Information Theory and Software Testing

David Clark

Papers

- *Squeeziness: A Information Theoretic Measure for Avoiding Fault Masking.* D. Clark and R. Hierons. IPL. 2012.
- *Fault Localization Prioritization: Comparing Information Theoretic and Coverage Based Approaches.* S. Yoo, M. Harman and D. Clark. ToSEM. 2013.
- *An Analysis of the Relationship between Conditional Entropy and Failed Error Propagation in Software Testing.* K. Androutsopoulos, D. Clark, H. Dan, R. Hierons, and M. Harman. ICSE. 2014.
- *Information Transformation: An Underpinning Theory for Software Engineering.* D. Clark, R.Feldt, S. Poulding and S. Yoo. ICSE. 2015.
- *Test Set Diameter: Quantifying the Diversity of Sets of Test Cases.* R. Feldt, S. Poulding, D. Clark and S. Yoo. ICST. 2016.
- *Test Oracle Assessment and Improvement.* G. Jahangirova, D. Clark, M. Harman and P. Tonella. ISSTA. 2016.

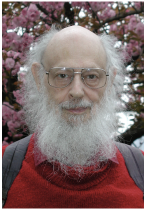
Problems

- What is the test execution order that locates a software fault as quickly as possible?
- How can we choose tests that don't suffer from coincidental correctness?
- How do we know that we have enough tests?
- How do we know that our test suite is sufficiently diverse?
- How can we measure how much a real oracle deviates from an ideal oracle?

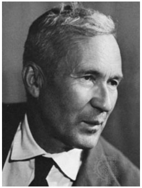
Shannon Entropy

$$\mathcal{H}(X) = -\sum_{x \in X} p(x) \log_2 p(x)$$

randomness of a
random variable



Solomonoff



Kolmogorov

Chaitin

The length of the shortest program that can produce a given string from no inputs

randomness of a string

Use Entropy to speed Fault Location

- Program with m statements, $S = \{s_0, s_1, \ldots, s_{m-1}\}$
- Test suite with n tests, $T = \{t_0, t_1, \ldots, t_{n-1}\}$
- S contains a single fault
- Random variable X models fault locality
- $p(X = s_j)$ is the probability that s_j is the faulty statement
- $\mathcal{H}(X) \longrightarrow 0$ as fast as possible
- Estimate the change in entropy due to each test
- Employ a greedy algorithm to select the next test

Localisation Metrics

- AKA "suspiciousness" metrics: likelihood of statement containing the fault
- Tarantula, Ochiai, Jaccard etc.

Tarantula Metric

$$\tau(s) = \frac{\frac{fail(s)}{totalfail}}{\frac{pass(s)}{totalpass} + \frac{fail(s)}{totalfail}}$$

Tarantula Metric illustration

Structural Elements	Test t_1	Test t_2	Test t_3	Tarantula Metric(τ)	Test t_4	Tarantula Metric(τ)
s_1	●		●	0.00		0.00
s_2	●		●	0.00		0.00
s_3	●		●	0.00		0.00
s_4	●			0.00		0.00
s_5	●		●	0.00		0.00
s_6		●		1.00	●	1.00
s_7 (faulty)		●	●	0.67	●	1.00
s_8		●		1.00	●	1.00
s_9	●	●		0.67	●	0.50
Result	P	F	P	-	F	-

- $B(s_j)$ is the event that s_j is faulty
- $T_i = T_{i-1} \cup \{t_i\}$ is a set of tests
- $\tau(s|T_i)$ is the suspiciousness of s after executing T_i

Tarantula induced Probability Distribution

$$\mathbf{P}_{T_i}(B(s_j)) = \frac{\tau(s_j|T_i)}{\sum_{j=1}^{m} \tau(s_j|T_i)}$$

Tarantula induced Entropy

$$\mathbf{H}_{T_i}(S) = -\sum_{j=1}^{m} \mathbf{P}_{T_i}(B(s_j)) \cdot \log \mathbf{P}_{T_i}(B(s_j))$$

Entropy Lookahead

Lookahead Probability Distribution on Failure

$$\alpha = \mathbf{P}_{T_{i+1}}(F(t_{i+1})) \approx \frac{TF_i}{TP_i + TF_i}$$

Lookahead Probability Distribution on Fault location

$$\begin{aligned}\mathbf{P}_{T_{i+1}}(B(s_j)) = \quad & \mathbf{P}_{T_{i+1}}(B(s_j)|F(t_{i+1})) \cdot \alpha + \\ & \mathbf{P}_{T_{i+1}}(B(s_j)|\neg F(t_{i+1})) \cdot (1-\alpha)\end{aligned}$$

- $F(t_i)$ is the event that t_i is identified as a failing test
- Use $P_{T_{i+1}}(B(s_j))$ to calculate $\mathcal{H}_{T_{i+1}}(S)$, the estimated entropy of B that results from adding the executiont_{i+1}

Outcomes

- Approach is independent of the fault localisation method used
- Experimental evidence from four SUTs plus their test suites drawn from the Software Infrastructure Repository (SIR)
- Increased the suspiciousness ranking and decreased the cost of fault localisation for 70% of the faults examined

Paper

Fault Localization Prioritization: Comparing Information Theoretic and Coverage Based Approaches. *Yoo, Harman and Clark*. ToSEM 2013.

