One-Page Summary of the Paper:

Feature-Based Software Design Pattern Detection

Najam Nazar, Aldeida Aleti, and Yaokun Zheng

Instructor: Dr. Morteza Zakeri-Nasrabadi
Prepared by: Mohsen Elahifard (Stu. ID: 404131094)

This paper investigates a structured and automated approach for detecting software design patterns using feature-based analysis enhanced with machine learning algorithms. Design patterns play an important role in software engineering because they capture proven architectural solutions and improve maintainability, comprehension, and reuse. However, identifying the presence of these patterns in existing systems—especially large legacy codebases—remains challenging when done manually. The authors formulate the pattern detection problem as a classification problem and propose a method that represents each design pattern as a set of structural (e.g., class relationship types) and behavioral (e.g., method calls) features, after parsing the source code into an AST. These features are then compared with predefined pattern descriptions to automatically determine whether a pattern exists.

The framework begins by analyzing the system's class relationships, method interactions, and dependency structures. It then creates feature vectors that encode essential properties such as inheritance, composition, delegation, method overriding, and message passing. Design patterns (e.g., Singleton, Observer, Factory Method, Decorator) is formally defined through a configuration of expected features. Detection occurs by comparing extracted system features with these formal configurations using rule-based or similarity-based matching.

The results show that feature-based detection is more flexible and robust than traditional graph-based techniques. Unlike strict structural matching—which often fails when patterns are implemented with minor variations—the feature-based approach tolerates common real-world common structural differences. The experimental evaluation on several open-source projects, such as JHotDraw, compared its performance against existing state-of-the-art baseline methods, demonstrates that the method can effectively detect patterns while reducing false positives that arise from simplistic heuristics, achieving 80% precision and 79% recall.

Overall, the paper highlights that representing patterns as feature sets provides a scalable and maintainable foundation for automated pattern detection. This contributes to better program comprehension, facilitates reverse engineering, and supports refactoring efforts in long-lived software systems. The authors conclude that feature-based detection is a promising direction for tools aimed at analyzing complex object-oriented software.