Scaling Regression Testing to Large Software Systems

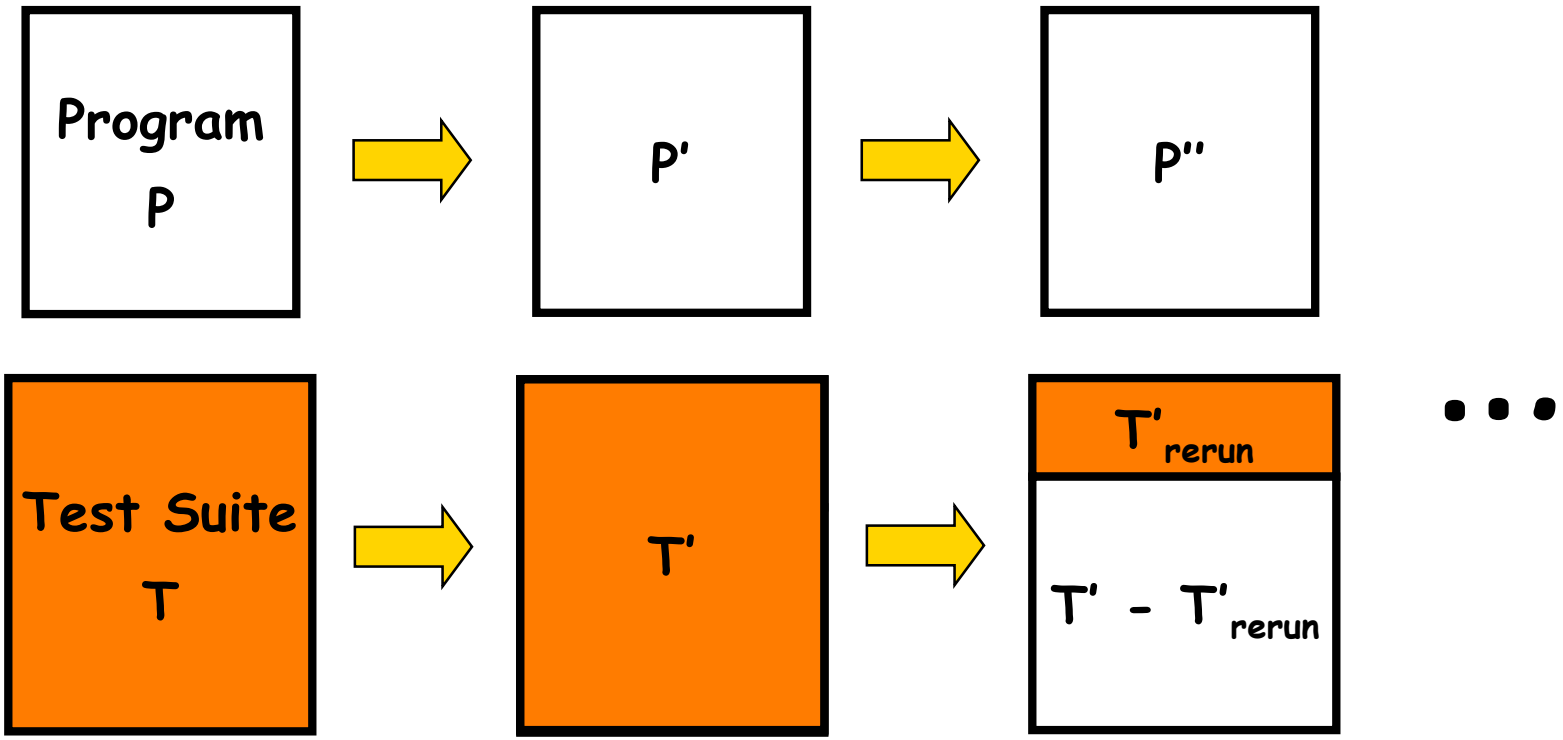
Alessandro Orso

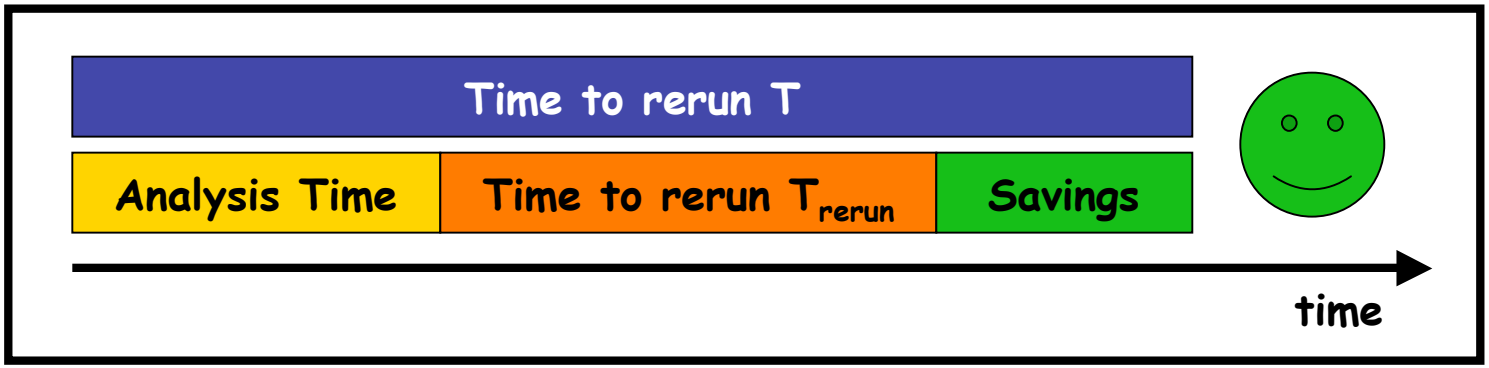
Co-authors: Nanjuan Shi,
Mary Jean Harrold

College of Computing
Georgia Institute of Technology

Supported in part by National Science Foundation (awards CCR-0306372, CCR-0205422, CCR-9988294, CCR-0209322, SBE-0123532, and CCR-0080900) and Boeing Commercial Airplane Group.

