Discovering causal links in log files using machine learning

Vithor Bertalan
Prof. Daniel Aloise

Polytechnique Montréal

DORSAL Laboratory

Introduction

- Machine learning is very good with labeling and clustering data
- But not as good with tracking the root cause of data
- Therefore, we can we do to identify data responsible for creating other data?

Object of Study

- Log files contain fundamental information about the execution of systems
- Modern software systems routinely generate a large volume of logs
- How to know which of the logs have a higher potential of causing further problems?

Causal Structure Learning

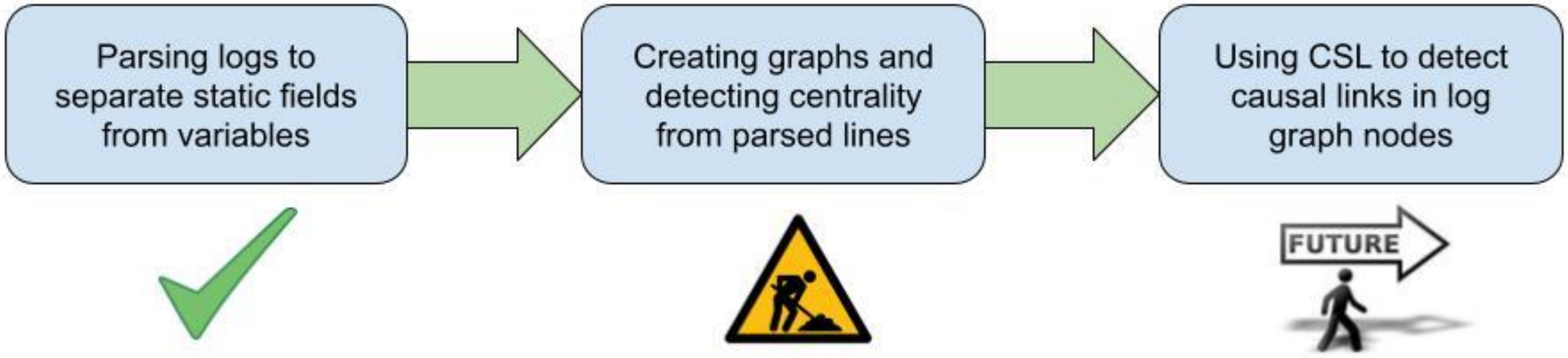
- A new field emerges: Causal Structure Learning (CSL)

- Causality inference is basically *identifying the effect of an experimental intervention given observational data or a combination of observational and interventional data*. (Kalisch, 2014).

- The main goal of the area is to track the consequence of inserting/altering a single piece of data in a dataset.

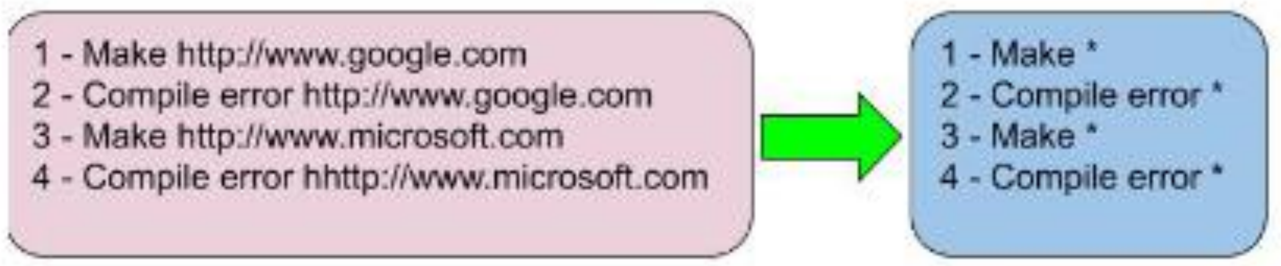
Research Structure

- In the first step of our research, we developed a new parsing method, to separate static fields from variables in log files.
- In the second step, we will create graphs and detect centrality from the parsed log lines.
- In the third step we will use CSL techniques to identify causality in log lines.


