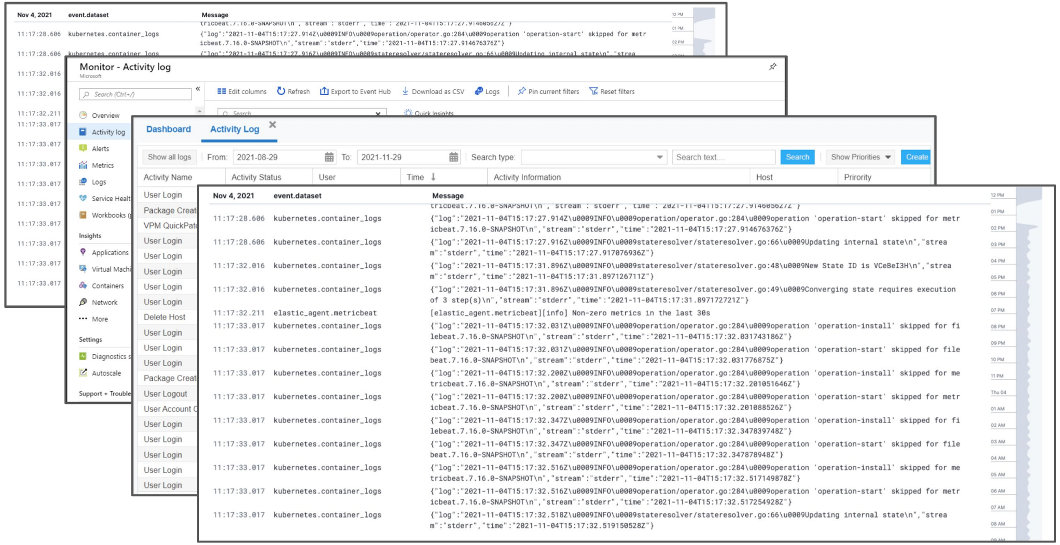
Log Parsing Evaluation in the Era of Modern Software Systems

Stefan Petrescu, Floris den Hengst, Alexandru Uta, and Jan S. Rellermeyer

To be published at ISSRE '23

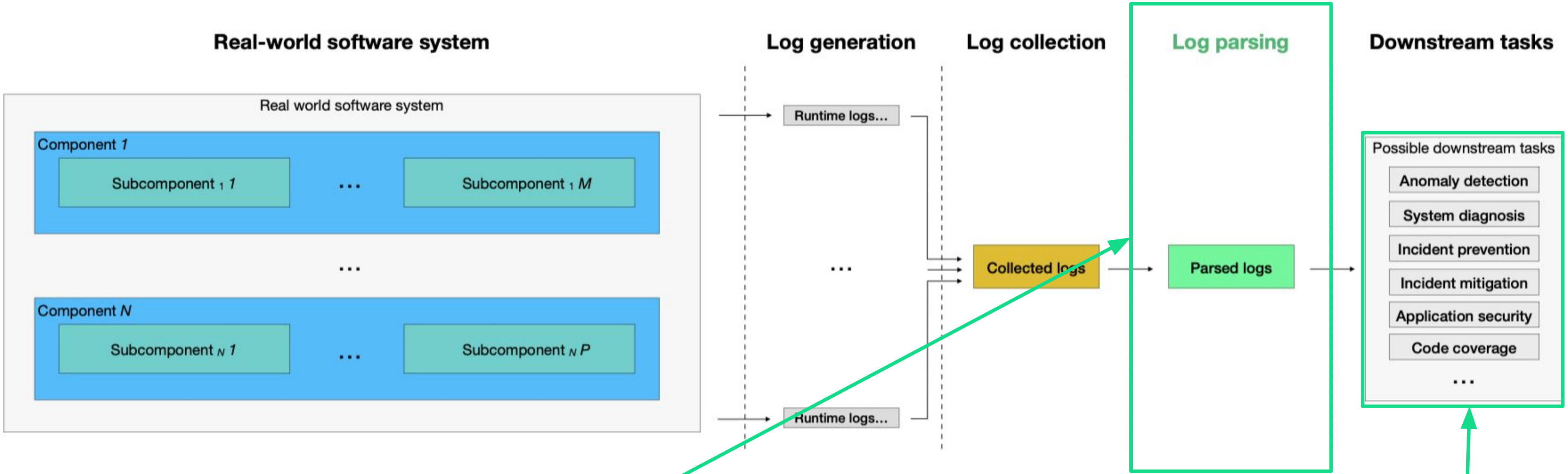
Bamberg, September 29, 2023

Logs in Modern Organizations



- TBs of logs daily
- Thousands of machines & different software
- High complexity

