**Week 5: Review and Analyze Conference Venue Review**

Nate Bachmeier

April 7th, 2019

TIM-8101: Principals of Computer Science

Northcentral University

# Describe the types of papers presented/research conducted at the venue

The Automated Software Engineering conference focuses on tooling and frameworks for simplifying engineering tasks.

# Describe each track and types of papers that are submitted to each track

* Research Papers
  + A forum for original and unpublished results
* Journal-First
  + A forum to discuss published papers which have not been demonstrated yet
* Demonstrations
  + Discussions around tooling and concrete implementations
* Industry Showcase
  + Discussions around the application of software engineering
* Doctoral Symposium
  + Forum for students to present their research with the community experts

# Paper 1: Cobra – An Interactive Static Code Analyzer

## What problems did they solve?

Custom static analysis tools are time consuming to write and then can take hours to run across large code bases (Holzmann, 2017). COBRA reduces the required time down to a few seconds by using efficient data structures.

## What was their method?

A lexical parser converts the source code into a token stream which is represented by a linked list. If the token declares the start of a scope, then it will contain a pointer to the token which completes the same level. This allows for a range of tokens to be rapidly fetched or discarded.

Users can then perform read-only queries across the token stream by specifying sequences of regular expression and viewing the matches in an interactive console.

## What are areas of future work/improvement?

An indexing feature could be added to the linked list to expand the size of repositories supported. It might also be useful to encode inheritance information as pointers in the linked list.

## What other works does this expand?

This effort expands on decades of static analysis and combines it with tooling that is inspired by grep and awk.

## Why is this important?

Having the ability to rapidly query large bodies of code enables code examples to be quickly found. It can also be used to identify multiple locations that share the same defect.

# Paper 2: Mining Structures from Massive Text Data

## What problems did they solve?

The Internet is full of unstructured textual content and a series of transforms are required to make it semi-structured for consumption by applications (Han, 2017).

## What was their method?

Han described the process for combining words into phrases, then clustering them into topics. The clustering is based on popularity, concordance, informativeness, and completeness.

Next entities are identified along with their type and co-type which describe their usage. Then meta patterns are detected from the usage of entities within the sentences. The meta patterns becomes attributes of the data.

Attributes are used to further identify more dimensions and tags to annotate the data fragment into text cubes.

## What are areas of future work/improvement?

## What other works does this expand?

Mining structure from massive text relies on the collaboration of many published research efforts. The presenter describes a framework to combine these disjoined efforts into a single workflow process.

The process begins with identifying words and then performing statistical operations to combine words into phrases. Next topic mining was performed to cluster the phrases. Then entities are identified along with their type and co-type. Entities and topics are then attributed with metadata recursively as rich context allows the discovery of more context. The result of this process is a multi-dimensional knowledge graph. It can be explored with standard graph traversal strategies.

## Why is this important?

# Paper 3: Software Without Borders

## What problems did they solve?

Log analysis is a complex task of finding the needles in a haystack (Van Deursen, 2017). It takes a lot of time and effort to perform these tasks.

## What was their method?

Adyen’s global web presence collects 1 billion Java Exception objects daily. A feature extraction process is then performed to cluster these exceptions into unique issues. For each issue stack traces are randomly sampled and run through a custom symbolic execution engine. The executor uses reinforcement learning to steer the symbolic executor into a reproduction of the failure. Finally, a defect is opened for the development team with a short script to trigger the fault.

## What are areas of future work/improvement?

If the system can identify the root cause of the defect, why can’t it also patch simple issues? For example, if argument checking is enough then add the check and send a patch.

## What other works does this expand?

## Why is this important?

Automated root cause analysis is important as they address common bottlenecks in the software development lifecycle. The less time developers need to spend hunting defects, the more time they can spend on new features

# References

Han, J. (2017, November 1). *Mining Structures from Massive Text Data.* Retrieved from YouTube: https://www.youtube.com/watch?v=jy-Rg9r2lBQ

Holzmann, G. (2017, October 31). *COBRA - an Interactive Static Code Analyzer.* Retrieved from YouTube: https://www.youtube.com/watch?v=mVkz-ETpb0w

Van Deursen, A. (2017, November 2). *Software Engineering without Borders.* Retrieved from YouTube: https://www.youtube.com/watch?v=BxizjBmHXdA