JOHANNES KEPLER
UNIVERSITY LINZ

Master's Thesis
Analyzing Memory Anti-Patterns over Time

Student: Roman Sperl
Advisor: Dipl.-Ing. Dr. Markus Weninger, BSc
Start date: August 2022

Dipl.-Ing. Dr. Markus Weninger, BSc
Institute for System Software
T +43-732-2468-4361
markus.weninger@jku.at

Modern programming languages such as Java are so-called *managed languages* that leverage garbage collection to free the programmer from the error-prone task of manually managing memory. While garbage collection prevents certain memory problems, anomalies such as memory leaks can still happen. Such anomalies are often connected to the inproper use of collections such as lists or maps.

In “Patterns of Memory Inefficiency”, Chis et. al (https://doi.org/10.1007/978-3-642-22655-7_18) explain a number of anti-patterns that can lead to memory problems. They extract these patterns for data structures detected in heap dumps.
Yet, heap dump analysis naturally restricts the analysis to a single point in time. This leads to the possible problem that metrics and patterns present at that point in time, even though providing good initial insights, may not be representative for the whole run of the application.
For example, taking a heap snapshot at the wrong point in time may present a lot of empty collections, yet they emerge as false positives, since the collections will be filled with objects a few moments later.
In addition to that, only taking into account a single point in time does not allow to make assumptions about container behavior such as container growth. Ample work exists that suggest that most memory leaks are related to misused and growing data structures. In previous work, we have shown that keeping track of the evolution of data structures can provide metrics that emphazise collections that cause extraordinary growth.
The patterns by Chis et. Al mostly relate to "memory bloat", i.e., the inefficient use of memory (e.g., more memory is used as "glue" instead of for "real data"). Yet, there are also other problems that are not always related to containers such as memory churn, i.e., very short-living objects that are allocated frequently. Again, this is a time-based problem that cannot be detected in a single heap state, since one has to keep track of object births and deaths.

Taking all this points into account, this work should build upon the existing patterns by Chis et al. but extend them in multiple ways.
Thus, the main contributions of this work should be:

- extending the existing metrics and patterns by a time component. This can increase the accuracy and reduce the false-positive rate. For example, to know that a collection has been empty not only at a single point in time, but over a longer period of time, increases the likelyhood of misuse.
- new metrics, for single point in time as well as time-based metrics.
    - This may include metrics to detect memory leaks (for example due to container growth), patterns for memory churn, or similars.
- a summarization of all patterns, based on their goal (leak detection, bloat detection, churn detection, etc.) their timeliness (single point in time, two points in time, continous), and how to calculate them.
- an evaluation that shows how the developed prototype based on the AntTracks memory

monitoring tool is able to detect these patterns in data reconstructed from memory traces.

The scope of this work is defined in a broad scope on purpose since this work will be continued as a Master’s thesis, in which certain aspects of this work will be investigated and improved in more detail.

Modalities:

The progress of the project should be discussed at least every four weeks with the advisor. A time schedule and a milestone plan must be set up within the first 3 weeks and discussed with the advisor and the supervisor. It should be continuously refined and monitored to make sure that the thesis will be completed in time. The final version of the thesis must be submitted not later than 31.01.2023.