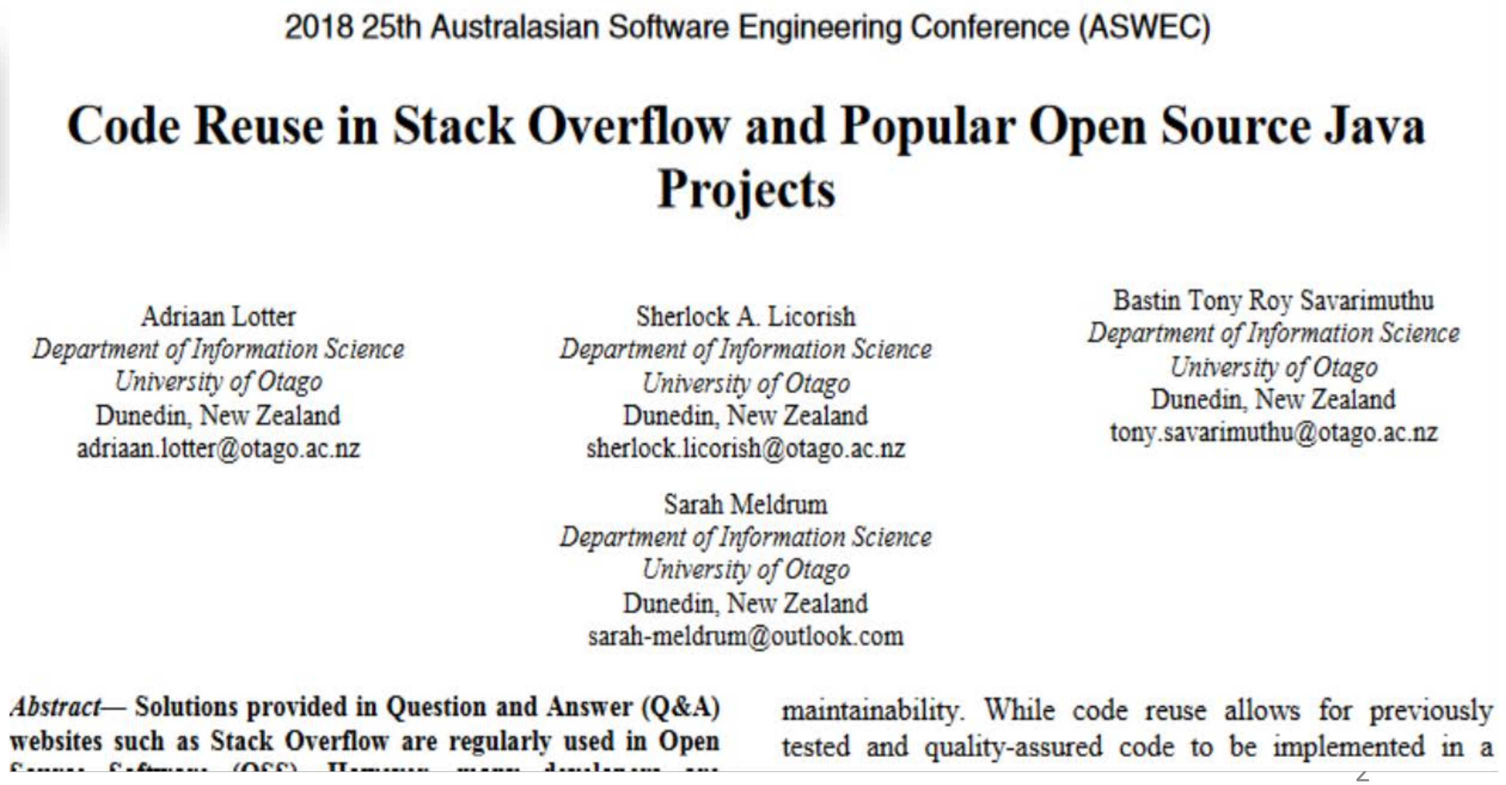
Can genetic improvement enhance online code snippets?

Sherlock A. Licorish
University of Otago
Dunedin, New Zealand
sherlock.licorish@otago.ac.nz

Situation



2018 25th Australasian Software Engineering Conference (ASWEC)

Code Reuse in Stack Overflow and Popular Open Source Java Projects

Adriaan Lotter
Department of Information Science
University of Otago
Dunedin, New Zealand
adriaan.lotter@otago.ac.nz

Sherlock A. Licorish
Department of Information Science
University of Otago
Dunedin, New Zealand
sherlock.licorish@otago.ac.nz

Bastin Tony Roy Savarimuthu
Department of Information Science
University of Otago
Dunedin, New Zealand
tony.savarimuthu@otago.ac.nz

