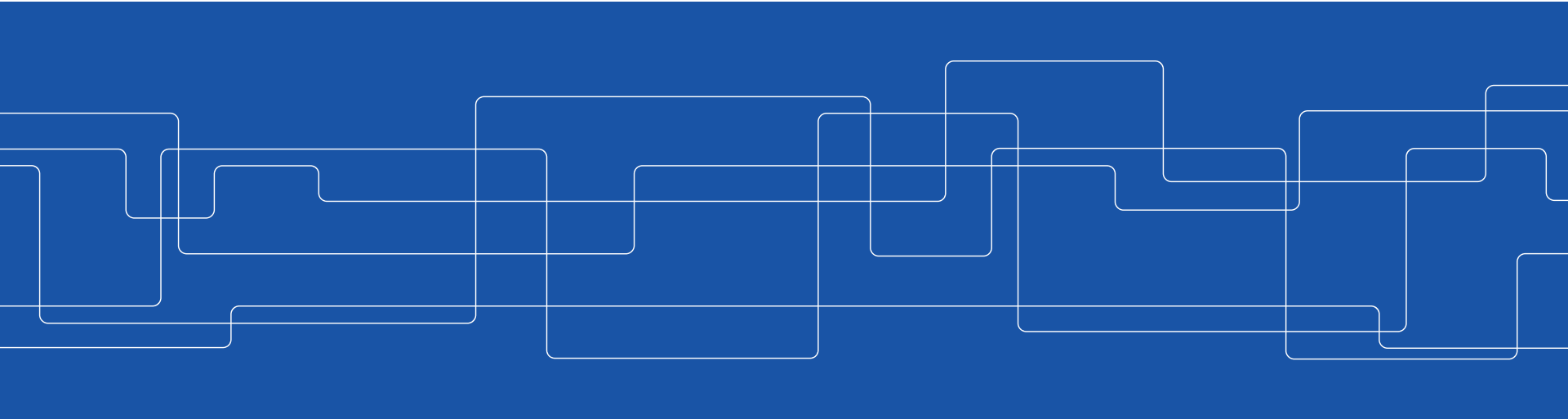
KTH ROYAL INSTITUTE
OF TECHNOLOGY

Augmenting Diffs With Runtime Information

Khashayar Etemadi, Aman Sharma, Fernanda Madeiral, Martin Monperrus

April, 2024



Outline

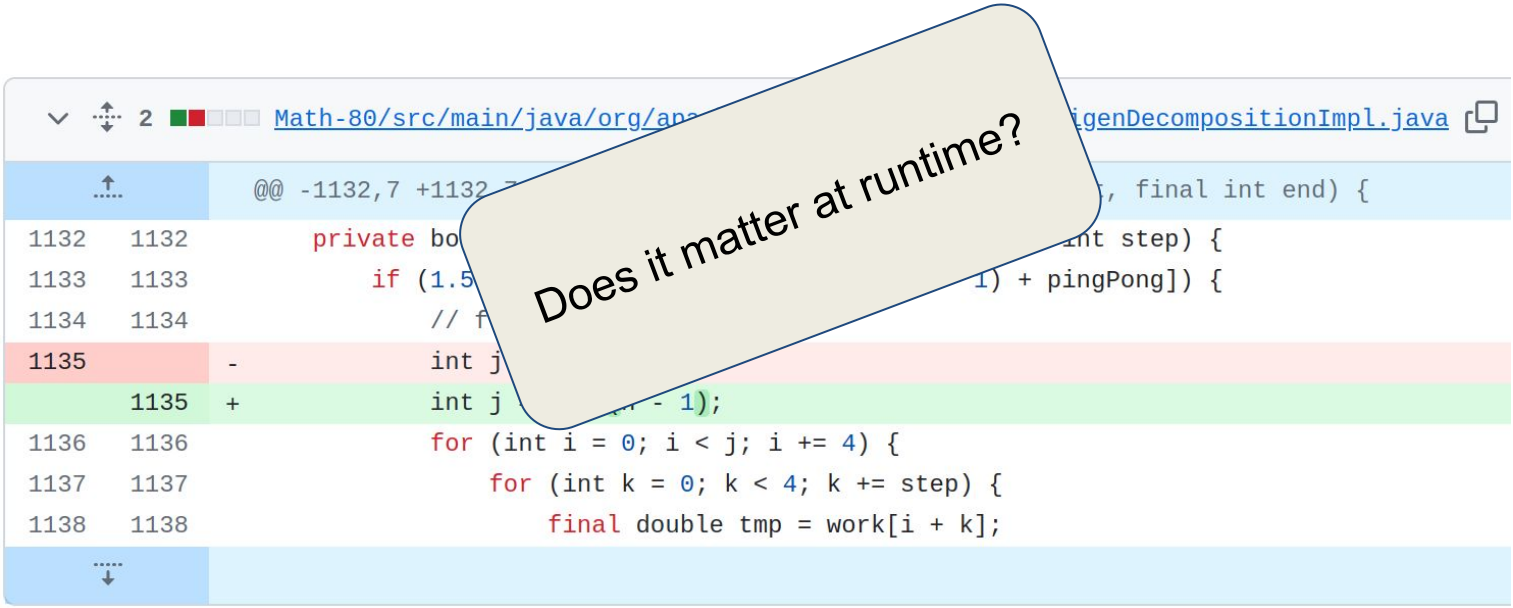
- Code Diff Incompleteness
- Improving Code Diff
- Augmentation by Collector-Sahab
- Technical Details
- Experiments
- Takeaway and Future Work

