Maintaining Fine-grained Code Metadata Regardless of Moving, Copying and Merging

Christian Prause

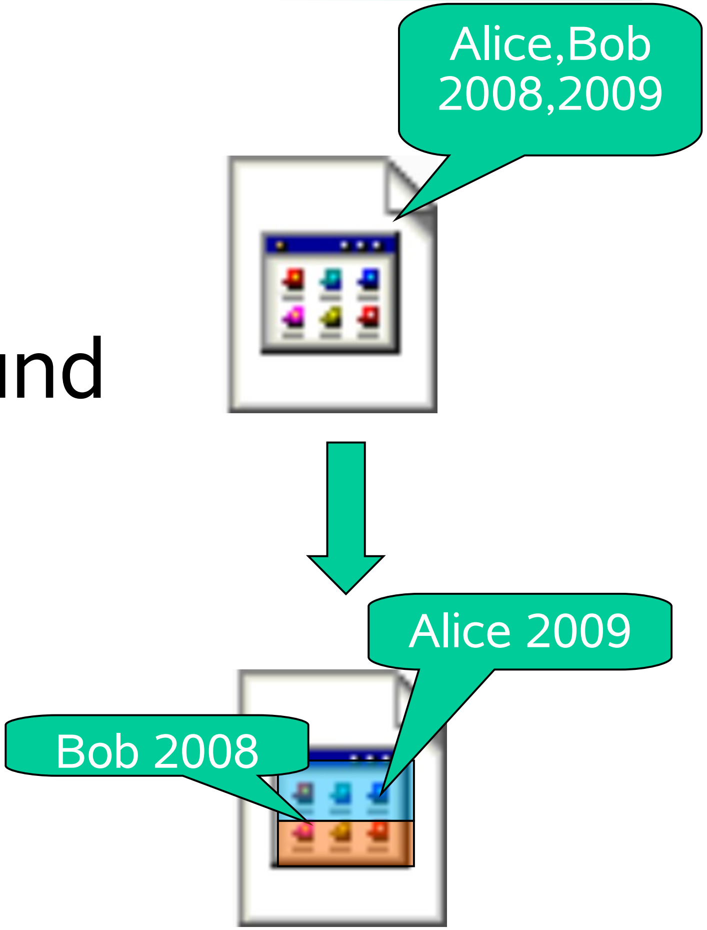
IEEE Working Conference on Source Code Analysis and Manipulation

Edmonton, 2009-09-20

Fraunhofer
FIT

Why fine-grained Metadata?

- File-level metadata coarse
  - e.g. authorship
  - e.g. change background
- Markup
- Advantage: stick to code (not file)



