Analyzing the Effect of Preprocessor Annotations on Code Clones

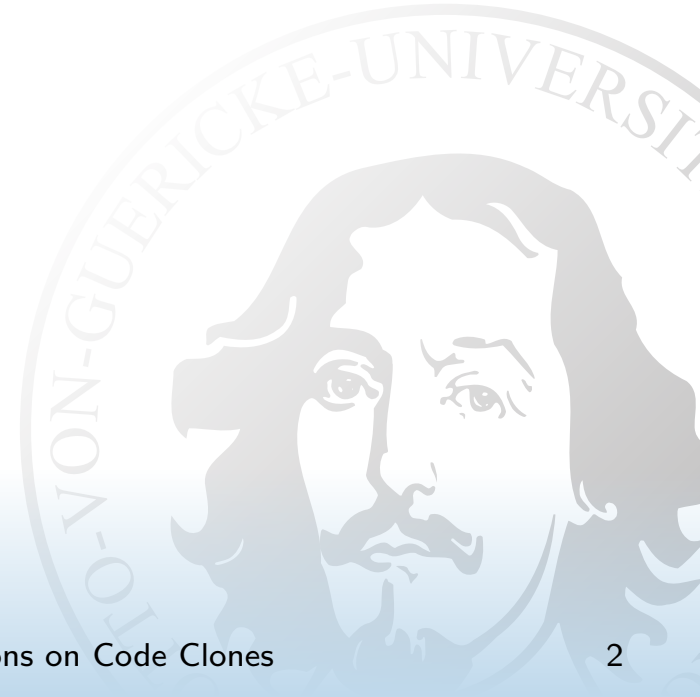
Sandro Schulze, Elmar Juergens, Janet Feigenspan

SCAM 2011 – September 25, 2011

INF

Preprocessor Annotations – #ifdef

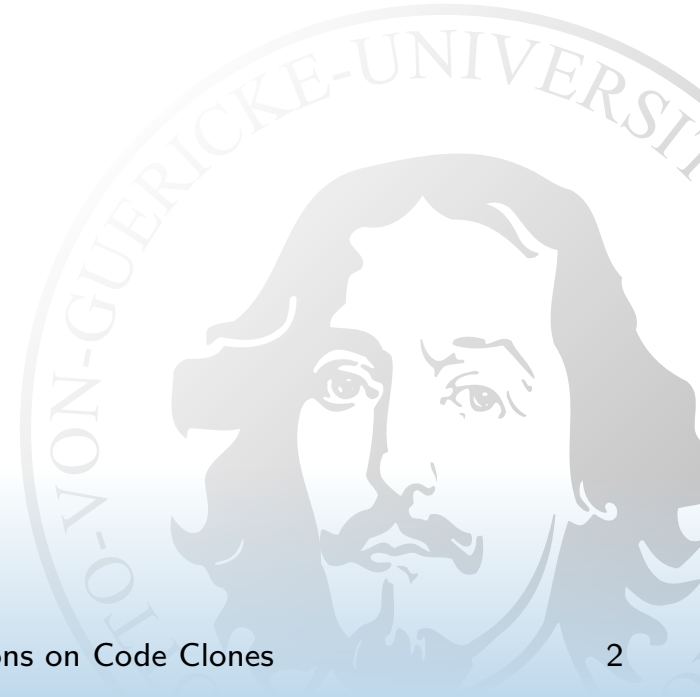
```
 void push(Object o
#ifdef SYNC
 , Transaction txn
#endif
 ){
    if (o==null
#ifdef SYNC
   || txn==null
#endif
   )
   return;
#ifdef SYNC
   Lock l=txn.lock(o);
#endif
   elementData[size++]
        = o;
#ifdef SYNC
   l.unlock();
#endif
   fireStackChanged();
 }
```



Preprocessor Annotations – #ifdef

undisciplined annotation

```
 void push(Object o
#ifdef SYNC
 , Transaction txn
#endif
 ){
    if (o==null
#ifdef SYNC
   || txn==null
#endif
   )
   return;
#ifdef SYNC
   Lock l=txn.lock(o);
#endif
   elementData[size++]
        = o;
#ifdef SYNC
   l.unlock();
#endif
   fireStackChanged();
 }
```



Preprocessor Annotations – #ifdef

undisciplined annotation

```
 void push(Object o
#ifdef SYNC
 , Transaction txn
#endif
 ){
    if (o==null
#ifdef SYNC
   || txn==null
#endif
   )
   return;
#ifdef SYNC
   Lock l=txn.lock(o);
#endif
   elementData[size++]
        = o;
#ifdef SYNC
   l.unlock();
#endif
   fireStackChanged();
 }
```

```
#ifdef SYNC
 void push(Object o,
      Transaction txn) {
   if (o==null || txn==null)
     return;
   Lock l = txn.lock(o);
   elementData[size++] = o;
   l.unlock();
   fireStackChanged();
 }
#else
 void push(Object o) {
   if (o==null)
     return;
   elementData[size++] = o;
   fireStackChanged();
 }
#endif
```

Preprocessor Annotations – #ifdef

undisciplined annotation

```
 void push(Object o
#ifdef SYNC
 , Transaction txn
#endif
 ){
    if (o==null
#ifdef SYNC
   || txn==null
#endif
   )
   return;
#ifdef SYNC
   Lock l=txn.lock(o);
#endif
   elementData[size++]
        = o;
#ifdef SYNC
   l.unlock();
#endif
   fireStackChanged();
 }
```

disciplined annotation

```
#ifdef SYNC
 void push(Object o,
      Transaction txn) {
   if (o==null || txn==null)
     return;
   Lock l = txn.lock(o);
   elementData[size++] = o;
   l.unlock();
   fireStackChanged();
 }
#else
 void push(Object o) {
   if (o==null)
     return;
   elementData[size++] = o;
   fireStackChanged();
 }
#endif
```

