

# CS202: Software Tools and Techniques for CSE

## Lecture 4

Shouvick Mondal

shouvick.mondal@iitgn.ac.in  
August 2025

More on *diff*...

# git diff --help

```
shouvick@shouvick: ~/demo/temp
GIT-DIFF(1)                               Git Manual                               GIT-DIFF(1)

NAME
    git-diff - Show changes between commits, commit and working tree, etc

SYNOPSIS
    git diff [<options>] [<commit>] [--] [<path>...]
    git diff [<options>] --cached [<commit>] [--] [<path>...]
    git diff [<options>] <commit> <commit> [--] [<path>...]
    git diff [<options>] <blob> <blob>
    git diff [<options>] --no-index [--] <path> <path>

DESCRIPTION
    Show changes between the working tree and the index or a tree, changes between the index
    and a tree, changes between two trees, changes between two blob objects, or changes
    between two files on disk.

    git diff [<options>] [--] [<path>...]
    This form is to view the changes you made relative to the index (staging area for the
    next commit). In other words, the differences are what you could tell Git to further
    add to the index but you still haven't. You can stage these changes by using git-
    add(1).

    git diff [<options>] --no-index [--] <path> <path>
```

# Source code *diff* (unified)

github.com/SET-IITGN/cs202/commit/45a77e06f0b1b13907fcca80e7debd614fe0ba3f?diff=unified#diff-5cc5f480c15364f8b27b8941ae1e9498d3170d3a6e13421e7d527e7e1b5e5410

Filter files... src.c vuln.c

2 files changed +1 -2 lines changed

Search within code

src.c

```
@@ -12,7 +12,7 @@ int main() {
12 12     char *buffer;
13 13     char message[] = "Hello, worlcome to CS202";
14 14
15 -     int numBytes = strlen(message);
15 +     int numBytes = strlen(message);
16 16     buffer=(char*)malloc(numBytes+1);
17 17     memset(buffer, '\0', numBytes+1);
18 18     copyMemory(buffer, message, numBytes);
```

vuln.c

```
@@ -1,4 +1,3 @@
1 - //base file
2 1     #include <stdio.h>
3 2
4 3     void copyMemory(char* destination, const char* source, int numBytes) {
```

git diff <parent\_commit> <curr\_commit>

```
diff --git a/src.c b/src.c
index 0d6a029..85bb4fb 100644
--- a/src.c
+++ b/src.c
@@ -12,7 +12,7 @@ int main() {
     char *buffer;
     char message[] = "Hello, worlcome to CS202";

-    int numBytes = strlen(message);
+    int numBytes = strlen(message);
     buffer=(char*)malloc(numBytes+1);
     memset(buffer, '\0', numBytes+1);
     copyMemory(buffer, message, numBytes);
diff --git a/vuln.c b/vuln.c
index 838cba5..c1a0e33 100644
--- a/vuln.c
+++ b/vuln.c
@@ -1,4 +1,3 @@
-//base file
#include <stdio.h>
```

Empirical Software Engineering (2020) 25:790–823

<https://doi.org/10.1007/s10664-019-09772-z>

---

# How different are different *diff* algorithms in Git?

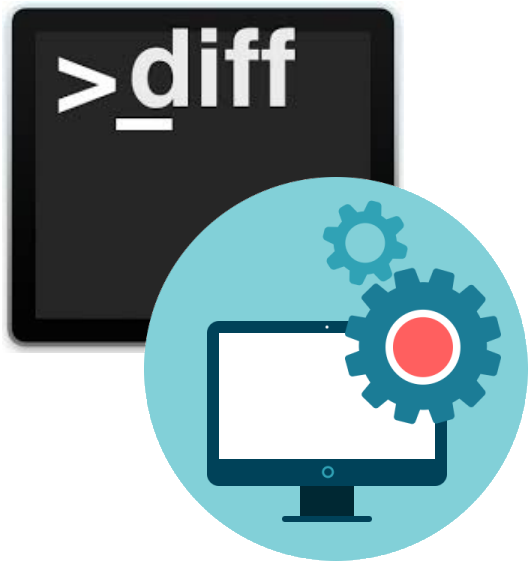
**Use `--histogram` for code changes**

**Yusuf Sulistyo Nugroho<sup>1</sup>  · Hideaki Hata<sup>1</sup> · Kenichi Matsumoto<sup>1</sup>**

Published online: 11 September 2019

© The Author(s) 2019

# *diff* is essential in SE research field



**Empirical Software  
Engineering Research**



**GitHub**

```
git diff [<options>] <commit> <commit> [--] [<path>...]
```

# Git offers 4 diff algorithms



## Documentation

`--diff-algorithm={algorithm name}`

- **Myers** default algorithm
- **Minimal** (improved Myers)
- **Patience** (try to give contextual diff)
- **Histogram** (enhanced Patience, normally faster)

Histogram was introduced in git 1.7.7 in 2011

Tree: 77bd3ea9f5 [git / Documentation / RelNotes / 1.7.7.txt](#)

Latest commit 783f05a on Oct 1, 2011 [History](#)

134 lines (95 sloc) | 5.28 KB [Raw](#) [Blame](#) [Icons](#)

```
1  Git v1.7.7 Release Notes
2  =====
3
4  Updates since v1.7.6
5  -----
6
7  * The scripting part of the codebase is getting prepared for i18n/l10n.
8
9  * Interix, Cygwin and Minix ports got updated.
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58  * "git diff" learned a "--histogram" option to use a different diff
59  generation machinery stolen from jgit, which might give better
60  performance.
61
62
63
64
65
66
67
68
69
70
71
72  * "git diff" had a weird worst case behaviour that can be triggered
73  when comparing files with potentially many places that could match.
74
```



# Different algorithms produce different *diff* outputs

```
1 @@ -169,11 +284,36 @@ public class AmqpMessage {
2     * @throws IllegalStateException if the message is read only.
3
4     9 added lines
5     + checkReadOnly();
6     + AmqpValue body = new AmqpValue(value);
7     + getWrappedMessage().setBody(body);
8     + }
9     +
10    + //----- Internal implementation -----
11    +
12    + private void checkReadOnly() throws IllegalStateException {
13        if (delivery != null) {
14            throw new IllegalStateException("Message is read only.");
15        }
16    + }
17
18    - AmqpValue body = new AmqpValue(value);
19    - getWrappedMessage().setBody(body);
```

2 deleted lines

a) **diff** extraction using Myers

## Differences:

- Number of changed lines
- Position of changed lines

```
1 @@ -169,11 +284,36 @@ public class AmqpMessage {
2     * @throws IllegalStateException if the message is read only.
3
4     4 deleted lines
5     - if (delivery != null) {
6     -     throw new IllegalStateException("Message is read only.");
7     - }
8     -
9     + checkReadOnly();
10    AmqpValue body = new AmqpValue(value);
11    getWrappedMessage().setBody(body);
12    }
13    +
14    + //----- Internal implementation -----
15    +
16    + private void checkReadOnly() throws IllegalStateException {
17    +     if (delivery != null) {
18    +         throw new IllegalStateException("Message is read only.");
19    +     }
20    + }
```

9 added lines

b) **diff** extraction using Histogram

# https://github.com/ome/openmicroscopy

The screenshot shows the GitHub interface for a specific commit. At the top, the browser address bar displays the URL: `github.com/ome/openmicroscopy/commit/844e0fde447d2d069fb17c480e95acf4d372afc4#diff-07322c93ef4fb3f0dd245932b74b10e1`. The repository name `ome / openmicroscopy` is shown in the header, along with a search bar and navigation icons. Below the header, a tab bar includes `<> Code`, `Issues 26`, `Pull requests 6`, `Actions`, `Projects`, `Wiki`, `Security`, and `Insights`.

