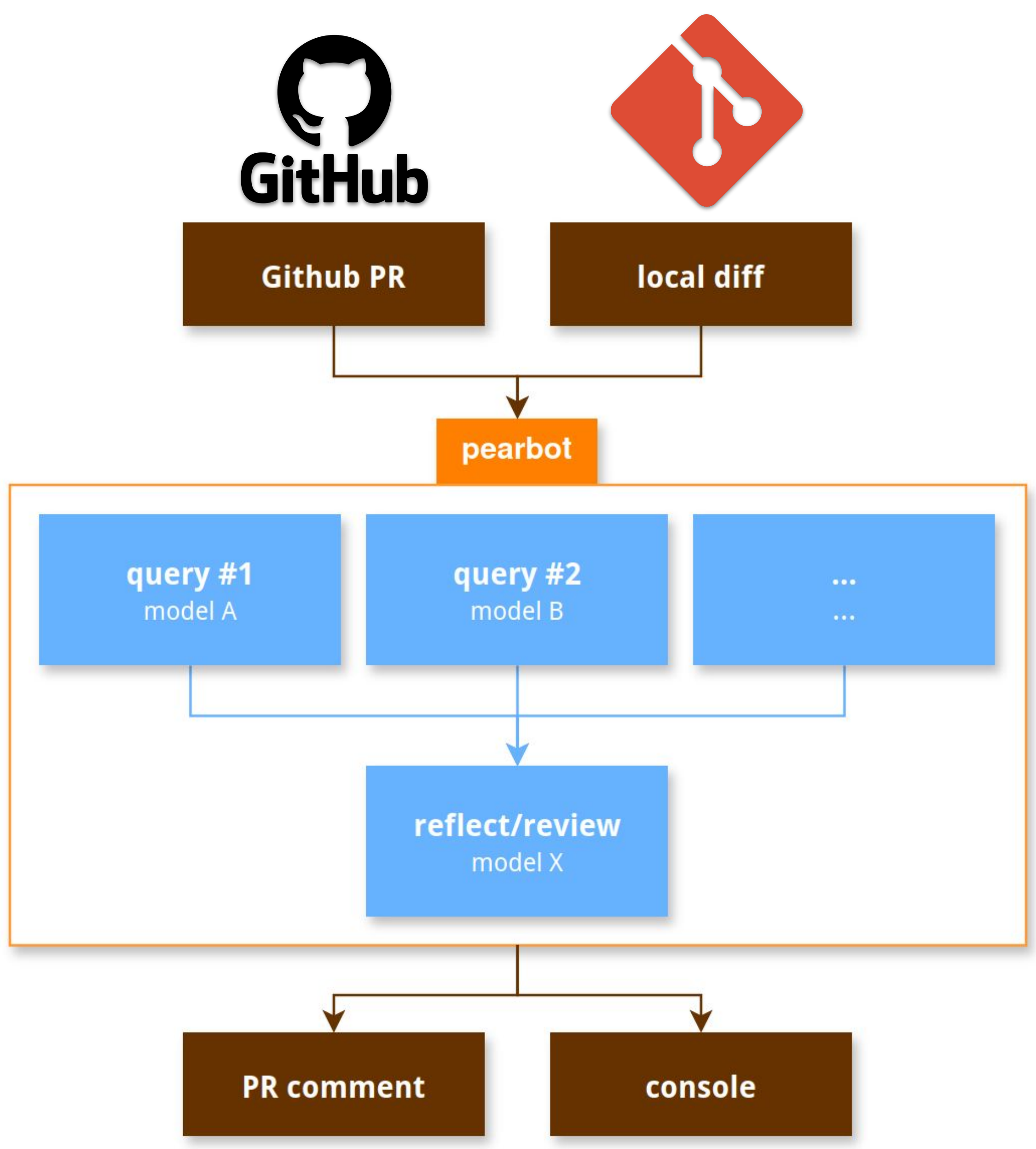
LLMs for Enhanced Code Review

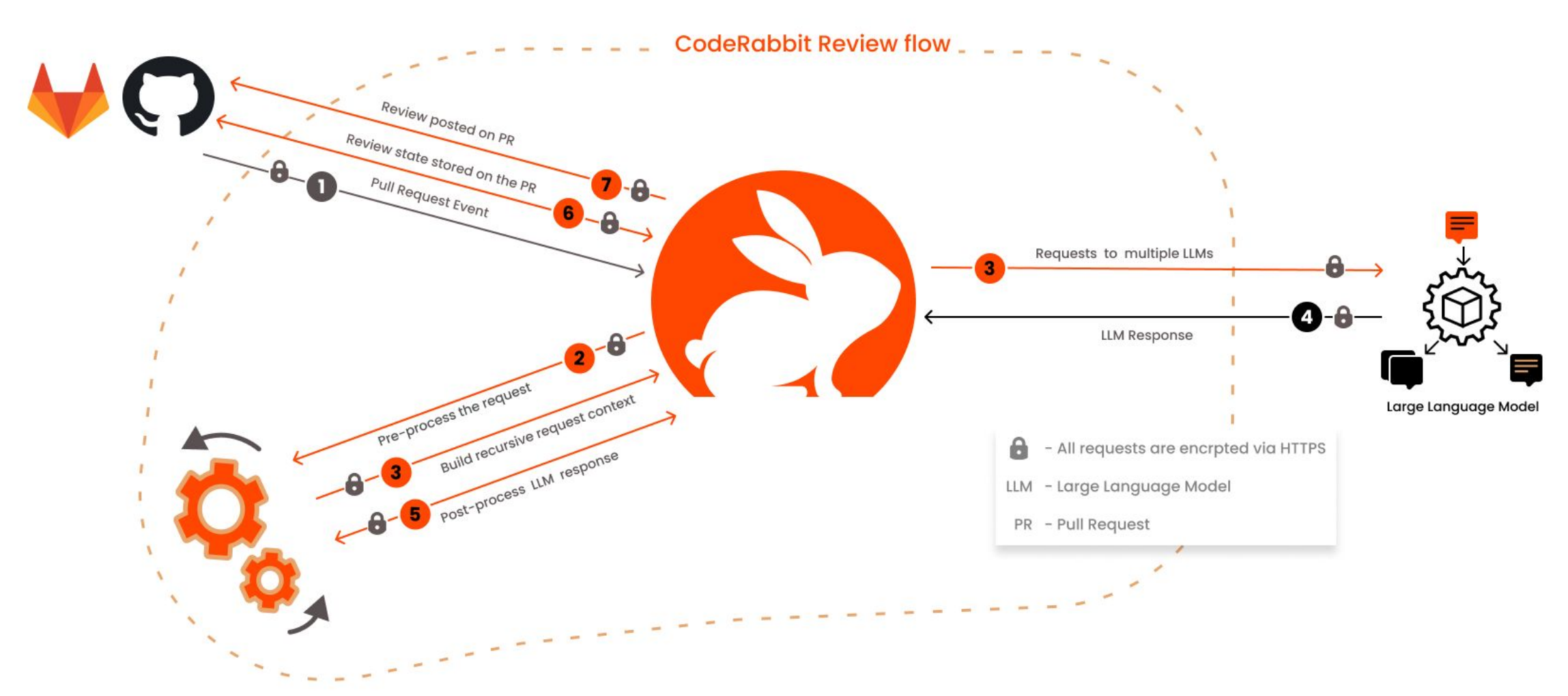
Alexey Rybalchenko, Mohammad Al-Turany
GSI Helmholtz Centre for Heavy Ion Research GmbH, Darmstadt, Germany

Collaborative software development demands rigorous code review processes to ensure maintainability, reliability, and efficiency. We integrate Large Language Models (LLMs) into the code review process, utilizing both commercial and open models. We present a comprehensive code review workflow that incorporates open-weights LLMs, integrating various enhancements such as multi-agent capabilities and reflection. By harnessing the capabilities of LLMs, the review process can uncover faults and identify improvements that traditional automated analysis tools may overlook. This integration shows promise for improving code quality and reducing errors. We deploy **coderabbit.ai** for commercial models & develop our own tool - **pearbot** for usage with local models.

