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

Université de Montréal

POLYTECHNIQUE MONTRÉAL
UNIVERSITÉ D'INGÉNIERIE

Co-authors

Mouna Dhaouadi

Bentley James Oakes

Michalis Famelis

Co-authors

Mouna Dhaouadi

Bentley James Oakes

Michalis Famelis

PhD topic :

Rationale extraction and management

Motivation: Understanding Rationale

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

Rationale in Code Commits

Scope: Code commit messages

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’’

Rationale in Code Commits

Scope: Code commit messages

```
``Fix:  Added check to prevent null pointer exception.''
```

versus

```
``changes''
```

Challenges:

- What is rationale *exactly*?
- Implicit or unrecorded

Rationale in Code Commits

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

Rationale in Code Commits

Scope: Code commit messages

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

versus

```
‘‘changes’’
```

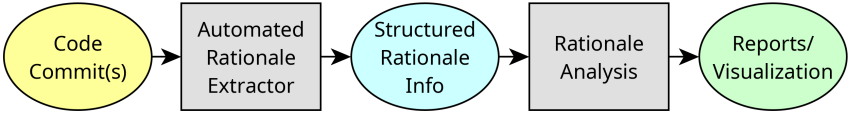
Challenges:

- 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?

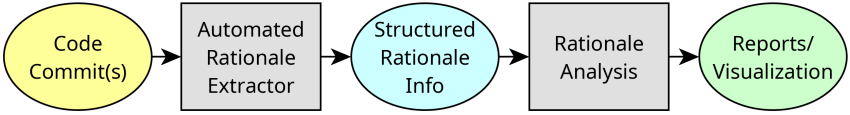
Kantara: Rationale Framework

"*Kantara: a framework for end-to-end rationale reconstruction*"



Kantara: Rationale Framework

"*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

Running Example: Linux OOM Subsystem

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

Commit Example

Commit Example

#	Sentence
0	signal: Use SEND_SIG_PRIV not SEND_SIG_FORCED with SIGKILL and SIGSTOP

Commit Example

#	Sentence
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.

Creating a Labelled Rationale Dataset

Goal: Build a dataset of OOM-Killer commit sentences,
classified wrt rationale

Creating a Labelled Rationale Dataset

Goal: Build a dataset of OOM-Killer commit sentences,
classified wrt rationale

Motivation:

- Gain insights through classification
- Empirical analysis
- Ground truth for automated classification

Creating a Labelled Rationale Dataset

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

Sentence Classification

Sentence Classification

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

Sentence Classification

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

- 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

Classification Example

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

OOM Subsystem Insights

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

OOM Subsystem Insights

Common structure:

Decision summary phase ,

Supporting facts , Rationale , Decisions

Amount of rationale:

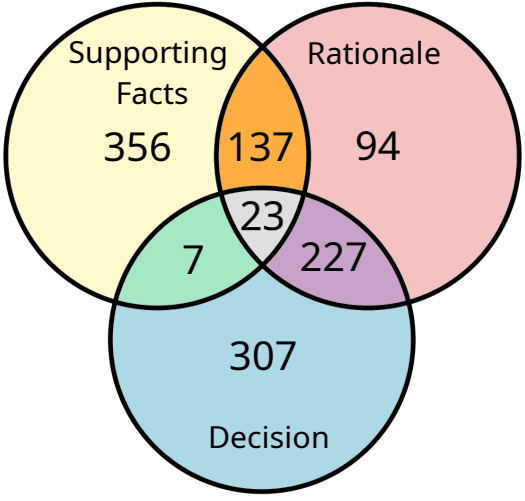
- **97.5%** of commits contain rationale
- About **40-50%** of sentences per commit contain rationale

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



Automatic Classification of Sentences

Automatic Classification of Sentences

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

XGBoost classification evaluation			
Label	**Precision**	**Recall**	**F1-score**
Decision	**0.76**	**0.69**	**0.72**
Rationale	0.62	0.41	0.49
Supporting Facts	0.64	0.68	0.66

