

GETTING IT RIGHT: SYSTEM TESTING OF SCIENTIFIC SOFTWARE

Myra Cohen

<https://www.cs.iastate.edu/~mcohen>

mcohen@iastate.edu



IOWA STATE UNIVERSITY
OF SCIENCE AND TECHNOLOGY



Laboratory for Variability-Aware Assurance and
Testing of Organic Programs



Scientific Software is Everywhere

The screenshot shows the ECP website with a dark blue header featuring the ECP logo and navigation links for Home, About, Research, News, Training, and Library. The main content area has a blue background. A large "Feature" section on the left contains a diagram illustrating the workflow from experimental data to molecular simulations. The title "EXAFEL AND COPA: RAPID IMAGING OF MOLECULAR SYSTEMS" is displayed, along with a brief description and a source link. Below this are three "Highlight" boxes: one about EXAALT and KOKkos for material behavior simulations, another about the ECP's role in NASA's new era, and a third about contributions to cancer research.

Feature

EXAFEL AND COPA: RAPID IMAGING OF MOLECULAR SYSTEMS

Using CoPA tools, ExaFEL prevents computational data throughput from bottlenecking experimental progress in x-ray free electron laser facilities.

Source: [ECP](#)

Highlight

EXAALT AND KOKKOS: MAKING EXASCALE SIMULATIONS OF MATERIAL BEHAVIOR A "SNAP"

Molecular dynamics has become a cornerstone of computational science and is a key component of developing materials with enhanced properties.

Source: [ECP](#)

Highlight

EXASCALE COMPUTING PROJECT SOFTWARE HELPS LAUNCH A NEW ERA FOR NASA

ECP Software Helps Launch a New Era for NASA

Source: [ECP](#)

Highlight

EXASCALE COMPUTING PROJECT CONTRIBUTES TO ACCELERATING CANCER RESEARCH

The Exascale Computing Project's CANDLE application will improve cancer research techniques and clinical outcomes

Source: [ECP](#)

Yet it sometimes Fails..

```
Error: NCBI C++ Exception:
```

```
T0 "/tmp/BLAST/2.11.0/gompi-2020b/ncbi-blast-2.11.0+-src/c++/src/serial/objistrasnb.cpp"
T0 "/tmp/BLAST/2.11.0/gompi-2020b/ncbi-blast-2.11.0+-src/c++/src/serial/member.cpp", l-
```

Yet it sometimes Fails..

```
Error: NCBI C++ Exception:
```

```
T0 "/tmp/BLAST/2.11.0/gompi-2020b/ncbi-blast-2.11.0+-src/c++/src/serial/objistrasnbp.cpp"
T0 "/tmp/BLAST/2.11.0/gompi-2020b/ncbi-blast-2.11.0+-src/c++/src/serial/member.cpp", l-
```

```
ial/objistrasnbp.cpp", line 499: Error: (CSerialException::eOverflow) byte 132: overf
ial/member.cpp", line 767: Error: (CSerialException::eOverflow) ncbi::CMemberInfoFur
```

Yet it sometimes Fails..

NCBI blastp bug - changing max_target_seqs returns incorrect top hits

[2015-11-30-blastp-bug.md](#)

Raw

NCBI blastp seems to have a bug where it reports different top hits when -max_target_seqs is changed. This is a serious problem because the first 20 hits (for example) should be the same whether -max_target_seqs 100 or -max_target_seqs 500 is used.

The bug is reproducible on the command line when searching NCBI's nr blast database (dated 25-Nov-2015) using NCBI 2.2.28+, 2.2.30+ and 2.2.31+.

Yet it sometimes Fails..

NCBI blastp bug - changing max_target_seqs returns incorrect top hits

2015-11-30-blastp-bug.md Raw

NCBI blastp problem I
500 is us
The bug
2.2.28+,

Bioinformatics, 35(9), 2019, 1613–1614
doi: 10.1093/bioinformatics/bty833
Advance Access Publication Date: 24 September 2018
Letter to the Editor

OXFORD serious et_seqs
ig NCBI

Sequence analysis

Misunderstood parameter of NCBI BLAST impacts the correctness of bioinformatics workflows

Nidhi Shah¹, Michael G. Nute², Tandy Warnow  ³ and Mihai Pop  ^{1,*}

¹Department of Computer Science, University of Maryland College Park, MD 20742, USA, ²Department of Statistics and ³Department of Computer Science, University of Illinois Urbana-Champaign, Champaign, IL 61820, USA

*To whom correspondence should be addressed.
Associate Editor: John Hancock
Contact: mpop@umd.edu

Received and revised on August 13, 2018; editorial decision on September 19, 2018; accepted on September 21, 2018

Sometimes it seems to fail..

Bioinformatics, 35(15), 2019, 2699–2700
doi: 10.1093/bioinformatics/bty1026
Advance Access Publication Date: 24 December 2018
Letter to the Editor

Sequence analysis

**Reply to the paper: Misunderstood parameters
of NCBI BLAST impacts the correctness of
bioinformatics workflows**

Thomas L. Madden*, Ben Busby and Jian Ye

National Center for Biotechnology Information, National Library of Medicine, National Institutes of Health,
Bethesda, MD 20894, USA

*To whom correspondence should be addressed.
Associate Editor: John Hancock
Contact: madden@ncbi.nlm.nih.gov

Received and revised on November 30, 2018; editorial decision on December 10, 2018; accepted on December 19, 2018



OXFORD

Sometimes it seems to fail..

Dear Editor,

A recent letter by [Shah et al. \(2018\)](#) addressed the use of a command-line parameter in BLAST ([Altschul et al., 1997](#); [Camacho et al., 2009](#)). BLAST is a very popular tool, so it is not surprising that this topic has provoked a great deal of interest. The authors have, however, conflated three different issues. One is a bug that will be fixed in the BLAST+ 2.8.1 release due out in December 2018, another is simply how BLAST works and the third might be viewed as a [shortcoming of our implementation of composition-based statistics \(CBS\)](#). Here, we address these issues and describe some new documentation about the BLAST process.

[Shah et al. \(2018\)](#) did not provide their own example in the letter, but later provided one at https://github.com/shahnidhi/BLAST_maxtargetseq_analysis. At the NCBI, we examined the new example and it became clear that the demonstrated behavior was a bug, resulting from an overly aggressive optimization, introduced in 2012 for BLASTN and MegaBLAST (DNA–DNA alignments). This bug has been fixed in the BLAST+ 2.8.1 release, due out in December 2018. The aberrant behavior seems to occur only in alignments with an extremely large number of gaps, which is the case in the example provided by Shah and collaborators.

Cost of Poor Quality Software

- CISQ Consortium for Information & Software Quality 2020 report
\$2.08 trillion cost in United States
- 2002 National Institutes of Health Report
Up to \$59 Billion per year in United States
- Scientific Software
?

Why Test Software?

contributed articles



DOI:10.1145/2667219

Dynamic analysis techniques help programmers find the root cause of bugs in large-scale parallel applications.

BY IGNACIO LAGUNA, DONG H. AHN, BRONIS R. DE SUPINSKI, TODD GAMBLIN, GREGORY L. LEE, MARTIN SCHULZ, SAURABH BAGCHI, MILIND KULKARNI, BOWEN ZHOU, ZHEZHE CHEN, AND FENG QIN

**Debugging
High-Performance
Computing
Applications at
Massive Scales**



Why Test Software?

contributed articles



An approach to reproducibility problems related to porting software across machines and compilers.