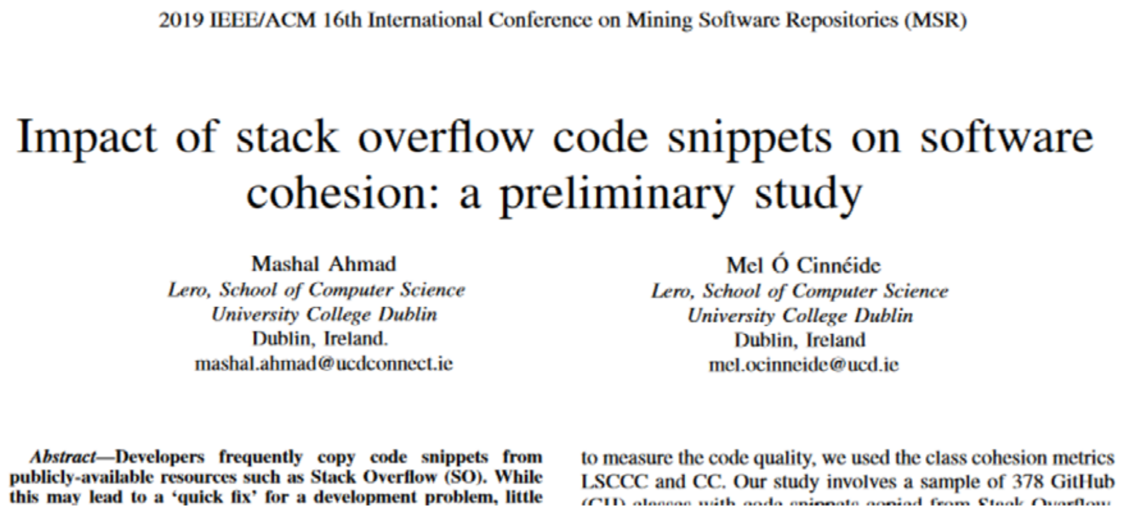
Sarah Meldrum
Department of Information Science
University of Otago
Dunedin, New Zealand
sarah-meldrum@outlook.com

Abstract— **Solutions provided in Question and Answer (Q&A) websites such as Stack Overflow are regularly used in Open**

maintainability. While code reuse allows for previously tested and quality-assured code to be implemented in a

Challenge

- Snippets online can often be incorrect, insecure, and incomplete
- We have observed errors in Stack Overflow code
- These observations extend to students' work, across multiple universities
- Errors have also been reported by open source developers, proprietary developers, and end users… the software development community



2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR)

Impact of stack overflow code snippets on software cohesion: a preliminary study

Mashal Ahmad
Lero, School of Computer Science
University College Dublin
Dublin, Ireland.
mashal.ahmad@ucdconnect.ie

Mel Ó Cinnéide
Lero, School of Computer Science
University College Dublin
Dublin, Ireland
mel.ocinneide@ucd.ie

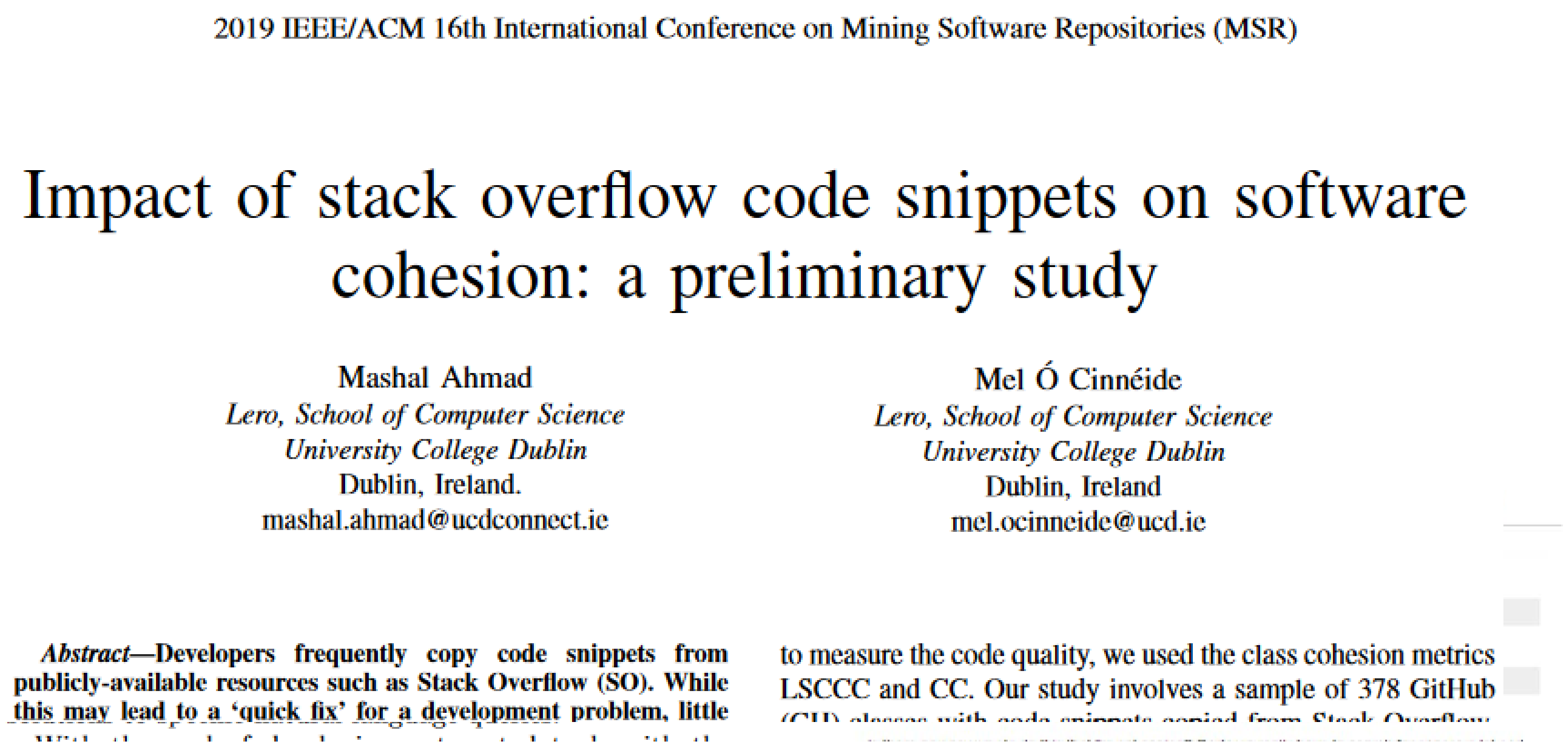
***Abstract*—Developers frequently copy code snippets from publicly-available resources such as Stack Overflow (SO). While this may lead to a 'quick fix' for a development problem, little**