Pearbot [1]:



CodeRabbit [2]:


CodeRabbit Review flow
Review posted on PR
Review state stored on the PR
Pull Request Event
1
2
3
4
5
6
7
Requests to multiple LLMs
LLM Response
Large Language Model
Pre-process the request
Build recursive request context
Post-process LLM response
- All requests are encrpted via HTTPS
LLM - Large Language Model
PR - Pull Request

- AI-based Pull Request summarizer and reviewer with chat capabilities via a Github/Gitlab App.
- Combination of various OpenAI models.
- Previously open source, now a closed project.
- Free to use for open source projects.

Pearbot
Instructions

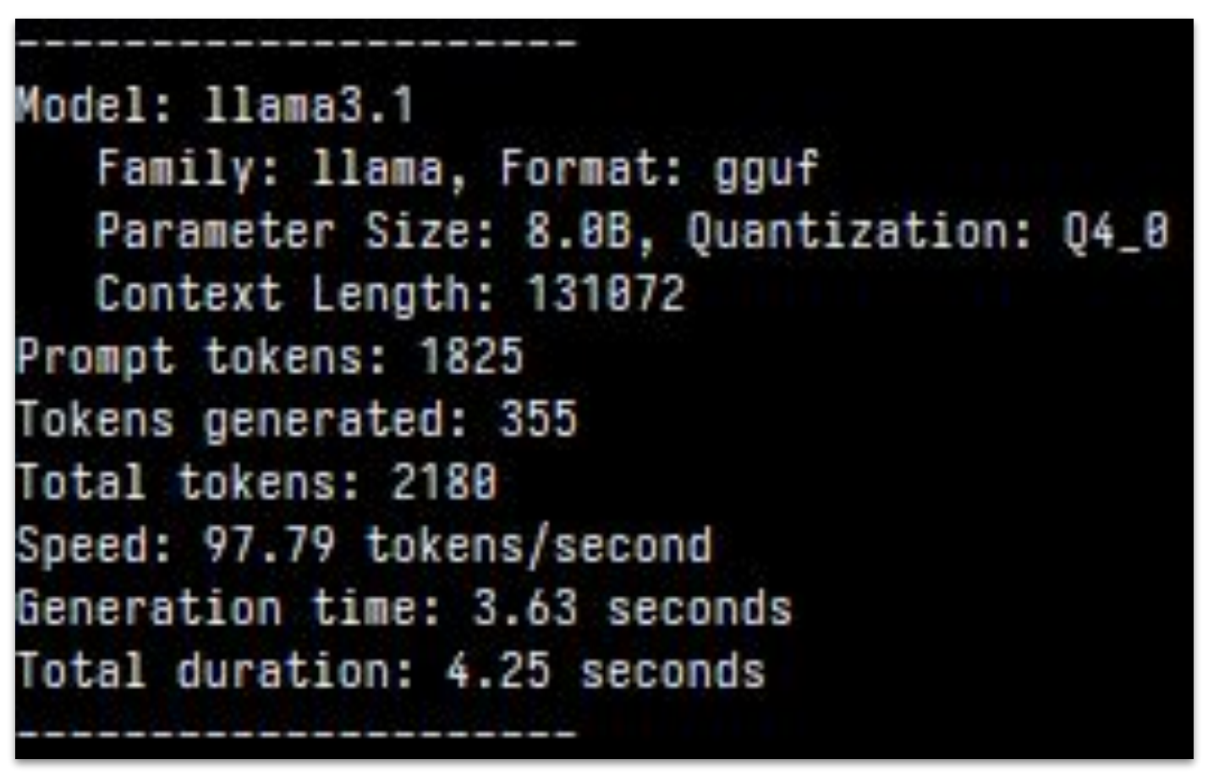
LLM query
- **instructions**
- **PR description**
- **additional context**
- **code changes**