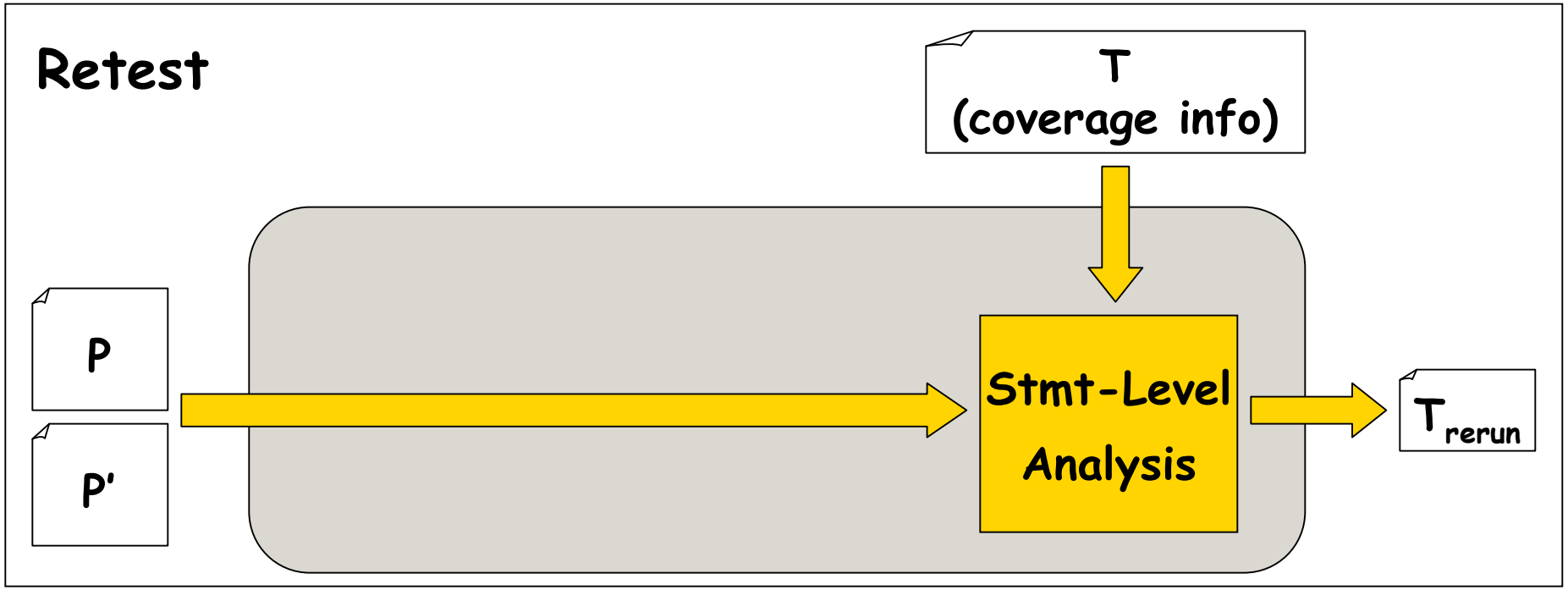
Testing of Evolving Software

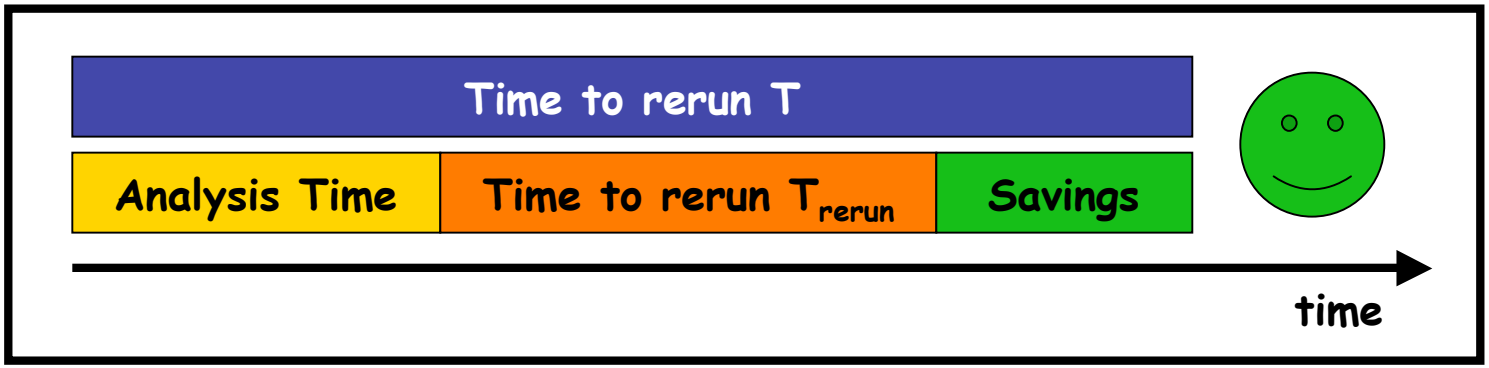




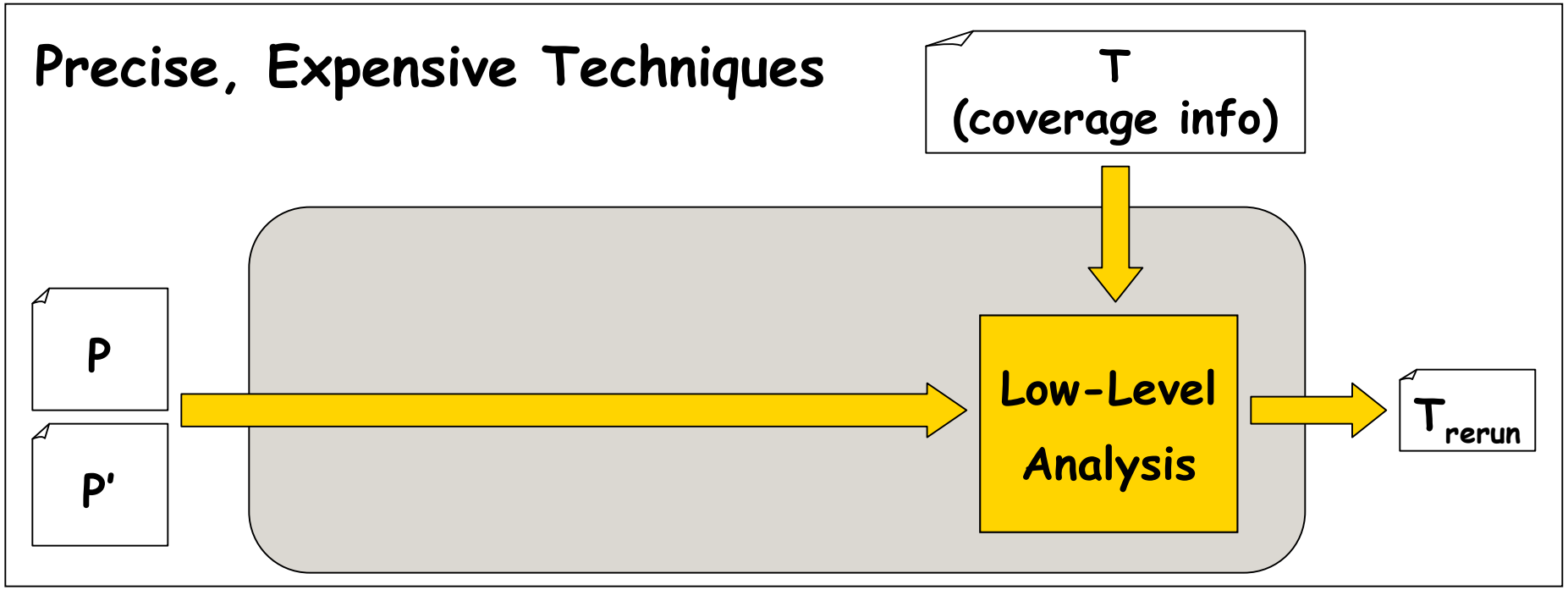
Previous and Related Work

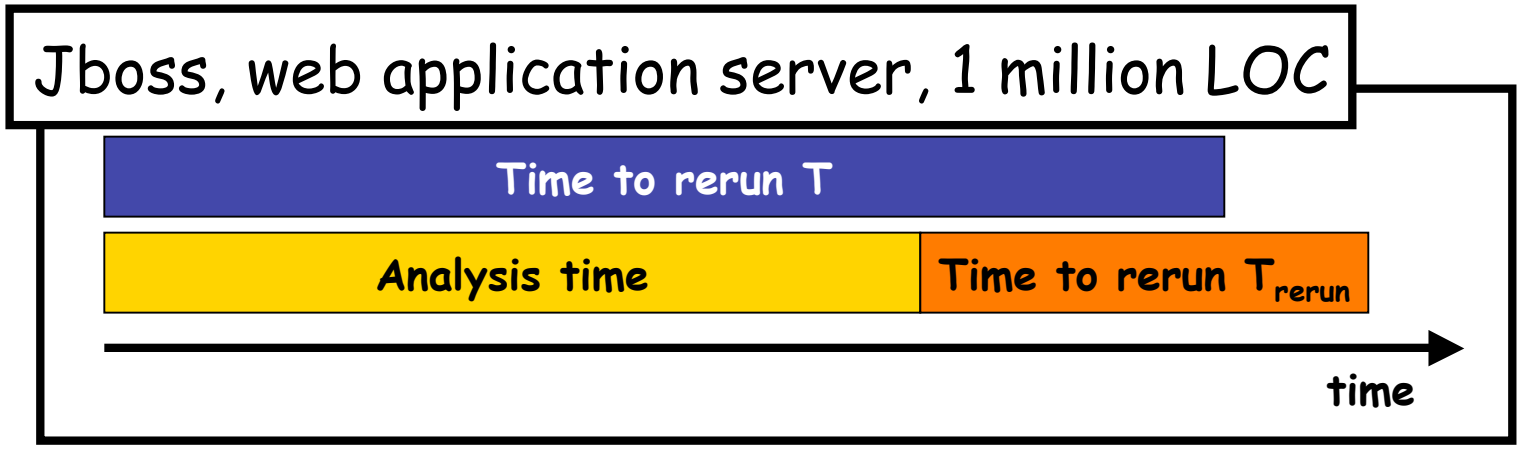
[OOPSLA01]



