Clone Overflow: An empirical study of online code clone

Research Questions

- **RQ1 (online code clone):** *How source code are reused between between Q&A sites and open source projects?* We would like to observe whether this phenomenon has happened and at what scale.
- **RQ2 (flow of online code clone):** *what are the directions source code have been reused?* If the code reuse between the two locations exist, we would like to discover the which direction the code that have been copied. Is it from Q&A site to open source projects, or the other way around, or both?
- **RQ3 (classification of online code clone):** *How and why did these online code clone happen?* Can we categorise them?
- **RQ4 (effects of online code clone):** *Is this phenomenon of online code clone harmful and how?* is there any evidence of problems caused by reusing code between Q&A sites and open source projects?

Experimental Setup

Dataset

In our study, we selected Qualitas corpus containing 64 Java open source projects [5]. However, we found that *eclipse* project does not contain source code so we removed it from the dataset. This results in totally 63 projects being analysed. The details of the 63 Qualitas projects are listed in Table IV.

Clone Detectors

We selected two clone detectors for this study: Simian [1] and NiCad [2], [3]. **FIXME: Add more info about clone detection tools in general and more details of these two tools**

Results

The results of running 2 clone detectors: Simian and NiCad, to detect clones between 144,075 Stackoverflow fragments (Java accepted answers) and 63 open-source projects in Qualitas dataset is presented below. There are 2 tools selected: Simian and NiCad. They are configured using two different settings: default settings, and settings from another study [6]. Full Simian's parameter names can be found from the footnote[1].

Manual investigation of Simian's clone report showed that there were problematic 11 fragments. These fragments generate false clone containing array initialisation. Hence, they were removed from the result set before analysis.

Table I: Qualitas-*O* (2013-09-01r): Simian and NiCad clone results

Statistics	Simian Default settings threshold=6 iCharC,iCurlB,iIdC,iMod,iStrC				Simian FSE'13 settings threshold=5 iCurlB,iId,iIdC,iStr,iChar,iSbtNm,bSqBrck				NiCad Default settings threshold=6, UPI=0.30 rename=none,abstract=none				NiCad FSE'13 settings threshold=5, UPI=0.2 rename=blind,abstract=literal			
	Frag.	C_{pairs}	C_{SLOC}	$C_{\%}$	Frag.	C_{pairs}	C_{SLOC}	$C_{\%}$	Frag.	C_{pairs}	C_{SLOC}	$C_{\%}$	Frag.	C_{pairs}	C_{SLOC}	$C_{\%}$
Total	654	24,929	–	–	1,175	16,957,362	–	–	985	105,118	–	–	12,084	113,557,298	–	–
Mean	–	38	7.54	0.27	–	14,444	4.80	0.28	–	107	9.52	0.25	–	9,397	5.21	0.20
Std Dev.	–	87	3.21	0.22	–	281,747	1.22	0.18	–	198	3.07	0.18	–	12,098	1.73	0.16
Max	–	551	49.00	0.94	–	9,599,676	18.00	0.89	–	1,792	39.00	0.80	–	227,077	44.00	0.86
Min	–	1	5.00	0.01	–	1	4.00	0.02	–	1	7.00	0.02	–	1	3.00	0.01
Median	–	3	7.00	0.23	–	22	5.00	0.24	–	15	8.00	0.19	–	6,105	5.00	0.15
Mode	–	1	7.00	0.25	–	1	4.00	0.50	–	1	8.00	0.53	–	1	4.00	0.33

[1]Simian's parameters: iChar = ignoreCharacters, iCurlB = ignoreCurlyBraces, iId = ignoreIdentifiers, iIdC = ignoreIdentifierCase, iMod = ignoreModifiers iStrC = ignoreStringCase, iStr = ignoreStrings, iSbtNm = ignoreSubtypeNames, bSqBrck=balanceSquareBrackets

Table II: Qualitas-*N* (2016-08-05): Simian and NiCad clone results (44 new + 19 old Qualitas projects)

Statistics	Simian Default settings threshold=6 iCharC,iCurlB,iIdC,iMod,iStrC				Simian FSE'13 settings threshold=5 iCurlB,iId,iIdC,iStr,iChar,iSbtNm,bSqBrck				NiCad Default settings threshold=6, UPI=0.30 rename=none,abstract=none				NiCad FSE'13 settings threshold=5, UPI=0.2 rename=blind,abstract=literal			
	Frag.	C_{pairs}	C_{SLOC}	$C_{\%}$	Frag.	C_{pairs}	C_{SLOC}	$C_{\%}$	Frag.	C_{pairs}	C_{SLOC}	$C_{\%}$	Frag.	C_{pairs}	C_{SLOC}	$C_{\%}$
Total	792	61,268	–	–	1,229	25,725,857	–	–	1,141	666,140	–	–	12,400	615,263,639	–	–
Mean	–	78	7.31	0.26	–	20,932	4.86	0.28	–	584	9.49	0.25	–	49,618	5.25	0.20
Std Dev.	–	208	2.70	0.21	–	382,015	1.30	0.19	–	1,488	3.11	0.18	–	62,659	2.80	0.16
Max	–	1,300	49.00	0.94	–	13,202,807	21.00	0.92	–	10,980	39.00	0.84	–	1,319,242	250.00	0.96
Min	–	1	5.00	0.01	–	1	4.00	0.12	–	1	7.00	0.02	–	1	2.00	0.01
Median	–	2	7.00	0.20	–	21	5.00	0.25	–	25	8.00	0.19	–	52,329	5.00	0.16
Mode	–	1	7.00	0.33	–	1	4.00	0.50	–	1	8.00	0.67	–	1	4.00	0.33

