# Copying GC & Reference Counting

Presented By: Jonathan Kalechstain

Date: 11/11/2014

In this lecture chapters 4,5 were presented.

Chapter 4 presents the algorithm of copying Garbage collection.
The main idea is that the heap is divided into two separate equally sized spaces denoted by source, dst. All objects are stored in source, and once collect takes place, all reachable objects from roots in source are moved into dst and the spaces switch jobs (source->dst,dst->source).
The copying is done by BFS using a single pointer added. The chapter also discusses the pros and cons of different traversal orders with regard to cache misses and page faults.

Chapter 5 introduces a few algorithms for reference counting. Reference counting is an algorithm in which every object in the heap contains a reference count field. If that count reaches zero, then no reachable pointer points to that object, and it can be **immediately** freed.
The first algorithm updates the reference counts of all objects in every read and writes operation, costing huge overhead and is extremely inefficient. The chapter also presents deferred reference counting, which suggests postponing some of the updates to a "stop the world" phase. These algorithms do not consider cycles, so the last algorithm in the chapter introduce "The Recycler", which discover cycles by detecting garbage structures. These garbage cycles are detected by pointers deletion that leaves an object reference count greater than zero.

My original contribution was providing proofs of the correctness of the copying collection algorithm. This proof was not presented in the book or elsewhere and was presented fully in the presentation. The presentation also included very extensive presentation and animations of the algorithms.

Discussion in class was mainly about the correctness of the algorithms presented especially in the copying garbage collection algorithm. An alternative to the proof was discussed. Another subject discussed was the importance of different search patterns over the reachability graph (BFS Vs DFS).