The main section is titled **Commit 844e0fd**. It indicates that the commit was made by `bwzloranger` on `Aug 4, 2010`. The commit message is `Fixes for importer findbugs`, followed by the `git-svn-id` string: `file:///home/svn/omero/trunk@7668 05709c45-44f0-0310-885b-81a1db45b4a6`. The commit is linked to the `develop` branch and the `v5.6.14` tag, with a parent commit `cb85815`.

On the left, a file explorer shows the directory structure, with `components/tools/OmeroIm...` expanded to show files like `FileQueueHandler.java`, `FileQueueTable.java`, `GuiCommonElements.java`, `HistoryTable.java`, and `HistoryTableAbstractDataS...`.

The right side displays the diff for **7 files changed**, with a total of `+114 -88` lines changed. The files listed are `components/tools/OmeroImporter/src/ome/formats/importer/gui/FileQueueHandler.java`, `components/tools/OmeroImporter/src/ome/formats/importer/gui/FileQueueTable.java`, and `components/tools/OmeroImporter/src/ome/formats/importer/gui/GuiCommonElements.java`. The third file is highlighted with a red box, showing a diff of `+11 -7` lines. The visible code snippet is: `@@ -673,17 +673,21 @@ public static JComboBox addComboBox(Container container, String label,`

# git diff --diff-algorithm=myers ...

```
components/tools/OmeroImporter/src/ome/formats/importer/gui/GuiCommonElements.java
@@ -673,17 +673,21 @@ public static JComboBox addComboBox(Container container, S
673 673      */
674 674      public static ImageIcon getImageIcon(String path)
675 675      {
676 +          if (path == null)
677 +          {
678 +              log.error("Icon path is null");
679 +              return null;
680 +          }
681 +
682      java.net.URL imgURL = GuiImporter.class.getResource(path);
683
684 -          if (imgURL != null)
685 +          if (imgURL == null)
686 +          {
687 -              return new ImageIcon(imgURL);
688 -          }
689 -          else
690 -          {
691 -              log.error("Couldn't find icon: " + imgURL);
692 +              log.error("Couldn't find icon: " + imgURL);
693 +              return null;
694 +          }
695 +          return new ImageIcon(imgURL);
696      }
697 697  }
```

```
shouvik@shouvik:~/demo/openmicroscopy$ git diff --diff-algorithm=my
nts/tools/OmeroImporter/src/ome/formats/importer/gui/GuiCommonElement
diff --git a/components/tools/OmeroImporter/src/ome/formats/importer/
omponents/tools/OmeroImporter/src/ome/formats/importer/gui/GuiCommonE
index 244701eb0e..d783fb51b6 100644
--- a/components/tools/OmeroImporter/src/ome/formats/importer/gui/Gui
+++ b/components/tools/OmeroImporter/src/ome/formats/importer/gui/Gui
@@ -673,17 +673,21 @@ public class GuiCommonElements
    */
    public static ImageIcon getImageIcon(String path)
    {
+        if (path == null)
+        {
+            log.error("Icon path is null");
+            return null;
+        }
        java.net.URL imgURL = GuiImporter.class.getResource(path);

-        if (imgURL != null)
+        if (imgURL == null)
+        {
+            return new ImageIcon(imgURL);
+        }
        else
        {
            log.error("Couldn't find icon: " + imgURL);
+            log.error("Couldn't find icon: " + imgURL);
+            return null;
+        }
        return null;
+    else
+        return new ImageIcon(imgURL);
    }

    /**
```

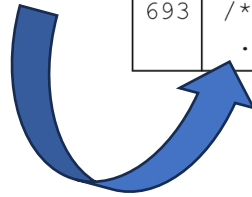
# A set of changes from an older file into a newer file

## Version 1

```
673  ... other code here ...
673  */
674  public static ImageIcon getImageIcon(String path)
675  {
676      java.net.URL imgURL = GuiImporter.class.getResource(path);
677
678      if (imgURL != null)
679      {
680          return new ImageIcon(imgURL);
681      }
682      else
683      {
684          log.error("Couldn't find icon: " + imgURL);
685      }
686      return null;
687  }
688
689  /**
689  ... other code here ...
```

## Version 2

```
673  ... other code here ...
673  */
674  public static ImageIcon getImageIcon(String path)
675  {
676      if (path == null)
677      {
678          log.error("Icon path is null");
679          return null;
680      }
681
682      java.net.URL imgURL = GuiImporter.class.getResource(path);
683
684      if (imgURL == null)
685      {
686          log.error("Couldn't find icon: " + imgURL);
687          return null;
688      }
689      else
690          return new ImageIcon(imgURL);
691  }
692
693  /**
693  ... other code here ...
```



# git diff -w --ignore-blank-lines --diff-algorithm=myers

```
shouvick@shouvick:~/demo/openmicroscopy$ git diff -w --ignore-blank-lines --diff-algorithm=myers cb85815 84
Elements.java
diff --git a/components/tools/OmeroImporter/src/ome/formats/importer/gui/GuiCommonElements.java b/component
index 244701eb0e..d783fb51b6 100644
--- a/components/tools/OmeroImporter/src/ome/formats/importer/gui/GuiCommonElements.java
+++ b/components/tools/OmeroImporter/src/ome/formats/importer/gui/GuiCommonElements.java
@@ -673,18 +673,22 @@ public class GuiCommonElements
    */
    public static ImageIcon getImageIcon(String path)
    {
-        java.net.URL imgURL = GuiImporter.class.getResource(path);
-
-        if (imgURL != null)
+        if (path == null)
+        {
+            return new ImageIcon(imgURL);
+            log.error("Icon path is null");
+            return null;
+        }
-        else
+        java.net.URL imgURL = GuiImporter.class.getResource(path);
+        if (imgURL == null)
+        {
+            log.error("Couldn't find icon: " + imgURL);
+            return null;
+        }
+        else
+            return new ImageIcon(imgURL);
+    }

    /**
     * Quit Confirmation pop up dialog

```

# git diff -w --ignore-blank-lines --diff-algorithm=myers

Version 1

```
... other code here ...
673 */
674 public static ImageIcon getImageIcon(String path)
675 {
676     java.net.URL imgURL = GuiImporter.class.getResource(path);
677
678     if (imgURL != null)
679     {
680         return new ImageIcon(imgURL);
681     }
682     else
683     {
684         log.error("Couldn't find icon: " + imgURL);
685     }
686     return null;
687 }
688 /**
689 ... other code here ...
```

Version 2

```
... other code here ...
673 */
674 public static ImageIcon getImageIcon(String path)
675 {
676     if (path == null)
677     {
678         log.error("Icon path is null");
679         return null;
680     }
681     java.net.URL imgURL = GuiImporter.class.getResource(path);
682
683     if (imgURL == null)
684     {
685         log.error("Couldn't find icon: " + imgURL);
686         return null;
687     }
688     else
689         return new ImageIcon(imgURL);
690 }
691 /**
692 ... other code here ...
693
```

Start scanning from line#1 (relative within context) in both v1 and v2. Match up v1->v2 or v2->v1 whichever comes first. Then align v1 and v2 for further comparison. Perform these actions recursively.

