[simula . research laboratory]

Conducting and Analyzing Empirical Studies in Search-based Software Engineering

Lionel Briand

Simula Research Laboratory

and University of Oslo, Norway

SSBSE 2011, Szeged, Hungary

Objectives

- Assumptions:
  - Familiarity with SBSE and main search techniques
  - Familiarity with basic probabilities and statistics
- Scope:
  - Introduction and fundamentals
  - Provide rationale for recommendations
- Discussions and exchange of views are welcome
- Slides are verbose to use as documentation and make the argument structure apparent

Acknowledgments

Based in part on the following articles:

- S. Ali, L. Briand, H. Hemmati, R. Panesar-Walawege, "A Systematic Review of the Application and Empirical Investigation of Search-based Test-case Generation", IEEE Transactions on Software Engineering 36(6): 742-762 , 2010.
- A. Arcuri and L. Briand, "A Hitchhiker's Guide to Statistical Tests for Assessing Randomized Algorithms in Software Engineering", (Simula TR, Extension from ICSE 2011 paper, Submitted)

Outline

- Problem definition
- Empirical study design
- Statistical analysis

Background

- Search techniques are increasingly used in SE to solve a variety of problems, ranging from requirements prioritization, re-engineering, to test generation and fault fixing
- Undecidable problems for which optimal solutions cannot be obtained within reasonable time
- Search -> Randomized algorithms
- The body of knowledge is increasing fast, as well as the research community.
- Now is the time to get our scientific procedures and reporting straight

Problem Definition

- For every problem addressed by SBSE, there is usually an alternative not based on search, e.g., constraint solver, model checker, static analysis
- Random search is always an alternative, and can be effective
- Many factors can affect the effectiveness and cost of a search technique, e.g., parameters, selected artifacts/problems, implementation
- Random variation (randomized algorithms)
- Designing, running, and analyzing SBSE empirical studies must be done with care to build a reliable body of knowledge
- Recent surveys shows this is not always the case

Random Variation

- Running a search or any randomized algorithm twice on the same instance of a software engineering problem usually produces different results.
- Probability distribution of search algorithm output in terms of various performance metrics
- Run an algorithm many times, collect data about performance
  - How many runs? How to draw reliable conclusions?
- Hypothesis testing used in all scientific fields
  - Different constraints, even within software engineering: sample sizes, distributions
  - Need specific guidelines for SBSE

Motivating Example

- Two algorithms, A and B
- Run n=10 times
- Time limit (e.g., 5 minutes)
- Binary output: *pass* or *fail*
  - A: successful 7 times
  - B: successful 5 times
- 20% of success rate difference
- Is A better than B???

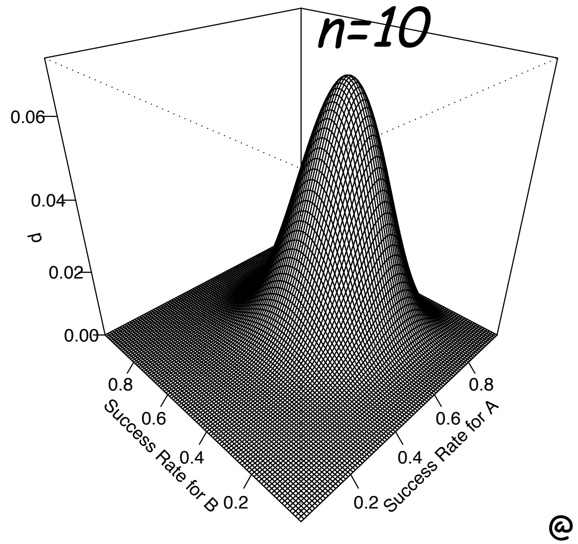
Binomial Distribution B(n,r)

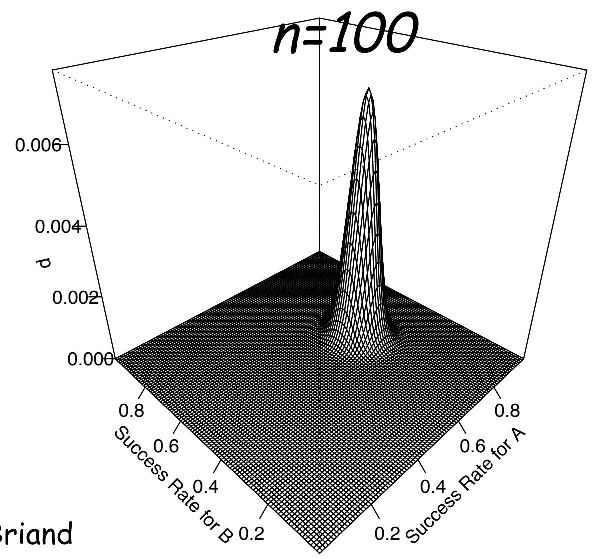
- *k* successes out of *n* runs, with success rate *r*
- *k* successes, n-k fails, and any order

$P(B(n,r)=k) = C(n,k) * r^k * (1-r)^{(n-k)}$

- $P(B(10,0.7)=7) = 0.26$!!!
- $P(B(10,.7)=7) * P(B(10,.5)=5) = 0.06$!!!