Logs, Downstream Tasks, Log Parsing



Log parsing is a critical step

The Good

- Many log parsing solutions exist, with various algorithmic approaches (ML, heuristics, etc.)
- 70% average accuracy for SOTA log parsers; even 99% accuracy

The Bad

The performance of these solutions may have been inflated
(20% instead of 70%)

The Ugly

Log parsing in real-world scenarios has received too little attention:
disconnect between research and industry (5% on industry logs)

Main Contributions

What is the performance of log parsing?

Rethinking evaluation methodology

Our tool: `logchimera`

How can we connect industry log parsing and academia?

Log Parsing Example

Runtime log

```
Input size for job job_1445062781478_0011 = 1256521728. Number of splits = 10
```

Parsed log

Template	`Input size for job <*> = <*>. Number of splits = <*>`
Variables	`["job_1445062781478_0011", "1256521728", "10"]`

Evaluating Log Parsing Performance: Goal

- Performance on publicly available data
- Performance on publicly available data that resemble industry
- Performance on industry data

- Reproducing claimed results
- Scrutinizing results (which led to rethinking metrics)

Evaluating Log Parsing Performance: How

- 11 log datasets
- 14 methods
- Reproducing claimed results (70% accuracy)
- Scrutinizing previous evaluation & rethinking metrics (20% accuracy)

- Next up, two tables, measurements averaged over 10 runs
- Metrics used: *edit-distance* and *log template accuracy*
- For consistency with further experiments, we highlight results for AEL, Drain, and IPLoM

Evaluating Log Parsing Performance: Results

Dataset	AEL	Drain	IPLoM	LenMa	LFA	LKE	LogC
Apache	10.426	10.426	10.442	13.760	10.576	14.872	16.
Com							
Ind							

Dataset	AEL	Drain	IPLoM	LenMa	LFA	LKE								Spell
Apache	**0.694**	**0.694**	**0.694**	0.000	0.688	0.000								**0.694**
BGL	0.341	0.341	0.292	0.082	0.230	0.057								0.196
HDFS	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	**0.435**	0.000	0.000	0.000
HealthApp	0.164	0.238	0.158	0.136	0.149	0.133	0.138	0.220	0.126	0.166	**0.341**	0.041	0.322	0.152
HPC	0.644	0.620	0.638	0.632	0.609	0.360	0.632	0.632	0.509	0.632	**0.827**	0.226	0.661	0.530
Mac	0.172	0.224	0.041	0.132	0.101	0.172	0.162	0.228	0.118	0.042	**0.274**	0.163	0.148	0.032
OpenStack	0.018	0.018	0.000	0.018	0.008	0.010	0.010	0.010	0.010	0.000	**0.359**	0.018	0.119	0.000
Spark	0.194	0.194	0.192	0.004	0.190	0.001	0.006	0.038	0.000	0.208	0.204	0.004	**0.543**	0.192
Windows	0.154	0.159	0.001	0.154	0.142	0.148	0.153	0.156	0.150	0.006	**0.387**	0.151	0.140	0.004
Combined Dataset	0.267	0.258	0.214	0.140	0.180	0.140	0.128	0.258	0.092	0.180	**0.323**	0.067	0.280	0.186
Industry Dataset	0.054	**0.056**	0.041	0.001	0.022	0.001	0.002	0.054	0.000	0.048	0.050	0.002	0.034	0.041

