_\tau(X, \zeta_j)$ AND $\nexists Z \in \zeta_i \setminus \{X\} : Y = match_\tau(Z, \zeta_j)$
the cluster is split into multiple clusters	$X \xrightarrow{\subseteq} \{Y_1, \ldots, Y_p\}$	$(\forall u = 1 \ldots p : overlap(X, Y_u) \geq \tau_{split}) \wedge overlap(X, \cup_{u=1}^{p} Y_u) \geq \tau \wedge$ $(\nexists Y \in \zeta_j \setminus \{Y_1 \ldots, Y_p\} : overlap(X, Y) \geq \tau_{split})$
the cluster is absorbed	$X \xrightarrow{\subseteq} Y$	$Y = match_\tau(X, \zeta_j)$ AND $\exists Z \in \zeta_i \setminus \{X\} : Y = match_\tau(Z, \zeta_j)$
the cluster disappears	$X \rightarrow \odot$	none of the above cases holds for X
a new cluster has emerged	$\odot \rightarrow Y$	

Table 1: External transitions of a cluster

Transition type	*Subtype*	*Notation*	*Indicators*
1. Size transition	1a. the cluster shrinks	$X \searrow Y$	$\sum_{x \in X} age(x, t_i) > \sum_{y \in Y} age(y, t_j) + \varepsilon$
	1b. the cluster expands	$X \nearrow Y$	$\sum_{y \in Y} age(y, t_j) > \sum_{x \in X} age(x, t_i) + \varepsilon$
2. Compactness transition	2a. the cluster becomes compacter	$X \xrightarrow{\bullet} Y$	$\sigma(Y) < \sigma(X) - \delta$
	2b. the cluster becomes diffuser	$X \xrightarrow{\star} Y$	$\sigma(Y) > \sigma(X) + \delta$
3. Location transition	Shift of center (I1) or distribution (I2)	$X \cdots \rightarrow Y$	I1. $\lvert\mu(X) - \mu(Y)\rvert > \tau_1$ `//mean`
			I2. $\lvert\gamma(X) - \gamma(Y)\rvert > \tau_2$ `//skewness`
No change		$X \leftrightarrow Y$	

Table 2: Internal transitions of a cluster

become more diffuse. Other statistics can be used instead of the standard deviation, e.g. kurtosis, while a significance test can be applied instead of using a constant δ.

Location transitions over a static metric space are cluster "movements" inside the invariant trajectory. If there is no static metric space, then we define location transitions as shifts in the distribution: Indicator I1 detects shifts of the mean, I2 traces changes in the skewness $\gamma()$.

4.3 Lifetime of Clusters and Clusterings

Intuitively, if most clusters in a clustering survive from one period to the next, then the population is static and the clustering describes it well. If transitions are frequent, then the population is volatile and the clustering cannot describe it well. We model the lifetime of clusters and clusterings and use them to gain insights on the evolution of the population.

DEFINITION 5. *Let C be a cluster and t_i be the first timepoint where it emerged (as part of clustering ζ_i). The lifetime of C is the number of timepoints, in which C has survived. We define (i) a "strict lifetime"* lifetimeS *as the number of consecutive survivals without internal transition, (ii) a "lifetime under internal transitions"* lifetimeI *for which all survivals are counted and (iii) a "lifetime with absorptions"* lifetimeA *that further counts absorptions of C.*

We compute cluster lifetime in a backward fashion: We start with ζ_n and set the lifetime of its clusters to 1. At an earlier timepoint t_i, the strict lifetime of cluster X is 1 if X did not survive in t_{i+1}. If there is a $Y \in \zeta_{i+1}$ with $X \leftrightarrow Y$, then $lifetimeS(X) = lifetimeS(Y) + 1$. If there is a $Y \in \zeta_{i+1}$ with $X \rightarrow Y$, then the lifetime of X under internal transitions is $lifetimeI(X) = lifetimeI(Y) + 1$. If there is a $Y \in \zeta_{i+1}$ with either $X \rightarrow Y$ or $X \xrightarrow{\subseteq} Y$, then the lifetime of X with absorptions is $lifetimeA(X) = lifetimeA(Y)+1$.

