Garbage Collection

A Fixed Point Problem

Brian Gernhardt - 2014-02-16

A Unified Theory of Garbage Collection

David F. Bacon
dfb@watson.ibm.com

Perry Cheng
perryche@us.ibm.com

V.T. Rajan
vtrajan@us.ibm.com

IBM T.J. Watson Research Center
P.O. Box 704
Yorktown Heights, NY 10598

ABSTRACT

Tracing and reference counting are uniformly viewed as being fundamentally different approaches to garbage collection that possess very distinct performance properties. We have implemented high-performance collectors of both types, and in the process observed that the more we optimized them, the more similarly they behaved — that they seem to share some deep structure.

We present a formulation of the two algorithms that shows that they are in fact duals of each other. Intuitively, the difference is that tracing operates on live objects, or "matter", while reference counting operates on dead objects, or "anti-matter". For every operation performed by the tracing collector, there is a precisely correspond-

1. INTRODUCTION

By 1960, the two fundamental approaches to storage reclamation, namely tracing [33] and reference counting [18] had been developed.

Since then there has been a great deal of work on garbage collection, with numerous advances in both paradigms. For tracing, some of the major advances have been iterative copying collection [15], generational collection [41, 1], constant-space tracing [36], barrier optimization techniques [13, 45, 46], soft real-time collection [2, 7, 8, 14, 26, 30, 44], hard real-time collection [5, 16, 23], distributed garbage collection [29], replicating copying collection [34], and multiprocessor concurrent collection [21, 22, 27, 28, 39].

A Unified Theory of Garbage Collection
Bacon, Cheng, and Rajan, OOPSLA 2004

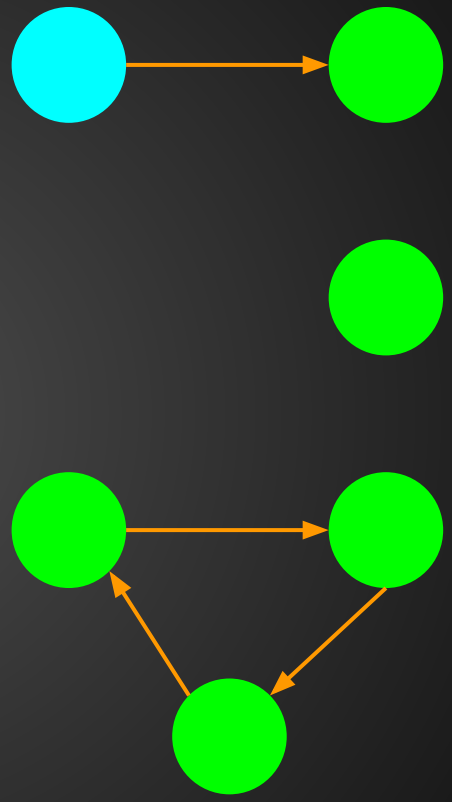
Object Graph

V - Vertices, Objects

E - Edges, Pointers

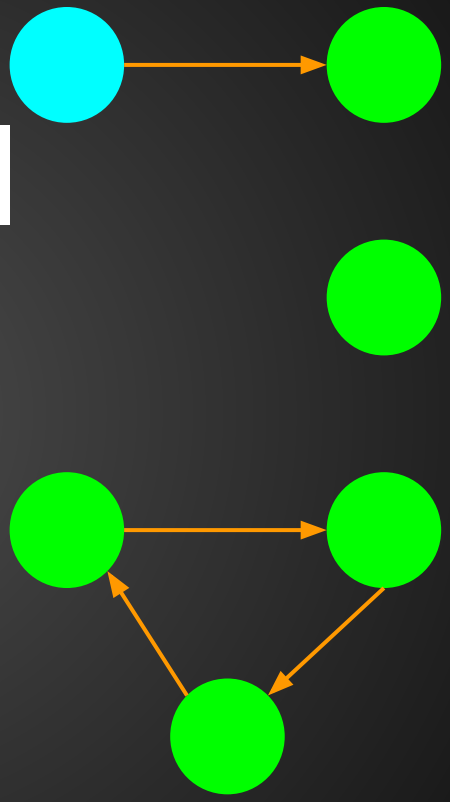
R - Roots, Globals

ρ - Reference Count



ρ = Reference Count

$$\rho(x) = \left|[x : x \in R]\right| + \left|[(w, x) : (w, x) \in E \wedge \rho(w) > 0]\right|$$