Preprocessor Annotations – #ifdef

undisciplined annotation

```
 void push(Object o
#ifdef SYNC
 , Transaction txn
#endif
 ){
#if
   [illegible]
#en
   )
   r
#if
   I
#endif
   elementData[size++]
        = o;
#ifdef SYNC
   l.unlock();
#endif
   fireStackChanged();
 }
```

disciplined annotation

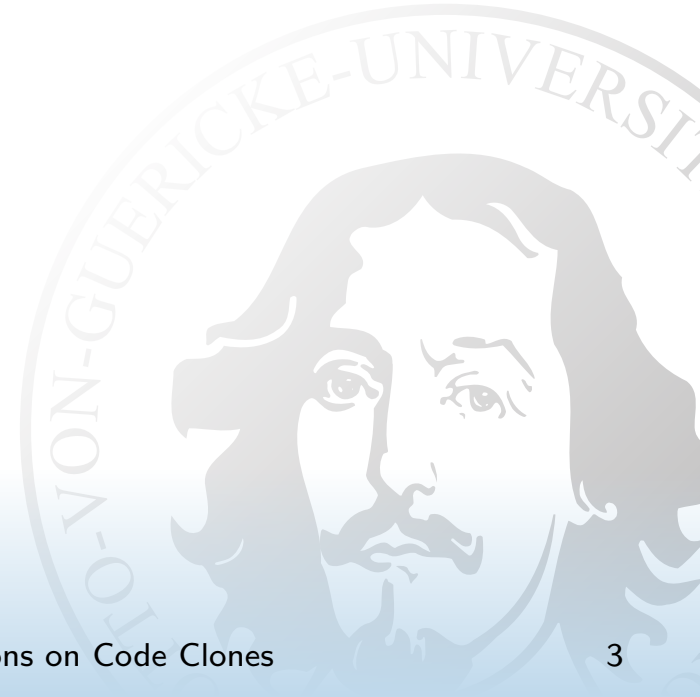
```
#ifdef SYNC
 void push(Object o,
      Transaction txn) {
                    ==null)

                    o);
                    ] = o;

                    ;
 void push(Object o) {
   if (o==null)
     return;
   elementData[size++] = o;
   fireStackChanged();
 }
#endif
```

Definition Disciplined Annotations (Liebig et al.)

Annotations on one or a sequence of *entire functions* and *type defintions* . . . annotations on one or a sequence of *entire statements* and *elements inside type definitions* . . . are *disciplined*.

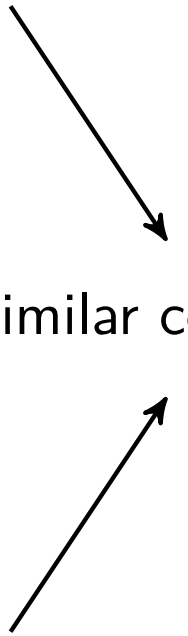
Code Clones

```
#ifdef SYNC
 void push(Object o,
      Transaction txn) {
   if (o==null || txn==null)
     return;
   Lock l = txn.lock(o);
   elementData[size++] = o;
   l.unlock();
   fireStackChanged();
 }
#else
 void push(Object o) {
   if (o==null)
     return;
   elementData[size++] = o;
   fireStackChanged();
 }
#endif
```

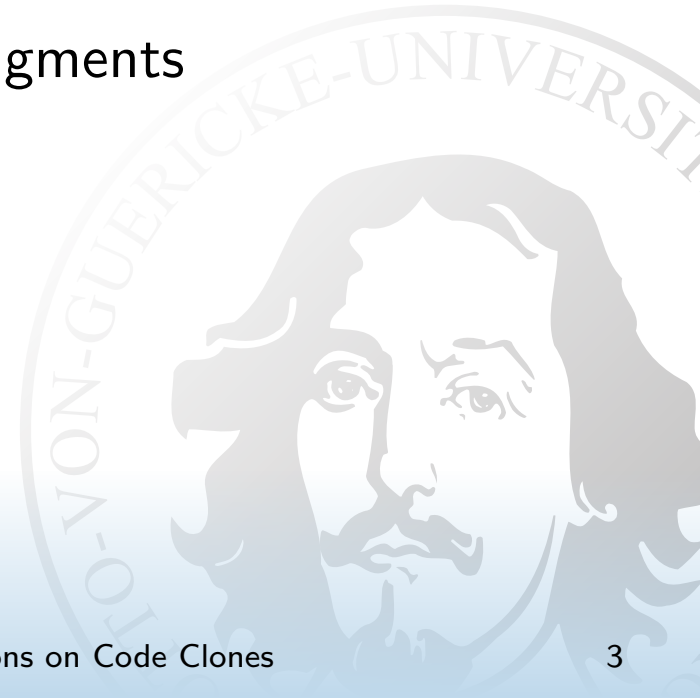


Code Clones

```
#ifdef SYNC
 void push(Object o,
      Transaction txn) {
   if (o==null || txn==null)
     return;
   Lock l = txn.lock(o);
   elementData[size++] = o;
   l.unlock();
   fireStackChanged();
 }
#else
 void push(Object o) {
   if (o==null)
     return;
   elementData[size++] = o;
   fireStackChanged();
 }
#endif
```



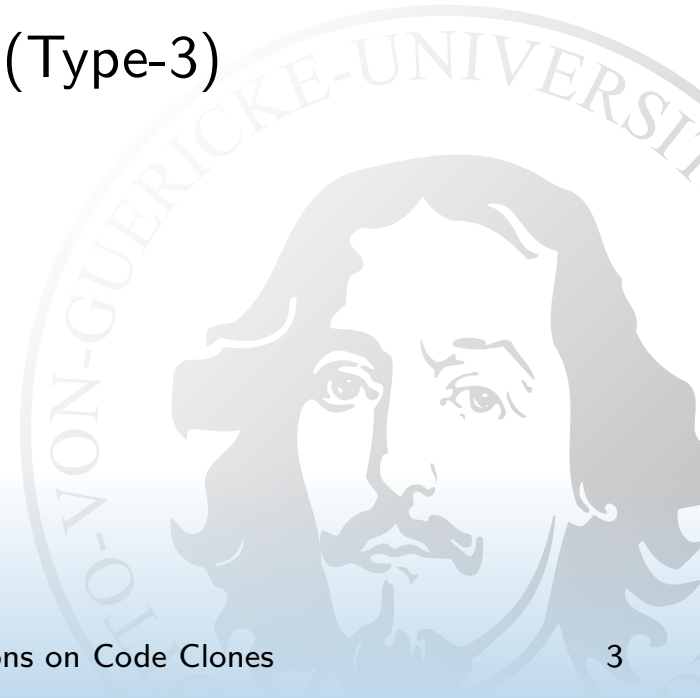
similar code fragments



Code Clones

```
#ifdef SYNC
 void push(Object o,
      Transaction txn) {
   if (o==null || txn==null)
     return;
   Lock l = txn.lock(o);
   elementData[size++] = o;
   l.unlock();
   fireStackChanged();
 }
#else
 void push(Object o) {
   if (o==null)
     return;
   elementData[size++] = o;
   fireStackChanged();
 }
#endif
```

