Distributed Spectral Cluster Management: A Method For Building Dynamic Publish/Subscribe Systems

Muhammad Adnan Tariq, Boris Koldehofe, Gerald G. Koch, Kurt Rothermel
University of Stuttgart
first name.last name@ipvs.uni-stuttgart.de

ABSTRACT

In recent years peer-to-peer (P2P) networking has gained high popularity for large-scale content distribution. Prominent systems expect a large user base with rather diversified demands. Yet it is highly challenging to achieve scalability without sacrificing the expressiveness of queries in such systems. This paper proposes distributed spectral cluster management, an approach which adapts the techniques from spectral graph theory to work in distributed settings. The proposed approach is applied to content-based publish/subscribe to i) significantly reduce the cost for event dissemination by clustering subscribers exploiting the similarity of events, ii) preserve the expressiveness of the subscription language, and iii) perform robustly in the presence of workload variations. The evaluations analyze the accuracy of the proposed distributed spectral mechanisms and show their effectiveness to significantly reduce the efforts to disseminate events under many practical workloads.

Categories and Subject Descriptors

C.2.4 [**Computer Systems Organization**]: Distributed Systems; C.2.1 [**Computer Systems Organization**]: Network Architecture and Design

General Terms

Content-based, Publish/Subscribe, Broker-less, Spectral Clustering, P2P

1. INTRODUCTION

Clustering is a popular approach to reduce the complexity of building and maintaining P2P based overlays by efficiently identifying service-specific similarities between peers, e.g. clustering by performance measures in multicast groups or by semantic similarities in content distribution networks. In particular, for content-based publish/subscribe – which itself is a highly popular and important paradigm to build large-scale distributed applications – clustering has proven its potential for significant efficiency gains [31, 25, 12]. Yet it is highly challenging for broker-less content-based publish/subscribe systems to use the full potential of cluster management.

In content-based publish/subscribe, publishers and subscribers are loosely coupled and exchange information in the form of events. Subscribers are provided with expressive means to specify their interest in certain events by complex filter operations on the event content without needing to know the set of publishers. Likewise, publishers can inject events into the system regardless of the set of subscribers. In a broker-less publish/subscribe system peers do not only act as subscribers but also form the event forwarding overlay network, i.e., all subscribers contribute in forwarding events. Therefore, the rate of events that peers receive and forward though lacking a matching subscription (*false positives*) is an important factor for the resource overhead in such a system. In particular, false positives waste system resources by increasing network bandwidth utilization and inducing extra processing load on the peers in an overlay network [30]. However, as observed by Cao et al. [9], false positives cannot be avoided completely without sacrificing the scalability of a publish/subscribe system. The interest of subscribers can be highly diverse and in the worst case generate 2^n distinct subscriber groups out of n subscribers.

Subscription clustering is one promising way to reduce the effect of false positives by grouping subscribers with similar subscriptions in a limited number of clusters such that the event dissemination within each cluster can be very efficient w.r.t. the number of false positives [10, 28, 25].

Subscription clustering in a publish/subscribe system is complicated due to the lack of global knowledge, high dynamicity of subscribers and continuously evolving event traffic. Existing approaches to clustering mostly consider the structural (absolute) similarity between the subscriptions such as the area occupied by the intersection of two subscriptions [7, 3, 33, 37, 10]. These approaches restrict the expressiveness of the content-based model to predefined numeric attributes and lack mechanisms to further reduce false positives by considering the similarities between subscriptions according to the current event traffic. Only a handful of approaches take into account the current event load of the system to create clusters of subscribers. These approaches either assume global knowledge [28, 25, 12] or rely on the presence of a network of reliable brokers [5].

This paper presents an efficient and scalable approach to cluster management coping with high workload variations. We study its effectiveness in the context of broker-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
DEBS '12, July 16–20, 2012, Berlin, Germany.
Copyright 2012 ACM 978-1-4503-1315-5 ...$15.00.

© ACM, 2012. This is the author's version of the work. It is posted here by permission of ACM for your personal use. Not for redistribution. The definitive version was published in Proceedings of the 6th ACM International Conference on Distributed Event-Based Systems, pp. 213-224, Berlin, Germany, 2012. http://doi.acm.org/10.1145/2335484.2335508

less publish/subscribe systems. The approach preserves the expressiveness of the content-based model and can work with any method to estimate similarities between subscriptions. Techniques from spectral graph theory are used to perform subscription clustering, because these techniques are proved to be more accurate in finding clusters than traditional mechanisms such as k-means [24]. This work is the first to study spectral clustering mechanisms in the context of content-based publish/subscribe systems. Furthermore, the presented approach can be seen as a general framework to perform distributed spectral clustering and can be applied to other areas such as document clustering. Spectral clustering in a distributed setting is highly challenging and to the best of our knowledge has not been previously addressed in literature.

The main contributions of this paper are: i) identification of different centralized spectral methods that can solve the subscription clustering problem and comparison of the effectiveness of those methods with the related approaches (cf. *Section 3*), ii) development of efficient and scalable mechanisms to perform spectral clustering in a distributed manner with accuracy closely matching that of centralized methods (cf. *Section 4*), iii) an approach to create and maintain clusters of subscribers using the developed distributed spectral mechanisms in a highly dynamic P2P based system (cf. *Section 5*), and iv) thorough evaluations of different aspects of the proposed distributed approach under different workloads and dynamics (cf. *Section 6*).

2. SYSTEM MODEL AND PROBLEM FORMULATION

We consider a content-based publish/subscribe system without broker infrastructure. Publishers and subscribers contribute as peers to the maintenance of the system. Publishers disseminate events into the system, while the subscribers specify filtering criteria for the selection of desired events using (possibly many) *subscriptions.*

A subscription f is a stateless boolean function [35] that accepts an event e as an argument. An event e matches a subscription f if $f(e) = true$. A *containment relation* can be defined on subscriptions. Let E_{f_1} and E_{f_2} denote the set of events matching the subscriptions f_1 and f_2 respectively. Then f_1 is said to be *covered* by the subscription f_2, denoted by $f_1 \prec f_2$, iff $E_{f_1} \subseteq E_{f_2}$ holds. Similarly, two subscriptions f_1 and f_2 are said to be overlapping, denoted by $f_1 \simeq f_2$, iff $E_{f_1} \cap E_{f_2} \neq \emptyset$.

In a broker-less publish/subscribe system as described above, the main concern is to reduce the cost of event dissemination by avoiding *false positives* (events a peer is required to forward while they do not match any of the peer's subscriptions). Subscription clustering is an effective approach for this purpose. It partitions the subscriber peers with similar subscriptions into a small number of groups such that the event dissemination within each group incurs few false positives.

To perform clustering, usually the absolute (structural) similarity between the subscriptions, such as the overlap relation ($\simeq$) defined on E_f, is considered to calculate their closeness in receiving the same events. However, the overlap relation does not take into account the number of events recently matched by the subscriptions for calculating the similarities. For instance, a low event rate causes little false positives in a cluster of two similar, yet not identical subscriptions, while with a high event rate, the same cluster encounters a large number of false positives, so that the two subscriptions had better not be clustered. Therefore, to achieve good clustering that minimizes false positives, the current event load of the system should be considered.

In more detail, let $\overline{E}_f^t$ be the set of last m events matched by the subscription f before a given time t. Let $sim(i, j, t)$ be a function that defines the similarity between two arbitrary subscriptions f_i and f_j at time t using their recently matched events i.e. $\overline{E}_{f_i}^t$ and $\overline{E}_{f_j}^t$.

The set of N subscriptions in the system $\Pi = \{f_1, ..., f_N\}$ and the pairwise similarities between them can be represented by a *similarity graph.* A similarity graph denoted by $G = (\Pi, \mathcal{E}, t)$ is a weighted undirected graph, where $\mathcal{E}$ is the set of edges connecting distinct subscriptions with non-zero similarity values, i.e., $\mathcal{E} = \{\varepsilon_{i,j} : f_i \neq f_j \wedge sim(i, j, t) > 0\}$. The similarities between subscriptions and hence the similarity graph change over time, however for the clearness in presentation we will omit time t in the following. Using the adjacency matrix of the similarity graph, which is a symmetric matrix we denote by $W \in \mathbb{R}^{N \times N}$, we can define the *weighted degree* d of a subscription f_i as the sum of its pairwise similarities with all other subscriptions in the system, i.e., $d_i = \sum_{j=1}^{N} w_{i,j}$. The degree matrix denoted by $D \in \mathbb{R}^{N \times N}$ is a diagonal matrix with degrees $(d_1, ...d_N)$ of all subscriptions on the diagonal. Let Π_1, Π_2 denote two groups of vertices in G. The inter-group weight or the sum of edge weights between the vertices of Π_1 and Π_2 can be defined as $W(\Pi_1, \Pi_2) = \sum_{i \in \Pi_1, j \in \Pi_2} w_{i,j}$. Furthermore, the degree of a group Π_1 is defined as $d_{\Pi_1} = W(\Pi_1, \Pi)$.