Table III: Qualitas-*N* (2016-08-05): Simian and NiCad clone results (only 44 new Qualitas projects)

Statistics	Simian Default settings threshold=6 iCharC,iCurlB,iIdC,iMod,iStrC				Simian FSE'13 settings threshold=5 iCurlB,iId,iIdC,iStr,iChar,iSbtNm,bSqBrck				NiCad Default settings threshold=6, UPI=0.30 rename=none,abstract=none				NiCad FSE'13 settings threshold=5, UPI=0.2 rename=blind,abstract=literal			
	Frag.	C_{pairs}	C_{SLOC}	$C_{\%}$	Frag.	C_{pairs}	C_{SLOC}	$C_{\%}$	Frag.	C_{pairs}	C_{SLOC}	$C_{\%}$	Frag.	C_{pairs}	C_{SLOC}	$C_{\%}$
Total	707	60,536	–	–	1,205	22,797,190	–	–	1,068	648,165	–	–	10,231	573,438,528	–	–
Mean	–	86	7.37	0.26	–	18,919	4.85	0.28	–	607	9.39	0.25	–	47663	5.22	0.20
Std Dev.	–	217	2.22	0.21	–	326,588	1.31	0.19	–	1,499	2.98	1.81	–	58,833	2.77	0.15
Max	–	1,298	32.00	0.94	–	11,127,286	21.00	0.92	–	10,550	36.00	0.84	–	1,246,598	250	0.96
Min	–	1	5.00	0.01	–	1	4.00	0.02	–	1	7	0.02	–	1	2.00	0.01
Median	–	2	7.00	0.21	–	20	5.00	0.25	–	33	8	0.19	–	52,077	5.00	0.15
Mode	–	1	7.00	0.25	–	1	4.00	0.50	–	1	8	0.67	–	1	4.00	0.33

A. Agreement based clone pairs vs. Non-agreement based clone pairs

The agreement-based clone pairs are the ones discovered using Bellon's *good*-match(0.7) and *ok*-match(0.7) criteria as listed in Table VI. Non-agreement based clone pairs are the ones that are solely reported by a single tool. The agreement-based pairs provide higher confident that they are real clones than the non-agreement based ones.

B. Agreement based clone pairs

For agreement-based clone pairs, we use the threshold of 0.7 for both *good*-match and *ok*-match. Visualisation of *good*-match common clone pairs between sets of parameter settings can be seen from Figure 1. There are 1,357 unique *good*-match pairs. The *ok*-match subsumes the *good*-match resulting in totally 10,139 common clone pairs (Figure 2).

However, for NiCad's settings, we found a few renaming and clustering errors which resulted in fewer projects with clone reports. For NiCad default settings (NiCad$_{df}$), 6 projects had clustering failed errors. For NiCad FSE13 settings (NiCad$_{fse13}$), 4 projects had renaming failed errors and 13 projects had clustering failed errors as depicted in Table V. So these projects are also missing from agreed clone pairs. **FIXME: Report the errors to NiCad creator.**

FIXME: Maybe no longer needed? We are interested in discovering reused code in the latest versions of Qualitas projects. So, we downloaded the newest release of each project and found 44 of them having newer updates. Then, we reran the experiment again on these 44 projects. Several projects triggered NiCad problem of clustering and renaming again as listed in Table V. The agreed clone pairs using Bellon's *good*-match(0.7) and *ok*-match(0.7) criteria of this new dataset are also listed in Table VI.

Table V: No. of projects in Qualitas Original (Qualitas-*O*) and New (Qualitas-*N*) successfully analysed by Simian and NiCad

Qualitas-*O*			Qualitas-*N*		
Simian$_{df}$/Simian$_{fse13}$	NiCad$_{df}$	NiCad$_{fse13}$	Simian$_{df}$/Simian$_{fse13}$	NiCad$_{df}$	NiCad$_{fse13}$
63	57	46	44	40	34
	6 clustering failed	*13 clustering failed*		*4 clustering failed*	*6 clustering failed*
	cayenne	cayenne		jboss (wildfly)	ArgoUML
	checkstyle	checkstyle		hadoop	checkstyle
	db-derby	db-derby		db-derby	db-derby
	geotools	geotools		hibernate	cayenne
	iReport	iReport			jena
	hibernate	ArgoUML			geotools
		castor			
		drjava			
		ganttproject			
		ivatagroupware			
		jasperreports			
		jboss			
		jchempaint			
		4 renaming failed			*4 renaming failed*
		Vuze			Vuze
		aspectj			hadoop
		eXoPortal			jboss (wildfly)
		hibernate			hibernate