Parsing logs to separate static fields from variables
Creating graphs and detecting centrality from parsed lines
Using CSL to detect causal links in log graph nodes
FUTURE

Research Part 1

- After log parsing, the *variable tokens* carry the most important information to track causality among log lines.



```
1 - Make http://www.google.com
2 - Compile error http://www.google.com
3 - Make http://www.microsoft.com
4 - Compile error hhttp://www.microsoft.com
```

```
1 - Make *
2 - Compile error *
3 - Make *
4 - Compile error *
```

Research Part 1

- Thus, we have to develop/adapt a technique to save variable tokens, instead of replacing them
- However, the most popular log parsing techniques use small inputs, instead of processing the whole text as a batch
- Using Transformers might be a good alternative

Parsing Method

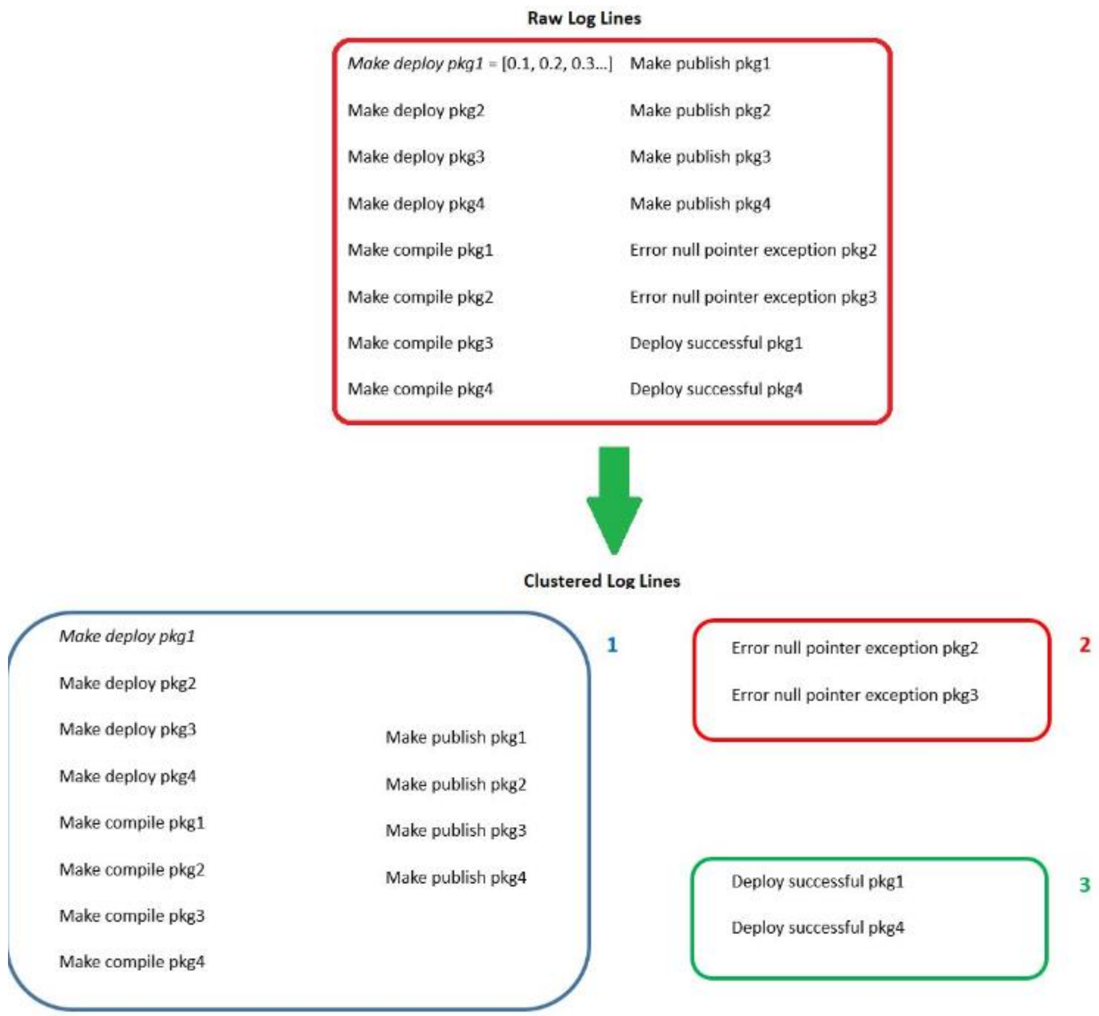
1. Pre-processing raw log texts, using state-of-the-art tokenization techniques

2. Creating transformer embeddings from raw log texts using RoBERTa

3. Clustering data into smaller subsets

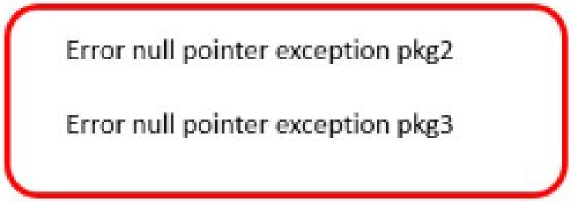
Why cluster the logs?