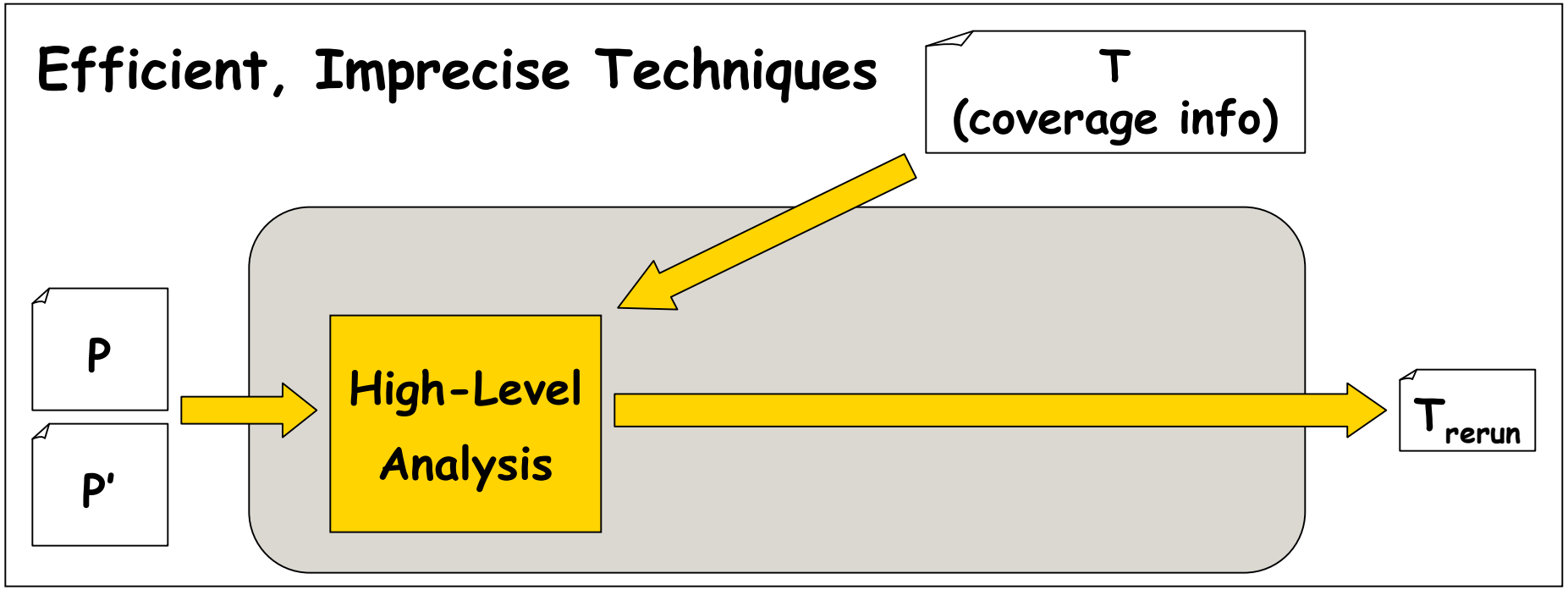


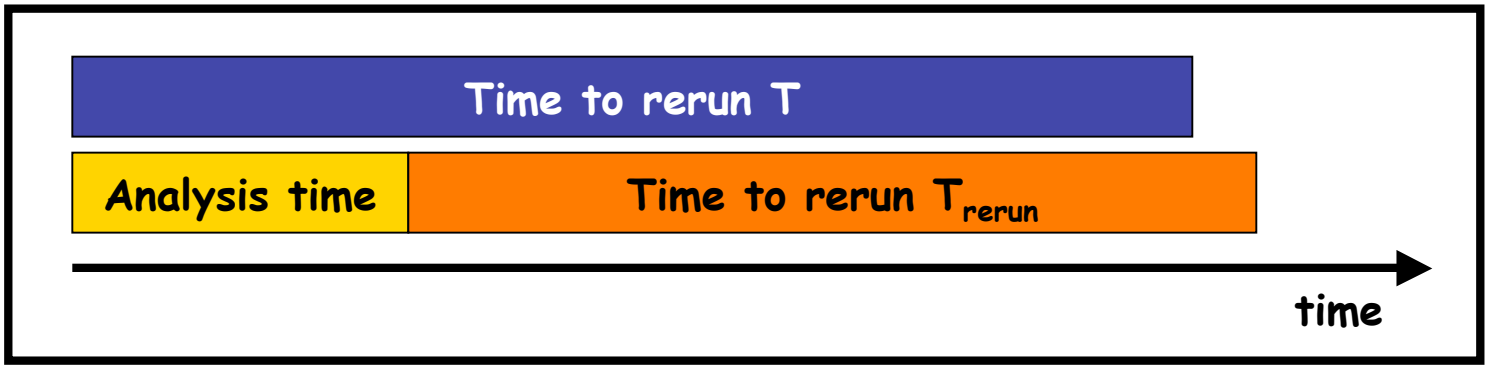
Previous and Related Work



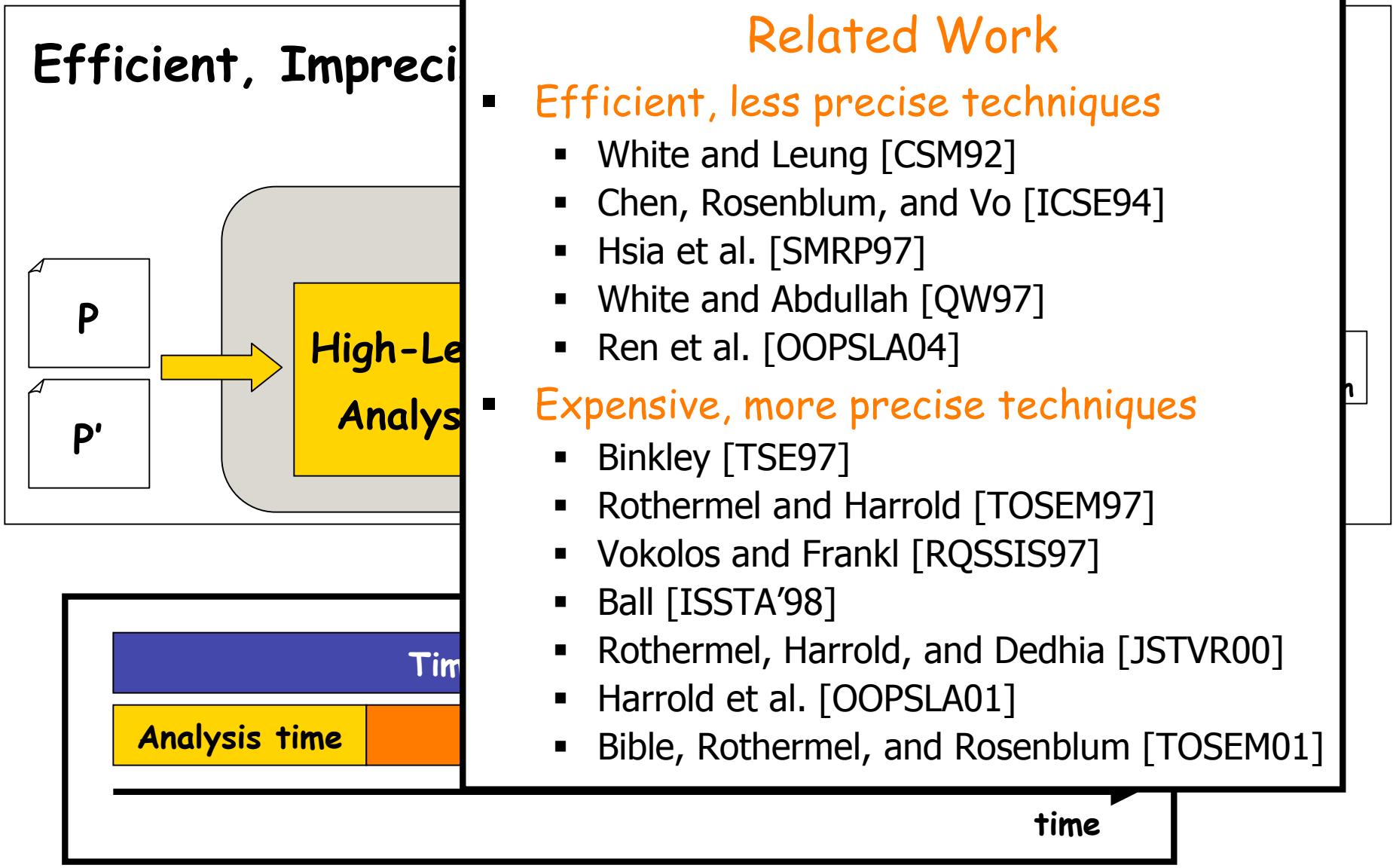


Previous and Related Work





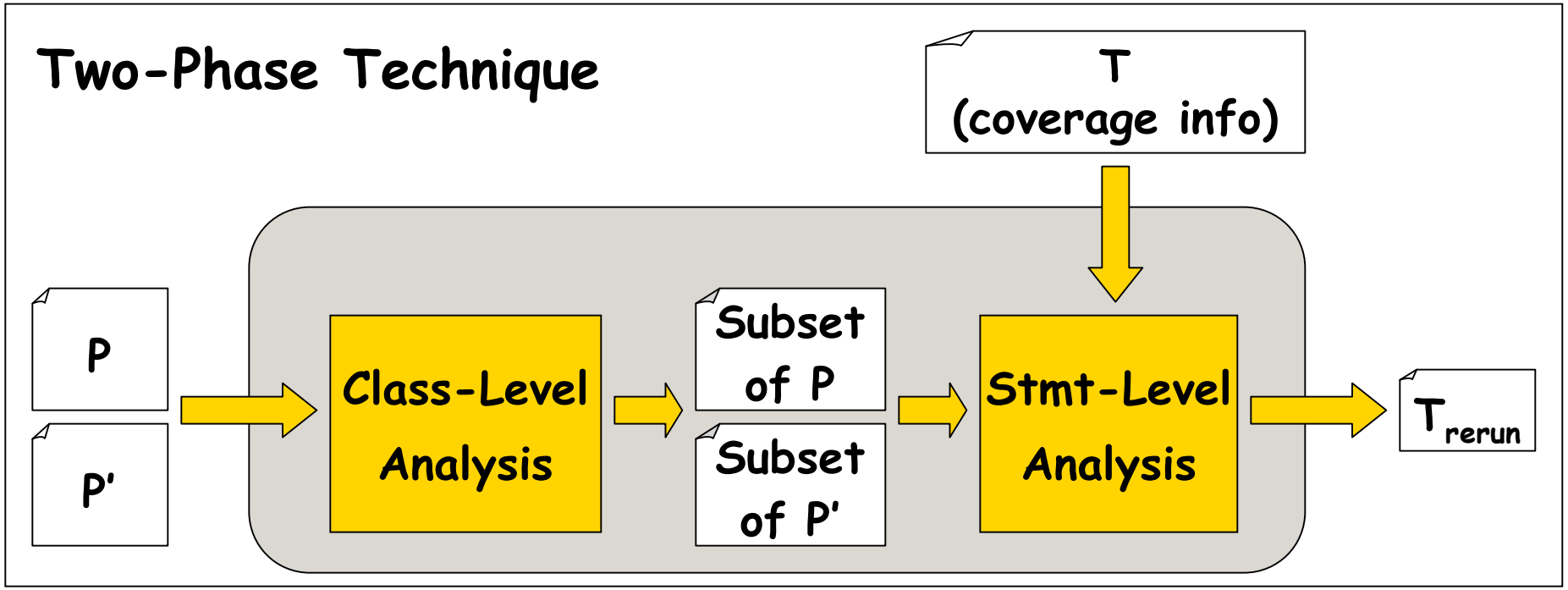
Previous and Related Work



Related Work

- Efficient, less precise techniques
  - White and Leung [CSM92]
  - Chen, Rosenblum, and Vo [ICSE94]
  - Hsia et al. [SMRP97]
  - White and Abdullah [QW97]
  - Ren et al. [OOPSLA04]
- Expensive, more precise techniques
  - Binkley [TSE97]
  - Rothermel and Harrold [TOSEM97]
  - Vokolos and Frankl [RQSSIS97]
  - Ball [ISSTA'98]
  - Rothermel, Harrold, and Dedhia [JSTVR00]
  - Harrold et al. [OOPSLA01]
  - Bible, Rothermel, and Rosenblum [TOSEM01]

Proposed Solution



Two-phase approach

- Class-Level analysis ➔ subset of P and P'
- Stmt-Level analysis on the subset ➔ T_{rerun}

Outline

- Background
- Technique
- Empirical Evaluation
- Conclusion

Motivating Example

P

```
class A {
  void foo() {…} }
class B extends A {

}
class C extends B {}