BY DONG H. AHN, ALLISON H. BAKER, MICHAEL BENTLEY, IAN BRIGGS, GANESH GOPALAKRISHNAN, DORIT M. HAMMERLING, IGNACIO LAGUNA, GREGORY L. LEE, DANIEL J. MILROY, AND MARIANA VERTENSTEIN

DOI:10.1145/3382037

Keeping Science on Keel When Software Moves

the machine instructions that actually get executed. Unfortunately, such changes do affect the computed results to a significant (and often worrisome) extent. In a majority of cases, there are not easily definable *a priori* answers one can check against. A programmer ends up comparing the new answer against a trusted baseline previously established or checks for indirect confirmations such as whether physical properties such as energy are conserved. However, such non-systematic efforts might miss underlying issues, and the code may keep misbehaving until these are fixed.

In this article, we present real-world evidence to show that ignoring numerical result changes can lead to misleading scientific conclusions. We present techniques and tools that can help computational scientists understand and analyze compiler effects on their scientific code. These techniques are applicable across a wide range of examples to narrow down the root-causes to single files, functions within files, and even computational expressions that affect specific variables. The developer may then rewrite the code selectively and/or suppress the application of certain optimizations to regain more predictable behavior.

Going forward, the frequency of required ports of computational software will increase, given that performance gains can no longer be obtained by mere-

In this article, we present real-world evidence to show that ignoring numerical result changes can lead to misleading scientific conclusions. We present tech-

Why Test Software?



Merali, Zeeya. “Computational Science: ...Error.” *Nature* 467, no. 7317 (Oct, 2010): 775–77



hen hackers leaked thousands of e-mails from the Climatic Research Unit (CRU) at the University of East Anglia in Norwich, UK, last year, global-warming sceptics pored over the documents for signs that researchers had manipulated data. No such evidence emerged, but the e-mails did reveal another problem — one described by a CRU employee named “Harry”, who often wrote of his wrestling matches with wonky computer software.

“Yup, my awful programming strikes again,” Harry lamented in one of his notes, as he attempted to correct a code analysing weather-station data from Mexico.

Overview



Types of
Testing



Challenges



Models



Coverage

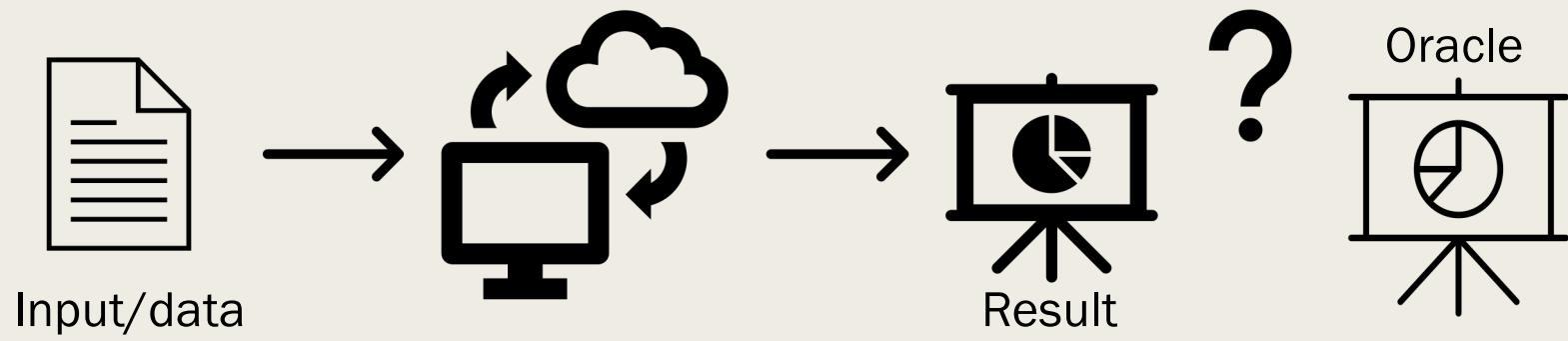


Oracles



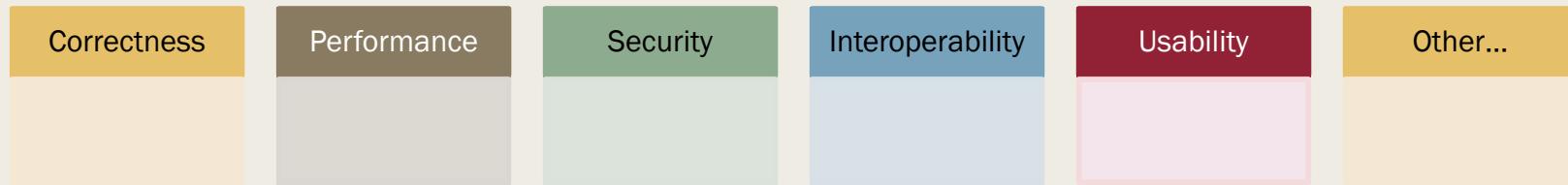
Configurability

What is Testing



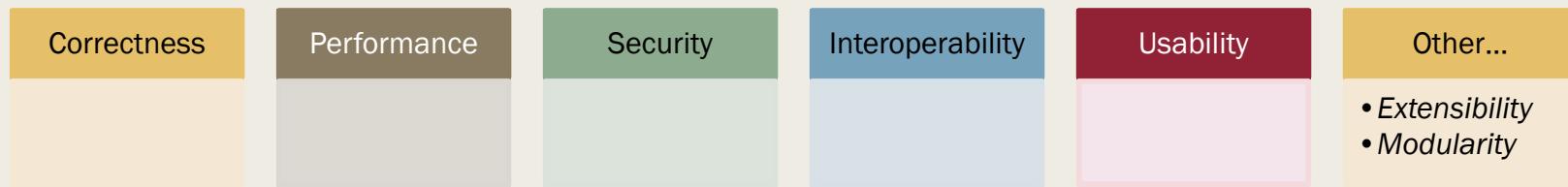


What Should We Test?





What Should We Test?



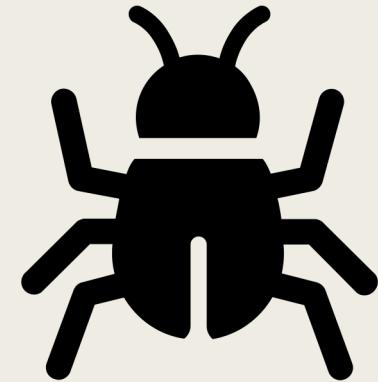
Unit tests are important, but there is more....



Limitations

Testing can only **show the presence of faults**.
It **cannot** determine their absence.

Edsger W. Dijkstra





Challenge 1

- To detect a program FAILURE we need to:
 - Reach a FAULT in the code
 - Infect the code (change to incorrect state) - ERROR
 - Propagate the error out of program
 - Reveal (detect) the error – (ORACLE)

RIPR model Ammann, Offutt (*Introduction to Software Testing, 2016*)



Challenge 2

- Covering code during testing, only tests the logic that is there!
 - *Also need to test from the system – does the software meet the specifications*



Tests Can Miss Faults

```
def classify_triangle(a, b, c):
    # Sort the sides so that a <= b <= c
    if a > b:
        tmp = a
        a = tmp    #fault should be a=b
        b = tmp

    if a > c:
        tmp = a
        a = c
        c = tmp

    if b > c:
        tmp = b
        b = c
        c = tmp

    if a + b <= c:
        return TriangleType.INVALID
    elif a == b and b == c:
        return TriangleType.EQUILATERAL
    elif a == b or b == c:
        return TriangleType.ISOSCELES
    else:
        return TriangleType.SCALENE
```



Tests Can Miss Faults

```
def classify_triangle(a, b, c):
    # Sort the sides so that a <= b <= c
    if a > b:
        tmp = a
        a = tmp    #fault should be a=b
        b = tmp

    if a > c:
        tmp = a
        a = c
        c = tmp

    if b > c:
        tmp = b
        b = c
        c = tmp

    if a + b <= c:
        return TriangleType.INVALID
    elif a == b and b == c:
        return TriangleType.EQUILATERAL
    elif a == b or b == c:
        return TriangleType.ISOSCELES
    else:
        return TriangleType.SCALENE
```