- Separating common log sequences



Frequency Count

- After clustering, we count the frequencies of tokens in each cluster.
- Then, we define a *parsing threshold* that separates static fields from variable tokens.



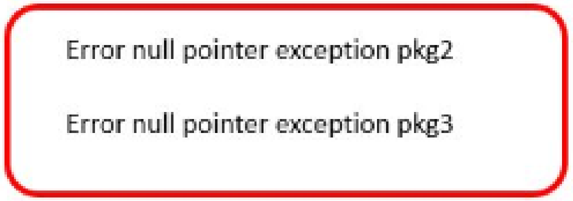
```
Error null pointer exception pkg2

Error null pointer exception pkg3
```

$$rf(i) = \frac{f(i)}{max_{i \in C} f(i)},$$

Token	Cluster	Frequency	Relative Frequency
Error	2	2	1
Null	2	2	1
Pointer	2	2	1
Exception	2	2	1
Pkg2	2	1	0.5
Pkg3	2	1	0.5

English Vocabulary

- After the definition of frequencies, we check if the word in present in the English vocabulary.
- If “pkg2” and “pkg3”, not English words, were not parsed in the previous step, they are going to be parsed now.

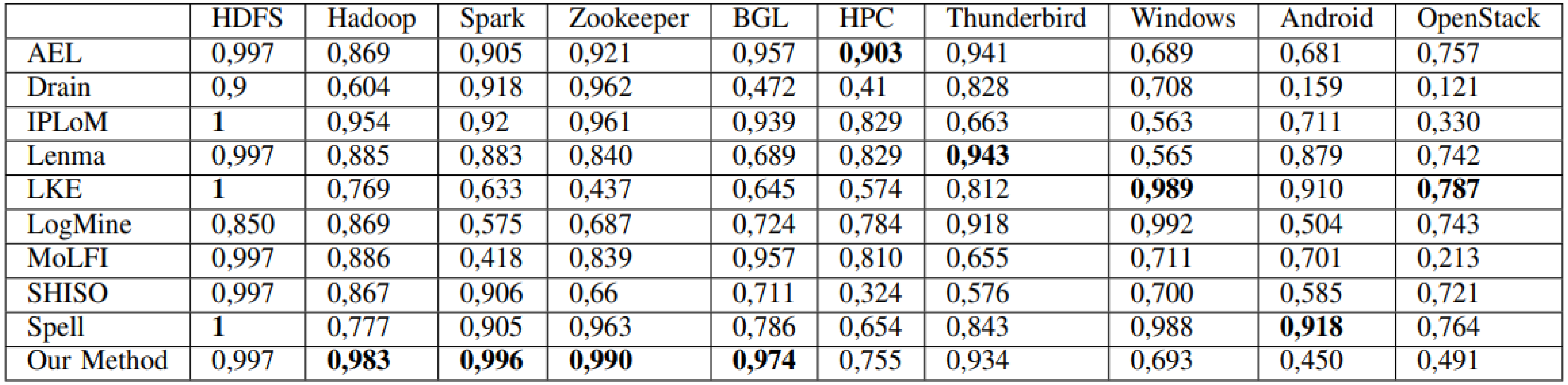


```
Error null pointer exception pkg2

Error null pointer exception pkg3
```

Results of Part 1

- Using LogHub's accuracy standards, we are able to have the highest accuracy in 4 of 17 LogHub's datasets.

Accuracies of the methods using LogHub's framework. Top values in each column in bold.



	HDFS	Hadoop	Spark	Zookeeper	BGL	HPC	Thunderbird	Windows	Android	OpenStack
AEL	0,997	0,869	0,905	0,921	0,957	**0,903**	0,941	0,689	0,681	0,757
Drain	0,9	0,604	0,918	0,962	0,472	0,41	0,828	0,708	0,159	0,121
IPLoM	**1**	0,954	0,92	0,961	0,939	0,829	0,663	0,563	0,711	0,330
Lenma	0,997	0,885	0,883	0,840	0,689	0,829	**0,943**	0,565	0,879	0,742
LKE	**1**	0,769	0,633	0,437	0,645	0,574	0,812	**0,989**	0,910	**0,787**
LogMine	0,850	0,869	0,575	0,687	0,724	0,784	0,918	0,992	0,504	0,743
MoLFI	0,997	0,886	0,418	0,839	0,957	0,810	0,655	0,711	0,701	0,213
SHISO	0,997	0,867	0,906	0,66	0,711	0,324	0,576	0,700	0,585	0,721
Spell	**1**	0,777	0,905	0,963	0,786	0,654	0,843	0,988	**0,918**	0,764
Our Method	0,997	**0,983**	**0,996**	**0,990**	**0,974**	0,755	0,934	0,693	0,450	0,491