class D {
void bar() {
    A ref=null;
    switch(somevar) {
      case '1': ref=new A(); break;
      case '2': ref=new B(); break;
      case '3': ref=new C(); break; }
   ref.foo();
} }
class E extends D {}
class F {
  void bar(D d) {…}}
```

Motivating Example

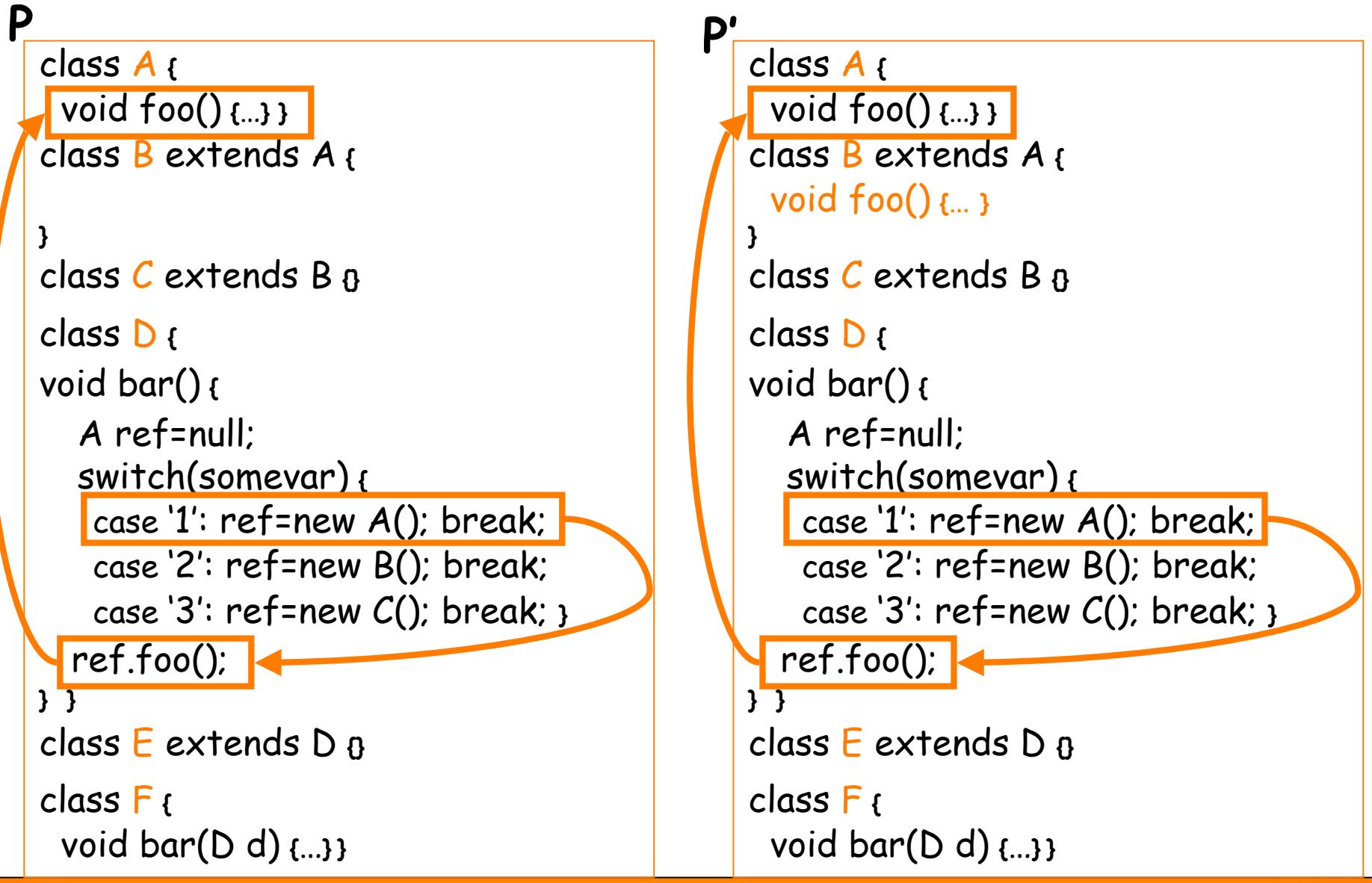
P

```
class A {
  void foo() {…} }
class B extends A {
                    }
class C extends B {}
class D {
void bar() {
    A ref=null;
    switch(somevar) {
      case '1': ref=new A(); break;
      case '2': ref=new B(); break;
      case '3': ref=new C(); break; }
   ref.foo();
} }
class E extends D {}
class F {
  void bar(D d) {…}}
```

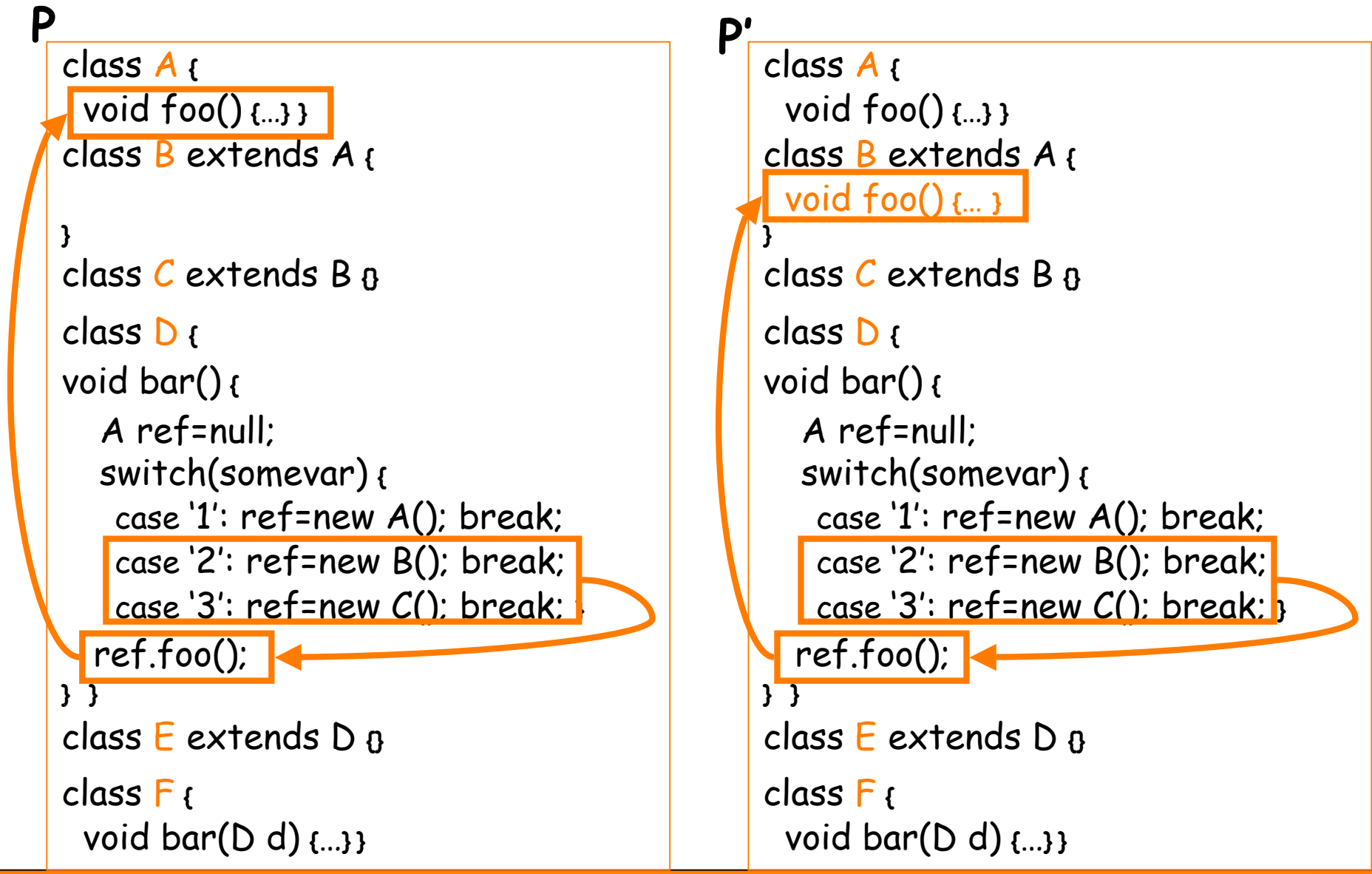
P'

```
class A {
  void foo() {…} }
class B extends A {
  void foo() {… }
}
class C extends B {}
class D {
void bar() {
    A ref=null;
    switch(somevar) {
      case '1': ref=new A(); break;
      case '2': ref=new B(); break;
      case '3': ref=new C(); break; }
   ref.foo();
} }
class E extends D {}
class F {
  void bar(D d) {…}}
```