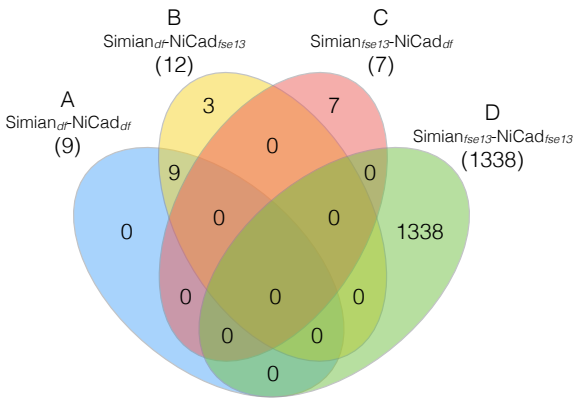

Fig. 1: Qualitas-*O* *good*-match(0.7) pairs

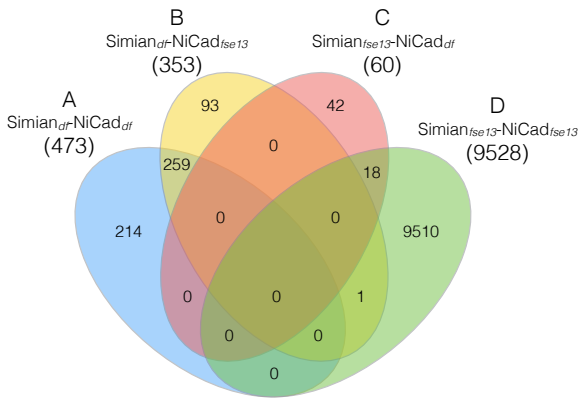

Fig. 2: Qualitas-*O* *ok*-match(0.7) pairs

Table VI: Distribution of agreement-based clone pairs reported using Bellon's criteria

Tool		Qualitas-*O*		Qualitas-*N*	
Simian	NiCad	*good*-match	*ok*-match	*good*-match	*ok*-match
default	*default*	9	473	0	6,604
default	*fse13*	12	353	0	79
fse13	*default*	7	60	0	23
fse13	*fse13*	1,338	9528	1,170	6,690
Total		1,366	10,414	1,170	13,396
Total (unique)		1,357	10,139	1,170	13,386

C. Manual investigation of agreement-based clone pairs

Table VII: Classifications of clone creation

Category	Descriptions
A	Code in Stackoverflow is copied from Qualitas (Q $\rightarrow$ S).
A'	Code in Qualitas is copied from Stackoverflow (S $\rightarrow$ Q).
B	Code is copied either from each other or a third source (unknown) (S $\leftrightarrow$ Q $\vee$ (T $\rightarrow$ S $\wedge$ T $\rightarrow$ Q)).
C	Code in both places are copied from a third source T (known) (T $\rightarrow$ S $\wedge$ T $\rightarrow$ Q).
D	Code is a boiler-plate or IDE auto-generated.
E	Code in both places initialise a similar/the same object; extend the same class/its subclass; implement the same interface.
F	Accidental similarity, false clone

The classification scheme is described in Table VII and the classification results are shown in Table VIII. We have manually investigated all of the 1,357 *good-match* ones reported by agreement of four different Simian and NiCad settings. However, for the *ok-match*, we could not investigate all of the 10,139 pairs manually. According to the distribution of category from *good-match* results, we can see that Simian$_{fse13}$–NiCad$_{fse13}$ produces a large number, 1,338, of false positive results (D, E, and F). Thus, we decided to leave them out of the manual investigation of *ok-match* pairs. There are totally 608 *ok-match* pairs that were investigated. The 39 true positive pairs found are combinations of 8 unique Stackoverflow fragments, and 9 unique Qualitas Java files from 6 different projects.

Since we are not certain about the direction of copying in the B-classified pairs, we checked the modification time of each Java file in Qualitas project and compare it to the timestamp of Stackoverflow answers. We found that all Stackoverflow code fragments were posted after their respectively similar Java files in Qualitas project. This means that the copying can only be either (1) Q $\rightarrow$ S or (2) from a third source to both S and Q independently.

Table VIII: Qualitas-*O*: Classification results of *good-* and *ok*-match pairs which excludes the subsumed *good*-match and Simian$_{fse13}$-NiCad$_{fse13}$ pairs.

Classificaiton	A	A'	B	C	**Sum**	S_u	Q_u	Q_{up}	D	E	F	**Sum**	S_u	Q_u	Q_{up}	**Total**	S_u	Q_u	Q_{up}
good-match(0.7)	1	0	1	3	**5**	5	4	4	26	6	1320	**1352**	56	402	31	**1357**	61	406	32
ok-match(0.7)	8	0	23	8	**39**	8	9	6	480	28	61	**569**	76	60	16	**608**	83	68	19