to measure the code quality, we used the class cohesion metrics LSCCC and CC. Our study involves a sample of 378 GitHub

"App explanation: the sprit of stack overflow is coders helping coders"

- *NissanConnect EV mobile app*

Academic Community's Awareness



2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR)

Impact of stack overflow code snippets on software cohesion: a preliminary study

Mashal Ahmad
Lero, School of Computer Science
University College Dublin
Dublin, Ireland.
mashal.ahmad@ucdconnect.ie

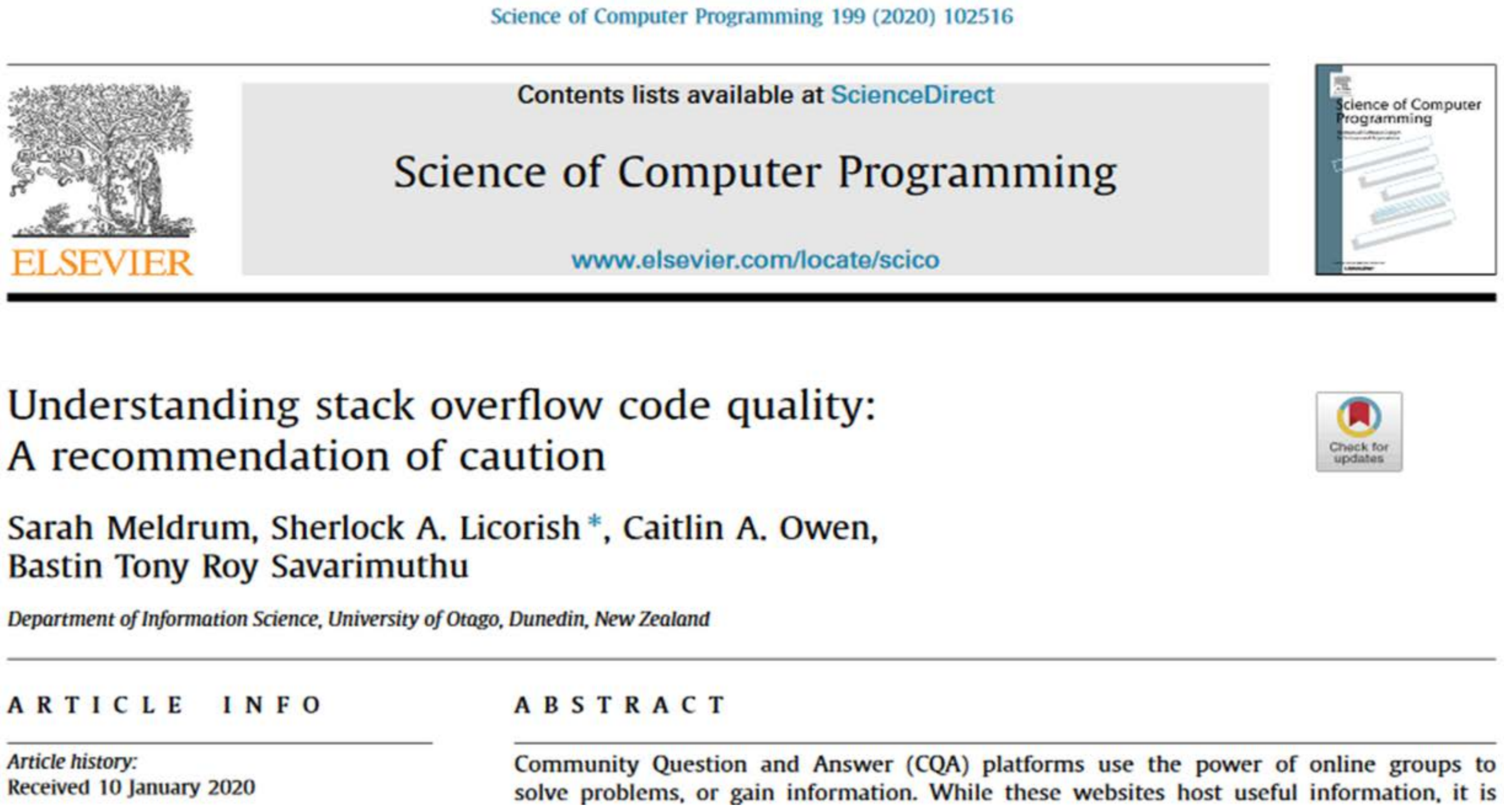
Mel Ó Cinnéide
Lero, School of Computer Science
University College Dublin
Dublin, Ireland
mel.ocinneide@ucd.ie