Code Diff Incompleteness

2 Math-80/src/main/java/org/apache/commons/math/linear/EigenDecompositionImpl.java

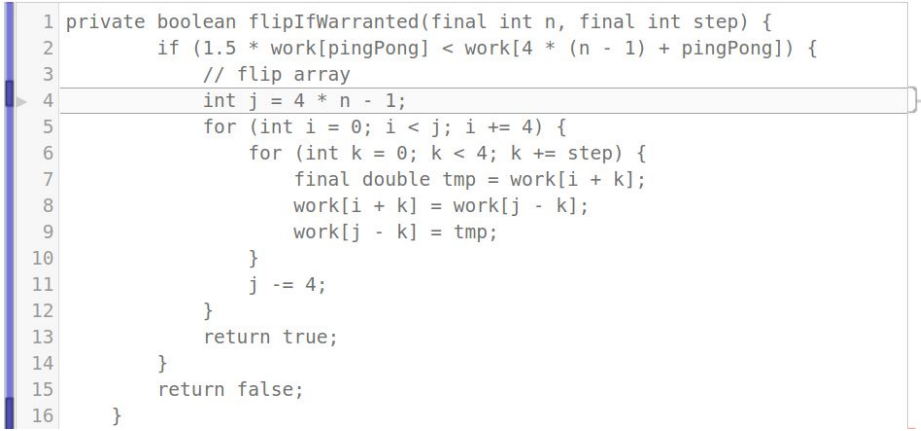
```
@@ -1132,7 +1132,7 @@ private int goodStep(final int start, final int end) {
1132 1132     private boolean flipIfWarranted(final int n, final int step) {
1133 1133         if (1.5 * work[pingPong] < work[4 * (n - 1) + pingPong]) {
1134 1134             // flip array
1135      -           int j = 4 * n - 1;
     1135 +           int j = 4 * (n - 1);
1136 1136             for (int i = 0; i < j; i += 4) {
1137 1137                 for (int k = 0; k < 4; k += step) {
1138 1138                     final double tmp = work[i + k];
```

