Using Mutation Analysis for Assessing and Comparing Test Coverage Criteria

IEEE Transactions on Software Engineering

(TSE 2006)

James H. Andrews, Lionel C. Briand, Yvan Labiche, and Akbar Siami Namin

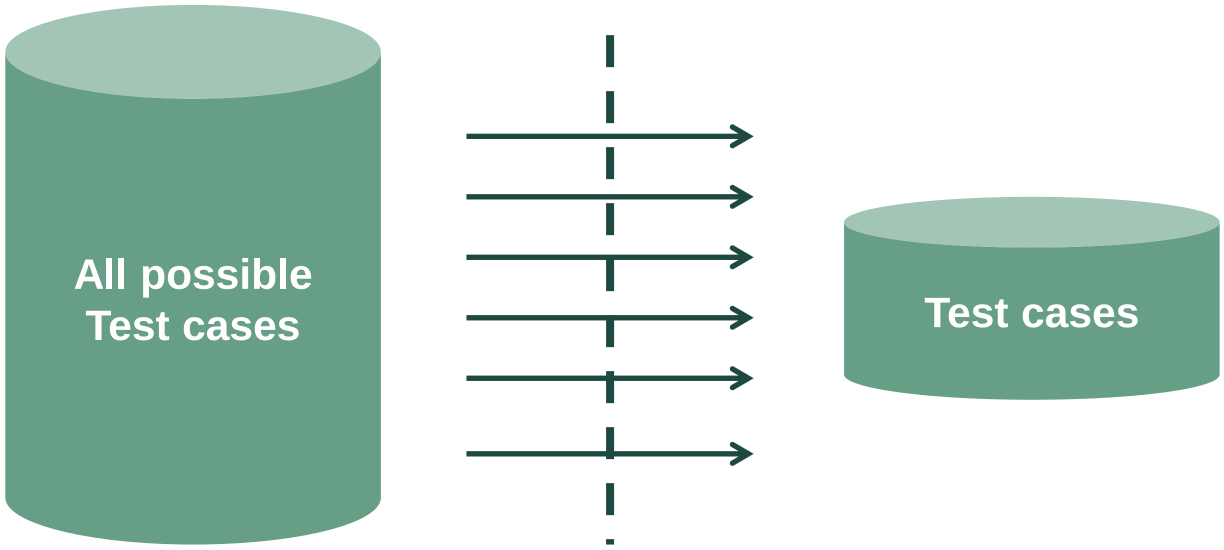
2013-01-10

Dongwon Seo

Contents

Introduction (1/6)

- Primary purpose of software testing is to detect software faults as much as possible.
- In order to detect many faults, many good test cases are needed.



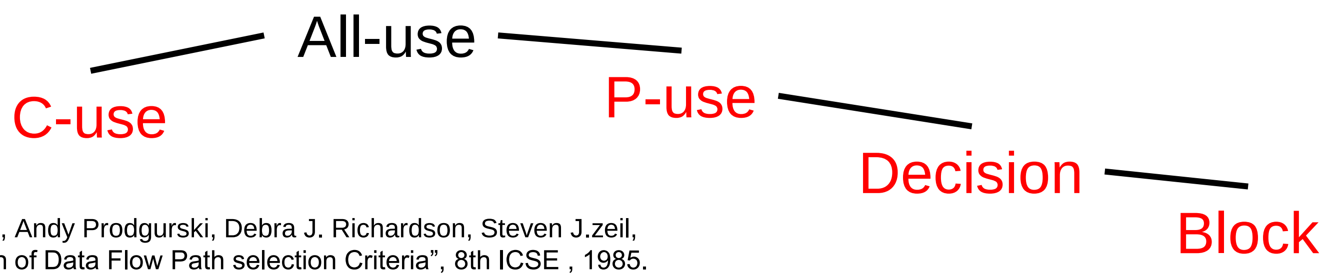
* What are the criteria of good test cases to detect many faults?

Introduction (2/6)

- Code coverage
  - It is one of the testing techniques to distinguish whether test cases is good.
  - It describes the degree (coverage level) to which the source code of a program has been tested by test cases.
  - It assumes that if coverage level is high, fault detection probability is also high.

Introduction (3/6)

- Control flow code coverage
  - Block coverage ~~is composed of statement.~~
  - Decision coverage is ~~composed~~ of branch statement.
- Data flow code coverage
  - C-use coverage is composed of variable defined and computational expression.
  - P-use coverage is composed of variable defined and conditional expression.
- Subsumption relationship*



* Lori A. Clarke, Andy Prodgurski, Debra J. Richardson, Steven J.zeil, "A comparison of Data Flow Path selection Criteria", 8th ICSE , 1985.

Introduction (4/6)

- Real programs of appropriate size with real faults are hard to find and hard to prepare.
- We have to make faulty version.

Problem to make faulty version

Manually

Realistic
But subjective
And too small

Automatically

Systematic
Repeatable
But unrealistic

Introduction (5/6)

- Mutation Analysis
  - Well defined fault-seeding process.
  - Imitate programmer's mistakes (actual faults)
  - Generate automatically and systematically variant (mutant) as the result of applying an operator to the original code.

```
if (a && b) {
        c = 1;
} else {
        c = 0;
}
```

Original code

Applying mutation operator →

```
if (a || b) {
        c = 1;
} else {
        c = 0;
}
```

Mutant

Introduction (6/6)

- Motivation
  - Hand seeded and real faulty version is not enough to compare coverage criteria.
  - Generated mutants can be used as faulty versions.
  - Our analysis process can be used to compare testing coverage criteria more systematically than previous studies.
- Goal
  - Providing more systematic analysis to compare testing techniques
  - Providing analysis results comparing to coverage criteria

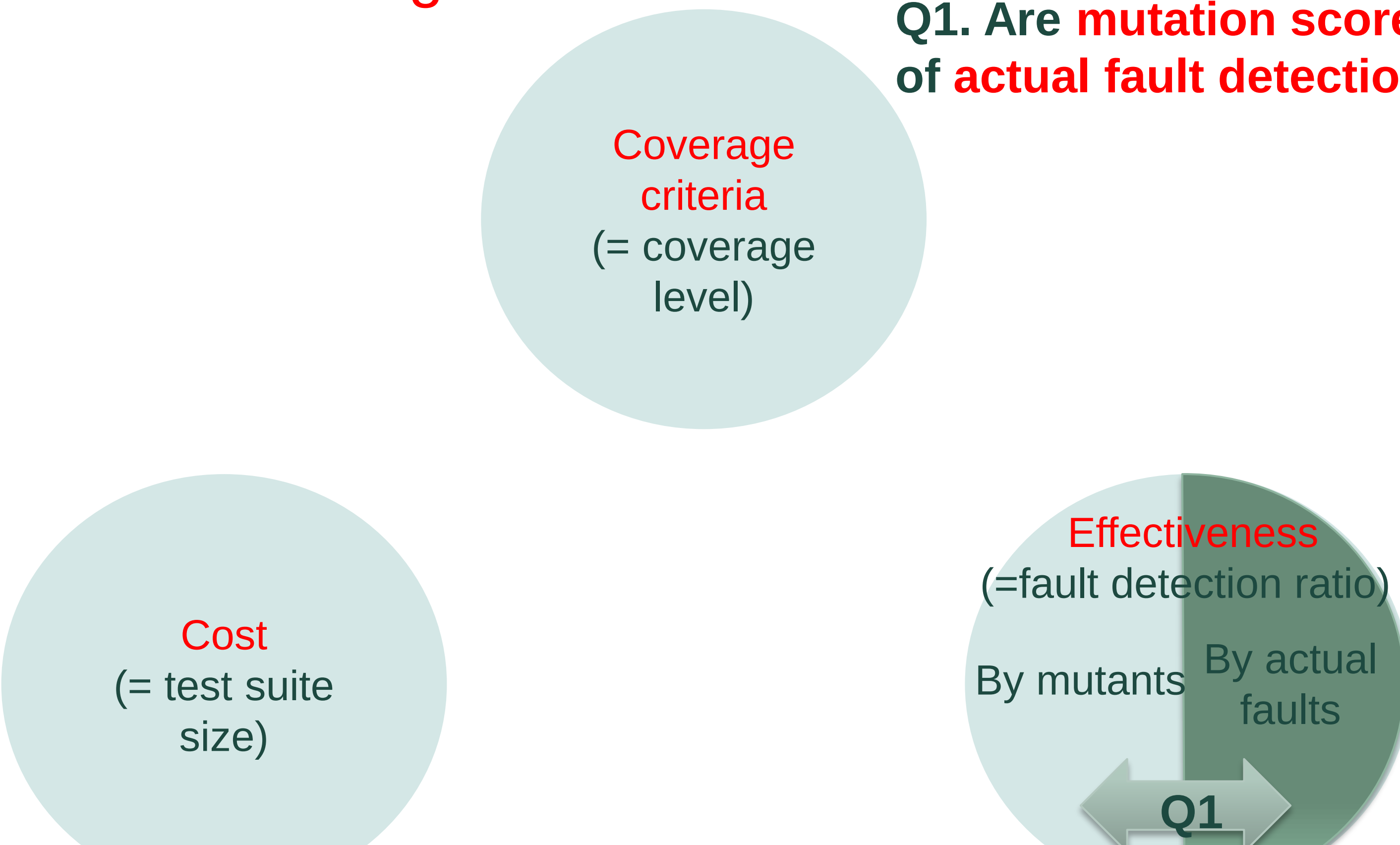
Related work

		Hutchins (ICSE 1994)	Frankl and Iakounenko (ACM SIGSOFT 1998)	This work (TSE 2006)
Goal	Comparing to coverage criteria	Comparing to coverage criteria	Comparing to coverage criteria	Comparing to coverage criteria
Subject program	33 to 66 LOCs	141 to 512 LOCs	6218 LOCs	6218 LOCs
Faults	7 actual faults	130 hand seeded Faults	33 actual faults	34 actual faults 736 mutants
Analysis	Coverage, Test suite size, Fault detection effectiveness	Coverage, Test suite size, Fault detection effectiveness	Coverage, Fault detection effectiveness	Coverage, Test suite size, Fault detection effectiveness

Experimental Description (1/5)

- Experiments design
  - We investigate the relative cost and effectiveness of the coverage criteria.