- *P(B(10,.7)=7) * P(B(10,.5)=5) = 0.06*
- What about if they had the same success rate: 60% ???
- *P(B(10,.6)=7) * P(B(10,.6)=5) = 0.04!!!*
- Impact of number of runs:

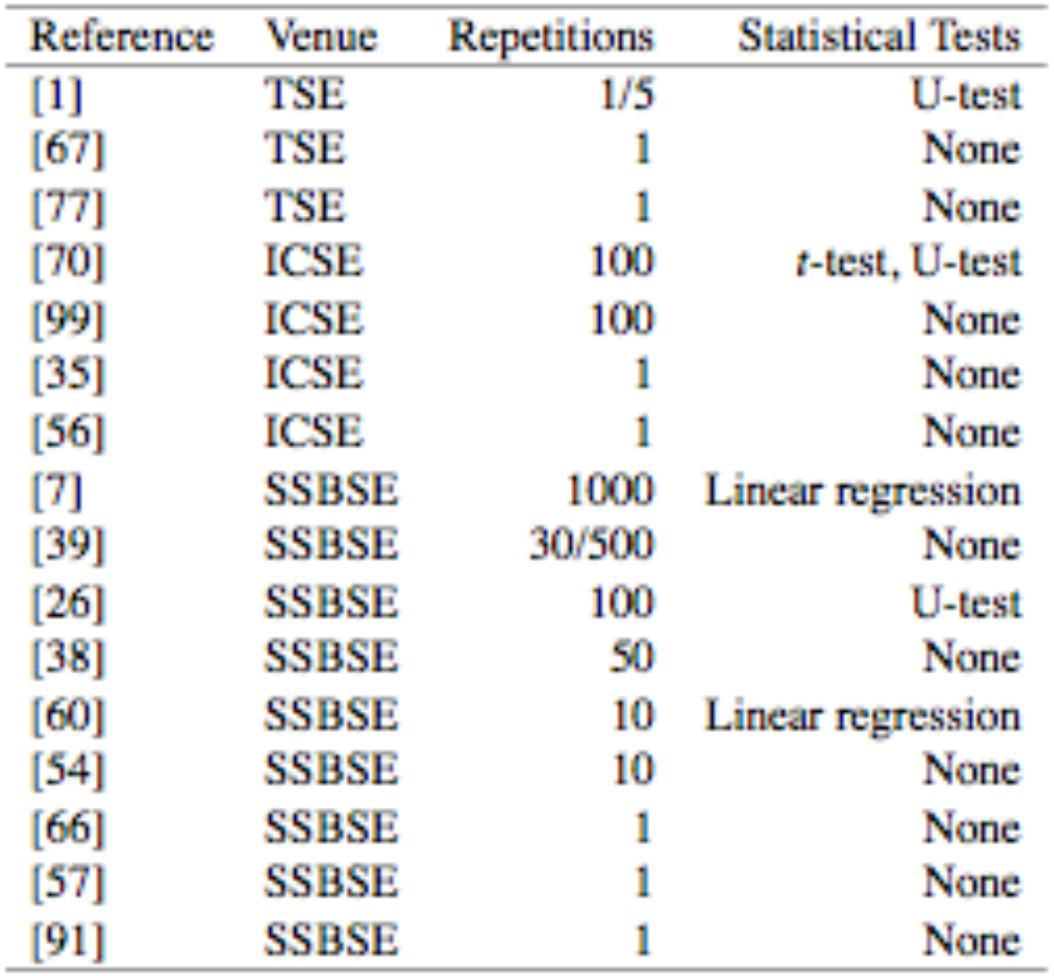




Research Survey

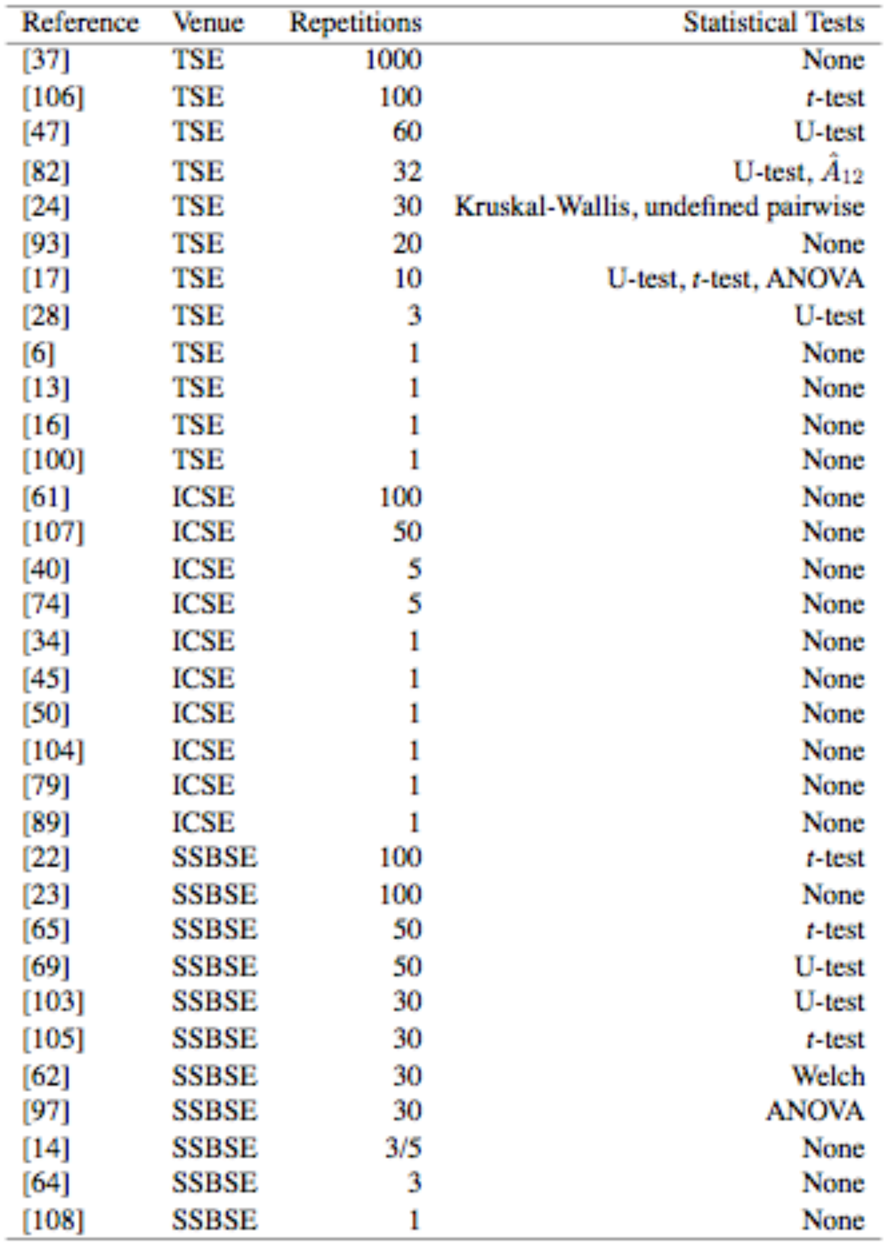
- Snapshot of current practice
- Venues: TSE, ICSE and SSBSE
- Year 2009 and 2010
- Results:
  - Large number of papers involving randomized algorithms
    - TSE/ICSE: 7% (2009) and 21% (2010)
  - Randomness often not taken into account properly
  - Seldom/inappropriate use of *statistical tests*…

2009



Reference	Venue	Repetitions	Statistical Tests
[1]	TSE	1/5	U-test
[67]	TSE	1	None
[77]	TSE	1	None
[70]	ICSE	100	*t*-test, U-test
[99]	ICSE	100	None
[35]	ICSE	1	None
[56]	ICSE	1	None
[7]	SSBSE	1000	Linear regression
[39]	SSBSE	30/500	None
[26]	SSBSE	100	U-test
[38]	SSBSE	50	None
[60]	SSBSE	10	Linear regression
[54]	SSBSE	10	None
[66]	SSBSE	1	None
[57]	SSBSE	1	None
[91]	SSBSE	1	None

2010



Reference	Venue	Repetitions	Statistical Tests
[37]	TSE	1000	None
[106]	TSE	100	t-test
[47]	TSE	60	U-test
[82]	TSE	32	U-test, $\hat{A}_{12}$
[24]	TSE	30	Kruskal-Wallis, undefined pairwise