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

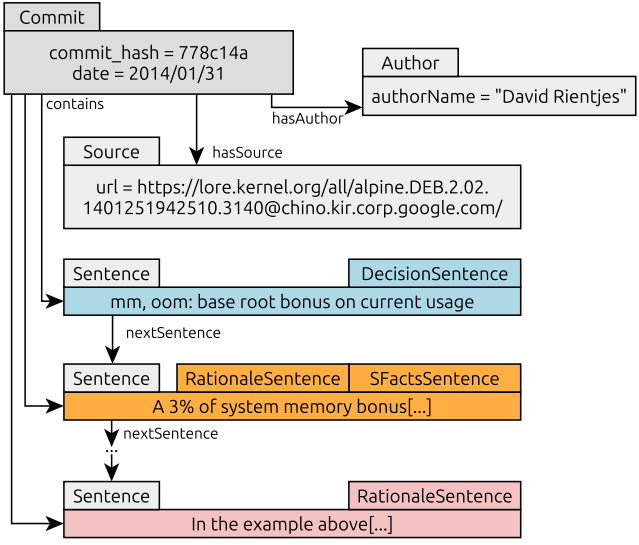
XGBoost classification evaluation			
Label	**Precision**	**Recall**	**F1-score**
Decision	**0.76**	**0.69**	**0.72**
Rationale	0.62	0.41	0.49
Supporting Facts	0.64	0.68	0.66

Insights:

- Overall poor performance
- Decisions easier to classify, rationale is harder

Rationale Info as a Graph

- Prior work modelled relationships between commits
- Here, we model the *sentences* as a *knowledge graph*



openCAESAR Framework

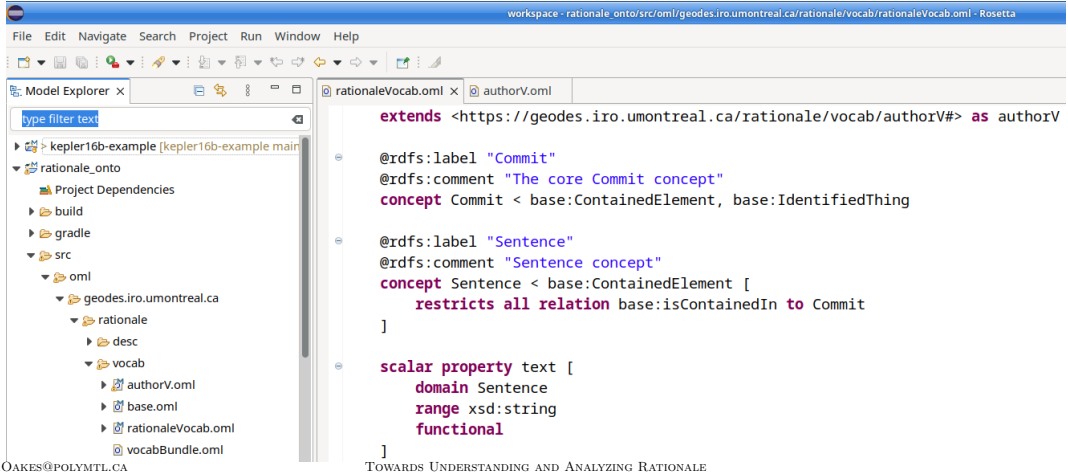
Elaasar *et al.*, “openCAESAR: Balancing agility and rigor in model- based systems engineering,” in Proceedings of SAM Conference, 2023

Elaasar *et al.*, "openCAESAR: Balancing agility and rigor in model- based systems engineering," in Proceedings of SAM Conference, 2023

