Yichao SHAO, Zhiqiu HUANG, Weiwei LI, Yaoshen YU, 2022. Fast code recommendation via approximate sub-tree matching. *Frontiers of Information Technology & Electronic Engineering,* 23(8):1205-1216. https://doi.org/10.1631/FITEE.2100379

Fast code recommendation via approximate sub-tree matching

Key words: Code reuse; Code recommendation; Tree similarity; Structure information

Corresponding author: Zhiqiu HUANG
E-mail: zqhuang@nuaa.edu.cn
ORCID: https://orcid.org/0000-0001-6843-1892

Motivation

1. Software developers often write code that has similar functionality to existing code segments. A code recommendation tool that helps developers reuse these code fragments can significantly improve their efficiency.

2. Some methods use sequence matching algorithms to find the related recommendations. Most of them are time-consuming and can leverage only low-level textual information from code.

3. Some other methods try to obtain structural information from the abstract syntax tree (AST) of source code. However, traditional sub-tree matching is time-consuming and thus cannot be applied to code recommendation tasks directly.

Contributions

1. We introduce a new code recommendation algorithm based on sub-tree hashing and the Smith–Waterman (SW) algorithm. It takes programming context as input and recommends relevant code snippets to assist developers in software development.

2. We implement a code recommendation tool for Java and C. Experimental results show that it has good performance in terms of both time consumption and accuracy for different recommending tasks.

Method

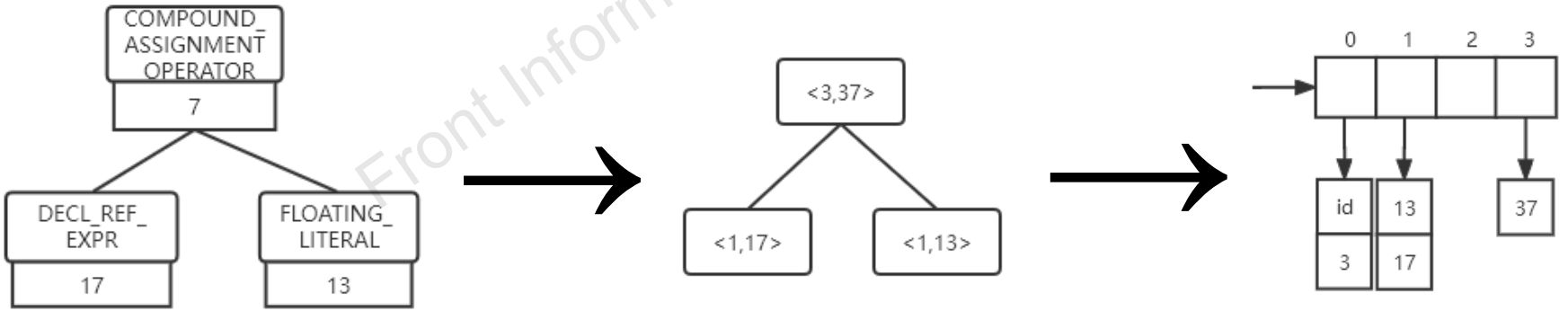
1. Data processing

For the AST of each code snippet, we carry out the following steps:

(1) Map each node to a given hash value according to the function.

(2) Calculate the subtree information as <node number, hash value>.

(3) Store the node information in the form of lists.

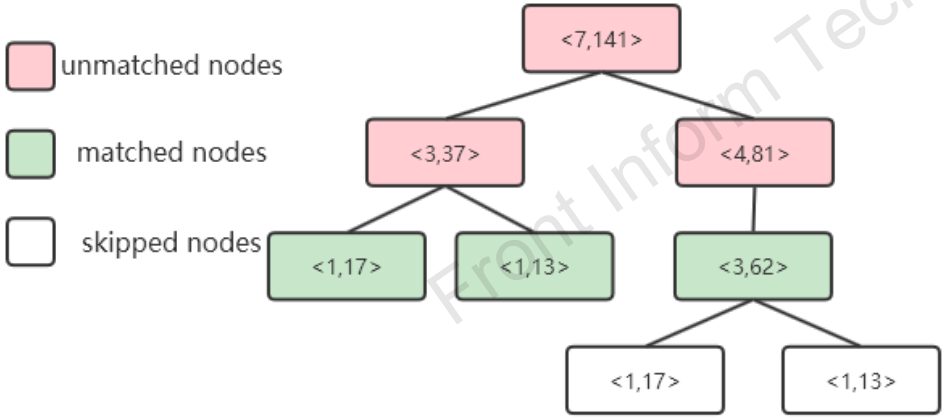


(1) (2) (3)

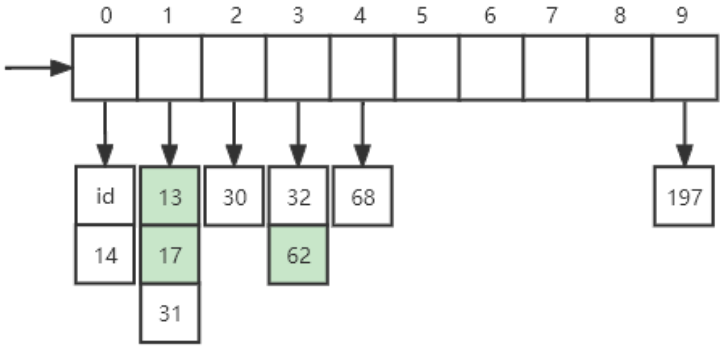
Method (Cont'd)

2. Search algorithm

We calculate the similarity scores between the query code and each candidate code, and obtain a candidate code set containing *K* code snippets with the highest similarities.



query tree



candidate code list

Method (Cont’d)

3. Re-ranking

To further consider sequential information and exclude false positive errors, we introduce a re-ranking method based on the SW algorithm. We combine the SW sequence similarity with the tree-list similarity obtained in the filtering phase as the basis for the re-ranking process.

Major results

Recall and MRR

Table 2 The experimental results from different methods over all datasets

Dataset	Metric	Value		
		SENSORY	Aroma	Our method
IJaDataset 2.0	Recall@1	100%	99%	99.3%
	Recall@10	100%	100%	100%
	MRR	1	0.995	0.996
OJSystem	Recall@1	67.3%	76.2%	81.1%
	Recall@10	78.6%	86.7%	92.1%
	MRR	0.723	0.821	0.847
OJNUAA	Recall@1	81.5%	91.8%	94.5%
	Recall@10	90.3%	95.5%	97.3%
	MRR	0.909	0.946	0.967

Major results (Cont’d)

Time costs

Table 3 Time costs of three methods applied to IJaDataset

Approach	Average recommending time (s)
SENSORY	232.3
Aroma	0.4
Our method	2.1

Conclusions

1. In this study, we have presented a search-based code recommendation technique that combines an approximate sub-tree matching algorithm and a sequence matching algorithm.

2. Experimental results have shown that our method can recommend code with high recall and significantly outperforms compared methods.