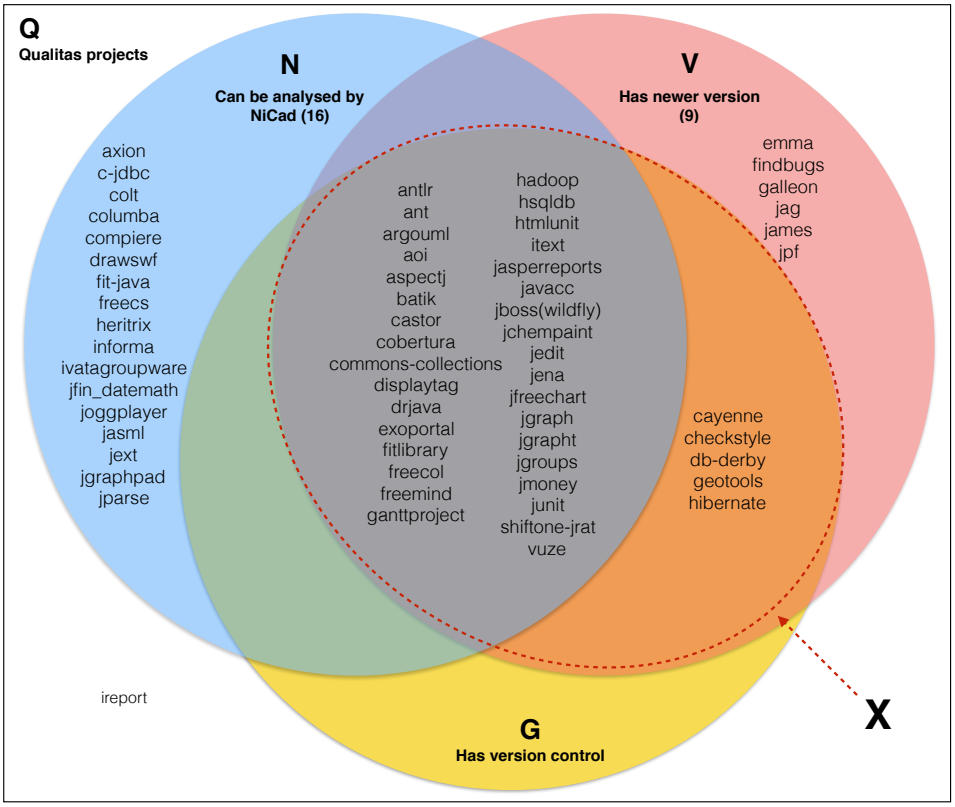

Fig. 3: Qualitas projects categorised by NiCad's results, existence of newer versions, and version control

Table IX: Qualitas-*O*: Distribution of classification category A–F according to *good*-match pairs

Category	A	A'	B	C	D	E	F	Total
Simian$_{df}$–NiCad$_{df}$	1	0	1	3	0	4	0	9
Simian$_{df}$–NiCad$_{fse13}$	1	0	1	3	1	5	1	12
Simian$_{fse13}$–NiCad$_{df}$	0	0	0	0	7	0	0	7
Simian$_{fse13}$–NiCad$_{fse13}$	0	0	0	0	18	1	1,319	1,338
Total	2	0	2	6	26	10	1,320	1,366
Total (unique)	1	0	1	3	26	6	1,320	1,352

Table X: Qualitas-*O*: Distribution of classification category A–F according to the *ok*-match pairs

Category	A	A'	B	C	D	E	F	Total
Simian$_{df}$–NiCad$_{df}$	3	0	10	6	433	5	7	464
Simian$_{df}$–NiCad$_{fse13}$	8	0	22	4	250	25	32	341
Simian$_{fse13}$–NiCad$_{df}$	0	0	0	0	29	0	24	53
Total	11	0	32	10	712	30	63	858
Total (unique)	8	0	23	8	480	28	61	608

D. Non-agreement based clone pairs

In the preliminary stage of our experiment, we found that there are 41 Stackoverflow fragments reported by Simian with default configurations. However, only 10 of them appear in the new results using tool's agreement. Thus, we further investigated the clone pairs reported by Simian and NiCad but *without* an agreement.

With our 4 settings, we decided to investigate only 2 settings, Simian$_{df}$, and NiCad$_{df}$, and drop Simian$_{fse13}$ and NiCad$_{fse13}$ due to their large number of false positives as shown in Table IX and X. With the 2 selected settings, we investigated clone pairs having the minimum clone size of 10 SLOC as they are meaningful and tend to be real clone in modern clone detection [4].

For Simian$_{df}$, there were 9,383 clone pairs reported by the tool. Out of 9,383 pairs, 140 of them are the ones found in *ok*-pairs using agreement-based detection. We filtered the results further by removing false positives such as similar equals(), hashCode() methods, getters and setters out by using regular expression. We managed to remove 8,956 pairs using this method. Eventually, there were 287 clone pairs remaining for manual investigation. For NiCad$_{df}$, we obtained 7,040 clone pairs to look

at which is infeasible for manual investigation. Hence, result filtering was also needed. However, regular expressions could not be used effectively as in Simian's case since NiCad allowed clones that are different at keywords/variable names or even added/deleted lines. So we decided to filter the results by selecting pairs that pass stricter clone criteria with $\mathrm{UPI} = 0.2$. By reducing the UPI to 0.2, there were totally 166 pairs left. Out of 166, 52 are *ok*-pairs and 114 are remaining pairs for manual check (18 pairs are from *cayenne* and *iReport* that could not be analysed using UPI = 0.3). The statistics of the clones and classification results are reported in Table XI and XII.