1. Test Case 3, 4, 5 (scalene)
 - doesn't reach fault





Tests Can Miss Faults

```
def classify_triangle(a, b, c):
    # Sort the sides so that a <= b <= c
    if a > b:
        tmp = a
        a = tmp  #fault should be a=b
        b = tmp  5,5,1

    if a > c:
        tmp = a
        a = c
        c = tmp

    if b > c:
        tmp = b
        b = c
        c = tmp

    if a + b <= c:
        return TriangleType.INVALID
    elif a == b and b == c:
        return TriangleType.EQUILATERAL
    elif a == b or b == c:
        return TriangleType.ISOSCELES
    else:
        return TriangleType.SCALENE
```

1. Test Case 3, 4, 5 (scalene)
 - doesn't reach fault
2. Test Case 5, 1, 1 (invalid)
 - reaches fault and infects
 - reveals (returns isosceles)





Tests Can Miss Faults

```
def classify_triangle(a, b, c):
    # Sort the sides so that a <= b <= c
    if a > b:
        tmp = a
        a = tmp    #fault should be a=b
        b = tmp    2,2,-1

    if a > c:
        tmp = a
        a = c
        c = tmp

    if b > c:
        tmp = b
        b = c
        c = tmp

    if a + b <= c:
        return TriangleType.INVALID
    elif a == b and b == c:
        return TriangleType.EQUILATERAL
    elif a == b or b == c:
        return TriangleType.ISOSCELES
    else:
        return TriangleType.SCALENE
```

1. Test Case 3, 4, 5 (scalene)
 - doesn't reach fault
2. Test Case 5, 1, 1 (invalid)
 - reaches fault and infects
 - reveals (returns isosceles)
3. Test Case 2, 1, -1 (invalid)
 - reaches fault and infects
 - Doesn't propagate (2, 2,-1) is still INVALID



kbase-all-hands-on

v1 - KBaseFBA.FBA-13.2

[Overview](#)[Reaction fluxes](#)[Exchange fluxes](#)[Genes](#)[Biomass](#)[Pathways](#)[Bar charts](#)

ID	myracohen:narrative_1501886992034/kbase-all-hands-on
Object type	KBaseFBA.FBA-13.2
Owner	myracohen
Version	1
Mod-date	2024-02-26T06:15:44+0000
Objective value	0.507834
Model	E-Coli_k12_MG1
Media	Carbon-D-Glucose
Single KO	0
Number reactions	857
Number compounds	29
Gene KO	0
Reaction KO	0
Custom bounds	0



kbase-all-hands-on-different-media

v2 - KBaseFBA.FBA-13.2

[Overview](#)[Reaction fluxes](#)[Exchange fluxes](#)[Genes](#)[Biomass](#)[Pathways](#)[Bar charts](#)

ID	myracohen:narrative_1501886992034/kbase-all-hands-on-different-media
Object type	KBaseFBA.FBA-13.2
Owner	myracohen
Version	2
Mod-date	2024-02-26T06:36:45+0000
Objective value	13.4555
Model	E-Coli_k12_MG1
Media	Complete
Single KO	0
Number reactions	857
Number compounds	29
Gene KO	0
Reaction KO	0
Custom bounds	0





The Future?

, or alternatively, Machine learning (ML) classifiers will select the templates for model reconstruction

Build and gap-fill genome-scale metabolic models with ModelSEED v2 (MS2)

ModelSEED 2 genome-scale metabolic reconstruction pipeline enabling quantitative prediction of ATP production

A genome annotated with RAST is provided as input. Users can select a template for reconstruction, or alternatively, Machine learning (ML) classifiers will select the templates for mode reconstruction. ATP production is tested across 54 media formulations representing many diverse energy biosynthesis strategies, with gap-filling performed as necessary to ensure ATP is produced in at least one condition. Next, we expand the core metabolism model to the genome-scale. The genome-scale model's gap-filling is then performed using our default auxotrophic medium or an optional user-specified custom media formulation.



Overview



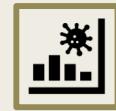
Types of
Testing



Challenges



Models



Coverage



Oracles



Configurability



Models



Provide an **abstraction** of the software we are testing



Can be **for different dimensions** of the software (specifications, interface, code)



Allow us to reason about **how much** we have tested



The foundation for **automated test generation**



Example Models

Graphs

Tabular

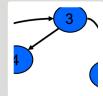
Relational

Grammar based

Logic based



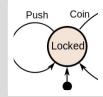
Graph Models



Program control flow graph



User interface



Program state machine



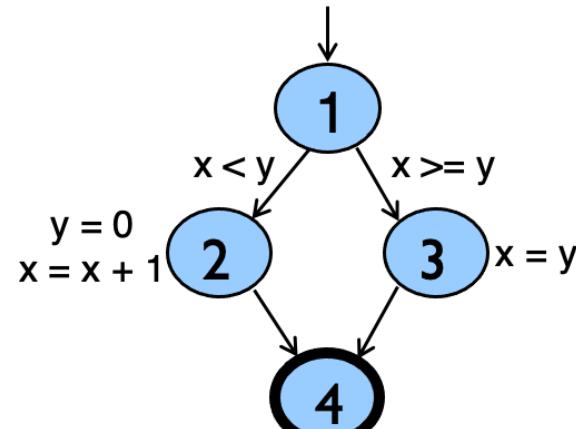
Types of Graph Coverage

- All **nodes**
- All **edges** (pairs of nodes)
- All **length N** paths
- **M random** length N paths



Program Code Coverage

```
if (x < y)
{
    y = 0;
    x = x + 1;
}
else
{
    x = y;
}
```



Control flow graph



Program Code Coverage

Cog coverage: 38.75%

coverage.py v7.2.7, created at 2023-05-29 15:26 -0400

Module	statements	missing	excluded	branches	partial	coverage
cogapp/__init__.py	1	0	0	0	0	100.00%
cogapp/__main__.py	3	3	0	0	0	0.00%
cogapp/cogapp.py	500	224	1	210	30	49.01%
cogapp/makefiles.py	22	18	0	14	0	11.11%
cogapp/test_cogapp.py	845	591	2	24	1	29.57%
cogapp/test_makefiles.py	70	53	0	6	0	22.37%
cogapp/test_whiteweutils.py	68	50	0	0	0	26.47%
cogapp/whiteutils.py	43	5	0	34	4	88.31%
Total	1552	944	3	288	35	38.75%

coverage.py v7.2.7, created at 2023-05-29 15:26 -0400

Triangle.java

```
1. public class Triangle {  
2.     public enum TriangleType {  
3.         INVALID, SCALENE, EQUILATERAL, ISOSCELES  
5.     }  
6.  
7.     public static TriangleType classifyTriangle(int a, int b, int c) {  
8.         if (a > b) {  
9.             int tmp = a;  
10.            a = b;  
11.            b = tmp;  
12.        }  
13.  
14.        if (a > c) {  
15.            int tmp = c; // original: int tmp = a;  
16.            a = c;  
17.            c = tmp;  
18.        }  
19.  
20.        if (b > c) {  
21.            int tmp = b;  
22.            b = c;  
23.            c = tmp;  
24.        }  
25.  
26.        if (a + b <= c) {  
27.            return TriangleType.INVALID;  
28.        } else if (a == b && b == c) {  
29.            return TriangleType.EQUILATERAL;  
30.        } else if (a == b || b == c) {  
31.            return TriangleType.ISOSCELES;  
32.        } else {  
33.            return TriangleType.SCALENE;  
34.        }  
35.  
36.    }  
37.}
```

Example tools: jacoco, coverage.py, gcov

Interface (graph) Coverage

Web

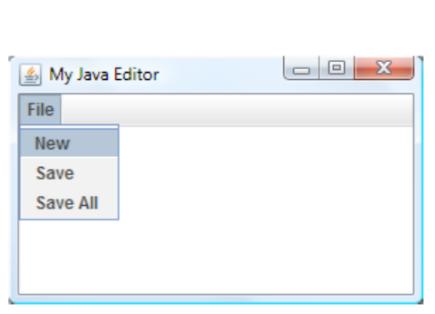


The screenshot shows a web browser window with the URL <https://www.exascaleproject.org/research-group/data-and-visualization>. The page title is "Data and Visualization". It features a sub-section titled "Software Technology Reports" with a "Data and Visualization" section. The text discusses the importance of data management and visualization tools for handling large volumes of simulation data. A red arrow points from the "File" menu in the GUI diagram to this "Data and Visualization" section.

