Integrating Logical Dependencies in Software Clustering: A Case Study on Apache Ant

Adelina Stana Ioana Șora

Computer Science and Engineering Department
"Politehnica" University of Timișoara, Romania

Outline

Introduction

- Software architecture helps developers understand the system and its behavior.
- Architectural reconstruction is needed when documentation is missing or outdated.
- Software clustering is used to identify modules or subsystems for reconstruction.
- Logical dependencies, extracted from co-changes in versioning systems, can provide additional insights.

Motivation

- Previous research shows logical dependencies are distinct from structural dependencies.
- Incorporating logical dependencies may improve software clustering results.
- Aim: Analyze the impact of logical dependencies on clustering solutions.

Logical Dependencies

Definition

Logical dependencies are relationships between software entities identified by analyzing co-changes in the versioning system, representing functional dependencies not evident from code analysis.

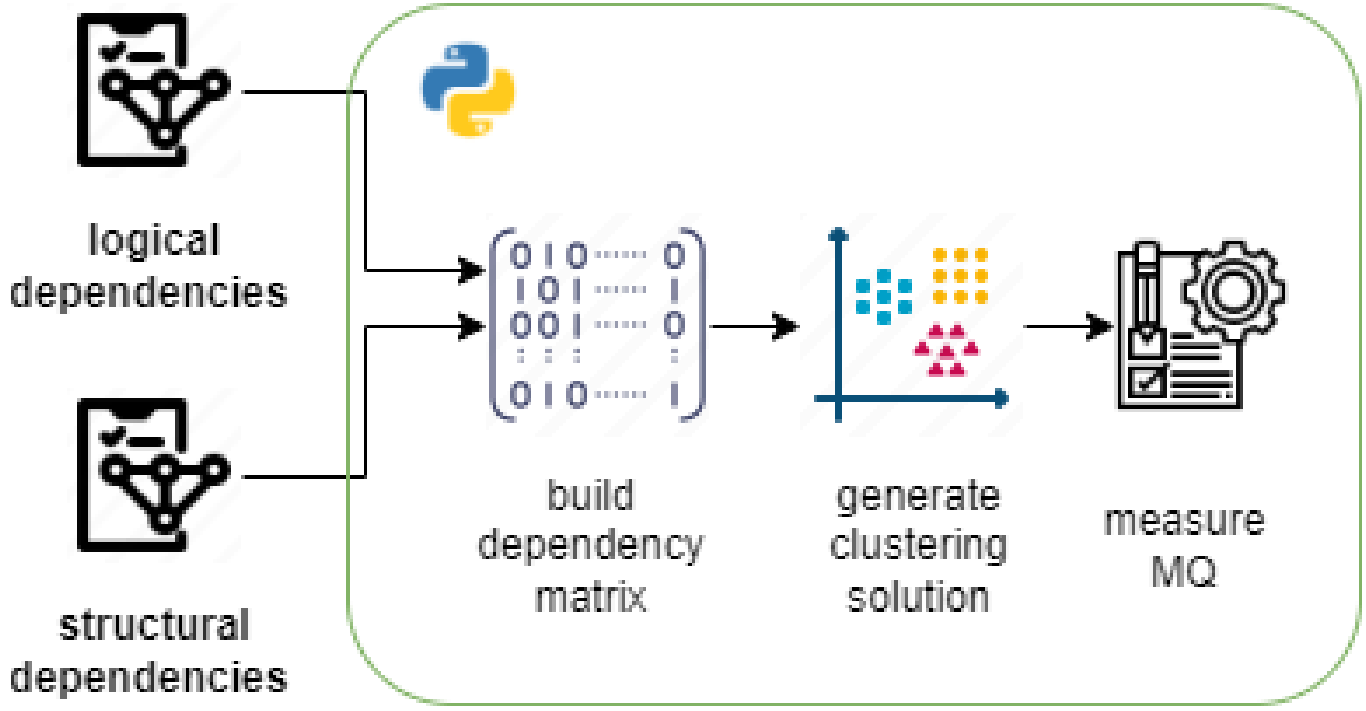

Figure 1: Clustering solution creation process diagram

Metrics for Logical Dependencies

- **Support**: Frequency of updates where entities A and B change together.
- **Confidence**: Proportion of co-changes relative to total updates of an entity.
- **Strength**: Adjusted confidence metric that accounts for the system's average number of updates.

Related Work

- Software clustering often uses structural dependencies from code.
- Other approaches use lexical dependencies or co-change data.
- Evaluation metrics like MOJO and Modularization Quality (MQ) are used to assess clustering results.