Code Diff Incompleteness

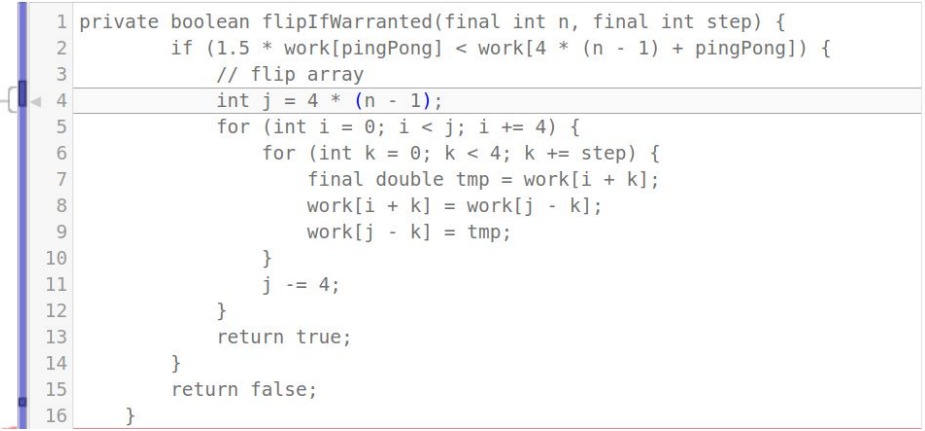


Improving Code Diff

- Mergely [1]
  - Extracts changes at code element level



```
1  private boolean flipIfWarranted(final int n, final int step) {
2          if (1.5 * work[pingPong] < work[4 * (n - 1) + pingPong]) {
3              // flip array
4              int j = 4 * n - 1;
5              for (int i = 0; i < j; i += 4) {
6                  for (int k = 0; k < 4; k += step) {
7                      final double tmp = work[i + k];
8                      work[i + k] = work[j - k];
9                      work[j - k] = tmp;
10                 }
11                 j -= 4;
12             }
13             return true;
14         }
15         return false;
16     }
```



```
1  private boolean flipIfWarranted(final int n, final int step) {
2          if (1.5 * work[pingPong] < work[4 * (n - 1) + pingPong]) {
3              // flip array
4              int j = 4 * (n - 1);
5              for (int i = 0; i < j; i += 4) {
6                  for (int k = 0; k < 4; k += step) {
7                      final double tmp = work[i + k];
8                      work[i + k] = work[j - k];
9                      work[j - k] = tmp;
10                 }
11                 j -= 4;
12             }
13             return true;
14         }
15         return false;
16     }
```

Improving Code Diff

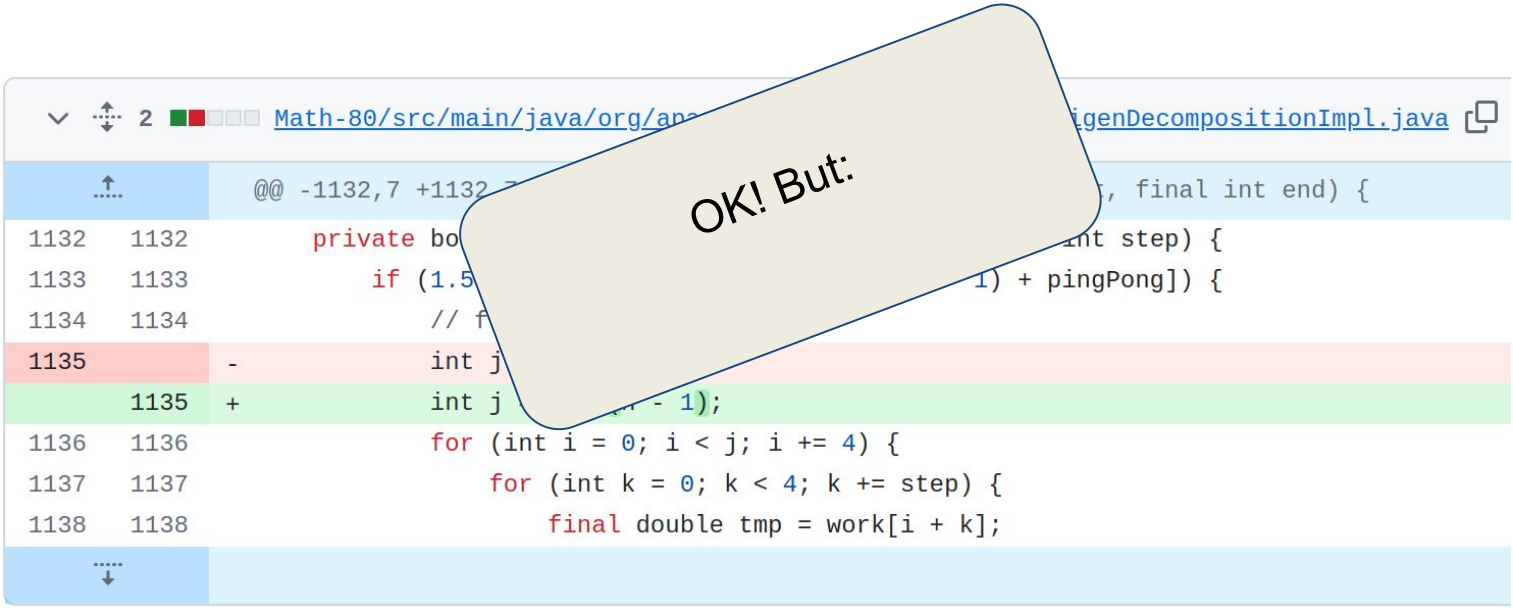
- Mergely [1]
  - Extracts changes at code element level