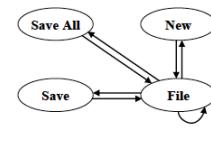
The screenshot shows a web browser window with the URL <https://www.exascaleproject.org/research-group/development-tools>. The page title is "Development Tools". It features a sub-section titled "Software Technology" with a "Development Tools" section. The text discusses the enhancement of existing performance tools and development of new ones for next-generation platforms. A red arrow points from the "File" menu in the GUI diagram to this "Development Tools" section.

The screenshot shows a web browser window with the URL <https://www.exascaleproject.org/research-group/software-technology>. The page title is "Software Technology". It features a sub-section titled "Reports" with a "Data and Visualization" section. The text discusses the ECP's software portfolio for managing and visualizing data. A red arrow points from the "File" menu in the GUI diagram to this "Data and Visualization" section.

The screenshot shows a web browser window with the URL <https://www.exascaleproject.org/about>. The page title is "Overview of the ECP". It features a sub-section titled "The Compelling Case for Exascale Computing" which discusses the role of information technology and applied science engineering in society. A red arrow points from the "File" menu in the GUI diagram to this "The Compelling Case for Exascale Computing" section.



GUI





Other Coverage

Specification coverage

- Cover the system requirements

Interaction coverage

- Measure interactions between components
 - Pairs, n-way coverage

Overview



Types of
Testing



Challenges



Models



Coverage



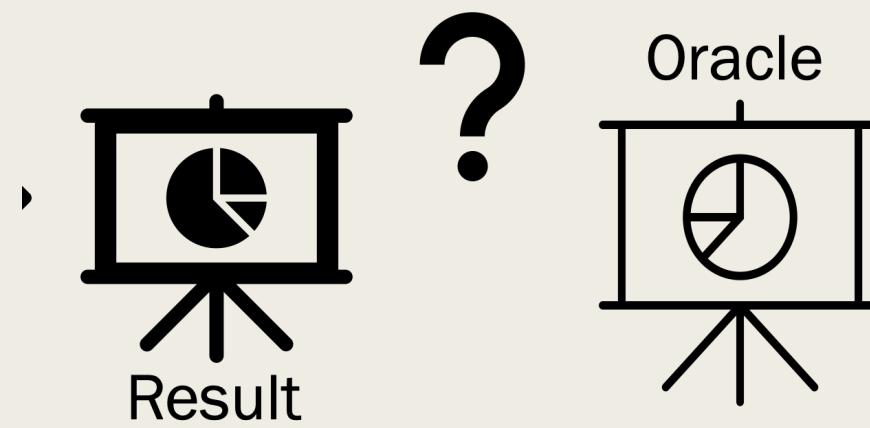
Oracles



Configurability



What is the Correct Answer?





Trivial Oracles

Program crashes

Core dump

Segmentation error

Overflow

Program hangs



Trivial Oracles

- Good when we don't have a known result
- **Weakest oracle** since it only shows that the program fails/not that the result is incorrect
- Exact oracles are easy to compute in some programs

```
def classify_triangle(a, b, c):
    # Sort the sides so that a <= b <= c
    if a > b:
        tmp = a
        a = tmp #fault should be a=b
        b = tmp

    if a > c:
        tmp = a
        a = c
        c = tmp

    if b > c:
        tmp = b
        b = c
        c = tmp

    if a + b <= c:
        return TriangleType.INVALID
    elif a == b and b == c:
        return TriangleType.EQUILATERAL
    elif a == b or b == c:
        return TriangleType.ISOSCELES
    else:
        return TriangleType.SCALENE
```



Harder Oracles

MEGA
HIT Assemble Reads with MEGAHIT v1.1.
Assemble metagenomic reads using the MEGAHIT assembler.

Run Configure Job Status Result

Input Objects

Read Library

Parameters (5 advanced parameters showing) hide advanced

Parameter preset

--min-count

--k-min 1 ≤127

--k-max 1 ≤255

--k-step 1 ≤28

--k-list

--min-contig-len 300 2000

Output Objects

Output Assembly name





Making Oracles Hard

- Results may differ by small **epsilons** (due to rounding)
- Expected result **may not be computable** without program
- May have **time series** results
- Takes a long time to manually compute each oracle (even when we can)
- Programs may be **stochastic** (or non-deterministic)



Examples

Python docs

Note: The behavior of `round()` for floats can be surprising: for example, `round(2.675, 2)` gives `2.67` instead of the expected `2.68`. This is not a bug: it's a result of the fact that most decimal fractions can't be represented exactly as a float. See [Floating Point Arithmetic: Issues and Limitations](#) for more information.

Same growth values?

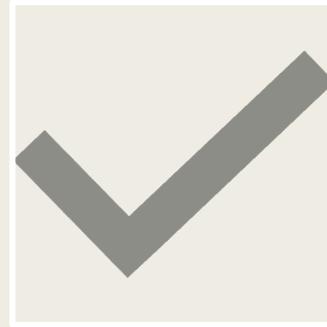
Expected:	0.35695124
Observed:	0.35695122

Correct hits?

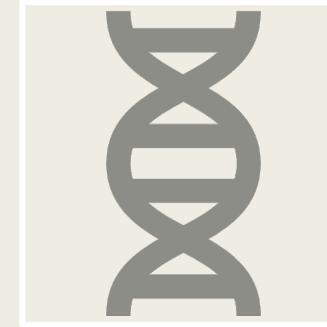
Descriptions	Graphic Summary	Alignments	Taxonomy	Download	Select columns	Show 100	?
Sequences producing significant alignments							
<input checked="" type="checkbox"/> select all 100 sequences selected				GenBank	Graphics	Distance tree of results	MSA Viewer
	Description	Scientific Name	Max Score	Total Score	Query Cover	E value	Per. Ident
<input checked="" type="checkbox"/>	Saccharomyces pastorianus strain CBS 1483 chromosome ScXII	Saccharomy...	1040	1040	100%	0.0	100.00%
<input checked="" type="checkbox"/>	Saccharomyces cerevisiae strain CEN.PK113-7D chromosome XII	Saccharomy...	1040	1040	100%	0.0	100.00%
<input checked="" type="checkbox"/>	Saccharomyces cerevisiae strain ySR128 chromosome XII, complete sequence	Saccharomy...	1040	1040	100%	0.0	100.00%
<input checked="" type="checkbox"/>	Saccharomyces cerevisiae strain V168 chromosome 12	Saccharomy...	1040	1040	100%	0.0	100.00%



Some Techniques



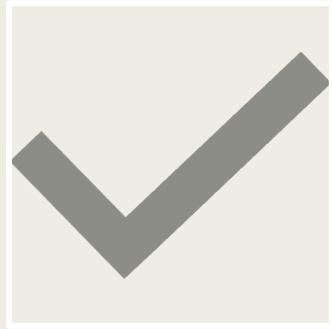
Differential testing



Metamorphic testing



Differential Testing



Differential testing

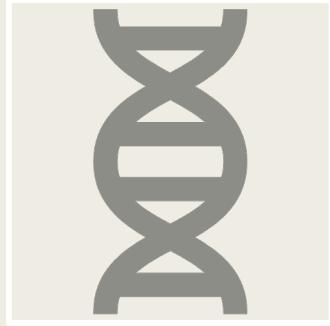
*Run same tests using different programs
that have the same functionality*

