Improving Bug Localisation Using Lexical Information and Call Relations

by Tezcan Dilshener

The Open University

Research Aim

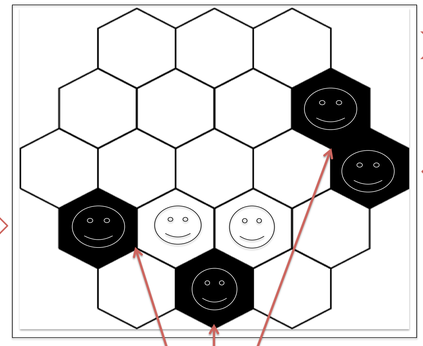
- Identify opportunities of using CR vocabulary and application call relations to
  - address challenges faced by current combined approaches.
  - reduce the program comprehension overhead.
- ***RQ1***: Does utilizing a combined approach based on lexical information and call relations improve the search performance with respect to a simple string search?
- ***RQ2***: How does the combined approach, implemented in our tool perform compared to another state-of-the-art tool?

Our Approach

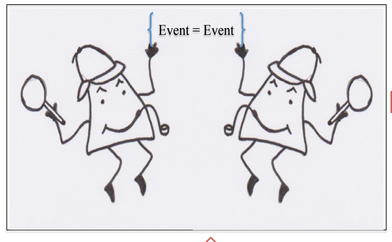
- Integrate lexical information retrieval with structural program dependency search.
- Present a ranked list of relevant classes for a change request (located in the top-*N* positions).
  1. Search for classes using the terms extracted from CR's.
  2. Assign a score to class based on where search terms occur.
  3. For each class, adjust score based on the call-relations to neighbours.
  4. Sort result list using new score to group relevant classes together.

ConCodeSe

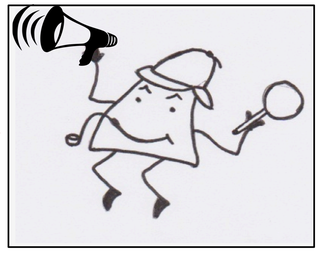
Contextual Model



Lexical Similarity



Class Call Relations



Search "Algo" Rhythm

Bug Report

```
<bug id="2093" opendate="2010-03-30 04:03:07" fixdate="2010-04-01">
  <buginformation>
    <summary>
        Make Quota event limit selectable
    </summary>
    <description>
        By adding this feature we should allow the selection of specific events the
        limit is applied to.
    </description>
  </buginformation>
</bug>
```

Ranked Result List of Relevant Classes

Pillar-Two classes affected by CR # 2093	Lexical ranking	Lexical + call relations rank
QuotaShareContractStrategy.java	2	3
EventAalLimitStrategy.java	31	1
LimitStrategyType.groovy	33	2
EventLimitStrategy.java	4	4
NoneLimitStrategy.java	38	10
ILimitStrategy.java	5	5

Evaluation of Results

- RQ1: Lexical search locates less than 20% of CRs where as our approach locates more than 70%.

	Simple lexical match	Lexical scoring + call relations
Average	19%	72%

- RQ2: ConCodeSe locates more CRs per application and more relevant classes in the top-*N* than BugLocator.

	Top-1		Top-10	
	Bug Locator	Con CodeSe	Bug Locator	Con CodeSe
Average	26%	35%	59%	69%

	Better		Same		Worse	
	top-1	top-10	top-1	top-10	top-1	top-10
Average	19%	30%	72%	54%	9%	16%

Contributions

- An algorithm using lexical and structural information suggests, in a ranked order, classes to be changed.
- Vastly superior to simple string search, as performed by developers using an IDE.
- Compared to an existing approach,
  - improved ranking of affected classes.
  - increased percentage of CRs for which relevant classes are retrieved.
  - enhanced number of relevant classes suggested among the top-1 or -10.
- Challenging to find the classes referred by a CR:
  - in spite our improvements, over 40% of CRs may not be localised
  - further enhance algorithm by considering domain concept relations.