Cold Object Segregation

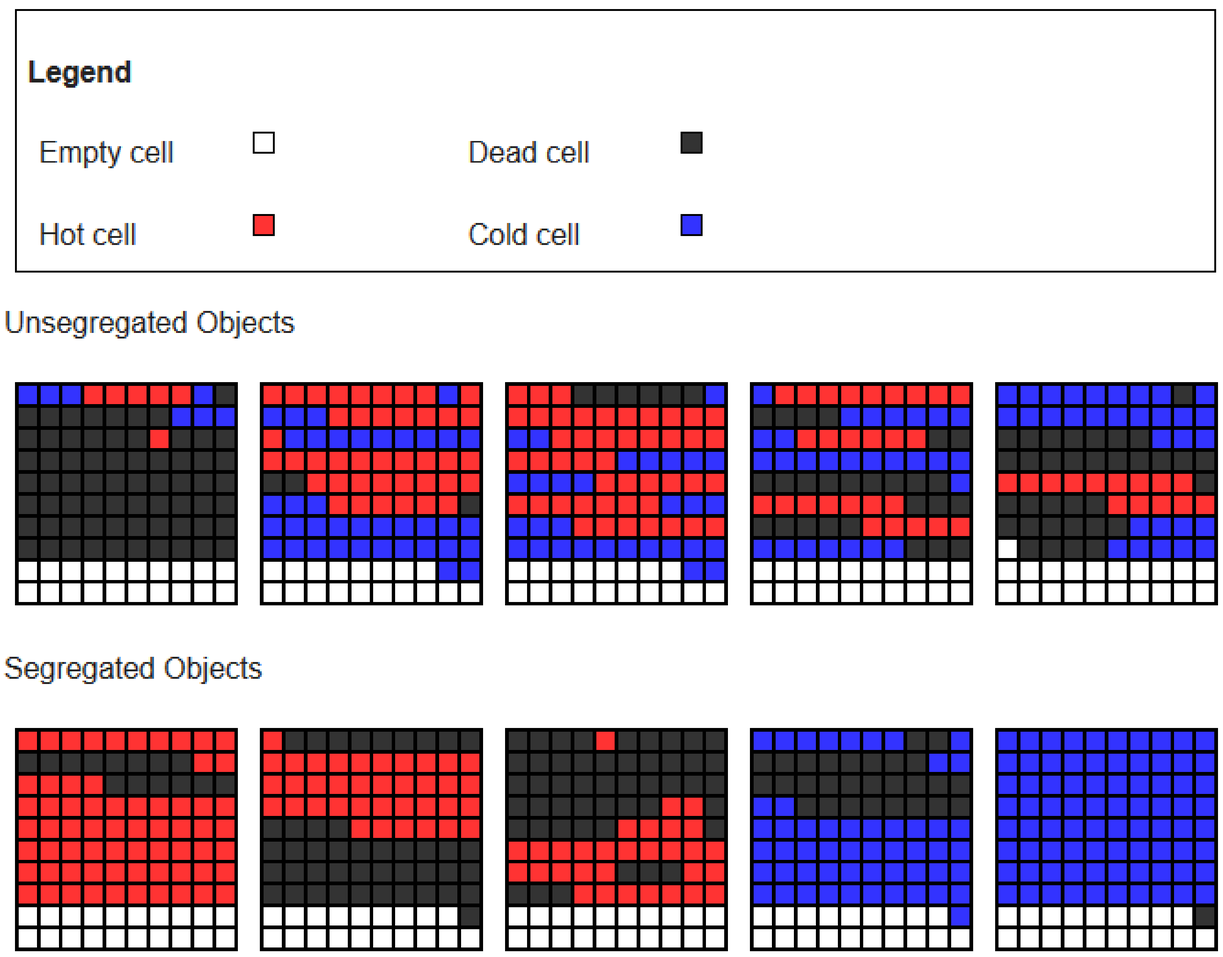
Scott Young, Kenneth B. Kent, Gerhard Dueck
University of New Brunswick, Faculty of Computer Science
Charlie Gracie
IBM Canada
{scott.young, ken, gdueck}@unb.ca, charlie_gracie@ca.ibm.com

Cold objects

Cold objects are live objects that are infrequently referenced. The frequency of reference for a given object changes over time, therefore an object may become cold, or conversely, become hot.

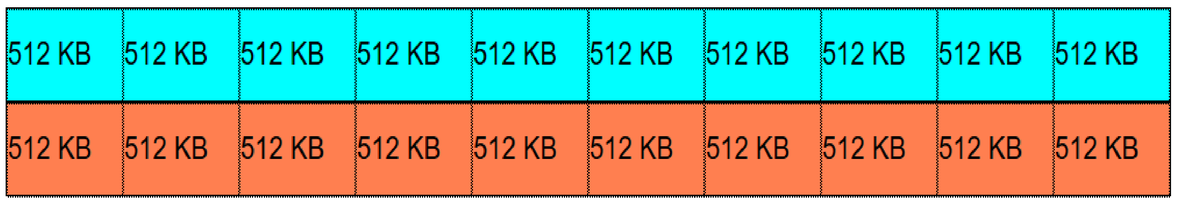
To be considered infrequently accessed, an object must exist for a long time; otherwise it would not have lasted long enough to cool down. These objects will cause the heap to fill faster (since they are less likely to die and must take up space) and will pollute the cache if they are adjacent to hot objects (since they will not likely be referenced) .

If we move all cold objects to their own area of memory, we should speed up execution. We likely increase the cache hit rate and, more importantly, reduce the frequency and execution time of garbage collections, as there is more memory available for hot objects, which are less likely to still be alive when collection occurs.



Region-Based Heap without Cold Memory

Region-Based Heap with Cold Memory



Region-Based Copy-Forward Garbage Collection

Region-based memory management schemes logically divide their heap into fixed-sized regions. These regions maintain information about incoming references in a structure called a remember set (remset).

In a Copy-Forward Garbage Collector, collection happens by scanning objects starting at the root set (thread stacks) and all their descendants, copying into a survivor space and updating pointers as new live objects are discovered.

In a Region-Based Copy-Forward Garbage Collector, a subset of all regions are chosen to be collected, and are copied into free regions. The remset is used as the root set as it contains all incoming references.

Segregating objects

For decades, secondary storage has been used as a means to increase the effective size of memory through paging. It is possible to memory map files in secondary storage and use them as slower memory.

Since cold objects are infrequently accessed, we can store them in secondary storage to allow for more live objects in the main memory heap, decreasing the frequency of garbage collections.

Since objects tend to die young (due to the Generational Hypothesis) and cold objects are all long lived-objects, moving them to secondary memory should lead to a higher concentration of garbage in the main memory heap. Copy collectors operate on live objects only, and their runtime is linked to the number of live objects in the collection space. This means that the collector should also collect faster, if we segregate cold objects.