Table XI: Statistics of Simian$_{df}$ and NiCad$_{df}$ clone pairs.

Tool	Clone pairs	*ok*-pairs	filtered pairs	remaining pairs
Simian$_{df}$	9383	140	8956	287
NiCad$_{df}$	7040	226	6700	114

E. Manual investigation of non-agreement based clone pairs

We performed manual investigation of the clone pairs reported by Simian$_{df}$ and NiCad$_{df}$ in the same way as the agreement-based clone pairs. The results of the manual investigation is reported in Table XII.

Table XII: Classification results of 292 Simian$_{df}$ and 114 NiCad$_{df}$ individual unique pairs.

Tool/Classification	A	A'	B	C	**Sum**	S_u	Q_u	Q_{up}	D	E	F	**Sum**	S_u	Q_u	Q_{up}	**Total**	S_u	Q_u	Q_{up}
Simian$_{df}$	35	0	89	7	**133**	68	57	23	13	10	133	**159**	39	69	23	**287**	103	121	31
NiCad$_{df}$	4	0	5	0	**9**	9	5	4	24	3	78	**105**	41	39	12	**114**	48	44	14

F. Summary of true online clone pairs

Table XIII: Numbers of true online clone pairs (A+A'+B+C) found by manual investigation

Tool	A	A'	B	C	Total
good-pairs	1	0	1	3	5
ok-pairs	8	0	23	8	39
Simian$_{df}$ pairs	35	0	89	7	131
NiCad$_{df}$ pairs	4	0	5	0	9
Total	48	0	118	18	184

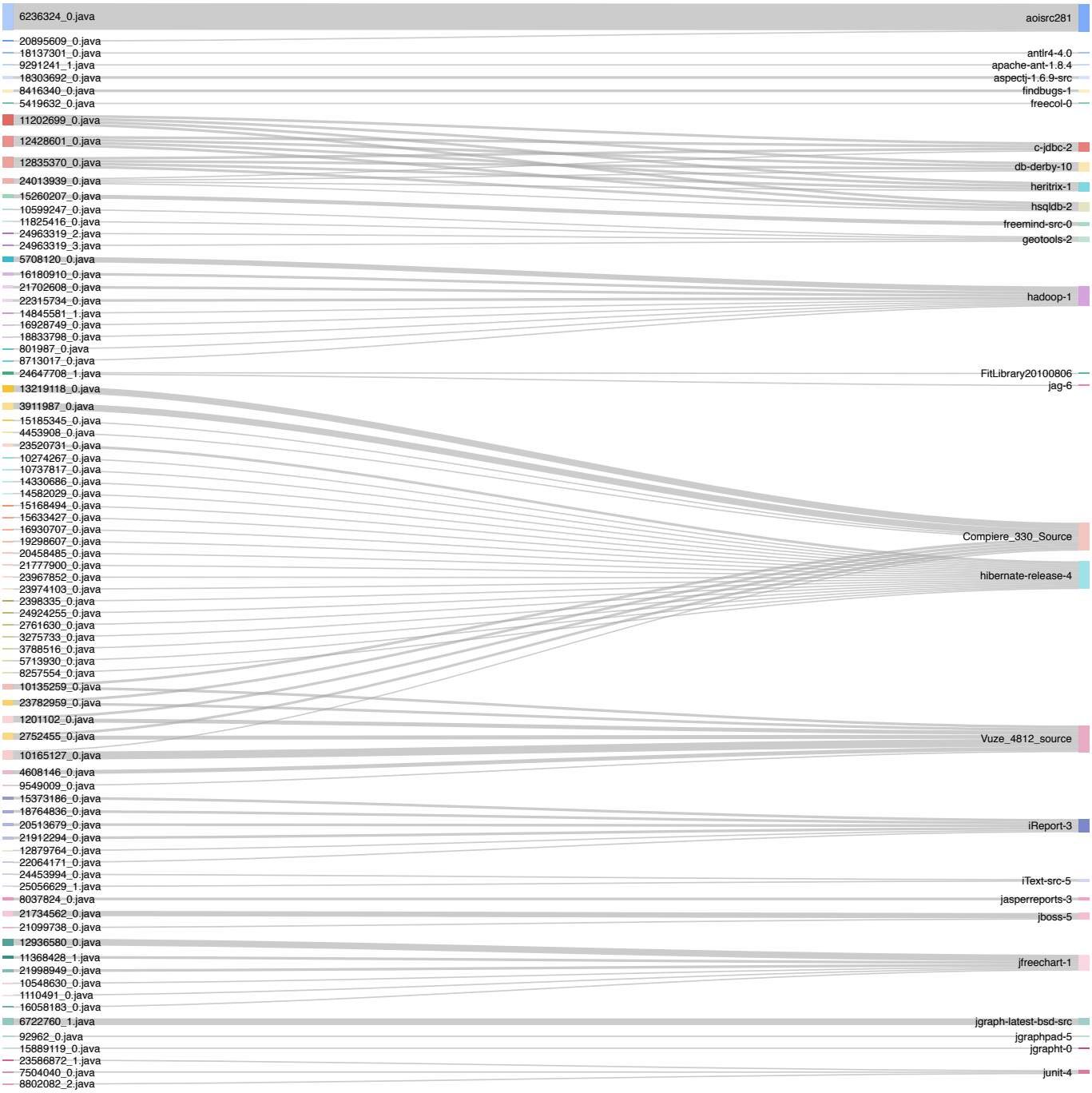

Fig. 4: Relationships between 184 true online code clone found between Stackoverflow and Qualitas projects

