Improving Automated Documentation to Code Traceability by Combining Retrieval Techniques

Xiaofan Chen

John Grundy

Contents

- Introduction
- Background
- Our Approach
- Implementation
- Evaluation
- Discussion
- Conclusion
- Future Works

Introduction

- Challenge: How to extract traceability links with both high precision and high recall.

Background

- Semi-automatic techniques
  - Rule-based
  - Scenario-based
  - Value-based
- Automatic techniques
  - Lightweight
    - Regular Expression
  - Heavyweight
    - Information Retrieval (IR) – PM, VSM, LSI etc
    - Text Mining

Information Retrieval (IR)

- Limitations
  - Very few true links at high cut points
  - Many fault links at low cut points
  - Some links are missed

Strategies

- VSM with general/context-specific thesaurus
- PM with hierarchical modeling, logical clustering, pruning of the probabilistic network
- LSI with source code clustering, identifier classifying, thesaurus, hierarchical structure

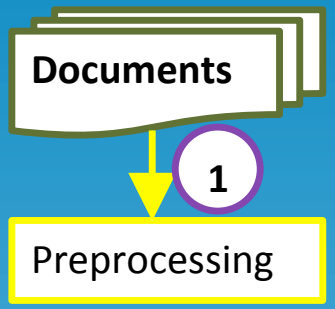
- No strategies can largely decrease fault links at low cut points and significantly increase true links at high cut points.

Our approach

- VSM with Regular Expression (RE), Key Phrases (KP), and Clustering

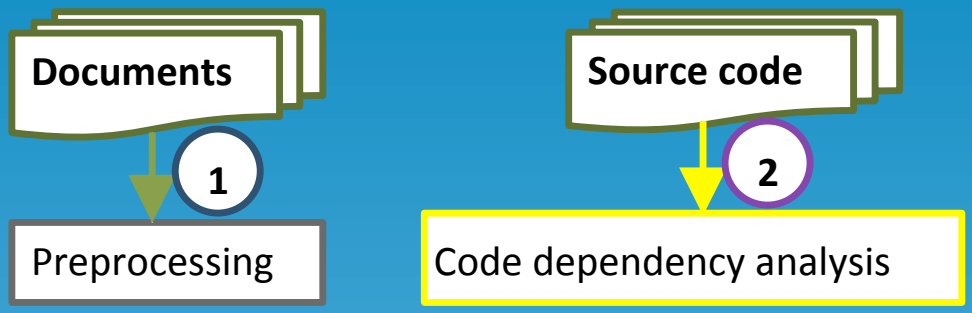
- Adding RE to retrieve more true links at high cut points
- Adding KP to retrieve all possible links
- Adding Clustering to reduce fault links at low cut points

Implementation



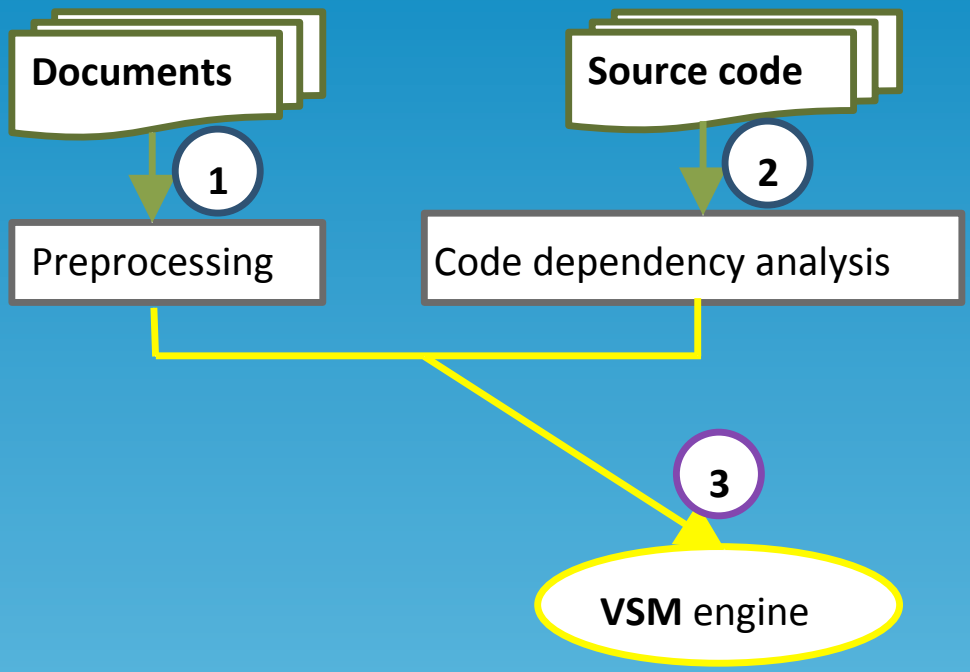
- Documents are divided into small documents based on sections/headings.
- Extract documents' inherent hierarchical structure

Implementation



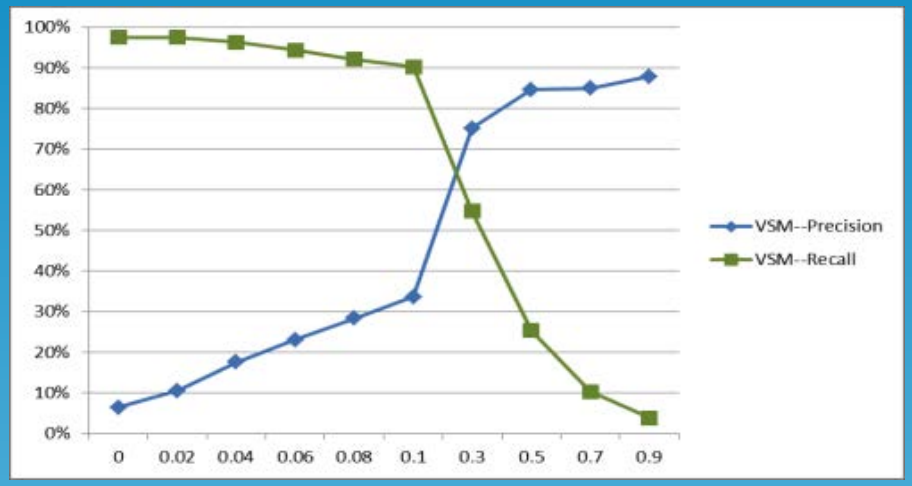
- Extract classes' identifiers/names
- Extract comments inside code