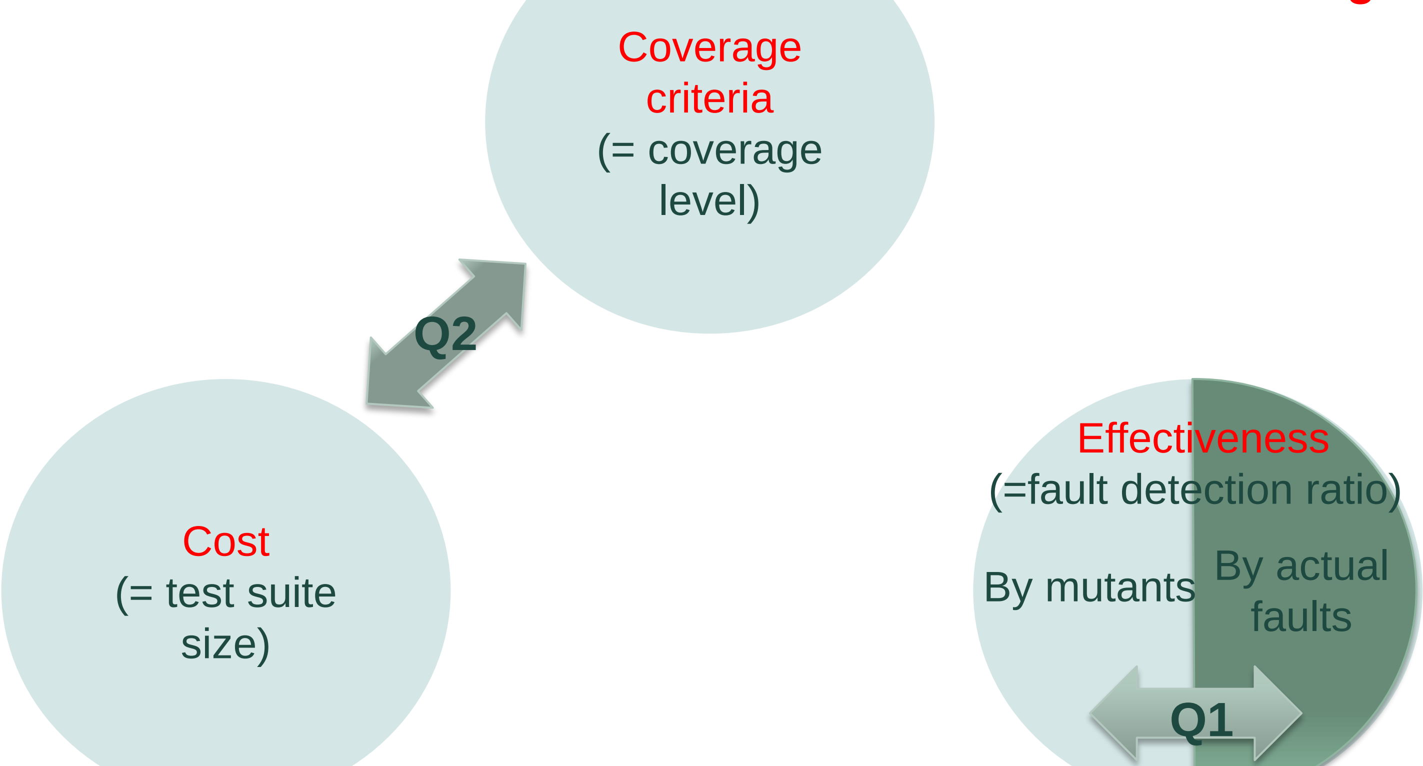
Q1. Are mutation scores good predictors of actual fault detection rates?



Experimental Description (1/5)

- Experiments design
  - We investigate the relative cost and effectiveness of the coverage criteria.

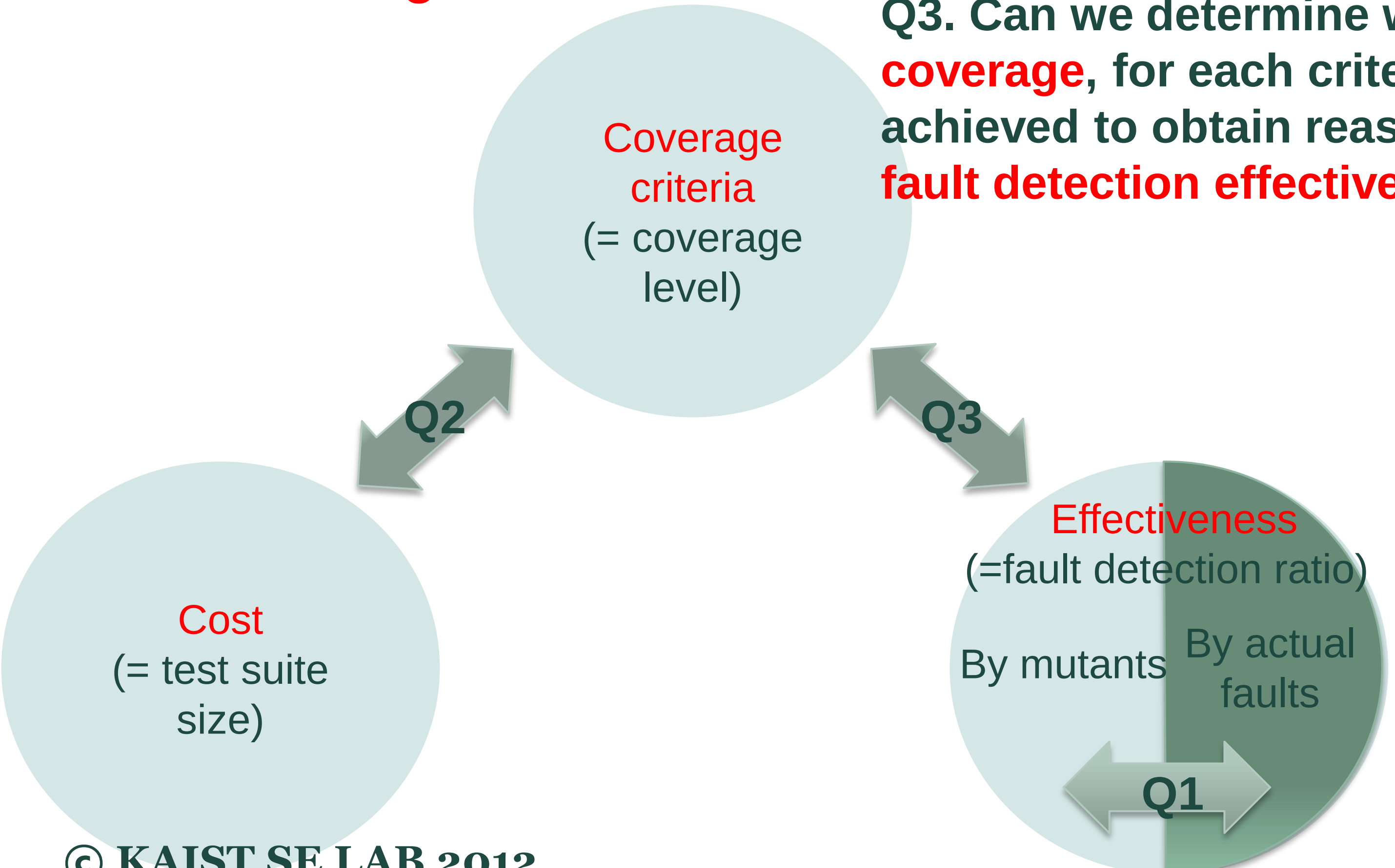
Q2. What is the cost of achieving given levels of the coverage?



Experimental Description (1/5)

- Experiments design
  - We investigate the relative cost and effectiveness of the coverage criteria.

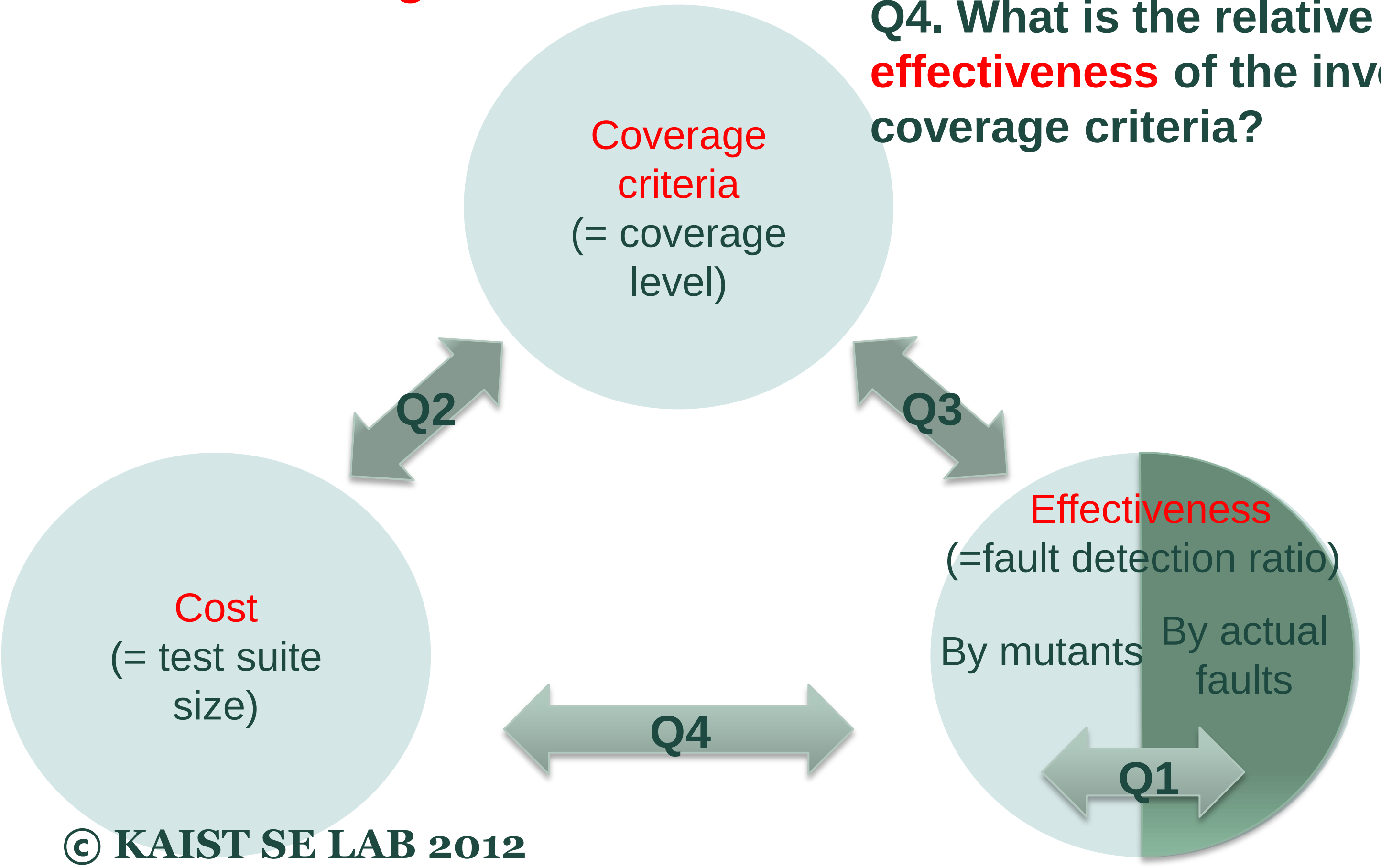
Q3. Can we determine what levels of coverage, for each criteria, should be achieved to obtain reasonable levels of fault detection effectiveness?