After pairing up first three matching lines... (step 1 of n)

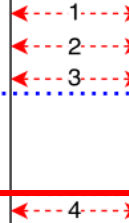
# git diff -w --ignore-blank-lines --diff-algorithm=myers

Version 1

```
... other code here ...
673 */
674 public static ImageIcon getImageIcon(String path)
675 {
676     java.net.URL imgURL = GuiImporter.class.getResource(path);
677
678     if (imgURL != null)
679     {
680         return new ImageIcon(imgURL);
681     }
682     else
683     {
684         log.error("Couldn't find icon: " + imgURL);
685     }
686     return null;
687 }
688 /**
689 ... other code here ...
```

Version 2

```
... other code here ...
673 */
674 public static ImageIcon getImageIcon(String path)
675 {
676     if (path == null)
677     {
678         log.error("Icon path is null");
679         return null;
680     }
681     java.net.URL imgURL = GuiImporter.class.getResource(path);
682
683     if (imgURL == null)
684     {
685         log.error("Couldn't find icon: " + imgURL);
686         return null;
687     }
688     else
689     {
690         return new ImageIcon(imgURL);
691     }
692 }
693 /**
... other code here ...
```



*After pairing up the fourth matching line... (step 2 of n)*



# git diff -w --ignore-blank-lines --diff-algorithm=myers

Version 1

```
... other code here ...
673 */
674 public static ImageIcon getImageIcon(String path)
675 {
676 -     java.net.URL imgURL = GuiImporter.class.getResource(path);
677 -
678 -     if (imgURL != null)
679     {
680 -         return new ImageIcon(imgURL);
681     }
682 - else
683     {
684         log.error("Couldn't find icon: " + imgURL);
685 -     }
686     return null;
687 }
688 /**
689 ... other code here ...
```

Version 2

```
... other code here ...
673 */
674 public static ImageIcon getImageIcon(String path)
675 {
676 +     if (path == null)
677     {
678 +         log.error("Icon path is null");
679 +         return null;
680     }
681 +
682 +     java.net.URL imgURL = GuiImporter.class.getResource(path);
683 +
684 +     if (imgURL == null)
685     {
686         log.error("Couldn't find icon: " + imgURL);
687     }
688     return null;
689 + else
690 +     return new ImageIcon(imgURL);
691 + }
692 /**
693 ... other code here ...
```

*After pairing up all matching lines... (step n of n)*



# git diff -w --ignore-blank-lines --diff-algorithm=myers

```
shouvick@shouvick:~/demo/openmicroscopy$ git diff -w --ignore-blank-lines --diff-algorithm=myers cb85815 84
Elements.java
diff --git a/components/tools/OmeroImporter/src/ome/formats/importer/gui/GuiCommonElements.java b/component
index 244701eb0e..d783fb51b6 100644
--- a/components/tools/OmeroImporter/src/ome/formats/importer/gui/GuiCommonElements.java
+++ b/components/tools/OmeroImporter/src/ome/formats/importer/gui/GuiCommonElements.java
@@ -673,18 +673,22 @@ public class GuiCommonElements
    */
    public static ImageIcon getImageIcon(String path)
    {
-        java.net.URL imgURL = GuiImporter.class.getResource(path);
-
-        if (imgURL != null)
+        if (path == null)
+        {
-            return new ImageIcon(imgURL);
+            log.error("Icon path is null");
+            return null;
+        }
-        else
+        java.net.URL imgURL = GuiImporter.class.getResource(path);
+        if (imgURL == null)
+        {
-            }
+            log.error("Couldn't find icon: " + imgURL);
-            return null;
+        }
+        else
+            return new ImageIcon(imgURL);
+    }

    /**
     * Quit Confirmation pop up dialog

```

... -w --ignore-blank-lines --diff-algorithm=histogram

```
shouvick@shouvick:~/demo/openmicroscopy$ git diff -w --ignore-blank-lines --diff-algorithm=histogram cb85815
commonElements.java
diff --git a/components/tools/OmeroImporter/src/ome/formats/importer/gui/GuiCommonElements.java b/components
index 244701eb0e..d783fb51b6 100644
--- a/components/tools/OmeroImporter/src/ome/formats/importer/gui/GuiCommonElements.java
+++ b/components/tools/OmeroImporter/src/ome/formats/importer/gui/GuiCommonElements.java
@@ -673,18 +673,22 @@ public class GuiCommonElements
    */
    public static ImageIcon getImageIcon(String path)
    {
+   if (path == null)
+   {
+       log.error("Icon path is null");
+       return null;
+   }
    java.net.URL imgURL = GuiImporter.class.getResource(path);

    if (imgURL != null)
    {
        return new ImageIcon(imgURL);
    }
    else
+   if (imgURL == null)
    {
        log.error("Couldn't find icon: " + imgURL);
    }
    return null;
+   else
+   return new ImageIcon(imgURL);
+   }

    /**
     * Quit Confirmation pop up dialog
```

... -w --ignore-blank-lines --diff-algorithm=histogram

*Constructs like function definition/declarations are unique. Singletons like '{' and '\n' are noises.*

Version 1

```
673 ... other code here ...
674 */
675 public static ImageIcon getImageIcon(String path)
676 {
677     java.net.URL imgURL = GuiImporter.class.getResource(path);
678     if (imgURL != null)
679     {
680         return new ImageIcon(imgURL);
681     }
682     else
683     {
684         log.error("Couldn't find icon: " + imgURL);
685     }
686     return null;
687 }
688 /**
689 ... other code here ...
```

Version 2

```
673 ... other code here ...
674 */
675 public static ImageIcon getImageIcon(String path)
676 {
677     if (path == null)
678     {
679         log.error("Icon path is null");
680         return null;
681     }
682     java.net.URL imgURL = GuiImporter.class.getResource(path);
683     if (imgURL == null)
684     {
685         log.error("Couldn't find icon: " + imgURL);
686         return null;
687     }
688     else
689         return new ImageIcon(imgURL);
690 }
691 /**
692 ... other code here ...
693
```

upper section

separator

lower section

*Compute histogram of each line (within context) in v1. Extract each line from v2 and try to match with v1 (in sequence). The unique match or low-occurrence common match is the separator. This partitions the code into: <upper, separator, lower>. Align v1 and v2 respecting the partition. Perform these actions recursively.*

*After locating the first separator... (step 1 of n)*

... -w --ignore-blank-lines --diff-algorithm=histogram

#### Version 1

```
673 ... other code here ...
674 */
675 public static ImageIcon getImageIcon(String path)
676 {
677     java.net.URL imgURL = GuiImporter.class.getResource(path);
678     if (imgURL != null)
679     {
680         return new ImageIcon(imgURL);
681     }
682     else
683     {
684         log.error("Couldn't find icon: " + imgURL);
685     }
686     return null;
687 }
688 /**
689 ... other code here ...
```

#### Version 2

```
673 ... other code here ...
674 */
675 public static ImageIcon getImageIcon(String path)
676 {
677     if (path == null)
678     {
679         log.error("Icon path is null");
680         return null;
681     }
682     java.net.URL imgURL = GuiImporter.class.getResource(path);
683     if (imgURL == null)
684     {
685         log.error("Couldn't find icon: " + imgURL);
686         return null;
687     }
688     else
689     {
690         return new ImageIcon(imgURL);
691     }
692 }
693 /**
... other code here ...
```

upper section

separator

lower section

*After locating the second separator... (step 2 of n)*

... -w --ignore-blank-lines --diff-algorithm=histogram

Version 1

```
... other code here ...
673 */
674 public static ImageIcon getImageIcon(String path)
675 {
676     java.net.URL imgURL = GuiImporter.class.getResource(path);
677
678 -     if (imgURL != null)
679 -     {
680 -         return new ImageIcon(imgURL);
681 -     }
682 -     else
683     {
684         log.error("Couldn't find icon: " + imgURL);
685 -     }
686     return null;
687 }
688
689 /**
690 ... other code here ...
```

Version 2

```
... other code here ...
673 */
674 public static ImageIcon getImageIcon(String path)
675 {
676 +     if (path == null)
677 +     {
678 +         log.error("Icon path is null");
679 +         return null;
680 +     }
681 +
682     java.net.URL imgURL = GuiImporter.class.getResource(path);
683
684 +     if (imgURL == null)
685     {
686         log.error("Couldn't find icon: " + imgURL);
687         return null;
688     }
689 +     else
690 +         return new ImageIcon(imgURL);
691 + }
692
693 /**
694 ... other code here ...
```