Motivating Example



Motivating Example



Class-Level Analysis

P

```
class B extends A {

}
```

P'

```
class B extends A {
  void foo() {… }
}
```

Class-Level Analysis

P

```
class A {
  void foo() {…} }
class B extends A {

}
class C extends B {}
```

P'

```
class A {
  void foo() {…} }
class B extends A {
  void foo() {… }
}
class C extends B {}
```

Class-Level Analysis

P

```
class A {
  void foo() {…} }
class B extends A {

}
class C extends B {}
class D {
void bar() {
    A ref=null;
    switch(somevar) {
     case '1': ref=new A(); break;
     case '2': ref=new B(); break;
     case '3': ref=new C(); break; }
   ref.foo();
} }
```

P'

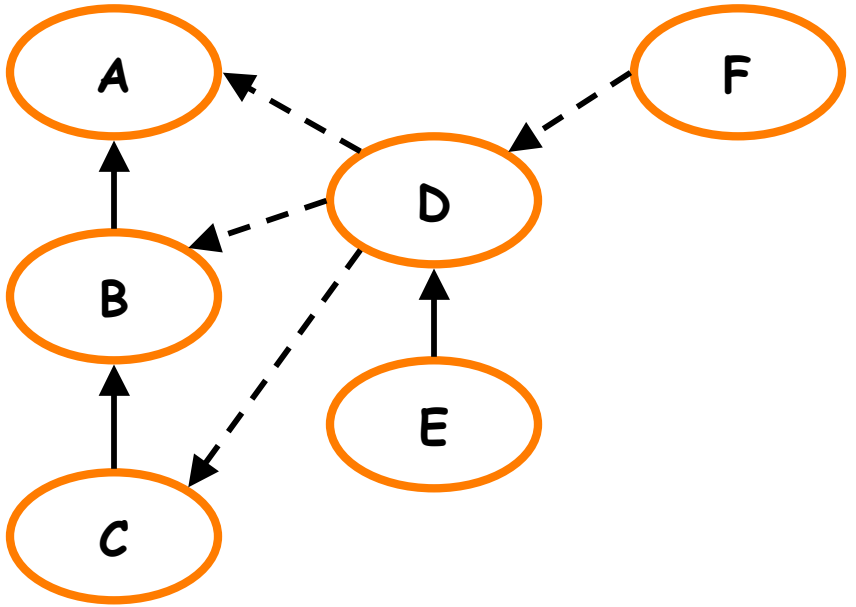
```
class A {
  void foo() {…} }
class B extends A {
  void foo() {… }
}
class C extends B {}
class D {
void bar() {
    A ref=null;
    switch(somevar) {
     case '1': ref=new A(); break;
     case '2': ref=new B(); break;
     case '3': ref=new C(); break; }
   ref.foo();
} }
```

Class-Level Analysis

P / P'

```
class A {
  void foo() {…} }
class B extends A {
  void foo() {… }
}
class C extends B {}
class D {
void bar() {
    A ref=null;
    switch(somevar) {
      case '1': ref=new A(); break;
      case '2': ref=new B(); break;
      case '3': ref=new C(); break; }
   ref.foo();
} }
class E extends D {}
class F {
  void bar(D d) {…}}
```

Interclass Relation Graph (for P an P')

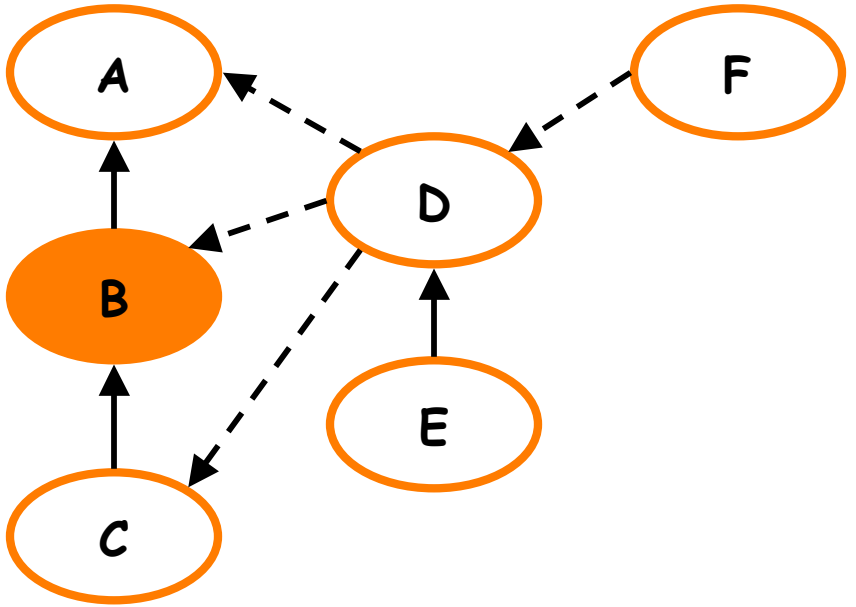


Class-Level Analysis

P / P'

```
class A {
  void foo() {…} }
class B extends A {
  void foo() {… }
}
class C extends B {}
class D {
void bar() {
    A ref=null;
    switch(somevar) {
     case '1': ref=new A(); break;
     case '2': ref=new B(); break;
     case '3': ref=new C(); break; }
   ref.foo();
} }
class E extends D {}
class F {
  void bar(D d) {…}}
```

Interclass Relation Graph (for P an P')

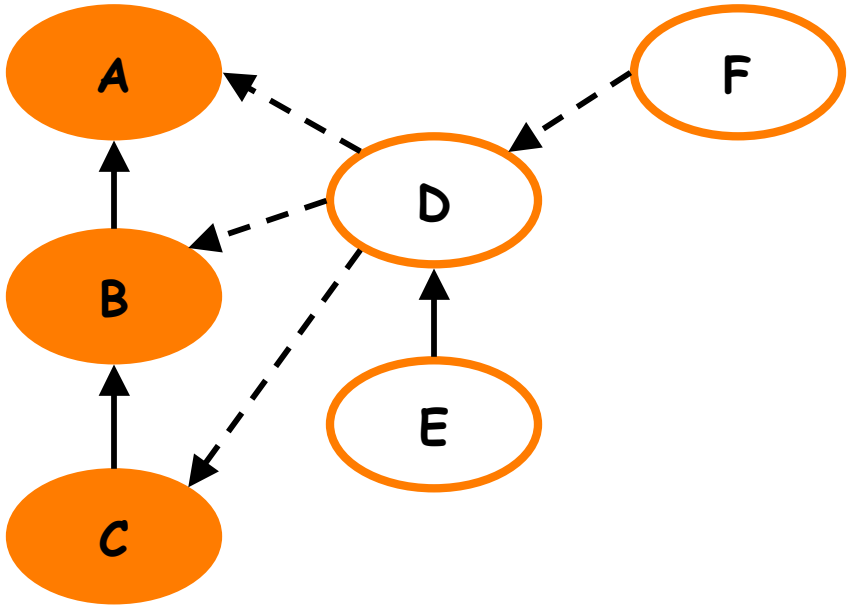


Class-Level Analysis

P / P'

```
class A {
  void foo() {…} }
class B extends A {
  void foo() {… }
}
class C extends B {}
class D {
void bar() {
    A ref=null;
    switch(somevar) {
      case '1': ref=new A(); break;
      case '2': ref=new B(); break;
      case '3': ref=new C(); break; }
   ref.foo();
} }
class E extends D {}
class F {
  void bar(D d) {…}}
```