- run tests with BLAST
- run tests with HPC-BLAST

Challenge is determining equivalency



Metamorphic Testing



Metamorphic testing

Define relations on sets of tests:

e.g. (subtraction)

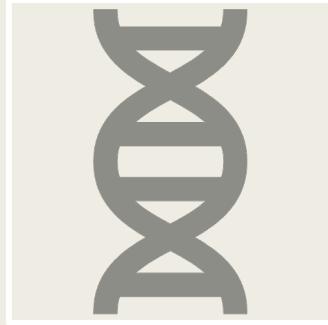
(1) $A - B = C$

Create A' (greater than A)

(2) $A' - B = C'$ means C' is greater than C



Metamorphic Testing



Use Domain Knowledge

e.g. Ocean temperature modeling

A. Compute predicted temperature

B. Modify to increase expected temperature

Confirm relation holds

Metamorphic testing



kbase-all-hands-on
v1 - KBaseFBA.FBA-13.2

Overview

Reaction fluxes

Exchange fluxes

Genes

Biomass

Pathways

Bar charts

ID	myracohen:narrative_1501886992034/kbase-all-hands-on
Object type	KBaseFBA.FBA-13.2
Owner	myracohen
Version	1
Mod-date	2024-02-26T06:15:44+00:00
Objective value	0.507834
Model	E-Coli_k12_MG1
Media	Carbon-D-Glucose
Single KO	0
Number reactions	857
Number compounds	29
Gene KO	0
Reaction KO	0
Custom bounds	0



kbase-all-hands-on-different-media
v2 - KBaseFBA.FBA-13.2

Overview

Reaction fluxes

Exchange fluxes

Genes

Biomass

Pathways

Bar charts

ID	myracohen:narrative_1501886992034/kbase-all-hands-on-different-media
Object type	KBaseFBA.FBA-13.2
Owner	myracohen
Version	2
Mod-date	2024-02-26T06:36:45+0000
Objective value	13.4555
Model	E-Coli_k12_MG1
Media	Complete
Single KO	0
Number reactions	857
Number compounds	29
Gene KO	0
Reaction KO	0
Custom bounds	0

Carbon-D-Glucose <= Complete

Number compounds
Gene KO
Reaction KO
Custom bounds

Overview



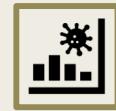
Types of
Testing



Challenges



Models



Coverage



Oracles



Configurability



Configurability



Enter Query Sequence

Enter accession number(s), gi(s), or FASTA sequence(s) 

gactacgatcgggc

[Clear](#)

Query subrange 

From

To

BLAST

Search **database Nucleotide collection (nr/nt)** using **Megablast (Optimize for highly similar sequences)**

Show results in a new window

Algorithm parameters

[Restore default search parameters](#)

General Parameters

Max target
sequences

100 

Select the maximum number of aligned sequences to display 

Short queries

Automatically adjust parameters for short input sequences 

Expect threshold

10 

Word size

28  

Max matches in a
query range

0 

Scoring Parameters

Match/Mismatch

1,-2  

Scores

Gap Costs

Linear  

✓ Linear

- Existence: 5 Extension: 2
 - Existence: 2 Extension: 2
 - Existence: 1 Extension: 2
 - Existence: 0 Extension: 2
 - Existence: 3 Extension: 1
 - Existence: 2 Extension: 1
 - Existence: 1 Extension: 1
- for: Homo sapiens (Human) 

Filters and Mask

Filter

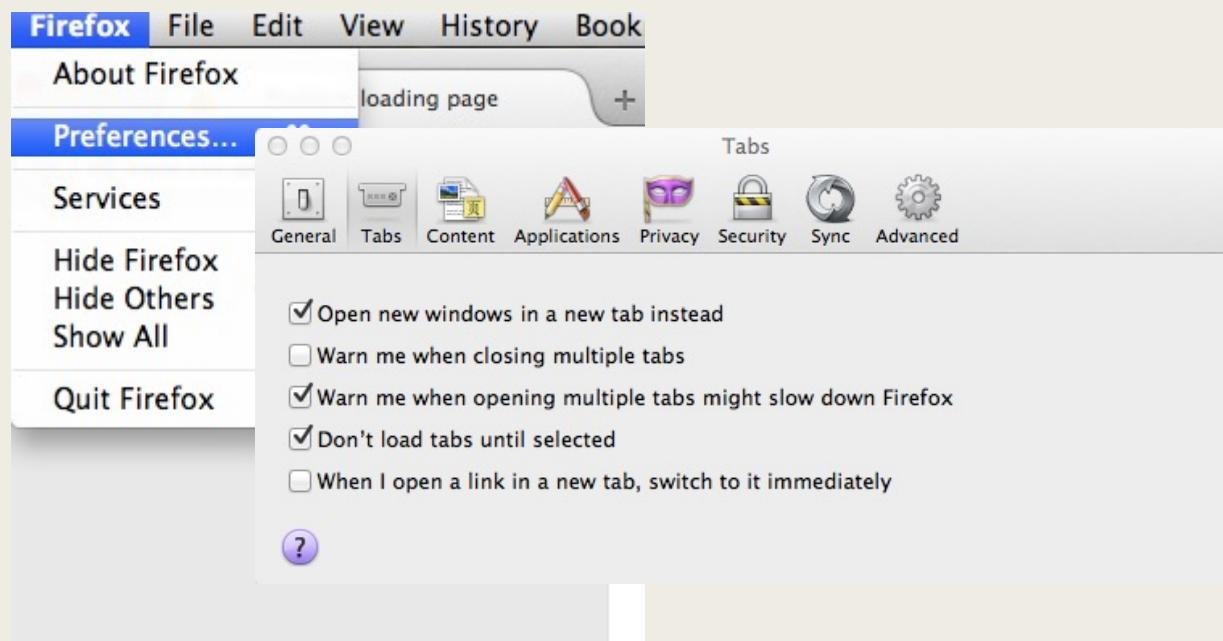
Mask

Mask for lookup table only 

Mask lower case letters 



Configurability





MEGAHIT (DNA Assembler)

- DNA sequenced into small segments (reads)
- Assembly combines reads into longer *continuous sequences*
- Result is a certain number of *continuous sequences*



Credit to Mikaela Cashman

M. Cashman, M. B. Cohen, P. Ranjan, R. W. Cottingham, Navigating the Maze: the Impact of Configurability in Bioinformatics Software, ASE, 2018

**MEGA
HIT** Assemble Reads with MEGAHIT v1.1.1
Assemble metagenomic reads using the MEGAHIT assembler.

Run **Configure** **Job Status** **Result**

Input Objects
Read Library **rhodo.art.q20.PE.reads**

Parameters (5 advanced parameters showing) [hide advanced](#)

Parameter preset

--min-count

--k-min ≤ ≤127

--k-max ≤ ≤255

--k-step ≤ ≤28

--k-list

--min-contig-len ≤

Output Objects
Output Assembly name



**MEGA
HIT** Assemble Reads with MEGAHIT v1.1.1
Assemble metagenomic reads using the MEGAHit assembler.

Run Configure Job Status Result

Input Objects

Read Library **rhodo.art.q20.PE.reads**

Parameters (5 advanced parameters showing) hide advanced

Parameter preset

--min-count	
--k-min	1 ≤ <input type="text"/> ≤ 127
--k-max	1 ≤ <input type="text"/> ≤ 255
--k-step	1 ≤ <input type="text"/> ≤ 28
--k-list	<input type="button" value="+"/>
-min-contig-len	300 ≤ <input type="text"/> 2000

Output Objects

Number of continuous sequences **284**

Configuration Default

53





MEGA
HIT Assemble Reads with MEGAHIT v1.1.1
Assemble metagenomic reads using the MEGAHIT assembler.