Experimental Description (1/5)

- Experiments design
  - We investigate the relative cost and effectiveness of the coverage criteria.

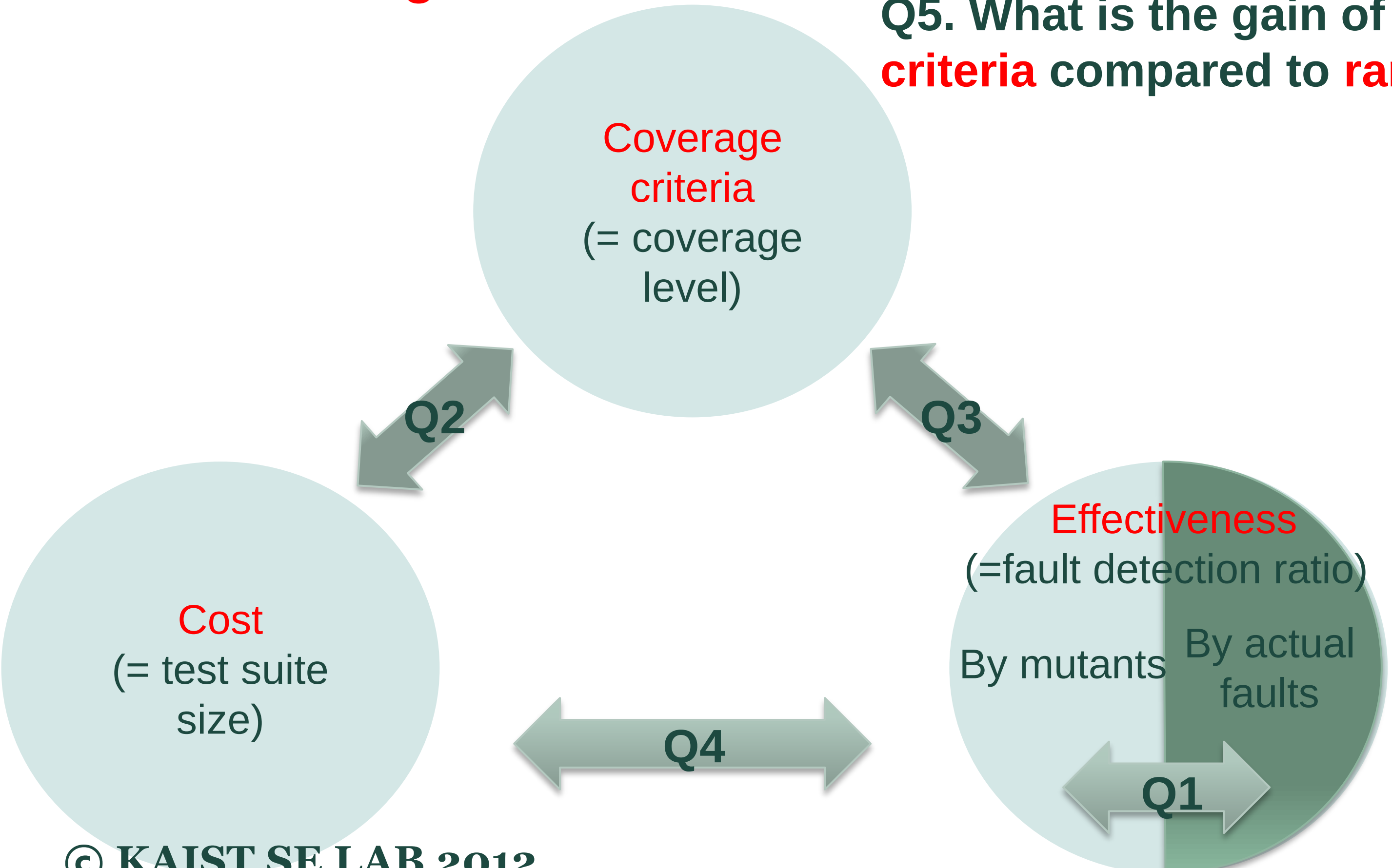
Q4. What is the relative cost effectiveness of the investigated coverage criteria?



Experimental Description (1/5)

- Experiments design
  - We investigate the relative cost and effectiveness of the coverage criteria.

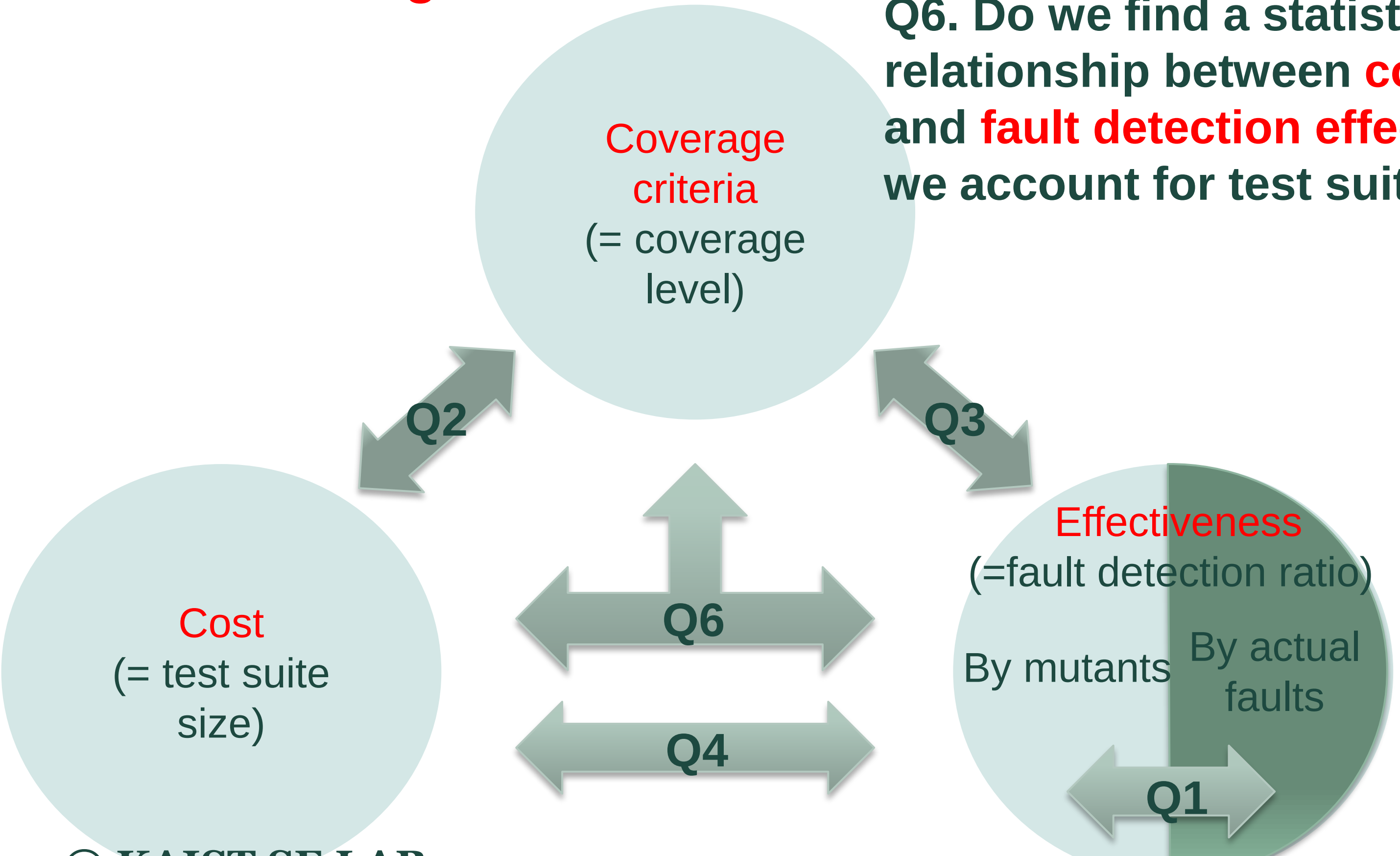
Q5. What is the gain of using coverage criteria compared to random test suites?



Experimental Description (1/5)

- Experiments design
  - We investigate the relative cost and effectiveness of the coverage criteria.