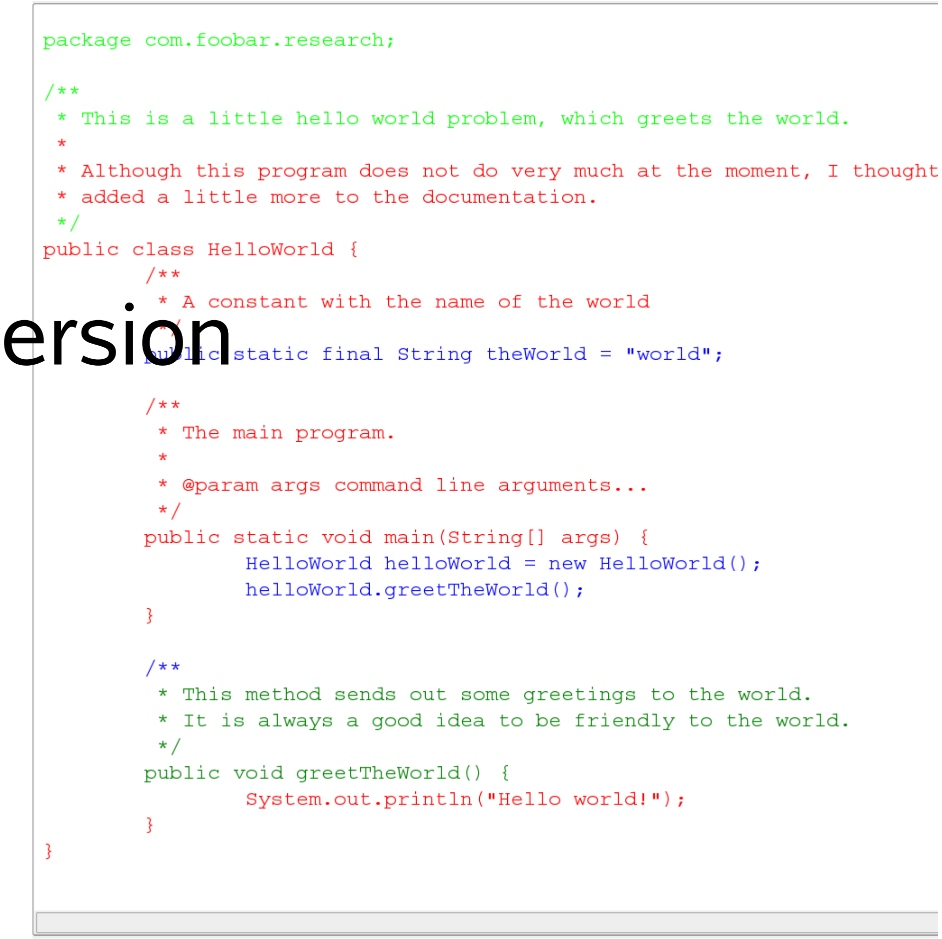
Alice,Bob
2008,2009

Alice 2009

Bob 2008

Example: Code authorship

- svn blame
- Problems:
  - Only previous version
  - Rename
  - Merge
  - Code move
  - Indentation



```
package com.foobar.research;

/**
 * This is a little hello world problem, which greets the world.
 *
 * Although this program does not do very much at the moment, I thought
 * added a little more to the documentation.
 */
public class HelloWorld {
        /**
         * A constant with the name of the world
         */
        public static final String theWorld = "world";

        /**
         * The main program.
         *
         * @param args command line arguments...
         */
        public static void main(String[] args) {
                HelloWorld helloWorld = new HelloWorld();
                helloWorld.greetTheWorld();
        }

        /**
         * This method sends out some greetings to the world.
         * It is always a good idea to be friendly to the world.
         */
        public void greetTheWorld() {
                System.out.println("Hello world!");
        }
}
```

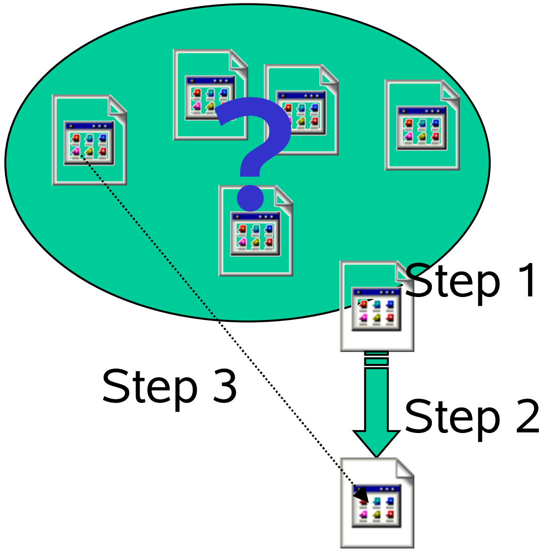
Problem in Practice

- How to maintain?
- Theoretical:
    - Change-aware repository
    - IDE record edit sequence
- Subversion widely used
- Only “snapshots”

