A Search System for Mathematical Expressions on Software Binaries

Ridhi Jain
Sai Prathik
Venkatesh Vinayakarao
Rahul Purandare
Indraprastha Institute of Information Technology, Delhi
Introduction

Is there any C library to implement FFT?

Are there any libraries in C to implement FFT's?

C++ Math Library

4,450 results
Motivation

Developers search for Math Expressions

Inverse Chi Square Function in C#

\[ H = C^{-1}(-2 \ln \prod_{\omega} f(\omega), 2n) \]

This is the equation of what I want to implement.

I have already made the inside up to comma, but I have really no idea how to implement \( C^{-1} \) with 2n degrees of freedom, is there any function in Math class?

Looking at the description on the wiki I am still confused like a lil kid. [wikipedia inverse chi function]

I have found the Python implementation: [Python implementation of chi function]

No search systems for binaries!
System Overview

Mathematical Expression

\[
\frac{1}{\sqrt{2\pi \sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}
\]

Search System

Math Strands KB

Binaries

An Overview of the System
Can we locate a given Mathematical Expression in a given Software Binary?
Challenges in Handling Binaries

Optimization -O0

callq 4806c0 <pow@plt>

Optimization -O2 and -Os

mulsd %xmm0,%xmm0

Variants

Ghost Ops

Evaluation

Ordering

Operand

Resolution

X^2

\[ b^2 - 4ac \]

\[ a^2 - 4ac \]
Challenges in Working with ME

\[ xy = x*y = xy \]

Content MathML normalizes it

```
<apply>
  <times/>
  <ci>x</ci>
  <ci>y</ci>
</apply>
```

Specification of Operation

Types of Operations

- Optimization -O0
  - `callq 4006c0 <log@plt>`

- Optimization -O2 and -Os
  - `callq 400680 <_ZNSo9_M_insertIdEERSoT@plt>`
Related Work

- Tracelet-based Code Search in Executables
  PLDI’ 14

- Retrieving Documents with Mathematical Content
  SIGIR’ 13

- Detecting Code Clones in Binary Executables
  ISSTA’ 09

- Work for search in software binaries

- Retrieves ME in documents

- No comprehensive solution for searching ME in software binaries!!
Key Contributions

- A search system to search for ME in binaries.
- An approach to compare binaries and ME.
- A knowledge base of math operators mapped to their assembly opcodes.
Components of the System

Math Strands Knowledge Base ($M_s$ KB):

- $x^2$, pow, mulsd
- $e^x$, exp
- $\sqrt{x}$, sqrt
- ...

Binary Fingerprint Generator ($B_{fp}$):

- Binary Fingerprint Generator

Mathematical Expression Fingerprint Generator ($M_{fp}$):

Mathematical Expression Generator

- Mathematical Expression
- $\frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$

Math Fingerprint

- $/\sqrt{**}e^-/**$

Fuzzy Scorer:

Fuzzy Scorer

- Match Score
Approach
Evaluation: Selection of Expressions

- We focus on expressions whose fundamental building blocks are generally available in the in-built language features of high-level languages such as C, C++, and Java.
- We select 4 ME listed as follows:

<table>
<thead>
<tr>
<th>Name</th>
<th>Expression</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Heron’s Formula</td>
<td>$\sqrt{(s(s-a)(s-b)(s-c))}$</td>
<td>Algebraic</td>
</tr>
<tr>
<td>Sigmoid Function</td>
<td>$\frac{1}{1+e^{-x}}$</td>
<td>Transcendental</td>
</tr>
<tr>
<td>Compound Interest</td>
<td>$K_n = K_0 \times (1+(P/100))^n$</td>
<td>Algebraic</td>
</tr>
<tr>
<td>D1-Black-Scholes</td>
<td>$(\log(s_0/x)+t(r-q+(\sigma^2/2)))/(\sigma\sqrt{t})$</td>
<td>Transcendental</td>
</tr>
</tbody>
</table>

- We convert the selected ME to ContentML using Visual Math Editor.
Evaluation: Selection of Projects

- We pick 20 projects from GitHub from distinct domains such as Finance, Algebra, and Machine Learning.
- For each of the ME of interest, we pick 5 C/C++ projects.
- Next, we compile all 20 programs with gcc/g++ optimization levels: -O0, -O2, -Os
- As a result we had 60 binaries in which we searched for all the expressions.
Heuristics

● We use **Longest Common Subsequence** algorithm to find generate the match score for Binary Fingerprint (B\(_{fp}\)) and Mathematical Expression Fingerprint (M\(_{fp}\)), LCS(M\(_{fp}\), B\(_{fp}\)).

● LCS(M\(_{fp}\), B\(_{fp}\)) > \(\sigma\) implies that ME was found in the binary.

● To improve the precision we use two heuristics:
  ○ **Length Heuristics for short ME**: If |M\(_{fp}\)| < \(\alpha\) |B\(_{fp}\)| then ME \(\notin\) binary.
  ○ **Relevance Heuristics for Ghost Ops**: (Irr(B\(_{fp}\), M\(_{fp}\))) = |s| such that strand s \(\in\) B\(_{fp}\) and s \(\notin\) M\(_{fp}\). If (Irr(B\(_{fp}\), M\(_{fp}\))) > \(\beta\) |B\(_{fp}\)| then ME \(\notin\) binary.

● We derive the parameters \(\alpha\) and \(\beta\) empirically to be 0.25 and 0.40 respectively.
Results

- We compute precision and recall for all the binaries for all ME
- We compare our results against ground truth for various values of threshold
- $F_1$ score peaks to 0.61 for a threshold value, $\sigma = 0.45$
- With our approach we were able to identify ME in software binaries with an average precision of 80% and a recall of 53%.
Results

<table>
<thead>
<tr>
<th>ME</th>
<th>$M_{fp}$</th>
<th>Precision</th>
<th>Recall</th>
<th>F&lt;sub&gt;1&lt;/sub&gt; Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Heron’s Formula</td>
<td>-**-**q</td>
<td>0.69</td>
<td>0.56</td>
<td>0.61</td>
</tr>
<tr>
<td>Sigmoid Function</td>
<td>e+/-</td>
<td>1.00</td>
<td>0.54</td>
<td>0.70</td>
</tr>
<tr>
<td>Compound Interest</td>
<td>/+^*</td>
<td>0.50</td>
<td>0.62</td>
<td>0.56</td>
</tr>
<tr>
<td>D1-Black-Scholes</td>
<td>/l^/+-<em>+q</em>/</td>
<td>1.00</td>
<td>0.38</td>
<td>0.55</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td><strong>0.80</strong></td>
<td><strong>0.53</strong></td>
<td><strong>0.61</strong></td>
<td></td>
</tr>
</tbody>
</table>
Results

$F_1 = 0.61$ at $\sigma = 0.45$, $\alpha = 0.25$, $\beta = 0.40$

Peaks at $\sigma = 0.45$
Conclusion

- Mathematical expressions and software binaries pose different challenges.
- We solve these problems using data driven fingerprinting based approach.
- We show that ME can be located in binaries with 61% $F_1$ score for algebraic and transcendental expressions.
Future Work

- We envision automating the creation of $M_s$ KB for multiple system architectures.
- Classes of operations that still remain to be explored, such as, logical and relational.
- Iterative operations such as Summation ($\sum$) and Product ($\prod$) require the operation to be applied over a range of values.
- Precision and Recall can be improved by keeping track of the operands in the ME.
Thank You

Questions?