G. Effects of online code clone

Issue 1: outdated online code

After located and classified all the online clone pairs, we are now interested in analysing the true online clone pairs that are copied in the direction of Q $\rightarrow$ S (class-*A* online clone pairs) in Table XIII which results in 48 pairs selected. We restricted it further to only the ones having versioning system so we can trace changes made to these clone pairs. Fortunately, all of the pairs were from projects with either git or svn so we did not remove any pair from this set.

Our intuition behind his issue comes from a situation where a piece of code has been copied from a Qualitas project and posted on Stackoverflow. Later, that piece of code has been modified further to accommodate changes in the projects (or any other reasons). However, even the code in Qualitas project is already updated, the one posted on Stackoverflow is not changed. This results in outdated online code which can cause problems when programmers copy and use it in their projects.

Issue 2: software licensing violation

Table XIV: Projects that do not have resutls in Qualitas-*N* but do have results in Qualitas-*O*

Project	Qualitas-*O*				Qualitas-*N*
	good-match	TP	*ok*-match	TP	Reason of missing results
jboss	2	0	63	1	NiCad's failure
db-derby	0	0	0	0	NiCad's failure
hadoop	191	1	2432	5	NiCad's failure
hibernate	19	0	11	0	NiCad's failure
ArgoUML	0	0	0	0	NiCad's failure
checkstyle	0	0	0	0	NiCad's failure
cayenne	0	0	0	0	NiCad's failure
geotools	0	0	0	0	NiCad's failure
jena	0	0	0	0	NiCad's failure
Vuze	0	0	0	0	NiCad's failure
Compiere	127	2	178	8	No update
axion	7	0	12	0	No update
c-jdbc	66	0	768	0	No update
colt	0	0	5	0	No update
columba	14	0	177	0	No update
drawswf	0	0	0	0	No update
fit-java	0	0	2	0	No update
freecs	5	0	22	0	No update
heritrix	16	0	190	0	No update
iReport	0	0	0	0	No update
informa	31	0	22	0	No update
ivatagroupware	0	0	0	0	No update
jFin	15	0	72	0	No update
jOggPlayer	12	0	52	0	No update
jasml	23	0	47	0	No update
jext	80	0	85	0	No update
jgraphpad	1	1	55	0	No update
jmoney	0	0	0	0	No update
jparse	0	0	0	0	No update
freecol	1	1	0	0	*FreeColMenuTest.java* exists with the *createImage-Icon()* method. Still can't find the reason why it is not reported.
aoisrc281	0	0	33	33	*MovieEncoder.java* (in 2 locations) are missing from the latest version.
findbugs	0	0	2	2	*UnionBugs.java* has been changed.
jgraph	0	0	3	3	*HelloWorld.java* has been changed.
Total	610	5	4231	52	

INVESTIGATION OF MISSING A/B CLONE PAIRS REPORTED BY SIMIAN (DF)

We investigated the 41 clone pairs previously reported by Simian with default configurations and manually investigated. The 41 pairs were searched for in 4 new results sets: Simian$_{df}$, Simian$_{fse13}$, NiCad$_{df}$, NiCad$_{fse13}$. The investigation results are shown in Table XV.

Table XV: Results of matching the original 41 Simian(default) pairs in the pretty-printed result sets

Settings	Found	Not found
Simian$_{df}$	40	1*
Simian$_{fse13}$	0	41
NiCad$_{df}$	17	24
NiCad$_{fse13}$	24	17

The single missing Stackoverflow fragment (19051537_0.java) (denoted by *) is one of the 11 false clones generated by Simian. It is removed from the results of the pretty-printed version because it is an outlier. The rest are missing because of different parameter settings.

SIMIAN'S PARAMETERS

We have carefully investigated the effects of the Simian's parameter `-balanceSquareBrackets+`. I found that it works in the expected way of handling a pair of brackets (`[,]`) that span over multiple lines. For example, the two code fragments in Figure 5 would match by having `-balanceSquareBrackets+` enabled.

```
1          public class MagicSquare {                          public class MagicSquare2 {
2              private int[][] square;                             private int[
3              private boolean[] possible;                                    ][
4              private int totalSqs;                                          ] square;
5              private int sum;                                    private boolean[] possible;
6              private int numsquares;                             private int totalSqs;
7              public static void main ( String[] args ) {         private int sum;
8              MagicSquare m = new MagicSquare ( 3 );              private int numsquares;
9                                                                  public static void main ( String[] args ) {
10                                                                 MagicSquare m = new MagicSquare ( 3 );
```

Fig. 5: Two identical fragments with only differences in locations of the square brackets. All 7 lines are reported by Simian if `-balanceSquareBrackets+` is enabled. If not, the clone pairs is reported as (MagicSquare.java [3,8], MagicSquare2.java [5,10]).

