Identifying and Grouping Cold Objects

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

Object Temperature

A Hot object is an object that has been used frequently in recent history..

A Cold object is an object that is not dead, but has not been used for a long time. These objects may stay cold for some time, may not ever become Hot, and may not even ever be used again.

Many different types of objects can become cold, such as partial error strings or properties objects.

Cold Object Identification

Several methods have been attempted for identifying Cold Objects.
This includes:

- Local Stack Walking & Pinned Regions
- Access Counting

Exploration

- The Cost-Benefit trade off of storing the Cold Object Region in various locations.
- The Cost-Benefit trade off of using more or less precise Identifying techniques.
- How changing storage location changes the cost and benefit of each cold object identifying technique.

Goal

The goal is to eventually use the data from exploration to improve the performance of the JVM. It is currently unknown whether this region should remain in main memory, be paged out to disk, or if there is no one size fits all answer.

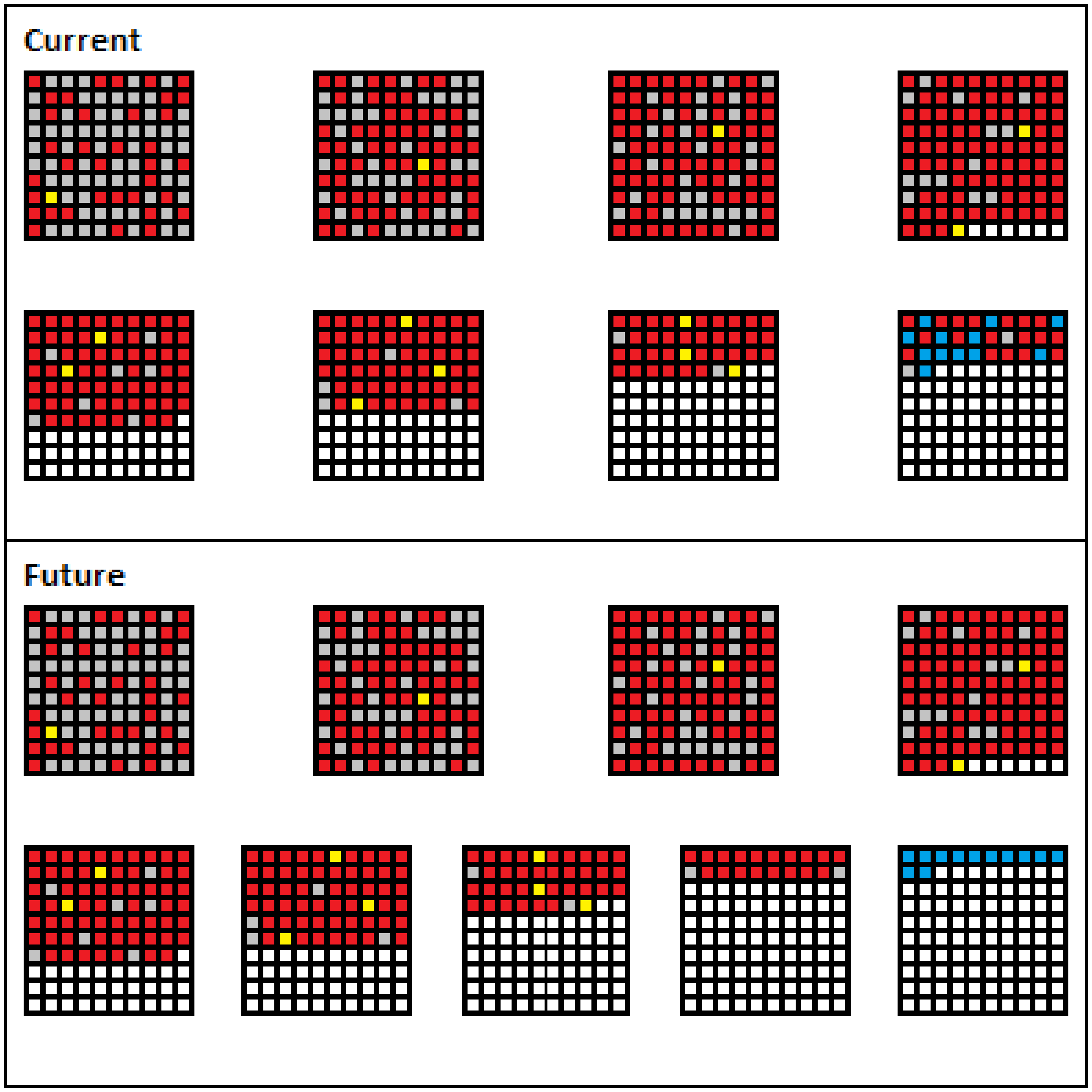
Generational Garbage Collection

In Generational Garbage Collection the heap is divided into regions. Objects that live through a number of GCs are copied to the next region.

Generational Hypothesis: “Objects tend to die young”.

This means we can collect some regions (Partial GCs) without collecting all the regions (Global GCs).

If we move Cold Objects to their own region then programs will benefit from improved caching, as the cold objects won’t be interleaved on the same cache lines as the other objects. The figure below shows what a Generational GC heap might look like before a Global GC today, and after implementing cold object identification and grouping.


Current
Future