Interclass Relation Graph (for P an P')

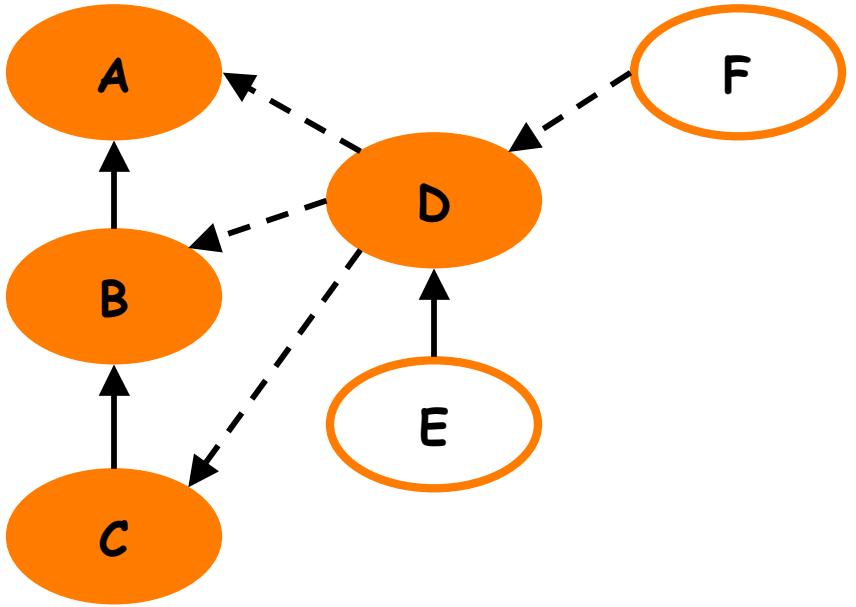


Class-Level Analysis

P / P'

```
class A {
  void foo() {…} }
class B extends A {
  void foo() {… }
}
class C extends B {}
class D {
void bar() {
    A ref=null;
    switch(somevar) {
     case '1': ref=new A(); break;
     case '2': ref=new B(); break;
     case '3': ref=new C(); break; }
   ref.foo();
} }
class E extends D {}
class F {
  void bar(D d) {…}}
```

Interclass Relation Graph (for P an P')



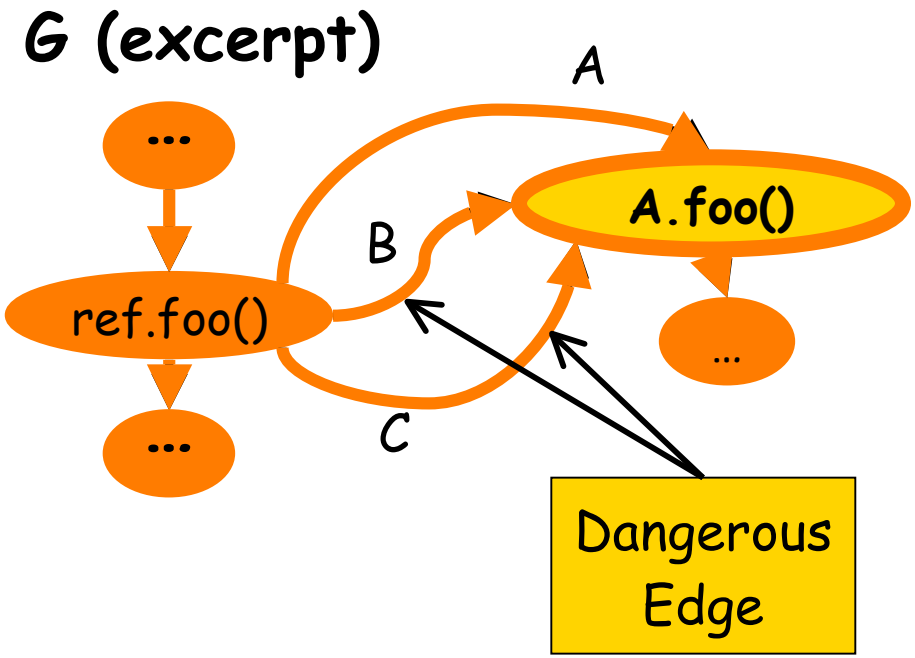
Example: Stmt-Level Analysis

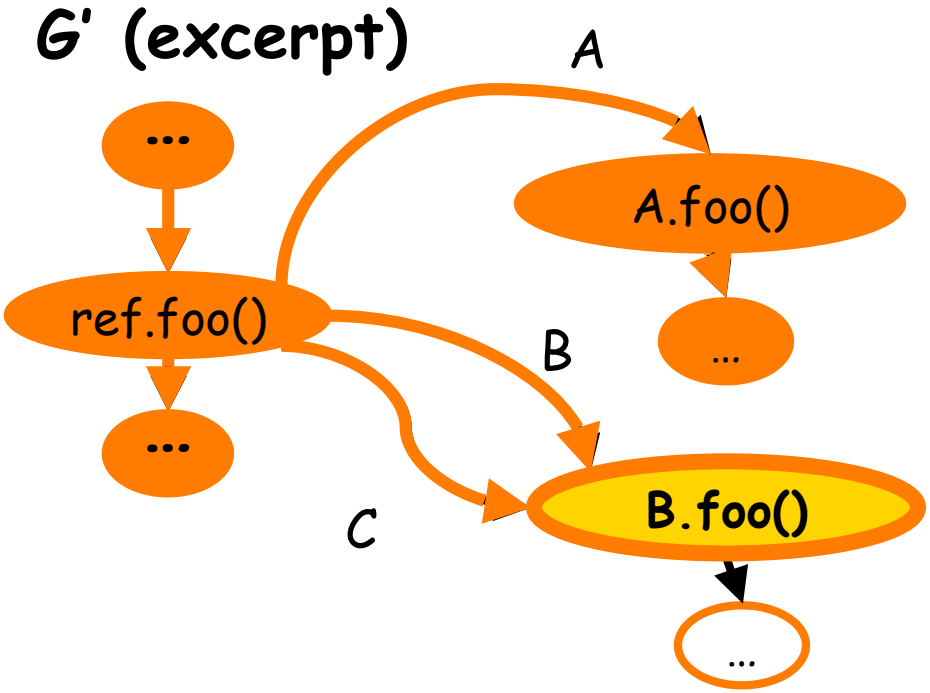
Subset of P

```
class A
class B {…}
class C
class D {
  void bar() {…; ref.foo(); …}
} }
```

Subset of P'

```
class A
class B {… void foo() {…} … }
class C
class D {
  void bar() {…; ref.foo(); …}
} }
```





Example: Stmt-Level Analysis

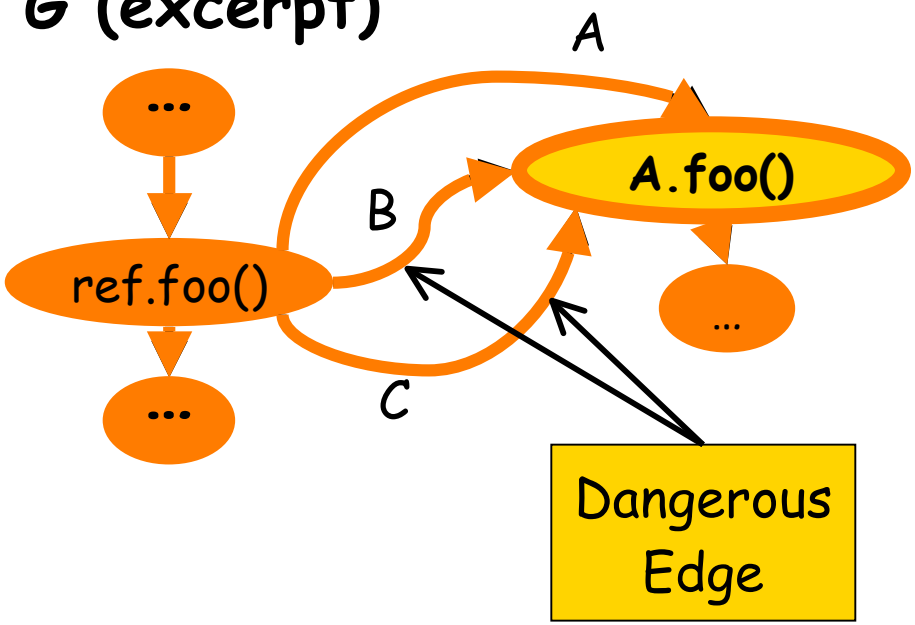
Subset of P

```
class A
class B {…}
class C
class D {
  void bar() {…; ref.foo(); …}
} }
```

Subset of P'

```
class A
class B {… void foo() {…} … }
class C
class D {
  void bar() {…; ref.foo(); …}
} }
```