- GumTree [2]
  - Computes minimum ast modifications

Improving Code Diff

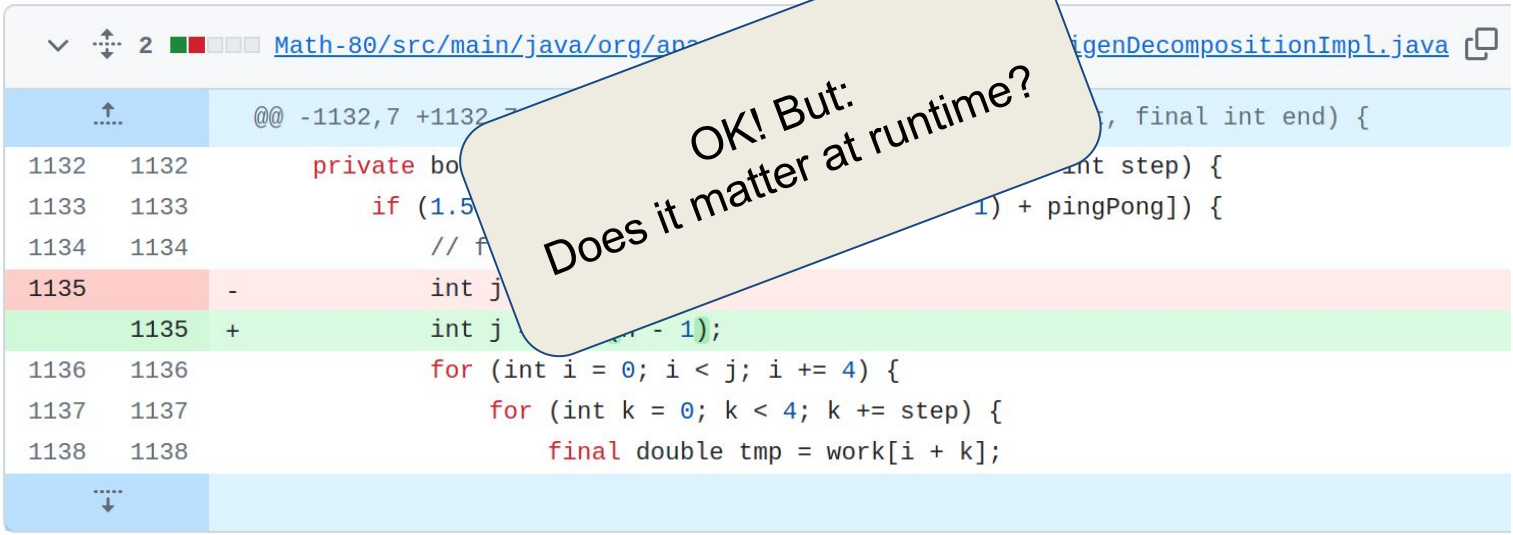
- Mergely [1]
  - Extracts changes at code element level
- GumTree [2]
  - Computes minimum ast modifications
- srcDIFF [3]
  - Produces diff similar to developer’s changes

Augmentation by Collector-Sahab



OK! But:

Augmentation by Collector-Sahab



Augmentation by Collector-Sahab

```
1132  1132        private boolean flipIfWarranted(final int n, final int step) {
1133  1133            if (1.5 * work[pingPong] < work[4 * (n - 1) + pingPong]) {
1134  1134                // flip array
1135        -           int j = 4 * n - 1;
      1135  +           int j = 4 * (n - 1);
1136  1136              for (int i = 0; i < j; i += 4) {
```