Pratical Utilization of Parsing

- Our method is completely unsupervised, only requiring the parsing threshold.
- It does not require the utilization of regular expressions, making the method dataset-agnostic.
- Results submitted to ISSRE'23.

Research Part 2

- In a graph, we presume that the most central nodes are the most relevant to the occurrence of descendant ones.
- Thus, we have to generate a graph from our log lines and track the centrality of each of the nodes.
- However, creating graphs from non-natural languages is not trivial.

Research Part 2

- The most central log nodes in a graph are the most relevant to the occurrence of descendant (posterior and consequential) log lines.

- Some recent work in the literature point to eigenvector centrality as a possible indicator of causality (Dablander and Hinne, 2019).

- No work using centrality in the field of log lines, to the best of our knowledge.

Research Part 2

- However, causality is not the only thing that we can extract from the centrality.

- Indicating the most central nodes in a log dataset could work as an indicator that a situation deserves more attention from a software developer or systems administrator.

- Our method could highlight those important lines. But converting raw log lines to graph is not trivial.

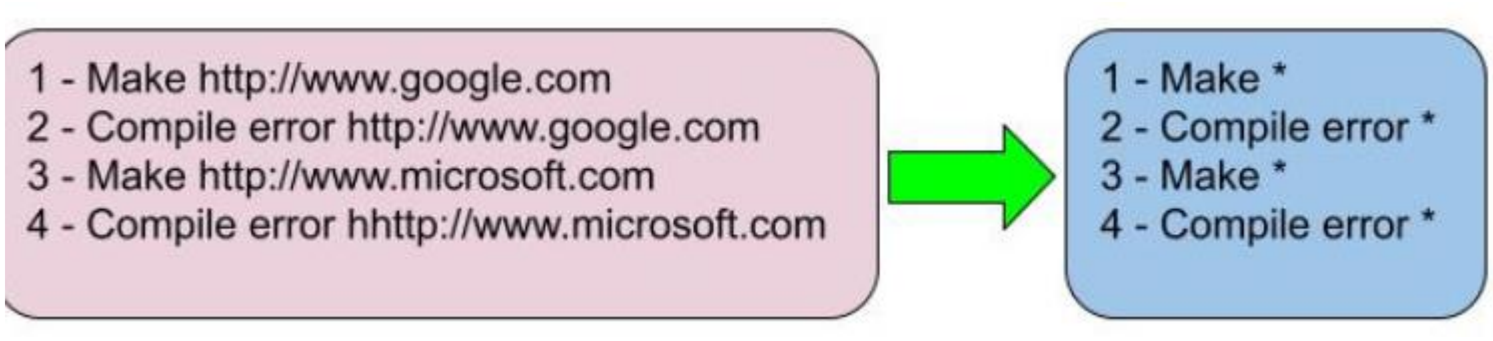
Research Part 2

- Not only the centrality can be used to highlight the most important lines, but other useful metadata information as well

- We plan to use other information, such as the timestamp of the log lines or textual information about the lines, such as the line length or PoS tagging in the possible words.