Implementation

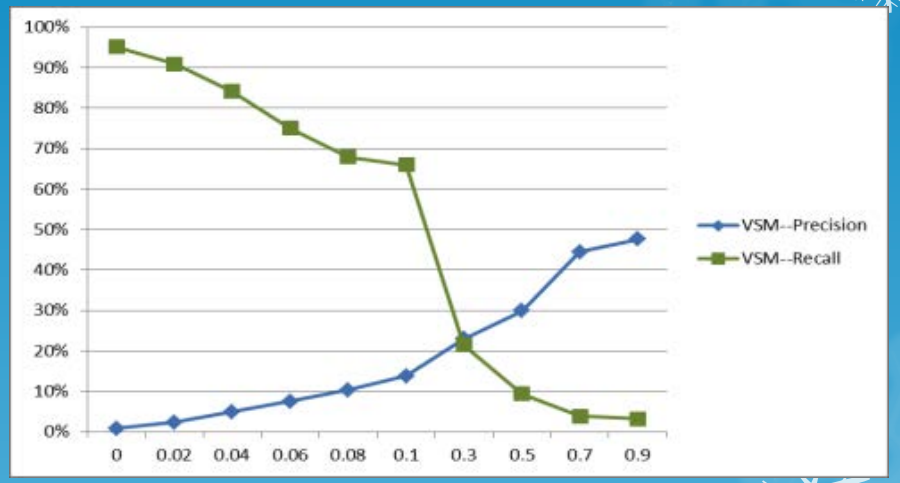


- Using VSM engine to retrieve links between sections and classes

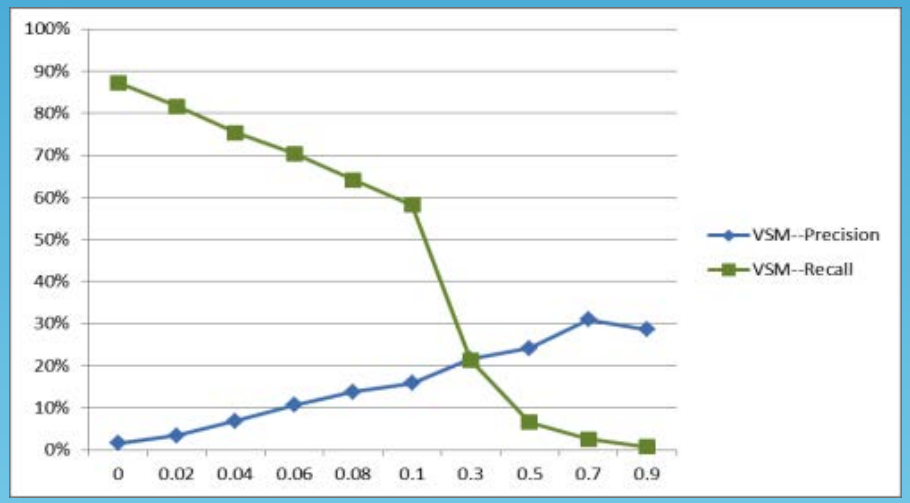
Results of VSM



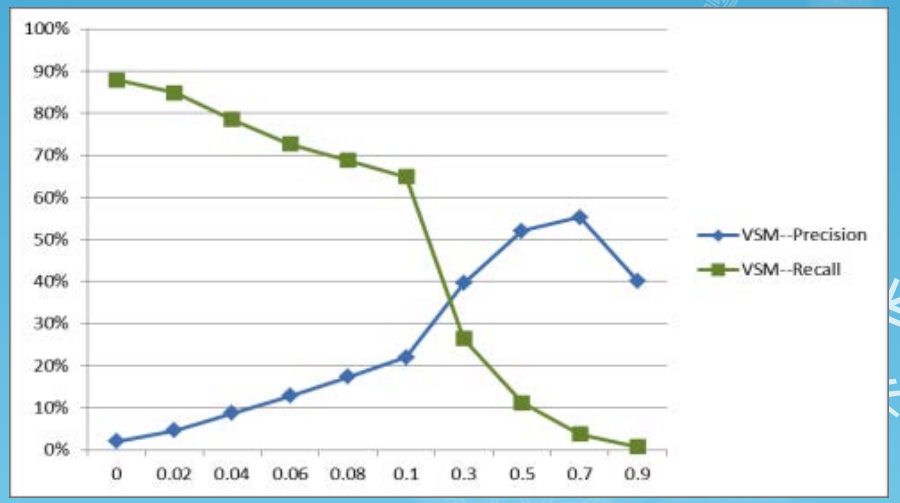
(a) JDK1.5



(b) ArgoUML

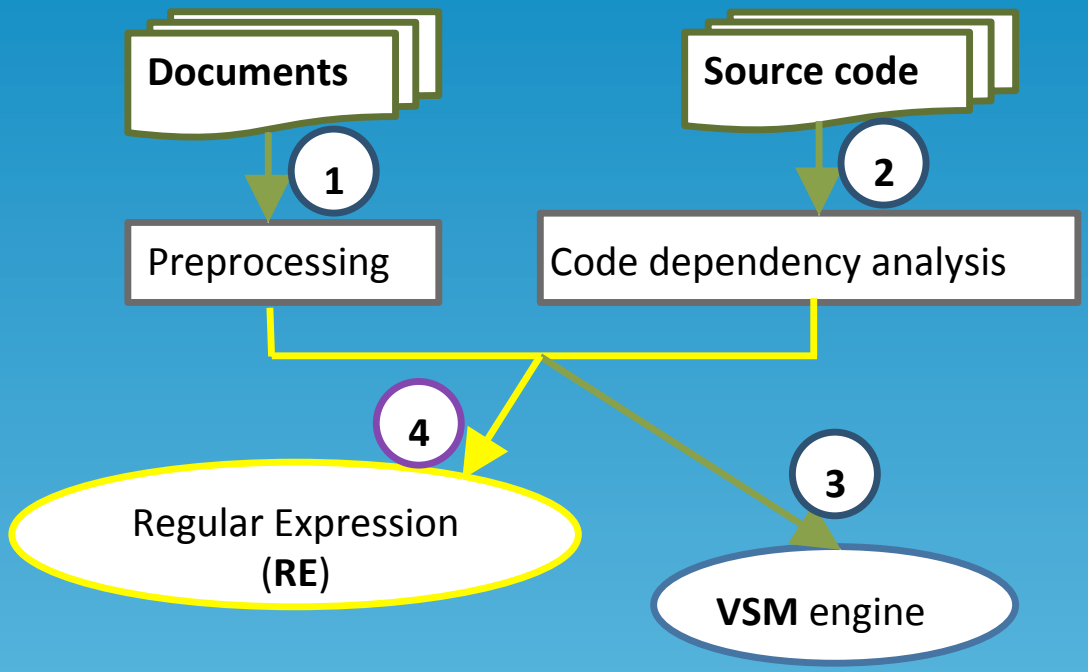


(c) Freenet



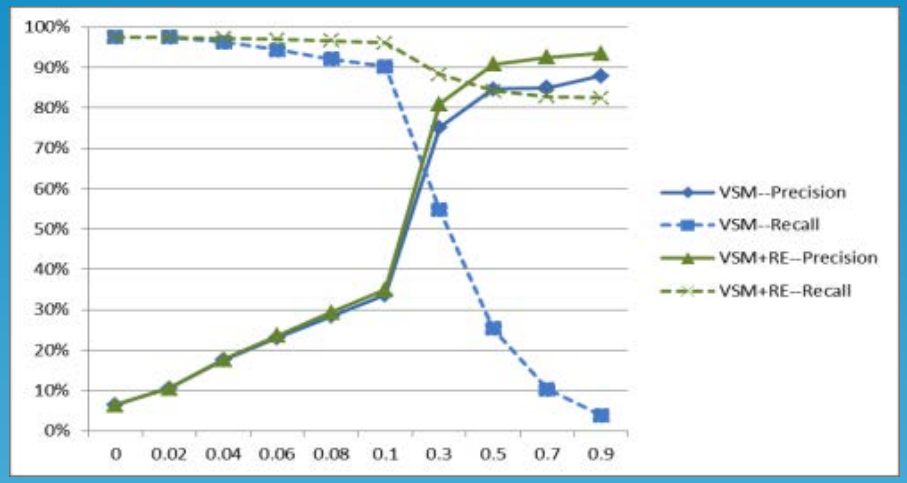
(d) JMeter

Implementation

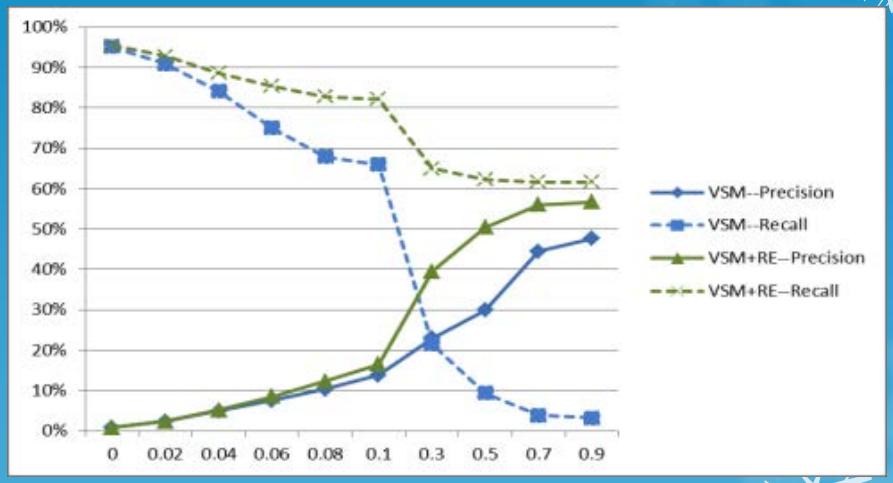


- Add RE into VSM
- Use two regular expressions to capture documents containing class names, and form the RE link set
  1. (.*)(^a-zA-Z0-9\-)<C-?o-?n-?t-?r-?o-?l>(^a-zA-Z0-9\-)(.*)
  2. (.*)(^a-zA-Z0-9\-)<each part of package name>(^a-zA-Z0-9\-)(.*)

