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

Guillaume Salagnac

Vérimag
Université Joseph Fourier
Grenoble - France

Motivation

The Java programming language

- Attractive language
- No manual dynamic memory management

Implementation pitfalls

- Non-determinism of Virtual Machines
- Garbage Collector pause times

$\Longrightarrow$ difficult to use in a real-time embedded context

Our approach

Non-determinism of Garbage Collector pause times :
the problem is in the JVM, not in the language

Proposition

- Keep the language
  - no *manual* memory management
- Change the implementation
  - replace the GC by a *controllable* allocator
    - use region-based memory management
  - compute objects lifetimes at compile-time
    - find a reasonnable over-approximation

Our approach

Non-determinism of Garbage Collector pause times :
the problem is in the JVM, not in the language

Proposition

- Keep the language
  - no *manual* memory management
- Change the implementation
  - replace the GC by a *controllable* allocator
    - use region-based memory management
  - compute objects lifetimes at compile-time
    - find a reasonnable over-approximation

Results

Pointer Interference Analysis

- compute relationships between objects lifetimes

Region allocation policy

- automatically place objects into regions

Experiments at runtime

- evaluate the impact on memory behaviour of the programs