

HP2PC: Scalable Hierarchically-Distributed Peer-to-Peer Clustering

Khaled M. Hammouda*

Mohamed S. Kamel†

Abstract

In distributed data mining models, adopting a flat node distribution model can affect scalability. To address the problem of modularity, flexibility and scalability, we propose a hierarchically-distributed peer-to-peer architecture and algorithm for data clustering (HP2PC). The architecture is based on a multi-layer overlay network of peer neighborhoods. Supernodes, which act as representatives of neighborhoods, are recursively grouped to form higher level neighborhoods. Peers at a certain level of the hierarchy cooperate within their respective neighborhoods to perform clustering. Using this model, we can partition the clustering problem in a modular way, solve each part individually, then successively combine clusterings up the hierarchy where increasingly global solutions are computed. The algorithm was applied to a distributed document clustering problem and achieved decent speedup with comparable clustering quality to the centralized approach.

Keywords: distributed data mining, peer-to-peer clustering, hierarchically-distributed clustering.

1 Introduction

A recent shift towards distributed data mining (DDM) was sparked when it was realized that analyzing massive distributed data sets using traditional centralized approaches can be intractable. Huge data sets are being collected daily in different fields but it is still extremely difficult to draw conclusions based on the collective characteristics of disparate data.

Four main approaches for performing DDM can be identified. A common approach is to bring the data to a central site, then apply centralized data mining on the collected data. Such approach clearly suffers from a huge communication and computation cost to pool and mine the global data.

A smarter approach is to perform local mining at each site to produce a local model. All local models can then be transmitted to a central site that combines them into a global model [1]. Ensemble methods also

fall into this category [10]. While this approach may not scale well with the number of sites, it is a better solution than pooling the data.

Another approach is for each site to carefully select a small set of representative data objects and transmit it to a central site, which combines the local representatives into one global representative data set. Data mining can then be carried on the global representative data set [7].

All previous three approaches involve a central site to facilitate the DDM process. A more departing approach does not involve any centralization, and thus belongs to the peer-to-peer (P2P) class of algorithms. In P2P DDM, sites communicate directly with each other to perform the data mining task [4, 2]. Communication in P2P networks can be very costly if care is not taken to localize traffic.

In this paper we introduce a hybrid approach for distributed clustering, based on a hierarchically-distributed architecture. The goal is to achieve modularity and scalability. The model is called Hierarchically-distributed Peer-to-Peer Clustering (HP2PC). It involves a hierarchy of P2P neighborhoods, in which each neighborhood is responsible for building a clustering based on the data it has access to in a P2P fashion. As we go up the hierarchy, peers merge clusters from lower levels in the hierarchy. At the root one global clustering is computed. Experiments performed on document clustering show that we can achieve comparable results to centralized clustering with better gain in speedup.

The model lends itself to real-world scenarios, such as hierarchically distributed organizations. In such scenario, different units can perform local clustering to draw conclusions upon their own data. Parent units can combine results from those in lower levels to draw conclusions on a more holistic view of the data.

The paper is organized as follows. Section 2 provides some background on the topic and identifies related work. Section 3 introduces the HP2PC architecture, and section 4 discusses the foundation behind the HP2PC algorithm. Section 5 presents experimental results. Finally, conclusion and future directions are presented in the last section.

*Systems Design Engineering, PAMI Group, University of Waterloo, ON, Canada. Email: hammouda@pami.uwaterloo.ca.

†Electrical & Computer Engineering, PAMI Group, University of Waterloo, ON, Canada. Email: mkamel@pami.uwaterloo.ca.

2 Background and Related Work

The central assumption in DDM is that data is distributed over a number of sites, and that it is desirable to derive, through data mining techniques, a global model that reflects the characteristics of the whole data set. A number of challenges (often conflicting) arise in DDM, including communication complexity, quality of global model, and privacy of local data.