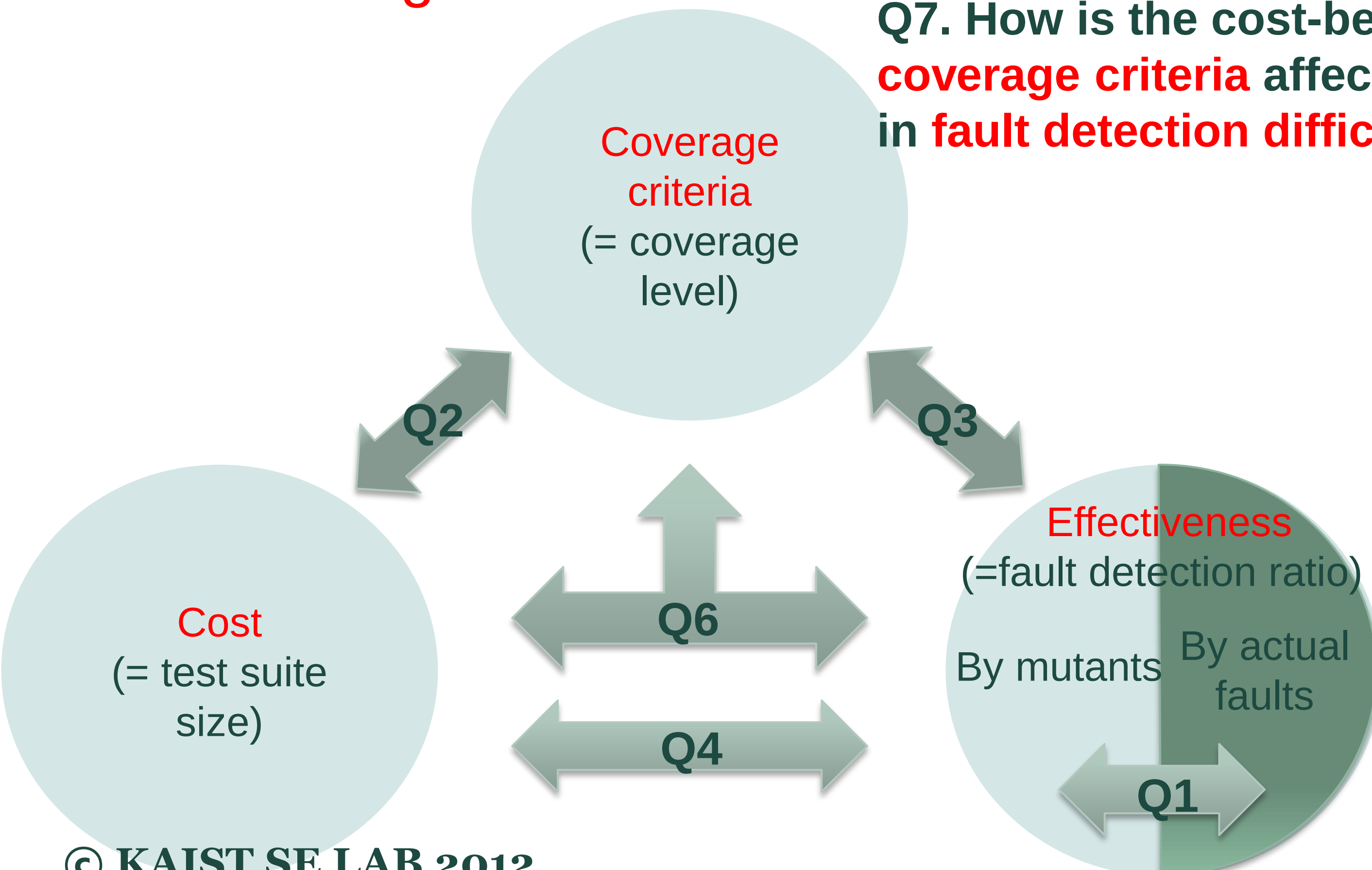
Q6. Do we find a statistically significant relationship between coverage level and fault detection effectiveness when we account for test suite size?



Experimental Description (1/5)

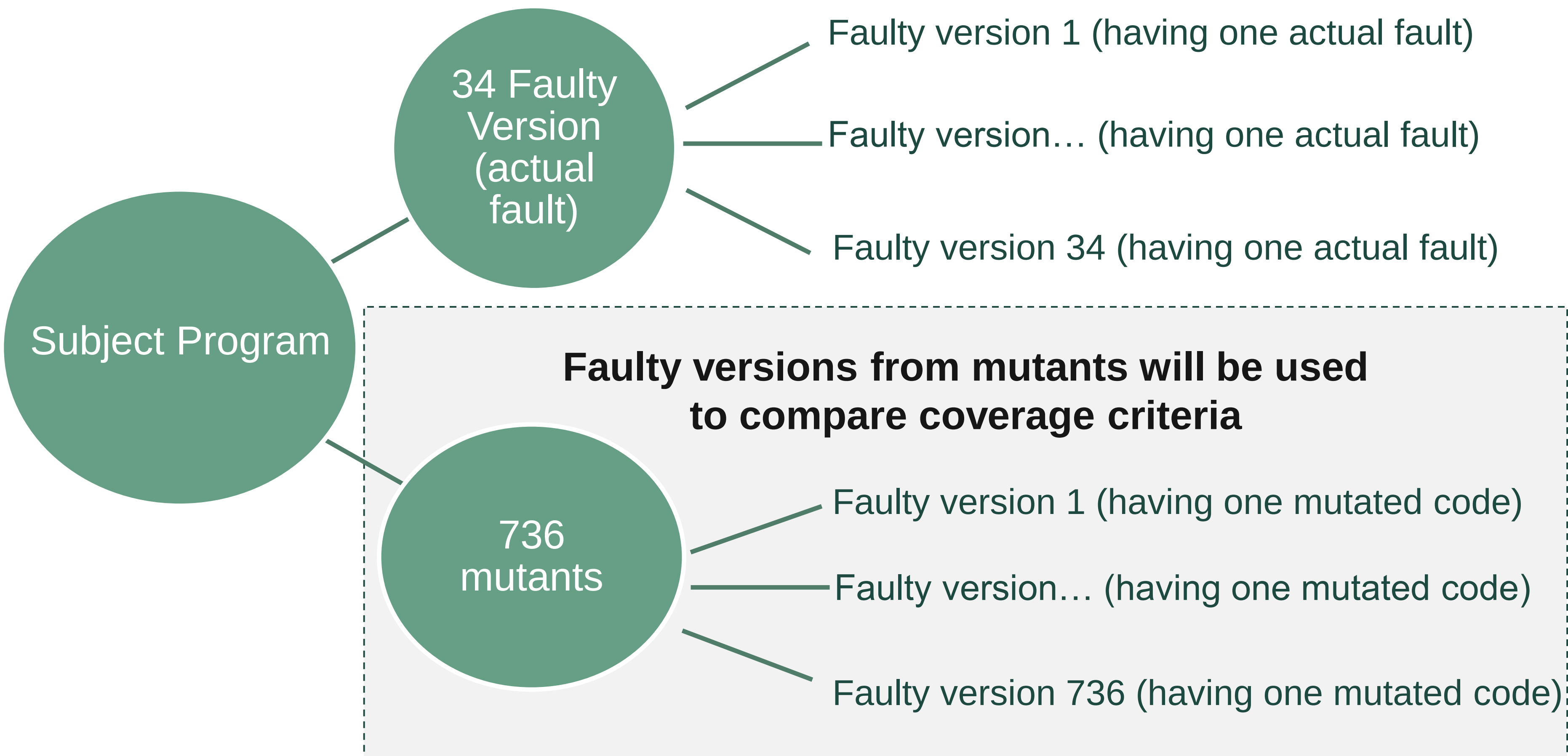
- Experiments design
  - We investigate the relative cost and effectiveness of the coverage criteria.

Q7. How is the cost-benefit analysis of coverage criteria affected by variations in fault detection difficulty?



Experimental Description (2/5)

- Subject programs

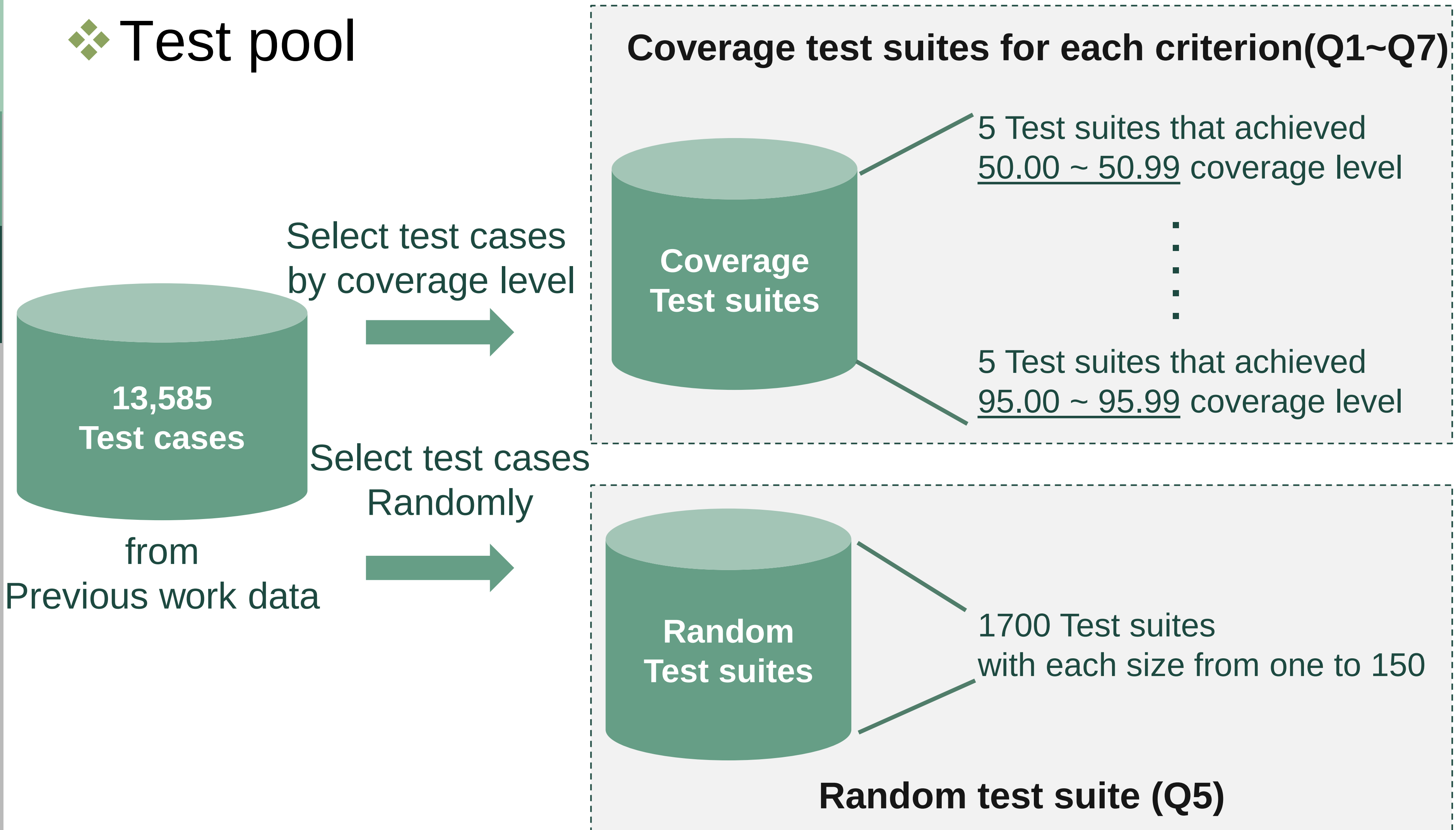


Experimental Description (3/5)

- Mutant generation
  - These four classes of mutation used.
    - Replace an integer constant C by 0, 1, -1, ((C)+1, or ((C)-1).
    - Replace an arithmetic, relational, logical, bitwise logical, increment/decrement, or arithmetic-assignment operator by another operator from the same class.
    - Negate the decision in an if or while statement.
    - Delete a statement.
  - So many mutants (11, 379) generated.
  - We only use 10^{th} mutant generated. (11,379 -> 1138)
  - We removed equivalent mutants (1138 -> 736)
    - The mutants that were not killed by any test case is referred as equivalent mutants