[93]	TSE	20	None
[17]	TSE	10	U-test, t-test, ANOVA
[28]	TSE	3	U-test
[6]	TSE	1	None
[13]	TSE	1	None
[16]	TSE	1	None
[100]	TSE	1	None
[61]	ICSE	100	None
[107]	ICSE	50	None
[40]	ICSE	5	None
[74]	ICSE	5	None
[34]	ICSE	1	None
[45]	ICSE	1	None
[50]	ICSE	1	None
[104]	ICSE	1	None
[79]	ICSE	1	None
[89]	ICSE	1	None
[22]	SSBSE	100	t-test
[23]	SSBSE	100	None
[65]	SSBSE	50	t-test
[69]	SSBSE	50	U-test
[103]	SSBSE	30	U-test
[105]	SSBSE	30	t-test
[62]	SSBSE	30	Welch
[97]	SSBSE	30	ANOVA
[14]	SSBSE	3/5	None
[64]	SSBSE	3	None
[108]	SSBSE	1	None

Study Design

- Objectives
- Research questions
- Selection of artifacts (problems)
- Measuring cost and effectiveness

Objectives

- Problem to be addressed: regression test selection, identify schedules with possible deadline misses
    - Why is that important and in which context?
- Population of artifacts targeted, e.g., object-oriented classes, real-time systems with hard deadlines
- Information available (fitness function) and how to retrieve it, e.g., source code analysis, model
    - Is it realistic?
- What alternative techniques will be considered to address the problem and why?
    - Will not only include search techniques