The survival of a clustering built at t_i is reflected on the number of its clusters that survive or are absorbed at t_{i+1}:

DEFINITION 6. *Let ζ_i be the clustering at $t_i, i = 1 \ldots n-1$. Its "survival ratio" is the portion of its clusters that survived (possibly with internal transitions) in ζ_{i+1}. Its absorption ratio is the portion of clusters absorbed by clusters of ζ_{i+1}.*

$$survivalRatio(\zeta_i) = \frac{|\{X \in \zeta_i | \exists Y \in \zeta_{i+1} : X \rightarrow Y\}|}{|\zeta_i|} \quad (1)$$

$$absorptionRatio(\zeta_i) = \frac{|\{X \in \zeta_i | \exists Y \in \zeta_{i+1} : X \xrightarrow{\subseteq} Y\}|}{|\zeta_i|} \quad (2)$$

where $|U|$ denotes the cardinality of set U. The "passforward ratio" of ζ_i is the sum of its survival and absorption ratios.

5. EXPERIMENTS

We have experimented with MONIC on the ACM library section H2.8 on "database applications". Goal was to gain insights on the evolution of the clusters and to study the impact of different parameter settings.

5.1 Section H.2.8 of the ACM digital library

ACM library section H.2.8 "Database applications" contains publications on (1) data mining, (2) spatial databases, (3) image databases, (4) statistical databases, (5) scientific databases – categorized in the corresponding classes. It further contains (6) uncategorized documents, i.e. those assigned in the parent class "database applications" only. For the time 1997–2004, we have selected the documents whose primary (or a secondary) class is one of the 6 classes in H.2.8. We must note here that the collection is unbalanced: the class "Data Mining" is larger than all the others together and grows faster than them. Many clustering algorithms have difficulties with such data distributions.

For each document, we have considered the title and the list of keywords. We have designed several feature spaces, ranging from the whole set of words to a small list of frequent words. We have also studied alternative weighting schemes, such as the embedded mechanism of CLUTO [1] and the entropy-based feature weighting function of [4]. The best clusterings with respect to the ACM categorization were acquired (a) for a feature space consisting of the 30 most frequent, TF×IDF-weighted words and (b) for the method of [4]. We chose the former, computationally simpler approach.

[1]http://glaros.dtc.umn.edu/gkhome/views/cluto/

For clustering, we have experimented with Expectation-Maximization [10], with a hierarchical clusterer using single linkage, with CLUTO and with bisecting K-means. Best results were obtained with bisecting K-means for K=10. Document vectorization and clustering were then performed with the DIAsDEM Workbench open source text miner [2]. For data ageing, we used a sliding window of size 2.

5.2 Cluster transitions and threshold impact

We first set $\tau_{split} = 0.1$, varied the threshold τ from 0.45 (rather than 0.50) to 0.7 in steps of 0.05 and counted the clusters with internal or external transitions. For τ larger than 0.7, there were hardly any cluster survivals, so we omit these values. In Fig. 3(a) we see that the number of surviving clusters drops as τ increases, while there are more disappearing clusters and splits, as shown in Fig. 3(b), (c). For τ larger than 0.6, the number of splits does not change any more, so it is not shown. All surviving clusters experience changes in size. There are no absorptions, i.e. the passforward ratio is equal to the survival ratio.

Fig. 3(a), (b), (c) show that there are more disappearances than splits in early timepoints, while the trend reverses later. A possible explanation is that there are larger homogeneous cluster chunks at late timepoints. To study this closer, we have fixed $\tau = 0.5$ and varied τ_{split} from 0.1 to 0.35 with a step of 0.05. We see the results in Fig. 4: Splits still occur only for small values of τ_{split}, i.e. there are no large chunks. Indeed, the dominant class "Data Mining" grows substantially in the recent timepoints but is not homogeneous enough to produce clusters with a long lifetime.

5.3 Lifetime of clusters and clusterings

We have next studied the persistence of the clusterings. The passforward ratios (cf. Def. 6) are shown in Table 3, using asbolute numbers of clusters for simplicity. The clusterings at some timepoints have a high passforward ratio, independently of the τ values. At 2002, the passforward ratio is very low, indicating a drastic change in the population of documents after a rather stable period of two years. The nature and origin of this shift are discussed below.

τ	*1999*	*2000*	*2001*	*2002*	*2003*	*2004*
0.45	4	7	7	1	5	4
0.50	4	5	7	1	3	4
0.55	3	3	3	0	2	3
0.60	3	2	3	0	1	1
0.65	3	0	1	0	0	1
0.70	2	0	1	0	0	0

Table 3: Passforward ratios for different values of τ

5.4 Clusters vs Classes of the ACM Library

The population shift detected by MONIC in 2002 called for inspection of the cluster semantics. We have labeled each cluster with its two most frequent words and mapped these labels/"topics" to the ACM classes; details can be found in [9]. For cluster transition detection, we have set $\tau = 0.5$ and $\tau_{split} = 0.1$ and concentrated on splits, disappearances and cluster lifetime with internal transitions (lifetimeI), since there were no strict survivals and no absorptions. On this basis, we have checked whether cluster