Given a dynamic set of subscribers and continuously evolving similarity graph, our objective is to maintain k disjoint clusters of subscriptions $(\Pi_1, ...\Pi_k)$ in a publish/subscribe system so that,

1. inter-cluster weights are minimized, i.e., overlap between the events in different clusters is minimized.

2. clusters are balanced in terms of their size to ensure even event dissemination load on each cluster.

In the presence of only first criterion (i.e. minimization of inter cluster weights), the subscription clusters can be efficiently created by solving minimum cut problem [24]. However in practice it has tendency to create singleton clusters i.e., clusters consisting of only individual subscriptions, and does not lead to satisfactory results. The second criterion is added to overcome this problem and create clusters with reasonably large number of subscriptions.

3. CENTRALIZED SUBSCRIPTION CLUSTERING

In this section we describe our approach to maintain the subscription clusters in a centralized fashion. Similar to other state of the art centralized approaches [31, 28, 25], the proposed centralized approach has potential to be used as a competent method to perform subscription clustering in a content-based publish/subscribe system. In particular our contributions in this section are twofold. First we propose different spectral methods that can effectively solve the subscription clustering problem. Second, we show the capability of the proposed methods to produce clusters with improved quality in comparison to the related approaches.

3.1 Subscription clustering using spectral methods

In the following, we describe the spectral methods and main steps needed to perform subscription clustering.

3.1.1 *Similarity function*

In order to perform clustering, the first step is to quantify the similarities between the subscriptions by defining an appropriate similarity function $sim(i, j)$. Intuitively, the similarity between the two subscriptions increases with the increase in overlapping events and decreases with the non-overlapping event traffic. Based on this intuition we propose to use the Jaccard[1] (in short Jac) similarity function [14], which is defined as a ratio of overlapping event sets matched by the subscriptions to the union of their overlapping and non-overlapping event sets, i.e., $Jac = \frac{|\overline{E}_{f_i} \cap \overline{E}_{f_j}|}{|\overline{E}_{f_i} \cup \overline{E}_{f_j}|}$. Similarities are assigned in the range $[0, 1]$, where 0 means complete disjointness.

3.1.2 *Clustering as graph partitioning*

Once the similarity function is in place and the similarity graph is calculated, the subscription clustering is performed by partitioning the similarity graph into k disjoint sub-graphs. In this paper, we select two graph partitioning objective functions which fulfill our requirements of strongly connected and balanced clusters:

$$Ratio\ Association = \underset{\Pi_1,\ldots\Pi_k}{max} \sum_{i=1}^{k} \frac{W(\Pi_i,\Pi_i)}{|\Pi_i|} \quad (1)$$

$$Normalized\ Cut = \underset{\Pi_1,\ldots\Pi_k}{min} \sum_{i=1}^{k} \frac{W(\Pi_i,\Pi\backslash\Pi_i)}{d_{\Pi_i}} \quad (2)$$

The *Ratio Association* (in short RAssoc) objective [16] aims to maximize the association between the cluster members (intra cluster), whereas the *Normalized Cut* [16] (in short NCut) tries to minimize the association between different clusters (inter cluster). The objective functions strive to create clusters that are balanced in terms of the number of vertices and the degree (edge weights) respectively. We selected these two objective functions because of their different characteristics. RAssoc is better in creating strongly connected clusters, whereas NCut is good at forming balanced clusters (cf. Section 3.2). Both objective functions are NP-hard to solve in the discrete domain. The relaxed versions of these objective functions can be solved by *spectral analysis* of the similarity graph.

3.1.3 *Spectral Analysis*

The spectral clustering consists of two main parts: i) initially the hard discrete problem is relaxed to obtain the global optimum in the continuous domain by means of eigendecomposition [14], ii) afterwards a k-means algorithm [21] is used to get discrete partitions from the real valued eigenvectors.

Intuitively, the first part (eigendecomposition) embeds the vertices of a similarity graph (i.e., the subscriptions) in a low-dimensional space with k dimensions by treating eigenvectors as geometrical coordinates and hence is termed as *dimensionality reduction* or *embedding*. The dimensionality reduction step is beneficial in reducing the effect of noise and extracting the main features of the data to cluster. This part depends on the partitioning objective function.

Algorithm 1 Centralized Spectral Clustering Steps

```
1: Calculate the similarities between $N$ subscriptions
       Generate similarity graph $G$
       Apply Guassian function on $G$ (NCut)
           $\varepsilon_{i,j} = exp(\frac{-\|sim(i,j)\|}{\sigma^2})$
2: Perform eigenvector decomposition
       $Y = \Lambda_{k+} V_{k+}^T : \overline{W} = U\Lambda V^T$ (RAssoc)
       $Y = Q_{k-}^T : (D - W)q = \lambda Dq$ ( NCut)
3: Apply k-means clustering algorithm
       Cluster $(y_i)_{i=1,..N} \in Y$ into k clusters
```

The Ratio Association function performs linear dimensionality reduction using *Principal Component Analysis* (PCA) [14]. PCA performs dimensionality reduction by means of *singular value decomposition* (SVD) [14], i.e., $\overline{W} = U\Lambda V^T$, where $\overline{W}$ is a centered matrix obtained by subtracting the mean of the similarity matrix W from its columns, $\Lambda = \{\lambda_i, ..., \lambda_N\}$ is a diagonal matrix of eigenvalues, and V and U are the matrices of right and left singular vectors respectively. The k-dimensional coordinates are obtained as $\Lambda_{k+} V_{k+}^T$, where the subscript $k+$ identifies that only the k largest non-zero eigenvalues and corresponding eigenvectors are used.

As an alternative, Normalized Cut embeds the coordinates in non-linear space which only preserves the geometry of the local neighborhood. To model neighborhood, the pairwise similarities between the vertices in the similarity graph are weighted by Gaussian kernel, i.e., $\varepsilon_{i,j} = exp(\frac{-\|sim(i,j)\|}{\sigma^2})$, where σ controls the width of the neighborhood. The low-dimensional coordinates correspond to the k eigenvectors (Q_{k-}^T) with smallest non-zero eigenvalues (represented by subscript $k-$) obtained by solving the generalized eigensystem. *Algorithm 1* shows the overall steps needed to perform spectral clustering.

3.2 Effectiveness of spectral clustering

We show the effectiveness of our approach w.r.t. the quality of the identified clusters and reduction of false positives as compared to two related approaches.

3.2.1 *Experimental Setup*

Experiments are performed using PeerSim [1]. We assume a content-based schema with up to 5 integer attributes, where the domain of each attribute is in the range [0,30]. Our approach, however, is not limited to a certain number, type or domain of attributes[2]. Experiments are performed on two different models for the distributions of subscriptions and events. The uniform model (M_1) generates random subscriptions/events independent of each other. The interest popularity model (M_2) chooses five hotspot regions around which subscriptions/events are generated using the widely used zipfian distribution. For the experiments, up to $32,000$ subscriptions and $6,000$ events are used. Each event matches

[1]We have evaluated many different similarity functions such as Russell [14], Dice [14], Simpson [14] and Simple matching [14]. However, the results of the Jaccard function were superior in consistency. Therefore, in this paper we only focus on the properties of this function.

[2]We would like to stress here that the results presented in this paper are of general validity and are independent of the content-based schema or language.

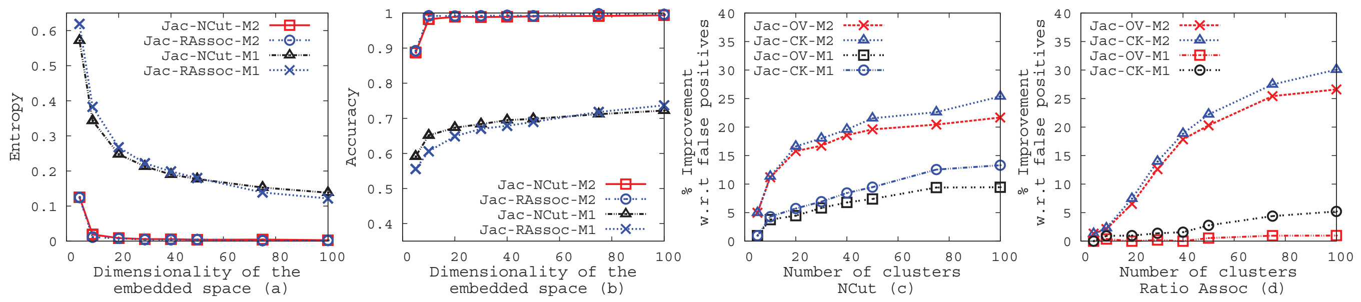

Figure 1: Evaluations for centralized spectral clustering

5% of subscriptions and each subscription maintains a list of 20 most recently matched events.

