Memory Management Prototyper

Designing and tuning a garbage collection algorithm and memory management in a virtual machine is a very time consuming task: next to verifying the correctness of the algorithms an important task is tuning them to be competitive with existing systems. This problem is amplified due to long turn-around time of the tuning process: you have to change the algorithm, recompile the VM, run the benchmark, analyze its behavior, try to reconstruct the situation leading any unwanted behavior, then change the algorithm and finally repeat the procedure.

The idea of this task is to create a tool for prototyping memory management algorithms in a VM. This should work as follows: The developer prototype the memory management algorithm in a program simulator using a scripting language or a domain specific language. This simulator is fed an object graph trace of the program to examine, resulting in changes to the heap layout, controlled by the GC prototype. These changes are directly sent to a memory management visualizer, that in turn is used to control the execution of the simulator. At any point the developer should be able to change the GC prototype of the simulator and even control the execution of the program.

E.g. if the developer detects by visual inspection that the algorithm does not perform as requested, he simply pauses the simulation, adapts the prototype algorithm, and then continues execution of the program at the beginning of the latest garbage collection again.

This would decrease GC algorithm turn-around times tremendously.

The task is limited to the simulator and the integration into the visualization. A recently presented tool called “Elephant Tracks” [1] provides object graph traces with (mostly) accurate object death records. Another tool, recently updated at the institute, GCSpy [2,3], visualizes memory management behavior.

Other information

- Programming language: Java

Further information

- Thomas Schatzl, HF305, thomas.schatzl@jku.at

[1] http://www.cs.tufts.edu/research/redline/elephantTracks/
[2] http://www.cs.kent.ac.uk/projects/gc/gcspy/
[3] http://ssw.uni-linz.ac.at/General/Staff/TS/