Self-Organization in Autonomous Sensor/Actuator Networks

[SelfOrg]

Dr.-Ing. Falko Dressler

Computer Networks and Communication Systems
Department of Computer Sciences
University of Erlangen-Nürnberg

http://www7.informatik.uni-erlangen.de/~dressler/
dressler@informatik.uni-erlangen.de
Overview

- **Self-Organization**
  Basic methodologies of self-organization; comparison of central and hierarchical control, distributed systems, and autonomous behavior; examples of self-organization

- **Mobile Sensor/Actuator Networks**
  Ad hoc routing; reliable communication and congestion control; sensor assistance for mobile robots; applications

- **Coordination of Autonomous Systems**
  Coordination and synchronization; communication aspects; clustering

- **Bio-inspired Mechanisms**
  Swarm intelligence; artificial immune system; intra/inter cellular information exchange
Clustering

- Introduction and classification
- K-means
- LEACH
Clustering

- Clustering can be considered the most important *unsupervised learning* problem; so, as every other problem of this kind, it deals with finding a *structure* in a collection of unlabeled data.

- A loose definition of clustering could be *“the process of organizing objects into groups whose members are similar in some way”*

- A cluster is therefore a collection of objects which are “similar” between them and are “dissimilar” to the objects belonging to other clusters.
Clustering

- **Distance-based clustering**: two or more objects belong to the same cluster if they are “close” according to a given distance (in this case geometrical distance). The “distance” can stand for any similarity criterion.

- **Conceptual clustering**: two or more objects belong to the same cluster if this one defines a concept common to all that objects, i.e. objects are grouped according to their fit to descriptive concepts, not according to simple similarity measures.
Clustering Algorithms

- **Centralized**
  - If centralized knowledge about all local states can be maintained
    → central (multi-dimensional) optimization process

- **Distributed / self-organized**
  - Clusters are formed dynamically
  - A cluster head is selected first
  - Usually based on some election algorithm known from distributed systems
  - Membership and resource-management is maintained by the cluster head
    → distributed (multi-dimensional) optimization process
Applications

- General
  - Marketing: finding groups of customers with similar behavior given a large database of customer data containing their properties and past buying records;
  - Biology: classification of plants and animals given their features;
  - Libraries: book ordering;
  - Insurance: identifying groups of motor insurance policy holders with a high average claim cost; identifying frauds;
  - City-planning: identifying groups of houses according to their house type, value and geographical location;
  - Earthquake studies: clustering observed earthquake epicenters to identify dangerous zones;
  - WWW: document classification; clustering weblog data to discover groups of similar access patterns.

- Autonomous Sensor/Actuator Networks
  - Routing optimization
  - Resource and task allocation
  - Energy efficient operation
Clustering Algorithms

- Requirements
  - Scalability
  - Dealing with different types of attributes
  - Discovering clusters with arbitrary shape
  - Minimal requirements for domain knowledge to determine input parameters
  - Ability to deal with noise and outliers
  - Insensitivity to order of input records
  - High dimensionality
  - Interpretability and usability
Clustering Algorithms

- Problems
  - Current clustering techniques do not address all the requirements adequately (and concurrently)
  - Dealing with large number of dimensions and large number of data items can be problematic because of time complexity
  - The effectiveness of the method depends on the definition of “distance” (for distance-based clustering)
  - If an obvious distance measure doesn’t exist we must “define” it, which is not always easy, especially in multi-dimensional spaces
  - The result of the clustering algorithm (that in many cases can be arbitrary itself) can be interpreted in different ways
Clustering Algorithms

- Classification
  - Exclusive – every node belongs to exactly one cluster (e.g. K-Means)
  - Overlapping – nodes may belong to multiple clusters
  - Hierarchical – based on the union of multiple clusters (e.g. single-linkage clustering)
  - Probabilistic – clustering is based on a probabilistic approach

- Distance measure
  - The quality of the clustering algorithm depends first on the quality of the distance measure!
K-Means