Selecting Techniques

- In our context, it must include random search
  - The problem may be inherently easy (e.g., large proportion of acceptable solutions)
  - The problem may be artificially constrained so as to become easy, e.g., change operators tailored to bug sample in bug fixing
- Many (search) techniques
  - Can't try them all with all parameter settings
  - Match my problem to a standard optimization problem
  - Focus on well-studied, state-of-the-art techniques
  - Simulation studies for parameter settings
  - Our job in SBSE is to use these techniques in a smart way to solve SE problems, not do research in evolutionary computation
  - E.g., using GA, finding the right representation, fitness function, and change operators is hard

Describing Techniques

- Customizations and parameter settings must be reported and justified
- Often missing or incomplete in many SBSE papers
- Important for
  - Supporting the interpretation of results
  - Assess their validity
  - Compare results in future studies
  - Meta-analysis of results from multiple studies

Research Questions

- Will lead to formulate experimental hypotheses to be tested (see data analysis)
- Which techniques is better with various measures of effectiveness and cost?
  - Effectiveness under constant cost (e.g., generations, fitness evaluations, execution time to achieve coverage)
  - Cost to achieve an objective (e.g., execution time to trigger a failure)
- Discuss cost and effectiveness measurement next

Choice of Artifacts

- Potentially strong impact on results
- The No Free Lunch theorem states that, on average across all possible problems, all search algorithms have the same performance
- We usually do not know how to define target populations, but we need to do or best
- Random sampling from population not possible
- Often resort to convenience sample, e.g., open source projects
- Not always access to large collections of artifacts
- Threats to external validity are common

Choice of Artifacts II

- Representativeness, e.g., are container classes representative for OO class testing?
- Maximize diversity (subdomain, complexity) is one strategy and analyze differences in results
- In some cases, controlled generation of artificial cases are possible
- Building an empirical body of knowledge over time: replications

Measuring Cost and Effectiveness

- Validity of measures is often context dependent – justifications are required
- Cost measures should be comparable across techniques and clearly related to practical cost considerations
  - #fitness evaluations: cannot be used to compare with random search and is only valid if we can assume this is the main cost driver in algorithms
  - Several cost measures are typically used for various comparisons
- Effectiveness
  - Test generation: coverage or fault (real, mutation)
  - Directly related to fitness function or assumptions

Cost Measures

Two common goals: (1) Comparison across techniques, (2) Assess feasibility and scalability

Use surrogate measures to facilitate comparisons (1)

A study typically uses one or more of the following:

- The number of iterations, e.g., the number of generations in genetic algorithms, or cycles in ant colony optimization algorithms.
- The cumulative number of individuals in all iterations
- The number of fitness evaluations
- Execution time. This time can be either measured using clock time or CPU cycles.
- Problem specific cost measures of the *output* of the algorithm, e.g., the size of the resulting test suite.

Scalability

- How does cost and effectiveness evolve as a function of the size of the problem, e.g., number of philosophers when looking for deadlocks
- Several measures of "size" may be relevant, problem specific
- We characterize the relationship between size, cost and effectiveness, e.g., linear or exponential. Regression only usable if many artifacts
- Scalability analysis requires the selection or generation of artifacts of different sizes

Data Analysis

- Statistical difference
- Parametric versus non-parametric tests
- Censored data
- Effect size
- Sample size (runs)
- Multiple tests
- Comparing diverse artifacts

Statistical Difference

- Compare technique A and B, e.g., test generation using random testing and genetic algorithms
- Goal of hypothesis testing: Make statement(s) regarding unknown populations' parameter values based on sample data
- Main principle of hypothesis testing: Mathematical operations on data to produce a test statistic (e.g., *t*, *F*, *U*). The test statistic is evaluated in reference to a sampling distribution: a theoretical distribution to be expected if no difference (null hypothesis)
- Null hypothesis (H_0): no difference between A and B
- Can we reject H_0 with confidence?
- What aspect of the distribution is compared depends on the statistical test selected, e.g., mean, median

Statistical Difference 2

- Two types of error:
  - Type I: We reject H_0 when true
  - Type II: We accept H_0 when false
- p-value: probability of type I error
- Significance level (α): highest p-value we accept for rejecting H_0, e.g., 0.05 is common in other disciplines
- Statistical power ($1 - \beta$): probability of rejecting H_0 when false
- The two types of error are conflicting

H_0	T	F
Reject	I	▨
~~Reject~~	▨	II

Statistical Difference 3

- Establishing truths versus decision making
- α = 0.05, p-value = 0.06, we fail to reject the null hypothesis. Does that make sense?
- Just report p-values and let the reader decide in context?
- However, in our context this should not be an issue (many observations can be generated)
- The focus should rather be on the *effect size (difference)* and its confidence interval
- In practice: decision = f (cost of algorithms, effect size, p-value), minimize risks and maximize benefits