We compare our work with two widely used related approaches. The first approach [10] (denoted as CK) clusters subscriptions according to the coordinates of their centroid using a k-means algorithm. The second approach (denoted as OV) is a representative of many prominent publish/subscribe systems [7, 31]. It considers the absolute overlap between the subscriptions as a measure of their similarity and uses k-means or R-tree [39] algorithms to cluster them.

3.2.2 *Quality of clusters*

The employed spectral clustering mechanisms provide no guarantee on the quality of the solution [24]. We therefore evaluate the quality of generated clusters by measuring their *Entropy* [11] and *Accuracy* [11]. The entropy of a cluster specifies the disorder within the clusters, indicated by the distribution of ideal subscription clusters on the generated clusters, whereas accuracy measures the extent to which subscriptions are assigned to the correct clusters. Entropy and accuracy obtain scores in the range of [0,1]. A lower entropy and higher accuracy score imply better clustering. Figures 1(a) and (b) show that for both of the alternative partitioning functions (RAssoc and NCut) in Algorithm 1, the quality of clusters improves with the increase in the dimensionality of the embedded space. However, sufficiently good quality clusters can be obtained with just 20-dimensional space. Moreover, the proposed mechanisms perform reasonably good for both workloads (M_1 and M_2). In case of M_2, which is the more realistic of the two workloads, clusters of almost all the subscriptions are identified correctly. However, the quality is not at its optimum for M_1 because the subscriptions are uniformly distributed and some subscriptions may not share event traffic with any other subscription, making it hard to cluster them.

3.2.3 *Reduction in false positives*

The reduction in the false positives is measured as the percentage improvement to the related centroid (CK) and overlap (OV) based clustering approaches. In all the experiments, subscriptions are generated using only uniform distribution, whereas events follow uniform and zipfian distribution. Figures 1(c) and (d) show that in the case of zipfian event distribution (M_2) there is a considerable (up to 30%) improvement in reducing false positives in comparison to both the related approaches. However, for uniformly distributed event traffic the improvement drops to just $6-9\%$ for NCut and under 5% for RAssoc, which is predictable as there is no advantage of taking into consideration the event load based similarity in comparison to the absolute similarity metrics. Figures 1(c) and (d) indicate that RAssoc is better at reducing false positives in case of zipfian event distribution whereas NCut performs better in case of uniform event workload.

3.3 Properties of centralized clustering

In practice, subscriptions of peers as well as event traffic change dynamically and therefore, over time the previously calculated clusters may become suboptimal. In order to adapt to the changes, a central coordinator periodically collects information about the events matched (in the recent time window) by the subscriptions of the peers and repeats the clustering process.

The centralized subscription clustering approach, although still widely researched in literature [31, 28, 25], has some scalability issues. First, periodic fetching of the event histories, calculation of clusters and distribution of cluster membership information incurs significant overhead (in terms of bandwidth and processing resources) for the central coordinator. Second, the centralized spectral clustering is very expensive in terms of time and memory requirements. The calculation of the similarity graph encounters quadratic computational and memory requirements. Similarly, the calculation of k eigenvectors (of a dense matrix) involves $O(N^3)$ operations [18].

4. DISTRIBUTED SUBSCRIPTION CLUSTERING

Our aim is to develop an efficient and scalable approach to perform spectral clustering in a distributed fashion that can compute clusters with accuracy closely approximating the accuracy of the proposed centralized solution. In the subsequent sections, we will first describe the organization of the subscriber peers in an overlay (*Section 4.1*) and afterwards present mechanisms to perform dimensionality reduction (*Section 4.2*) and k-means algorithm (*Section 4.3*) in a distributed manner, exploiting the overlay organization.

4.1 Organization of subscriber peers

Subscriber peers are arranged into a multilevel hierarchy of small manageable groups as shown in *Figure 2*. The levels are numbered sequentially, with zero being the lowest level (denoted as L_0). All subscribers participate in the lowest level (L_0) groups such that the similarity graph formed

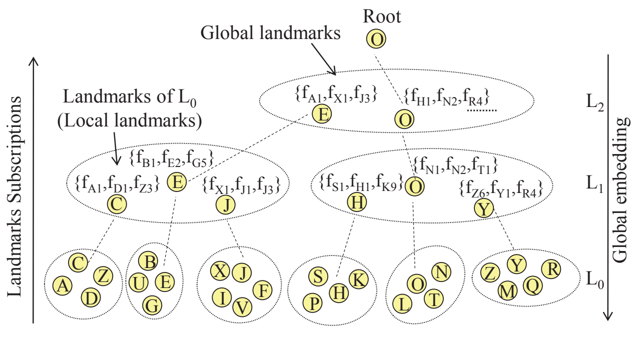
Root
Global landmarks
Landmarks of L_0
(Local landmarks)
Landmarks Subscriptions
Global embedding
$\{f_{A1},f_{X1},f_{J3}\}$
$\{f_{H1},f_{N2},f_{R4}\}$
$\{f_{B1},f_{E2},f_{G5}\}$
$\{f_{A1},f_{D1},f_{Z3}\}$
$\{f_{X1},f_{J1},f_{J3}\}$
$\{f_{S1},f_{H1},f_{K9}\}$
$\{f_{N1},f_{N2},f_{T1}\}$
$\{f_{Z6},f_{Y1},f_{R4}\}$
L_2
L_1
L_0

Figure 2: Hierarchical organization of subscribers

by the subscriptions of the members of a L_0 group is connected.[3]

Each group selects a *coordinator* peer that joins the higher level group. A group at level l with coordinator p is denoted as G_p^l and the number of subscriptions maintained by this group is denoted by $|G_p^l|$. In the subsequent formulation the subscript (p) will be dropped if it can be inferred from the context. At each level l, a coordinator p maintains a list of subscriptions (along with their matched events) called landmarks, which are selected uniformly at random from the members of group G_p^l.[4] The landmarks participate in the higher-level group to create the similarity graph. For example, in *Figure 2*, coordinator E maintains three landmarks subscriptions (from the peers B, G and E) which become part of the level L_1 group. The landmarks are used during distributed dimensionality reduction and for an accurate calculation of low-dimensional embedding. The number of landmarks should be higher than the dimension k of the embedded space.

Maintaining a hierarchical overlay network as described above in dynamic conditions is a well researched topic [6, 7]. For this reason we will not discuss the maintenance algorithms in this paper but rather focus on the more challenging issue of performing distributed spectral clustering.

4.2 Dimensionality reduction

To perform dimensionality reduction (low-dimensional embedding, cf. *Section 3.1*) in a distributed manner, three main issues should be considered. First, the resulting low-dimensional space should be consistent so that the coordinates of different subscriptions can be compared. Second, the embedding should adapt to reflect dynamic changes in the similarities between the subscriptions. Third, the error induced due to the distributed calculation of the embedding should be small.

Our approach for distributed dimensionality reduction addresses the above issues and comprises two steps: i) separate (local) embedding of a small subset of subscriptions in low-dimensional space and, ii) transformation of these independently generated local embeddings into a globally unified coordinate system.

Each group in the hierarchical organization separately calculates low-dimensional coordinates of its subscriptions. The local coordinates (embeddings) are calculated in isolation from each other and therefore use different coordinate systems. As a consequence, similarities between the subscriptions with different local embeddings are not preserved, and thus the k-means algorithm cannot be applied. To overcome this problem, local embeddings are transformed (projected) into a globally unified coordinate system. The landmarks of the highest level (root) group define the basis of the global coordinate system. The global basis is progressively projected to adjust the local embeddings level by level (by traversing the hierarchy) until the lowest level groups are reached.

Each group G^l at level l maintains a *projection* matrix w.r.t the parent group at level $l+1$. The projection matrix is used to transform the local coordinates of the group G^l to global coordinates as follows:

$$Y_{G^l} = H_{G^l} X_{G^l} \tag{3}$$

where $Y_{G^l} \in \mathbb{R}^{k \times |G^l|}$ are the projected (global) coordinates of the members of group G^l, $X_{G^l} \in \mathbb{R}^{k \times |G^l|}$ are the local coordinates and $H_{G^l} \in \mathbb{R}^{|k| \times |k|}$ (in short H) is the projection matrix. The calculation of local coordinates and the projection matrix depends on the properties of the low-dimensional space (i.e. linear or non-linear) as well as on the partitioning objective function. In the following we will describe mechanisms for both the objective functions (RAssoc and NCut).