Algorithm: Passing on Markup

- Idea: Reconstruct edit from snapshots

- Step 1: Ancestor
- Step 2: Reconstruct
- Step 3: Insertions

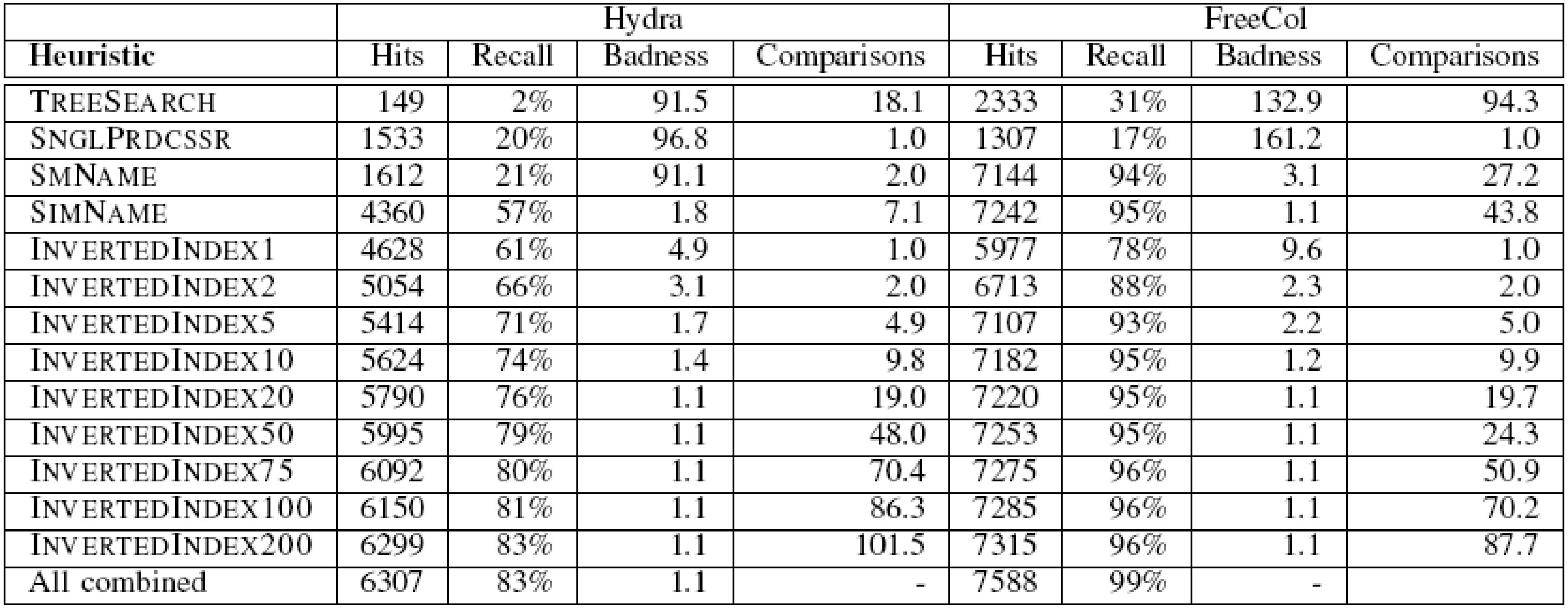


Determining Artifact's Ancestor

Step 1: Ancestor
Step 2: Reconstruct
Step 3: Insertions

- Exhaustive Search
- Heuristics:
    - Previous version
    - Same name
    - Similar name
    - Inverted index
    - Ancestor tree