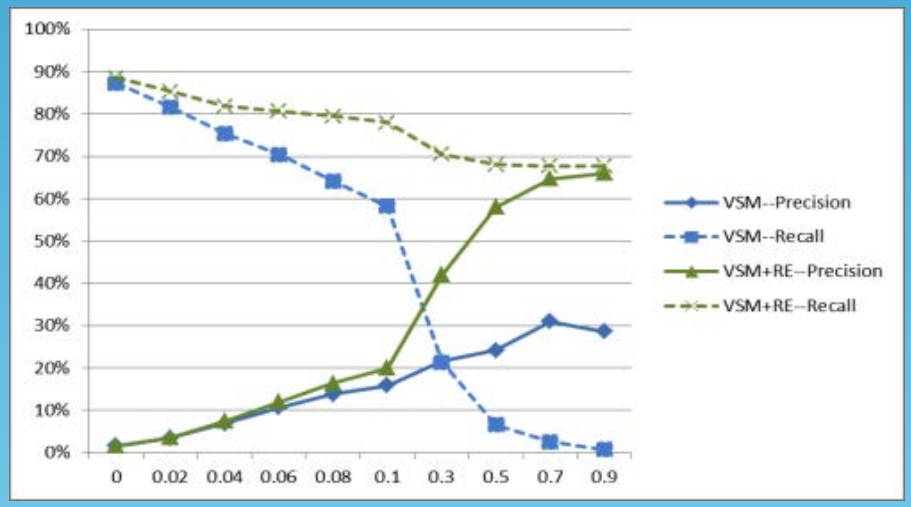
Results after adding RE



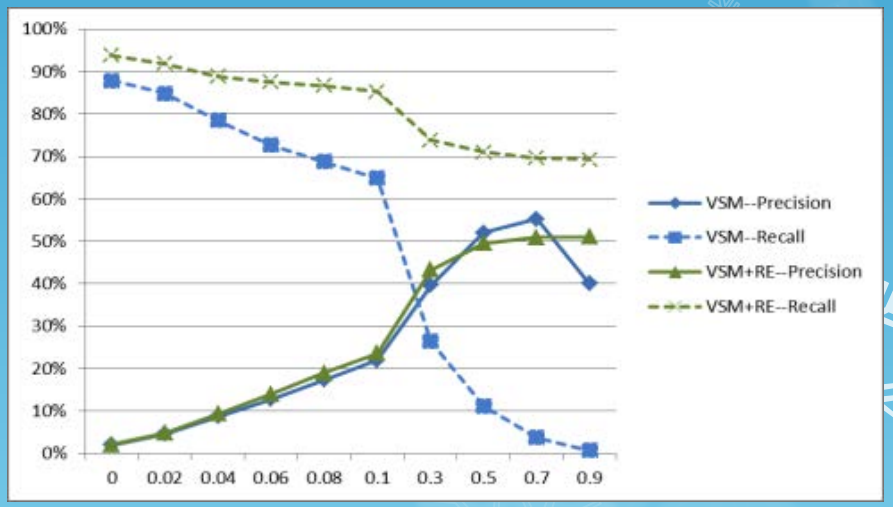
(a) JDK1.5



(b) ArgoUML

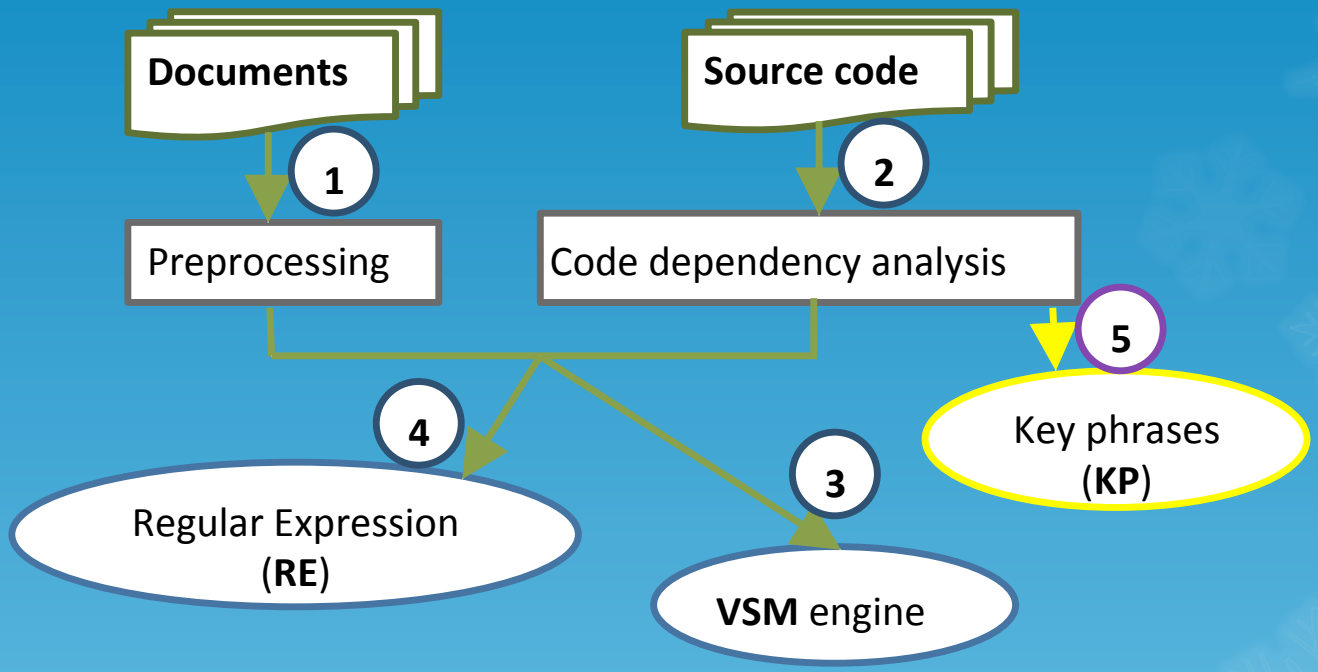


(c) Freenet



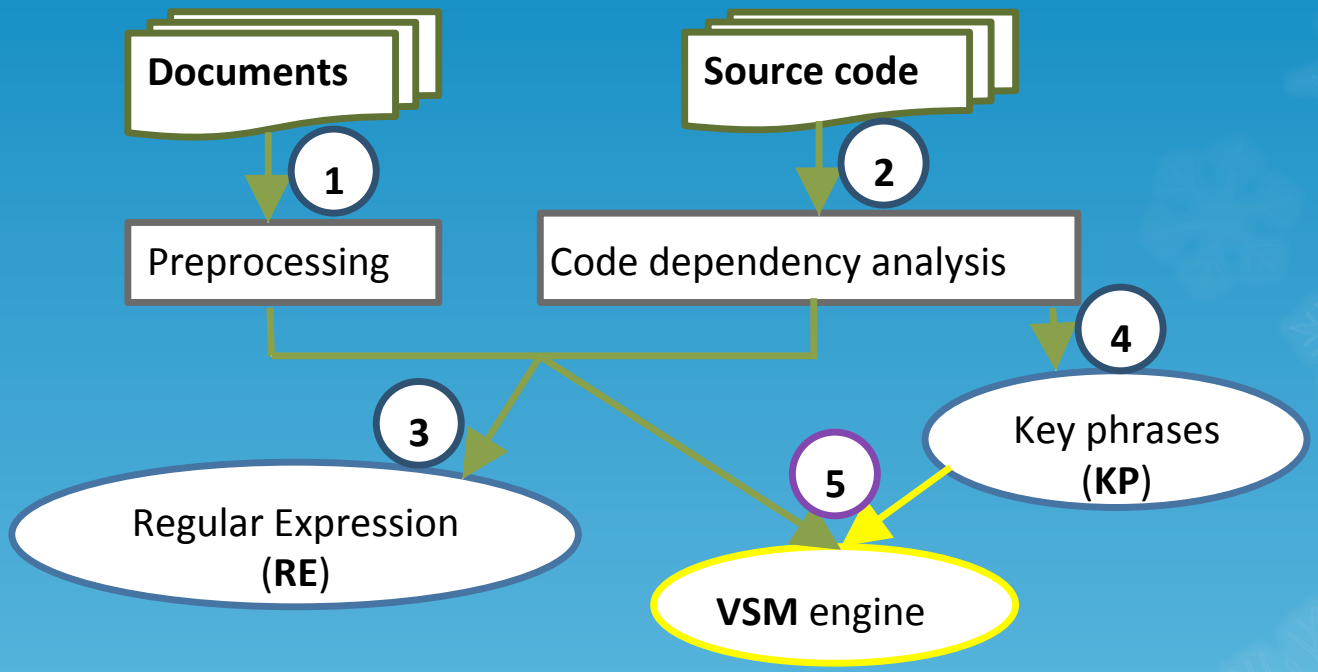
(d) JMeter

Implementation



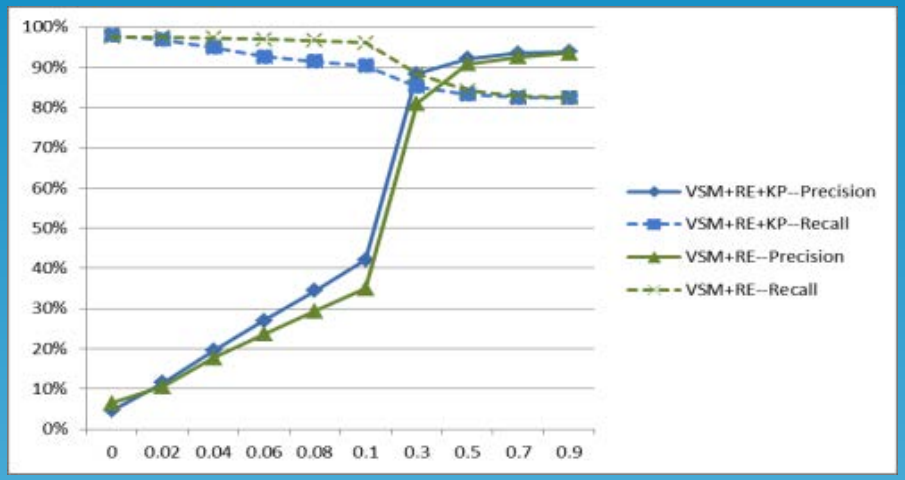
- Adding KP into VSM
- Extract key phrases from code's comments
- Argument VSM queries