4.2.1 Ratio association (Linear embedding)

Again, the low-dimensional coordinates are calculated using PCA [14] to compute an explicit linear mapping between the original space (similarities between subscriptions) and the embedded space. In our distributed organization we avoid the use of a global projection matrix in favor of separate matrices for each group which are maintained based only on the similarities between the groups' member subscriptions and local landmarks.

Local embedding: The local coordinates of the landmark subscriptions ($X_{G^l}^L$) maintained by the coordinator p of a group G^l are obtained by using the standard technique (cf. *Algorithm* 1, *line 2*). The local coordinates of all other subscriptions (X_{G^l}) of the group G^l are calculated w.r.t. these landmarks. Let $W_{G^l \to L} \in \mathbb{R}^{|L| \times |G^l|}$ denote the similarities between the subscriptions of G^l and the landmarks. The local coordinates X_{G^l} can be calculated using Nyström Approximation [18] as follows: $X_{G^l} = U^T W_{G^l \to L}$.

Global embedding: In order to convert the local coordinates (X_{G^l}) to the global coordinates (Y_{G^l}), a projection matrix is calculated by the coordinator. The landmarks maintained by the coordinator of a group G^l also participate in the parent group G^{l+1}. The projection matrix is calculated as a basis change matrix between the coordinates of the landmarks at two levels by solving a linear least square problem, i.e.,

$$min \ ||(X_{G^l}^L)^T H - (Y_{G^{l+1}}^L)^T||$$

The matrix $(X_{G^l}^L)^T$ can be decomposed into $U \Lambda V^T$ using singular value decomposition (SVD) [14], where $U \in \mathbb{R}^{|L| \times |k|}$,

[3]A subscriber with completely dissimilar (disjoint) subscriptions may participate in multiple lowest level groups.

[4]The random strategy yields reasonably good results (cf. Section 6). However, more sophisticated mechanisms can also be used for the selection of landmarks, such as maximum independent set of the similarity graph [17] formed by the similarities between the group members. We envision the study of different landmark selection mechanisms as future work.

$\Lambda \in \mathbb{R}^{|k|\times|k|}$ and $V \in \mathbb{R}^{|k|\times|k|}$. Thus the projection matrix is calculated as follows:

$$U\Lambda V^T H = (Y^L_{G^{l+1}})^T$$
$$H = V\Lambda^{\dagger}U^T(Y^L_{G^{l+1}})^T$$

If the matrix $X^L_{G^l}$ is rank deficient then Λ has some diagonal elements that are zero and cannot be inverted. Hence, the Moore-Penrose pseudo-inverse [14] $\Lambda^{\dagger}$ is used in the calculations.

4.2.2 Normalized Cut (Non-linear embedding)

In the case of Normalized Cut the dimensionality reduction step embeds the coordinates in a non-linear space. The reduced space does not preserve the global structure, it rather captures the geometries at a local neighborhood in the similarity graph. Let y_i be the k-dimensional coordinates associated with the i^{th} subscription, then the dimensionality reduction minimizes the following cost function [24]: $\Phi = \sum_{i,j=1}^{N} w_{i,j}\|\frac{y_i}{\sqrt{d_i}} - \frac{y_j}{\sqrt{d_j}}\|_2^2$. The cost function ensures that neighbors with larger weights in the similarity graph stay close in the low-dimensional space.

Local embedding: Let W_{G^l} denote the similarities between the subscriptions of group G^l, and D_{G^l} is the corresponding degree matrix (see *Section 2*). The local coordinates (X_{G^l}) of the subscriptions of G^l can be calculated by performing generalized eigendecomposition and selecting the k smallest non-zero eigenvectors (cf. *Algorithm* 1, *line 2*).

Global embedding: The projection matrix of a group G^l is calculated by minimizing the cost function of Normalized cut, i.e.,

$$\Phi = \sum_{i\in G^{l+1}}\sum_{j\in G^l} w_{i,j}||(\frac{y_i}{\sqrt{d_i}} - \frac{Hx_j}{\sqrt{d_j}})||^2 \qquad (4)$$

Equation 4 calculates the projection matrix H such that the error between the sum of squared distances of the projected coordinates of the subscriptions of G^l and subscriptions of the parent group G^{l+1} is minimized. We have omitted the derivation steps of projection matrix from Equation 4 for space reasons. In summary, to minimize Φ we set the first-order derivative of Equation 4 to zero so that the projection matrix H can be obtained as:

$$H = Y_{G^{l+1}}D^{-\frac{1}{2}}_{G^{l+1}}W^T D^{-1} D^{\frac{1}{2}}_{G^l} X^{\dagger}_{G^l}$$

where $W \in \mathbb{R}^{|G^l|\times|G^{l+1}|}$ is the weight matrix which specifies similarities between the subscriptions of the groups G^{l+1} and G^l, and $D \in \mathbb{R}^{|G^l|\times|G^l|}$ is the degree matrix obtained by summing the row of weight matrix W. Moreover, $D^{\frac{1}{2}}_{G^l} \in \mathbb{R}^{|G^l|\times|G^l|}$ and $D^{\frac{1}{2}}_{G^{l+1}} \in \mathbb{R}^{|G^{l+1}|\times|G^{l+1}|}$ represent the degree matrices of the groups G^l and G^{l+1} respectively.

Once the projection matrix H is calculated, Equation 3 can easily convert local low-dimensional coordinates into global coordinates.

4.3 Distributed k-means algorithm

After low-dimensional embedding, the next step is to partition the subscriptions into clusters by performing a k-means algorithm. We propose two methods to obtain subscription clusters. Both methods benefit from the hierarchical organization of subscriber peers (cf. *Section 4.1*).

Sampling method: In the hierarchical organization, landmark subscriptions maintained by a parent group are representative of the subscriptions in the subtree. Similarly, subscriptions of the root group represent the uniform sample of all the subscriptions in the system. In the sampling method, the coordinator of the root group performs the k-means algorithm on its landmark subscriptions and the resultant cluster centers are treated as the centers of the global clusters.

Hierarchical method: We extended the HP2PC [21] approach where each base (L_0) group independently performs the k-means algorithm to calculate local clusters. In a *Merge* phase, those clusters are merged level by level by their parent groups until the global clusters are obtained at the root. As the k-means algorithm is very sensitive to its initialization, we prepend a *Selection* phase in order to select good initial cluster centers for the k-means clustering performed at each L_0 group. The *Selection* phase starts at the root and propagates down the hierarchy. While the root uses random initialization, each subsequent group uses the cluster centers calculated by its parent group to initialize its k-means clustering and then forwards the adjusted centers to its child groups. Afterwards, a *Merge* phase similar to the HP2PC algorithm sets in so that finally, the root obtains information about the global clusters.

5. CLUSTER CREATION AND MAINTENANCE

Once the global clusters are obtained by means of distributed k-means algorithm, the information about the new cluster centers is distributed among the subscribers using the hierarchical organization (*Section 4.1*). In addition, bootstrap peers to join the clusters are announced along with the cluster centers. On reception of this information, each subscriber peer joins the cluster Π_j whose center is nearest to the global coordinates of its subscription f (i.e., $|y_f - y_{\Pi_j}|$ is minimum, where y_f and y_{Π_j} are k-dimensional coordinates associated with the subscription f and the center of cluster Π_j respectively). If the subscriptions of a subscriber are dissimilar and match different set of events, then the subscriber may join more than one cluster. Each cluster is maintained separately. Existing techniques such as IP multicast [15] or Application layer multicast (ALM) [6] can be used to disseminate events within each cluster. Furthermore, additional mechanisms such as subscription forwarding [9, 5, 23] can be performed within each cluster to even further reduce the rate of false positives.

P2P systems are very dynamic in nature and therefore it is infeasible to completely recalculate the clusters for every minor change in the set of subscriptions. Thus, upon arrival of a new subscription in the system, its subscriber first contacts the coordinator of its group in the hierarchical organization to obtain the global coordinates of its subscription by means of the locally maintained projection matrix. Afterwards, the subscriber joins the cluster whose center is nearest to the global coordinates of its subscription. Similarly, to counter the effect of minor changes in the events' distribution, the coordinator of each group locally calculates the global coordinates of its member subscriptions using the projection matrix and distributes them in its group. The newly computed global coordinates of a subscription may change its nearest cluster center; the subscriber of the corresponding subscription then joins the new cluster.