Use Conditional Entropy to avoid Coincidental Correctness

Intended

input
`t1:x==3`
`t2:x==-5`

Unintended

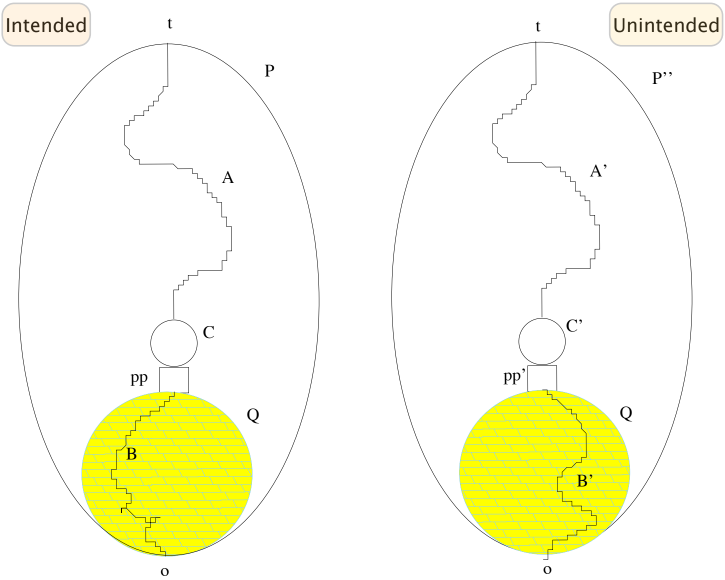
```
x=x+2;
if(x>0)
   x=x%4;
   else x=x;
```

```
x=3*x;
if(x>0)
   x=x%4;
   else x=x;
```

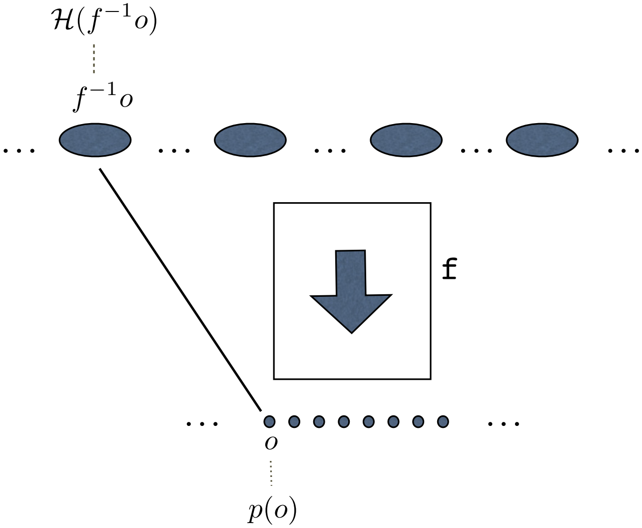
output
`t1:x==1`
`t2:x==-3`

output
`t1:x==1`
`t2:x==-15`

The Abstract View



Information Based View



Loss of information from running program $\mathbf{P}$

deterministic case

$$\mathcal{H}(I) - \mathcal{H}(O) \quad = \mathcal{H}(I|O)$$

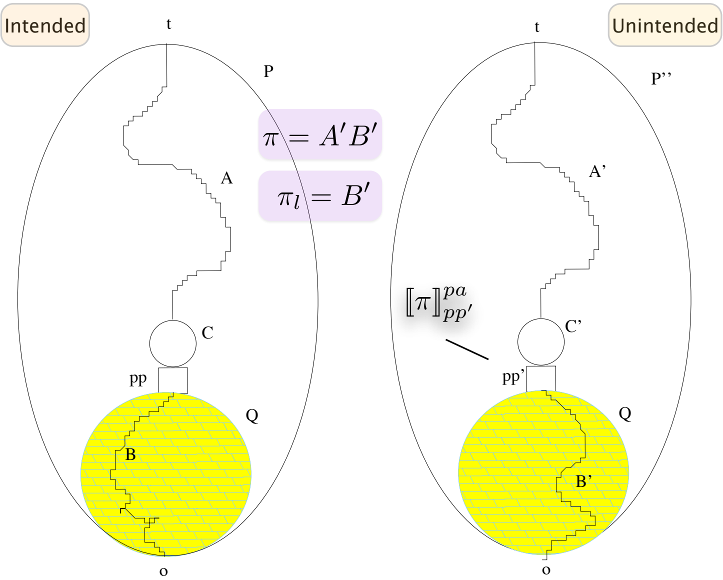
where $[\![P]\!]I = O$

Conditional entropy of I given O:
Squeeziness.

$$Sq(f) = \mathcal{H}(I) - \mathcal{H}(O) = \sum_{o \in O} p(o)\, \mathcal{H}(f^{-1}o)$$

via the partition property

Example Hypothesis



Summary

- 30 SUTS
- 1,408 Mutants
- 7,140,00 test cases
- Five different IT metrics experimentally investigated
- Two metrics showed 0.95 Spearman rank correlation with the probability of failed error propagation
- 10% of all 7,140,000 test inputs suffered from FEP

Paper

An Analysis of the Relationship between Conditional Entropy and Failed Error Propagation in Software Testing. *Androutsopoulos, Clark, Dan, Hierons and Harman*. ICSE 2014.