***Abstract*—Developers frequently copy code snippets from publicly-available resources such as Stack Overflow (SO). While this may lead to a 'quick fix' for a development problem, little**

to measure the code quality, we used the class cohesion metrics LSCCC and CC. Our study involves a sample of 378 GitHub (GH) classes with code snippets copied from Stack Overflow

But we still use online code snippets!

Current State of Play



Science of Computer Programming 199 (2020) 102516

Contents lists available at ScienceDirect

Science of Computer Programming

www.elsevier.com/locate/scico

ELSEVIER

Understanding stack overflow code quality: A recommendation of caution

Sarah Meldrum, Sherlock A. Licorish*, Caitlin A. Owen, Bastin Tony Roy Savarimuthu

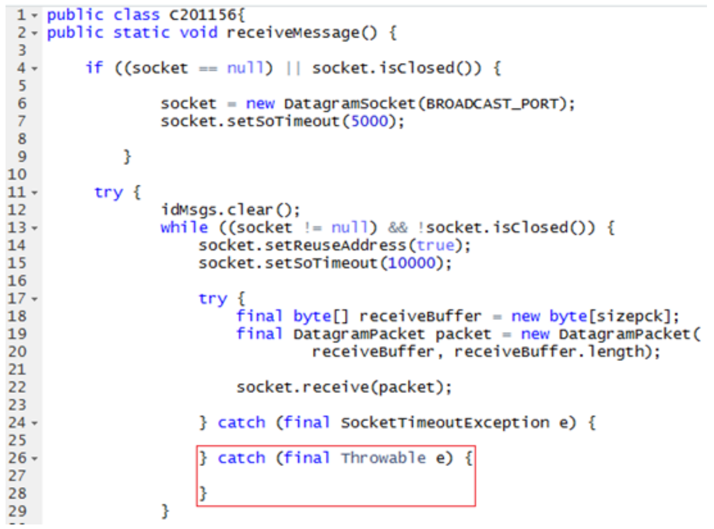
Department of Information Science, University of Otago, Dunedin, New Zealand