*After locating all separators... (step n of n)*

# Myers versus Histogram



```
... other code here ...
1  */
2  public static ImageIcon getImageIcon(String path)
3  {
4  -   java.net.URL imgURL = GuiImporter.class.getResource(path);
5  -
6  -   if (imgURL != null)
7  +   if (path == null)
8       {
9  -   return new ImageIcon(imgURL);
10  +   log.error("Icon path is null");
11  +   return null;
12       }
13  -   else
14  +
15  +   java.net.URL imgURL = GuiImporter.class.getResource(path);
16  +
17  +   if (imgURL == null)
18       {
19           log.error("Couldn't find icon: " + imgURL);
20  -   }
21           return null;
22       }
23  +   else
24  +   return new ImageIcon(imgURL);
25  +   }
26
27  /**
... other code here ...
```

```
... other code here ...
1  */
2  public static ImageIcon getImageIcon(String path)
3  {
4  +   if (path == null)
5  +   {
6  +       log.error("Icon path is null");
7  +       return null;
8  +   }
9  +
10         java.net.URL imgURL = GuiImporter.class.getResource(path);
11
12  -   if (imgURL != null)
13  -   {
14  -       return new ImageIcon(imgURL);
15  -   }
16  -   else
17  +   if (imgURL == null)
18       {
19           log.error("Couldn't find icon: " + imgURL);
20  -   }
21           return null;
22       }
23  +   else
24  +   return new ImageIcon(imgURL);
25  +   }
26
27  /**
... other code here ...
```

Empirical Software Engineering (2020) 25:790–823

<https://doi.org/10.1007/s10664-019-09772-z>

---

# How different are different *diff* algorithms in Git?

Use `--histogram` for code changes

Yusuf Sulistyo Nugroho<sup>1</sup>  · Hideaki Hata<sup>1</sup> · Kenichi Matsumoto<sup>1</sup>

Published online: 11 September 2019

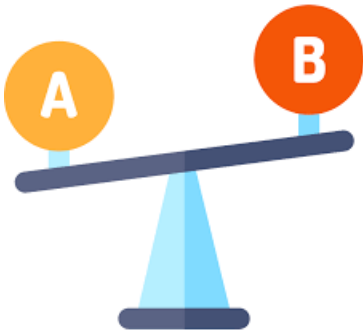
© The Author(s) 2019



# Two sequential analyses



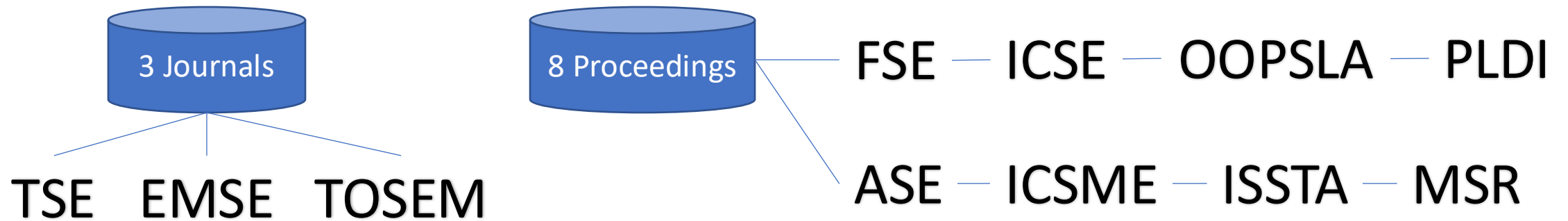
- Systematic Mapping Study
  - How previous studies used git diff?



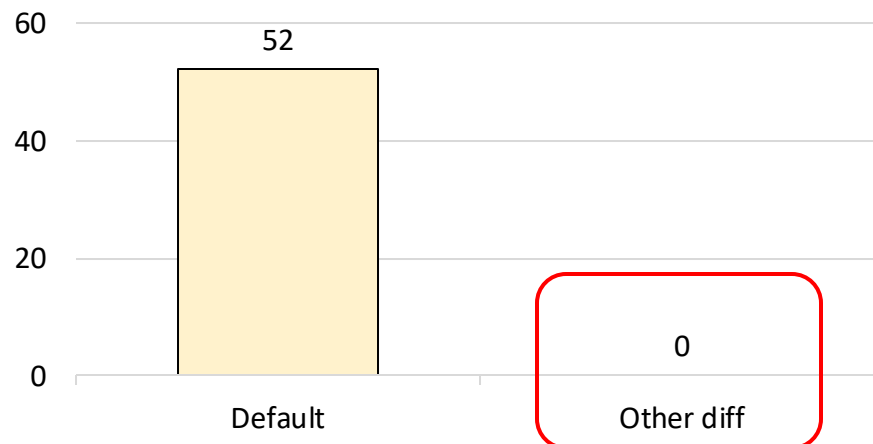
- Comparisons Study
  - differences of *diff* outputs between Myers and Histogram