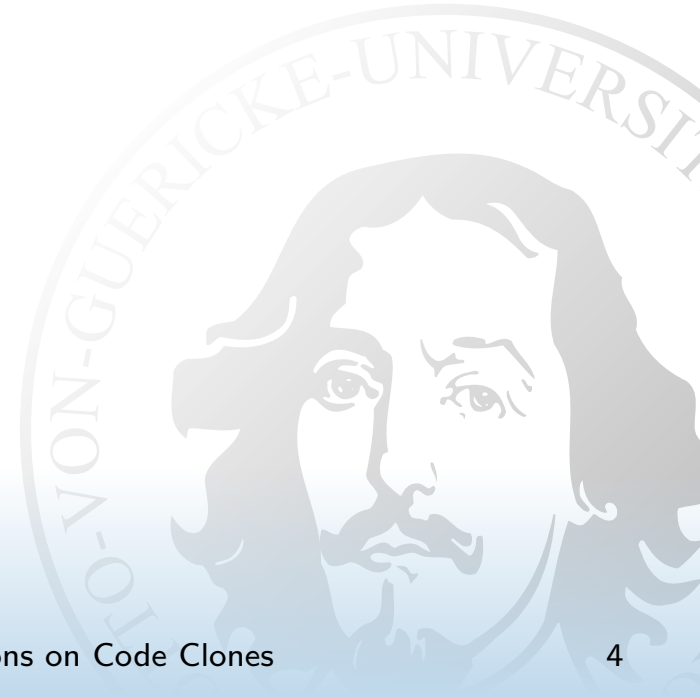
#ifdef clones (Type-3)



Motivation

Benefits and Drawbacks

- ***undisciplined*** annotations
  - high expressiveness
  - tangled source code, difficult to understand/modify
- ***disciplined*** annotations
  - improved readability, reduced programmer effort
  - lower expressiveness, prone to code clones



Motivation

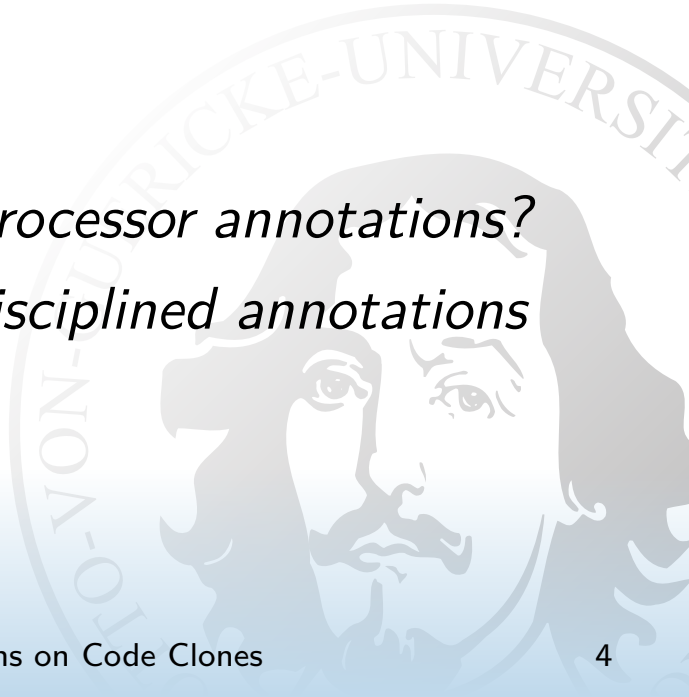
Benefits and Drawbacks

- ***undisciplined*** annotations
  - high expressiveness
  - tangled source code, difficult to understand/modify
- ***disciplined*** annotations
  - improved readability, reduced programmer effort
  - lower expressiveness, prone to code clones

Research Questions:

RQ1 *To what extent code clones exist in preprocessor annotations?*

RQ2 *Differences between disciplined and undisciplined annotations wrt code clones?*



Motivation

Benefits and Drawbacks

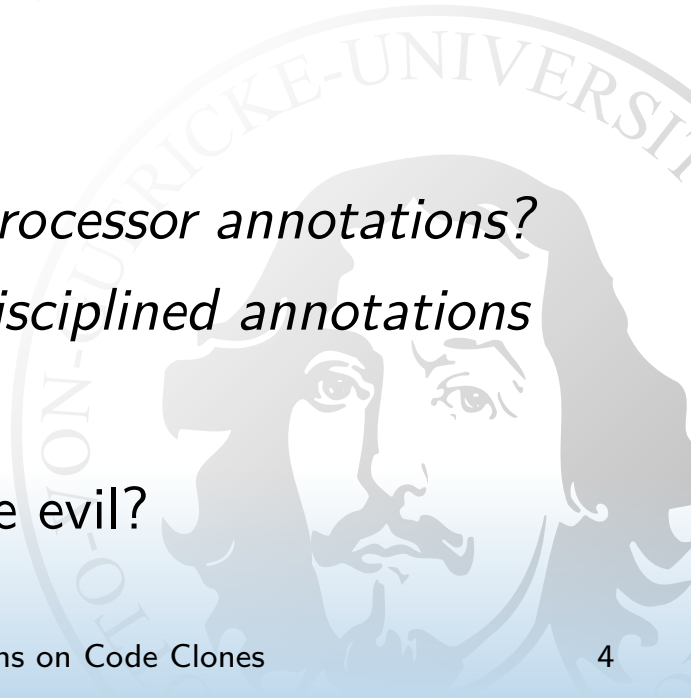
- ***undisciplined*** annotations
  - high expressiveness
  - tangled source code, difficult to understand/modify
- ***disciplined*** annotations
  - improved readability, reduced programmer effort
  - lower expressiveness, prone to code clones

Research Questions:

RQ1 *To what extent code clones exist in preprocessor annotations?*

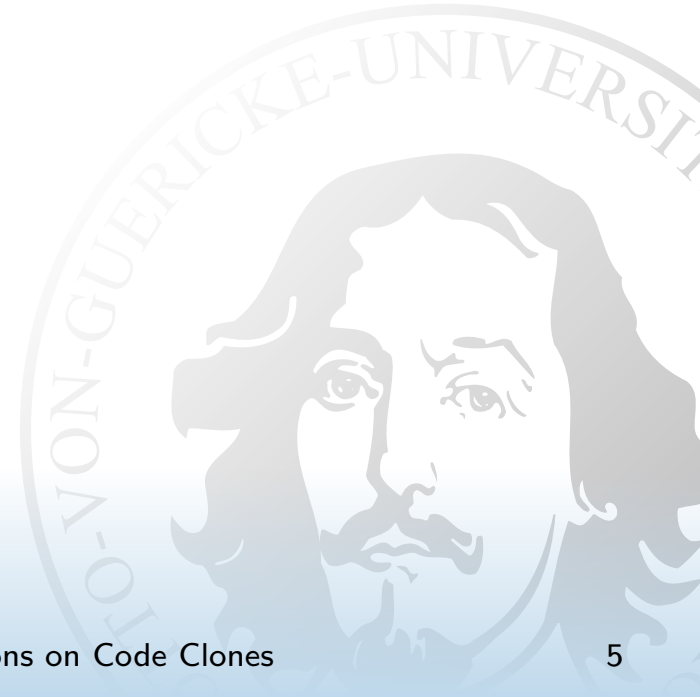
RQ2 *Differences between disciplined and undisciplined annotations wrt code clones?*

Big Picture: Which kind of annotation is more evil?