Implementation

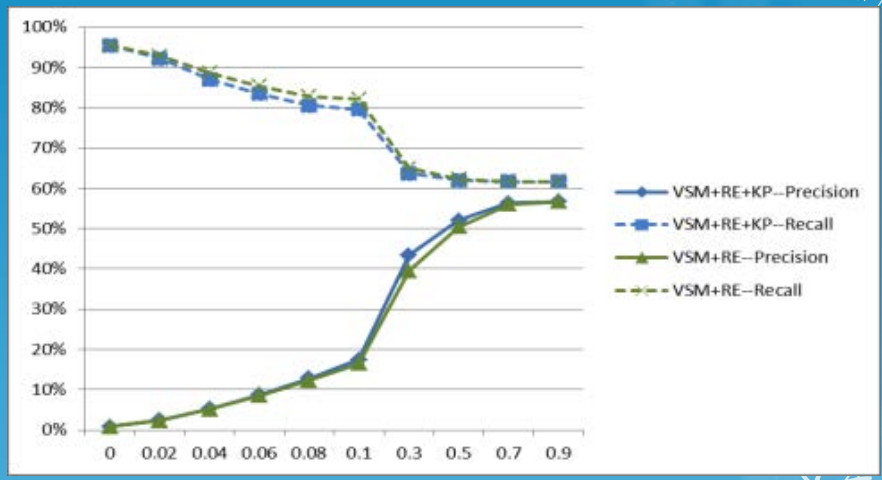


- Form VSM queries by combining extracted key phrases and class names
- Retrieve links according to VSM queries
- Then form the VSM link set