Methodology Overview

- Case study on Apache Ant.
- Three scenarios:
  1. Clustering using structural dependencies only.
  2. Clustering using logical dependencies only.
  3. Clustering using both logical and structural dependencies.
- Use of Louvain Clustering algorithm.
- Evaluation using Modularization Quality (MQ) metric.

Clustering Generation Process

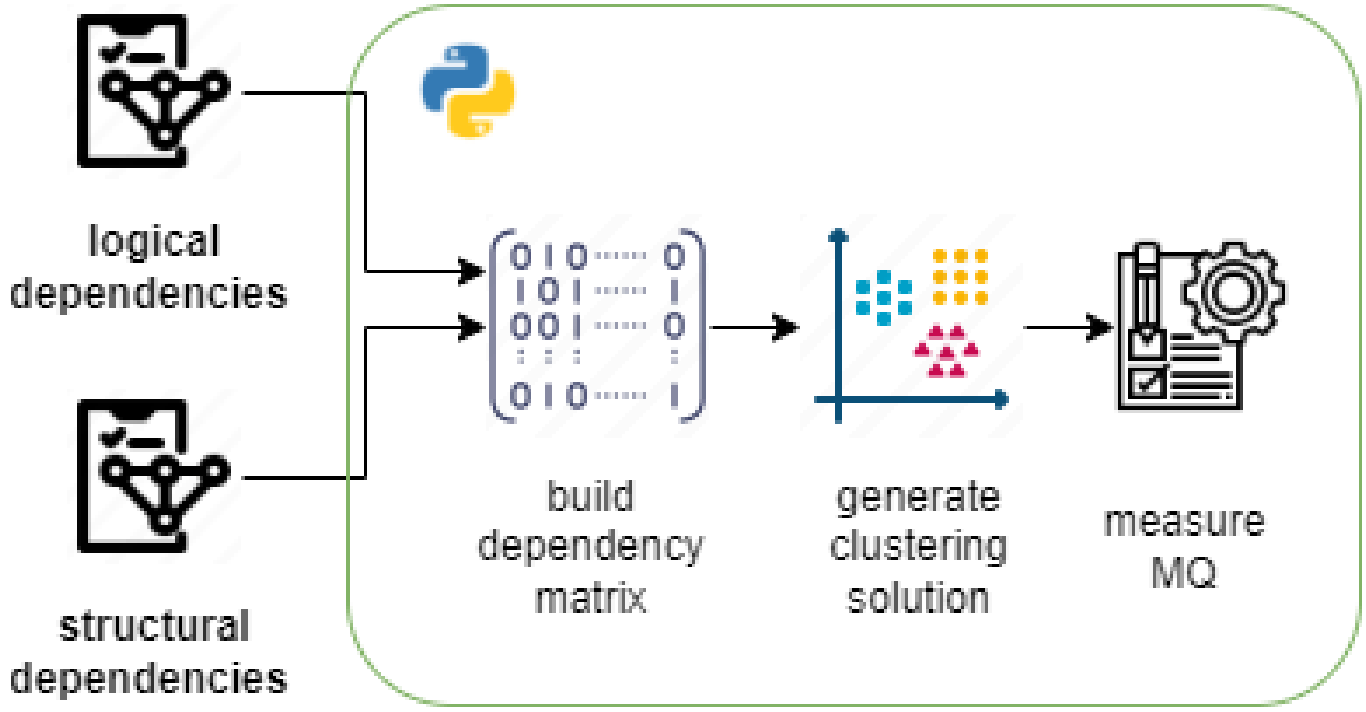

Figure 2: Clustering solution creation process diagram

Louvain Clustering Algorithm

- Community detection algorithm for complex networks.
- Optimizes modularity by moving nodes between clusters.
- Suitable for large-scale clustering tasks.

Clustering Results

Table 1: Louvain Clustering Results

Dataset	**Entities**	**Clusters**	**MQ Metric**
SD only	517	12	0.08
LD only (Strength 30%)	174	44	0.558
SD + LD (Strength 30%)	517	15	0.227

Analysis of Results

- Incorporating logical dependencies improves MQ scores.
- Clusters are more cohesive and functionally meaningful.
- Detailed analysis of specific classes demonstrates the benefits.

Conclusion

- Incorporating logical dependencies improves clustering quality.
- Provides additional insights not available from code analysis alone.
- The combined approach leads to clusters that better reflect the system's architecture.

Future Work

- Expand analysis to more projects.
- Explore alternative evaluation metrics.
- Further investigate the role of logical dependencies in software clustering.