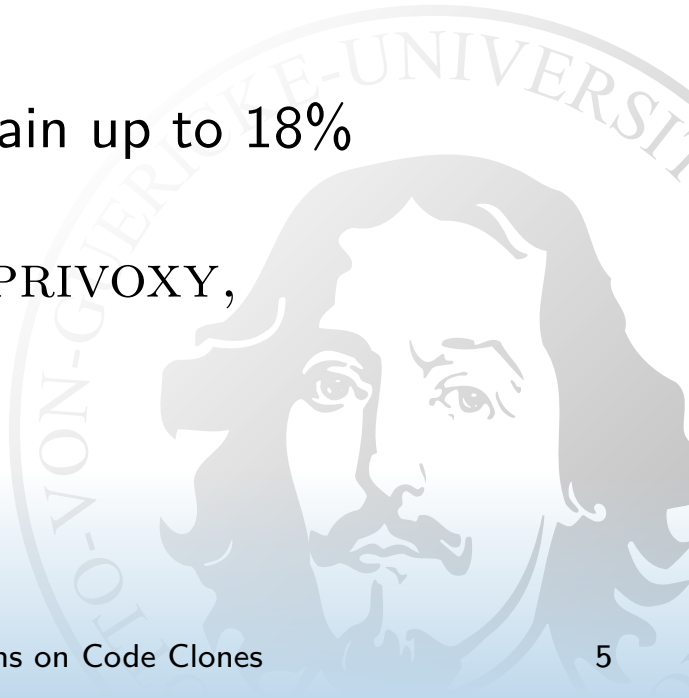
Case Study

- 15 open source systems in C programming language
- Size between 25 KLOC and 490 KLOC



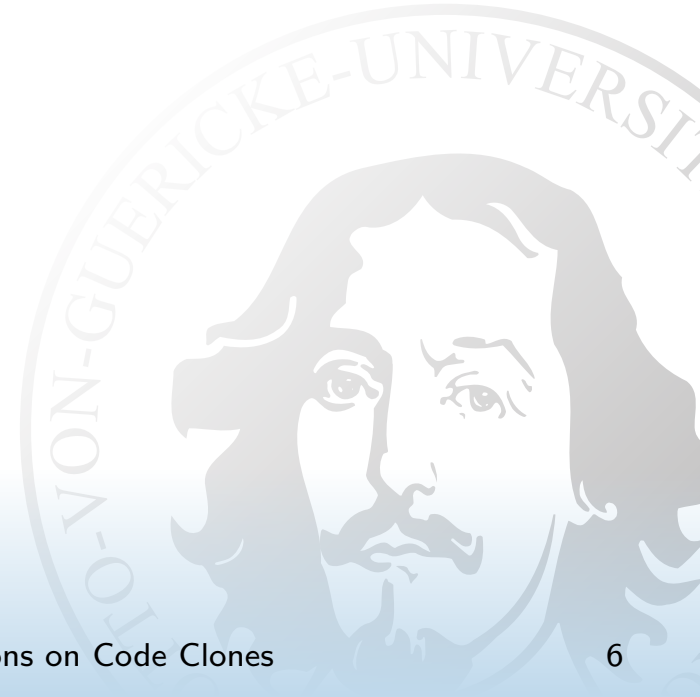
Case Study

- 15 open source systems in C programming language
- Size between 25 KLOC and 490 KLOC
- Split into two groups:
  1. Seven "disciplined" systems → almost no undisciplined annotations
     BERKELEYDB, DIA, GHOSTSCRIPT, LIGHTTPD, MINIX, PARROT, PYTHON
  2. Eight "undisciplined" systems → contain up to 18% undisciplined annotations
     CHEROKEE, GNUPLOT, LYNX, PHP, PRIVOXY, SENDMAIL, TCL, VIM



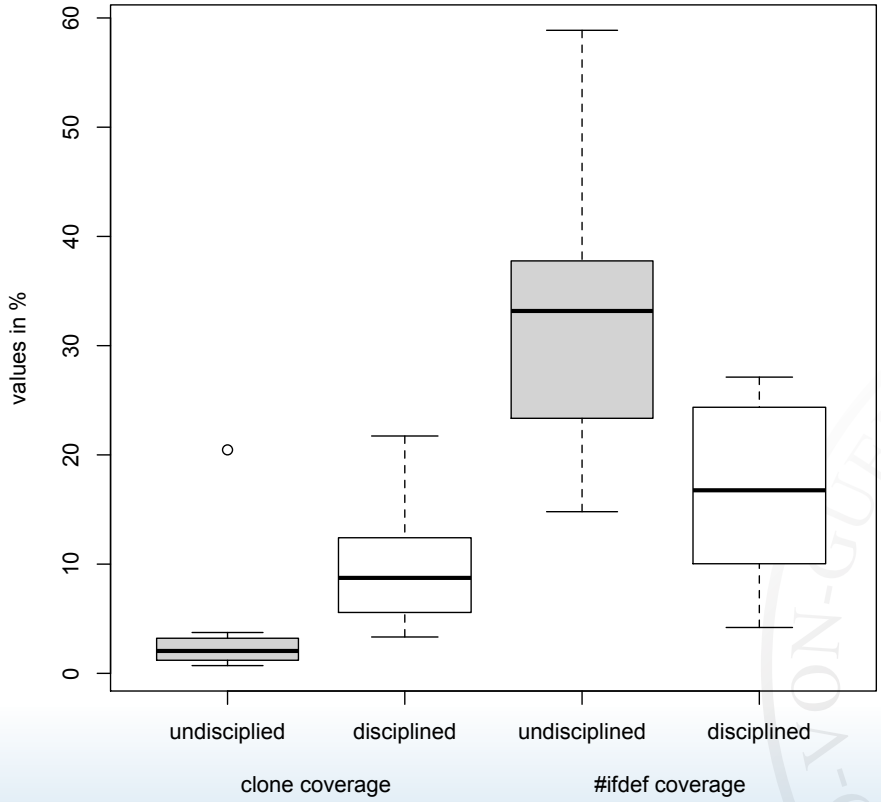
Subject Systems – Overview

- *Clone Coverage* – amount of code that contains code clones
- *#ifdef Coverage* – amount of code that is wrapped around by *#ifdefs*