The dynamic changes in the subscriptions, as well as the event workload, can accumulate over time, so that the current set of clusters becomes suboptimal. In order to adapt to the subscription and event workload changes, the coordinator of the root group periodically starts the clustering process by sending a control message. The control message propagates towards the base groups. Each group coordinator on the path updates the projection matrix and recalculates the global coordinates of its member subscriptions. The calculation of the projection matrix is very fast as each group accommodates only a small percentage of the overall subscriptions (cf. Section 6). Furthermore, to expedite the projection process the local embeddings[5] are pre-calculated asynchronously, independent of the control message. In case the hierarchical k-means algorithm is used, the *Selection* phase is performed simultaneously along with the projection process. Once the clusters are obtained, the coordinator of the root group compares the centers of the newly obtained clusters with the centers of the currently deployed clusters. If the new centers deviate more than a predefined threshold allows, new clusters will be installed, i.e., subscribers receive information about the new clusters and join their subscriptions one by one with the cluster whose center is nearest to the subscription's global coordinates.

6. EVALUATION OF THE DISTRIBUTED APPROACH

We evaluate four aspects of our distributed approach: i) accuracy of the distributed dimensionality reduction, ii) quality of clusters created by the distributed k-means algorithm, iii) effectiveness of the overall distributed approach to reduce the cost of event dissemination under dynamically changing workload and in comparison to the related P2P based approach, and iv) scalability of the cluster management in terms of computational time and load on peers.

The experimental setup is the same as described in Section 3.2. We modified the NICE protocol [6] to use it for managing the hierarchical organization of subscribers. The number of peers in the experiments ranges from 1000 to 2000, with different percentages of churn. Moreover, in Figure 3, Lvs denotes the number of levels in the hierarchical organization, LM denotes the percentage of subscriptions selected as landmarks in each group, and Gps denotes the number of L_0 groups.

6.1 Accuracy of distributed dimensionality reduction

Two separate metrics are adopted to measure the accuracy of linear (RAssoc) and non-linear (NCut) dimensionality reduction. The *stress* [14] metric evaluates the quality of linear embedding by calculating the sum of squares of relative errors in the similarities of the subscriptions between the original space and the low-dimensional embedded space, i.e., $Stress = \sum_{i\in\Pi}\sum_{j\in\Pi}(\frac{w_{i,j}-\overline{w}_{i,j}}{w_{i,j}})^2$, where $\overline{w}_{i,j}$ is the similarity between the subscriptions i and j in the low-dimensional space. The linear embedding is accurate if the value of stress is zero.

[5]The calculation of the embedding (cf. Step 2, Algorithm 1) is the most expensive operation (cf. *Section 6.4*).

To measure the quality of non-linear embedding, we evaluate *trustworthiness* [36] and *continuity* [36] of the neighborhood relationship between the original and the embedded space. The trustworthiness (denoted as Ts) measures how far the neighborhood in the embedded space differs to the original neighborhood. The continuity (denoted as Cn) quantifies the deviation of the original neighborhood from the embedded space neighborhood. The *harmonic mean* [17] (denoted as $HScore$) of the trustworthiness and continuity is expressed as: $HScore = \frac{2.Ts(\kappa).Cn(\kappa)}{Ts(\kappa)+Cn(\kappa)}$, where κ is the size of the neighborhood used for the calculation of trustworthiness and continuity. In our experiments we used a neighborhood size of 50. Higher values of HScore indicate a better embedding accuracy. The upper bound is 1, which indicates that the κ nearest neighbors of every subscription are exactly the same in the original and the embedded space.

Figures 3(a) and (b) show the effect of the number of L_0 groups on the accuracy of the (linear and non-linear) embedded space in comparison to the centralized approach. All the L_0 groups have the same number of subscriptions. For instance, a total of 8 groups means that each group maintains 12.5% of the overall subscriptions in the system. The following conclusions can be drawn from the figures. First, the accuracy of the distributed approach with two levels of hierarchy closely resembles the accuracy of the centralized method for both types of embeddings. Second, the accuracy decreases slightly with the increase in the number of groups due to the corresponding decrease in the number of subscriptions per group. Third, the centralized and the distributed approaches produce embedding with better accuracy if the subscription workload concentrates on spots of interests (M_2).

Figures 3(c) and (d) show the accuracy of the embedding as a function of the percentage of landmark subscriptions per group. In general, the accuracy increases with the increase in the percentage of landmarks. This behavior is more significant with the increase in the levels of the hierarchy because each additional intermediate level decreases the overall percentage of subscriptions managed by the higher level groups by a factor of the number of subscriptions managed in the current level and the percentage of allowed landmarks in each group. Therefore lower percentages of landmarks result in significantly smaller numbers of subscriptions managed by the higher level groups. For instance, in case of 50% landmark subscriptions per group, the root of a 3-level hierarchy manages 25% subscriptions, whereas with 10% landmarks, it manages just 1% subscriptions, decreasing the accuracy of the overall embedding. Figures 3(e) and (f) show that the accuracy decreases with the increase in the levels of the hierarchy. However up to 4 levels of hierarchy the embedding can be performed with reasonable accuracy. We argue that a 4-level system with 30% landmark subscriptions per group is very promising to perform good low-dimensional embedding. First of all the accuracy is only slightly decreased as compared to the centralized approach. Second, a 4-level system can handle subscriptions in a scalable manner. For instance, in the evaluated scenario, the groups at the intermediate levels manage 30% and 9% of subscriptions respectively, and the root group only manages 2.7% subscriptions. In Section 6.4, we show that 4-level system with only 25% landmark subscriptions can easily scale well beyond $32,000$ subscriptions.

