Difficulties in Understanding Library Code

CSE 504 Assignment 1

Qiao Zhang (qiao@cs.washington.edu)

Auto-generate Tutorials (text + (pseudo-)code snippets)

For reusability, library API is designed to cover many use cases, e.g. (A)Sync

Hard to find which functions to call for a particular use case e.g. only Sync

Tutorials often help, but may not exist yet => **Can we auto-generate them?**

Data: user programs, test programs, Stack Overflow Q&A, API doc

Algorithms:

1. cluster sample programs by API calls
2. infer use case of each cluster (Stack Overflow question: I'm trying to do ***)
3. generate tutorial (text + code snippet) for each use case
4. verify code snippet by testing

Auto-generate/Augment Architectural Documentation

Library API is sometimes well-documented (thanks to Doxygen)

But no top-down view of a codebase => **Can we auto-generate Arch Doc / augment README files?**

Data: class diagrams, API doc, user programs (to highlight important APIs)

Algorithms:

1. Write good arch doc templates (or learn it)
2. Mine for each piece of the doc from data

Distill skeletal code

Library code is often designed to be general (e.g. many if branches) and almost always correct to full details (e.g. close an open file before exit)

A sketch of the code (or hybrid code+text) can ease understanding, e.g. remove corner cases, remove boiler code, replace code snippet with text description

Data: version history, user program, API doc, inline doc, class diagram

Algorithms:

1. version history+user program can highlight common API/code path
2. API/inline doc provide text descriptions for code snippets
3. semantically clear function names can provide text descriptions