Run Configure Job Status Result

Input Objects

Read Library rhodo.art.q20.PE.reads

Parameters (5 advanced parameters showing) hide advanced

Job Status

```
56 [utils.h : 126] Real: 0.4915 user: 0.3962 sys: 0.0720 maxrss: 24688
57 --- [Sat Mar 24 04:35:57 2018] k-max reset to: 119 ---
58 --- [Sat Mar 24 04:35:57 2018] k list: 21,29,39,59,79,99,119 ---
59 --- [Sat Mar 24 04:35:57 2018] Extracting solid (k+1)-mers for k = 21 ---
```

--k-list

-min-contig-len 300 ≤

Output Objects

Number of continuous sequences 284

Configuration Default

**MEGA
HIT** Assemble Reads with MEGAHIT v1.1.1
Assemble metagenomic reads using the MEGAHit assembler.

Run Configure Job Status Result

Input Objects

Read Library: rhodo.art.q20.PE.reads

Parameters (5 advanced parameters showing) hide advanced

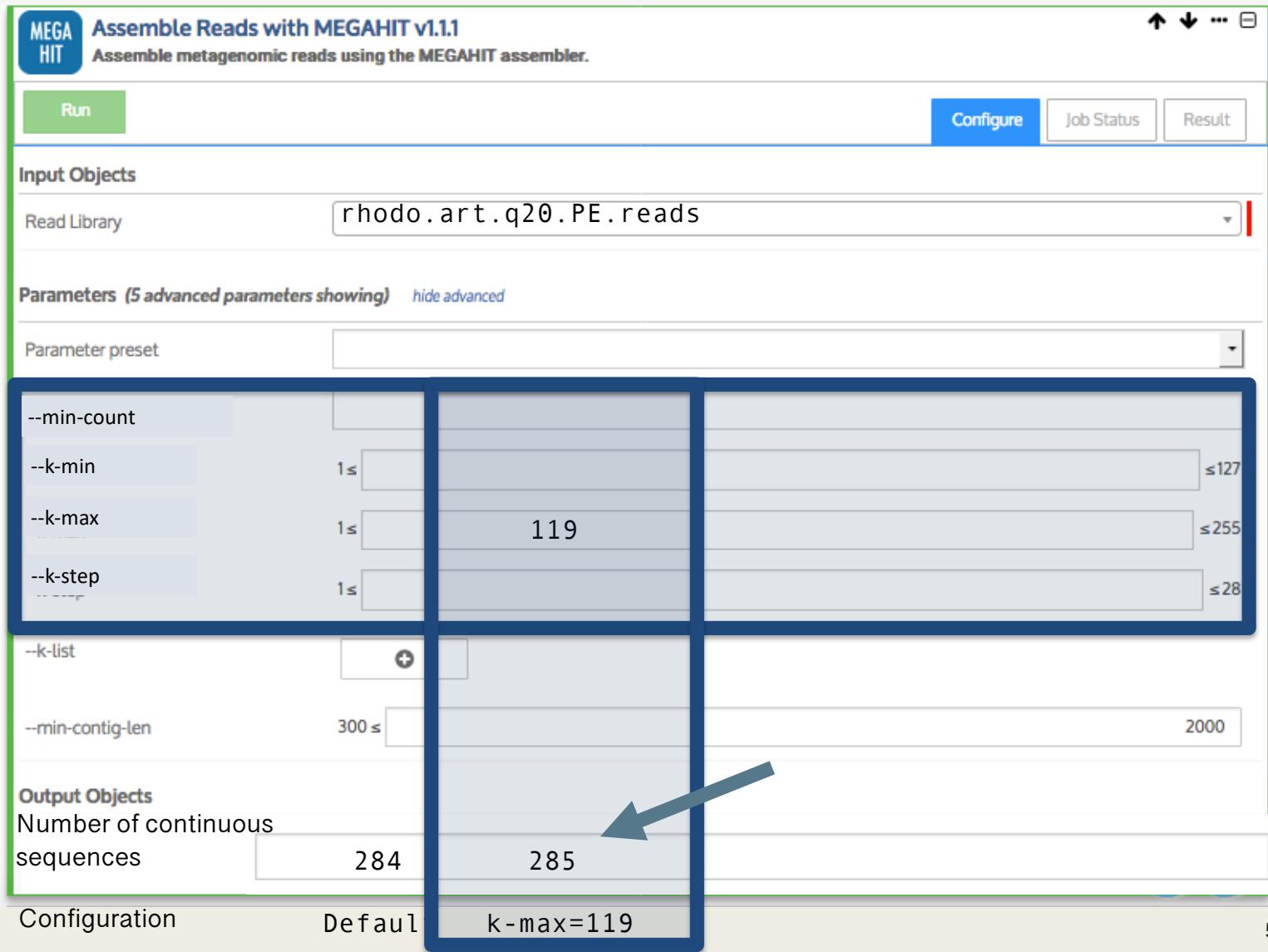
Parameter preset:

--min-count	1
--k-min	1 ≤ 119 ≤ 127
--k-max	1 ≤ 119 ≤ 255
--k-step	1 ≤ 119 ≤ 28
--k-list	+ 119
-min-contig-len	300 ≤ 2000

Output Objects

Number of continuous sequences: 284 → 285

Configuration: Default k-max=119



**MEGA
HIT** Assemble Reads with MEGAHIT v1.1.1
Assemble metagenomic reads using the MEGAHit assembler.

Run Configure Job Status Result

Input Objects

Read Library: rhodo.art.q20.PE.reads

Parameters (5 advanced parameters showing) hide advanced

Parameter preset:

--min-count	2
--k-min	21
--k-max	141
--k-step	12
--k-list	(+)
-min-contig-len	2000

Output Objects

Number of continuous sequences: 284 Default 285 k-max=119 Default-Manual 285

Configuration: Default k-max=119 Default-Manual



**MEGA
HIT** Assemble Reads with MEGAHIT v1.1.1
Assemble metagenomic reads using the MEGAHit assembler.

Run Configure Job Status Result

Input Objects

Read Library: rhodo.art.q20.PE.reads

Parameters (5 advanced parameters showing) hide advanced

Parameter preset:

--min-count	2	2
--k-min	1≤ 21	21
--k-max	1≤ 119 141	99
--k-step	1≤ 12	12
--k-list	+ 127	255
--min-contig-len	300≤ 200	200

Output Objects

Number of continuous sequences: 284 285 285 289

Configuration: Default k-max=119 Default-Manual k -max=99





Assemble Reads with MEGAHIT v1.1.1

Assemble metagenomic reads using the MEGAHIT assembler.



Run

Configure

Job Status

Result

Input Objects

Read Library

rhodo.art.q20.PE.reads

Parameters (5 advanced parameters showing) [hide advanced](#)

Parameter preset

--min-count

2 2

--k-min

1 ≤ 21 21 ≤ 127

--k-max

1 ≤ 119 141 99 ≤ 255

--k-step

1 ≤ 12 12 ≤ 28

--k-list



-min-contig-len

300 ≤ 2000

Output Objects

Number of sequences

284

285

285

289

Configuration





Challenge for Testing

Encoding	Format	Cache Level	Closed-Captioning	Network Access
MPEG	Audio	Low	Yes	Yes
RAW	Video	Medium	No	No
WAV	Stream	High		



Media Player

Encoding	Format	Cache Level	Closed-Captioning	Network Access
MPEG	Audio	Low	Yes	Yes
RAW	Video	Medium	No	No
WAV	Stream	High		



Media Player

Encoding	Format	Cache Level	Closed-Captioning	Network Access
MPEG	Audio	Low	Yes	Yes
RAW	Video	Medium	No	No
WAV	Stream	High		



Media Player

Encoding	Format	Cache Level	Closed-Captioning	Network Access
MPEG	Audio	Low	Yes	Yes
RAW	Video	Medium	No	No
WAV	Stream	High		