- One of the simplest unsupervised learning algorithms
- Main idea
  - Define $k$ centroids, one for each cluster
    - These centroids should be placed in a cunning way because of different location causes different result, so, the better choice is to place them as much as possible far away from each other
  - Take each point belonging to a given data set and associate it to the nearest centroid - when no point is pending, the first step is completed and an early grouping is done
  - Re-calculate $k$ new centroids as barycenters of the clusters resulting from the previous step
    - A new binding has to be done between the same data set points and the nearest new centroid
  - A loop has been generated. As a result of this loop we may notice that the $k$ centroids change their location step by step until no more changes are done, i.e. the centroids do not move any more
K-Means

- The algorithm aims at minimizing an objective function, in this case a squared error function

\[ J = \sum_{j=1}^{k} \sum_{i=1}^{n} \left\| x_i^{(j)} - c_j \right\|^2 \]

- Where \( \left\| x_i^{(j)} - c_j \right\|^2 \) is a chosen distance measure between a data point \( x_i^{(j)} \) and the cluster centre \( c_j \)

- The objective function is an indicator of the distance of the \( n \) data points from their respective cluster centers
K-Means – Example

- Suppose that we have n sample feature vectors $x_1, x_2, \ldots, x_n$ all from the same class, and we know that they fall into $k$ compact clusters, $k < n$. Let $m_i$ be the mean of the vectors in cluster $i$. If the clusters are well separated, we can use a minimum-distance classifier to separate them. That is, we can say that $x$ is in cluster $i$ if $|| x - m_i ||$ is the minimum of all the $k$ distances. This suggests the following procedure for finding the means:

  - Make initial guesses for the means $m_1, m_2, \ldots, m_k$
  - Until there are no changes in any mean
    - Use the estimated means to classify the samples into clusters
    - For $i$ from 1 to $k$
      - Replace $m_i$ with the mean of all of the samples for cluster $i$

- Here is an example showing how the means $m_1$ and $m_2$ move into the centers of two clusters:

[Demo]
Hierarchical Clustering Algorithms

- Given a set of $N$ items to be clustered, and an $N \times N$ distance (or similarity) matrix, the basic process of hierarchical clustering is this:
  1. Assign each item to a cluster ($N$ items result in $N$ clusters each containing one item); let the distances (similarities) between the clusters the same as the distances (similarities) between the items they contain
  2. Find the closest (most similar) pair of clusters and merge them into a single cluster
  3. Compute distances (similarities) between the new cluster and each of the old clusters
  4. Repeat steps 2 and 3 until all items are clustered into a single cluster of size $N$ (this results in a complete hierarchical tree; for $k$ clusters you just have to cut the $k-1$ longest links)

- This kind of hierarchical clustering is called **agglomerative** because it merges clusters iteratively
Hierarchical Clustering Algorithms

- Computation of the distances (similarities)
  - In **single-linkage clustering** (also called the *minimum* method), we consider the distance between one cluster and another cluster to be equal to the shortest distance from any member of one cluster to any member of the other cluster.
  - In **complete-linkage clustering** (also called the *diameter* or *maximum* method), we consider the distance between one cluster and another cluster to be equal to the greatest distance from any member of one cluster to any member of the other cluster.
  - In **average-linkage clustering**, we consider the distance between one cluster and another cluster to be equal to the average distance from any member of one cluster to any member of the other cluster.

- The main weaknesses of agglomerative clustering methods are:
  - they do not scale well: time complexity of at least $O(n^2)$, where $n$ is the number of total objects.
  - they can never undo what was done previously.
Single-Linkage Clustering Algorithm

- The $N \times N$ proximity matrix is $D = [d(i,j)]$; the clusterings are assigned sequence numbers $0, 1, \ldots, (n-1)$ and $L(k)$ is the level of the $k$-th clustering; a cluster with sequence number $m$ is denoted $(m)$ and the proximity between clusters $(r)$ and $(s)$ is denoted $d[(r),(s)]$.