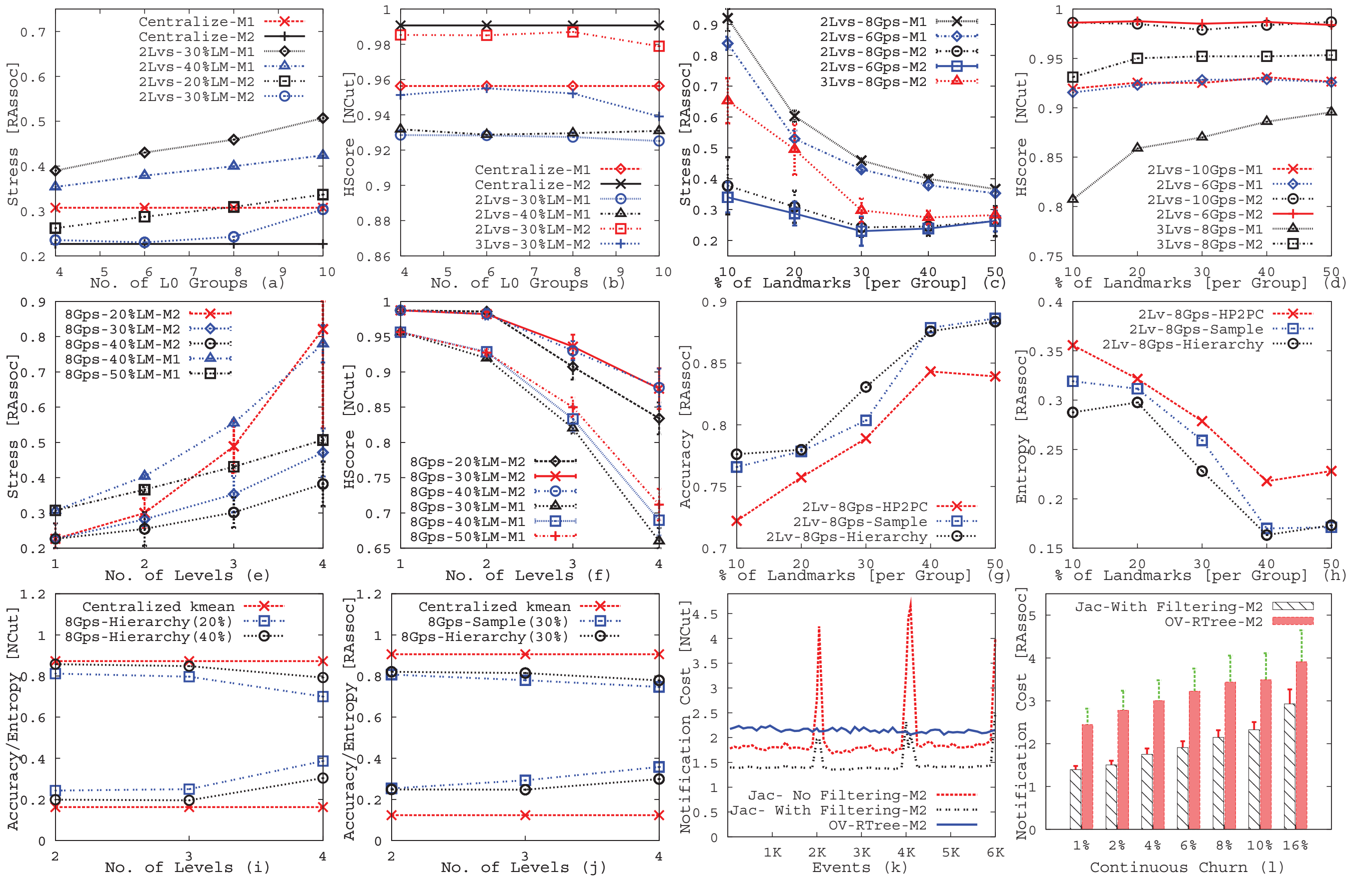

Figure 3: Evaluations for distributed clustering approach

6.2 Effectiveness of distributed k-means algorithm

We compare the quality of clusters obtained by the two proposed approaches, *Sampling method* (Sample) and *Hierarchical method* (Hierarchy), with the related distributed clustering approach HP2PC. The quality of clusters are evaluated by *Entropy* [11] and *Accuracy* [11] (cf. *Section 3.2*). Figures 3(g) and (h) show the values of entropy and accuracy versus the percentage of landmark subscriptions. As expected, the quality of clusters improves with the higher percentage of landmarks due to the corresponding increase in the accuracy of the low-dimensional embedded space. Moreover, both proposed methods outperform the HP2PC approach. Especially for the hierarchical method this indicates that the selection phase (cf. *Section 4*) is beneficial in improving the quality of clusters by providing a good estimate of the initial cluster centers. Figures 3(i) and (j) evaluate the effect of the number of levels on the quality of clusters. The trend shows that the quality of clusters gradually degrades with increasing levels of the hierarchy. There are many reasons for such a degradation: i) decrease in the accuracy of the low-dimensional embedding (cf. Figures 3(e) and (f)), ii) less accurate estimation of the initial cluster centers due to the decrease in the percentage of landmarks maintained by the root and iii) loss of cluster quality at higher levels of hierarchy as a result of the merging of clustering information from the lower level groups [21].

6.3 Effectiveness of overall distributed approach

We evaluate the effectiveness of our proposed clustering approach w.r.t. i) the cost of event dissemination, ii) the adaptability to dynamic changes in the event workload and iii) the resilience to peer churn. Similar to [5], we define *notification cost* as the ratio of the complete traffic generated in the system to the number of subscriptions matched by the disseminated events. The traffic includes all the control overhead due to hierarchical organization, distributed embedding, k-means algorithm, maintenance of dissemination structure for each cluster and subscription forwarding. The notification cost expresses the efficiency of the event dissemination such that lower cost means higher efficiency. We compare two variants of our proposed approach (with and without event filtering in the clusters) with an overlap metric (OV) based approach. The non-filtering variant uses multicast to disseminate events within each cluster, whereas the filtering variant uses subscription forwarding to further reduce false positives. Both variants use a two-level hierarchical organization to perform dimensionality reduction and calculate subscription clusters. The OV-RTree approach implements a distributed R-tree similar to [7, 39]. Figure 3(k) shows the notification cost during the evolution of the system as more and more events are disseminated. The measurements are taken after every 50 events. To evaluate adaptability, the hotspots of the event distribution are completely changed after every 2,000 events. The figure shows that immediately after the change in event workload, the notification cost of the proposed variants rises significantly for a small transient period, mainly because of the control overhead to calculate and install new clusters. However, the new clusters obtained after the transient period provide almost identical performance increase. Moreover, the figure depicts that the OV-RTree approach shows almost unchanged performance and is not responsive to the changes in the workload. Nevertheless, the proposed filtering and non-filtering variants shows 34% and 18% improvement in the cost of event dissemination (accounting all the control overhead associated with the P2P based implementation of the approach) respectively in comparison to the OV-RTree approach.

Figure 3(l) shows the average notification cost experienced by the OV-RTree and our proposed approach in the presence of continuously arriving and leaving subscriptions. The churn percentage is relative to the total number of peers in the system. For instance, for a total of $2,000$ peers, a churn of 10% means that in each time step, 200 online peers leave the system and the same number of new peers join the system. The churn percentages of 10% and 16% can be seen as worst case scenario that puts the system under very stressing condition. The figure shows that the performance of both the approaches degrades gracefully with the increase in continuous churn. However, our proposed approach is more resilient to churn than OV-RTree, the reason being that the OV-RTree degrades over time (especially, if the newly arriving subscriptions are not added from the root) resulting in a higher rate of false positives and thus higher notification cost. In our proposed system the periodic recalculation and installation of new clusters avoids the gradual degradation and thus the rate of false positives remains low over time.

Table 1: Parameters for scalability evaluations

Parameter	Value
Dim. of embedded space	50
% of landmarks	25%
No. of clusters	50
Max. levels of hierarchy	4
Max No. of L_0 *Gps*	64
Clustering method	Hierarchical

6.4 Scalability of cluster management

In the previous section, we have shown the scalability of the system in terms of the cost of event dissemination under dynamically changing workload. In this section, we evaluate the scalability of our proposed mechanisms w.r.t. i) the overall time to cluster increasingly large number of subscriptions and ii) the computational load on the peers participating in the hierarchical organization. Furthermore, we compare the computational times and the quality of clusters obtained by the linear and non-linear methods.

The matrix algebra needed to perform clustering is implemented using the JAMA library [22], a linear algebra package for Java. All the measurements are made on a 2.7 GHz Intel core i7 CPU with 4GB RAM, running a 64 bit Windows 7 operating system. The delays between the communication links are chosen rather conservatively in the range [224 ms, 384 ms].

Figure 4(a) shows the cost in terms of time to perform the dimensionality reduction[6] for different percentages of landmark subscriptions and levels of hierarchy (one level means the centralized approach). For a hierarchy with 2 or more levels, the time is measured from the instant the root starts the dimensionality reduction process till the low dimensional coordinates of all the subscriptions are calculated. Thus the overall cost includes the time to perform local and global embeddings as well as communication delays between the peers along the longest (or the slowest) branch of the hierarchy. The number of subscriptions is fixed to $2,000$ for this experiment. In the figure some values for linear embedding (RAssoc) are not visible because of their smaller magnitude. The figure shows that the distributed approach drastically decreases the time to perform dimensionality reduction in comparison to the centralized case. For instance, even for 1-level hierarchy with 50% landmarks the calculation time is decreased by 86% and 90% for linear (RAssoc) and non-linear (NCut) methods respectively. More than 99% decrease in time can be achieved with 4-level hierarchy. Moreover, the Figure 4(a) shows that the calculation time in general decreases with a decreasing percentage of landmark subscriptions and with an increasing number of levels of the hierarchy. Because of the decrease in landmarks and increase in levels, this reduction in time comes with a slight decrease in accuracy (cf. *Section 6.1*). However, with up to 4 levels of hierarchy and with only 25% landmark subscriptions, the clustering can be performed with reasonable accuracy (cf. *Figures 4(d) and (e)*). Another important observation is that the linear embedding (RAssoc) is less expensive compared to the non-linear embedding (NCut). For instance, in the centralized scenario RAssoc takes approximately 18% less time. The computationally intense generalized eigensystem problem is mainly responsible for the additional time in case of NCut. The difference in time (to perform dimensionality reduction) between RAssoc and NCut widens with the increase in levels of hierarchy. The reason is that in RAssoc, only landmark subscriptions are used to calculate the projection matrix, whereas in NCut, the projection matrix is calculated by performing least square optimization w.r.t. all the subscriptions in the parent group (cf. *Section 4.2*).

Figure 4(b) shows the overall time to perform dimensionality reduction and clustering in a distributed manner for different numbers of subscriptions. The parameters used in the experiment are specified in Table 1. The time is measured from the instant the root starts the clustering process till the clusters are calculated and clustering information is distributed among subscriber peers. This does not include the time to perform local embeddings because of the fact that the local embeddings are not calculated during the clustering (and projection) process (cf. *Section 5*). Figure 4(b) shows that even for computationally expensive NCut, up to $32,000$ subscriptions can be clustered in less than 15 seconds including the communication delays. On the other hand, clustering using RAssoc is very efficient and takes approximately 5 seconds. The time efficiency of RAssoc is due to the computationally less expensive eigen decomposition problem and the use of only landmark subscriptions for the calculation of the projection matrix as mentioned above. Moreover, Figure 4(b) depicts that the distributed k-means clustering itself is computationally inexpensive and most of

[6]In this experiment we solely focus on dimensionality reduction because it is computationally the most expensive operation.