ARTICLE INFO

Article history:
Received 10 January 2020

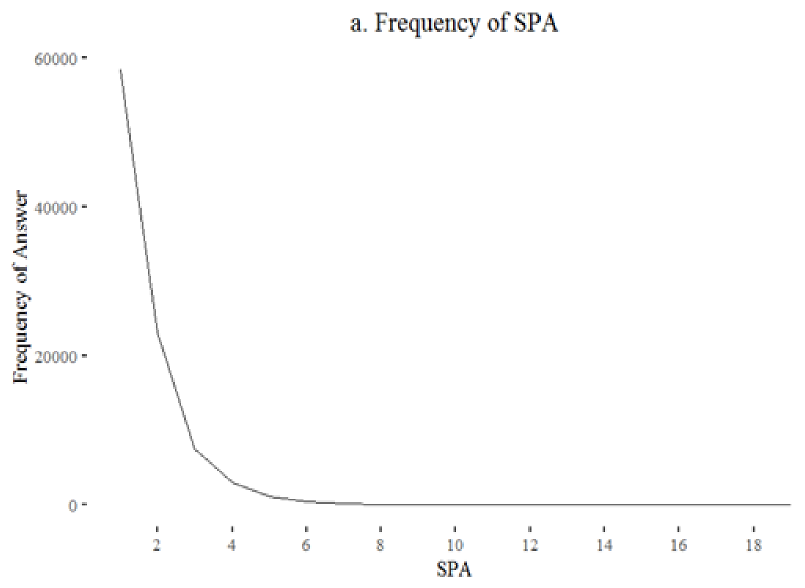
ABSTRACT

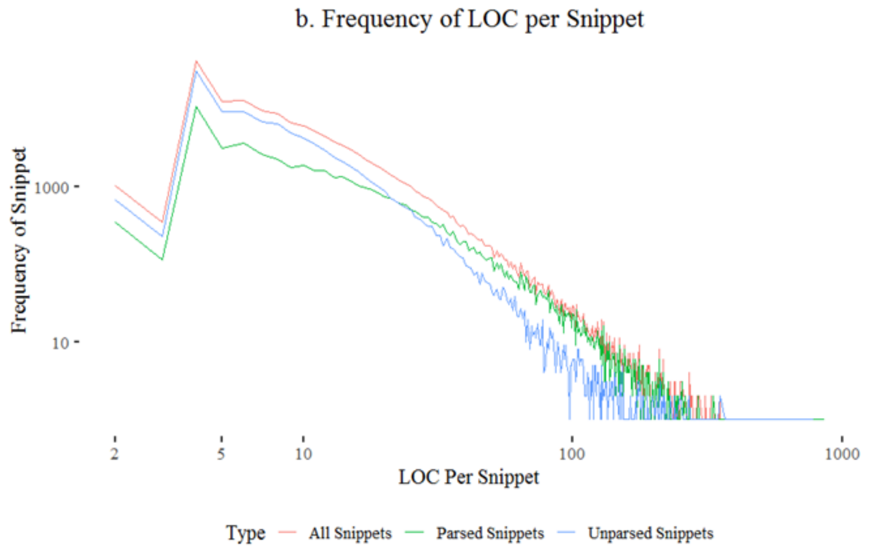
Community Question and Answer (CQA) platforms use the power of online groups to solve problems, or gain information. While these websites host useful information, it is

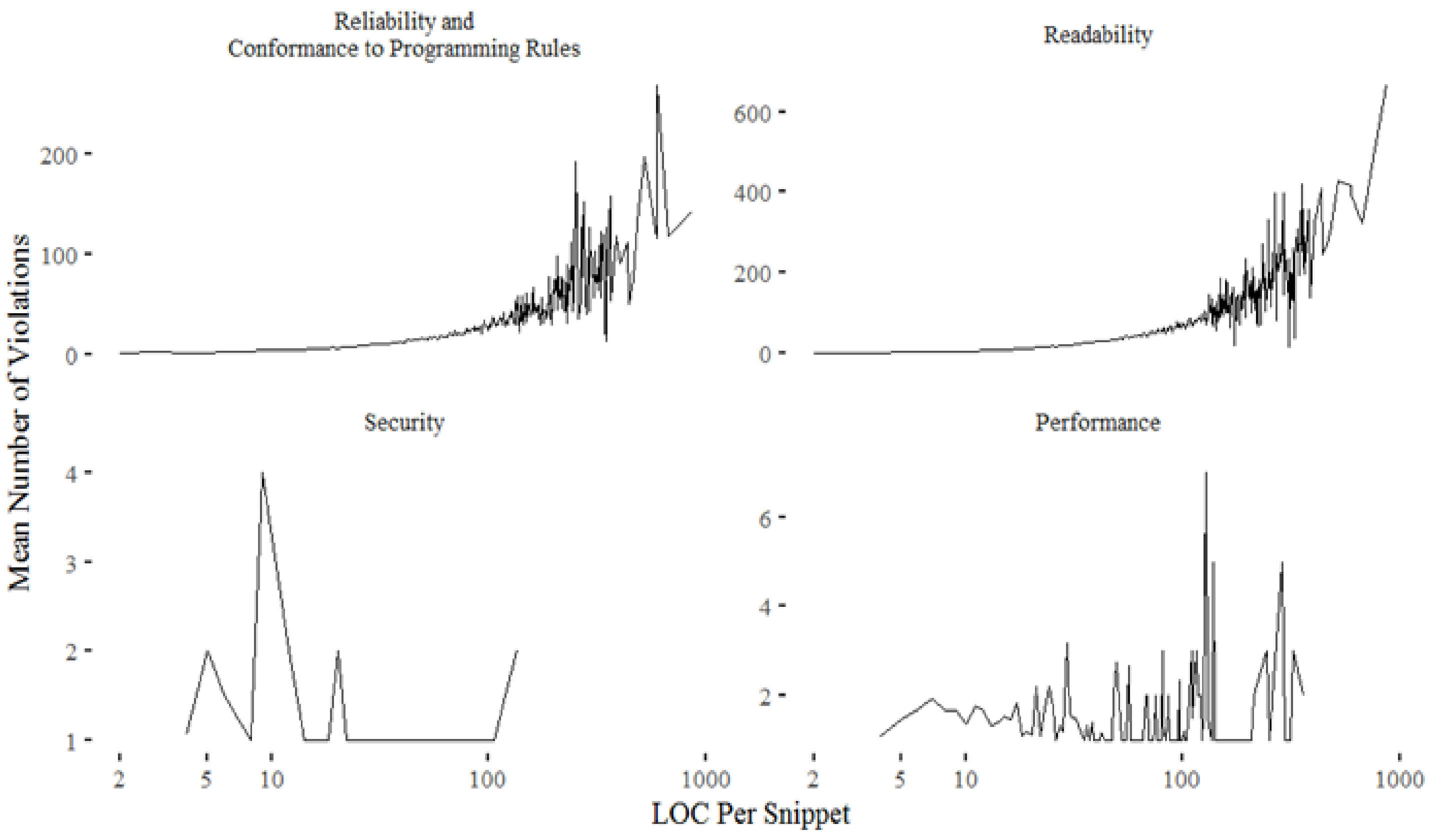


```
public class C201156{
public static void receiveMessage() {

    if ((socket == null) || socket.isClosed()) {

            socket = new DatagramSocket(BROADCAST_PORT);
            socket.setSoTimeout(5000);

        }

     try {
            idMsgs.clear();
            while ((socket != null) && !socket.isClosed()) {
                socket.setReuseAddress(true);
                socket.setSoTimeout(10000);

                try {
                    final byte[] receiveBuffer = new byte[sizepck];
                    final DatagramPacket packet = new DatagramPacket(
                            receiveBuffer, receiveBuffer.length);

                    socket.receive(packet);

                } catch (final SocketTimeoutException e) {

                } catch (final Throwable e) {

                }
            }
```