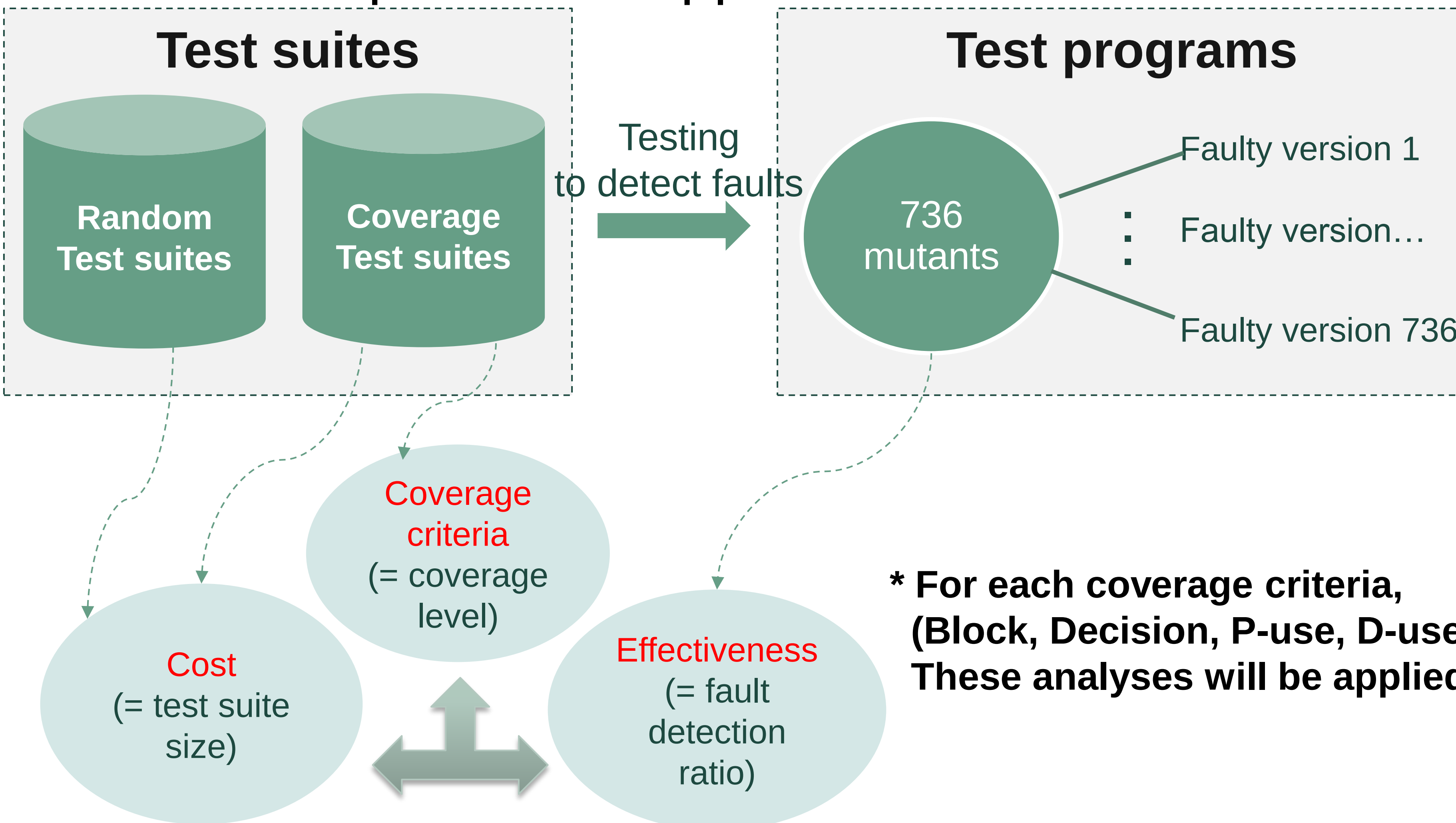
Experimental Description (4/5)

- Test pool



Experimental Description (5/5)

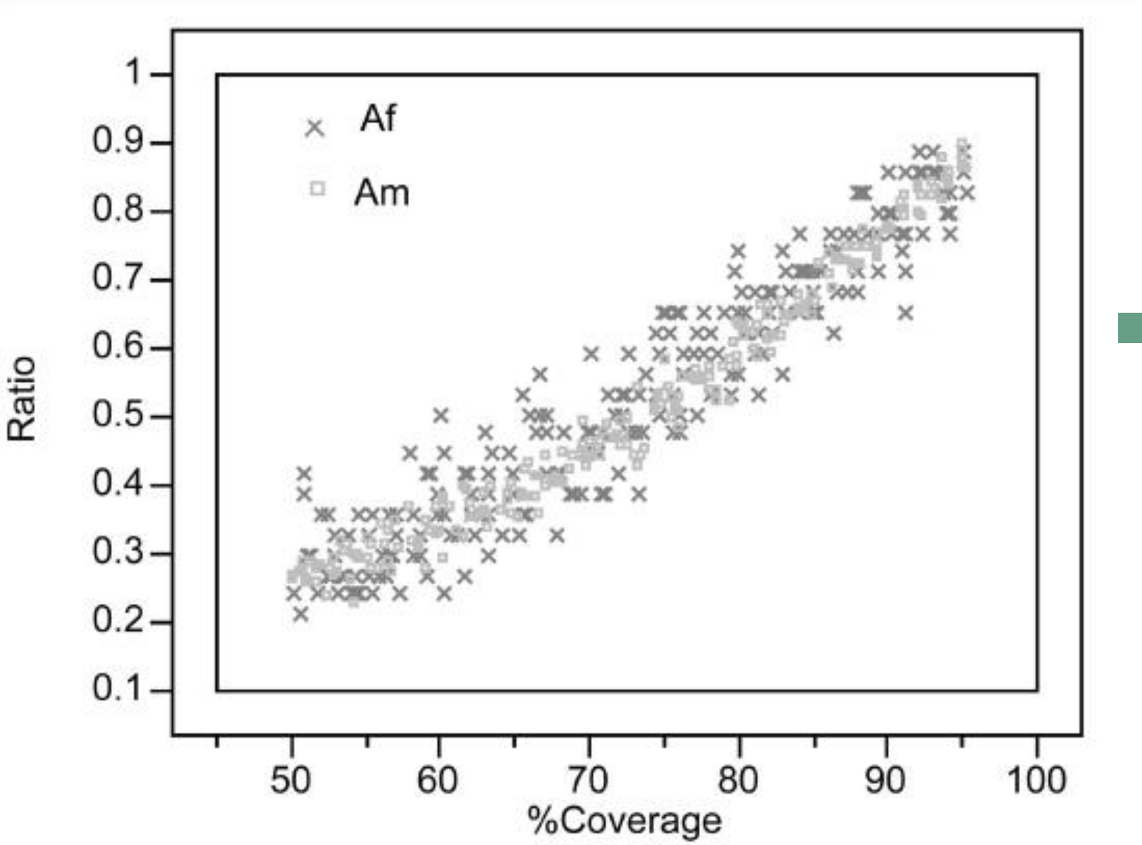
- Overall experiments approach



*** For each coverage criteria, (Block, Decision, P-use, D-use) These analyses will be applied.**

Analysis Results (1/12)

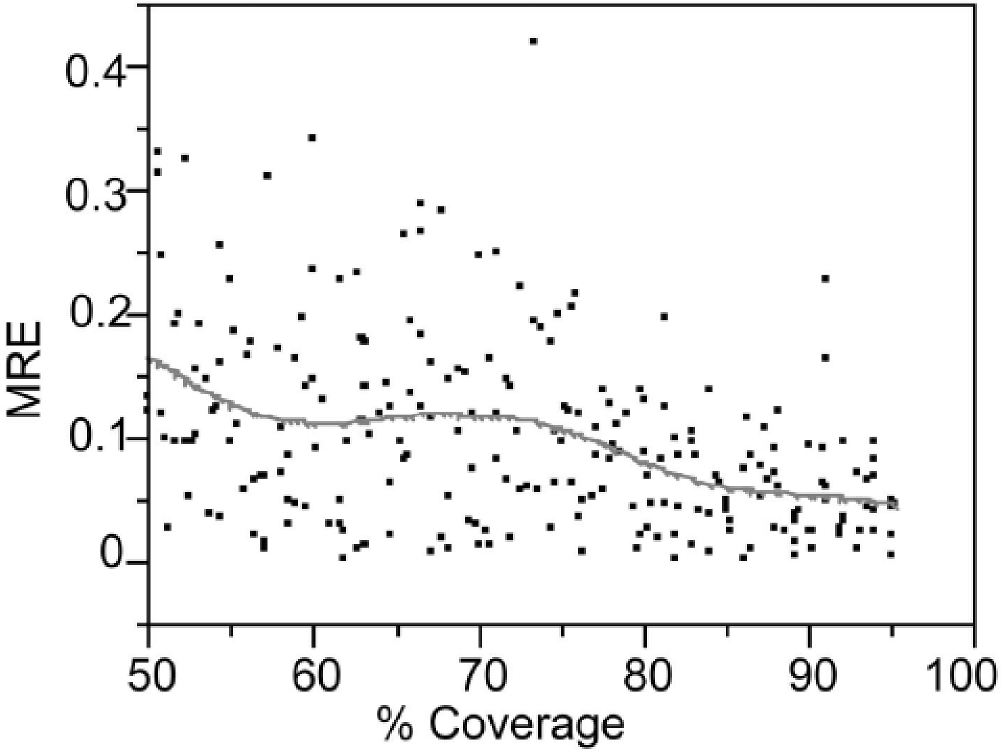
- Q1. Is Am good predictor of Af?
  - Am is mutant detection ratio (=mutant score).
  - Af is actual fault detection ratio



➡ Af and AM is both proportional to coverage level.