COLLECTOR-SAHAB / differentiating test: **EigenDecompositionImplTest**

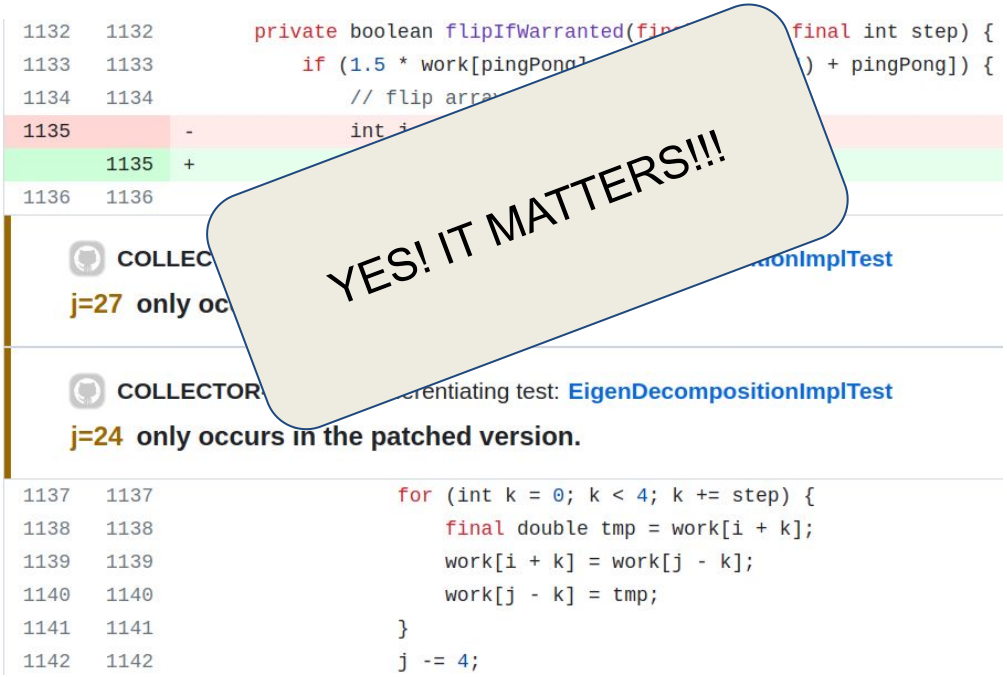
j=27 only occurs in the original version.

COLLECTOR-SAHAB / differentiating test: **EigenDecompositionImplTest**

j=24 only occurs in the patched version.

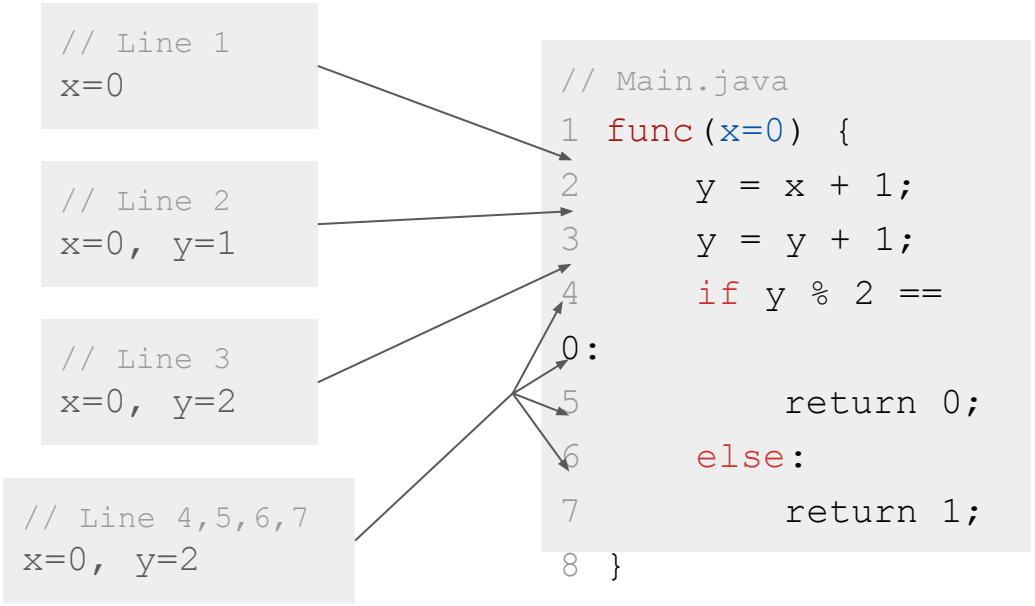
```
1137  1137                  for (int k = 0; k < 4; k += step) {
1138  1138                      final double tmp = work[i + k];
1139  1139                      work[i + k] = work[j - k];
1140  1140                      work[j - k] = tmp;
1141  1141                  }
1142  1142                  j -= 4;
```

Augmentation by Collector-Sahab



Technical Concepts: Program States

Program states are values of visible variables at each line.