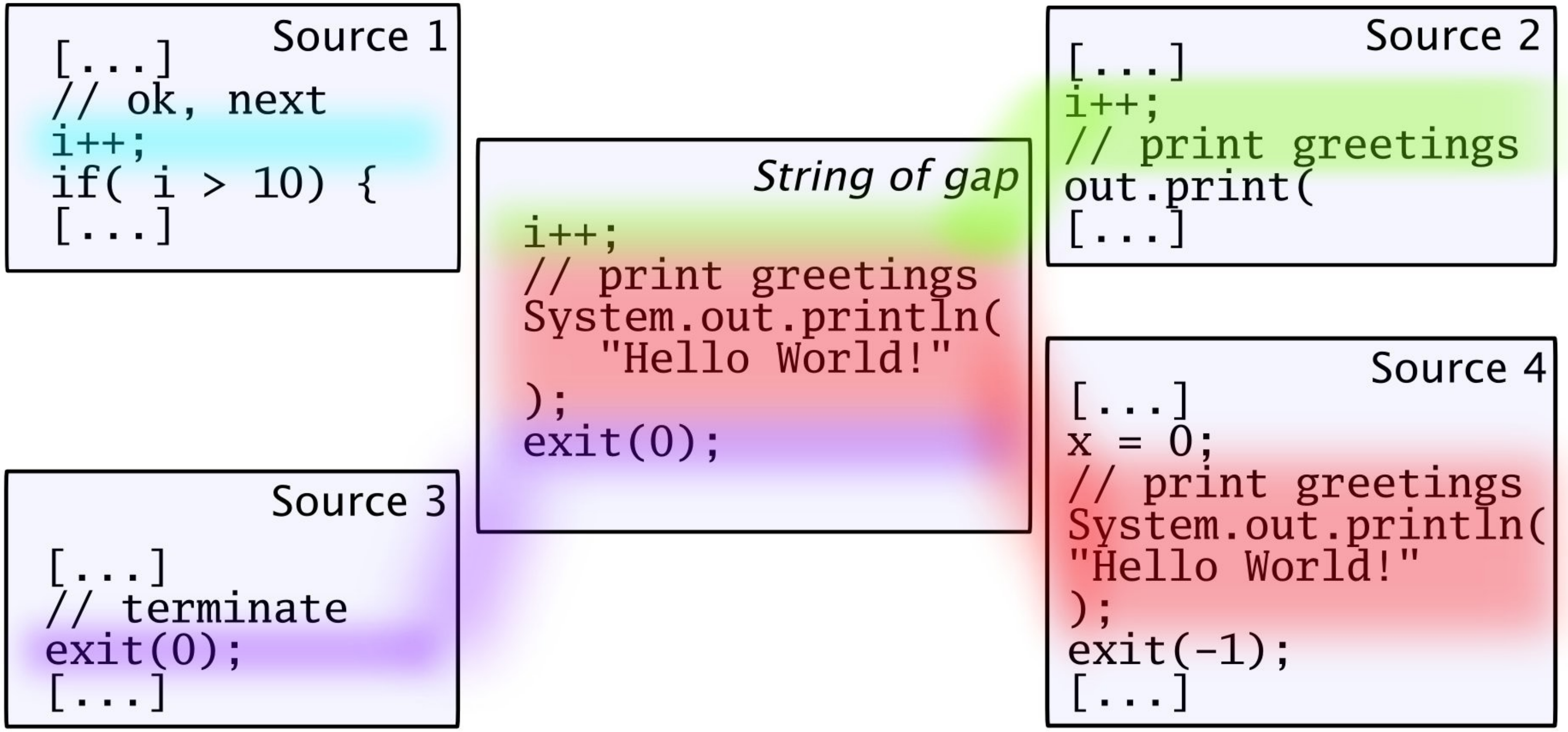
Comparison Ancestor Heuristics



	Hydra				FreeCol			
Heuristic	Hits	Recall	Badness	Comparisons	Hits	Recall	Badness	Comparisons
TreeSearch	149	2%	91.5	18.1	2333	31%	132.9	94.3
SnglPrdcssr	1533	20%	96.8	1.0	1307	17%	161.2	1.0
SmName	1612	21%	91.1	2.0	7144	94%	3.1	27.2
SimName	4360	57%	1.8	7.1	7242	95%	1.1	43.8
InvertedIndex1	4628	61%	4.9	1.0	5977	78%	9.6	1.0
InvertedIndex2	5054	66%	3.1	2.0	6713	88%	2.3	2.0
InvertedIndex5	5414	71%	1.7	4.9	7107	93%	2.2	5.0
InvertedIndex10	5624	74%	1.4	9.8	7182	95%	1.2	9.9
InvertedIndex20	5790	76%	1.1	19.0	7220	95%	1.1	19.7
InvertedIndex50	5995	79%	1.1	48.0	7253	95%	1.1	24.3
InvertedIndex75	6092	80%	1.1	70.4	7275	96%	1.1	50.9
InvertedIndex100	6150	81%	1.1	86.3	7285	96%	1.1	70.2
InvertedIndex200	6299	83%	1.1	101.5	7315	96%	1.1	87.7
All combined	6307	83%	1.1	-	7588	99%	-	

Copy-Origin of Inserted Text

Step 1: Ancestor
Step 2: Reconstruct
Step 3: Insertions



Source 1

```
[...]
// ok, next
i++;
if( i > 10) {
[...]
```

Source 3

```
[...]
// terminate
exit(0);
[...]
```

String of gap

```
i++;
// print greetings
System.out.println(
   "Hello World!"
);
exit(0);
```