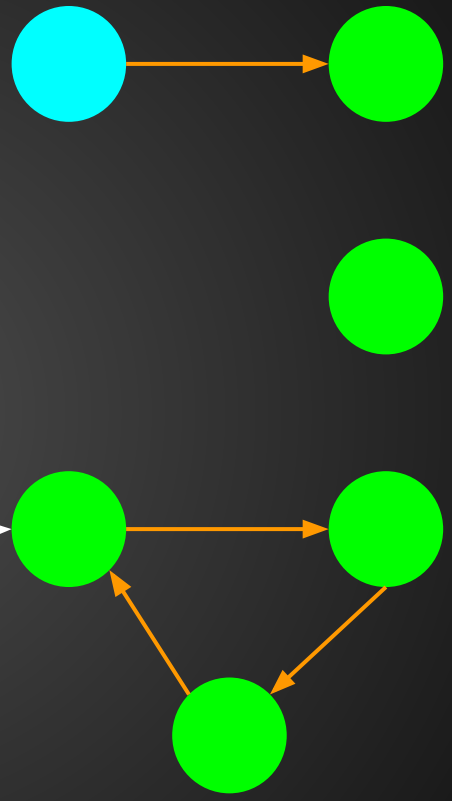


Live: ρ > 0

Dead: ρ = 0

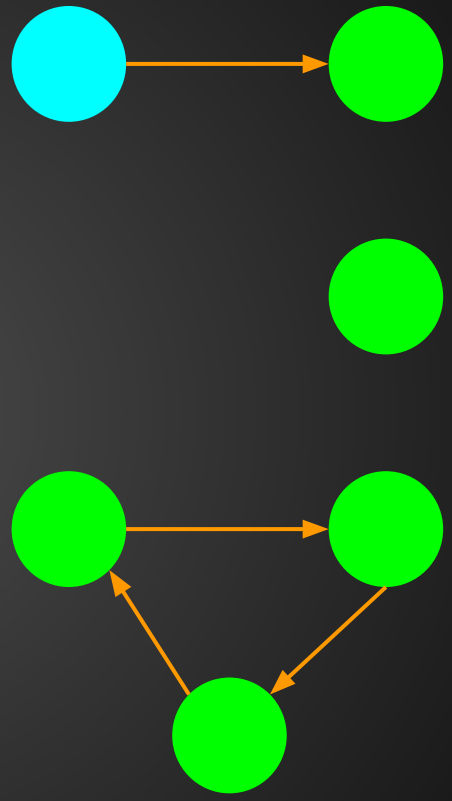
roots + # links from live objects

Cycles



What are the counts
for the cycle here?

Reference Counting



Create a pointer:

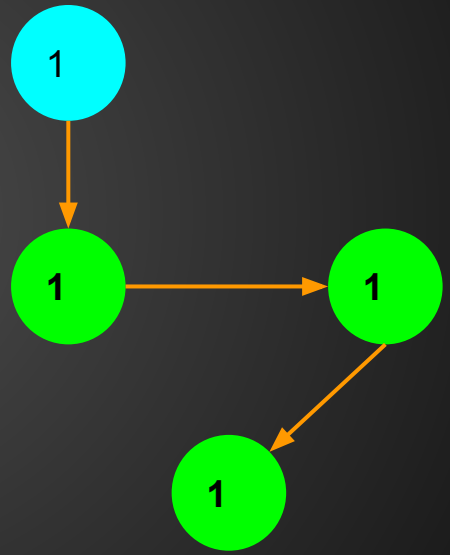
increment count

Remove pointer:

Decrement count

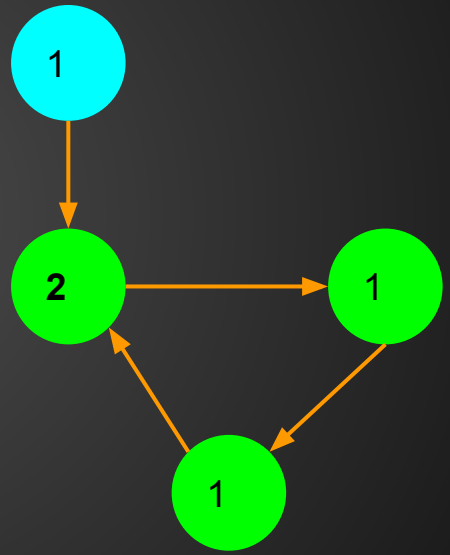
Circular List Example

Increment count as we add each node to the list.



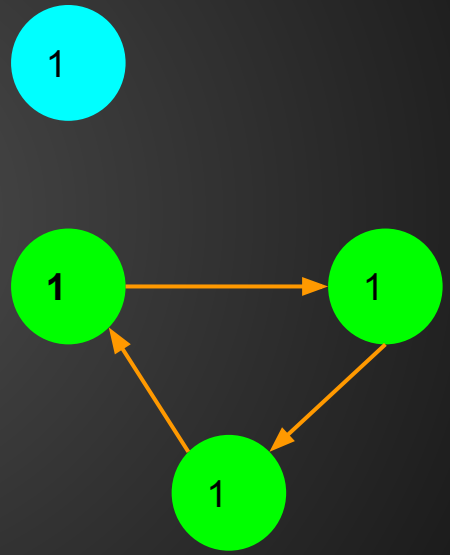
Circular List Example

Increment count when
we create cycle



Circular List Example

Decrement count
when we are done
with list



Tracing

Set all counts to 0

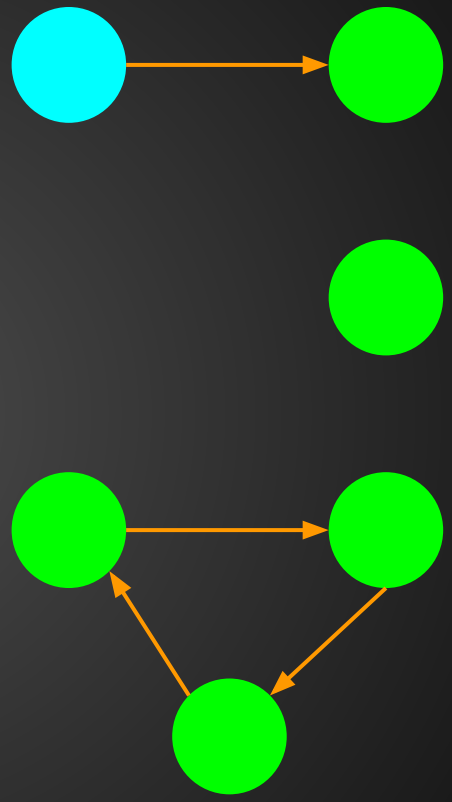
Add roots to list

For each object on list:

    increment count

    add connected

        objects to list



Comparison

Reference Counting

- Fast
- Immediate Update
- Cycle Problems

Tracing

- Simple
- Accurate
- Slow

Greatest Fixed-Point

Least Fixed-Point