Technical Concepts: Workflow

```
// Main.java
1  func(x=0) {
2      y = x + 1;
3 -    y = y + 1;
3 +    y = y + 2;
4      if y % 2 ==
   y = 2 only before
   y = 3 only after

5          return 0;
6      else:
7          return 1;
8  }
```

Instrumentation using ASM

Logged States

```
// Main.java
1  func(x=0) {
2      y = x + 1;
logState()
3 -    y = y + 1;
3 +    y = y + 2;
4      if y % 2 ==
logState()

logState()

6      else:
logState()

8  }
```

```
// Line 2
x=0, y=1
// No diff
```

```
// Line 4
x=0, y=2
x=0, y=3
```


Khashayar Etemadi, khaes@kth.se

Experiments Results

Dataset: 587
Plausible APR Patches
for Defects4J

Experiments Results

Dataset: 587
Plausible APR Patches
for Defects4J

95% (555/587)
Augmented by Collector-Sahab

Experiments Results

Dataset: 587
Plausible APR Patches
for Defects4J

95% (555/587)
Augmented by Collector-Sahab

Users find Augmentations
Useful | Clear | Novel

Takeaways & Future Work

Collector-Sahab effectively detects fine-grained runtime diffs and provides useful augmentations

Takeaway & Future Work

Collector-Sahab effectively detects fine-grained runtime diffs and provides useful augmentations

How can we detect and discard spurious runtime differences?

References

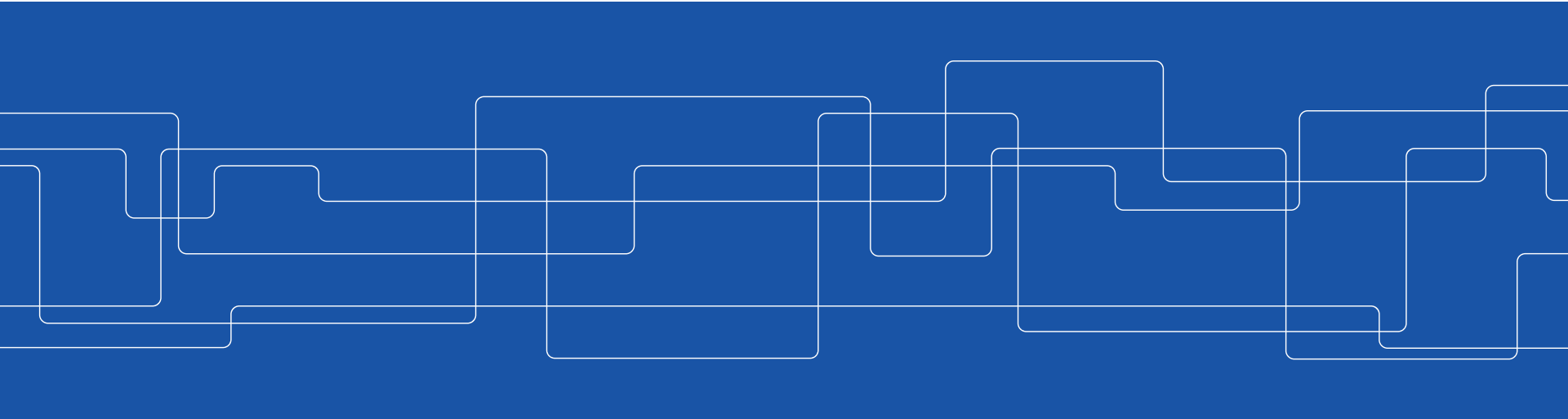
1. Mergely. (2022) Diff text documents online with mergely, an editor and html5 javascript library. [Online]. Available: https://www.mergely.com/
2. Falleri, J. R., Morandat, F., Blanc, X., Martinez, M., & Monperrus, M. (2014, September). Fine-grained and accurate source code differencing. In *Proceedings of the 29th ACM/IEEE international conference on Automated software engineering* (pp. 313-324).
3. Decker, M. J., Collard, M. L., Volkert, L. G., & Maletic, J. I. (2020). srcDiff: A syntactic differencing approach to improve the understandability of deltas. *Journal of Software: Evolution and Process*, *32*(4), e2226.

Thanks for listening!

Khashayar Etemadi

khaes@kth.se

https://github.com/ASSERT-KTH/collector-sahab



Q&A

Etemadi, K., Sharma, A., Madeiral, F., & Monperrus, M. (2023). Augmenting Diffs With Runtime Information. *IEEE Transactions on Software Engineering*.

https://github.com/ASSERT-KTH/collector-sahab