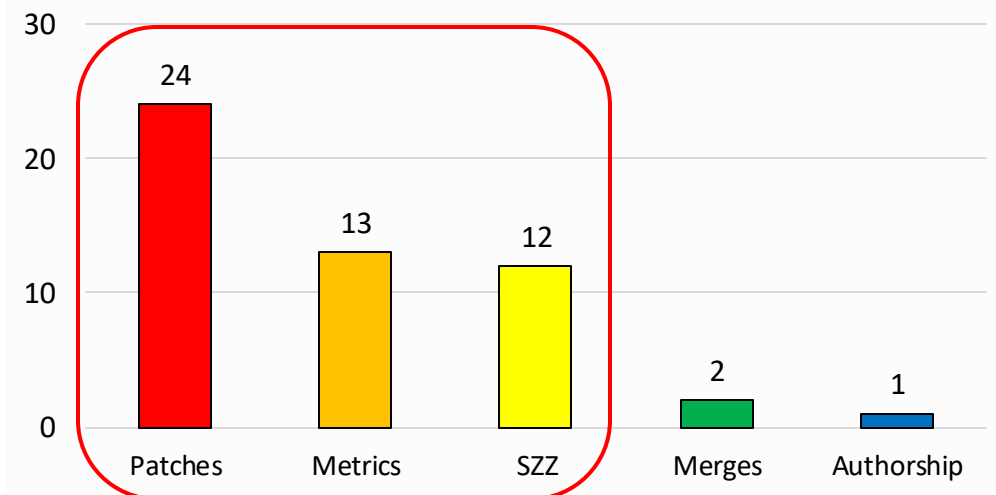
# Results of Mapping Study



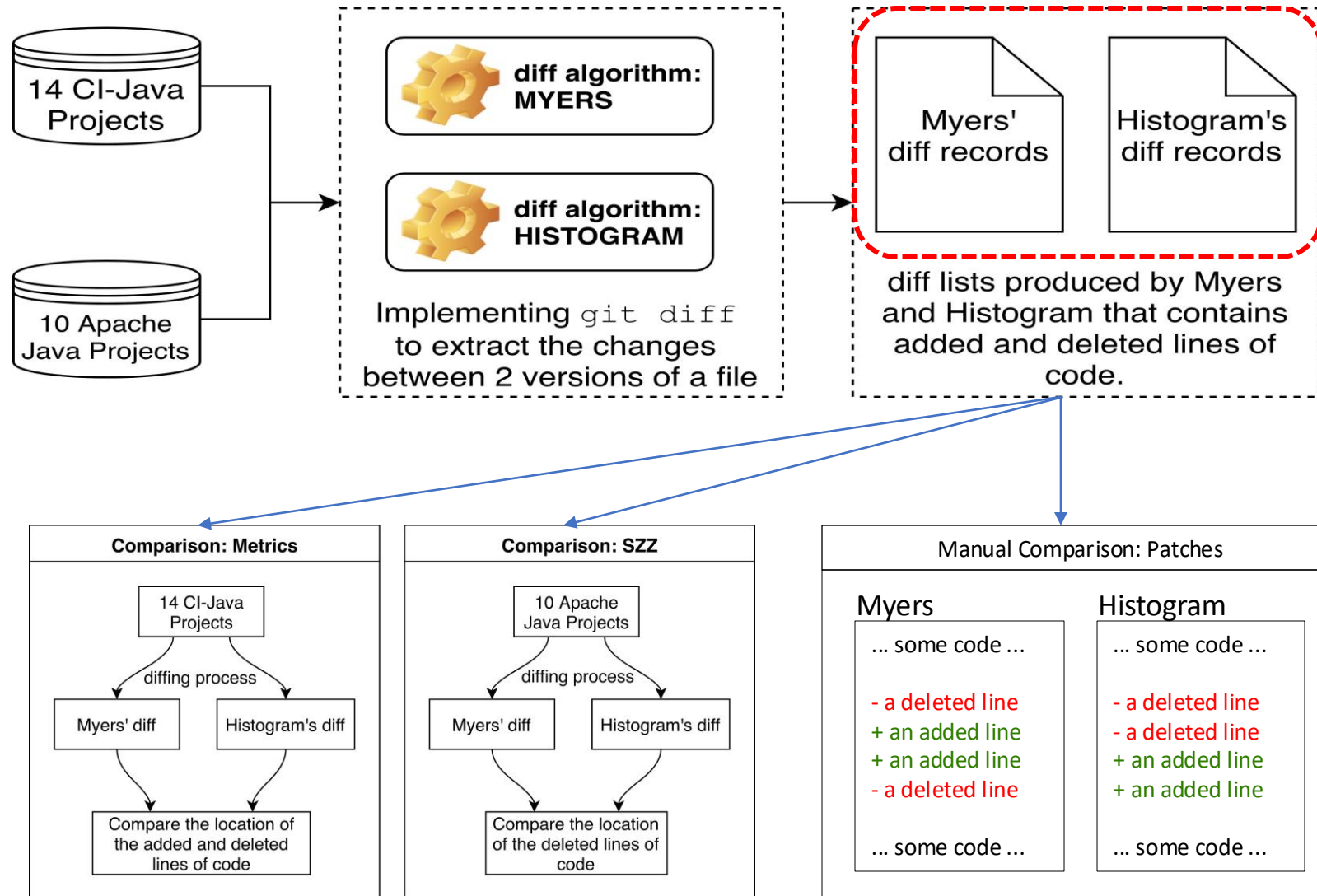
Type of diff algorithm

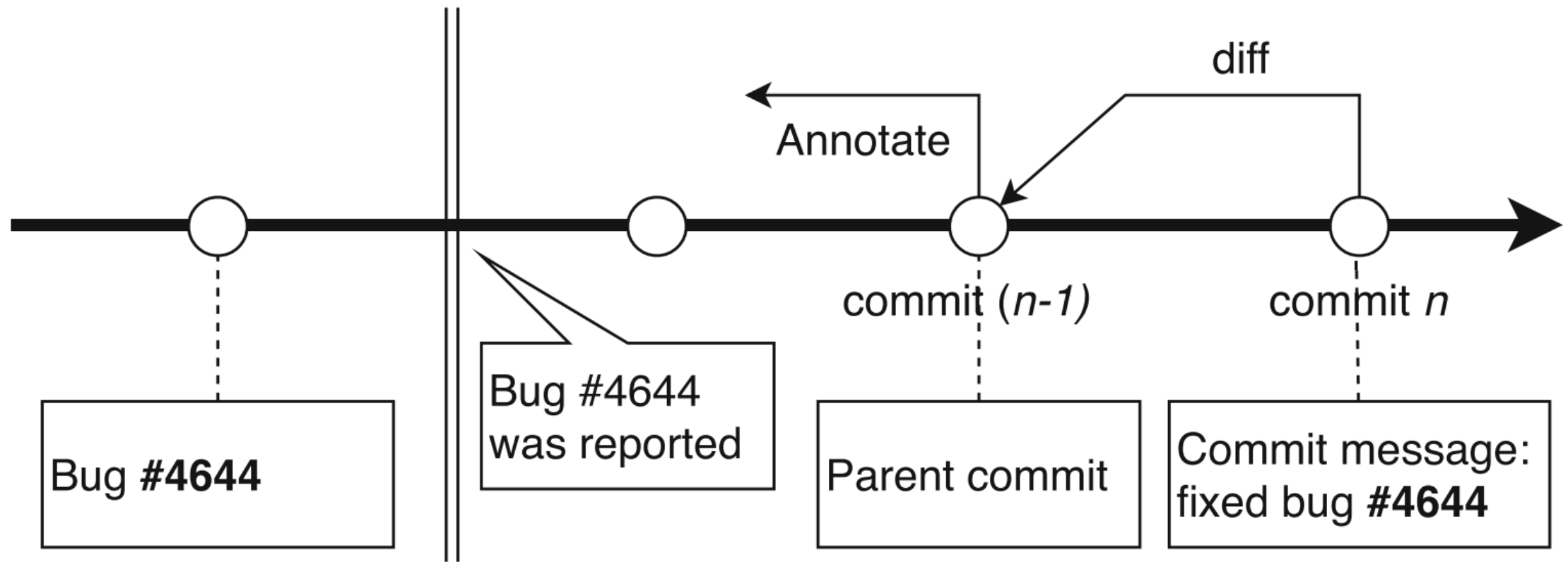


Top 3 purposes of using *diff*



# Comparing *diff* outputs in 3 applications





**SZZ:** Locating bug-introducing changes

# Recap of SZZ: Collection of Bug-fix Commits in **minecpp**

The collection of bug-fixing commits involved multiple strategies:

- Commit messages were analyzed to identify bug fix commits, leveraging keywords related to bug fixes. A total of **52 keywords** are used.
- Also, **Regular expressions** were utilized to perform precise searches for bug fix commit patterns
- The **SZZ algorithm** was applied to trace back and identify the buggy files associated with bug fix commits

“fixed ”, “ bug”, “fixes ”, “fix ”, “ fix”, “ fixed”, “ fixes”, “crash”, “solves”, “ resolves”, “resolves ”, “ issue”, “issue ”, “regression”, “fall back”, “assertion”, “coverity”, “reproducible”, “stack-wanted”, “steps-wanted”, “testcase”, “failur”, “fail”, “npe ”, “ npe”, “except”, “broken”, “differential testing”, “error”, “hang ”, “ hang”, “test fix”, “steps to reproduce”, “crash”, “assertion”, “failure”, “leak”, “stack trace”, “heap overflow”, “freez”, “problem ”, “ problem”, “ overflow”, “overflow ”, “avoid ”, “ avoid”, “workaround ”, “ workaround”, “break ”, “ break”, “ stop”, “stop ”

Commit messages with  
52 keywords [6] (c.f. Sec. III-C)

```
+ RegEx. [7]: ‘.((solv(ed|es|e|ing))  
|(fix(s|es|ing|ed)?))’  
|((error|bug|issue)(s)?)).’
```

→ SZZ Algorithm [8]

[J. **S**liwerski, T. **Z**immermann, and A. **Z**eller; (2005)]

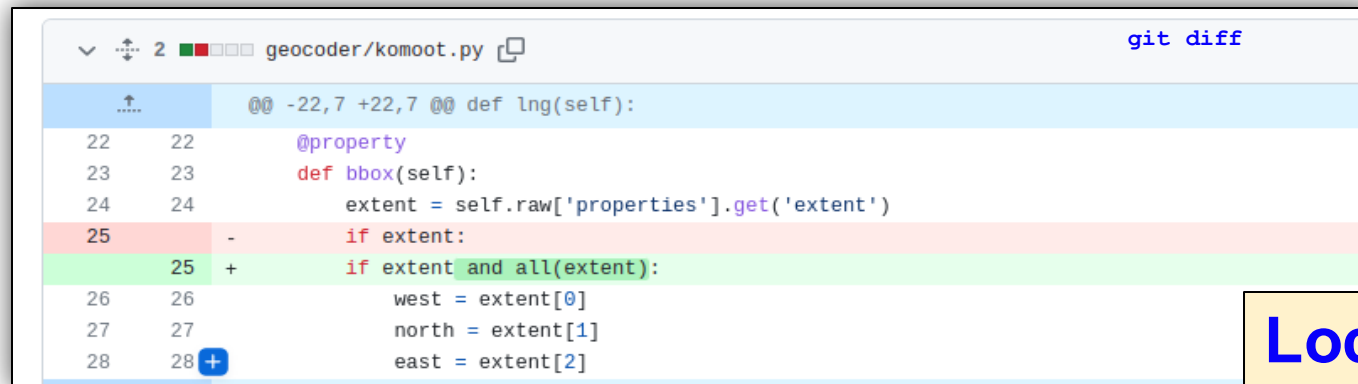
**SZZ** takes a commit (**bug-fixing**) as the input and returns a list of commits (**bug-introducing**) that last **added** the **deleted** lines in the input commit.