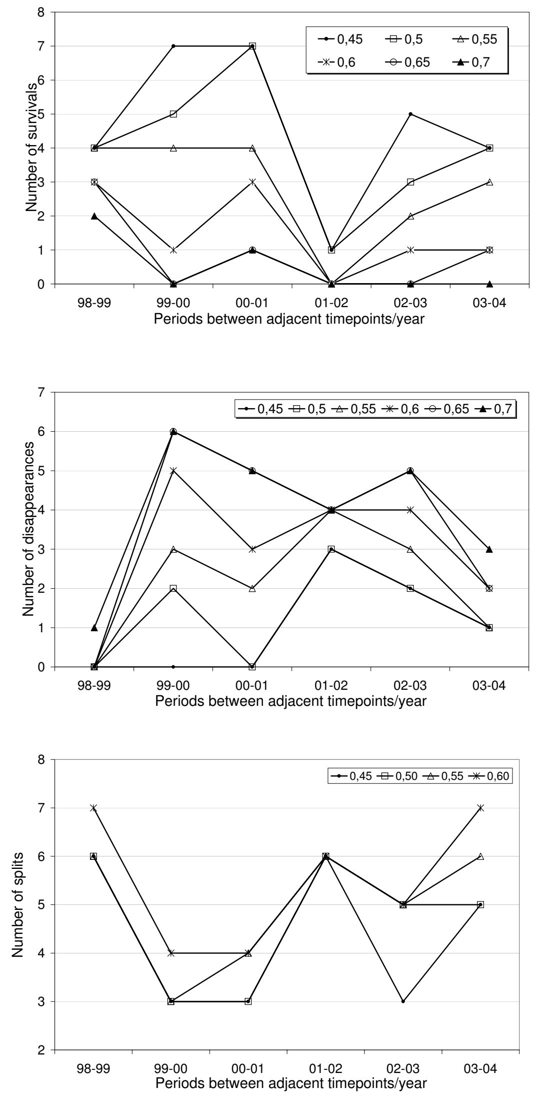


Figure 3: Cluster transitions for different values of τ: (a) Survivals, (b) disappearances and (c) splits

transitions correspond to comprehensible topic evolutions. The findings are as follows:

(a) There is always an unlabeled cluster, cluster "0", because K-means places in it all records that cannot be clustered properly. It survives (and grows) until 2002, whereupon it is scattered and replaced by a new garbage cluster.

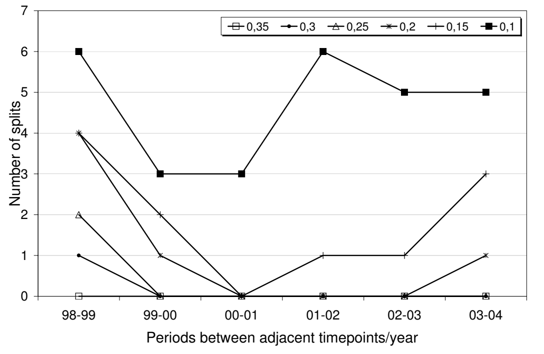
(b) Each clustering contains two or three clusters on data mining. In the first 4 timepoints, we find a growing cluster on "association rules". In 2002, it is split into a smaller cluster with the same label and a noisy cluster:

$$C_{1998_4} \nearrow C_{1999_9} \nearrow C_{2000_6} \nearrow C_{2001_4} \xrightarrow{\subseteq} \{C_{2002_7}, C_{2002_9}\}$$

where denote as C_{y_w} the identifier of the "association rules" cluster in year y, $w = 0 \ldots 9$ [3].
The small cluster C_{2002_7} disappears in 2003 ($C_{2002_7} \rightarrow \odot$).

[2]Registered under http://sourceforge.net/projects/hypknowsys

[3]Cluster identifiers are generated by the clustering algorithm at each timepoint. They do not indicate transitions.



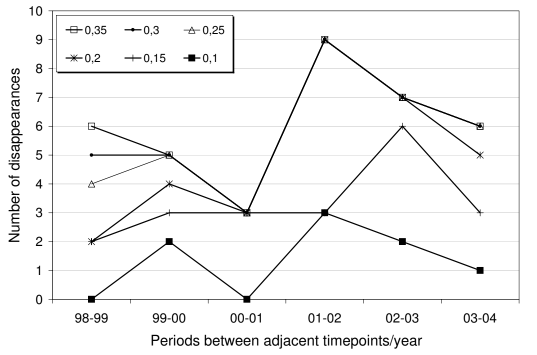


Figure 4: Cluster transitions for different values of τ_{split}: (a) Splits and (b) disappearances

One of the emerging clusters of 2004 ($\odot \rightarrow C_{2004_3}$) has again the label "association rules".