Current State of Play







Question: Can genetic improvement enhance online code snippets?

Preliminary Agenda

- Snippets (8,010) extracted from Stack Overflow for 2014, 2015, and 2016 using Stack Overflow's data explorer

- Answer posts which contained at least one "<code>" tag and were from a question tagged from Java were then sampled

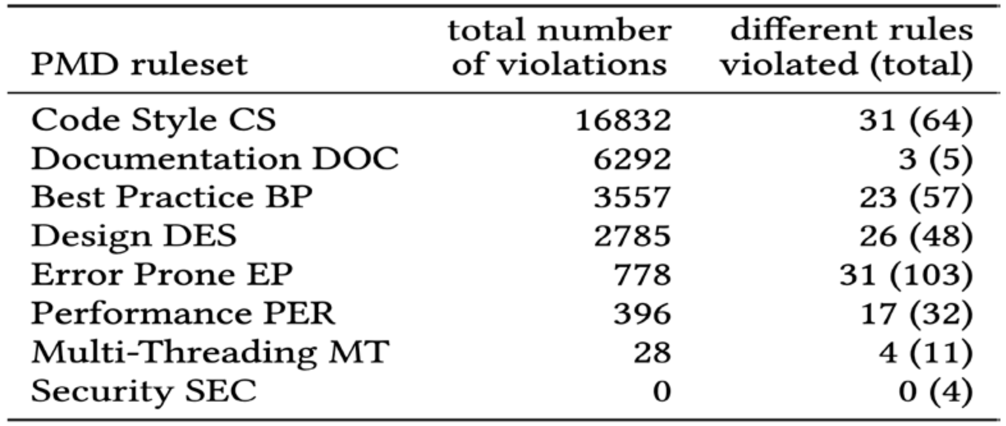
- Static checker PMD is used to identify faults, https://pmd.github.io/

- Genetic improvement tool GIN is used for code repair, https://github.com/gintool/gin