1. **Multi-Agent[5] Initial Reviews:**
   - Multiple AI models generate initial code reviews.
   - Each model provides a unique perspective on the code changes.
   - This approach gathers a diverse set of potential issues and improvements.
2. **Reflection[6][7] by a Final Agent:**
   - A separate, possibly more advanced, model analyzes the initial reviews.
   - This agent synthesizes and refines the feedback from multiple sources, rejects potentially less impactful comments.
   - It prioritizes the most important issues and suggestions.
   - The reflection step helps in producing a more comprehensive and coherent final review.
3. **Good Review Examples:**
   - Specific & useful code review examples, which encourages specific type of reply, guiding the agents.
   - Examples include Chain-of-Thought[7] type of reviews, that include some reasoning why the suggestions would be good.
   - Models are prompted to think through the implications of changes.

Pearbot
Backend

ollama[3] (via python lib and HTTP request): open-source large language model server, written in Go, backed by **llama.cpp**[4] (C++):

- Efficient serving of large language models
- CPU/GPU/CPU+GPU hybrid inference to partially accelerate models larger than the total VRAM capacity
- Supports many model architectures: llama, gemma2, qwen2, ...
- Support for multitude of model quantization techniques and precisions for faster inference and reduced memory use
- Usage Metrics



```
---------------------
Model: llama3.1
   Family: llama, Format: gguf
   Parameter Size: 8.0B, Quantization: Q4_0
   Context Length: 131072
Prompt tokens: 1825
Tokens generated: 355
Total tokens: 2180
Speed: 97.79 tokens/second
Generation time: 3.63 seconds
Total duration: 4.25 seconds
---------------------
```

Pearbot
Features

- GitHub App for reviewing Pull Requests.
- Local execution mode for diffs or annotated commits.
- Agent ensemble approach for comprehensive analysis:
  - initial reviews with any number of different models
  - followed by a final model to refine the initial reviews
- Customizable model(s) via the ollama setup.
- Execution on low-end hardware and/or without GPU.
- Customizable prompt(s).

As a GitHub App:

```
python pearbot.py --server
```

To analyze a local diff file:

```
python pearbot.py --diff path/to/your/diff/file
```

Or pipe a diff directly:

```
git diff | python pearbot.py
```

Generate detailed output with commit messages, e.g.:

```
git format-patch HEAD~3..HEAD --stdout | python pearbot.py
```