- The algorithm is composed of the following steps:
  1. Begin with the disjoint clustering having level $L(0) = 0$ and sequence number $m = 0$
  2. Find the least dissimilar pair of clusters in the current clustering, say pair $(r), (s)$, according to $d[(r),(s)] = \min d[(i),(j)]$ where the minimum is over all pairs of clusters in the current clustering
  3. Increment the sequence number: $m = m + 1$. Merge clusters $(r)$ and $(s)$ into a single cluster to form the next clustering $m$. Set the level of this clustering to $L(m)=d[(r),(s)]$
  4. Update the proximity matrix, $D$, by deleting the rows and columns corresponding to clusters $(r)$ and $(s)$ and adding a row and column corresponding to the newly formed cluster. The proximity between the new cluster, denoted $(r,s)$ and old cluster $(k)$ is defined in this way: $d[(k), (r,s)] = \min( d[(k),(r)], d[(k),(s)])$
  5. If all objects are in one cluster, stop. Else, go back to step 2
Case Study: LEACH

- LEACH: Low-Energy Adaptive Clustering Hierarchy

- Capabilities
  - Self-organizing, adaptive clustering protocol that uses randomization to distribute the energy load evenly among the sensors in the network. All nodes organize themselves into local clusters, with one node acting as the local base station or cluster-head.
  - Includes randomized rotation of the high-energy cluster-head position such that it rotates among the various sensors in order to not drain the battery of a single sensor.
  - Performs local data fusion to “compress” the amount of data being sent from the clusters to the base station, further reducing energy dissipation and enhancing system lifetime.
Case Study: LEACH

- **Election process**
  - Sensors elect themselves to be local cluster-heads at any given time with a certain probability
  - The clusterhead nodes broadcast their status to the other sensors in the network
  - Each sensor node determines to which cluster it wants to belong by choosing the cluster-head that requires the minimum communication energy

- **Dynamic clusters**
  - (a) clusterhead nodes $C$ at time $t_1$
  - (b) clusterhead nodes $C'$ at time $t_1 + d$
Case Study: LEACH

- Algorithm details
  - Operation of LEACH is broken into rounds
  - Cluster is initialized during the advertisement phase
  - Configuration during the set-up phase
  - Data transmission during the steady-state phase
Case Study: LEACH

- Advertisement phase
  - Each node decides whether or not to become a cluster head for the current round
    - Based on the suggested percentage of cluster heads for the network (determined a priori), and the number of times the node has been a cluster-head so far
    - The decision is made by the node \( n \) choosing a random number between 0 and 1; if the number is less than a threshold \( T(n) \), the node becomes a cluster-head for the current round
    - The threshold is set as:
      \[
      T(n) = \begin{cases} 
      \frac{P}{1 - P \cdot \left( r \mod \frac{1}{P} \right)} & \text{if } n \in G \\
      0 & \text{otherwise} 
      \end{cases}
      \]
    - where \( P \) is the desired percentage of cluster heads (e.g., \( P = 0.05 \)), \( r \) is the current round, and \( G \) is the set of nodes that have not been cluster-heads in the last \( 1/P \) rounds
  - Using this threshold, each node will be a cluster-head at some point within \( 1/P \) rounds; the algorithm is reset after \( 1/P \) rounds
Case Study: LEACH

- **Cluster-Head-Advertisement**
  - Each node that has elected itself a cluster-head for the current round broadcasts an advertisement message to the rest of the nodes.
  - All cluster-heads transmit their advertisement using the same transmit energy; the non-cluster-head nodes must keep their receivers on during this phase of set-up to hear the advertisements.
  - Each non-cluster-head node decides the cluster to which it will belong for this round based on the received signal strength of the advertisement; tiebreaker: randomly chosen cluster-head.

- **Cluster set-up phase**
  - Each node must inform the cluster-head node that it will be a member of the cluster by transmitting this information back to the cluster-head.
  - The cluster-head node receives all the messages for nodes that would like to be included in the cluster; based on the number of nodes in the cluster, the cluster-head node creates a TDMA schedule that is broadcast back to the nodes in the cluster.
Case Study: LEACH

- Steady-state phase
  - Assuming nodes always have data to send, they send it during their allocated transmission time to the cluster head
  - This transmission uses a minimal amount of energy (chosen based on the received strength of the cluster-head advertisement)
  - The radio of each non-cluster-head node can be turned off until the node’s allocated transmission time, thus minimizing energy dissipation in these nodes
  - The cluster-head node must keep its receiver on to receive all the data from the nodes in the cluster
  - The cluster-head is responsible to forward appropriate messages to the base station; since the base station is far away, this is a high-energy transmission
  - After a certain (a priori determined) time, the next round begins
Case Study: LEACH

- Some measurement results