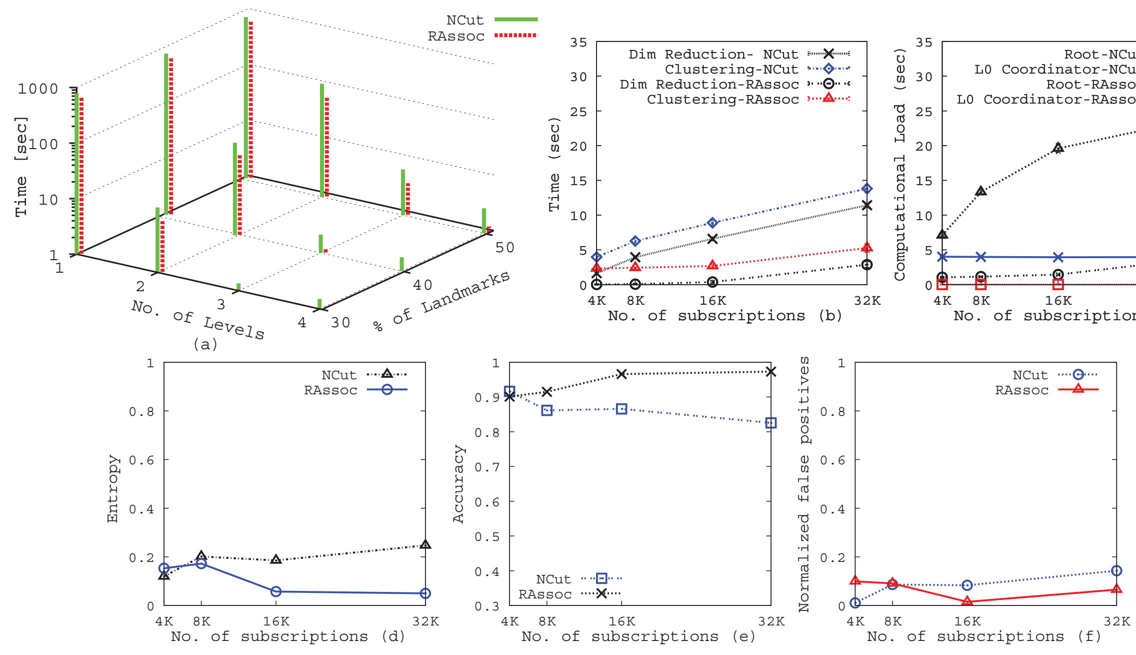

Figure 4: Evaluations for distributed clustering approach

the clustering time comprises communication delays during the merge and selection phases (cf. *Section 4.3*), and the distribution of the cluster centers among subscribers.

Figure 4(c) shows the computational load experienced by different peers participating in the hierarchical organization w.r.t. the total number of subscriptions to cluster. The computational load of a peer is calculated by adding up times to perform local embedding, projection (global embedding) and clustering at each level of the hierarchy where the peer participates. In particular, the computational load of the root which participates at all levels of the hierarchy is compared with the load on the coordinators of lowest-level (L_0) groups. Figure 4(c) shows that the L_0 coordinators experience a computational load of less than 5 seconds in the case of NCut, whereas the load is negligible for RAssoc. Moreover, the load for L_0 coordinators stays constant with the number of subscriptions. This behavior occurs because the peers can scale up the hierarchy to manage only a small number of subscriptions e.g. by dividing a group into two groups, each managing a smaller number of subscriptions[7]. The root, on the other hand, participates at higher level groups and the number of subscriptions managed at each higher group depends on the subscriptions in the subtree. For instance, in case of $4,000$ subscriptions in the system, 25% landmarks and 4-level hierarchy, the highest-level (root) group manages only 63 subscriptions, whereas for $32,000$ system wide subscriptions, the number rises to 500. Therefore, the computational load of the root rises with the increase in the number of subscriptions. We argue that even for $32,000$ subscriptions, the computational load of the root is negligible in case of RAssoc and is easily manageable (under 25 seconds) for NCut. However, if the reduction in the load of the root (or the coordinators at the intermediate levels) is still desirable there are two possible strategies that can be employed. First, the requirement that all the embedding calculations are performed by the coordinator of a group can be easily removed by allowing the coordinator to assign another participant peer from the group to perform calculations[8]. This way the root can offload its computational load to different peers at different levels of the hierarchy. Second, the computationally expensive local embedding calculations can be optimized to reduce their time. The local embeddings are calculated by the coordinator peers in a centralized fashion (cf. *Section 4.2*) and therefore the optimizations available for reducing the time of centralized spectral mechanisms [27], [16] can be directly applied.

Figures 4(d) and (e) show the quality of the clusters created during the evaluations of Figure 4(b). It is clear from the figures that the quality of clusters obtained by NCut decreases slightly with the increase in the number of subscriptions. NCut preserves the geometries at local neighborhood and is sensitive to the weights between the neighbors in the similarity graph (cf. *Equation 4*). With the increase in the number of subscriptions, the number of groups in the hierarchical organization also increases (to scale with the computational load as mentioned above) and therefore the neighborhood information across the groups at the same level degrades, resulting in some loss of cluster quality. However, the quality of clusters obtained by NCut can be enhanced by increasing the percentage of landmark subscriptions. In the case of RAssoc, the quality of the clusters improves with the number of subscriptions. This is because RAssoc does not preserve the geometries at the neighborhood but rather captures the global structure and therefore benefits from the

[7]Dynamic group management in hierarchical overlay networks is a well-researched topic [6, 7]. In our evaluations, a modified version of the NICE protocol [6] is used for managing hierarchy.

[8]The selection of another peer can be done in a random fashion or by considering some appropriate performance metrics. We envision study of such a strategy as future work.

number of subscriptions used as landmarks at the root. An increase in the number of subscriptions in the system results in a large number of landmarks managed by the root and thus in an improvement in the quality of clusters.

Finally, Figure 4(f) shows the quality of the created clusters w.r.t. the false positives in the system. The figure displays similar behavior as depicted by Figures 4(d) and (e). The evaluated scenario shows that NCut, which is computationally more expensive, is slightly better at reducing false positives for a moderate number of subscriptions, whereas RAssoc, which is computationally less expensive, performs relatively better in the presence of a very large number of subscriptions.

7. RELATED WORK

In the past few years, many content-based publish/subscribe systems have been developed [28, 5, 9, 12, 23, 32, 8]. The main goal is to preserve the scalability of the system while at the same time guaranteeing the expressiveness of the subscription model. Clustering subscribers with similar interests has been identified as a promising technique to achieve scalability [7, 31, 10, 28, 25, 12, 35, 20, 37]. Sub-2-Sub [37] completely eliminates false positives by clustering subscribers with non-intersecting subscriptions. However, the number of clusters depends on the run-time interactions between the subscribers that are active in the system and are impossible to limit [30]. Tariq et al. [34, 33] on the other hand, allow individual subscribers to dynamically adjust their rate of false positives according to their delay constraints and bandwidth requirements, but assume predefined set of attribute in the subscription model. DR-Tree [7, 3] uses distributed version of R-Trees to arrange subscribers in height balance topology, but restricts the expressiveness of the subscriptions to a small set of numeric attributes. Anceaume et al. [2] build separate trees for every attribute in the system. Published events are disseminated on every tree with matching attribute, which results in large number of false positives. Semcast [28] groups subscribers into semantic multicast channels for disseminating events, but this requires central coordination. Similarly, Riabov et al. [31] propose offline approaches to group subscribers into limited number of multicast channels using techniques from data mining. Our work differs from the current state of the art clustering approaches in four important ways. First, most of the existing approaches either assumes point (topic-based) subscriptions or only consider the structural (absolute) similarities between the subscriptions. For instance [29, 19, 4, 13, 26] address clustering in a topic based publish/subscribe systems. Similarly [7, 3, 31, 37, 10, 34] assume a predefined set of numeric attributes restricting the expressiveness of the subscription model. Second, the current load of the system in terms of event traffic is mostly neglected [7, 10, 37] and thus the efforts to place two subscriptions in a cluster are wasted if they will not be matched by any event [30]. Third, the approaches which take into account the current event load of the system are centralized [31, 28, 25, 12]. They assume the presence of a central coordinator which keeps track of the changing subscriptions and event load in the system. Furthermore, these approaches assume the presence of a dedicated network of brokers [12, 25, 5], which is fundamentally different from the P2P architecture of our work. Fourth, spectral clustering in the context of publish/subscribe systems has not been addressed previously in the literature.