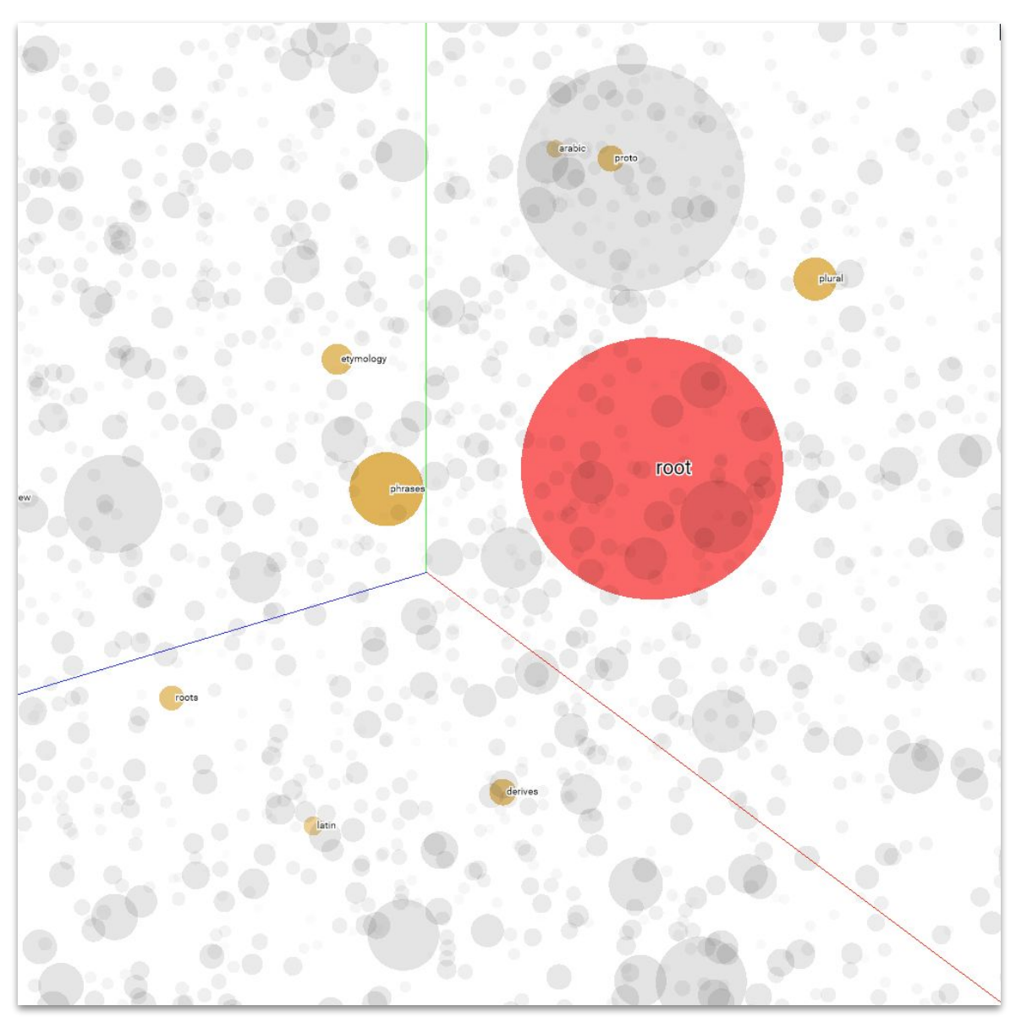
exploration in future work
Pearbot: Refining the Context