Testing the Player

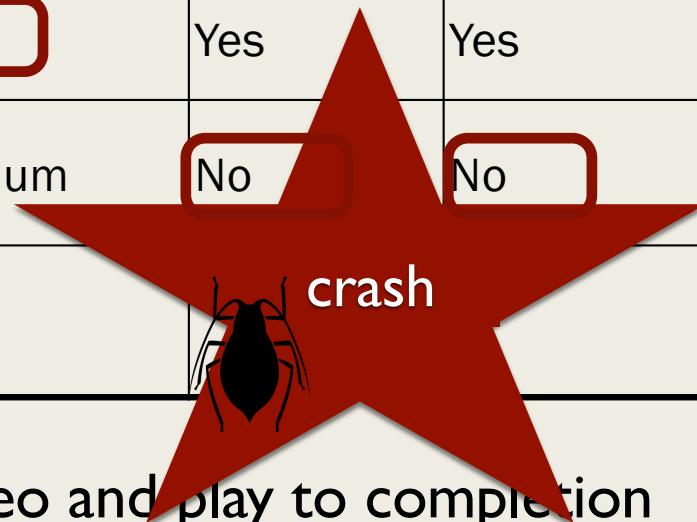
Encoding	Format	Cache Level	Closed-Captioning	Network Access
MPEG	Audio	Low	Yes	Yes
RAW	Video	Medium	No	No
WAV	Stream	High		

Test Case: Open video and play to completion



Interaction Fault

Encoding	Format	Cache Level	Closed-Captioning	Network Access
MPEG	Audio	Low	Yes	Yes
RAW	Video	Medium	No	No
WAV	Stream	High		



Test Case: Open video and play to completion



Configuration-Dependent Security Bugs (CERT)

Overview

Apache Struts 2 framework, versions 2.5 to 2.5.12, with REST plugin insecurely deserializes untrusted XML data. A remote, unauthenticated attacker can leverage this vulnerability to execute arbitrary code in the context of the Struts application.

Description

[CWE-502: Deserialization of Untrusted Data - CVE-2017-9805](#)

In Apache Struts 2 framework, versions 2.5 to 2.5.12, the REST plugin uses `XStreamHandler` with an instance of `XStream` to deserialize XML data. Because there is no type filtering, a remote, unauthenticated attacker may send a specially crafted XML payload to execute arbitrary code in the context of the Struts application.

Refer to the researcher's [blog post](#) for more information about this vulnerability. A [Metasploit module](#) with exploit code is publicly available.

Remove or limit the REST plugin

If it is not used, consider removing the REST plugin. Per the vendor, it is also possible to limit its functionality to normal server pages or JSON with the following configuration change in `struts.xml`:

```
<constant name="struts.action.extension" value="xhtml,,json" />
```