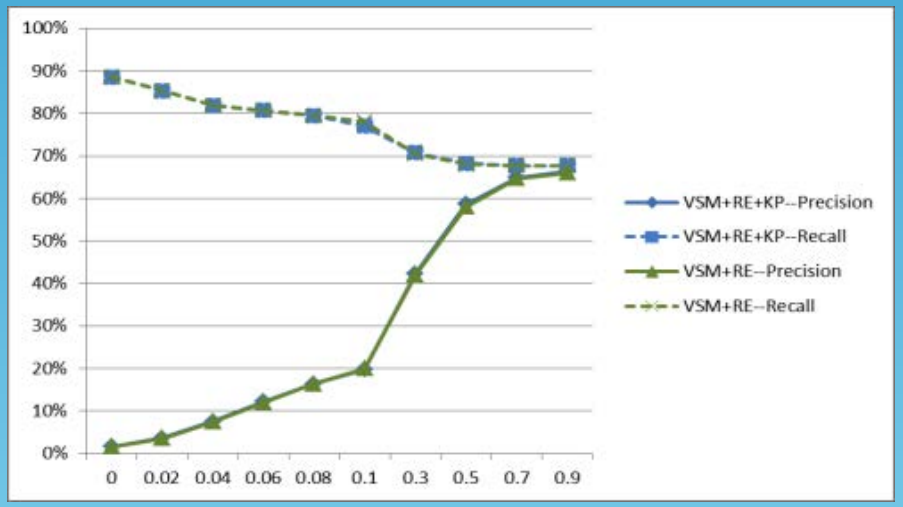
Results after adding KP



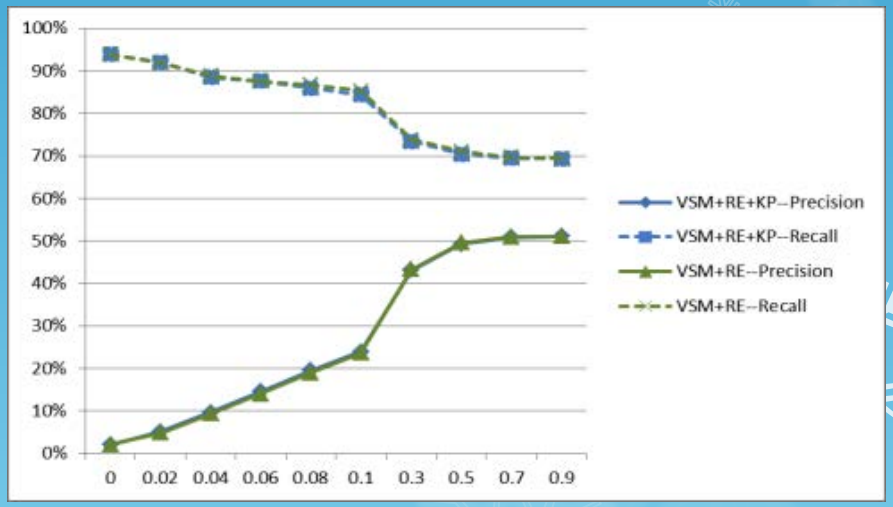
(a) JDK1.5



(b) ArgoUML

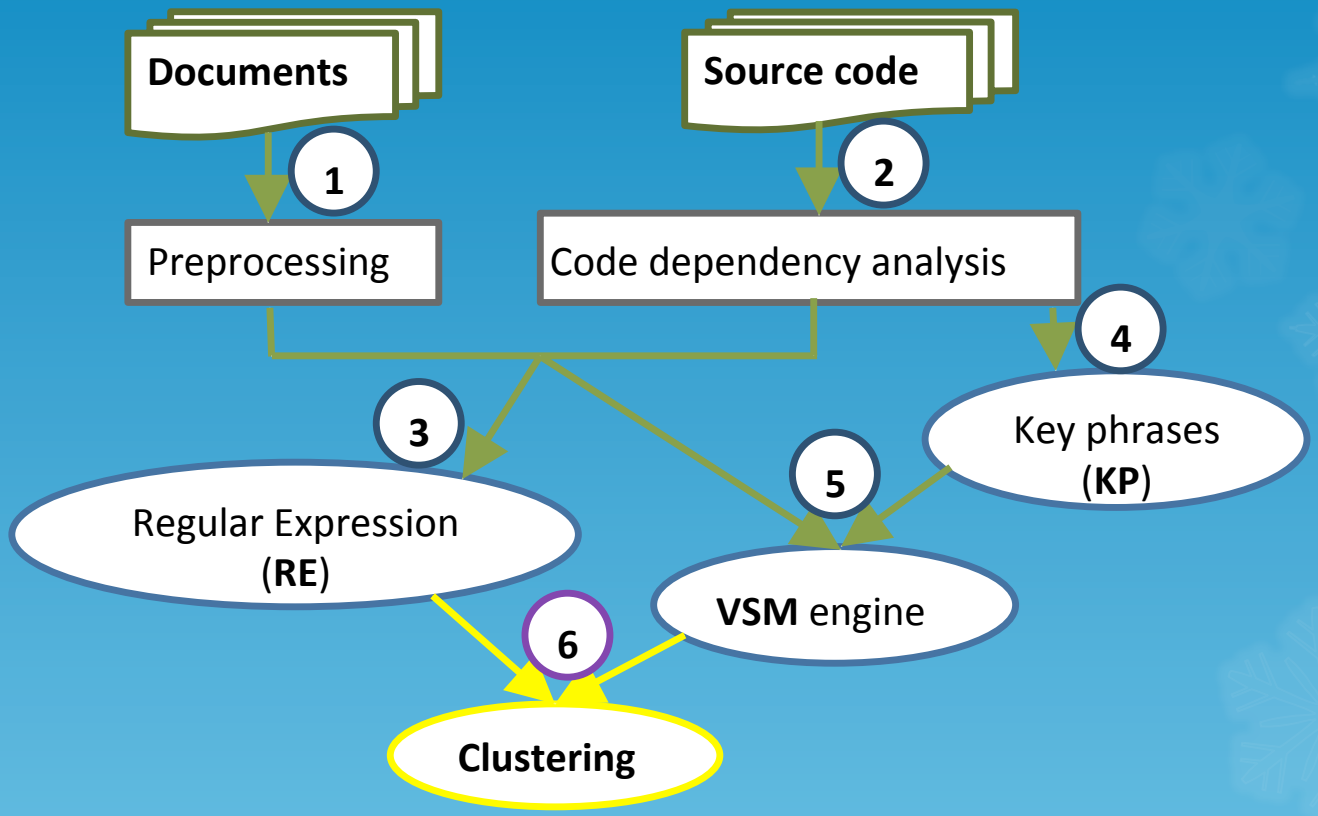


(c) Freenet



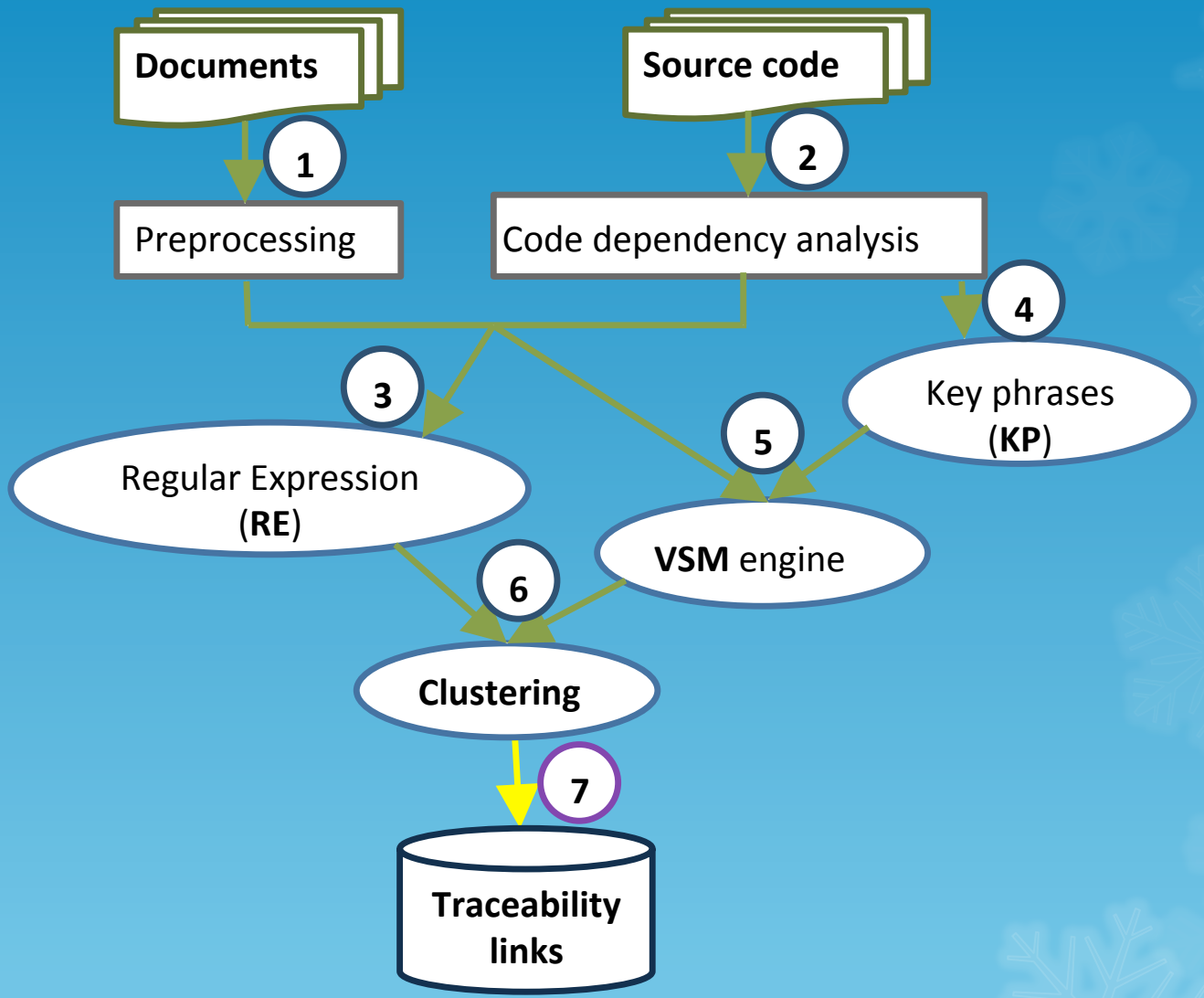
(d) JMeter

Implementation