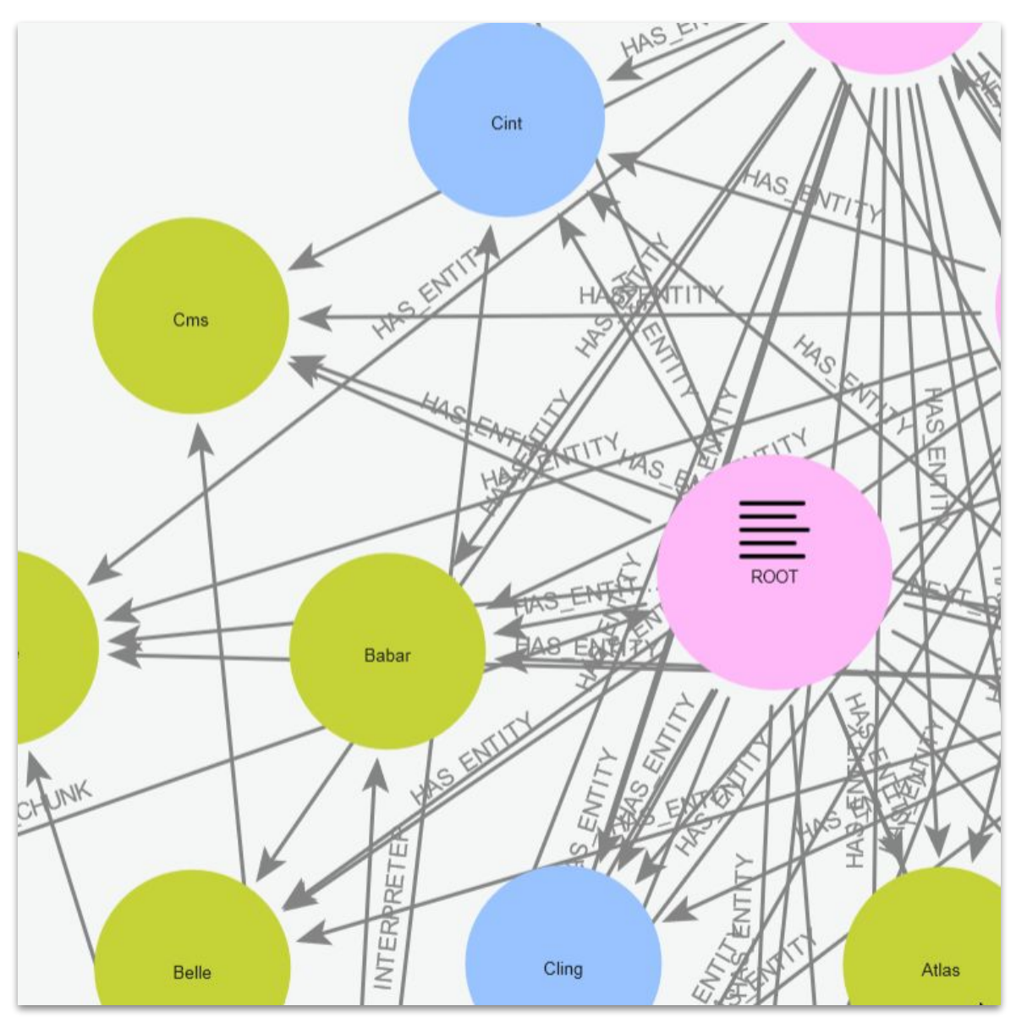
Adding relevant context to the LLM query can enhance the quality and relevance of the review. For example:

- Related issues and their summaries.
- Coding standards/guidelines excerpts.
- Historical code changes of the corresponding code.
- Experiment & Framework-specific coding conventions.

To select relevant information for the context and to avoid overloading context size of the model, relevant parts should be selected. One current technique to achieve this is **Graph Retrieval-Augmented Generation (GraphRAG)**:

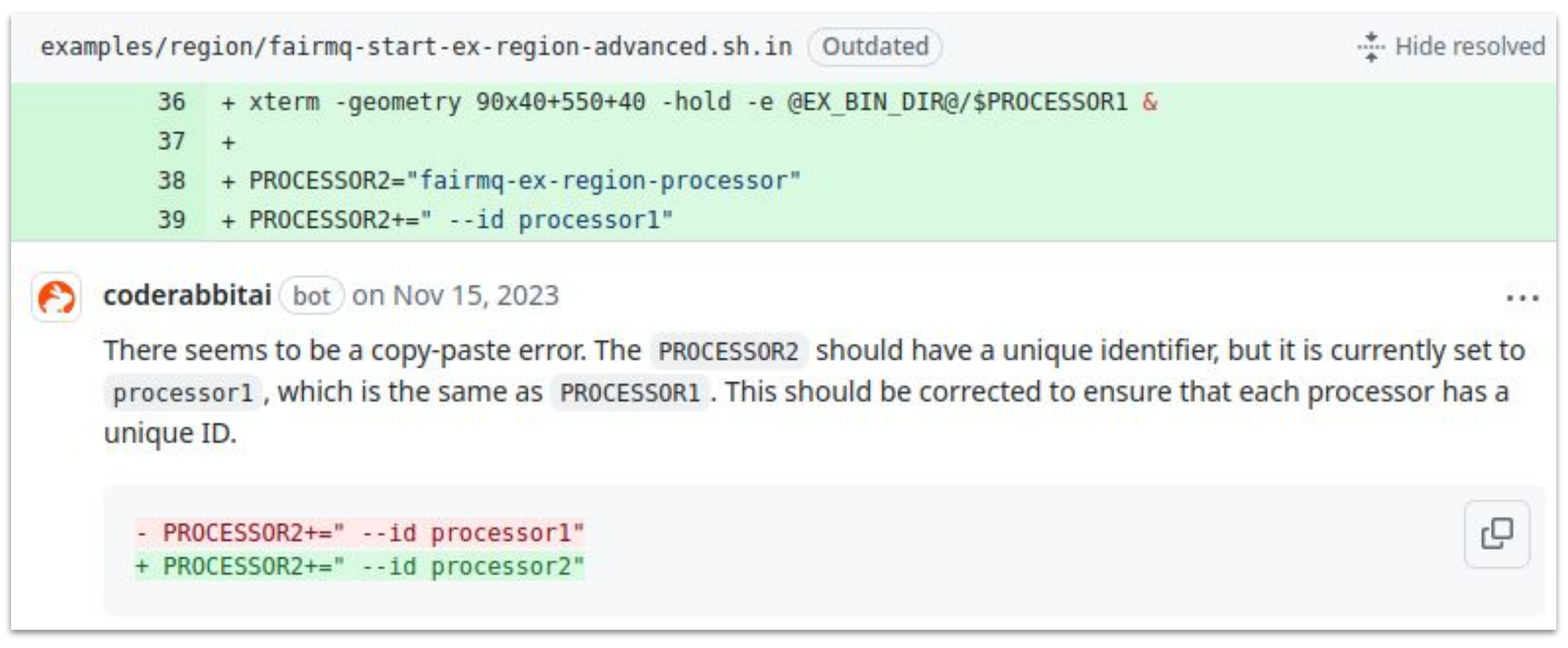
1. Knowledge Graph Construction:
   - entities & relationships representing the project's code, issues & guidelines
2. Contextual Retrieval:
   - traverse the knowledge graph to find relevant nodes
   - Use similarity measures and relationship strengths to rank the importance of related context.





LLM
Review Strengths

- Unbiased, 24/7, scalable, multi-lingual, customizable, with a broad knowledge base, can be kept up-to date with new data.
- Good at identifying logical errors early, that other automated tools or even human review may overlook, avoiding issues later in the project lifetime.
- Can potentially reduce the time and resources needed for code reviews.



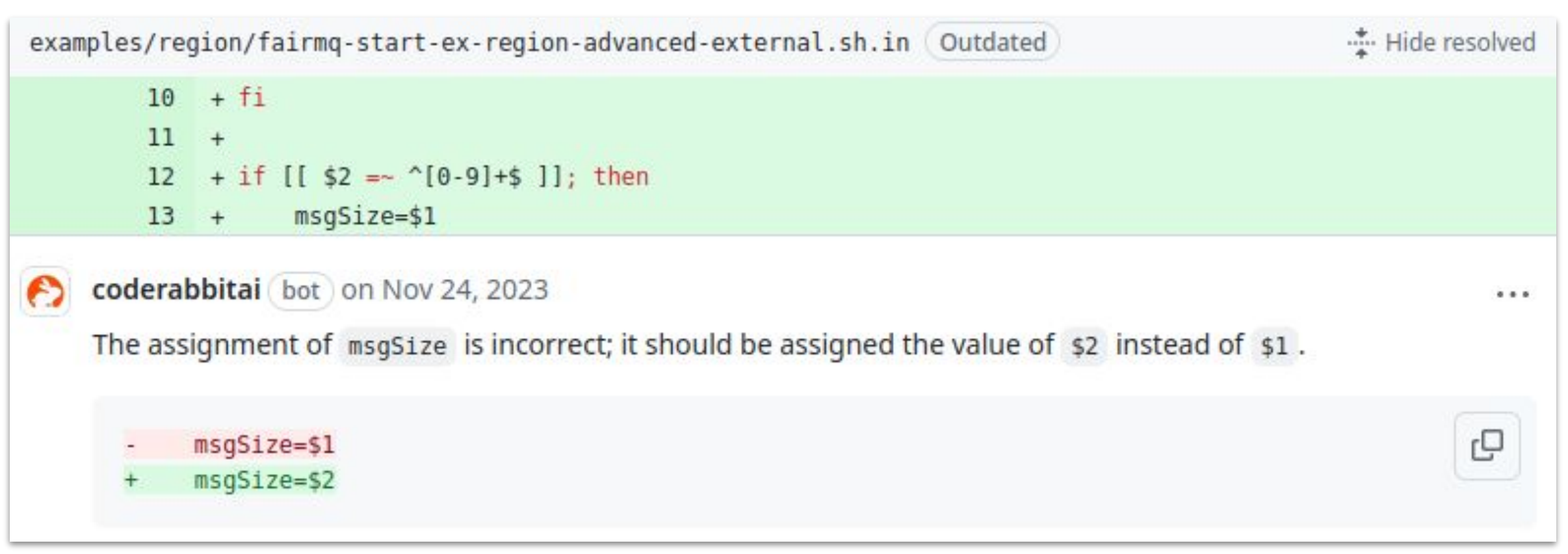
examples/region/fairmq-start-ex-region-advanced.sh.in Outdated · Hide resolved

```
+ xterm -geometry 90x40+550+40 -hold -e @EX_BIN_DIR@/$PROCESSOR1 &
+
+ PROCESSOR2="fairmq-ex-region-processor"
+ PROCESSOR2+=" --id processor1"
```

