Challenges in P2P Data Management: Clustering

Vesna Šešum-Čavić
University of Belgrade
Faculty of Civil Engineering
vsesumcavic@grf.bg.ac.rs
www.complang.tuwien.ac.at/vesna
• Associate Professor at University of Belgrade

• Research interests: swarm intelligence, evolutionary computation, network optimization, p2p systems, scalable distributed systems, theory and design of algorithms, combinatorial optimization, complex systems, self-organization, and multi-agent systems

• Chair of IEEE Women in Computational Intelligence in 2019 and member of the IEEE Women in Engineering Committee as well as IEEE CIS Member Activities Committee;

• Associated Editor of IEEE Transactions on Emerging Topics in Computational Intelligence (2019) and IEEE Open Journal of Intelligent Transportation Systems
Data management in Peer-to-Peer (P2P) systems

• This is a challenging issue due to the scale of network and extremely high dynamics

• There are many research issues regarding data management in P2P systems detected as [1]:
  – Indexing
  – Data integration
  – Query processing
  – Data replication
  – Clustering
  – Incentive mechanisms
  – etc.
About Clustering

- A method of unsupervised learning → no training step required
- Grouping collection of observations in smaller subsets

Fig. 1: (a) observations (b) clustered observations [2]
Clustering in P2P (1)

- P2P systems have capability of self-organization and fault-tolerance → a demand for an adaptive network topology due to churn [1].
- Peers in a P2P system are autonomous [10]
  - Therefore, characteristics of P2P systems make clustering a challenging task.
  - Autonomy is violated by data clustering.
- Very dynamic nature of P2P environments [1,6,7]
  - Another concern for the application of clustering.
  - Clusters needs to dynamically adapt to the frequent changes in peer populations and their data.
Clustering in P2P (2)

• Further concern is the lack of global knowledge of data and peer interests
  – A serious difficulty in forming clusters in P2P systems.
• We can differentiate two types of clustering in P2P:
  – Data clustering
  – Peer clustering
Data Clustering

• Data items with common attributes or properties can be grouped together → data clusters.

• The main goal of clustering:
  – To reduce the communication cost in query processing.
  – Related data are placed in nearby locations.

• In structured P2P systems, it is possible to store similar data at the same or neighboring peers by using an order-preserving hash function [3].

• What to do in unstructured P2P systems?
Load Clustering (1)

- Load clustering deals with the clustering of work loads in a computer system.
- We derived it from data clustering [4]:
  - data clustering
    - group and stores similar data
    - rather static
  - load clustering
    - not only attributes of data, but also consideration of the payload
    - group, temporarily store and process similar requests and reply
    - highly dynamic
- Benefits for applications (e.g., better performance)
Load Clustering (2)

• It makes further optimizations of the load distribution based on the content of the load items [4]:
  – A single load item → a task that consists of several attributes (e.g. a certain priority), has a payload, a dynamic life cycle and is handled by a computer or processor.

• The goal:
  – Cluster loads not only on the basis of simple attributes, but also take into consideration the payload as well as the dynamic.
  – Increase performance by allowing a worker in a computer system to process not only a single load at once but a cluster of loads which are similar.
Load Clustering (3)

• Load clustering systems are complex systems
  – They should be self-organizing and adaptive, and capable to flexibly adapt to dynamically changing loads and resources.

• There are many load clustering scenarios
  – Different algorithms and configurations are needed to satisfy different kinds of load clustering scenarios.

• **Self-Initiative Load Clustering Agents (SILCA) [4]**
  a load clustering framework that provides the possibility for plugging and benchmarking different clustering algorithms
  – It is based on autonomous agents with decentralized control and a blackboard based communication mechanism.
Design of a general software architecture framework [2,4]
  - component-based → “plug”-able algorithms and policies
  - pattern-oriented → composition towards different solutions
  - agent-based → adaptive
  - space-based middleware → decoupling, autonomy, agility

Evaluation through benchmarking
  - comparing different
    • algorithms and combinations
    • network topologies
    • parameter settings
SILCA

- A composable and agile software architecture pattern for load clustering
- Problem independent and allows for plugging different clustering algorithms
- Basic SILCA consists of several sub-patterns, implemented in a space-based architectural style,
  - decoupling of the agents
  - autonomic behavior of agents
- This allows finding the best algorithm for each specific problem.

Fig 2. SILCA pattern composition [4]
Algorithms applied


Fig. 3: Load Clustering [2]
Comparison

Fig. 4: Comparison of algorithms combinations in SILCA [4]
Results

• The combination of the Hierarchical algorithm with any other, except the Genetic K-Means algorithm, leads to a good execution time.
• For small networks, the unintelligent Hierarchical Clustering showed the best results.
• For large and more complex networks, an intelligent approach will help.