Statistical Difference 4

- One-tailed or two-tailed test? (definition next)
- p-value lower for one-tailed test
- Assumption about relative performance of algorithms based on theoretical/analytical grounds
- In our context, can we make such assumptions?
- Even naïve techniques can end up performing better
- When comparing randomized algorithms, it is advised to use two-tailed tests
- Lower p-values should be obtained by increasing the number of runs

1-tailed vs. 2-tailed test

- Definitions:
  - two-tailed: we do not hypothesize the difference to be in any particular direction
    - H_0: $A = B$, H_1: $A \neq B$
  - one-tailed: we hypothesize the difference to be in one direction
    - H_0: $A \leq B$, H_1: $A > B$
- The choice depends on prior knowledge.
- One-tailed tests lead to lower p-values
- Note: The choice should not be made after looking at the data!

Parametric versus non-Parametric Tests

- A *parametric* test makes assumptions about the data distributions, e.g., *t*-test assumes normality and equal variance in samples
- Nonparametric tests don't and are usually based on ranks. There are nonparametric versions of most parametric tests
- In our context: t-test and Mann-Whitney U-test
- In general non-parametric tests less powerful if parametric test assumptions are true. Useful when samples are small
- Parametric tests robust to some extent to violations ....
- Central Limit Theorem tells us the t-test is robust to strong departure from normality for large samples
- Large? rule of thumb: $n > 30$ in each sample? Assumptions of CLT

Central Limit Theorem

Central Limit Theorem (simplified): irrespective of the distribution of the parent population-given that its mean m and a variance s^2, and so long as the sample size n is large, the distribution of sample means is *approximately normal* with mean m and variance s^2 /n.

Parametric versus non-Parametric Tests 2

- Parametric and non-parametric tests analyze different properties
- *t*-test assess differences in means while Mann-Whitney U-test differences in stochastic order (median)
- For randomized algorithms we recommend using the U-test
- If one of the two algorithms is not randomized but deterministic, statistical testing still applies, but a *one-sample* test should be used: one-sample Wilcoxon test
- If B deterministic with performance m_b, we would test whether the performance of *A* is symmetric about m_b

Mann-Whitney U test

Objective: Alternative to the *t*-test. Test whether the mean ranks of two independent groups are significantly different from each other. Can be one-tailed or two-tailed.

Requirements: independent random samples, ordinal level data, may cope with tied ranks, U has a normal distribution if sample large enough (10 observations per sample if no tied scores).

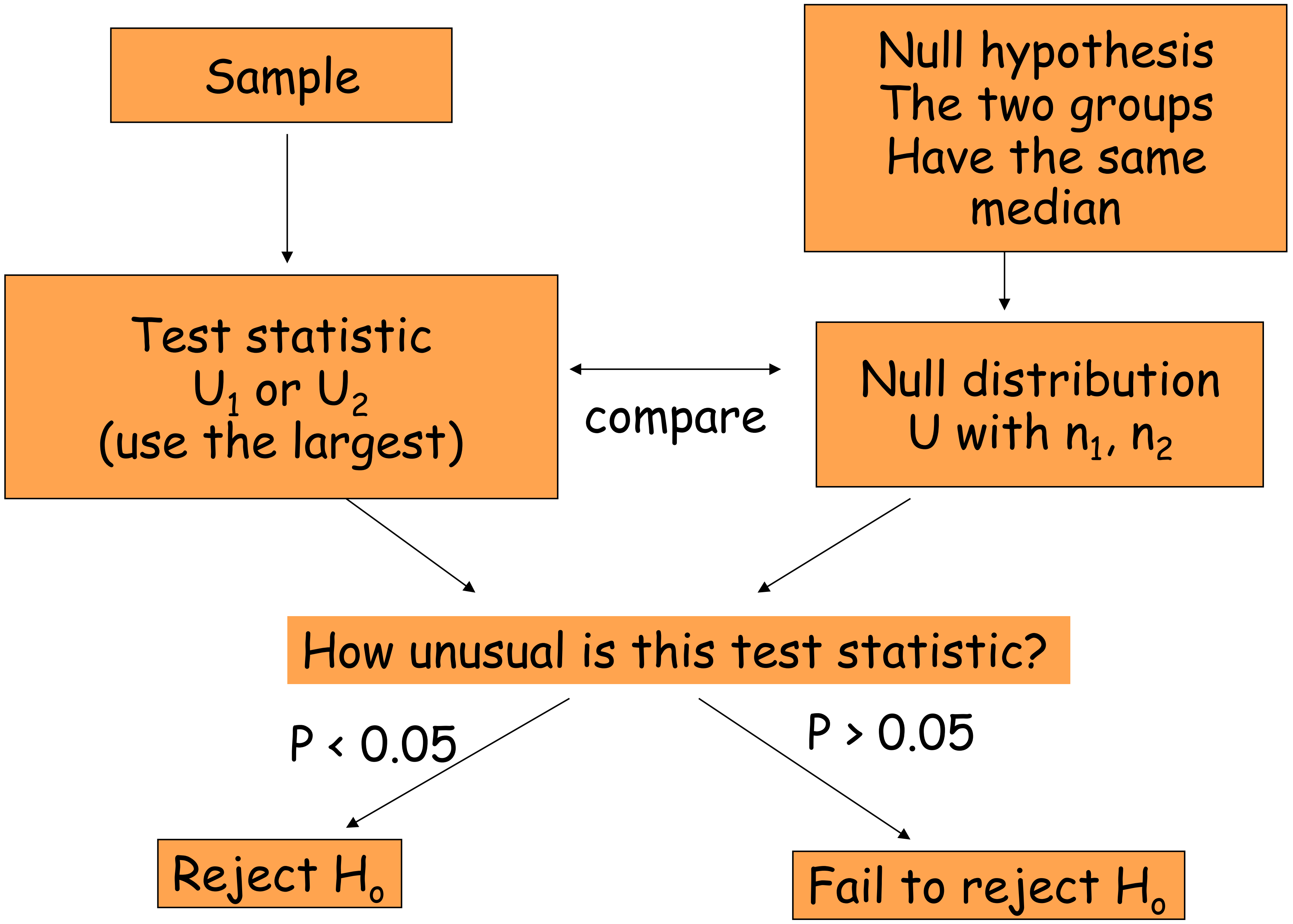
Statistic:

$$U_1 = n_1 n_2 + \frac{n_1(n_1+1)}{2} - R_1$$

n_1= sample size of group 1
n_2= sample size of group 2
R_1= sum of ranks of group 1

$$U_2 = n_1 n_2 - U_1$$

Use the larger of U1 or U2 for a two-tailed test


Sample
Null hypothesis The two groups Have the same median
Test statistic U_1 or U_2 (use the largest)
compare
Null distribution U with n_1, n_2
How unusual is this test statistic?
$P < 0.05$
$P > 0.05$
Reject H_o
Fail to reject H_o

$$U1 = n_1 n_2 + \frac{n_1(n_1+1)}{2} - R_1$$

$$U1 = (7)(5) + \frac{(7)(8)}{2} - 30$$

$$U1 = 35 + 28 - 30$$

$$\mathbf{U1 = 33}$$

$$U2 = n_1 n_2 - U$$

$$U2 = (7)(5) - 33$$

$$U2 = 2$$

$$\mathbf{U_{0.05(2),7,5} = U_{0.05(2),5,7} = 30}$$

As 33 > 30, H_o is rejected

Heights of males (cm)	Heights of females (cm)	Ranks of male heights	Ranks of female heights
193	175	1	7
188	173	2	8
185	168	3	10
183	165	4	11
180	163	5	12
178		6	
170		9	
$n_1 = 7$	$n_2 = 5$	$R_1 = 30$	$R_2 = 48$

