Automatic Region-Based Memory Management for Real-Time Embedded Systems

Guillaume Salagnac
Verimag - Grenoble - France
http://www-verimag.imag.fr/PEOPLE/Guillaume.Salagnac/

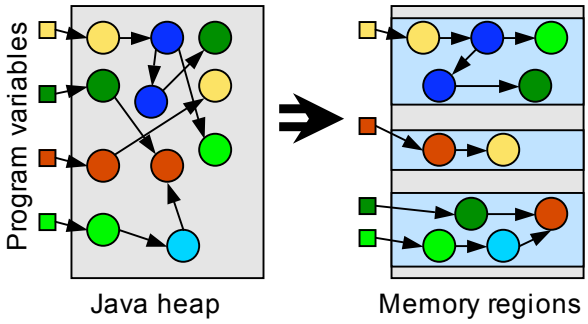
Motivation :

- Java means automatic memory management
- Garbage Collector means problems in a real-time context
  - o Unpredictable pause times
  - o Fragmentation of the heap
- *How can we provide automatic memory management without using a GC ?*

Our approach :

- Use region-based memory management
- Group data structures in regions
- Use a compile-time analyis to place objects in regions

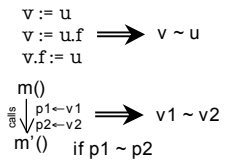
Memory management with regions



- Objects allocated side by side : no more fragmentation
- Regions destroyed as a whole : predictable times
- Drawback: each object must be placed when allocated

Pointer Interference Analysis :

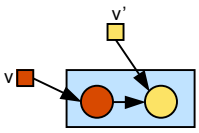
- Build a partition of local variables
- v~v' means they belong to the same data structure
- Simple algorithm :



Allocation Policy :

Simple allocation policy :
if two variables verify v~v',
place their objects in the same region

Data structures will be automatically grouped by region



Other kinds of pointer analysis :

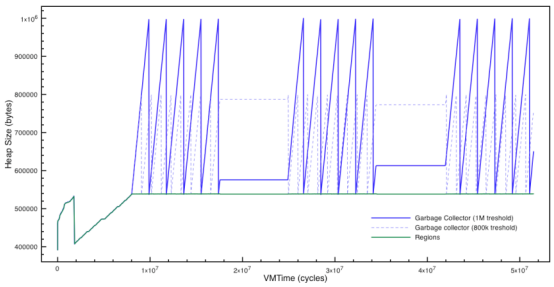
Escape analysis :
Does my object live longer than its method of origin ?

Points-to analysis :
Where do the objects of my variable come from ?

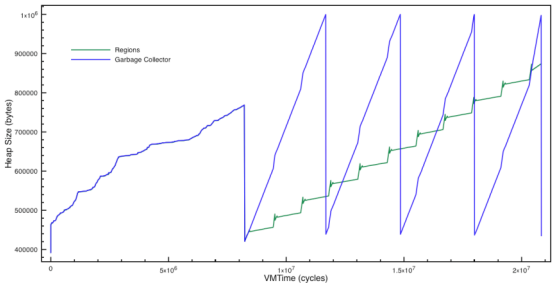
Purity analysis :
Does my method mutate the heap ?

Results :

Memory occupancy for two programs using a Garbage Collector or using Regions



- In this program, most regions are short-lived
- The program runs in nearly constant space
- No more need for a Garbage Collector



- This program uses a large mutating data structure
- Some of the generated garbage stays forever in the long-lived region
- Running this program without a GC may cause a memory leak

How to predict the runtime behaviour ?

Achieved :

- We propose a simple static analysis and allocation policy that groups data structures in regions
- Automatic region-based memory management can allow programs to run without a Garbage Collector

Perspectives :

- This approach has a tendency to place too many objects in the same region
- We need to find an algorithm to predict at compile time the behaviour of the region allocator

This work was partially supported by projects DYNAMO (Min. Research, France) and MADEJA (Rhône-Alpes, France)