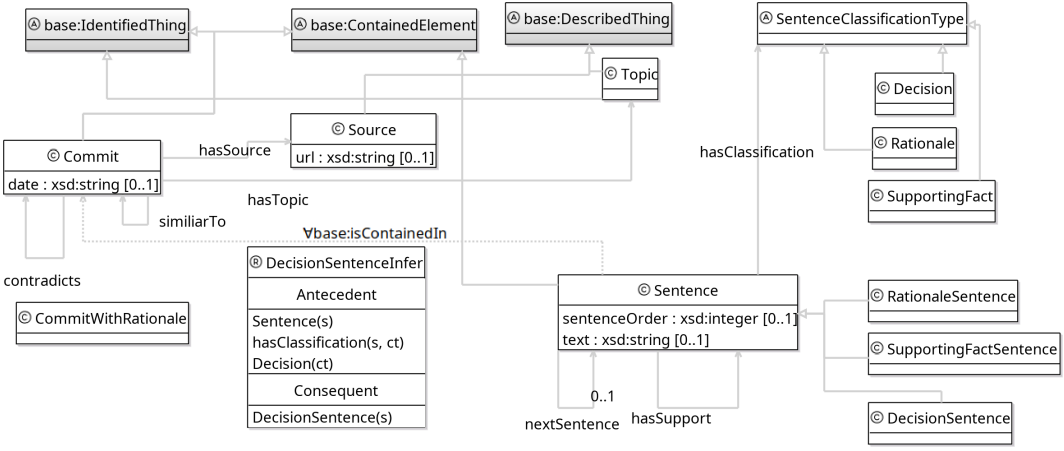
Ontological Modelling Language (OML) for easily creating OWL ontologies and the knowledge graph

Elaasar *et al.*, "openCAESAR: Balancing agility and rigor in model- based systems engineering," in Proceedings of SAM Conference, 2023

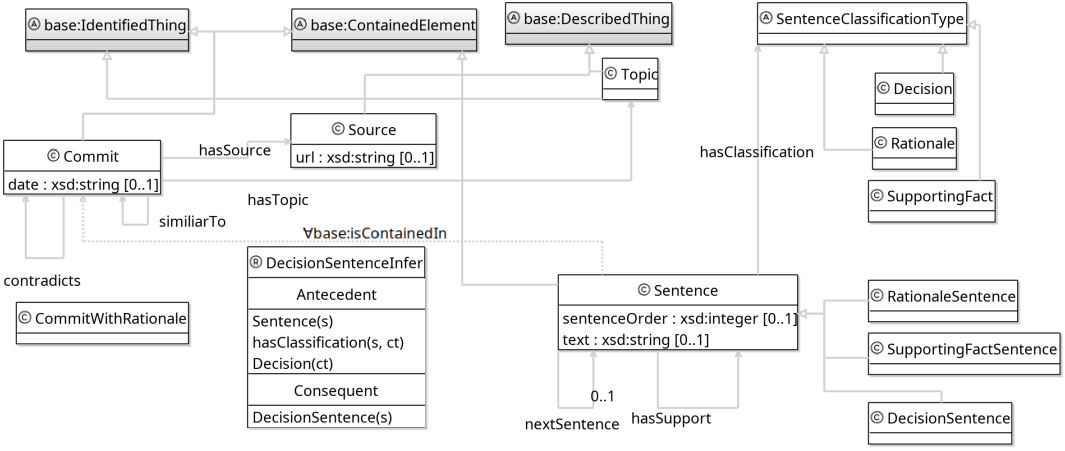
Ontological Modelling Language (OML) for easily creating OWL ontologies and the knowledge graph