Communication models. The data communicated between nodes in distributed clustering algorithms can be categorized into: (i) models, (ii) representatives, and (iii) raw data. The first case involves calculating local models, such as cluster centroids, *e.g.* P2P k -means [2], that are sent to peers or a central site. In the second case, nodes select a number of representative samples of the local data to be sent to a central site for global model generation, such as the case in the KDEC algorithm [7]. The last case involves exchange of actual data objects to facilitate construction of clusters that exist in certain sites only, such as the case in the collaborative clustering scheme in [6].

Distributed Text Mining. Applications of DDM in the text mining area such as text classification and clustering have received little attention. The work presented by Eisenhardt *et al.* [4] achieves document clustering using a distributed peer-to-peer network. They use the k -means clustering algorithm, modified to work in a distributed P2P fashion using a probe-and-echo mechanism. A similar system can be found in [9], but the problem is posed from the information retrieval point of view. In this work, a subset of the document collection is centrally partitioned into clusters, for which “cluster signatures” are created. Each cluster is then assigned to a node, and later documents are classified to their respective clusters by comparing their signature with all cluster signatures.

State of the art. In the latest issue of IEEE Internet Computing [8] (at the time of writing this paper), a few algorithms were presented representing the state of the art in DDM. Datta *et al.* [2] described an exact local algorithm for monitoring a k -means clustering, as well as an approximate local k -means clustering algorithm for P2P networks. The P2P k -means algorithm finds its roots in a parallel implementation of k -means proposed by Dhillon and Modha [3]. Although the k -means monitoring algorithm does not produce a distributed clustering, it helps a centralized k -means process know when to recompute the clusters by monitoring the distribution of centroids across peers, and triggering a re-clustering if the data distribution significantly changes over time. On the other hand, the P2P k -means algorithm in works by updating the centroids at each peer based on information received from their

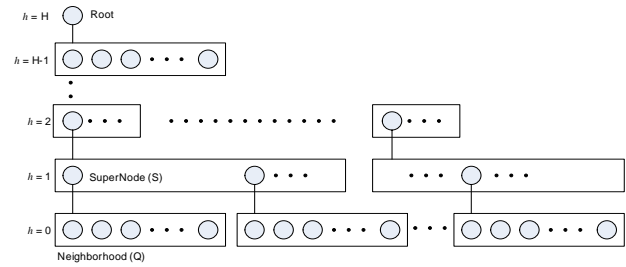


Figure 1: The HP2PC Hierarchy Architecture

immediate neighbors. The algorithm terminates when the information received does not result in significant update to the centroids of all peers.

3 The HP2PC Architecture

HP2PC is a hierarchically-distributed P2P architecture for scalable distributed clustering. Central to this hierarchical architecture design is the formation of peer *neighborhoods*, each is a group of peers forming a logical unit of isolation in an otherwise unrestricted open P2P network. Peers communicate only within neighborhood boundaries. Each neighborhood has a node designated as a *supernode*. Communication between neighborhoods is achieved through their respective supernodes. This model reduces flooding problems usually encountered in large P2P networks. This structure continues recursively to construct a multi-level overlay hierarchy of peers (figure 1).

3.1 Hierarchical Overlays A P2P network is comprised of a set of peers $\mathbf{P} = \{p_i : 1 \leq i \leq N_P\}$. An *overlay* network is a logical network on top of \mathbf{P} that connects a certain subset of \mathbf{P} . Most work in the P2P literature refers to a single level of overlay; in our work, this concept is extended further to allow multiple overlay levels.

To distinguish between the different levels of overlay, we will use the *height* parameter. An overlay network at height h is denoted $\mathbf{P}^{(h)}$. The lowest level network (the original P2P network) is at height $h = 0$, thus denoted $\mathbf{P}^{(0)}$; while the highest overlay possible is at height H , denoted $\mathbf{P}^{(H)}$, and consists of a single node (the *root* of the hierarchy). In subsequent formulations, we will drop the superscript (h) if it can be inferred from context.