Analysis Results (2/12)

- Q1. Is Am good predictor of Af?
  - MRE (Magnitude of Relative Error) is measure for evaluating the accuracy of predictive system.
    - MRE = |Af – Am| / Af

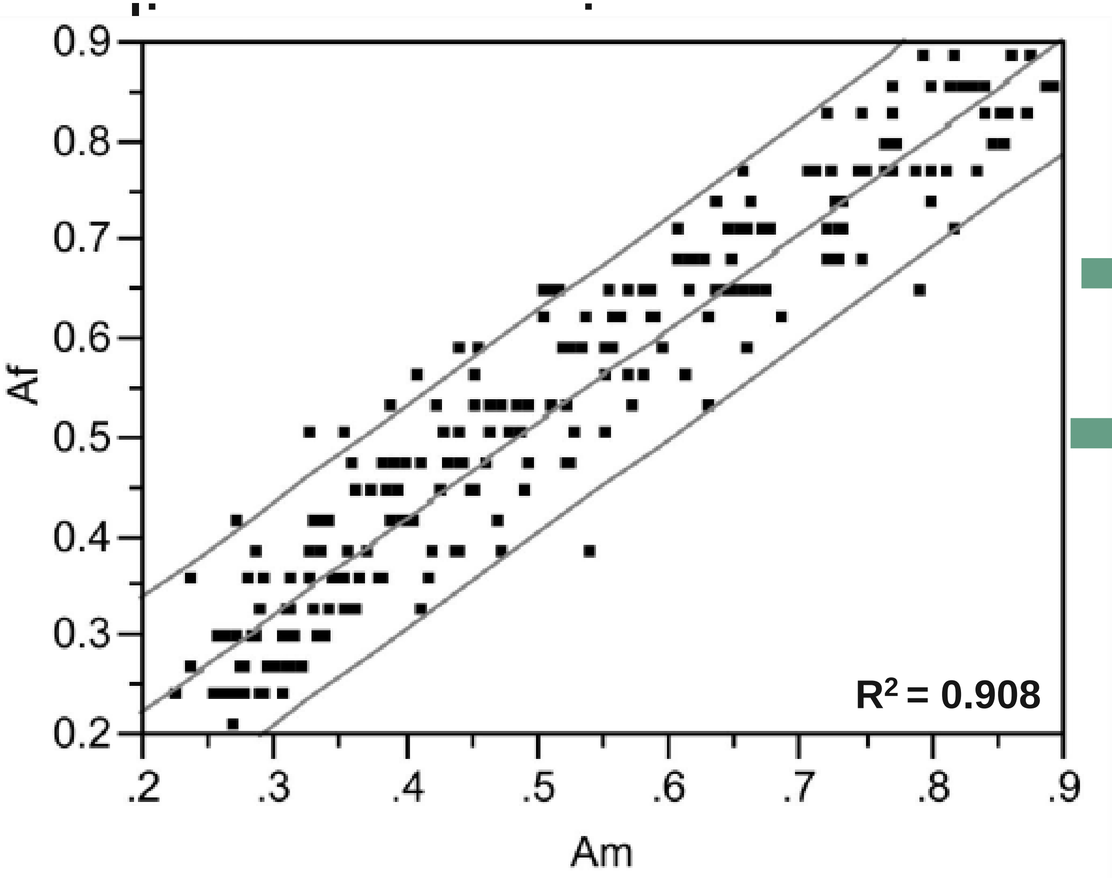


MRE decrease as %coverage increases

The higher %coverage will make the prediction of Af based on Am more accurate

Analysis Results (3/12)

- Q1. Is Am good predictor of Af?
- Modeling a linear regression betweeR^2 of the

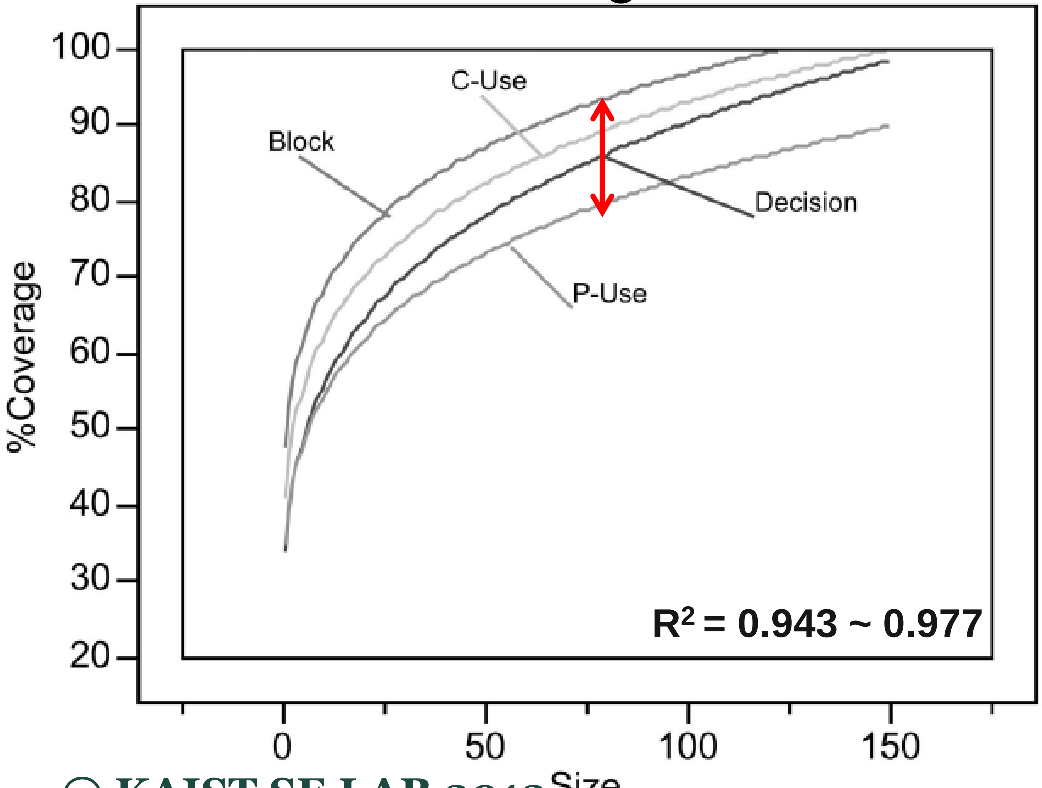


Based on such a model,
Am is a unbiased predictor of Af.

Based on such a model,
Am is a unbiased predictor of Af.

Analysis Results (4/12)

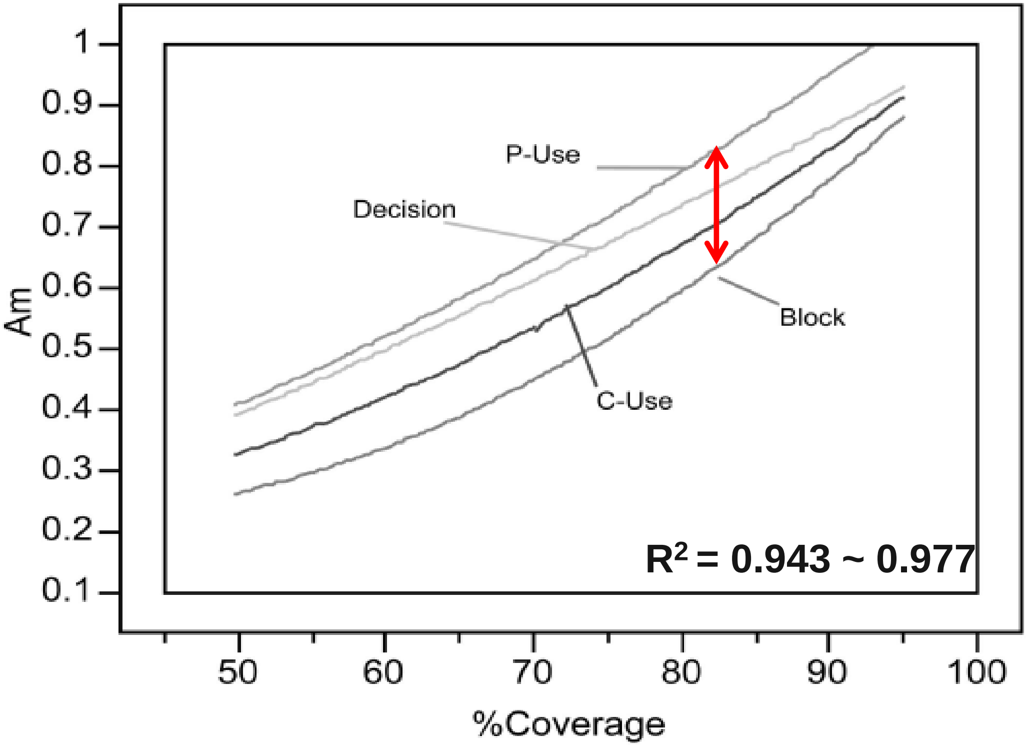
- Q2. What is the cost of achieving given % coverage criteria?
  - Modeling exponential regression between Size and %Coverage



Cost of achieving given level of coverage criteria : Block < C-Use < Decision < P-use

Analysis Results (5/12)

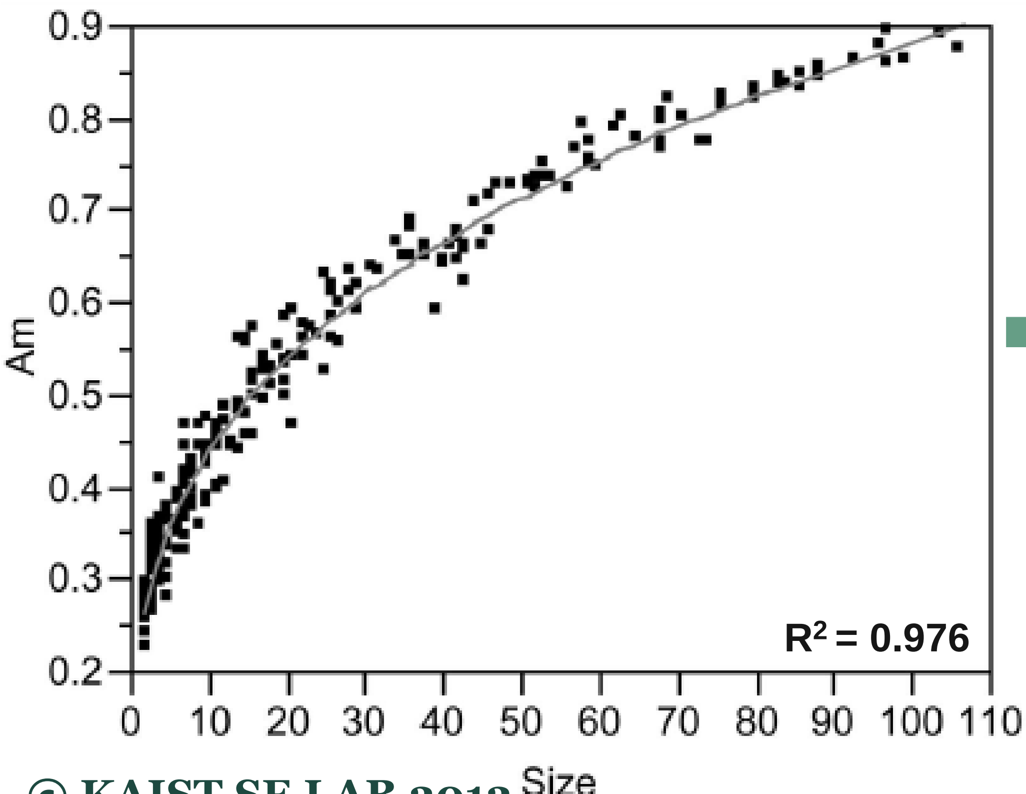
- Q3. Can we determine what % coverage criteria should be achieved to obtain reasonable Am?
  - Modeling exponential regression between %coverage and AF