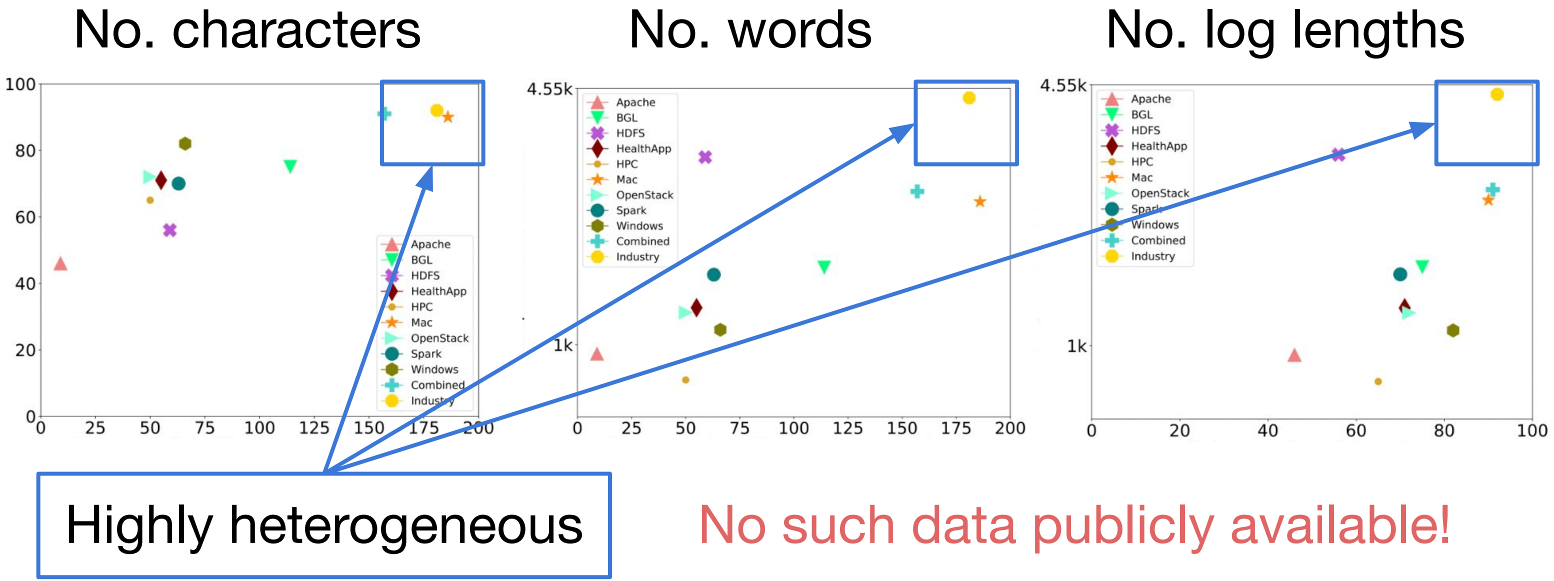
Public data: poor results
20% accuracy

Even worse, 5% accuracy on industry data

Why?

Industry Data Characteristics

- Industry data is significantly more heterogeneous

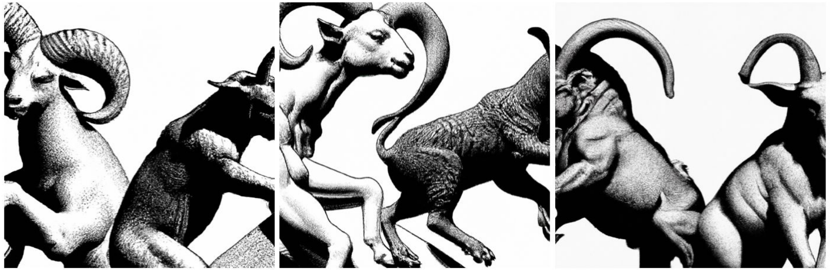


Connecting Industry Log Parsing to Academia

- We propose a tool: `logchimera`

logchimera

`logchimera` was born out of a research innitiative (Log Parsing Evaluation in the Era of Modern Software Systems), as a consequence of a general lack of access to heterogeneous log data typically found in industry. With `logchimera` you can generate and evaluate log parsing on heterogeneous industry-like data from publicly available logs. The name of the tool is inspired by the mythological creature *chimera*, which symbolizes a fusion or combination of different elements; and in this case, it reflects heterogeneity by enabling bringing together diverse formats from various logs to resemble industry-like contexts.



release v0.1.0

Access to industry-like logs

How?