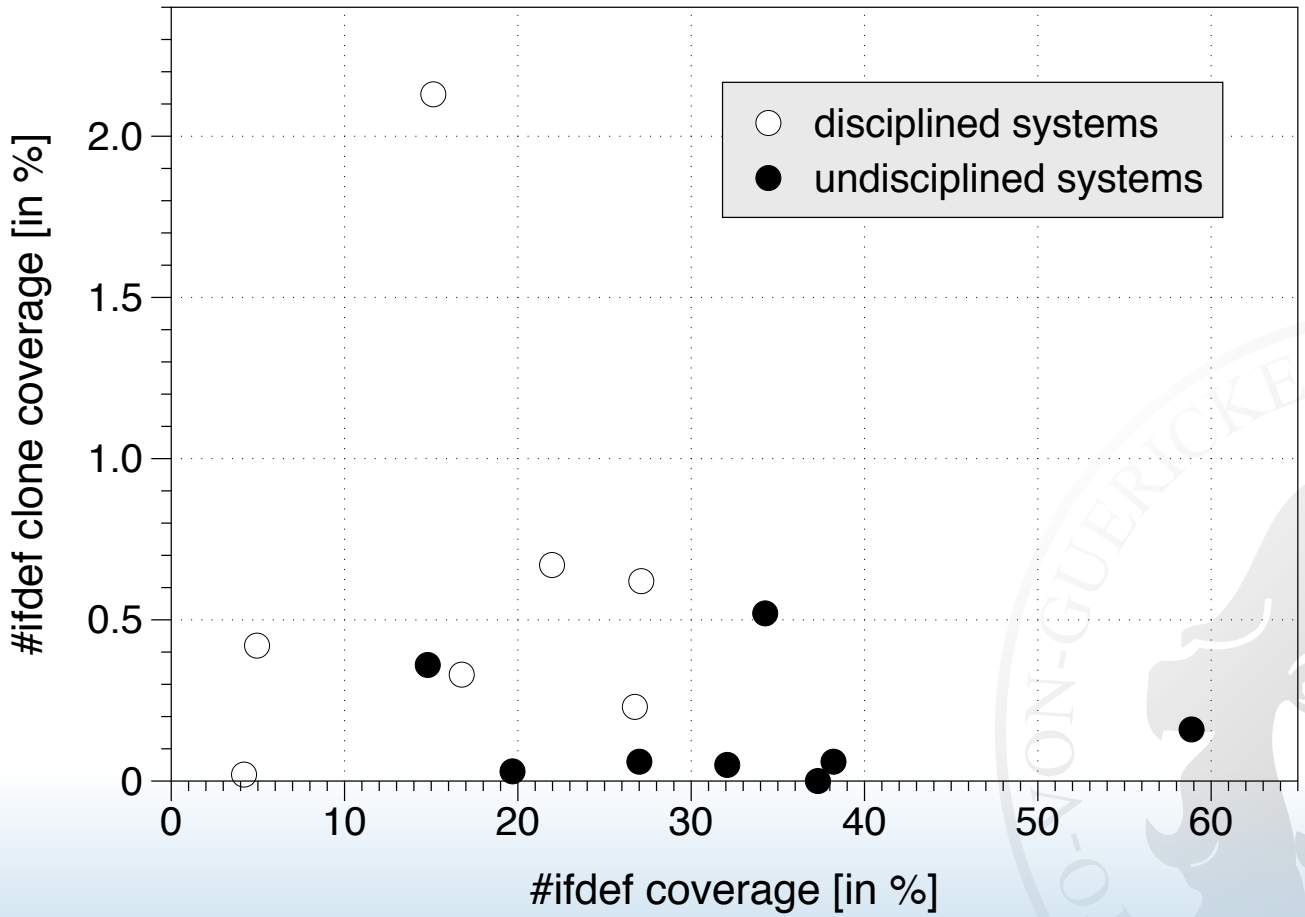
Subject Systems – Overview

- *Clone Coverage* – amount of code that contains code clones
- *#ifdef Coverage* – amount of code that is wrapped around by *#ifdefs*



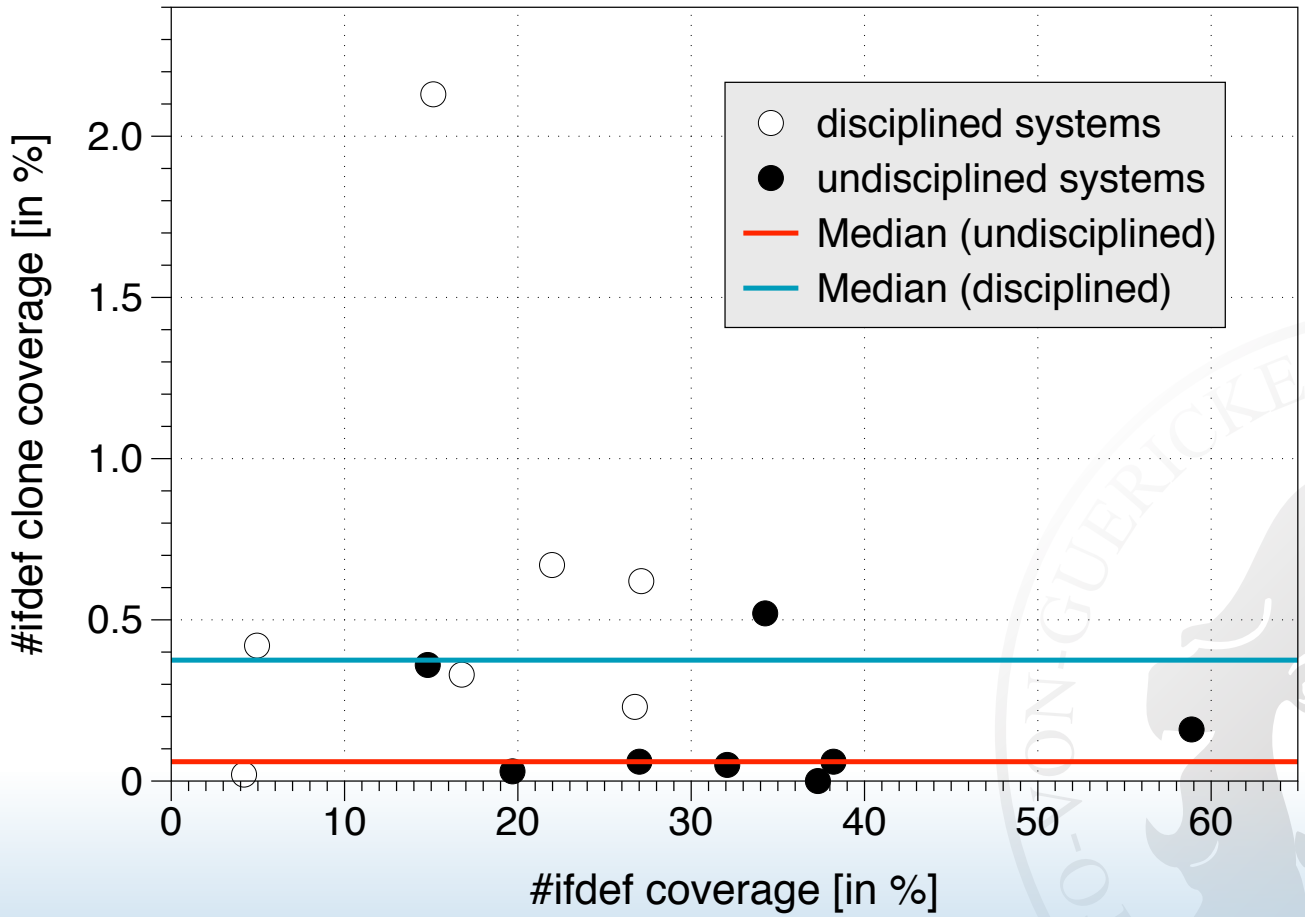
Results (I)