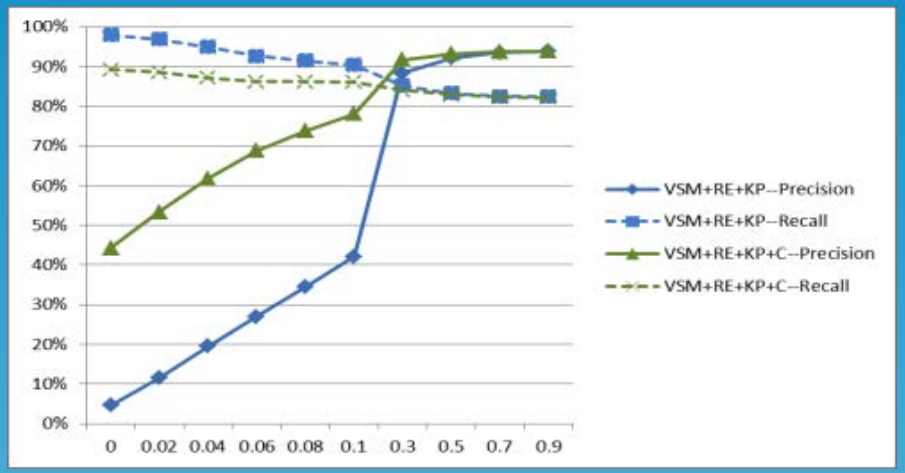


- Clustering: use the inherent hierarchical structure of documents
- Merge the RE and VSM link sets
  - If a link is in both RE and VSM link sets, then remove the one in VSM set and remain the one in RE set
- The merged link set is refined by Clustering

