Heap Heap Hooray: Improving memory management

Tyler Gutowski, Trevor Schiff, Dr. Ryan Stansifer (client)

Team

- Tyler Gutowski (member)
- Trevor Schiff (member)
- Dr. Ryan Stansifer (faculty advisor, client)

Meetings

- Wednesday, 30 August: Discuss and better understand project idea

Goal

- Improve memory management
- Develop runtime garbage collector (GC)
- Integrate with MiniJava compiler developed as part of Compiler Theory course
    - “MiniJava” refers to a simple, but non-trivial subset of Java
- Document how to optimize GC performance when dealing with certain algorithms

Motivation

- MiniJava runtime does not offer automatic resource management
  - Garbage collector is not a required part of the Compiler Theory course
  - “New” operator exists, but user is responsible for lifetime of allocation
- As MiniJava is a subset of Java, memory cannot be manually freed
  - No “delete” operator exists
  - Without GC, all heap allocations are permanent
  - Losing reference means losing memory block forever

Key Features

- Automated memory management in MiniJava, at runtime
  - “Garbage collection”
-

Novel Features

Technical Challenges

- Understand and implement GC algorithm(s)
- Learn how to integrate GC with MiniJava runtime
- Determine data/algorithm set for GC performance testing

Milestone 1

1. Literature Review
   - Analyse *The Garbage Collection Handbook* (Richard Jones, et. al) to understand GC architecture and algorithms
   - Evaluate strengths and weaknesses of different approaches to GC
   - Examine open-source projects to see real-world examples
2. Requirements Gathering
   - Define project objectives, scope
   - Determine metrics for testing
3. Feasibility
   - Assess project feasibility
   - Identify prospective project challenges
4. Design
   - Select algorithm for GC implementation
   - Develop high-level design and create design documents

Milestone 2

1. Architecture
   - Establish strategy for integration with compiler and runtime
   - Finalize architecture and validate with prototype
2. Tools and Setup
   - Select/setup development tools and environment
   - Select/setup project management tools, such as version control
   - Establish testing framework
3. Coding Phase
   - Begin implementation of GC core components
   - Implement debugging tools
   - Test and demo core components

Milestone 3

1. Coding Phase (cont.)
    - Implement identification and marking
    - Integrate memory management with compiler and runtime
2. Testing and Demoing
    - Address memory leaks
    - Demo added functionality
    - Demo project to the customer