- How many clones occur within preprocessor annotations ?



Results (I)

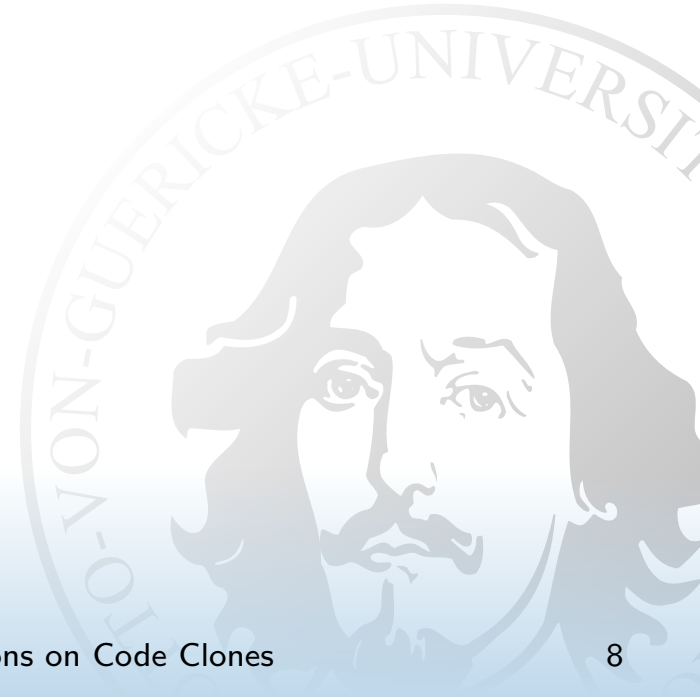
- How many clones occur within preprocessor annotations ?
- Does the kind of preprocessor annotation matter ?



Results (II)

Other correlations that influence the result ?

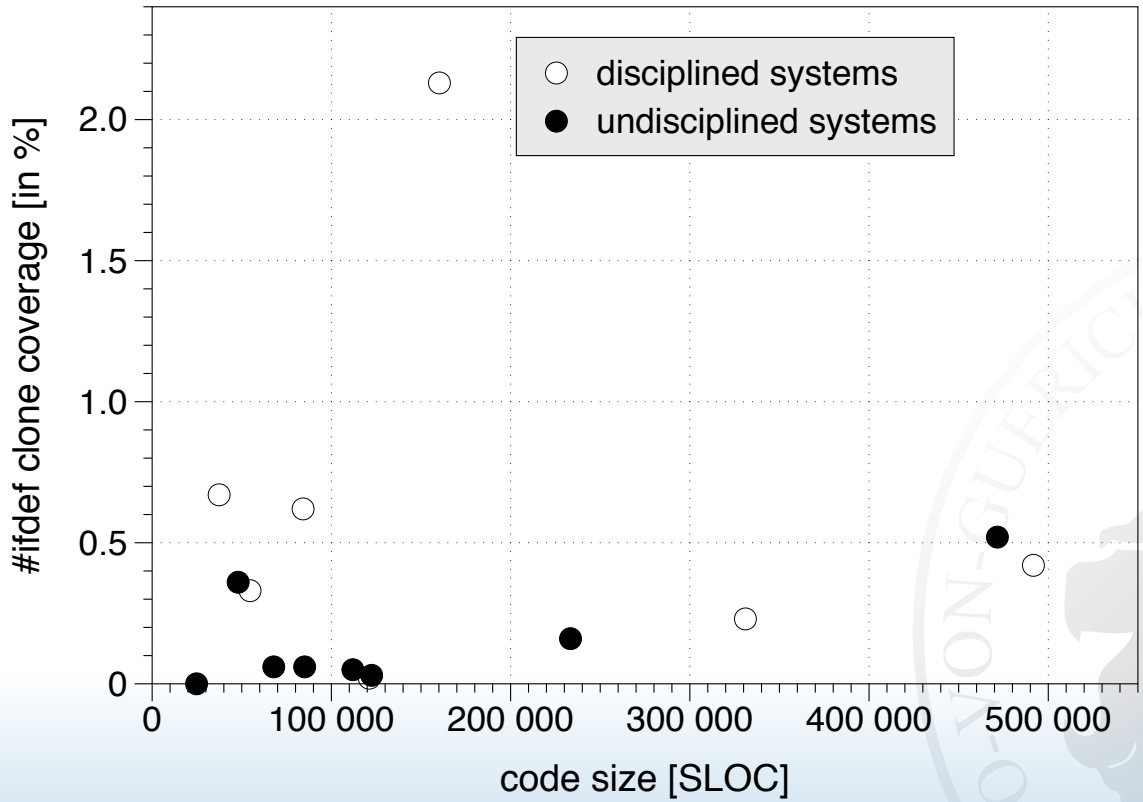
- Does code size matter ?



Results (II)

Other correlations that influence the result ?