However, the `-balanceSquareBrackets+` parameter only works on a small testing environment having toy programs or only small pairs from the full datasets. It does not work with the full complete set of 144,075 Stackoverflow fragments and Qualitas projects. Please find the summary of all the testing scenarios in Table XVI.

Table XVI: Simian's `-balanceSquareBrackets+` (`-bsb+`) is observed to have unpredictable behaviours when running against big datasets. CP_2 means the reported clone(s) do not contain lines having dislocated brackets (L_b) (i.e. $CP_2 = CP_1 - L_b$).

Project 1	**Project 2**	**Dislocated brackets?**	**-bsb+**	**Clones pair reported**
Only run Simian against the pair				
MagicSquare.java	MagicSquare_exact_copy.java	no	0,1	CP_1
MagicSquare.java	MagicSquare2.java	yes	0	CP_2
MagicSquare.java	MagicSquare2.java	yes	1	CP_1
stackoverflow/4298836_0.java	Qualitas/aoisrc281/../ExprModule.java	no	0	CP_3
stackoverflow/4298836_0.java	Qualitas/aoisrc281/../ExprModule.java	no	1	CP_3
stackoverflow/4533682_1.java	Qualitas/cobertura-1/../TouchCollector.java	no	0	CP_4
stackoverflow/4533682_1.java	Qualitas/cobertura-1/../TouchCollector.java	no	1	CP_4
Run Simian against the complete stackoverflow data and the project				
stackoverflow/4298836_0.java	Qualitas/aoisrc281/../ExprModule.java	no	0	CP_3
stackoverflow/4298836_0.java	Qualitas/aoisrc281/../ExprModule.java	no	1	–
stackoverflow/4533682_1.java	Qualitas/cobertura-1/../TouchCollector.java	no	0	CP_4
stackoverflow/4533682_1.java	Qualitas/cobertura-1/../TouchCollector.java	no	1	–

REFERENCES

[1] Simian. http://www.harukizaemon.com/simian. Accessed: 07.04.2016.

[2] James R. Cordy and Chanchal K. Roy. The NiCad Clone Detector. In *ICPC '11 Proceedings of the 2011 IEEE 19th International Conference on Program Comprehension*, pages 3–4, 2008.

[3] Chanchal K. Roy and James R. Cordy. NICAD: Accurate Detection of Near-Miss Intentional Clones Using Flexible Pretty-Printing and Code Normalization. In *2008 16th IEEE International Conference on Program Comprehension*, pages 172–181, 2008.

[4] Hitesh Sajnani, Vaibhav Saini, Jeffrey Svajlenko, Chanchal K Roy, and Cristina V Lopes. SourcererCC: Scaling Code Clone Detection to Big-Code. In *Proceedings of the 38th International Conference on Software Engineering - ICSE '16*, pages 1157–1168, 2016.

[5] Ewan Tempero, Craig Anslow, Jens Dietrich, Ted Han, Jing Li, Markus Lumpe, Hayden Melton, and James Noble. Qualitas corpus: A curated collection of java code for empirical studies. In *2010 Asia Pacific Software Engineering Conference (APSEC2010)*, pages 336–345, December 2010.

[6] Tiantian Wang, Mark Harman, Yue Jia, and Jens Krinke. Searching for Better Configurations: A Rigorous Approach to Clone Evaluation. In *Proceedings of the 2013 9th Joint Meeting on Foundations of Software Engineering*, pages 455–465, 2013.

Table IV: 63 Qualitas projects (new versions checked on 2016-09-27)