The size of the overlay network at each level will differ according to how many peers comprise the overlay. However, since a higher level overlay will always be a subset of its immediate lower level overlay, we maintain

that:

$$0 < |\mathbf{P}^{(h)}| < |\mathbf{P}^{(h-1)}|, \forall h > 0.$$

The choice of which subset of peers forms the next level overlay is closely related to the next subsection discussing peer neighborhoods.

3.2 Neighborhoods A neighborhood, N , is a subset of an overlay \mathbf{P} , and has a designated peer known as a *supernode*, p_s ; thus

$$N = (Q, p_s) : Q \subseteq \mathbf{P}, p_s \in Q.$$

The following neighborhood properties are enforced for the HP2PC architecture:

- A set of neighborhoods, $\mathbf{N} = \{N_j : 1 \leq j \leq N_N\}$, covers the network \mathbf{P} ; *i.e.* $\mathbf{P} \equiv \bigcup_{j=1}^{N_Q} Q_j$.
- Neighborhoods do not overlap; *i.e.* $\forall i, j \neq i : Q_i \cap Q_j = \emptyset$
- Any nodes must belong to some neighborhood; *i.e.* $\nexists p_i : p_i \notin Q_j, \forall Q_j \in \mathbf{Q}$

A network partitioning factor, $\beta \in \mathbb{R} : 0 \leq \beta \leq 1$, partitions the P2P network into equally sized neighborhoods; *i.e.*

$$N_Q = 1 + \beta(N_P - 1)$$

The size of each neighborhood as a fraction of the size of the network is calculated as

$$\forall j : |Q_j| = N_P/N_Q, 1 \leq j \leq N_Q.$$

Peer hierarchy formation is achieved recursively bottom-up, thus level $h = 1$ peers are the supernodes of level 0 neighborhoods. The root supernode will be at level H , and denoted $p^{(H)}$. If we apply the same network partitioning factor to every level of the hierarchy, we can deduce the height of the hierarchy as $H = \log N_P$.

4 The HP2PC Algorithm

The HP2PC algorithm is a distributed iterative clustering process that seeks a set of common centroids across peers. Each neighborhood converges to a distinct set of centroids, which are acquired by the supernode of that neighborhood. The supernode, in turn as part of its higher level neighborhood, collaborates with its peers to form a set of centroids for that higher-level neighborhood. This process continues hierarchically until one set of centroids are generated at the root of the hierarchy.

4.1 Estimating Clustering Quality The distributed search for cluster centroids is guided by a cluster quality measure that estimates intra-cluster cohesiveness and inter-cluster separation. The distribution of pair-wise similarities within a cluster is represented using a *cluster similarity histogram*, which is a concise statistical representation of the cluster tightness [5].

Let $\text{sim}(\cdot)$ be a similarity measure between two objects, and S_c be the set of pair-wise similarity between objects of cluster c :

$$(4.1a) \quad S_c = \{s_k : 1 \leq k \leq |c|(|c| + 1)/2\}$$

$$(4.1b) \quad s_k = \text{sim}(d_i, d_j), \quad d_i, d_j \in c$$

where $|c|$ is the number of the objects in a cluster. The histogram of the similarities in the cluster is represented as:

$$(4.2a) \quad H_c = \{h_i : 1 \leq i \leq B\}$$

$$(4.2b) \quad h_i = \text{count}(s_k),$$

$$(4.2c) \quad s_k \in S_c, \delta \cdot (i - 1) \leq s_k < \delta \cdot (i)$$

where B : the number of histogram bins,
 h_i : the count of similarities in bin i , and
 δ : the bin width of the histogram.