# Recap of (fixed-1, fixed): Extracting Code Snippets in minecpp

The extraction of code snippets is done as follows:

## Step 1:

- **Location:** The location of bug is found using **git diff** between **fixed - 1** and **fixed commit**. The location here indicates the **line number** in which the bug is detected and fixed.



```
git diff
@@ -22,7 +22,7 @@ def lng(self):
22 22         @property
23 23         def bbox(self):
24 24             extent = self.raw['properties'].get('extent')
25 -         if extent:
25 +         if extent and all(extent):
26 26             west = extent[0]
27 27             north = extent[1]
28 28             east = extent[2]
```

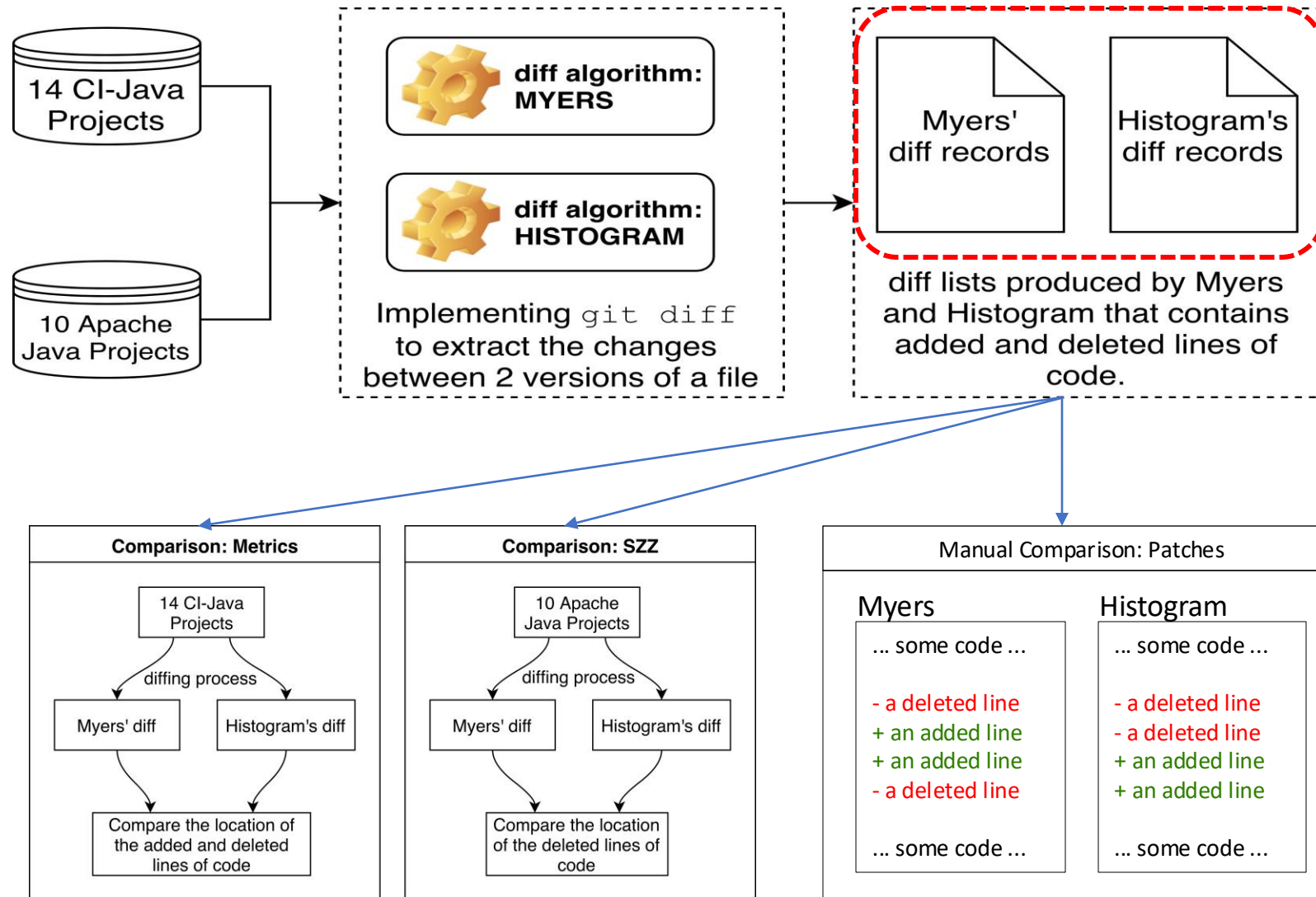
a **modification**  
typically involves an  
**addition** after a **deletion**

**Loc:** [before: 25, after: 25]

**Bug Type:** fix komoot.py when extent is none

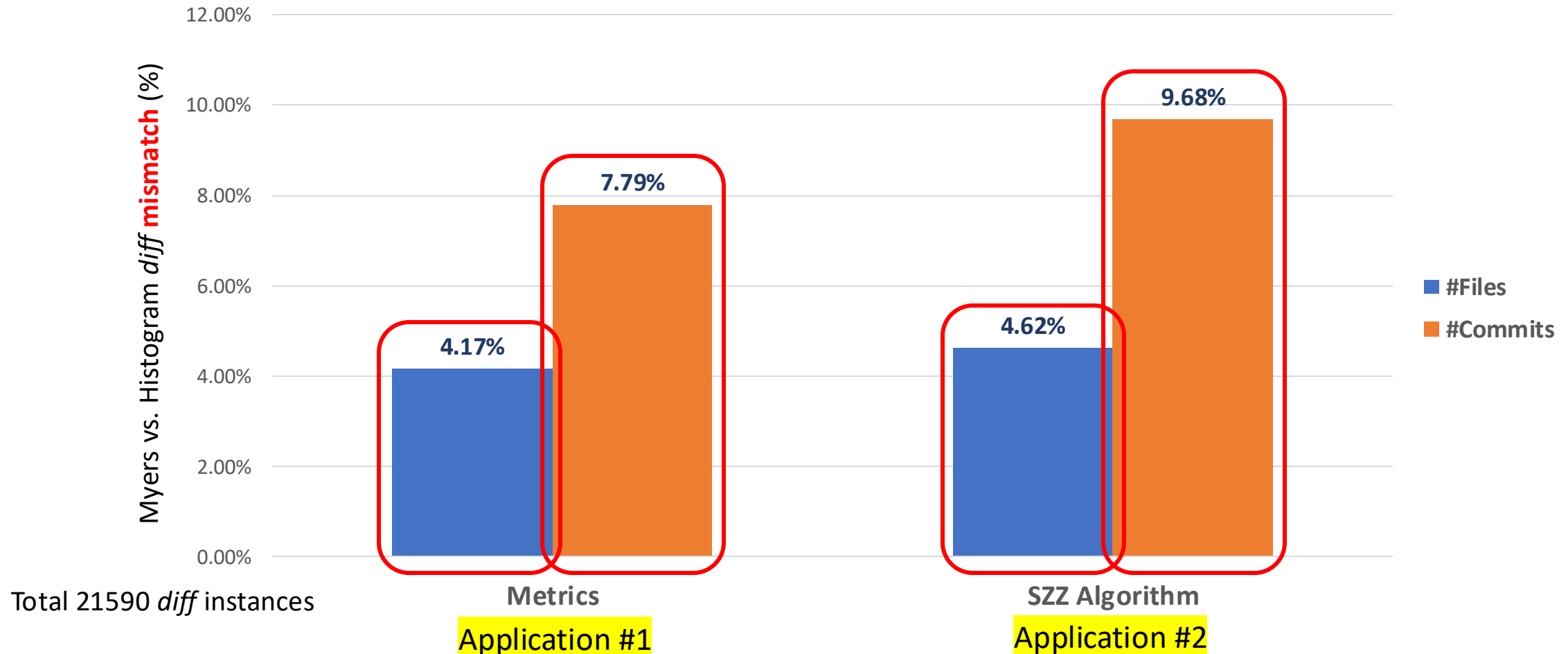
**Commit Message:** fix calls to BBox with invalid input