Projects	Old version	New versions	Latest change	Repo.	License	Notes
antlr4	4.0	4.5.4	25/09/2016	git	BSD License	
apache-ant	1.8.4	1.10.0	09/04/2016	git	Apache License, v.2.0	
argouml	0.34	0.35.4	11/01/2015	svn	Eclipse Public License v.1.0	
artofillusion	2.8.1	3.0.2	27/08/2016	svn	GNU General Public License v.2.0	
aspectj	1.6.9	1.8.9	12/05/2016	git	Eclipse Public License 1.0	
axion	1.0-M2	-	08/03/2013	-	Proprietary (BSD/Apache-style)	
batik	1.7	1.9.0	11/05/2016	svn	Apache License, v.2.0	
c-jdbc	2.0.2	-	16/09/2005	-	GNU Lesser General Public License, v.2.1	
castor	1.3.1	1.4.2	17/08/2016	git	Apache License, v.2.0	
cayenne	3.0.1	4.0.M4	26/09/2016	git	Apache License, v.2.0	
checkstyle	5.1	7.2	23/09/2016	git	GNU Lesser General Public License, v.2.1 & Apache License, v.2.0	*Cli, Logging and Beanutils* packages are from the Apache Commons project.
cobertura	1.9.4.1	2.1.2	01/06/2016	git	GNU General Public License v.2.0	
colt	1.2.0	-	09/09/2014	-	Proprietary (CERN)	Found multithreaded v.
columba	1.4	-	20/04/2007	-	Mozilla Public License 1.1	
commons-collections	3.2.1	4.2	12/09/2016	svn	Apache License, v.2.0	
compiere	330	-	-	-	GNU General Public License v.2.0	No longer OSS
db-derby	10.6.1.0	10.12.1	13/08/2016	svn	Apache License, v.2.0	
displaytag	1.2	2.0	17/08/2014	svn	MIT License	
drawswf	1.2.9	-	02/04/2013	-	GNU General Public License v.2.0	
drjava	20100913-r5387	???	03/09/2014	svn	BSD License	Build to see version?
exoportal	???	???		git	GNU Lesser General Public License v.3.0 & proprietary	Too many new projects
emma	2.0.5312	2.0.5312	09/05/2013	-	Common Public License 1.0	
findbugs	1.3.9	3.0.1	06/03/2015	-	GNU Lesser General Public License, v.2.0	
fit-java	1.1	-	04/06/2013	-	GNU General Public License v.2.0	
fitlibrary	20100806	???	29/07/2014	git	GNU General Public License v.2.0	
freecol	0.10.7	0.11.6	26/09/2016	git	GNU General Public License v.2.0	
freecs	1.3.20100406	-	22/04/2013	-	GNU General Public License v.3.0	
freemind	0.9.0	1.0.0	16/08/2016	git	GNU General Public License v.2.0+	
galleon	2.3.0	2.5.6	29/04/2013	-	GNU General Public License v.2.0	
ganttproject	2.0.9	2.8.1	16/08/2016	git	GNU General Public License v.3.0	
geotools	2.7-M3	16	27/09/2016	git	GNU Lesser General Public License, v.2.0	
hadoop	1.0.0	3.0.0-alpha2	26/09/2016	git	Apache License, v.2.0	
heritrix	1.14.4	-	05/06/2013	-	GNU Lesser General Public License, v.2.1	
hibernate	4.2.2	5.2.3	22/09/2016	git	GNU Lesser General Public License, v.2.1+	
hsqldb	2.0.0	2.3.4	13/09/2016	svn	BSD License	
htmlunit	2.8	2.24	26/09/2016	svn	Apache License, v.2.0	
ireport	3.7.5	-	28/05/2014	-	Affero GNU Public License, v.3.0	
itext	5.0.3	5.5.9	27/09/2016	git	Affero GNU Public License, v.3.0	
informa	0.7.0-alpha2	-	07/11/2008	-	GNU Lesser General Public License, v.2.1 & Apache Software License, v.1.1	
ivatagroupware	0.11.3	-	27/02/2013	-	GNU General Public License v.2.0	
jfin_datemath	R1_0_1	-	25/04/2013	-	GNU General Public License v.2.0	
joggplayer	114s	-	15/04/2013	-	GNU General Public License v.2.0	
jag	6.1	6.2	08/04/2013	-	GNU General Public License v.2.0 & BSD License	BSD License is for libraries.
james	2.2.0	2.3.2.1	14/08/2015	-	Apache License, v.2.0	
jasml	0.10	-	08/03/2013	-	Apache Software License	
jasperreports	3.7.4	6.3.1	27/09/2016	git	GNU Lesser General Public License, v.3.0	
javacc	5.0.0	7.0.0	15/08/2016	svn	Proprietary (Sun)	
jboss (wildfly)	5.1.0.GA	11.0.0.Alpha1	27/09/2016	git	GNU Lesser General Public License, v.2.1	Renamed to Wildfly.
jchempaint	3.0.1	3.4	01/09/2016	git	GNU Lesser General Public License, v.2.1+	
jedit	4.3.2	5.3.1	20/09/2016	svn	GNU General Public License v.2.0	
jena	2.6.3	3.1.1	16/09/2016	git	Apache License, v.2.0	
jext	5.0	-	18/08/2004	-	GNU General Public License v.2.0	
jfreechart	1.0.13	1.5.0	29/08/2016	git	GNU Lesser General Public License, v.2.0	
jgraph	5.13.0.0	3.6.0.0	07/09/2016	git	Proprietary (mxGraph License)	
jgraphpad	5.10.0.2	-	10/11/2006	-	GNU General Public License & GNU Lesser General Public License (derivatives)	
jgrapht	0.8.1	1.0.1	23/09/2016	git	GNU Lesser General Public License, v.2.1 & Eclipse Public License, v.1.0	
jgroups	2.10.0.GA	4.0.0	26/09/2016	git	Apache License, v.2.0	
jmoney	0.4.4	???	27/12/2015	git	GNU General Public License v.2.0	
jparse	0.96	-	29/07/2004	-	GNU Lesser General Public License, v.2.1	
jpf	1.5.1	???	13/01/2012	-	Apache License, v.2.0	
junit	4.11	4.12	04/12/2014	git	Eclipse Public License, v.1.0	
shiftone-jrat	0.6	1-beta-1	17/11/2007	svn	GNU Lesser General Public License, v.2.0	
vuze	4812	5730	23/09/2016	svn	GNU General Public License v.2.0	