To estimate the cohesiveness of cluster c , we calculate the histogram *skew*, the third central moment of the distribution. A positive skew indicates a longer tail in the higher interval of the histogram, and vice versa. A negatively-skewed similarity histogram indicates a tight cluster. Skew is calculated as

$$(4.3) \quad \text{skew} = \frac{\sum_i (x_i - \mu)^3}{N\sigma^3}$$

so, the tightness of a cluster, $\varphi(c)$, calculated as the skew of its histogram H_c , is

$$(4.4) \quad \varphi(c) = \text{skew}(H_c) = \frac{\sum_k (s_k - \mu_{S_c})^3}{|S_c|\sigma_{S_c}^3}, \quad s_k \in S_c.$$

A clustering quality measure based on skewness of similarity histograms of individual clusters can be derived as a weighted average of the individual clusters skew:

$$(4.5) \quad \varphi(C) = \frac{\sum_i |c_i| \varphi(c_i)}{\sum_i |c_i|}, \quad c_i \in C.$$

4.2 Distributed Clustering (Level $h = 0$) We define a general function for updating cluster models in a fully-connected neighborhood:

$$(4.6a) \quad C_i^t = f(\{C_j^{t-1}\}), \quad i, j \in Q$$

$$(4.6b) \quad C_i^0 = C^0$$

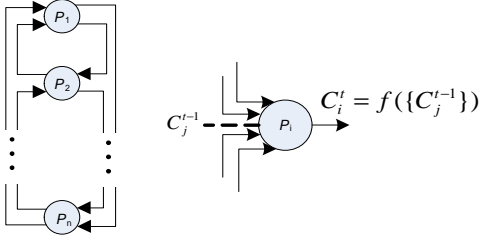


Figure 2: Distributed Clustering Iterative Loop

where C_i^t is the clustering model (usually a set of centroids) calculated by peer i at iteration t , and $f(\cdot)$ is an aggregating function. The equation can be illustrated by figure 2, where the output of each peer at iteration t depends on the models calculated by all other peers at iteration $t - 1$. In P2P k -means [2], $f(\cdot) \equiv \text{avg}(\cdot)$, and the neighborhood is based on network topology.

Algorithm 1 shows the HP2PC algorithm, which iteratively updates the cluster centroids at each node (the functions `CalcSimilarityMatrix` and `CalcSkew` are not shown for paper size constraints). For each neighborhood an initial set of centroids, m^0 , is calculated by the supernode and transmitted to all peers in the neighborhood. Like k -means, during each iteration each peer assigns local data to their nearest centroid, and calculates their new centroids. In addition, it also calculates cluster skews using equation 4.4.

The final set of centroids for each iteration is calculated from all peer centroids. Unlike k -means (or P2P k -means), those final centroids are weighed by the skew and size of clusters at individual peers. At the end of each iteration, peers broadcast cluster centroid, skew, and size information:

$$C_j^t \equiv (m_{jk}, \varphi_{jk}, |c_{jk}|)^t, \quad k \in [1, K]$$

The weight of a cluster k at peer j is defined as:

$$w_{jk} = \varphi(c_{jk}) \cdot |c_{jk}|$$

At peer i , the centroid of cluster k is updated according to the following equation, which favors tight and dense clusters:

$$(4.7) \quad m_{ik}^t = \frac{\sum_j w_{jk}^{t-1} \cdot m_{jk}^{t-1}}{\sum_j w_{jk}^{t-1}}, \quad j \in Q$$

This is followed by assigning objects to their nearest centroid, and calculating the new set of cluster skews, φ_{ik} , and sizes, $|c_{ik}|$, which are used in the next iteration. The algorithm terminates when object assignment does not change, or when $\forall i, k \quad |m_{ik}^t - m_{ik}^{t-1}| < \epsilon$, where ϵ is a sufficiently small parameter.