G (excerpt)



Test cases to be rerun:

Test cases in T that execute the call node with *ref*'s dynamic type being *B* or *C*

Outline

- Background
- Technique
- Empirical Evaluation
- Conclusion

Empirical Setup

- Tool

  DejaVOO

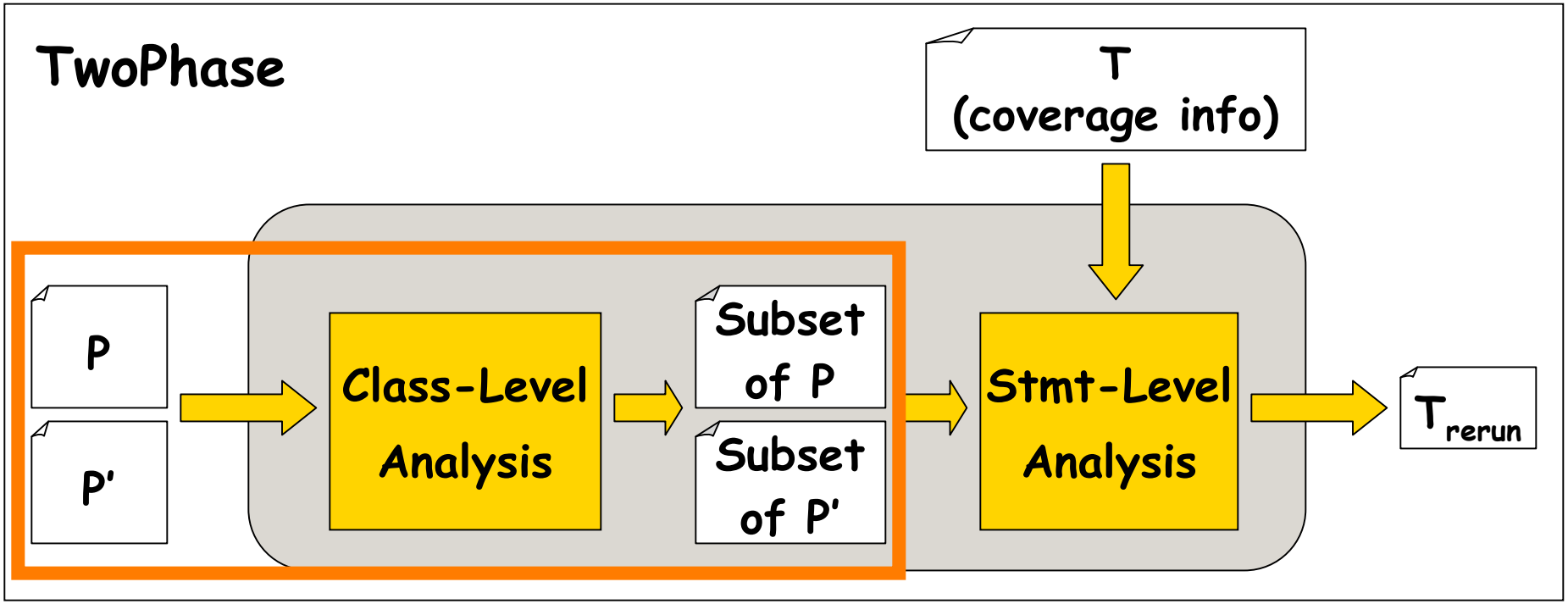
- Subjects

Program	Versions	Classes	KLOC	Test Cases	Retest Time
Jaba	5	525	70	707	54 min
Daikon	5	824	167	200	74 min
Jboss	5	2,403	1,000	639	32 min

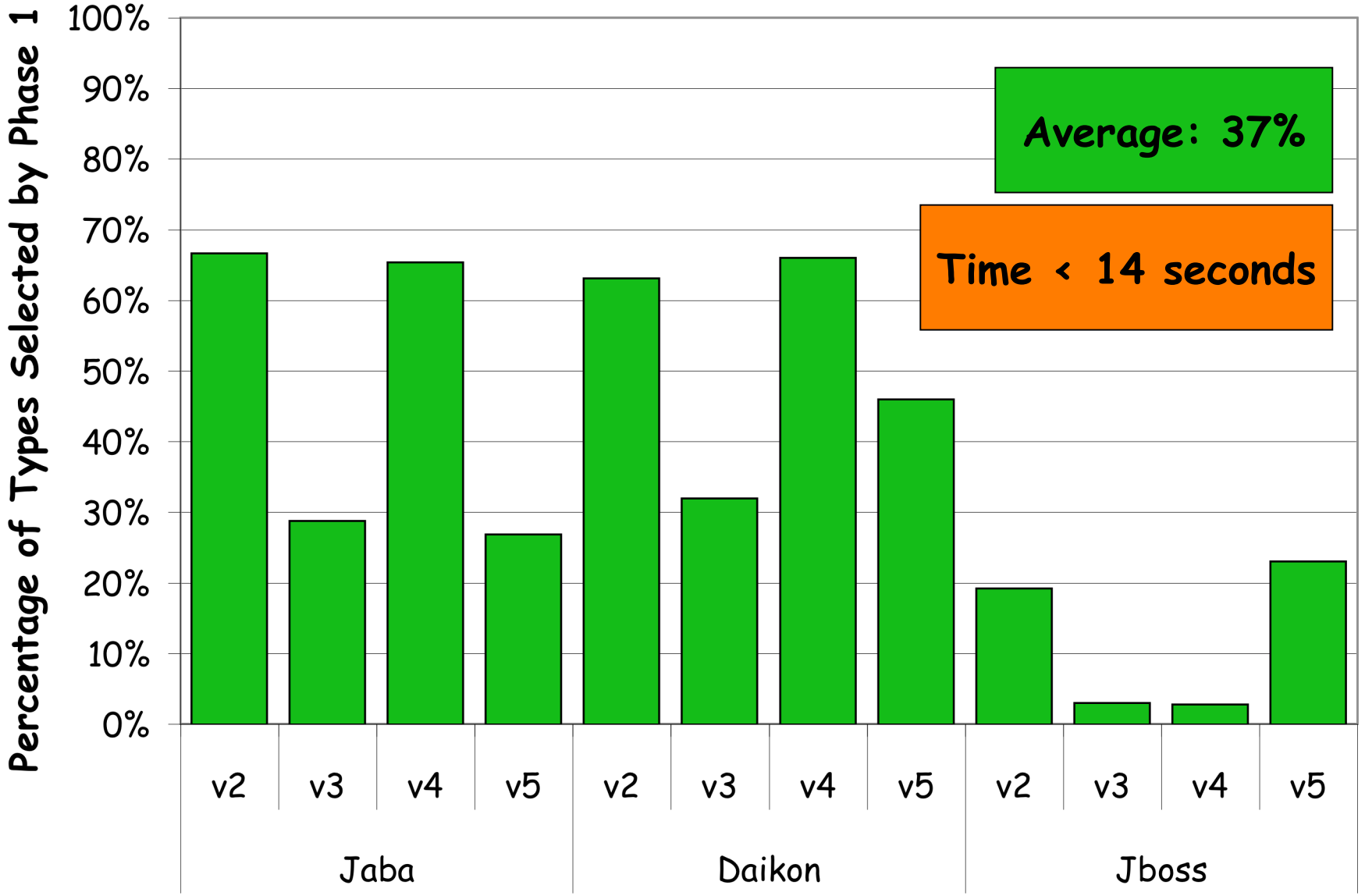
- Three empirical studies

  1. **Effectiveness of Phase 1**
  2. Precision gain of Phase 2
  3. **Overall savings in retesting time**