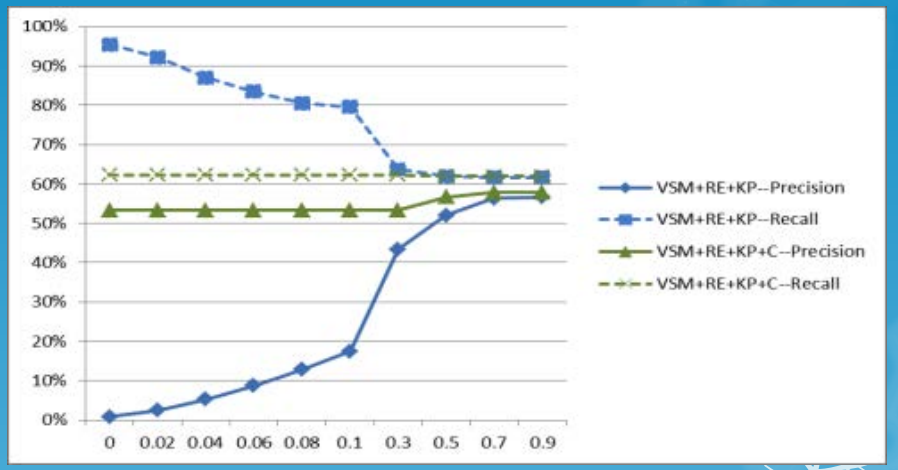
Implementation



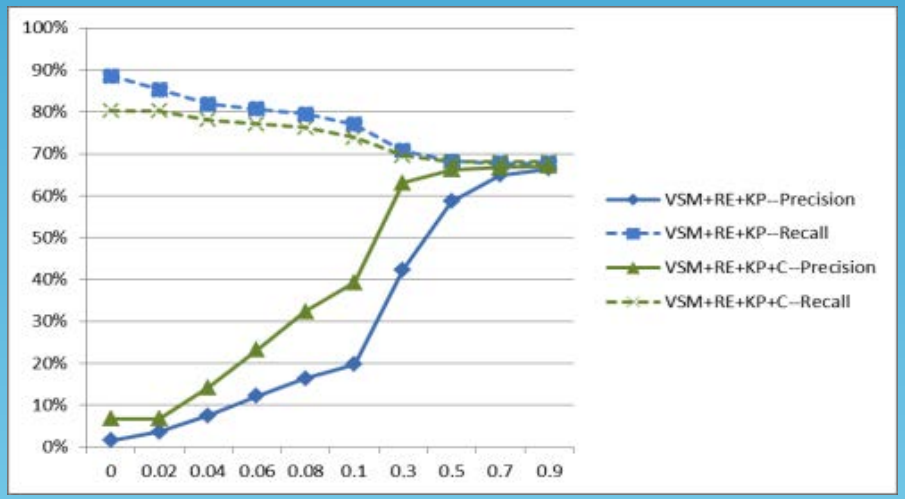
Results after adding Clustering



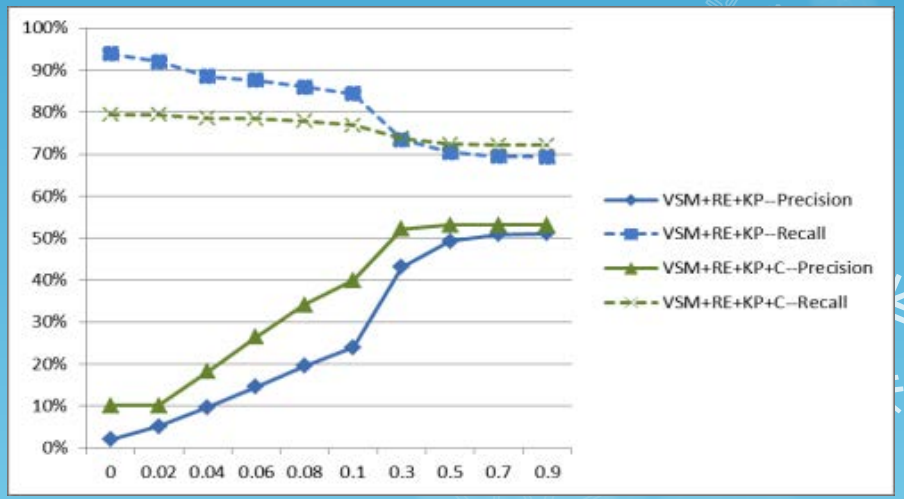
(a) JDK1.5



(b) ArgoUML

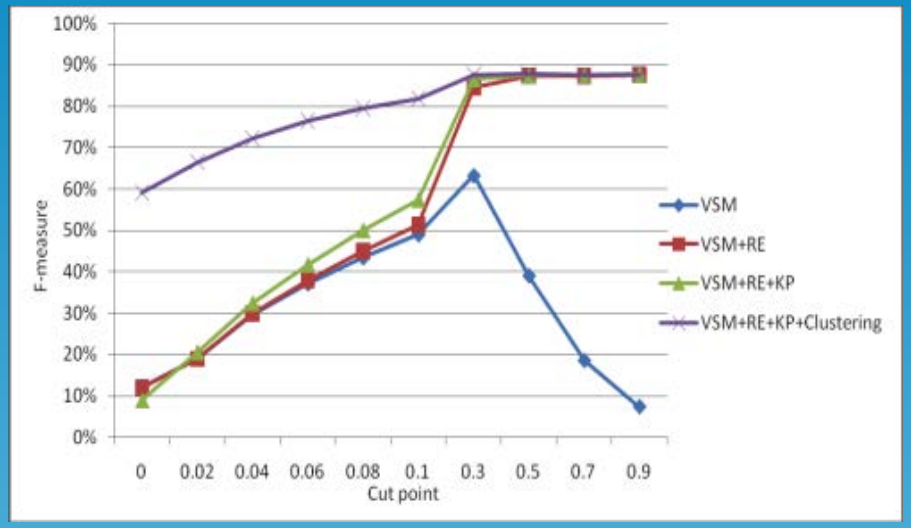


(c) Freenet

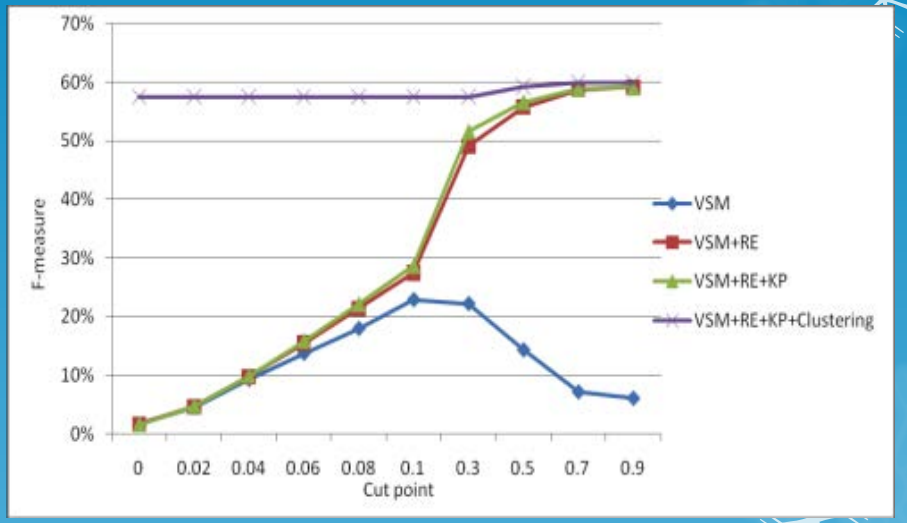


(d) JMeter

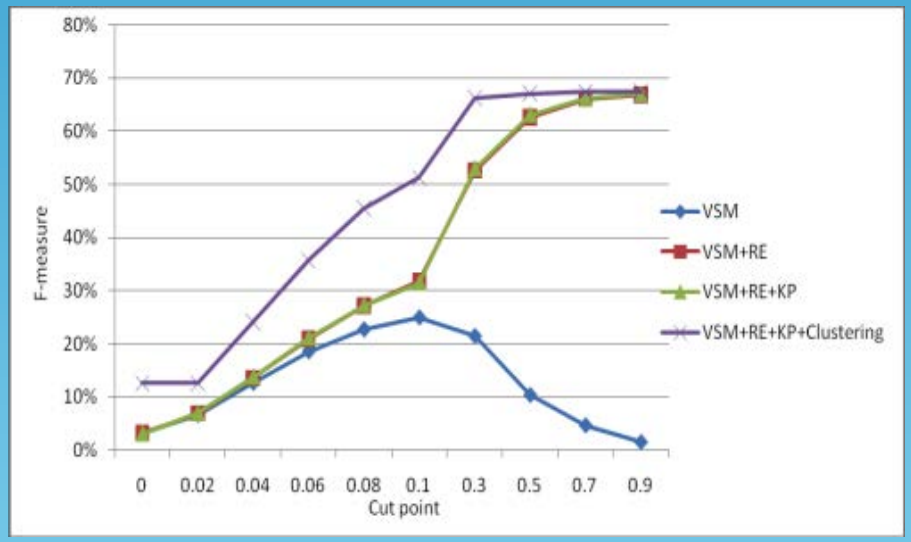
F-measure results



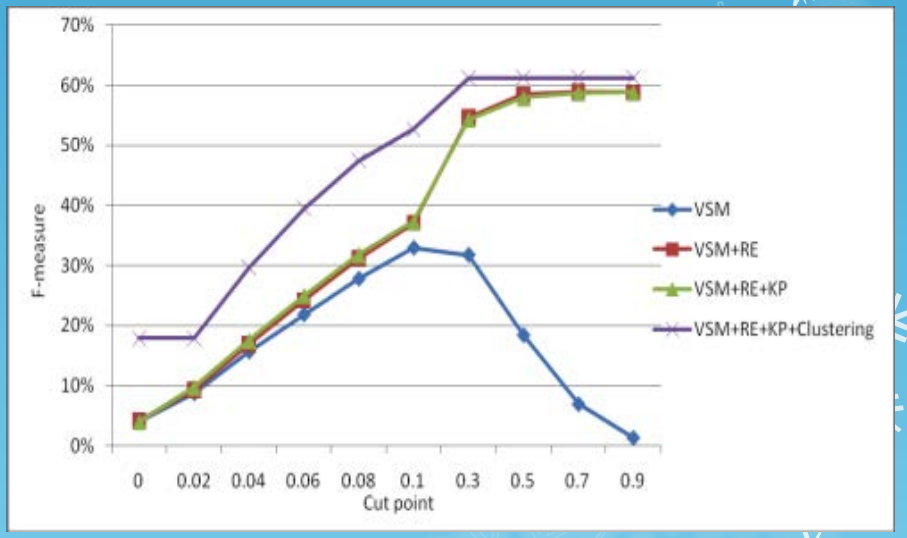
(a) JDK1.5



(b) ArgoUML



(c) Freenet



(d) JMeter

Discussion

- Adding RE increases precision and recall at high cut points
- Adding KP increases precision for JDK1.5 and ArgoUML, but Freenet and Jmeter is unresposive to it
- Adding Clustering increases precision at low cut points
- Some true links are discarded by Clustering

Conclusion

- Our approach eliminates some VSM's limitations
  - Reduce fault links at low cut points
  - Increase true links at high cut points

Future works

- Allow users to configure thresholds and select some or all techniques to extract traceability links
- Refine the extracted links by allowing user creation and editing of links
- Usability tests to determine how effective our traceability tool is in assisting users navigate between documents and source code

Thanks

References

Chen, X., Hosking, J.G., Grundy, J.C. and Amor, R. Development of robust traceability benchmarks, 2013 Australasian Conference on Software Engineering (ASWEC 2013), Melbourne, Australia, July 2013, IEEE CS Press.

Chen, X., Hosking, J.G. and Grundy, J.C. Visualizing Traceability Links between Source Code and Documentation, 2012 IEEE International Symposium on Visual Languages and Human-Centric Computing, Innsbruck, Austria, Sept 30-Oct 4 2012, IEEE CS Press.

Chen, X. and Grundy, J.C. Improving Automated Documentation to Code Traceability by Combining Retrieval Techniques, In proceedings of the 26th IEEE/ACM International Conference on Automated Software Engineering, Nov 6-10 2011, IEEE Press.

Chen, X., Hosking, J.G. and Grundy, J.C. A Combination Approach for Enhancing Automated Traceability, New Ideas and Emerging Results Track, In Proceedings of the 2011 International Conference on Software Engineering (ICSE2011), Honolulu, Hawaii, USA, May 21-28 2011.