Ontology and Inferencing



Ontology and Inferencing



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

Analysing Rationale Info

SPARQL queries on knowledge graph

SPARQL queries on knowledge graph

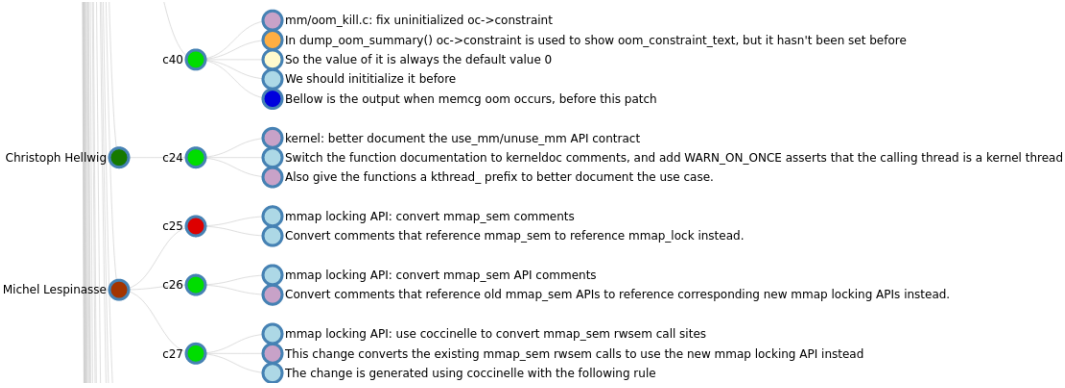
Implemented:

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

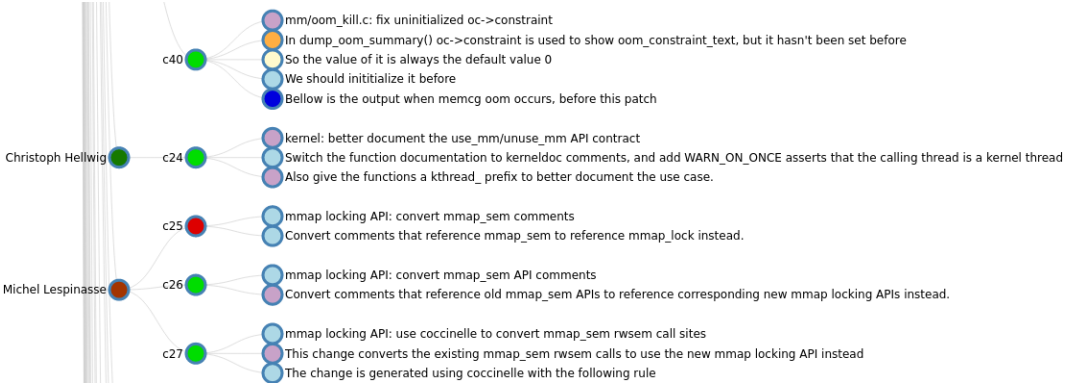

```
{
"author": { "value": "Michal Hocko" } ,
"commit_id": { "value": "c0" } ,
"text": { "value": "mm: reduce noise in show_mem for lowmem allocations" } ,
"isCommitWithRationale": { "value": "true" } ,
"isSentenceRationale": { "value": "true" } ,
"isSentenceDecision": { "value": "true" } ,
"isSentenceSupporting":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

Visualization

Visualization



Visualization



Intention:

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

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

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:

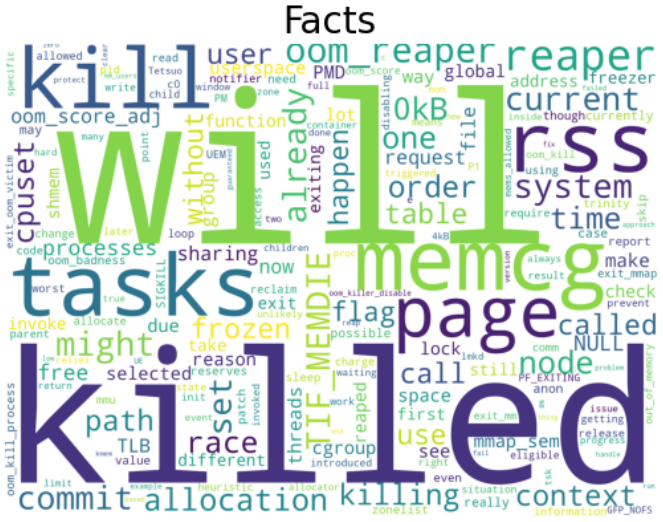
1. Classification of sentences
   - Supporting fact, rationale, decision
2. Insights into rationale presence
   - 40% of commit, category overlap
3. Poor automatic classification
4. Structuring as knowledge graph, inferencing, and analysis
5. Visualization of rationale presence

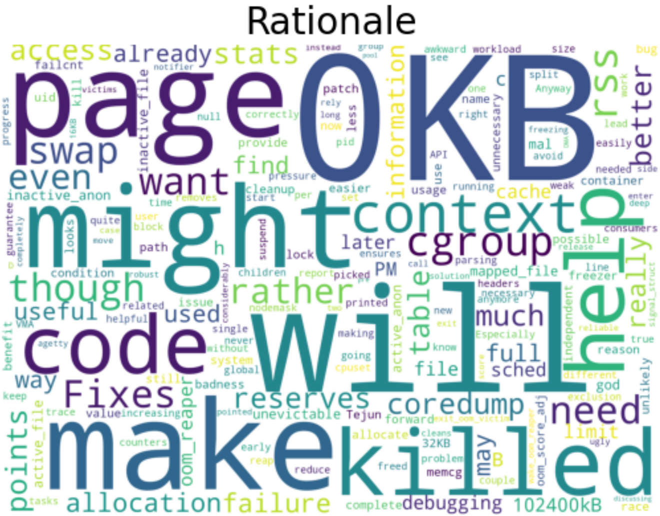
Future work:

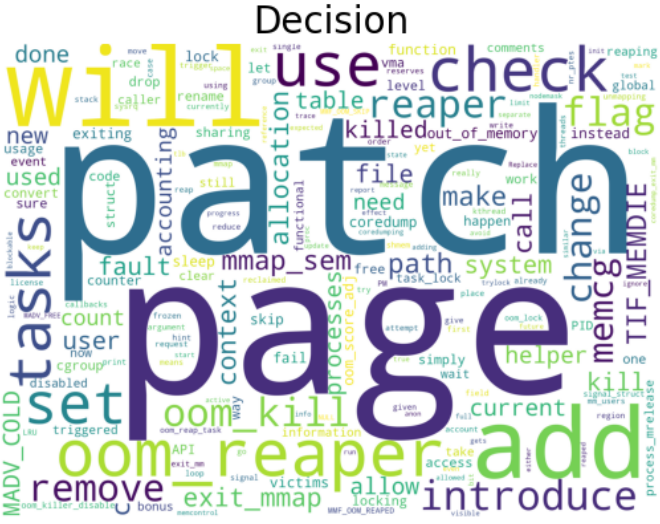
- Finish dataset
- Investigate indicators for rationale
  - Evolution over time
  - Developer characteristics
- Improve auto. classification
- Kantara as a pull request bot

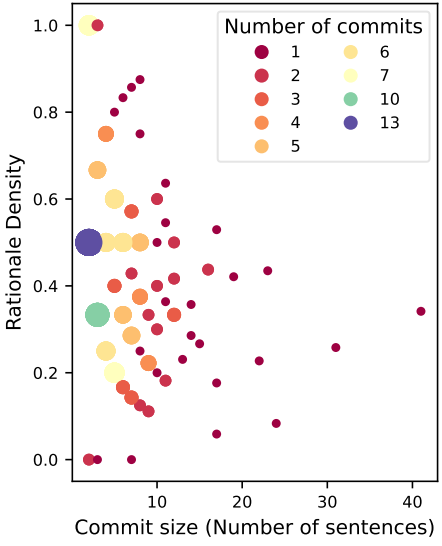
Towards Understanding and Analyzing Rationale in Commit Messages using a Knowledge Graph Approach

Mouna Dhaouadi, Bentley James Oakes, Michalis Famelis


Facts
kill
will
killed
tasks
memcg
page
rss
reaper
oom_reaper
user
oom_score_adj
current
system
order
table
time
called
might
frozen
flag
MEMDIE
TIF
set
path
race
commit
allocation
killing
context
node
call
lock
NULL
free
selected
reason
sharing
processes
cpuset
shmem
group
without
already
happen
exiting
function
request
file
one
0kB
PMD
global
way
address
freezer
though
currently
inside
require
skip
trinity
case
report
make
exit_mmap
check
prevent
always
result
now
exit
reclaim
due
invoke
allocate
parent
possible
take
charge
waiting
sleep
reaped
threads
cgroup
use
see
mmap_sem
eligible
first
space
anon
PF_EXITING
issue
getting
release
progress
different
TLB
value
zonelist
really
situation
information
GFP_NOFS
even
right
introduced
allocator
heuristic
example
limit
oom_kill_process
oom_badness
oom_killer_disable
SIGKILL
worst
code
change
later
loop
children
proc
4kB
oom_kill
using
fix
mems_allowed
triggered
PI
exit_oom_victim