<table>
<thead>
<tr>
<th>use-case</th>
<th>metric</th>
<th>the most successful algorithm(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Load Clustering</td>
<td>absolute time</td>
<td>Hierarchical /Fuzzy C-Means (amount of load = 20, chain topology)</td>
</tr>
</tbody>
</table>
Peer Clustering (1)

• Usually, peers are placed randomly or based on their geographical position in a P2P network → a performance bottleneck → extremely poor performance

• This problem can be solved by using peer clustering.

• Peer Clustering aims to group peers, which have certain characteristics in common, together as neighbors.

• Peer Clustering is a highly dynamic procedure as peers are leaving and entering the network dynamically.
• As a consequence, query performance can be significantly improved compared to a random network topology [5]:
  – Requests are routed more efficiently and only to nodes which are likely to fit the request.
  – If it is possible to find a cluster that contains a node, which should fit the request, query flooding through the whole network is not necessary.
  – Consequently, the workload on nodes, which are probably not fitting the request, can be reduced.
Fig. 5: Peer Clustering

• It allows plugging of different peer clustering algorithms with their easy exchangeability and enable systematic benchmarking and comparison of these algorithms.

• It is problem independent → it should be used to find the best suiting algorithm for a specific problem.
Algorithms applied

• The following conventional and swarm-based algorithms are competitively benchmarked, evaluated and compared [5]:

• The metrics used for the evaluation are [5]:
  • Execution time, the Davies-Bouldin index (DBI), the Dunn index (DI), the silhouette coefficient (SC) and Averaged Dissimilarity coefficient (ADC).
Comparison

Execution time

Davies-Bouldin

Averaged Dissimilarity coefficient

Dunn index

Silhouette coefficient

Legend

- ABC
- ABCK
- Ant-based Clustering
- Ant K-Means
- Fuzzy C-Means
- Genetic K-Means
- Hierarchical Clustering
- K-Means
- PSO
- Slime Mold
- Slime Mold K-Means
# Results

<table>
<thead>
<tr>
<th>use-case</th>
<th>metric</th>
<th>the most successful algorithm(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Peer Clustering</strong></td>
<td>absolute time</td>
<td>Fuzzy C-Means, Ant-based Clustering, Hierarchical Clustering, Slime Mold and Slime Mold K-Means</td>
</tr>
<tr>
<td></td>
<td>Davies-Bouldin index</td>
<td>Hierarchical Clustering</td>
</tr>
<tr>
<td></td>
<td>Dunn index</td>
<td>Ant K-Means</td>
</tr>
<tr>
<td></td>
<td>Silhouette coefficient</td>
<td>Hierarchical Clustering</td>
</tr>
<tr>
<td></td>
<td>Averaged Dissimilarity</td>
<td>Hierarchical Clustering</td>
</tr>
<tr>
<td>coefficient</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Results

• Slime Mold and Slime Mold K-Means scale very well regarding execution time and effectiveness.
• Those two algorithms never provide unwanted massive variation in clustering effectiveness results.
• Hierarchical Clustering algorithm outperforms all other implemented algorithms.
• A combination of Slime Mold with Hierarchical Clustering \(\rightarrow\) Hierarchical Clustering algorithm thoroughly provides top results in terms of time and effectiveness.
Summary – Challenges (1)

• Huge complexity → one of the main characteristics of nowadays distributed systems.
• Intelligent metaheuristics support optimization and robustness of highly dynamic distributed systems.
• The problem of fair comparison and evaluation of different approaches that use different metaheuristics with a huge number of parameters
Summary – Challenges (2)

• One of the prominent challenges in the P2P data management is the problem of clustering.

• We can differentiate between two similar scenarios: data → load clustering & peer clustering

• Requirements on the evaluation methodology [8,9]:
  – Provisioning of a general framework
  – Composability of the architecture
  – Autonomy and Self-Organizing Properties
  – Support of arbitrary configurations
  – Benchmarking in different environments
  – Possibility of reconstructing the solution
Summary - Perspectives

• A methodology for the evaluation of set of algorithms (conventional, swarm-based, etc.) in distributed systems [8]:
  – a high-level abstraction of the problem’s communication in form of composable, agent-based coordination patterns
  – generic and flexible components based on these patterns
  – a framework as a composition of components
    • flexibly exchange of algorithms through “plugging”
  – identification of configuration and evaluation parameters
  – systematic evaluation of different configurations of algorithms, topologies and parameters
Selected References


