Towards Understanding and Analyzing Rationale in Commit Messages using a Knowledge Graph Approach

Mouna Dhaouadi†, Bentley James Oakes‡†, Michalis Famelis†

† Université de Montréal
‡ Polytechnique Montréal

MDEIntelligence Workshop
at MODELS 2023
Co-authors

Mouna Dhaouadi

Bentley James Oakes

Michalis Famelis

PhD topic: Rationale extraction and management

Bentley.Oakes@polymtl.ca
Co-authors

Mouna Dhaouadi

Bentley James Oakes

Michalis Famelis

PhD topic:
Rationale extraction and management
For a **decision**, software/system developer has **rationale**

‘*Why* reasoning for their decision’
Motivation: Understanding Rationale

For a decision, software/system developer has rationale
‘Why’ reasoning for their decision

Rationale is useful information to deeply understand the system
Learn from mistakes, reuse solutions
Scope: Code commit messages
Rationale in Code Commits

**Scope:** Code commit messages

‘‘Fix: Added check to prevent null pointer exception.’’
Scope: Code commit messages

‘‘Fix: Added check to prevent null pointer exception.’’
versus
‘‘changes’’
Scope: Code commit messages

‘‘Fix: Added check to prevent null pointer exception.’’

versus

‘‘changes’’

Challenges:

- What is rationale exactly?
- Implicit or unrecorded
Scope: Code commit messages

‘‘Fix: Added check to prevent null pointer exception.’’  
versus  
‘‘changes’’

Challenges:

- What is rationale exactly?
- Implicit or unrecorded
- Ambiguous language
  - Especially past/future tenses
  - Non-native writers
- Need technical understanding
Scope: Code commit messages

‘‘Fix: Added check to prevent null pointer exception.’’
versus
‘‘changes’’

Challenges:
- What is rationale exactly?
- Implicit or unrecorded
- Ambiguous language
  - Especially past/future tenses
  - Non-native writers
- Need technical understanding

Research question:
How can we structure and extract rationale information from code commits?
Kantara: a framework for end-to-end rationale reconstruction
“Kantara: a framework for end-to-end rationale reconstruction”

**Purposes:**
- As research to gain insight into rationale itself
- Allow developers to better understand presence of rationale in their software
  - Who, what, where, when
- Decide if a code commit has *insufficient rationale*
  - Flag or reject
1 Introduction
2 Running Example
3 Creating a Dataset
4 Automatic Sentence Classification
5 Structuring Rationale Information
6 Analyzing Rationale Information
7 Conclusion
Linux kernel:
- Development is through Git commits
- Culture for motivating/describing changes
- Valuable (highest-quality?) source of rationale
**Running Example: Linux OOM Subsystem**

Linux kernel:
- Development is through Git commits
- Culture for motivating/describing changes
- Valuable (highest-quality?) source of rationale

Out-of-Memory Killer subsystem:
- When Linux kernel runs out of memory
- OOM-Killer is called to avoid crashing
- Two broad steps:
  1. Select “best” task to kill
  2. Force task to release memory and exit
- Meaningful heuristics
# Sentence

0 signal: Use `SEND SIG PRIV` not `SEND SIG FORCED` with `SIGKILL` and `SIGSTOP`.

Now that `siginfo` is never allocated for `SIGKILL` and `SIGSTOP` there is no difference between `SEND SIG PRIV` and `SEND SIG FORCED` for `SIGKILL` and `SIGSTOP`.

This makes `SEND SIG FORCED` unnecessary and redundant in the presence of `SIGKILL` and `SIGSTOP`.

Therefore change users of `SEND SIG FORCED` that are sending `SIGKILL` or `SIGSTOP` to use `SEND SIG PRIV` instead.

This removes the last users of `SEND SIG FORCED`.
<table>
<thead>
<tr>
<th>#</th>
<th>Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>signal: Use SEND_SIG_PRIV not SEND_SIG_FORCED with SIGKILL and SIGSTOP</td>
</tr>
</tbody>
</table>
0 signal: Use SEND_SIG_PRIV not SEND_SIG_FORCED with SIGKILL and SIGSTOP

1 Now that siginfo is never allocated for SIGKILL and SIGSTOP there is no difference between SEND_SIG_PRIV and SEND_SIG_FORCED for SIGKILL and SIGSTOP.