Estimating log heterogeneity

- No established way to estimate *H*
- We chose a proxy metric for heterogeneity (*H*)

No	**Metric**	σ^2
1	No. unique words	0.278
2	No. unique characters	0.159
3	No. unique log lengths	0.331

$$H = 0.4 * \mathtt{nuw}\% + 0.2 * \mathtt{nuc}\% + 0.4 * \mathtt{nuldl}\%$$

- Next, we considered *Mixing* and *Fuzzing*

Logchimera Experiments: *Mixing*

Steps
1. labeling*
2. pre-processing
3. statistical analysis
4. weighing function
5. log replacement
6. consistency check

**optional*

More details in the paper.

```
mod_jk child workerEnv in error state 6
mod_jk child workerEnv in error state 7
mod_jk child workerEnv in error state 8
...
```

Initial dataset

```
mod_jk child workerEnv in error state 6
setting hostname to "authorMacBook-Pro.loc"
mod_jk child workerEnv in error state 8
...
```

Mixed dataset

(25% max)

Logchimera Experiments: *Mixing*

What is the performance when mixing?

Performance drops consistently as logs are mixed

H level	Dataset	AEL	Drain	IPLoM
0.219	Apache init	0.694/10.426	0.694/10.426	0.694/10.442
0.530	Apache (5)	0.653/15.925	0.653/15.923	0.653/15.369
0.640	Apache (10)	0.618/19.596	0.618/19.602	0.618/19.414
0.737	Apache (15)	0.586/23.468	0.586/23.495	0.586/22.809
0.816	Apache (20)	0.552/27.504	0.552/27.547	0.552/26.877
0.886	Apache (25)	0.514/32.265	0.514/32.135	0.514/31.34
0.259	HPC init	0.644/1.405	0.620/2.015	0.638/2.323
0.524	HPC (5)	0.605/6.510	0.581/7.081	0.599/7.257
0.661	HPC (10)	0.566/11.734	0.542/12.272	0.560/12.301
0.735	HPC (15)	0.525/16.720	0.501/17.300	0.519/17.124
0.817	HPC (20)	0.486/22.040	0.462/22.566	0.480/21.809
0.881	HPC (25)	0.447/27.875	0.423/28.234	0.441/27.349
0.608	BGL init	0.341/5.014	0.341/4.930	0.292/6.882
0.756	BGL (5)	0.321/11.004	0.321/10.940	0.273/12.665
0.833	BGL (10)	0.303/15.623	0.303/15.495	0.254/16.325
0.908	BGL (15)	0.285/20.665	0.285/20.639	0.251/20.970
0.949	BGL (20)	0.265/25.665	0.265/25.771	0.216/25.926
0.868	Mac init	0.172/19.534	0.224/19.882	0.041/20.928
0.901	Mac (5)	0.169/25.184	0.217/25.518	0.041/26.410
0.919	Mac (8)	0.169/28.507	0.217/28.838	0.041/29.730
0.830	Combined Dataset	0.267/13.612	0.258/17.302	0.214/14.094
1	Industry Dataset	0.054/21.959	0.056/27.201	0.041/24.122

Logchimera Experiments: *Mixing*

What is the performance when mixing?

Performance drops consistently as logs are mixed

Increased log complexity

H level	Dataset	AEL	Drain	IPLoM
0.219	Apache init	0.694/10.426	0.694/10.426	0.694/10.442
0.530	Apache (5)	0.653/15.925	0.653/15.923	0.653/15.369
0.640	Apache (10)	0.618/19.596	0.618/19.602	0.618/19.414
0.737	Apache (15)	0.586/23.468	0.586/23.495	0.586/22.809
0.816	Apache (20)	0.552/27.504	0.552/27.547	0.552/26.877
0.886	Apache (25)	0.514/32.265	0.514/32.135	0.514/31.34
0.259	HPC init	0.644/1.405	0.620/2.015	0.638/2.323
0.524	HPC (5)	0.605/6.510	0.581/7.081	0.599/7.257
0.661	HPC (10)	0.566/11.734	0.542/12.272	0.560/12.301
0.735	HPC (15)	0.525/16.720	0.501/17.300	0.519/17.124
0.817	HPC (20)	0.486/22.040	0.462/22.566	0.480/21.809
0.881	HPC (25)	0.447/27.875	0.423/28.234	0.441/27.349
0.608	BGL init	0.341/5.014	0.341/4.930	0.292/6.882
0.756	BGL (5)	0.321/11.004	0.321/10.940	0.273/12.665
0.833	BGL (10)	0.303/15.623	0.303/15.495	0.254/16.325
0.908	BGL (15)	0.285/20.665	0.285/20.639	0.251/20.970
0.949	BGL (20)	0.265/25.665	0.265/25.771	0.216/25.926
0.868	Mac init	0.172/19.534	0.224/19.882	0.041/20.928
0.901	Mac (5)	0.169/25.184	0.217/25.518	0.041/26.410
0.919	Mac (8)	0.169/28.507	0.217/28.838	0.041/29.730
0.830	Combined Dataset	0.267/13.612	0.258/17.302	0.214/14.094
1	Industry Dataset	0.054/21.959	0.056/27.201	0.041/24.122