Am of achieving given level of coverage criteria :
Block < C-Use < Decision < P-use

Analysis Results (6/12)

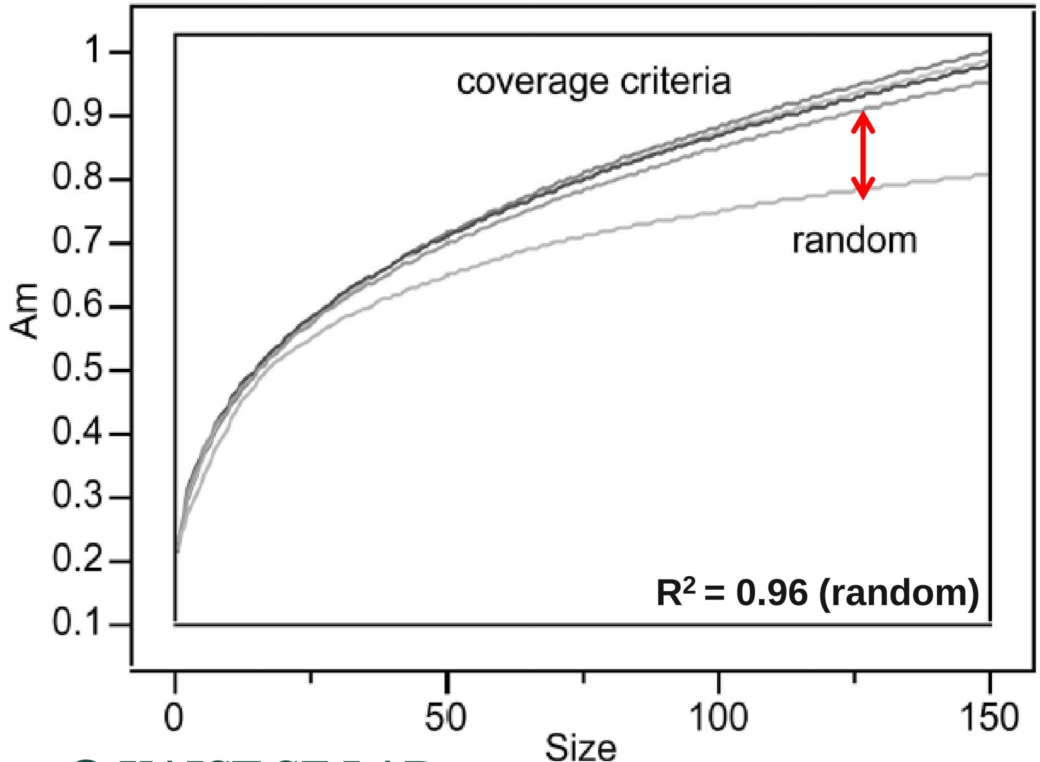
- Q4. What is the relative cost effectiveness of the investigated control and data flow coverage criteria?
  - Modeling exponential regression between size and Am



None of the four criteria is more cost-effective than the others.

Analysis Results (7/12)

- Q5. What is the gain of using coverage criteria compared to random test suites (=null criterion)?
  - Random test suites are used instead of coverage suites.
  - Random test suite means null criterion.

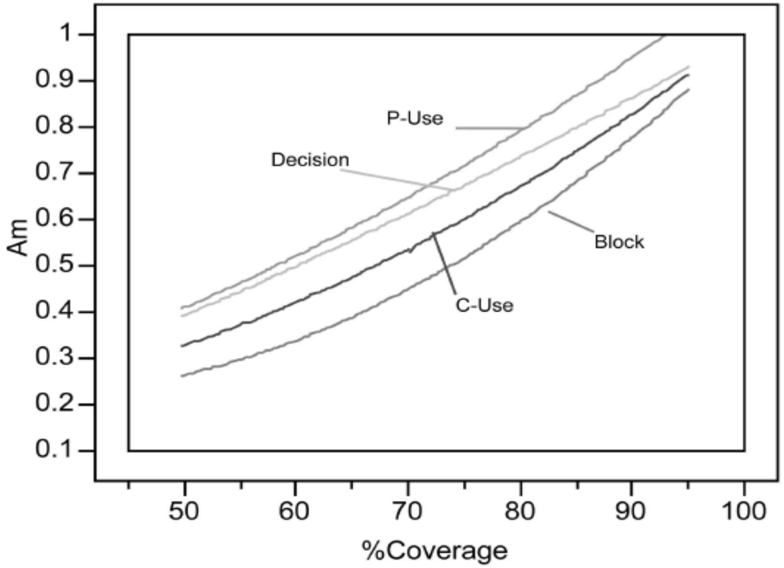


Coverage criteria more cost-effective than null criterion

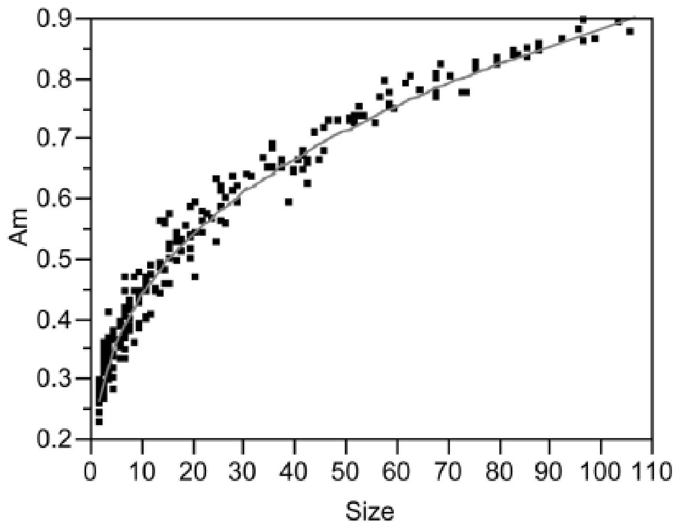
None of the four criteria is more cost-effective than the others.

Analysis Results (8/12)

- Q6. Do we still find a statistically significant relationship between coverage level and fault detection effectiveness when we account for test suite size?
  - Modeling multiple regression between Am and two covariates (%coverage, size)



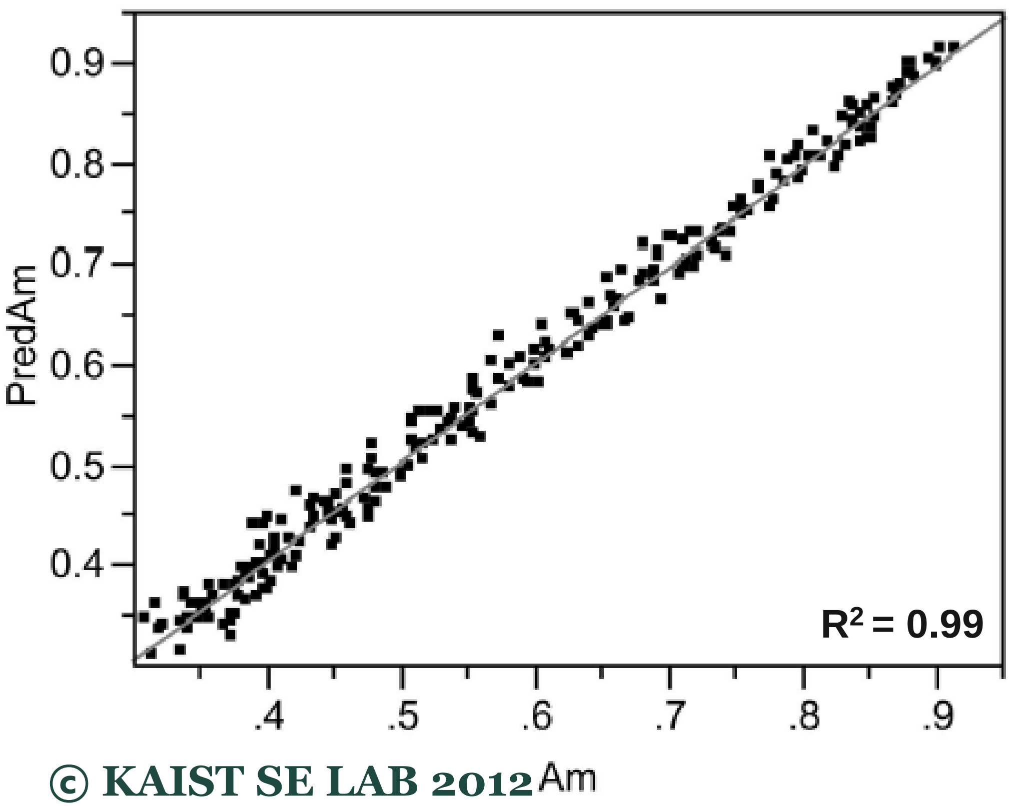
Q3



Q4

Analysis Results (9/12)

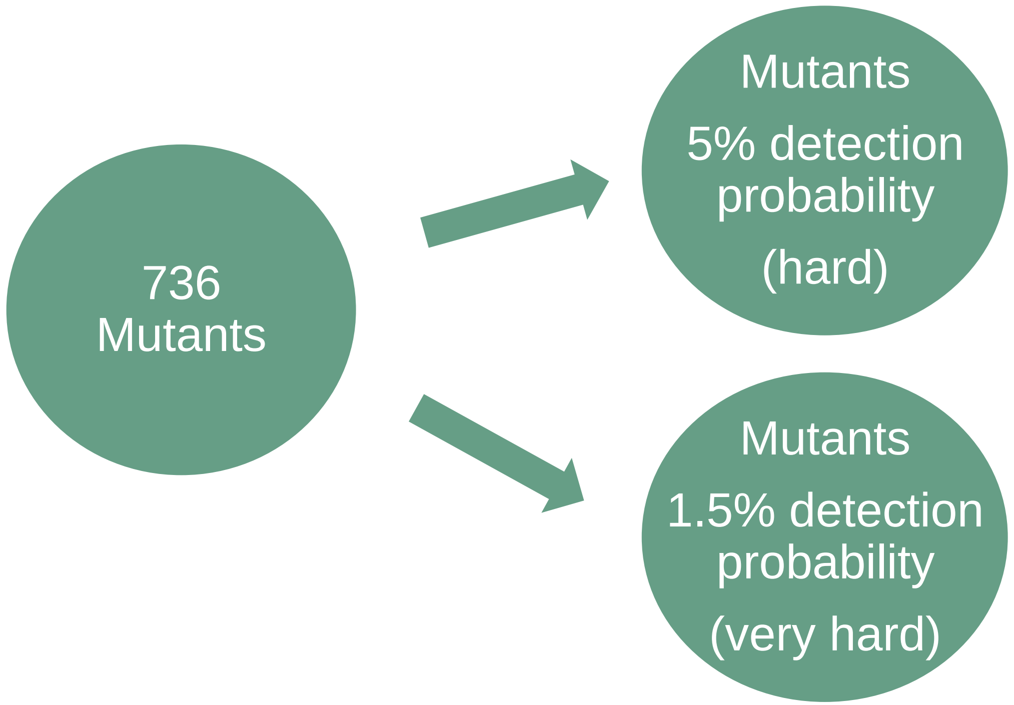
- Q6. Do we still find a statistically significant relationship between coverage level and fault detection effectiveness when we account for test suite size?