Algorithm 1 Level 0 Neighborhood Clustering

```

1:  $S = \text{CalcSimilarityMatrix}(D_i)$ 
2:  $\{(m_s, w_s)\} = \text{ReceiveFrom}(p_s) \{p_s: \text{supernode of } Q\}$ 
3:  $\forall j \neq s : \{(m_j, w_j)\} = 0$ 
4:  $\{c, \varphi\} = \text{UpdateClusters}(D_i, \{m_i\})$ 
5: while change in  $\{m_i\} > \epsilon$  do
6:   for all  $p_j \in Q, j \neq i$  do
7:      $\text{SendTo}(p_j, \{(m_i, w_i)\})$ 
8:      $\{(m_j, w_j)\} = \text{ReceiveFrom}(p_j)$ 
9:   end for
10:  for all  $k \in [1, K]$  do
11:     $m_{ik} = 0, w_k = 0$ 
12:    for all  $j \in Q$  do
13:       $m_{ik} = m_{ik} + (m_{jk} \cdot w_{jk})$ 
14:       $w_k = w_k + w_{jk}$ 
15:    end for
16:     $m_{ik} = m_{ik} / w$ 
17:  end for
18:   $\{c, \varphi\} = \text{UpdateClusters}(D_i, \{m_i\})$ 
19: end while

```

Function UpdateClusters($D, \{m\}$)

```

1:  $c = \{\}$ 
2: for all  $d \in D$  do
3:    $l = \text{argmin}_k \{|d - m_k|\}$ 
4:    $c_l \leftarrow \{c_l, d\}$ 
5: end for
6: for all  $c_k$  do
7:    $S_k = S \downarrow c_k \{ \text{part of } S \text{ indexed by objects in } c_k \}$ 
8:    $\varphi_k = \text{CalcSkew}(S_k)$ 
9: end for
10: return  $\{c, \varphi\}$ 

```

4.3 Distributed Clustering (Level $h > 0$) Once a neighborhood converges to a set of clusters, the centroids and weights of those clusters are acquired by the supernode as its initial set of clusters; *i.e.* for neighborhood Q with supernode p_s

$$C_s^{(0,h)} = C_Q^{(T,h-1)}$$

where T is the final iteration of the algorithm at level $h - 1$ for neighborhood Q .

Since at level h of the hierarchy the actual data objects are not available, we rely on *meta-clustering*: merging the clusters using centroid and weight information alone. At level $h > 0$, clusters are merged in a bottom-up fashion, up to the root of the hierarchy; *i.e.* $C^{(h)} = f(C^{(h-1)})$. This means once a neighborhood at level h converges to a set of K centroids, it is frozen, and the higher level clustering is invoked.

To merge the K centroids at higher levels, they are collected and clustered at the supernode, using k -means clustering (algorithm 2). This process repeats until one set of clusters is computed at the root.

A major benefit of this model is the ability to merge

Algorithm 2 HP2PC Clustering

```

1: for all  $Q_i \in \text{Neighborhoods}(h = 0)$  do
2:    $\{m_i\}^{(0)} = \text{NeighborhoodCluster}(Q_i)$ 
3: end for
4: for  $h = 1$  to  $H$  do
5:   for all  $Q_i \in \text{Neighborhoods}(h)$  do
6:     for all  $p_j \in Q_i$  do
7:        $\{m_j\} = \{m_j\}^{(h-1)}$ 
8:       SendTo( $p_s, \{m_j\}$ )
9:     end for
10:     $\{m_i\}^h = \text{kmeans}(\{m_j\})$  {only at peer  $p_s$ }
11:   end for
12: end for
  
```

Table 1: Data Sets

Data Set	Type	# Docs.	Categories
DS1- Yahoo! news	HTML	2340	20
DS2- SchoolNet	Metadata	2371	17
DS3- 20-newsgroups	Usenet	18,828	20

a forest of independent hierarchies into one hierarchy by putting all roots of the forest into one neighborhood and invoking the merge algorithm on that neighborhood.

5 Experimental Results

5.1 Data Sets Three document data sets, listed in Table 1, were used to evaluate the algorithm. DS1 is a collection of 2340 Yahoo! news articles. DS2 is a collection of 3271 metadata records collected from the SchoolNet learning resources website¹. DS3 is the standard 20-newsgroups data set.