specific
allowed
read
Tetsuo
kill
user_space
write
child
CO
protect
notifier
window
need
PM
zone
full
lot
container
done
disabling
may
many
point
hard
UEM
used
process
two
parent
exiting
oom_score
non
failed
version
approach
out_of_memory
problem
handle
fall
tsk
run
GFP_
zero
later
state
init
patch
invoked
event
end
work
exit_mm
found
comm
lmkd
reserves
return
UE
mmu
knen
example
heuristic
allocator
limit
zonelist


Rationale
access
already
stats
page
0KB
rss
better
swap
even
want
find
information
might
context
cgroup
help
really
though
rather
will
useful
used
much
full
code
file
sched
table
Fixes
reserves
coredump
need
make
killed
points
allocation
failure
debugging
102400kB
limit
cache
later
PM
patch
split
Anyway
container
mapped_file
usage
inactive_anon
active_anon
active_file
oom_reaper
oom_score_adj
unevictable
memcg
forward
allocate
trace
value
increasing
counters
early
reduce
freed
problem
couple
complete
tasks
race
bug
size
workload
awkward
instead
group
failcnt
notifier
victims
kill
uid
null
correctly
provide
pressure
cleanup
start
easier
running
weak
consumers
possible
parsing
freezer
headers
necessary
anymore
Especially
reason
different
god
exclusion
unlikely
true
ugly
32KB
Tejun
badness
global
system
cpuset
going
know
new
exit
making
printed
picked
nodemask
children
report
lock
ensures
single
never
without
still
keep
benefit
VMA
agetty
helpful
related
issue
condition
robust
path
quite
user
block
time
removes
guarantee
completely
looks
progress
suspend
less
rely
long
pid
API
name
right
one
unnecessary
freezing
mal
avoid
easily
lead
needed
side
work
enter
deep
line
reliable
independent
solution
call
see


Decision
patch
page
will
use
check
add
oom_reaper
remove
introduce
set
tasks
oom_kill
reaper
flag
change
memcg
TIF_MEMDIE
MADV_COLD
exit_mmap
done
lock
race
drop
caller
rename
table
killed
out_of_memory
instead
reaping
global
new
exiting
sharing
accounting
allocation
file
need
make
work
call
used
convert
sure
fault
sleep
clear
counter
mmap_sem
free
path
system
count
user
skip
fail
context
processes
oom_score_adj
helper
simply
wait
kill
current
information
triggered
disabled
cgroup
access
allow
locking
victims
bonus
API
exit_mm
oom_killer_disable
oom_reap_task
process_mrelease
PID
one
functional
coredump
happen
task_lock
MMF_OOM_REAPED


Number of commits
1
2
3
4
5
6
7
10
13
Rationale Density
0.0
0.2
0.4
0.6
0.8
1.0
10
20
30
40
Commit size (Number of sentences)