Elmar Jürgens, Florian Deissenboeck, Benjamin Hummel

Clone Detection Beyond Copy&Paste

March 24th
IWSC 2009

Positions

1. Independently developed semantically similar code is unlikely to be representationally similar.

2. Existing clone detection approaches are ill-suited for detecting such similarities.

3. Dynamic clone detection is a promising approach to detect semantically similar yet representationally different code.

Controlled Experiment

Goal: Understand representational similarity of independently developed, semantically similar code.

Setup: 400 Students independently implemented simple Specification.
Performed *aggressive* clone detection using *CloneDetective*:
Full normalization, min length: 5 stmts, edit distance: 1 stmt.

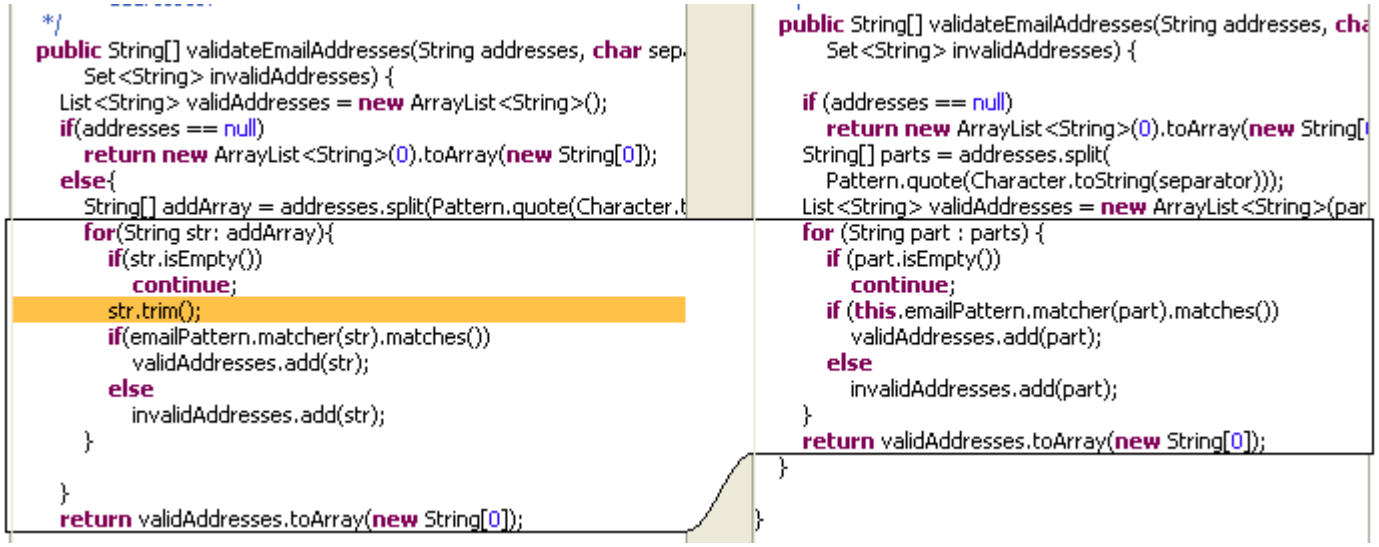
Results: 155 Implementations received, 89 correct and passed unit tests.
Only 154 of the 3916 pairs of impl. detected at least 1 clone.
Substantial differences. (Shortest: 41LoC, Longest: 150LoC).

Only **3,9%** probability to recognize similarity between two fragments!

Due to huge differences, other existing approaches probably not better!

Why CloneDetective?

- Implements scalable detection of gapped clones (=> better recall)
- Pipes an Filters architecture ideally suited for clone detection experiments



```
*/
public String[] validateEmailAddresses(String addresses, char sep
    Set<String> invalidAddresses) {
  List<String> validAddresses = new ArrayList<String>();
  if(addresses == null)
    return new ArrayList<String>(0).toArray(new String[0]);
  else{
    String[] addArray = addresses.split(Pattern.quote(Character.t
    for(String str: addArray){
      if(str.isEmpty())
        continue;
      str.trim();
      if(emailPattern.matcher(str).matches())
        validAddresses.add(str);
      else
        invalidAddresses.add(str);
    }

  }
  return validAddresses.toArray(new String[0]);
```