5.2 Evaluation Measure We relied on an external evaluation measure, the F-measure, for evaluating the clustering results against the reference categories (ground truth). The **F-measure** combines the *Precision* and *Recall* measures from the information retrieval literature; it is calculated as the harmonic mean of both quantities:

$$(5.8) \quad F = \frac{2PR}{P + R}$$

To find the F-measure for a clustering solution, we compute its value for each class and take its weighted average over all classes. Details of such calculation can be found in [5].

At level $h = 0$, we evaluated the quality of clustering for each neighborhood, with respect to the subset of the data in the neighborhood. At level $h > 0$, we evaluated the clustering computed by a supernode with respect to the data subset reachable from the supernode. Thus,

¹<http://www.schoolnet.ca/>

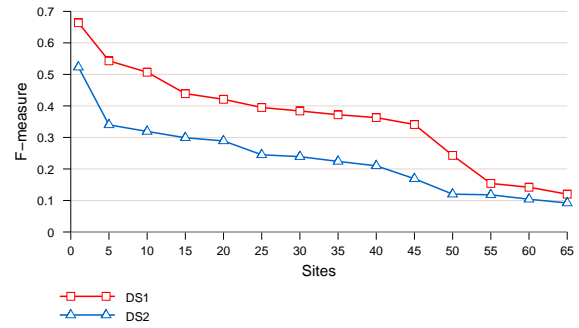


Figure 3: Clustering Quality vs. Number of Sites (DS1 & DS2)

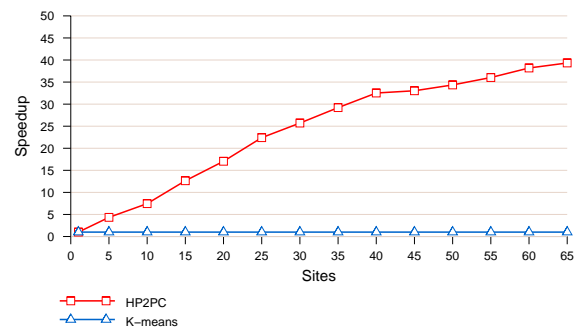


Figure 4: Speedup vs. Number of Sites (DS2)

evaluation of the clustering at the root node reflects the quality with respect to the whole data set.

5.3 Results A simulation environment was built to test the HP2PC architecture. Parameters to the system are the network size and neighborhood size. Data was partitioned randomly over all nodes of the network. Clustering was invoked at level 0 neighborhoods, and was propagated to the root of the hierarchy as described in section 4.

We first evaluate the distributed clustering algorithm at level 0, as described in section 4.2. This is achieved by setting the height of the hierarchy to 1; *i.e.* there is one neighborhood comprised of all nodes in the network, and one supernode (the root).

Figure 3 illustrates the quality of clustering achieved at different number of sites. The trend shows that global clustering quality decreases as we increase the number of sites. The problem lies in that we distribute the data equally among the sites. This increasingly fine-grained partitioning of the data makes it in-

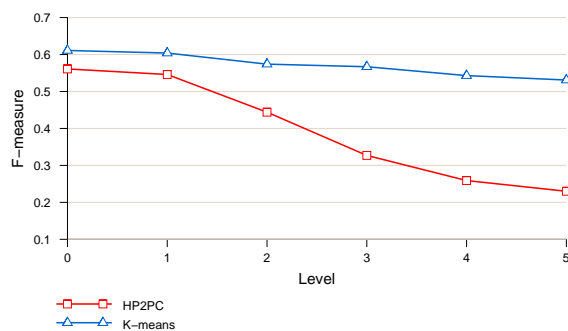


Figure 5: Clustering Quality vs. Hierarchy Level (DS3)

sufficient for each site to compute reliable cluster centroids. An interesting observation is that there is a slight *plateau* between the centralized case ($\#sites = 1$), and a point where the data is finely partitioned ($\#sites > \text{some } s$), after which quality degrades rapidly. This plateau provides a clue on the relation between the data set size and the number of sites, beyond which we should not increase the number of sites without increasing the data set size.

