Boehm garbage collector

From Wikipedia, the free encyclopedia

In computer science, the **Boehm–Demers–Weiser garbage collector**, often simply known as **Boehm GC**, is a conservative garbage collector for C and C++.[1]

Boehm GC is free software distributed under a permissive free software licence similar to the X11 license.

Boehm–Demers–Weiser Garbage Collector

Written in	C and C++
Type	garbage collector
License	similar to X11 (free software)
Website	http://www.hboehm.info/gc/

Design [edit]

The developer describes the operation of the collector as follows:

> The collector uses a mark-sweep algorithm. It provides incremental and generational collection under operating systems which provide the right kind of virtual memory support. (Currently this includes SunOS[45], IRIX, OSF/1, Linux, and Windows, with varying restrictions.) It allows finalization code to be invoked when an object is collected. It can take advantage of type information to locate pointers if such information is provided, but it is usually used without such information.
>
> — http://www.hboehm.info/gc/#details

Boehm GC can also run in leak detection mode[2] in which memory management is still done manually, but the Boehm GC can check if it is done properly. In this way a programmer can find memory leaks and double deallocations.

Boehm GC is also distributed with a C string handling library called cords. This is similar to ropes in C++ (strings are trees of small arrays, and they never change), but instead of using reference counting for proper deallocation, it relies on garbage collection to free objects. Cords are good at handling very large texts, modifications to them in the middle, slicing, concatenating, and keeping history of changes (undo/redo functionality).

Operation [edit]

The garbage collector works with most unmodified C programs, simply by replacing malloc() with GC_MALLOC() calls, replacing realloc() with GC_REALLOC() calls, and removing free() calls.[1] The code piece below shows how one can use Boehm instead of traditional malloc and free in C.[3]

```
#include <assert.h>
#include <stdio.h>
#include <gc.h>

int main(void)
{
    int i;

    GC_INIT();
    for (i = 0; i < 10000000; ++i)
    {
        int **p = GC_MALLOC(sizeof(int *));
        int *q = GC_MALLOC_ATOMIC(sizeof(int));
```

Wikipedia The Free Encyclopedia

Main page
Contents
Featured content
Current events
Random article
Donate to Wikipedia
Wikipedia store

Interaction
Help
About Wikipedia
Community portal
Recent changes
Contact page

Tools
What links here
Related changes
Upload file
Special pages
Permanent link
Page information
Wikidata item
Cite this page

Print/export
Create a book
Download as PDF
Printable version

Languages
Deutsch
Polski
中文
Edit links

```
        assert(*p == 0);
        *p = GC_REALLOC(q, 2 * sizeof(int));
        if (i % 100000 == 0)
            printf("Heap size = %zu\n", GC_get_heap_size());
    }

    return 0;
}
```

Uses and ports [edit]

The Boehm GC is used by many projects that are implemented in C or C++ like Inkscape, as well as by runtime environments for a number of other languages, including the GNU Compiler for Java runtime environment, the Portable.NET project, Embeddable Common Lisp, GNU Guile, the Mono implementation of the Microsoft .NET platform (also using precise compacting GC since version 2.8), and libgc-d (a binding to libgc for the D programming language, used primarily in the MCI). It supports numerous operating systems, including many Unix variants (such as Mac OS X) and Microsoft Windows, and provides a number of advanced features including incremental collection, parallel collection and a variety of finalizer semantics.

References [edit]

1. ^ a b Koranne, Sandeep (2011), *Handbook of Open Source Tools*, Springer, pp. 151–154, ISBN 1441977198.
2. ^ Using the Garbage Collector as Leak Detector
3. ^ Using the Garbage Collector: A simple example

External links [edit]

- Official website
- Boehm garbage collector on SourceForge.net
- Git repo for BoehmGC development
- Transparent Programmer-Directed Garbage Collection for C++, Hans-J. Boehm and Michael Spertus
- Using the C/C++ Garbage Collection Library
- Dr. Dobbs The Boehm Collector for C and C++, Gene Michael Stover, March 01, 2003

v · t · e	**Memory management**	[hide]
Memory management as a function of an operating system		
Manual memory management	Static memory allocation · C dynamic memory allocation · new (C++) · delete (C++)	
Virtual memory	Demand paging · Page table · Paging · Virtual memory compression	
Hardware	Memory management unit · Translation lookaside buffer	
Garbage collection	**Boehm garbage collector** · Finalizer · Garbage · Mark-compact algorithm · Reference counting · Strong reference · Weak reference	
Memory segmentation	Protected mode · Real mode · Virtual 8086 mode · x86 memory segmentation	
Memory safety	Buffer overflow · Buffer over-read · Dangling pointer · Stack overflow	
Issues	Fragmentation · Memory leak · Unreachable memory	
Other	Automatic variable · International Symposium on Memory Management · Region-based memory management	

Categories: Automatic memory management | C++ libraries | C libraries | Free compilers and interpreters | Memory management software