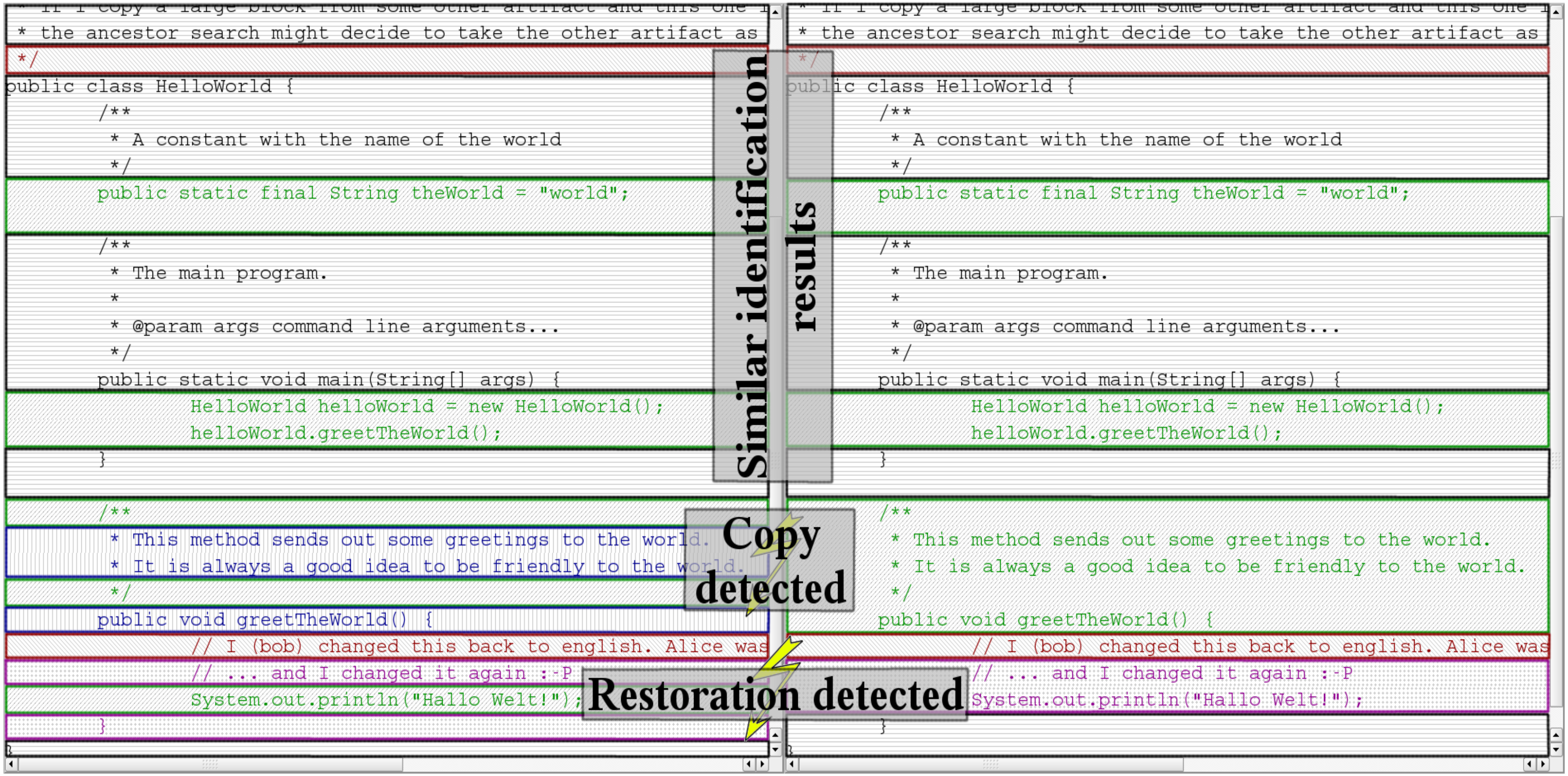
Source 2

```
[...]
i++;
// print greetings
out.print(
[...]
```

Source 4

```
[...]
x = 0;
// print greetings
System.out.println(
"Hello World!"
);
exit(-1);
[...]
```

Example: Author



```
 * If I copy a large block from some other artifact and this one i
 * the ancestor search might decide to take the other artifact as
 */
public class HelloWorld {
        /**
         * A constant with the name of the world
         */
        public static final String theWorld = "world";

        /**
         * The main program.
         *
         * @param args command line arguments...
         */
        public static void main(String[] args) {
                HelloWorld helloWorld = new HelloWorld();
                helloWorld.greetTheWorld();
        }

        /**
         * This method sends out some greetings to the world.
         * It is always a good idea to be friendly to the world.
         */
        public void greetTheWorld() {
                // I (bob) changed this back to english. Alice was
                // ... and I changed it again :-P
                System.out.println("Hallo Welt!");
        }
```

```
 * If I copy a large block from some other artifact and this one i
 * the ancestor search might decide to take the other artifact as
 */
public class HelloWorld {
        /**
         * A constant with the name of the world
         */
        public static final String theWorld = "world";

        /**
         * The main program.
         *
         * @param args command line arguments...
         */
        public static void main(String[] args) {
                HelloWorld helloWorld = new HelloWorld();
                helloWorld.greetTheWorld();
        }

        /**
         * This method sends out some greetings to the world.
         * It is always a good idea to be friendly to the world.
         */
        public void greetTheWorld() {
                // I (bob) changed this back to english. Alice was
                // ... and I changed it again :-P
                System.out.println("Hallo Welt!");
        }
```

Thank you for your attention!

- Summary
  - Why fine-grained metadata
  - Reconstructing edit operations
  - Ancestor search
  - Copy-origin search
- Discussion: Text-level sufficient, syntax-level not necessary