# Comparing *diff* outputs in 3 applications



# Different *diff* algorithms can report different location (mismatch) of identified changed lines of code

Frequency of different location of identified changed lines of code



# Empirically, Histogram is better for describing code changes

A sample of 377/21590 *diff* instances

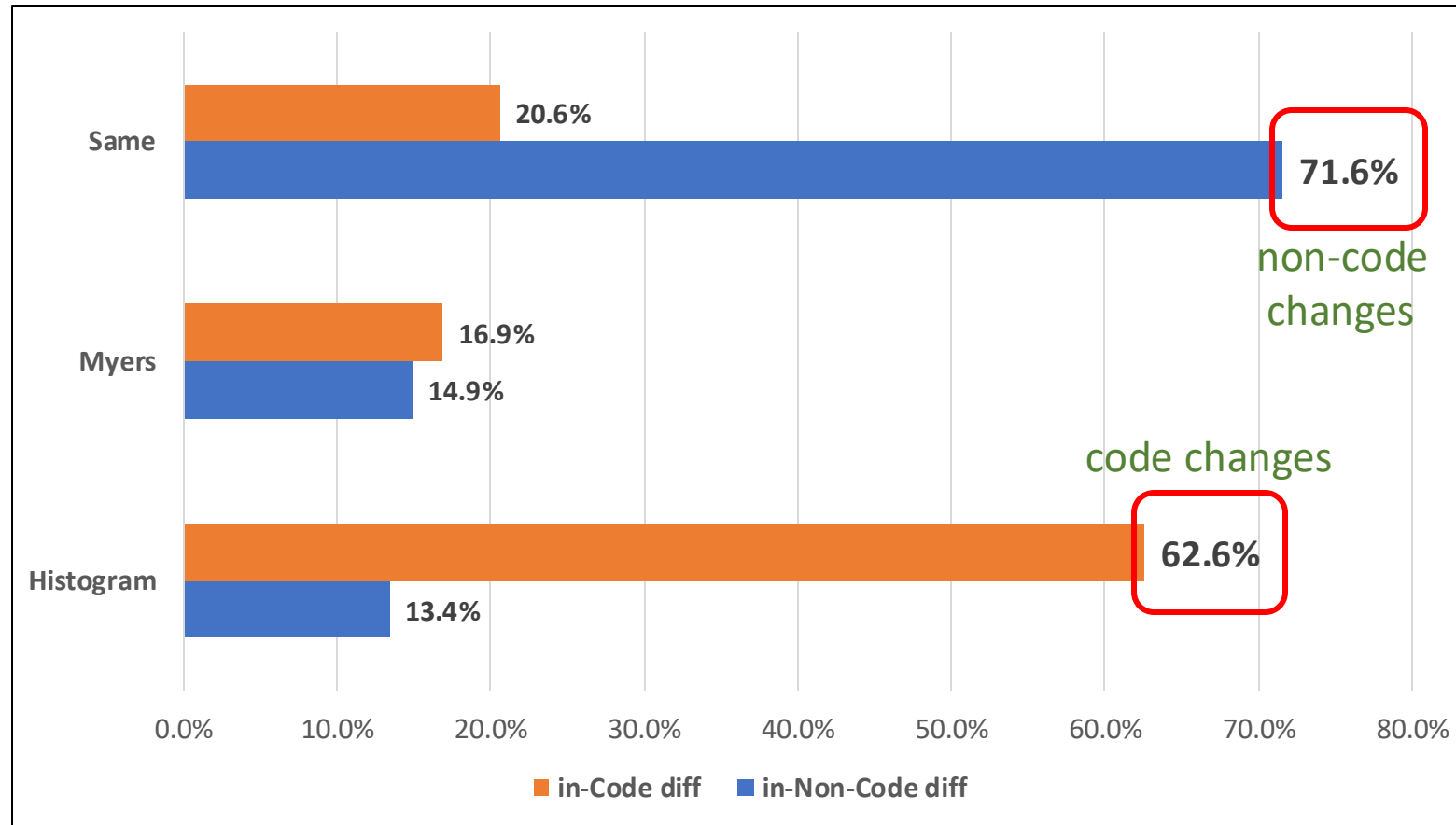
Application #3: manual investigation of patches

Who performed better?

Myers = Histogram

Myers > Histogram

Myers < Histogram



Percentage (%) of occasions



# Use *histogram* diff algorithm when analyzing code changes

- Different diff algorithms can produce different amount and location of changed lines.
- Histogram detects the changed lines more appropriately from source code.



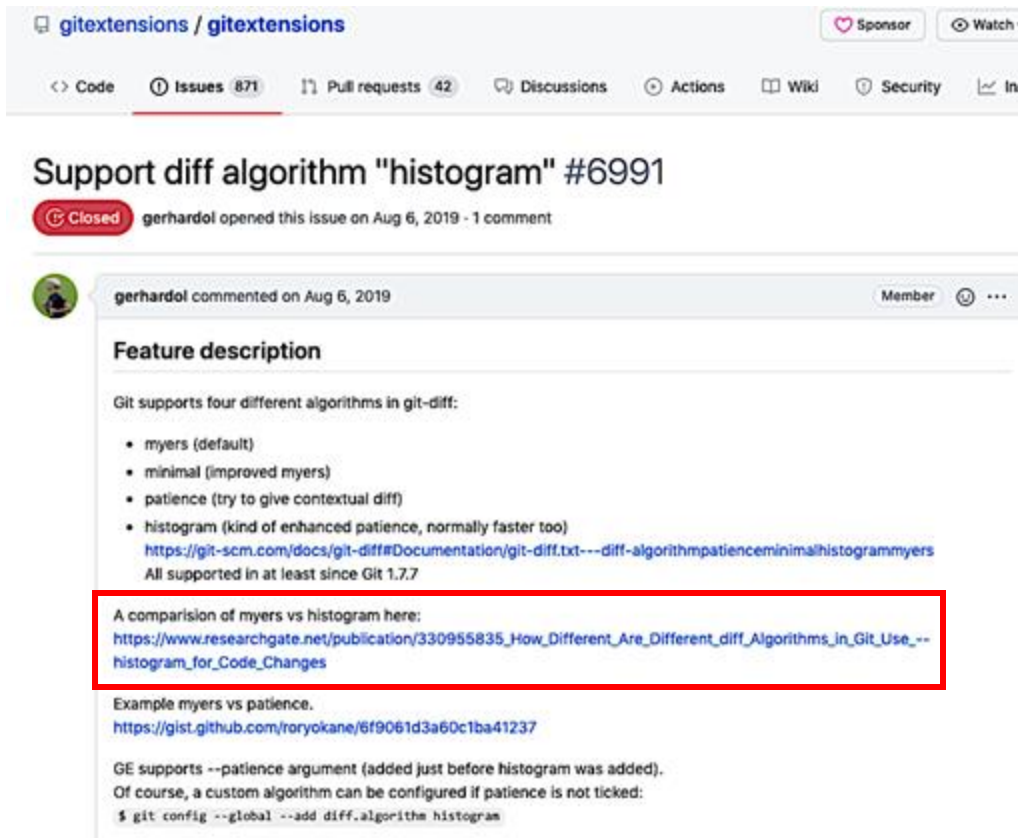
So, let's put Histogram to  
git diff when mining code histories