- PreAm is Am predictor obtained from between size and %coverage.
- The both size and coverage play a complementary role in explaining fault detection.

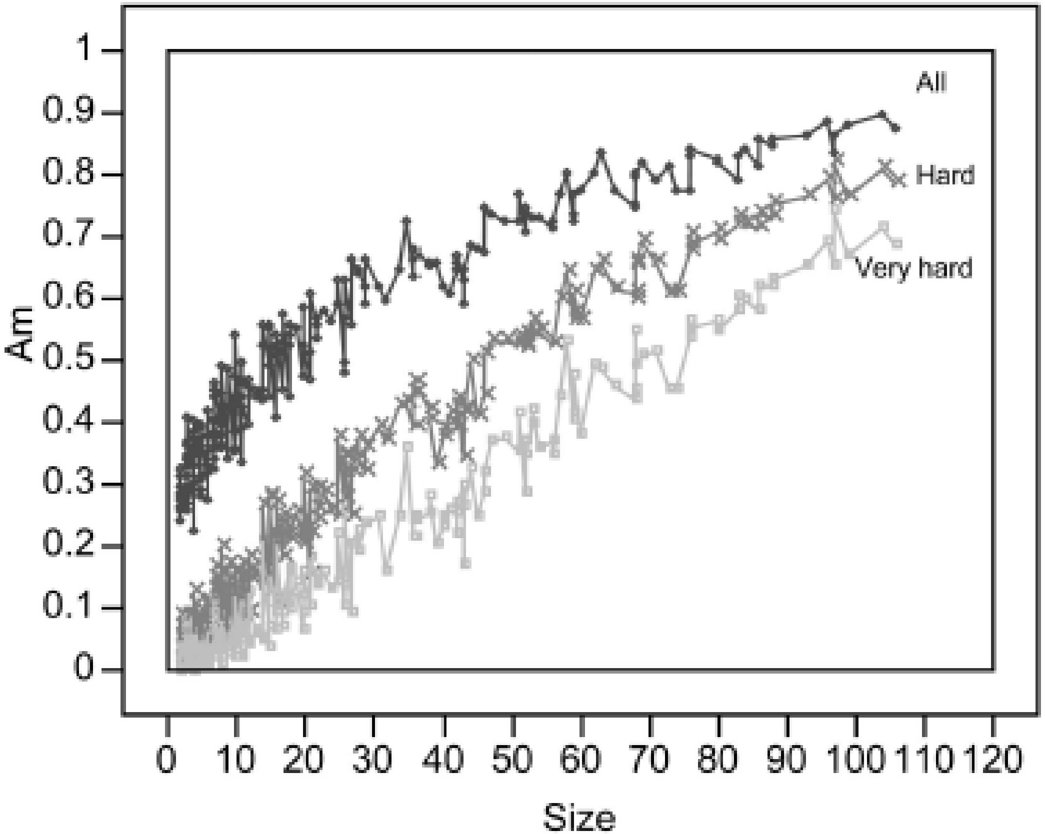
Analysis Results (10/12)

- Q7. How is the cost-benefit analysis of coverage criteria affected by variations in fault detection difficulty?
  - we focus on two subset of mutants.



Analysis Results (11/12)

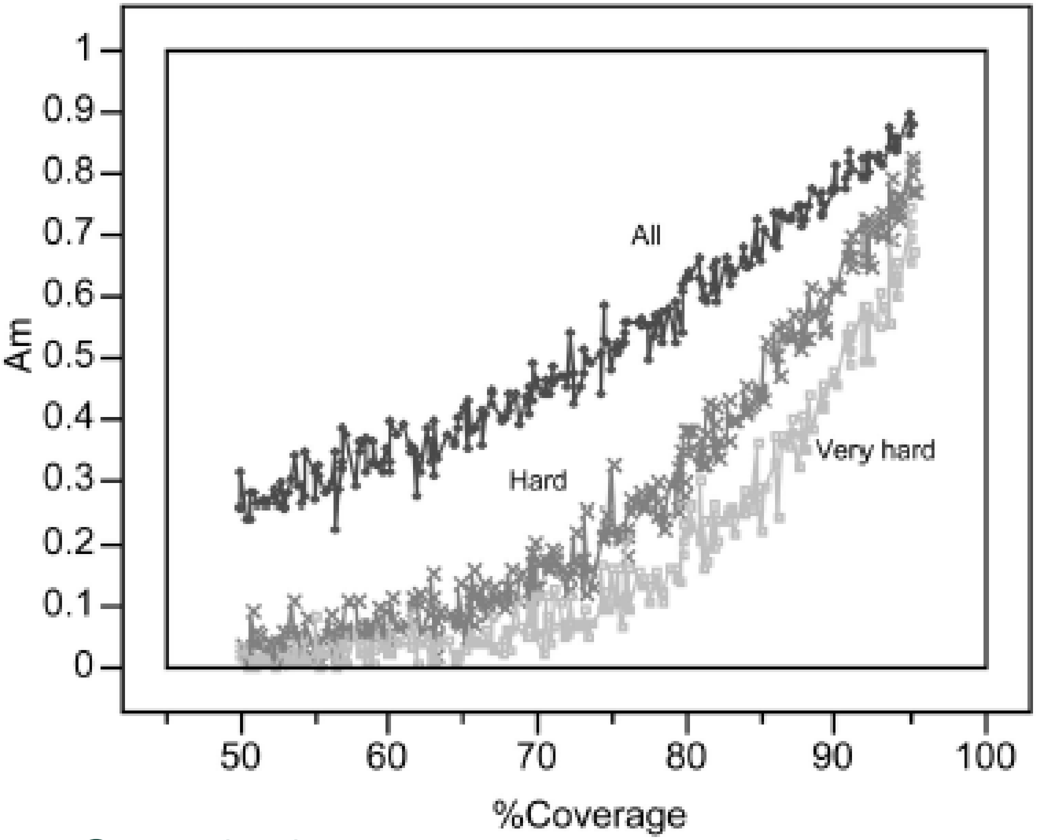
- Q7. How is the cost-benefit analysis of coverage criteria affected by variations in fault detection difficulty?



- Nearly linear relationship in "Hard" mutants
- Previous study reported relationship between Am and size as linear relationship
- But it is because they remove fault to be detect easily through hand seeding

Analysis Results (12/12)

- Q7. How is the cost-benefit analysis of coverage criteria affected by variations in fault detection difficulty?



- In Last 10~20%, All and others have some different exponential relationship.
- Previous study reported relationship between Am and coverage as extreme exponential relationship.
- But it is because they remove fault to be detect easily through hand seeding.

Conclusion

- Contribution
  - Introducing the feasibility of using mutation analysis
  - Applying mutation analysis to fundamental questions regarding the relationships between fault detection, test suite size, and control/data flow coverage.
  - Showing a way to tune the mutation analysis process to possible differences in fault detection probabilities in a specific environment.

Discussion

- Pros
  - It provides detail analysis from many experiments.
  - It provides results of previous studies in order to justify their experiments.

- Cons
  - The number of actual faults is small.
  - It uses only test suite size to asses cost.

Thank You .

Appendix

❖ Control flow code coverage

```
if(c==' ' || c == '\n'
  || c == '\t')
state = OUT;
else if (state == OUT) {
state = IN;
++nw;
}
}
*p_nl = nl;
*p_nw = nw;
*p_nc = nc;
```

Block coverage

```
        state = OUT;
        nl = nw = nc = 0;
          while(EOF != (c = getc(file))){
                ++nc;
          if( c== '\n')
                ++nl;
TRUE→     if(c==' ' || c == '\n' || c == '\t')
                state = OUT;
        else if (state == OUT) {
                state = IN;
                ++nw;
          }
        }

        state = OUT;
        nl = nw = nc = 0;
          while(EOF != (c = getc(file))){
                ++nc;
          if( c== '\n')
                ++nl;
FALSE→    if(c==' ' || c == '\n' || c == '\t')
                state = OUT;
         else if (state == OUT) {
                 state = IN;
                 ++nw;
          }
        }
```

Decision coverage

Appendix

Data flow code coverage

```
        if(c==' ' || c == '\n'
           || c == '\t')
        state = OUT;
        else if (state == OUT) {
        state = IN;
Def→    ++nw;
        }
        }
        *p_nl = nl;
Use→    *p_nw = nw;
        *p_nc = nc;
```

C-Use of variable "nw"

```
Def→  state = OUT;
      nl = nw = nc = 0;
        while(EOF != (c = getc(file))){
           ++nc;
        if( c== '\n')
           ++nl;
        if(c==' ' || c == '\n' || c == '\t')
           state = OUT;
TRUE→   else if (state == OUT) {
           state = IN;
           ++nw;
        }
      }

Def→  state = OUT;
      nl = nw = nc = 0;
        while(EOF != (c = getc(file))){
           ++nc;
        if( c== '\n')
           ++nl;
        if(c==' ' || c == '\n' || c == '\t')
           state = OUT;
FALSE→  else if (state == OUT) {
           state = IN;
           ++nw;
        }
      }
```

P-Use of variable "state"