(c) The other clusters on data mining have less specific labels, such as "knowledge discovery" or "data mining". Their lifetime does not exceed 3 timepoints, during which they experience splits and size transitions.

(d) At the early timepoints, there are clusters labeled "spatial" and "image" (later: "image retrieval"). The labels appear in several periods but are associated with different clusters, so the cluster lifetime is low. Clusters associated to classes other than "Data Mining" appear only until 2002.

Hence, MONIC detected a remarkable shift in the accummulating H.2.8 section between 2001 and 2002, signalled by cluster splits and disappearances. The history of H.2.8 contains at least one event that may explain this shift: Starting with KDD'2001, the proceedings of the conference and of some adjoint workshops are being uploaded in the ACM Digital Library, enriching the H.2.8 section with a lot of documents on many subtopics of data mining.

6. CONCLUSION AND OUTLOOK

We have presented the framework MONIC for the monitoring of cluster transitions. MONIC encompasses a cluster transition model and a transition detection algorithm, operating upon clusterings over an accummulating dataset. We have applied MONIC on a section of the ACM library and have shown how cluster transitions give insights to changes of the data population. Currently, we work on heuristic enhancements of the transition detection algorithm to reduce the matrix computation overhead at each timepoint. This includes the use of summary data, as discussed below.

Data records are not always available for cluster monitoring, though, e.g. because of the storage demand or due to privacy considerations. In such cases, only summary data are available. MONIC does use summary data to detect internal transitions. In future work, we want to use summary data also for the detection of external transitions and to elaborate on the tradeoff between performance gain and information loss (for example, splits and absorptions cannot be traced). Finally, we plan to use MONIC to test the stability of clusters and clusterings over time.

7. REFERENCES

[1] C. Aggarwal. On change diagnosis in evolving data streams. *IEEE TKDE*, 17(5):587–600, May 2005.

[2] S. Baron, M. Spiliopoulou, and O. Günther. Efficient monitoring of patterns in data mining environments. In *Proc. of 7th East-European Conf. on Advances in Databases and Inf. Sys. (ADBIS'03)*, LNCS, pages 253–265. Springer, Sept. 2003.

[3] I. Bartolini, P. Ciaccia, I. Ntoutsi, M. Patella, and Y. Theodoridis. A unified and flexible framework for comparing simple and complex patterns. In *Proc. of ECML/PKDD 2004*, Pisa, Italy, Sept. 2004.

[4] C. Borgelt and A. Nürnberger. Experiments in Document Clustering using Cluster Specific Term Weights. In *Proc. Workshop Machine Learning and Interaction for Text-based Information Retrieval (TIR 2004)*, pages 55–68, Ulm, Germany, 2004.

[5] V. Ganti, J. Gehrke, and R. Ramakrishnan. A Framework for Measuring Changes in Data Characteristics. In *Proc. of the 18th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 126–137, Philadelphia, Pennsylvania, May 1999. ACM Press.

[6] P. Kalnis, N. Mamoulis, and S. Bakiras. On Discovering Moving Clusters in Spatio-temporal Data. In *Proc. of 9th Int. Symposium on Advances in Spatial and Temporal Databases (SSTD'2005)*, number 3633 in LNCS, pages 364–381, Angra dos Reis, Brazil, Aug. 2005. Springer.

[7] Q. Mei and C. Zhai. Discovering Evolutionary Theme Patterns from Text - An Exploration of Temporal Text Mining. In *Proc. of KDD'05*, pages 198–207, Chicago, IL, Aug. 2005. ACM Press.

[8] S. Moringa and K. Yamanichi. Tracking Dynamics of Topic Trends Using a Finite Mixture Model. In *Proc. of KDD'04*, pages 811–816, Seattle, Washington, Aug. 2004. ACM Press.

[9] R. Schult and M. Spiliopoulou. Discovering emerging topics in unlabelled text collections. In *Proc. of ADBIS'2006*, Thessaloniki, Greece, Sept. 2006. Springer. to appear.

[10] I. H. Witten and E. Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, San Fransisco, 2nd edition, 2005.

[11] H. Yang, S. Parthasarathy, and S. Mehta. A generalized framework for mining spatio-temporal patterns in scientific data. In *Proc. of KDD'05*, pages 716–721, Chicago, IL, Aug. 2005. ACM Press.