Spectral clustering has been shown to be more effective than traditional mechanisms such as k-means [24]. However, nearly all existing spectral approaches are centralized and hence they cannot be directly applied in a distributed settings. Dhillon et al. [16] developed a centralized multilevel approach to perform clustering using different graph partitioning objective functions without eigenvector computation. Ning et al. [27] proposed an incremental but centralized algorithm to project new samples in non-linear space without recalculating the whole embedding. Fowlkes et al. [18] reduced the overhead of Normalized cut by first solving the problem for a small random subset of data points and then extrapolating this solution to the full data set. However, the orthogonalization of eigenvectors require complex calculations which cannot be performed in a distributed manner. Our approach for non-linear embedding is inspired from the work of Fang et al. [17]. Similar to our work, Fang et al. perform non-linear dimensionality reduction by minimizing the least square error. However, Fang et al. approach is offline and centralized, which cannot be directly applied in distributed settings. Moreover, Fang et al. consider Ratio Cut [16] as a graph partitioning objective function, which results in a different eigenvalue problem in comparison to the Normalized Cut and Ratio Association objective functions addressed in this paper. Nevertheless, none of the approaches is targeted to handle continuously evolving workloads as is the case in P2P based systems.

8. CONCLUSIONS

In this paper we have presented an approach to perform spectral clustering in a distributed manner. Although we focused on subscription clustering in a content-based publish/subscribe system, the proposed methods are general and can be applied to other areas such as document clustering or data mining. We have identified different spectral methods to perform subscription clustering and adapted them to work in distributed settings. In particular, we have developed mechanisms to perform linear and non-linear dimensionality reduction as well as k-means clustering using the hierarchical overlay organization. The evaluations show that our distributed mechanisms can i) drastically reduce the time ($86\% - 99\%$) to perform clustering, ii) effectively identify good quality clusters, and iii) significantly reduce the cost of event dissemination ($18\% - 34\%$) in a content based publish/subscribe system.

9. ACKNOWLEDGMENTS

This work was partially funded by the SpoVNet project [38] of Baden-Wurttemberg Stiftung gGmbH.

10. REFERENCES

[1] PeerSim. A P2P Simulator:. http://peersim.sourceforge.net/.

[2] E. Anceaume, M. Gradinariu, A. K. Datta, G. Simon, and A. Virgillito. A semantic overlay for self-peer-to-peer publish/subscribe. In *ICDCS*, 2006.

[3] L. Arantes, M. G. Potop-Butucaru, P. Sens, and M. Valero. Enhanced DR-Tree for low latency filtering in publish/subscribe systems. In *AINA*, 2010.

[4] R. Baldoni, R. Beraldi, V. Quema, L. Querzoni, and S. Tucci-Piergiovanni. Tera: topic-based event routing for peer-to-peer architectures. In *DEBS*, 2007.

[5] R. Baldoni, R. Beraldi, L. Querzoni, and A. Virgillito. Efficient publish/subscribe through a self-organizing broker overlay and its application to SIENA. *The Comp. Journal*, 2007.

[6] S. Banerjee, B. Bhattacharjee, and C. Kommareddy. Scalable application layer multicast. *SIGCOMM Comput. Commun. Rev.*, 2002.

[7] S. Bianchi, P. Felber, and M. G. Potop-Butucaru. Stabilizing distributed R-trees for peer-to-peer content routing. *IEEE Trans. Parallel Distrib. Syst.*, 21:1175–1187, 2010.

[8] J. A. Briones, B. Koldehofe, and K. Rothermel. Spine : Adaptive publish/subscribe for wireless mesh networks. *Studia Informatika Universalis*, 7, 2009.

[9] F. Cao and J. P. Singh. Efficient event routing in content-based publish-subscribe service networks. In *INFOCOM'04*.

[10] E. Casalicchio and F. Morabito. Distributed subscriptions clustering with limited knowledge sharing for content-based publish/subscribe systems. In *NCA*, 2007.

[11] D. Cheng, R. Kannan, S. Vempala, and G. Wang. A divide-and-merge methodology for clustering. *ACM Trans. Database Syst.*, 2006.

[12] A. Cheung and H.-A. Jacobsen. Green resource allocation algorithms for publish/subscribe systems. In *ICDCS*, 2011.

[13] G. Chockler, R. Melamed, Y. Tock, and R. Vitenberg. Spidercast: a scalable interest-aware overlay for topic-based pub/sub communication. In *Proceedings of the inaugural international conference on Distributed event-based systems (DEBS)*, pages 14–25. ACM, 2007.

[14] T. F. Cox and M. Cox. *Multidimensional Scaling, Second Edition*. Chapman and Hall/CRC, 2000.

[15] S. E. Deering. Multicast routing in internetworks and extended LANs. *SIGCOMM Comput. Commun. Rev.*, 1988.

[16] I. S. Dhillon, Y. Guan, and B. Kulis. Weighted graph cuts without eigenvectors a multilevel approach. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2007.

[17] H. Fang, S. Sakellaridi, and Y. Saad. Multilevel manifold learning with application to spectral clustering. In *CIKM*, 2010.

[18] C. Fowlkes, S. Belongie, F. Chung, and J. Malik. Spectral grouping using the nyström method. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26, 2004.

[19] S. Girdzijauskas, G. Chockler, Y. Vigfusson, Y. Tock, and R. Melamed. Magnet: practical subscription clustering for internet-scale publish/subscribe. In *DEBS*, 2010.

[20] M. Guimarães and L. Rodrigues. A genetic algorithm for multicast mapping in publish-subscribe systems. In *Proc. of IEEE Sym on Network Computing and Applications*, 2003.

[21] K. Hammouda and M. Kamel. HP2PC: Scalable hierarchically-distributed peer-to-peer clustering. In *SDM'07*.

[22] J. Hicklin, C. Moler, P. Webb, R. F. Boisvert, B. Miller, R. Pozo, and K. Remington. Jama library. http://math.nist.gov/javanumerics/jama/.

[23] K. R. Jayaram and P. Eugster. Split and subsume: Subscription normalization for effective content-based messaging. In *ICDCS*, 2011.

[24] U. Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 2007.

[25] A. Majumder, N. Shrivastava, R. Rastogi, and A. Srinivasan. Scalable content-based routing in pub/sub systems. In *INFOCOM*, 2009.

[26] M. Matos, A. Nunes, R. Oliveira, and J. Pereira. StAN: exploiting shared interests without disclosing them in gossip-based publish/subscribe. In *IPTPS*, 2010.

[27] H. Ning, W. Xu, Y. Chi, Y. Gong, and T. S. Huang. Incremental spectral clustering by efficiently updating the eigen-system. *Pattern Recogn. (Elsevier)*, 2010.

[28] O. Papaemmanouil and U. Cetintemel. Semcast: Semantic multicast for content-based data dissemination. In *ICDE'05*.

[29] J. A. Patel, E. Rivière, I. Gupta, and A.-M. Kermarrec. Rappel: Exploiting interest and network locality to improve fairness in publish-subscribe systems. *Comput. Netw.*, 2009.

[30] L. Querzoni. Interest clustering techniques for efficient event routing in large-scale settings. In *DEBS*. ACM, 2008.

[31] A. Riabov, Z. Liu, J. L. Wolf, P. S. Yu, and L. Zhang. Clustering algorithms for content-based publication-subscription systems. In *ICDCS'02*.

[32] A. Tariq, B. Koldehofe, G. Koch, and K. Rothermel. Providing probabilistic latency bounds for dynamic publish/subscribe systems. In *Proceedings of the 16th ITG/GI Conference on Kommunikation in Verteilten Systemen (KiVS)*. Springer, 2009.

[33] M. A. Tariq, G. G. Koch, B. Koldehofe, I. Khan, and K. Rothermel. Dynamic publish/subscribe to meet subscriber-defined delay and bandwidth constraints. In *Euro-Par*, 2010.

[34] M. A. Tariq, B. Koldehofe, G. G. Koch, I. Khan, and K. Rothermel. Meeting subscriber-defined QoS constraints in publish/subscribe systems. *Concurrency and Computation: Practice and Experience*, 2011.

[35] S. Tarkoma. Dynamic content-based channels: meeting in the middle. In *DEBS*, 2008.

[36] J. Venna and S. Kaski. Local multidimensional scaling. *Neural Netw.*, 2006.

[37] S. Voulgaris, E. Rivière, A.-M. Kermarrec, and M. van Steen. Sub-2-sub: Self-organizing content-based publish and subscribe for dynamic and large scale collaborative networks. In *IPTPS*, Feb 2006.

[38] O. P. Waldhorst, C. Blankenhorn, D. Haage, R. Holz, G. G. Koch, B. Koldehofe, F. Lampi, C. P. Mayer, and S. Mies. Spontaneous Virtual Networks: On the road towards the Internet's Next Generation. *it - Information Technology*, 2008.

[39] E. Yoneki and J. Bacon. eCube: hypercube event for efficient filtering in content-based routing. In *OTM conf. (2)*, 2007.