# Application on actual tools

- Git extension -- (Feature request)

<https://github.com/gitextensions/gitextensions/issues/6991>



The screenshot shows a GitHub issue page for 'Support diff algorithm "histogram" #6991'. The issue is closed and was opened by gerhardol on Aug 6, 2019. A comment from gerhardol, dated Aug 6, 2019, describes the feature request. It states that Git supports four different algorithms in git-diff: myers (default), minimal (improved myers), patience (try to give contextual diff), and histogram (kind of enhanced patience, normally faster too). A link is provided to a researchgate.net publication titled 'How Different Are Different diff Algorithms in Git Use -- histogram\_for\_Code\_Changes'. A red box highlights this link. Below the link, there is an example of myers vs patience and a note that GE supports a --patience argument.

Support diff algorithm "histogram" #6991

Closed gerhardol opened this issue on Aug 6, 2019 - 1 comment

gerhardol commented on Aug 6, 2019

**Feature description**

Git supports four different algorithms in git-diff:

- myers (default)
- minimal (improved myers)
- patience (try to give contextual diff)
- histogram (kind of enhanced patience, normally faster too)

<https://git-scm.com/docs/git-diff#Documentation/git-diff.txt---diff-algorithmpatienceminimalhistogrammyers>

All supported in at least since Git 1.7.7

A comparison of myers vs histogram here:  
[https://www.researchgate.net/publication/330955835\\_How\\_Different\\_Are\\_Different\\_diff\\_Algorithms\\_in\\_Git\\_Use\\_-\\_histogram\\_for\\_Code\\_Changes](https://www.researchgate.net/publication/330955835_How_Different_Are_Different_diff_Algorithms_in_Git_Use_-_histogram_for_Code_Changes)

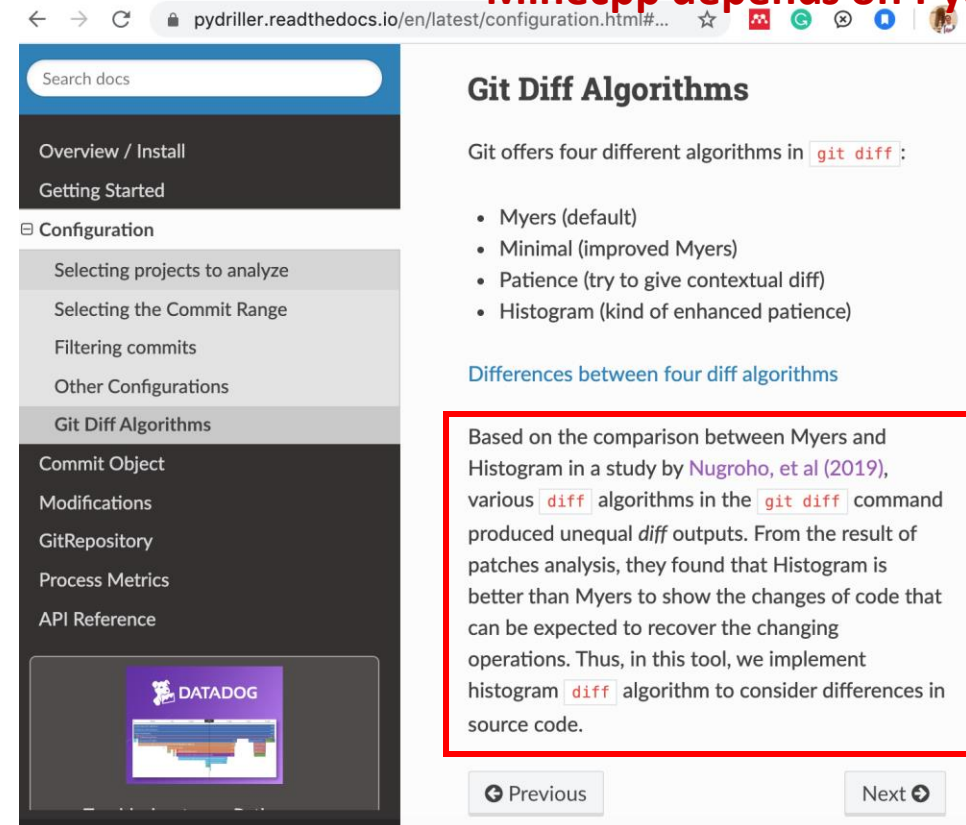
Example myers vs patience.  
<https://gist.github.com/roryokane/6f9061d3a60c1ba41237>

GE supports --patience argument (added just before histogram was added).  
Of course, a custom algorithm can be configured if patience is not ticked:  
\$ git config --global --add diff.algorithm histogram

- Pydriller

<https://pydriller.readthedocs.io/en/latest/configuration.html#git-diff-algorithms>

Minecpp depends on Pydriller



The screenshot shows the Pydriller documentation page for 'Git Diff Algorithms'. It lists four different algorithms in git diff: Myers (default), Minimal (improved Myers), Patience (try to give contextual diff), and Histogram (kind of enhanced patience). A red box highlights a paragraph that discusses a study by Nugroho, et al (2019) comparing Myers and Histogram algorithms. The study found that Histogram is better than Myers to show the changes of code that can be expected to recover the changing operations. Thus, in this tool, they implement histogram diff algorithm to consider differences in source code.

Search docs

Overview / Install  
Getting Started

Configuration

- Selecting projects to analyze
- Selecting the Commit Range
- Filtering commits
- Other Configurations
- Git Diff Algorithms

Commit Object  
Modifications  
GitRepository  
Process Metrics  
API Reference

**Git Diff Algorithms**

Git offers four different algorithms in `git diff`:

- Myers (default)
- Minimal (improved Myers)
- Patience (try to give contextual diff)
- Histogram (kind of enhanced patience)

Differences between four diff algorithms

Based on the comparison between Myers and Histogram in a study by Nugroho, et al (2019), various `diff` algorithms in the `git diff` command produced unequal `diff` outputs. From the result of patches analysis, they found that Histogram is better than Myers to show the changes of code that can be expected to recover the changing operations. Thus, in this tool, we implement histogram `diff` algorithm to consider differences in source code.

Previous Next

# Texts, References, and Acknowledgements

## Online:

- Continuous Integration and Delivery (**CircleCI**: <https://circleci.com>)

## Textbook:

- Sharp, J. (2022). *Microsoft Visual C# Step by Step*, 10th edition, Microsoft Press.
- Watson, K., Nagel, C., Pedersen, J. H., Reid, J. D., & Skinner, M. (2008). *Beginning Microsoft Visual C# 2008*. John Wiley & Sons.
- Mark J. Price (2024). *C# 13 and .NET 9 – Modern Cross-Platform Development Fundamentals*, 9th edition, Packt Publishing Ltd.

## Reference:

- Soni, M. (2016). *DevOps for Web Development*. Packt Publishing Ltd.
- Yusuf Sulistyo Nugroho, Hideaki Hata, and Kenichi Matsumoto. 2020. *How different are different diff algorithms in Git? Use --histogram for code changes*. Empirical Softw. Engg. 25, 1 (Jan 2020), 790–823.