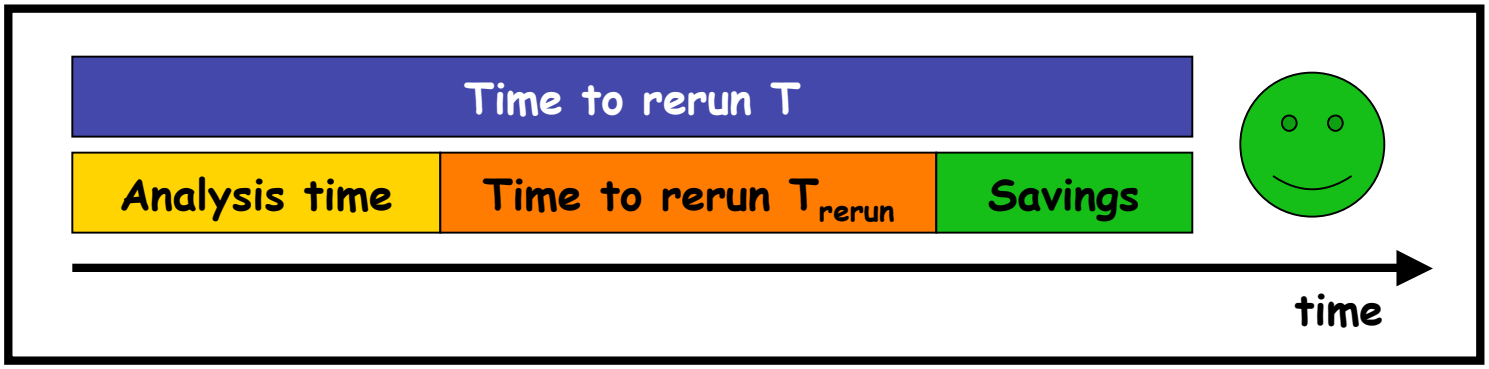
RQ1: Effectiveness of Phase 1



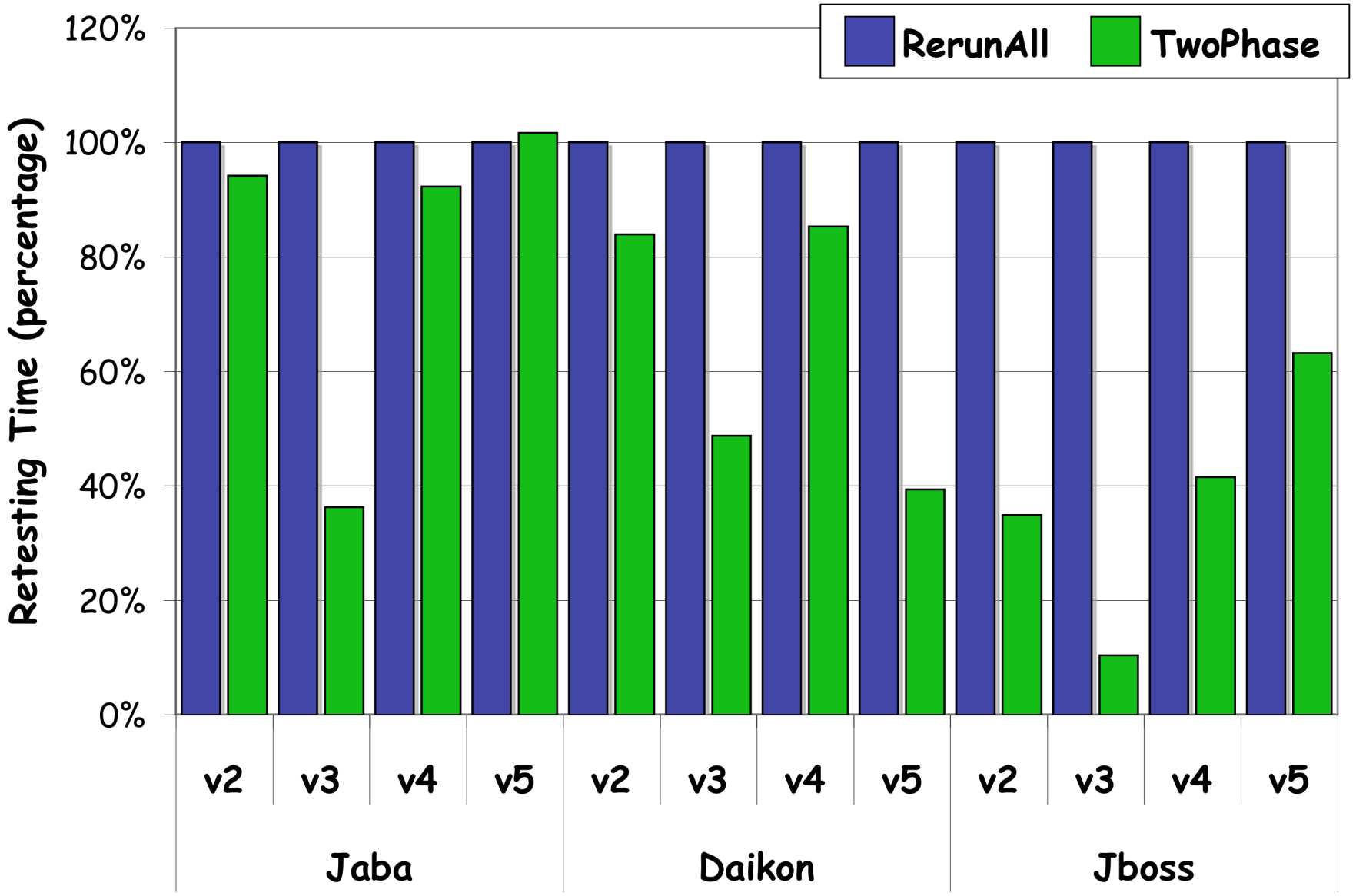
RQ1: Effectiveness of Phase 1



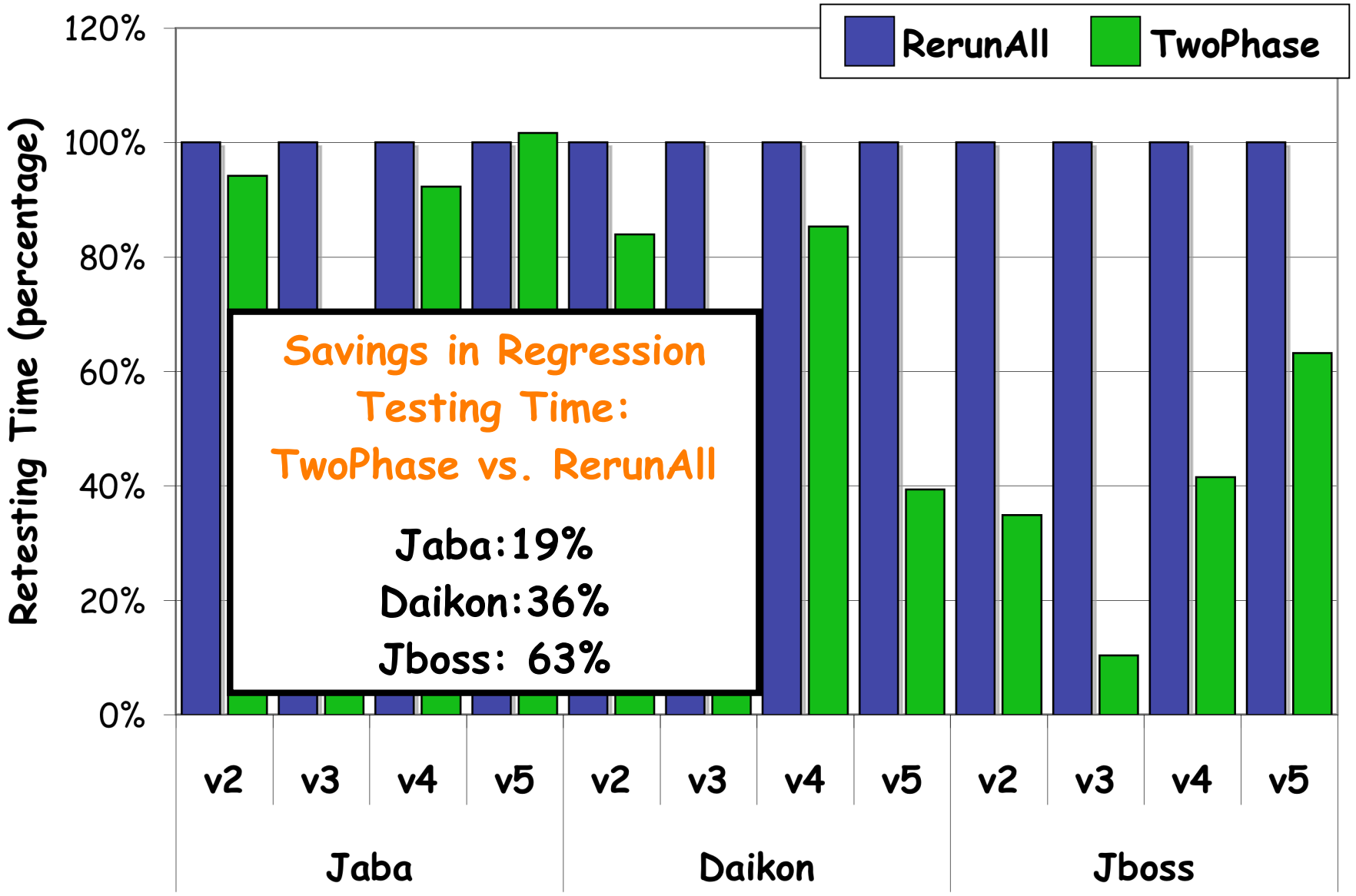
RQ3: Overall Savings in retesting Time



RQ3: Overall Savings in Retesting Time



RQ3: Overall Savings in Retesting Time



Outline

- Background
- Technique
- Empirical Evaluation
- Conclusion

Contribution

- Extended our existing regression-testing technique
  - Considerable increase in efficiency
  - Same precision
- First RTS technique that, at the same time:
  - is safe
  - combines coarse and fine-grained analysis
  - handles the features of the Java language
- Tool (DejaVOO) that implements the technique
- Empirical results showing the effectiveness of the technique

Current and Future Work

- Further investigate cost-effectiveness of the technique
  - Continuous integration
  - Larger systems
- Investigate test-suite augmentation
  - Based on changes (program differencing)
  - Based on field-data (impact analysis)

Questions?