Also Impacts Program Performance

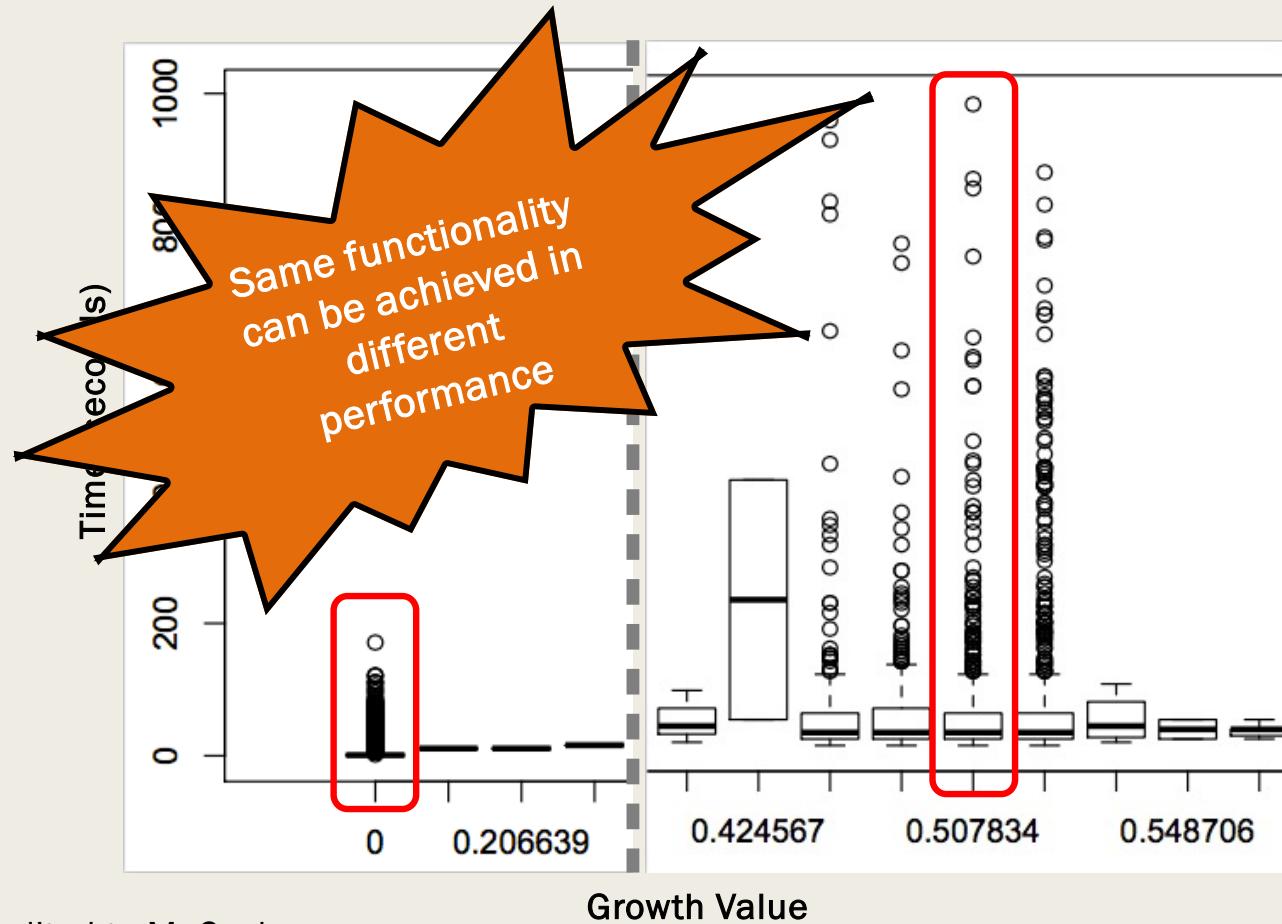


Figure credited to M. Cashman



Configuration Dependence

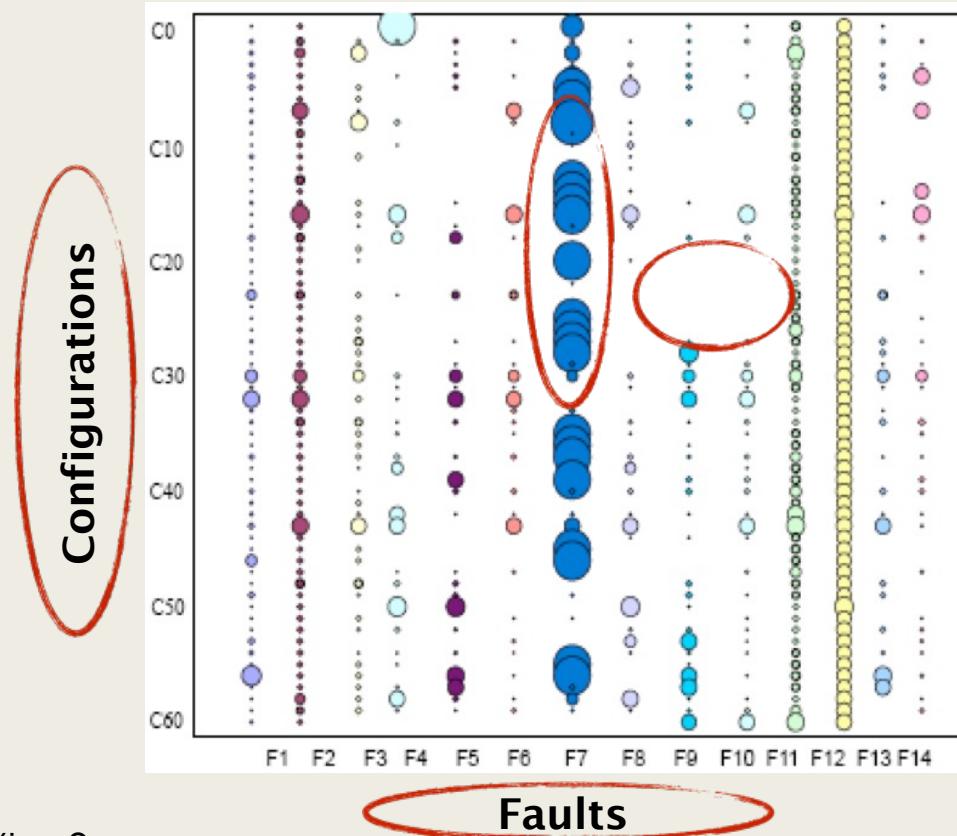


Figure credited to Xiao Qu



Real Configuration Spaces

Encoding	Format	Caching Level	Closed-Captioning	Network Access
MPEG	Audio	Low	Yes	Yes
RAW	Video	Medium	No	No
WAV	Stream	High		



Real Configuration Spaces

$3 \times 3 \times 3 \times 2 \times 2 = 108$ configurations

10 features with 5 options = 9,765,625 configs

4 hours to run test suite —> 4,459 years to run

gcc optimizer (199 options) — 10^{61}

Linux > 10,000 features





Combinatorial Interaction Testing (CIT)

- Sample the space so that all *t-way* combinations of values occur **AT LEAST** once
- *t* is defined as strength of testing



Combinatorial Interaction Testing (CIT)

Encoding		Format	Caching Level	Closed-Captioning	Network Access
1	MPEG	Stream	Medium	Yes	Yes
2	RAW	Video	High	No	No
3	MPEG	Video	Low	No	Yes
4	WAV	Stream	High	No	Yes
5	RAW	Stream	Low	Yes	No
6	MPEG	Audio	High	Yes	No
7	WAV	Video	Medium	Yes	No
8	RAW	Audio	Medium	No	Yes
9	WAV	Audio	Low	Yes	Yes



Combinatorial Interaction Testing (CIT)

	Encoding	Format	Caching Level	Closed-Captioning	Network Access
1	MPEG	Stream	Medium	Yes	Yes
2	RAW	Video	High	No	No
3	MPEG	Video	Low	No	Yes
4	WAV	Stream	High	No	Yes
5	RAW	Stream	Low	Yes	No
6	MPEG	Audio	High	Yes	No
7	WAV	Video	Medium	Yes	No
8	RAW	Audio	Medium	No	Yes
9	WAV	Audio	Low	Yes	Yes



Combinatorial Interaction Testing (CIT)

	Encoding	Format	Caching Level	Closed-Captioning	Network Access
1	MPEG	Stream	Medium	Yes	Yes
2	RAW	Video	High	No	No
3	MPEG	Video	Low	No	Yes
4	WAV	Stream	High	No	Yes
5	RAW	Stream	Low	Yes	No
6	MPEG	Audio	High	Yes	No
7	WAV	Video	Medium	Yes	No
8	RAW	Audio	Medium	No	Yes
9	WAV	Audio	Low	Yes	Yes



Combinatorial Interaction Testing (CIT)

	Encoding	Format	Caching Level	Closed-Captioning	Network Access
1	MPEG	Stream	Medium	Yes	Yes
2	RAW	Video	High	No	No
3	MPEG	Video	Low	No	Yes
4	WAV	Stream	High	No	Yes
5	RAW	Stream	Low	Yes	No
6	MPEG	Audio	High	Yes	No
7	WAV	Video	Medium	Yes	No
8	RAW	Audio	Medium	No	Yes
9	WAV	Audio	Low	Yes	Yes



Some Combinatorial Testing Tools

ACTS

National Institutes of Standards

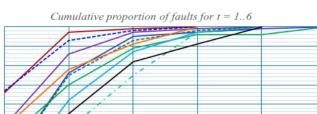
Combinatorial Methods for Trust and Assurance

f t in e

Overview

Combinatorial methods reduce costs for testing, and have important applications in software engineering:

- **Combinatorial or t-way testing** is a proven method for better testing at lower cost. The key insight underlying its effectiveness resulted from a series of studies by NIST from 1999 to 2004. NIST research showed that most software bugs and failures are caused by one or two parameters, with progressively fewer by three or more, which means that combinatorial testing can provide more efficient fault detection than conventional methods. [Multiple studies](#) have shown fault detection equal to



PROJECT LINKS

- Overview
- FAQs

ADDITIONAL PAGES

- Quick start
- Downloadable Tools
- Tutorials and Documentation

[microsoft/pict](#) Public

[Code](#) [Issues](#) 12 [Pull requests](#) 1 [Discussions](#) [Actions](#) [Projects](#) [Security](#) [Insights](#)

[main](#) [3 Branches](#) [5 Tags](#) [Code](#)

[apodhrad](#) Containerize the pict tool (#114) [...](#) ✓ a3d373b · 6 months ago [138 Commits](#)

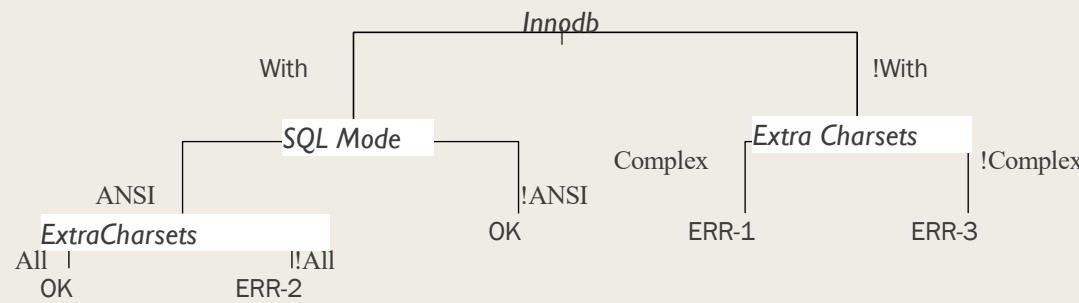
[.github/workflows](#) Update nuget-publish.yml, upgrade checkout action to v3 last year

[api-usage](#) Upgrade projects to VS2022, bump the ver number. (#106) last year



Characterizing Test Failures

- Use machine learning techniques (classification trees) to model option setting patterns that explain test failures



Summary

Overview



Types of Testing



Challenges



Models



Coverage



Oracles



Configurability



Some Techniques



Differential testing



Metamorphic testing

The screenshot shows a software interface with a table of model details. Key fields include:

- ID: mpraatkhanabave_30
- Object type: mpraatkhanabave_30
- Owner: mpraatkhanabave
- Version: 1
- Model date: 2020-02-26T06:34:45+0000
- Object type: mpraatkhanabave_30
- Owner: mpraatkhanabave
- Model date: 2020-02-26T06:34:45+0000



Configurability



This work was supported in part by the Better Scientific Software Fellowship Program, funded by the Exascale Computing Project (17-SC-20-SC), a collaborative effort of the U.S. Department of Energy (DOE) Office of Science and the National Nuclear Security Administration; and by the National Science Foundation (NSF) under Grant Nos. 2154495, 2234908, 1909688

Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the DOE or NSF.