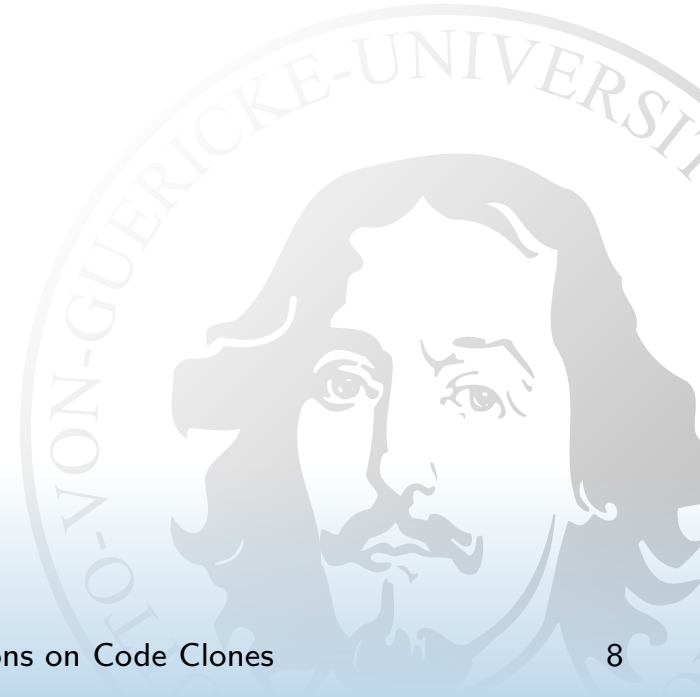
- Does code size matter ?



Results (II)

Other correlations that influence the result ?

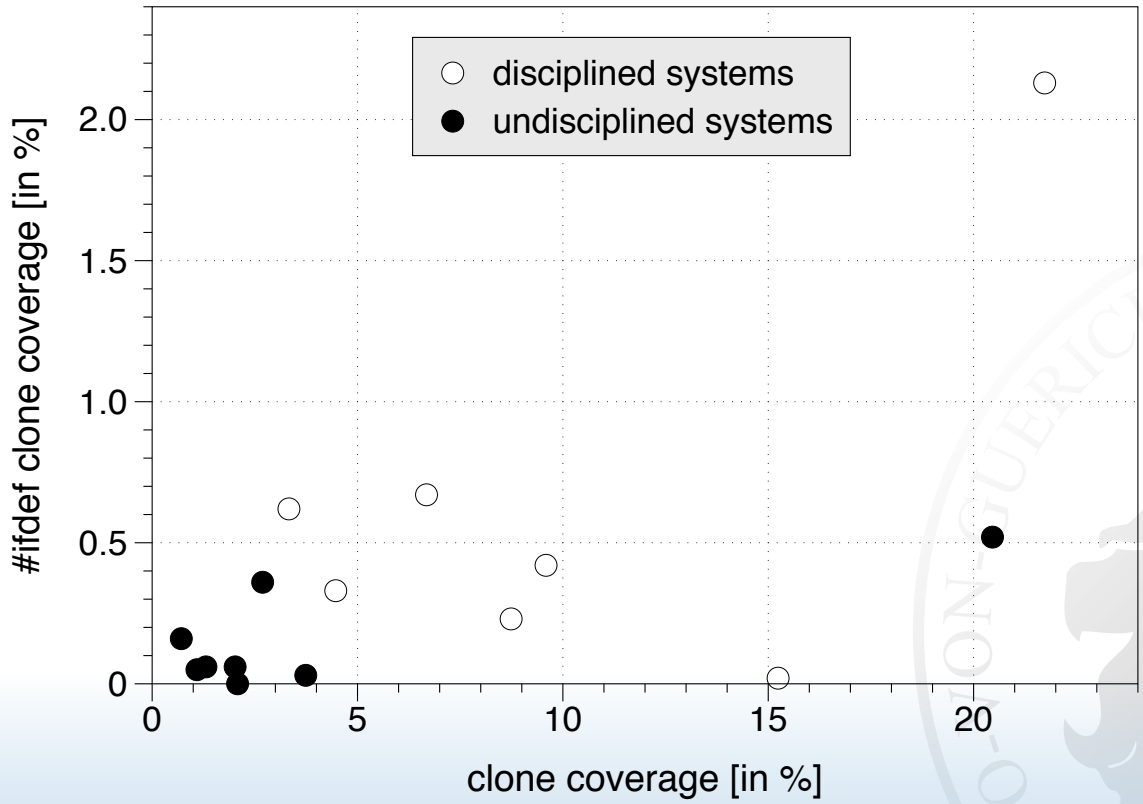
- Does code size matter ?
- Does clone coverage matter ?



Results (II)

Other correlations that influence the result ?

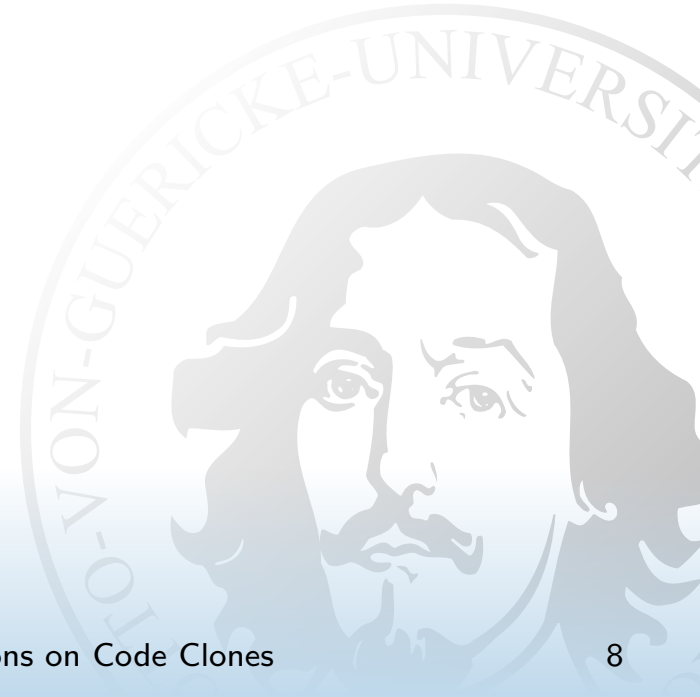
- Does code size matter ?
- Does clone coverage matter ?



Results (II)

Other correlations that influence the result ?