2 This makes SEND_SIG_FORCED unnecessary and redundant in the presence of SIGKILL and SIGSTOP.

3 Therefore change users of SEND_SIG_FORCED that are sending SIGKILL or SIGSTOP to use SEND_SIG_PRIV instead.

4 This removes the last users of SEND_SIG_FORCED.
1 Introduction
2 Running Example
3 Creating a Dataset
4 Automatic Sentence Classification
5 Structuring Rationale Information
6 Analyzing Rationale Information
7 Conclusion
Goal: Build a dataset of OOM-Killer commit sentences, classified wrt rationale
**Goal:** Build a dataset of OOM-Killer commit sentences, classified wrt rationale

**Motivation:**
- Gain insights through classification
- Empirical analysis
- Ground truth for automated classification
**Goal:** Build a dataset of OOM-Killer commit sentences, classified wrt rationale

**Motivation:**
- Gain insights through classification
- Empirical analysis
- Ground truth for automated classification

**Procedure:**
- Collect 410 commits (paper reports 180)
- Remove merge commits, filter code sentences
- Three authors label sentences
- Resolve conflicts in discussion
<table>
<thead>
<tr>
<th>Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inapplicable</td>
<td>Pre-processing error or bad sentences (i.e., does not contain English sentences)</td>
</tr>
<tr>
<td>Supporting Facts</td>
<td>A narration of facts used to support a decision</td>
</tr>
<tr>
<td>Rationale</td>
<td>Reason for a decision or value judgment</td>
</tr>
<tr>
<td>Decision</td>
<td>An action or a change that has been made, including a description of the patch behaviour</td>
</tr>
</tbody>
</table>
### Sentence Classification

<table>
<thead>
<tr>
<th>Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inapplicable</td>
<td>Pre-processing error or bad sentences (i.e., does not contain English sentences)</td>
</tr>
<tr>
<td>Supporting Facts</td>
<td>A narration of facts used to support a decision</td>
</tr>
<tr>
<td>Rationale</td>
<td>Reason for a decision or value judgment</td>
</tr>
<tr>
<td>Decision</td>
<td>An action or a change that has been made, including a description of the patch behaviour</td>
</tr>
</tbody>
</table>

- Obtained through piloting rounds and discussions
- **Multiple classifications per sentence**
- In disagreement, take classification union
- Fleiss kappa: Around 0.65 (fair to good agreement)
Classification Example

Decision signal: Use SEND_SIG_PRIV not SEND_SIG_FORCED with SIGKILL and SIGSTOP.

Supporting Fact & Rationale
Now that siginfo is never allocated for SIGKILL and SIGSTOP there is no difference between SEND_SIG_PRIV and SEND_SIG_FORCED for SIGKILL and SIGSTOP.

Rationale
This makes SEND_SIG_FORCED unnecessary and redundant in the presence of SIGKILL and SIGSTOP.

Decision
Therefore change users of SEND_SIG_FORCED that are sending SIGKILL or SIGSTOP to use SEND_SIG_PRIV instead.

Decision
This removes the last users of SEND_SIG_FORCED.
0 Decision

signal: Use SEND_SIG_PRIV not SEND_SIG_FORCED with SIGKILL and SIGSTOP
Classification Example

0 Decision signal: Use SEND_SIG_PRIV not SEND_SIG_FORCED with SIGKILL and SIGSTOP

1 Supporting Fact & Rationale Now that siginfo is never allocated for SIGKILL and SIGSTOP there is no difference between SEND_SIG_PRIV and SEND_SIG_FORCED for SIGKILL and SIGSTOP.