Use Kolmogorov Complexity to Measure Input Diversity

Normalised Information Distance

For two strings x and y,

$$\mathrm{NID}(x, y) = \frac{\max\{K(x|y), K(y|x)\}}{\max\{K(x), K(y)\}}$$

Enables comparisons between strings of different lengths

NCD: The Normalised Compression Distance

For two strings x and y,

$$\mathrm{NCD}(x, y) = \frac{C(xy) - \min\{C(x), C(y)\}}{\max\{C(x), C(y)\}}$$

Computable approximation using compressors such as 7zip, Bzip

Experiments

- Use a version of NCD for multisets – calculate the set "diameter"
- Bigger diameter means more diversity
- Purely consider sets of inputs – no information from executions except in the course of evaluation
- Inputs for three SUTs: JEuclid, NanoXML, ROME
- Controlled for input size
- Compared test sets using three fixed sizes: 10, 25 and 50

Outcomes for Higher Diameter Test Sets

- On average higher code coverage
- Higher code coverage than randomly selected test sets
- Leads to higher code coverage even if we control for the size of test inputs
- May have better fault-finding ability
- Selection scales quadratically in the size of the initial pool of tests and linearly with the average length of the tests

Paper

Test Set Diameter: Quantifying the Diversity of Sets of Test Cases. *Feldt, Poulding, Clark and Yoo*. ICST 2016.

Oracle Deficiencies

```
public class Subtract {
    public double value(double x, double y) {
        double result = x-y;
        assert (result != x);
        assert (result == x-y);
        return result;
    }
}
```

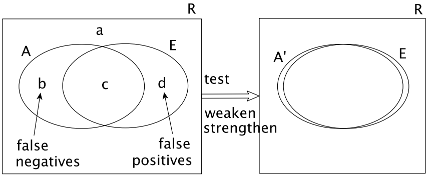
False alarm

```
public class FastMath {
    public int max (int a, int b) {
        int max;
        if (a >= b) {
            max = a;
        } else {
            max = b; // max = a;
        }
        assert (max >= a);
        return m
```

Missed fault

Oracles may be too strong (false alarms) or too weak (missed faults)

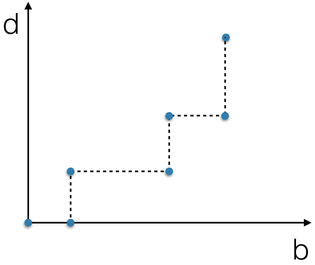
Oracle Improvement Steps



Since E is fixed:
a + b = const
c + d = const
(repartitioning)

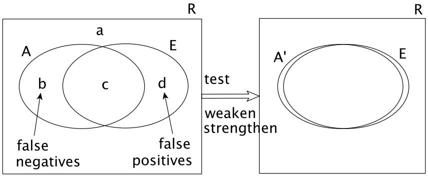
False negative reduction:
a' = a + Δ
b' = b - Δ

False positive reduction:
c' = c + Γ
d' = d - Γ



Mutual information

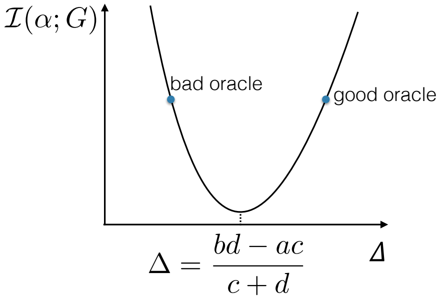
$$\mathcal{I}(X;Y) = \sum_{x \in X} \sum_{y \in Y} p(x,y)\, log_2 \frac{p(x,y)}{p(x)\,p(y)}$$



$$\mathcal{I}(\alpha;G) = \left\{ \begin{array}{l} -(b+c)log_2(b+c) - (a+d)log_2(a+d) \\ -(a+b)log_2(a+b) - (c+d)log_2(c+d) \\ +a\, log_2\, a + b\, log_2\, b + c\, log_2\, c + d\, log_2\, d \end{array} \right.$$

Bad Oracles

A **bad oracle** α is one for which ac < bd



Paper

Test Oracle Assessment and Improvement. *Jahangirova, Clark, Harman and Tonella*. ISSTA 2016.

In Conclusion

- Looked at contributions both theoretical and practical to
  - oracle improvement
  - test set diversity
  - coincidental correctness
  - test set prioritisation
- More to come:
  - InfoTestSS EPSRC funded project
  - Applying information theoretic ideas to test set selection and exploring relationships with coverage and mutation testing
  - EPSRC contribution approx £900,000 shared between UCL and Brunel
  - Industrial contribution approx £230,000 from J.P.Morgan and Berner Mattner
  - Project collaborators include *Rob Hierons, Mark Harman, Robert Feldt, Michele Boreale, Paolo Tonella*