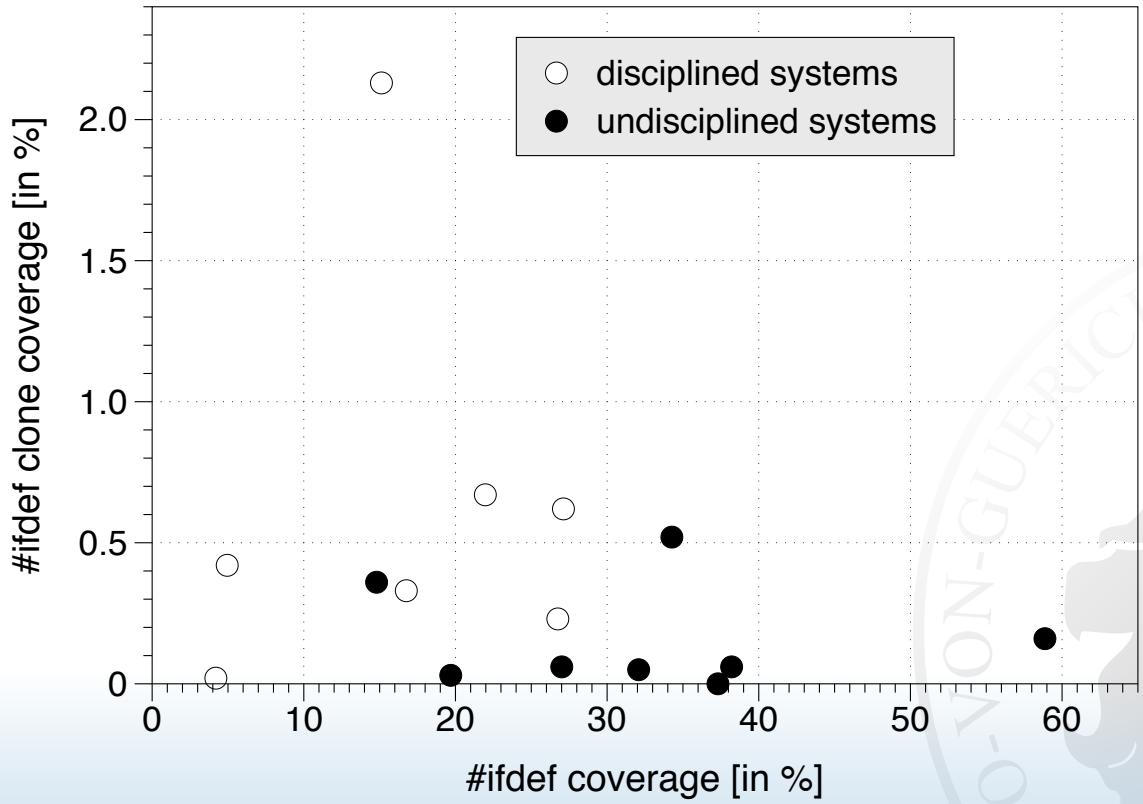
- Does code size matter ?
- Does clone coverage matter ?
- Does *#ifdef* coverage matter ?



Results (II)

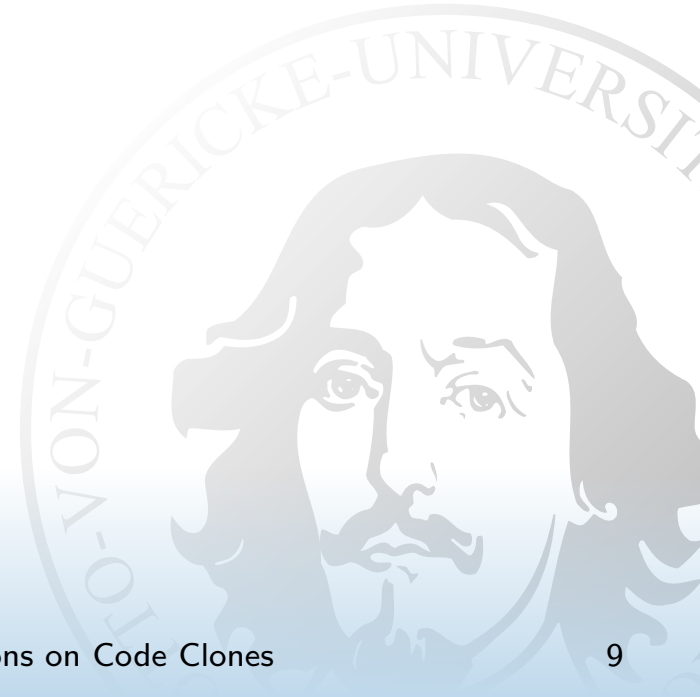
Other correlations that influence the result ?

- Does code size matter ?
- Does clone coverage matter ?
- Does *#ifdef* coverage matter ?



Conclusions

- #ifdef clone coverage is rather small (between 0 and 2 %)
- Higher amount of *#ifdef clones* in disciplined systems
- No further correlations found

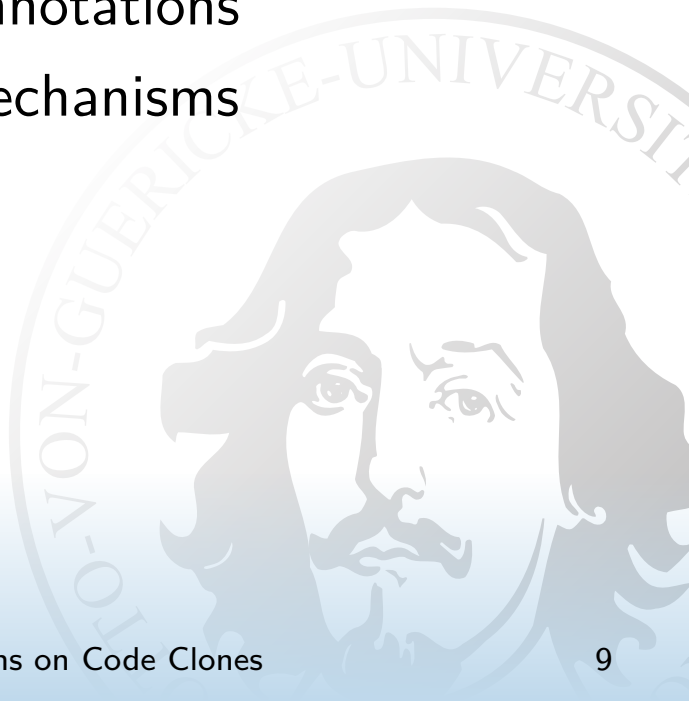


Conclusions

- #ifdef clone coverage is rather small (between 0 and 2 %)
- Higher amount of *#ifdef clones* in disciplined systems
- No further correlations found

Future Work:

- Detailed analysis of *#ifdef clones*
- Study on harmfullness of undisciplined annotations
- Relation of code clones and variability mechanisms



Thank You !