- We focus on performance related faults in Stack Overflow's code

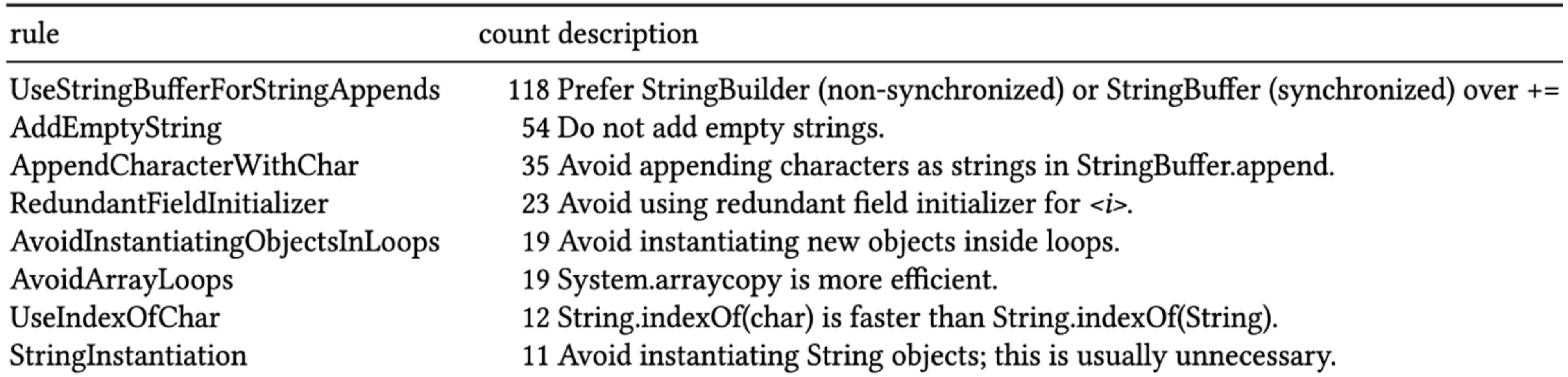
Characterizing PMD’s Treatment of 8,010 Snippets

- PMD finds 30,668 rule violations in 3,034 snippets, covering 135 of its 324 rules:



PMD ruleset	total number of violations	different rules violated (total)
Code Style CS	16832	31 (64)
Documentation DOC	6292	3 (5)
Best Practice BP	3557	23 (57)
Design DES	2785	26 (48)
Error Prone EP	778	31 (103)
Performance PER	396	17 (32)
Multi-Threading MT	28	4 (11)
Security SEC	0	0 (4)

- Examples of performance related rule violations:



rule	count	description
UseStringBufferForStringAppends	118	Prefer StringBuilder (non-synchronized) or StringBuffer (synchronized) over +=
AddEmptyString	54	Do not add empty strings.
AppendCharacterWithChar	35	Avoid appending characters as strings in StringBuffer.append.
RedundantFieldInitializer	23	Avoid using redundant field initializer for *<i>*.
AvoidInstantiatingObjectsInLoops	19	Avoid instantiating new objects inside loops.
AvoidArrayLoops	19	System.arraycopy is more efficient.
UseIndexOfChar	12	String.indexOf(char) is faster than String.indexOf(String).
StringInstantiation	11	Avoid instantiating String objects; this is usually unnecessary.

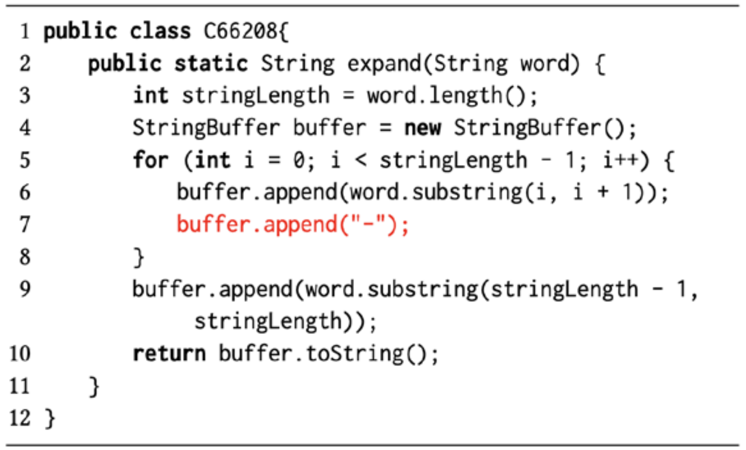
Characterizing GIN’s Single-edit Space

- GIN’s RandomSampler samples and runs 17,986 unique single-edit patches (DeleteLine, ReplaceLine, CopyLine, and SwapLine; and DeleteStatement, ReplaceStatement, CopyStatement, and SwapStatement; in total 31.4% compile)

- 770 patched files no longer have any performance issues – according to PMD

- 58 patches (from 44 unique files) produce compilable code without performance issues
  - 36 are Delete edits that delete the offending code
  - most others either replace or modify the offending code

```
Example: Code snippet C66208 with error AppendCharacterWithChar,
mutation DeleteStatement(64). The deleted statement is shown in red.
For more examples, see the GI@GECCO paper “Dissecting
Copy/Delete/Replace/Swap mutations: Insights from a GIN Case Study”.
```



```
public class C66208{
    public static String expand(String word) {
        int stringLength = word.length();
        StringBuffer buffer = new StringBuffer();
        for (int i = 0; i < stringLength - 1; i++) {
            buffer.append(word.substring(i, i + 1));
            buffer.append("-");
        }
        buffer.append(word.substring(stringLength - 1,
            stringLength));
        return buffer.toString();
    }
}
```