2 Rationale This makes SEND_SIG_FORCED unnecessary and redundant in the presence of SIGKILL and SIGSTOP.

3 Decision Therefore change users of SEND_SIG_FORCED that are sending SIGKILL or SIGSTOP to use SEND_SIG_PRIV instead.

4 Decision This removes the last users of SEND_SIG_FORCED.
OOM Subsystem Insights

Common structure:
- Decision summary phase,
- Supporting facts,
- Rationale,
- Decisions

Amount of rationale:
- 97.5% of commits contain rationale
- About 40-50% of sentences per commit contain rationale
Common structure:

<table>
<thead>
<tr>
<th>Decision summary phase</th>
<th>Supporting facts</th>
<th>Rationale</th>
<th>Decisions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Common structure:
- Decision summary phase
- Supporting facts
- Rationale
- Decisions

Amount of rationale:
- 97.5% of commits contain rationale
- About 40-50% of sentences per commit contain rationale
Common structure:
- Decision summary phase,
- Supporting facts, Rationale, Decisions

Amount of rationale:
- 97.5% of commits contain rationale
- About 40-50% of sentences per commit contain rationale
1 Introduction
2 Running Example
3 Creating a Dataset
4 Automatic Sentence Classification
5 Structuring Rationale Information
6 Analyzing Rationale Information
7 Conclusion
Automatic Classification of Sentences

- Binary classification: Logistic regression, decision tree, SVM
- Multi-label classification: Random Forest, XGBoost, KNN

XGBoost classification evaluation

<table>
<thead>
<tr>
<th>Label</th>
<th>Precision</th>
<th>Recall</th>
<th>F1-score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decision</td>
<td>0.76</td>
<td>0.69</td>
<td>0.72</td>
</tr>
<tr>
<td>Rationale</td>
<td>0.62</td>
<td>0.41</td>
<td>0.49</td>
</tr>
<tr>
<td>Supporting Facts</td>
<td>0.64</td>
<td>0.68</td>
<td>0.66</td>
</tr>
</tbody>
</table>

Insights:
- Overall poor performance
- Decisions easier to classify, rationale is harder
Binary classification: Logistic regression, decision tree, SVM
Multi-label classification: Random Forest, XGBoost, KNN
Binary classification: Logistic regression, decision tree, SVM
Multi-label classification: Random Forest, XGBoost, KNN

<table>
<thead>
<tr>
<th>XGBoost classification evaluation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Label</strong></td>
</tr>
<tr>
<td>Decision</td>
</tr>
<tr>
<td>Rationale</td>
</tr>
<tr>
<td>Supporting Facts</td>
</tr>
</tbody>
</table>

Insights:
- Overall poor performance
- Decisions easier to classify, rationale is harder
Automatic Classification of Sentences

Binary classification: Logistic regression, decision tree, SVM
Multi-label classification: Random Forest, XGBoost, KNN

<table>
<thead>
<tr>
<th>XGBoost classification evaluation</th>
<th>Precision</th>
<th>Recall</th>
<th>F1-score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decision</td>
<td>0.76</td>
<td>0.69</td>
<td>0.72</td>
</tr>
<tr>
<td>Rationale</td>
<td>0.62</td>
<td>0.41</td>
<td>0.49</td>
</tr>
<tr>
<td>Supporting Facts</td>
<td>0.64</td>
<td>0.68</td>
<td>0.66</td>
</tr>
</tbody>
</table>

Insights:
- Overall poor performance
- Decisions easier to classify, rationale is harder
1 Introduction
2 Running Example
3 Creating a Dataset
4 Automatic Sentence Classification
5 Structuring Rationale Information
6 Analyzing Rationale Information
7 Conclusion
Prior work modelled relationships between commits
Here, we model the sentences as a knowledge graph

Commit

commit_hash = 778c14a
date = 2014/01/31

Author

authorName = "David Rientjes"

Source

url = https://lore.kernel.org/all/alpine.DEB.2.02.
1401251942510.3140@chino.kir.corp.google.com/

Sentence

mm, oom: base root bonus on current usage

nextSentence

Sentence

A 3% of system memory bonus[...]

nextSentence

Sentence

In the example above[...]