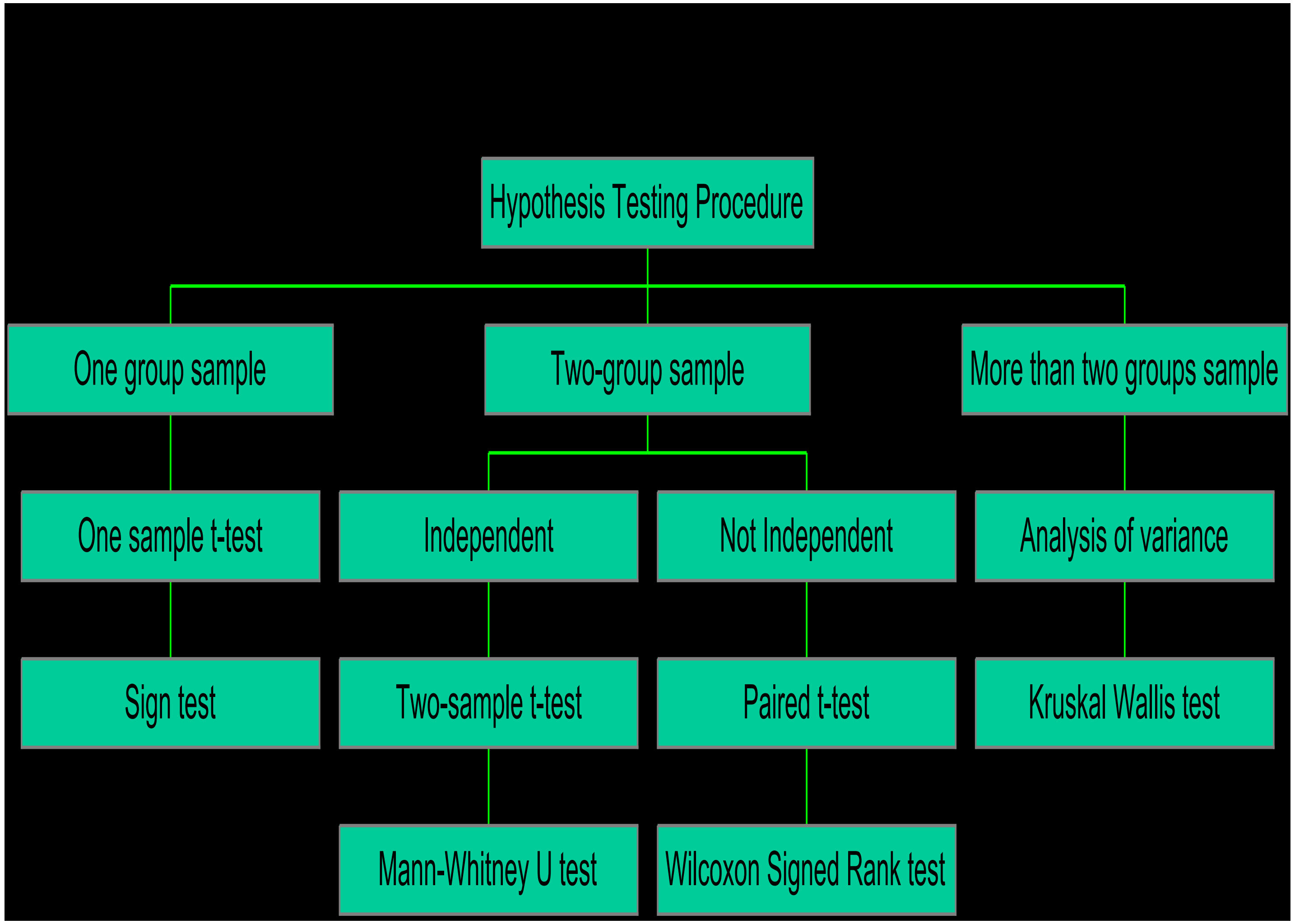
Zar, 1996

Hypothesis Testing Steps

1. State H_0 (i.e., what you are trying to disprove)
2. State H_1
3. Determine α (at your discretion)
4. Determine the test statistic and associated p-value
5. Determine whether to reject H_0 or fail to reject H_0

Reminder: p-values are only estimates based on assumptions, α is based on the level of risk you are willing to take

Classification of tests



Censored Data

- Compare two algorithms A and B
- X and Y are the time A and B take to find a solution, respectively
- Run A and B n times, e.g., n targets such as branches or bugs to be fixed
- Time limit L, right censored data
- Success rate: $\gamma = k/n$, follows a binomial distribution
- Test differences in proportion with *Fisher exact test*
- Up to n = 100, p-values are precise, not estimates
- If success rates are similar and high, then a practical question is which technique uses less time?
- Use Mann-Whitney U-test on successful runs

Fisher's Exact Test

- *Fisher's Exact Test* is a test for independence in a 2 X 2 table. The test holds the marginal totals fixed and computes the hypergeometric probability that the data arrangement in the table is at least as unbalanced.
- It is most useful when the total sample size and the expected values are small (any cell below 5). This is not uncommon in our context.
- This distribution gives probabilities for the number of 'successes' in a sample of size n drawn without replacement from a population of size N comprised of a known number of 'successes'

Fisher's Exact Test Example

Technique 1

Technique 2

	yes	no	total
yes	3	7	10
no	5	10	15
total	8	17	

Software testing example: Are differences in success proportions for techniques 1 and 2 significantly different for these 25 targets?

Hypergeometric distribution

Example: 2x2 table with cell counts a, b, c, d. Assuming marginal totals are fixed:

M1 = a+b, M2 = c+d, N1 = a+c, N2 = b+d.

N = a + b + c + d = N1 + N2 = M1 + M2

probability (p) of observing this particular arrangement of the data

p follows a hypergeometric distribution:

p= N1!N2!M1!M2! / [N!a!b!c!d!]

To compute statistical significance, Fisher showed that you need to account only for the cases where the marginal totals are the same and among those only the cases that are more or as extreme extreme

	yes	no	total
yes	a	b	M1
no	c	d	M2
total	N1	N2	N

Effect Size (ES)

- We typically have large numbers of runs
- Statistically significant difference might be irrelevant if low
  - E.g., success rates 50% vs 50.1%
- The magnitude of the improvement is key: effect size
- Need *standardized* effect size measure
  - Difference in "mean" values is not sufficient
  - Depends on raw value: 2-1=1 as good as 91-90=1
  - Ignores size effect variance in sample of artifacts
- Important when comparing results among case study artifacts of various "complexity"

ES For Success Rates

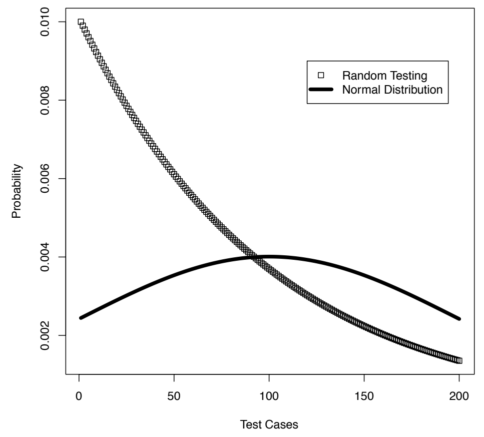
- *Odds ratio*: is a measure of how many times greater the odds are that a member of a certain population will fall into a certain category than the odds are that a member of another population will fall into that category
- In our context:
  - Odds_ratio = (a/n-a) * (b/n-b)