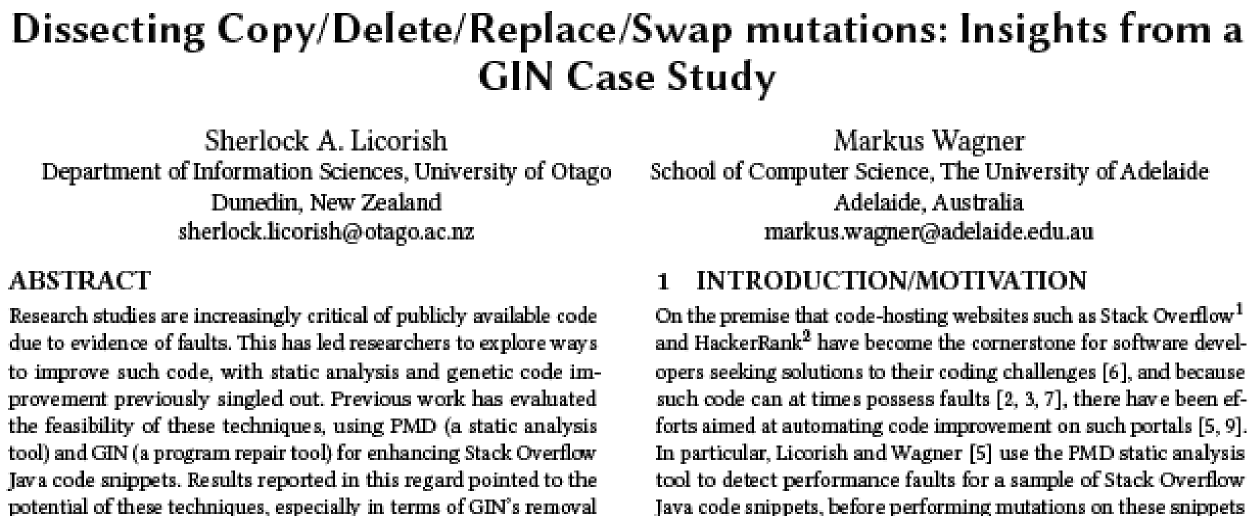
- Non-uniform effects of edits types
  - Copy edits attract disproportionally many violations
  - Delete edits perform best against the AvoidInstantiatingObjectsInLoops violations

Next Steps

- Better static analysis:
  - Mitigate false positive and trivial warnings
  - Improve parsing of non-compilable code
  - Crowd-source rules

- Better automated program improvement:
  - Bias sampling towards desired effects
  - Better code transformations
  - Other non-functional properties

Threat: GIN is normally accompanied by unit test suites to assess the validity of mutants. This work does not adopt such tests, and thus our successful patches that cleared performance issues and resulted in compilable code could have been inflated.



Dissecting Copy/Delete/Replace/Swap mutations: Insights from a GIN Case Study

Sherlock A. Licorish
Department of Information Sciences, University of Otago
Dunedin, New Zealand
sherlock.licorish@otago.ac.nz

Markus Wagner
School of Computer Science, The University of Adelaide
Adelaide, Australia
markus.wagner@adelaide.edu.au

ABSTRACT

Research studies are increasingly critical of publicly available code due to evidence of faults. This has led researchers to explore ways to improve such code, with static analysis and genetic code improvement previously singled out. Previous work has evaluated the feasibility of these techniques, using PMD (a static analysis tool) and GIN (a program repair tool) for enhancing Stack Overflow Java code snippets. Results reported in this regard pointed to the potential of these techniques, especially in terms of GIN's removal

1 INTRODUCTION/MOTIVATION

On the premise that code-hosting websites such as Stack Overflow[1] and HackerRank[2] have become the cornerstone for software developers seeking solutions to their coding challenges [6], and because such code can at times possess faults [2, 3, 7], there have been efforts aimed at automating code improvement on such portals [5, 9]. In particular, Licorish and Wagner [5] use the PMD static analysis tool to detect performance faults for a sample of Stack Overflow Java code snippets, before performing mutations on these snippets

Thanks!

Sarah Meldrum
Caitlin Owen
Tony Savarimuthu
Markus Wagner

National SCIENCE Challenges | SCIENCE FOR TECHNOLOGICAL INNOVATION | Kia kotahi mai – Te Ao Pūtaiao me Te Ao Hangarau

The Science for Technological Innovation
National Science Challenge Funding

Otago
Business School
Te Kura Pakihi

Commerce Research
Grant Funding

It Will Never Work in Theory

April 2023 Lightning Talks

Thank you!

Genetic improvement enhances online code snippets!

Your thoughts?

Sherlock A. Licorish
University of Otago
Dunedin, New Zealand
sherlock.licorish@otago.ac.nz