Figure 4 shows the speedup of the distributed algorithm. Speedup is calculated as ratio of time taken in the centralized case to the time taken in the distributed case. The figure clearly shows that the HP2PC algorithm exhibits decent speedup, although not linear.

To test the effect of hierarchy height on clustering quality, we performed experiments on a network of size 250 nodes, height of 5, on DS3, with $\beta = 0.2$ applied to all levels. We compared the results to flat k -means performed at each level of the hierarchy, and taking the average over all neighborhoods on that level. Figure 5 shows the quality achieved at each level, and compares it to the average k -means quality at the same level. We notice that HP2PC degrades in quality as we go up the hierarchy, which is predictable, as we only perform meta-clustering based on cluster centroids of the lower levels. However, it is clear that at level 0 we can achieve very close quality to the centralized k -means algorithm.

We can conclude that there is a tradeoff between quality of clustering and hierarchy height. If our goal is quality, we should maintain a low level hierarchy, optimally setting the network to one neighborhood, sacrificing running time. On the other hand, if our goal is speed, we should aim for fine-grained neighborhoods and taller hierarchies; sacrificing clustering quality.

6 Conclusion and Future Work

In this paper we introduced a novel architecture and algorithm for distributed clustering, the Hierarchically-Distributed Peer-to-Peer Clustering model (HP2PC), which allows building hierarchical networks for clustering data. We demonstrated the flexibility of the model, showing that it achieves comparable quality to standard k -means. For future work, we are investigating the possibility of making clustering bi-directional; *i.e.* clusters at lower level neighborhoods of the hierarchy should be affected by their immediate higher level clusters. We believe that this will create an opportunity for better global clustering solutions, but on the expense of computational complexity.

References

- [1] J.C. da Silva, C. Giannella, R. Bhargava, H. Kargupta, and M. Klusch. Distributed data mining and agents. *Engineering Applications of Artificial Intelligence*, 18(7):791–807, 2005.
- [2] Souptik Datta, Chris Giannella, and Hillol Kargupta. K-means clustering over a large, dynamic network. In *SIAM International Conference on Data Mining (SDM06)*, pages 153–164, 2006.
- [3] Inderjit S. Dhillon and Dharmendra S. Modha. A data-clustering algorithm on distributed memory multiprocessors. In *Large-Scale Parallel Data Mining*, volume 1759 of *LNAI*, pages 245–260. Springer, 2000.
- [4] M. Eisenhardt, W. Muller, and A. Henrich. Classifying documents by distributed p2p clustering. In *Informatik 2003: Innovative Information Technology Uses*, Frankfurt, Germany, 2003.
- [5] K. Hammouda and M. Kamel. Incremental document clustering using cluster similarity histograms. In *The 2003 IEEE/WIC International Conference on Web Intelligence (WI03)*, pages 597–601, Halifax, Canada, October 2003.
- [6] K. Hammouda and M. Kamel. Collaborative document clustering. In *Proceedings of the Sixth SIAM International Conference on Data Mining (SDM06)*, pages 453–463, Bethesda, MD, April 2006.
- [7] Matthias Klusch, Stefano Lodi, and Gianluca Moro. Agent-based distributed data mining: The KDEC scheme. In *AgentLink*, pages 104–122, 2003.
- [8] Anup Kumar, Mehmed Kantardzic, and Samuel Madden. Guest editors’ introduction: Distributed data mining—framework and implementations. *IEEE Internet Computing*, 10(4):15–17, 2006.
- [9] J. Li and R. Morris. Document clustering for distributed fulltext search. In *2nd MIT Student Oxygen Workshop*, Cambridge, MA, August 2002.
- [10] Alexander Strehl and Joydeep Ghosh. Cluster ensembles - a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research*, 3:583–617, December 2002.