    Where n runs, a and b successes for two techniques
- Use confidence intervals for a given α
  - Can replace hypothesis testing

ES In The Other Cases

- Most famous: Cohen D
- Difference in mean divided by *pooled variance*
- Rarely adequate our context
  - Far too sensitive to distribution shapes, e.g., strong departures from normality
  - Nearly meaningless when comparisons with random search (geometric distribution)

$$ES = \frac{\overline{X}_1 - \overline{X}_2}{\sqrt{\frac{s_1^2(n_1-1)+s_2^2(n_2-1)}{n_1+n_2-2}}} = \frac{\overline{X}_1 - \overline{X}_2}{s_{pooled}}$$



Vargha-Delaney *A* statistic

- Probability that one run of X gives better result than a run of Y
- E.g., *A*=0.8 means that, if you run X once and Y once, then there is an 80% chance that result of X is better
- $A = (R_1/m - (m+1)/2)/n$, where R_1 is the rank sum of the first group, m and n are the number of observations in the first group and second group, respectively
- In our context, often m = n (runs)
- We only found one paper in the literature using *A*
- Estimate => confidence intervals for *A*

Number of Runs

- As many as you can, e.g., 1000
  - Lower p-values, more accurate effect sizes, etc
- If if differences are large enough for selected α, additional runs help tighten confidence intervals for effect size
- Sometimes not possible:
  - E.g., execution required for fitness calculation and hardware-in-the-loop testing
- If not possible, state why
  - E.g., difficult to justify if all experiments take less than an hour…

Multiple Tests

- Often more than two algorithms to compare
- Or different algorithm settings
- Solution depends on question:
  - Has the choice of an algorithm/setting any effect?
    - Omnibus test such as ANOVA, Kruskal-Wallis
  - Which one is the best?
    - Pairwise comparisons: $Z = K(K - 1)/2$
    - Inflates the probability of Type I error
    - Probability that at least one null hypothesis is true could be as high as $1 - (1 - \alpha)^Z$

Multiple Tests II

- *Very tricky situation,* scientists disagree …
- Bonferroni adjustment: use α/Z as significance level (very conservative). Many more subtle variants.
- Using adjustment can hinder scientific progress by reducing the number of published results: tempting to leave out the poorly performing algorithms
- Our context is about decision making, e.g., choosing a test generation technique, not establishing truths about natural phenomena
- *Rule of thumb*: compute scores for each technique based on pair-wise comparisons with pair-wise effect sizes, to obtain overall ranking.
    - For each pair-wise comparison increment or decrement a score for each technique if difference significant (Hemmati et al., 2011)

Many Diverse Artifacts

- Sometimes, possible large number of diverse artifacts:
  - Random generators, simulators
  - Open-source software
- Difficult problem but: (1) define target and justify artifacts to the extent possible, (2) explain differences in results based on artifact characteristics
- Tradeoff # of runs vs # of artifacts
- Maximize # artifacts, but get at least enough runs (e.g., 30) for each artifact to get enough power
- Used paired statistical tests to draw conclusions from large samples of diverse artifacts

Paired Tests

- When comparing two techniques, we typically have two groups of cost or effectiveness observations across artifacts
- For each observation in group 1, there is a corresponding observation in group 2.
  - Key is that all variables, other than what we are interested in, are controlled for between samples
  - In our case, only the technique varies across paired observations
  - Null hypothesis: $\delta = \mu 1 - \mu 2 = 0$

Example

- Two techniques A and B to unit test object-oriented software
- Each algorithm run *n* times on five classes
- Assume the evaluation if the number of test cases generated before achieving full coverage
- X = {10k, 20k, 30k, 40k, 50k}, whereas for the second algorithm we obtain Y = {12k, 21k, 34k, 41k, 53k}
- Unpaired MannWhitney U-test: p-value = 0.69
- Paired Wilcoxon T test: p-value = 0.057
- Due to design or random sampling, different artifacts presents different levels of difficulty in achieving coverage
- With a paired test, artifacts act as their own control

Analyzing Differences in Results

- Comparing performance differences overall across artifacts is not enough
- The magnitude of differences should be looked at carefully
- Collect standardized effect sizes for each problem instance, and then average them
- But it does not fully solve the problem
- $A = \{0.6, 0.6, 0.6, 0.6, 0.1\}$. If we average those values on the entire case study, we would obtain $A = 0.5$, thus suggesting there is no difference among the two algorithms!

General Guidelines

- If binary output (e.g., success rates):
  - Fisher Exact test
- Otherwise, use:
  - Mann-Whitney U-test
- NEVER use a *t*-test
- Sometimes both: if same success rate, can check if "faster" with U-test
- When comparing scores of techniques across artifacts, use a paired test