Bentley.Oakes@polymtl.ca Towards Understanding and Analyzing Rationale 17 / 24
openCAESAR Framework


Ontological Modelling Language (OML) for easily creating OWL ontologies and the knowledge graph

Ontological Modelling Language (OML) for easily creating OWL ontologies and the knowledge graph
Ontology and Inferencing

- Inferred multi-classification simplified analysis
- Easier to query for CommitWithRationale, RationaleSentence

Towards Understanding and Analyzing Rationale

Bentley.Oakes@polymtl.ca
Ontology and Inferencing

- Inferred multi-classification simplified analysis
- Easier to query for CommitWithRationale, RationaleSentence
Introduction

Running Example

Creating a Dataset

Automatic Sentence Classification

Structuring Rationale Information

Analyzing Rationale Information

Conclusion
SPARQL queries on knowledge graph
Analysing Rationale Info

SPARQL queries on knowledge graph

Implemented:
- Listing authors and their commits
- List of sentences containing rationale
- List of commits containing rationale (uses inference)

```json
{
  "author": { "value": "Michal Hocko" } ,
  "commit_id": { "value": "c0" } ,
  "text": { "value": "mm: reduce noise in show_mem for lowmem allocations" } ,
  "isCommitWithRationale": { "value": "true" } ,
  "isSentenceRationale": { "value": "true" } ,
  "isSentenceDecision": { "value": "true" } ,
  "isSentenceSupporting": { "value": "false" }
}
```
Visualization
Visualization

**Intention:**
- Allow developers/researchers to understand rationale presence
- Identify commits/subsystems/developers without sufficient rationale

**Bentley.Oakes@polymtl.ca** Towards Understanding and Analyzing Rationale 22 / 24
**Visualization**

**Intention:**
- Allow developers/researchers to understand rationale presence
- Identify commits/subsystems/developers without sufficient rationale
Introduction
Running Example
Creating a Dataset
Automatic Sentence Classification
Structuring Rationale Information
Analyzing Rationale Information
Conclusion
Conclusion

Summary:

1. **Classification of sentences**
   - Supporting fact, rationale, decision

2. **Insights into rationale presence**
   - 40% of commit, category overlap

3. **Poor automatic classification**

4. **Structuring as knowledge graph, inferencing, and analysis**

5. **Visualization of rationale presence**

Future work:

- Finish dataset
- Investigate indicators for rationale evolution over time
- Developer characteristics
- Improve auto. classification
- Kantara as a pull request bot
Summary:

1. Classification of sentences
   - Supporting fact, rationale, decision
2. Insights into rationale presence
   - 40% of commit, category overlap
3. Poor automatic classification
4. Structuring as knowledge graph, inferencing, and analysis
5. Visualization of rationale presence

Future work:
- Finish dataset
- Investigate indicators for rationale
  - Evolution over time
  - Developer characteristics
- Improve auto. classification
- Kantara as a pull request bot
Conclusion

Summary:

1. Classification of sentences
   - Supporting fact, rationale, decision
2. Insights into rationale presence
   - 40% of commit, category overlap
3. Poor automatic classification
4. Structuring as knowledge graph, inferencing, and analysis
5. Visualization of rationale presence

Future work:

- Finish dataset
- Investigate indicators for rationale
  - Evolution over time
  - Developer characteristics
- Improve auto. classification
- Kantara as a pull request bot
**Conclusion**

**Summary:**

1. Classification of sentences
   - Supporting fact, rationale, decision
2. Insights into rationale presence
   - 40% of commit, category overlap
3. Poor automatic classification
4. Structuring as knowledge graph, inferencing, and analysis
5. Visualization of rationale presence

**Future work:**

- Finish dataset
- Investigate indicators for rationale
  - Evolution over time
  - Developer characteristics
- Improve auto. classification
- Kantara as a pull request bot

---

Towards Understanding and Analyzing Rationale in Commit Messages using a Knowledge Graph Approach
Mouna Dhaouadi, Bentley James Oakes, Michalis Famelis
Decision

will use check

patch page

extra bentley.oakes@polymtl.ca
Towards Understanding and Analyzing Rationale