Logchimera Experiments: *Fuzzing*

```
mod_jk child workerEnv in error state 6
mod_jk child workerEnv in error state 7
mod_jk child workerEnv in error state 8
…
```

Initial dataset

```
mod_jk child workerEnv in error state 6
mod_jk child workerEnv in error state 0xE0074
mod_jk child workerEnv in error state 8
…
```

Fuzzed dataset

Steps
1. labeling*
2. pre-processing
3. candidate discovery
4. variable replacement
5. consistency check

**optional*

More details in the paper.

Logchimera Experiments: *Fuzzing*

What is the performance when fuzzing?

H level	Dataset	AEL	Drain	IPLoM	LFA	LogMine
0.219	Apache init	0.694	0.694	0.694	0.688/10.576	0.694/10.426
0.886	Apache (25)	0.514/32.265	0.514/32.135	0.514/31.34	0.509/20.243	0.515/32.016
1	Apache (25) fuzzed	0.078/77.541	0.118/76.820	0.059/45.555	0.231/38.866	0.060/82.104
0.259	HPC init	0.644/1.405	0.620/2.015	0.638/2.323	0.609/3.182	0.632/3.218
0.881	HPC (25)	0.447/27.875	0.423/28.234	0.441/27.349	0.296/17.434	0.436/30.290
0.954	HPC (25) fuzzed	0.242/97.492	0.260/93.240	0.219/45.718	0.141/42.085	0.236/100.324
0.608	BGL init	0.341/5.014	0.341/4.930	0.292/6.882	0.230/12.524	0.220/18.598
0.949	BGL (20)	0.265/25.665	0.265/25.771	0.216/25.926	0.150/22.355	0.104/33.728
1	BGL (20) fuzzed	0.143/95.117	0.143/96.156	0.073/62.862	0.077/46.389	0.016/79.8035
0.868	Mac init	0.172/19.534	0.224/19.882	0.041/20.928	0.101/41.804	0.228/17.048
0.919	Mac (8)	0.169/28.508	0.217/28.838	0.041/29.73	0.098/51.167	0.224/25.933
1	Mac (8) fuzzed	0.018/135.994	0.011/116.706	0.001/112.063	0.013/79.454	0.022/207.376
0.830	Combined Dataset	0.267/13.612	0.258/17.302	0.214/14.094	0.180/24.144	0.258/15.858
1	Industry Dataset	0.054/21.959	0.056/27.201	0.041/24.122	0.022/41.960	0.054/23.506

Performance drops consistently with fuzzing

Similar performance to industry

Logchimera Experiments: Labels vs No Labels

Does fuzzing & mixing work without labels?

Dataset	H level gdth	H level parsed
Apache parsed (25) fuzzed	1	0.960
HPC parsed (25) fuzzed	0.954	0.938
BGL parsed (20) fuzzed	1	1
Mac parsed (8) fuzzed	1	0.906

Similar to industry, even without labels

Conclusion

- Log parsing is a critical process in automated log analysis
- Often times log parsing performance is misleading (70% instead of 20%)
- We rethink the evaluation methodology with an emphasis on industry: focus on log heterogeneity
- We propose `logchimera`, a tool to generate industry-like logs
- Open sourced (QR) at https://github.com/spetrescu/logchimera
- Rich potential for future research

Thanks!