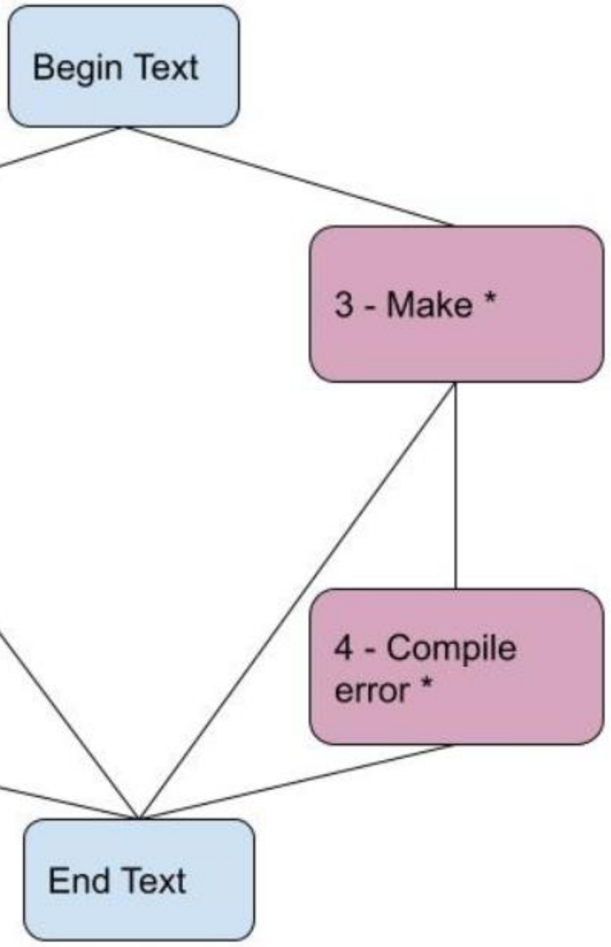
Research Part 2

- We can use the parsed logs from the previous step to create our graphs. First, we parse:


1 - Make http://www.google.com
2 - Compile error http://www.google.com
3 - Make http://www.microsoft.com
4 - Compile error hhttp://www.microsoft.com
1 - Make *
2 - Compile error *
3 - Make *
4 - Compile error *

Research Part 2

- Then, we connect the elements with common variable tokens, and track centrality.



Research Part 2

- To the best of our knowledge, there is no centrality datasets to compare our results to.
- Therefore, we plan to do a written analysis to explain our methods, and explain which centrality methods we adopted, and why.
- We also plan a proof of concept, in which we make publicly available a dataset which the centralities found by our methods.

Research Part 3

- CSL is a very recent field of study that tries to identify possible causality links in an undirected graph. Therefore, converting it into a directed graph (DAG).

- Is has been very used in the field of bioinformatics, but not in the field of log mining, to the best of our knowledge.

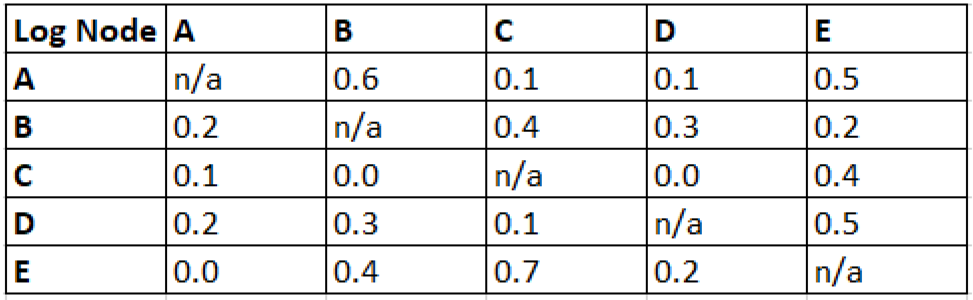
- Some algorithms, such as the Peter-Clark (PC) algorithm, have become the golden standard of the field.

Research Part 3

- Adopting CSL, using the parsed logs and the centrality of log lines created from previous steps as possible inputs.
- CSL shows good results in graphs, detecting arcs (directed vertices) between nodes.
- Our final goal: a matrix of probabilities of causalities between each pair of nodes.

Research Part 3

- We plan to create a matrix of causality, weighting our centrality measures and the results from the CSL algorithms between all pairs of log nodes.



Log Node	**A**	**B**	**C**	**D**	**E**
A	n/a	0.6	0.1	0.1	0.5
B	0.2	n/a	0.4	0.3	0.2
C	0.1	0.0	n/a	0.0	0.4
D	0.2	0.3	0.1	n/a	0.5
E	0.0	0.4	0.7	0.2	n/a

Research Part 3

- We intend to discuss several different methods in our conclusion, so as to determine which of the methods were the best overall

- As in the second hypothesis, we plan to produce a proof-of-concept dataset of causality between log lines, since there is not dataset publicly available with this goal.

References and Questions

- Dablander, Fabian, and Max Hinne. "Node centrality measures are a poor substitute for causal inference." Scientific reports 9.1 (2019): 1-13.
- M. Kalisch and P. Bühlmann, “Causal structure learning and inference: A selective review,” Quality Technology and Quantitative Management, vol. 11, no. 1, pp. 3–21, 2014, doi: 10.1080/16843703.2014.11673322.