coderabbitai bot on Nov 15, 2023

There seems to be a copy-paste error. The `PROCESSOR2` should have a unique identifier, but it is currently set to `processor1`, which is the same as `PROCESSOR1`. This should be corrected to ensure that each processor has a unique ID.

```
- PROCESSOR2+=" --id processor1"
+ PROCESSOR2+=" --id processor2"
```



examples/region/fairmq-start-ex-region-advanced-external.sh.in Outdated · Hide resolved

```
+ fi
+
+ if [[ $2 =~ ^[0-9]+$ ]]; then
+     msgSize=$1
```

coderabbitai bot on Nov 24, 2023

The assignment of `msgSize` is incorrect; it should be assigned the value of `$2` instead of `$1`.

```
-     msgSize=$1
+     msgSize=$2
```

LLM
Review Weaknesses

- May produce unnecessary output, when no actionable changes are necessary or such are not detected by the model, which would waste developer's time.
- Weaker models, especially with quantized weights, may more easily dive into hallucinations, potentially leading to inaccurate or irrelevant code suggestions.
- Limited understanding of complex projects, which human reviewers inherently possess.
- Potential for false positives, flagging issues that aren't actually problematic.
- Inability to fully comprehend the broader architectural implications of code changes, especially in large, complex systems.

LLM
Boost Strengths, Reduce Weaknesses

Allowing independent AI agents to generate separate reviews, followed by a final AI reviewer's synthesis, can improve the response quality. At the same time, the system should filter out superfluous reviews, presenting only relevant insights to the user, or none at all when there are none.

References:
[1] https://github.com/GSI-HPC/pearbot
[2] https://coderabbit.ai/
[3] https://ollama.com/
[4] https://github.com/ggerganov/llama.cpp
[5] Li, Junyou, et al. "More agents is all you need." arXiv preprint arXiv:2402.05120 (2024). https://doi.org/10.48550/arXiv.2402.05120
[6] Madaan, Aman, et al. "Self-refine: Iterative refinement with self-feedback." Advances in Neural Information Processing Systems 36 (2024). https://doi.org/10.48550/arXiv.2303.17651
[7] Shinn, Noah, et al. "Reflexion: Language agents with verbal reinforcement learning." Advances in Neural Information Processing Systems 36 (2024). https://doi.org/10.48550/arXiv.2303.11366
[8] Wei, Jason, et al. "Chain-of-thought prompting elicits reasoning in large language models." Advances in neural information processing systems 35 (2022): 24824-24837. https://doi.org/10.48550/arXiv.2201.11903