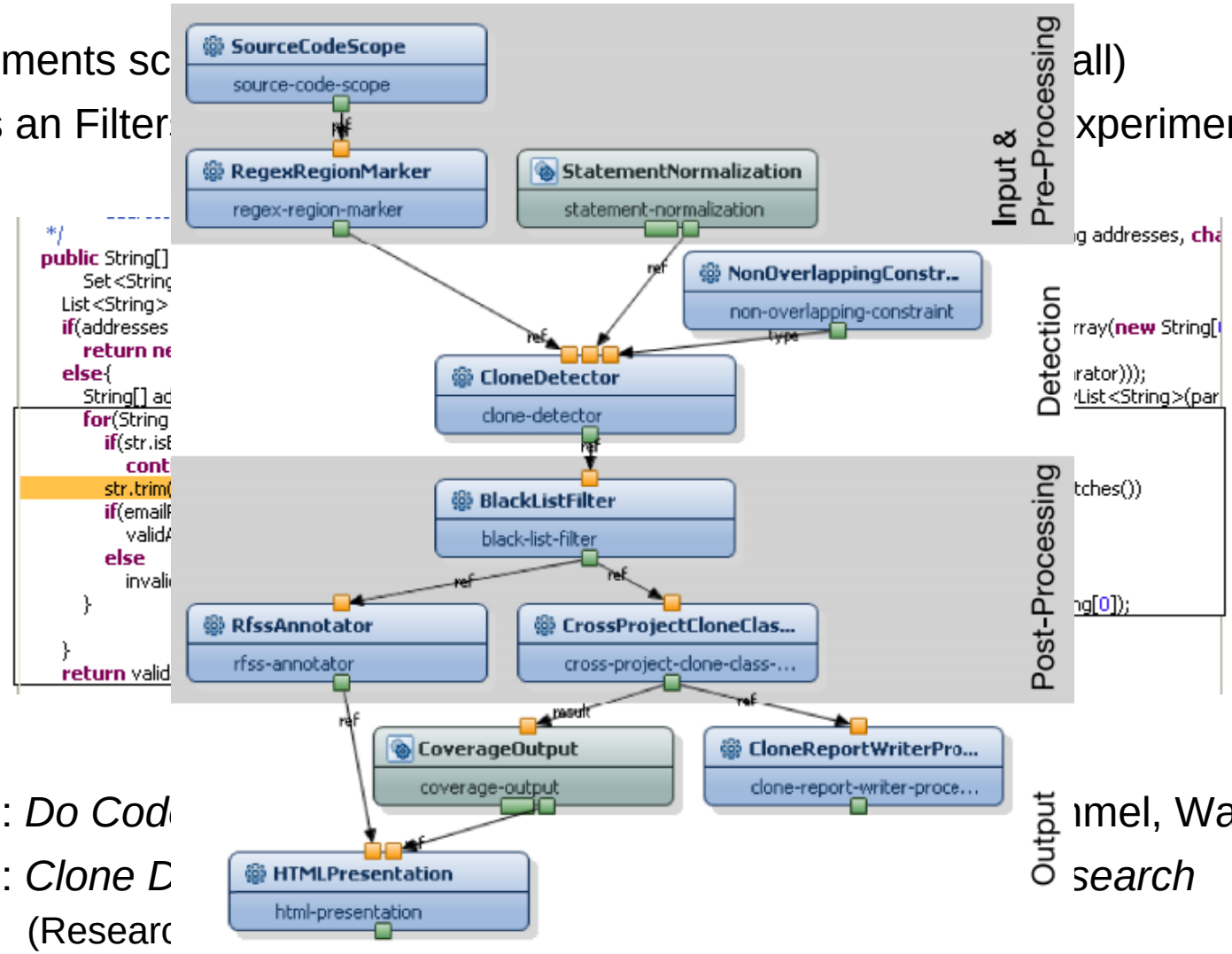
```
public String[] validateEmailAddresses(String addresses, cha
    Set<String> invalidAddresses) {

  if (addresses == null)
    return new ArrayList<String>(0).toArray(new String[
  String[] parts = addresses.split(
    Pattern.quote(Character.toString(separator)));
  List<String> validAddresses = new ArrayList<String>(par
  for (String part : parts) {
    if (part.isEmpty())
      continue;
    if (this.emailPattern.matcher(part).matches())
      validAddresses.add(part);
    else
      invalidAddresses.add(part);
  }
  return validAddresses.toArray(new String[0]);
}
}
```

ICSE ‘09: *Do Code Clones Matter?* Juergens, Deissenboeck, Hummel, Wagner.

ICSE ‘09: *Clone Detective – A Workbench for Clone Detection Research* (Research Demo) Juergens, Deissenboeck, Hummel

Why CloneDetective?

- Implements sc … all)
- Pipes an Filter… xperiments



ICSE '09: *Do Cod…* …mel, Wagner.

ICSE '09: *Clone D…* *…search* (Researc…

Dynamic Clone Detection

Search for **similar behaviour** instead of similar representation

(Undecidable in general, but acceptable approximations in practice?)

```
public static String fillString(int length, char c) {
   char[] characters = new char[length];
   Arrays.fill(characters, c);
   return new String(characters);
}
```

```
private static String padding(int repeat, char padChar) throws ... {
   if (repeat < 0) {
      throw new IndexOutOfBoundsException("..."+repeat); }
   final char[] buf = new char[repeat];
   for (int i = 0; i < buf.length; i++) {
      buf[i] = padChar;}
   return new String(buf);
}
```

Prototype Implementation

- Random-testing-like techniques (execute methods and compare results)
- On basis of CloneDetective, analyzes Java, still various limitations

Open Problems

Which definition of similarity? How to generate adequate test data? …