Conservative Garbage Collection

sk and er

11/27/2000

Conservative Garbage Collection

C and C++ do not come with a garbage collector. People even think that it's impossible to use garbage collection with C or C++. From the garbage collection perspective, the biggest differences between C and, say, Scheme, are that in C:

1. On the heap, everything may look like a pointer.
2. Because of pointer arithmetic, even things that don't look like pointers may be hiding pointers.

From the collector's perspective, the language (in combination with its minimal runtime environment) is considered hostile.

However, these problems do not prohibit garbage collection. The solution to the first problem is conservative garbage collection: if something looks like a pointer, then we consider it a pointer and traverse it. This algorithm might follow some links that are not pointers, but it will not miss anything that has to be followed. As for the second problem, ANSI C and C++ provide a number of guarantees about what a pointer can disguise as. (There is a handful of very "clever" techniques that break these guarantees, but if you write code using these techniques, you are probably better off managing your own memory, and, in any case, you deserve what you get).

Techniques and Modifications

Which of the techniques we have covered in class can be used directly in a conservative garbage collector? The basic mark-and-sweep technique is still applicable, although we need to redefine 'looks like a pointer' and 'points to' to accommodate for the two problems above. Stop-and-Copy garbage collection doesn't work (why?) and Generational garbage collection, the way we've seen it, requires copying.

What exactly is our definition of 'looks like a pointer'? The garbage collector knows exactly the address where the heap starts and the address where the heap ends; any bit pattern which, when looked at as a pointer, is between those two

values, is considered to be a pointer. In practice, most data will not look like like a pointer under this definition, because the heap space is relatively small compared to the whole address space. Also, most numbres in programs are small integers, whereas heap space is usually in the middle of the address space.

The original (non-conservative) definition of 'points to' is simple: if the memory allocator gave the program a memory region starting at address A, then a pointer points to that if its value is equal to A. The conservative definition of 'points to' is a generalization of this: if the memory allocator gave a program a memory region of size n, starting at address A, then a pointer points to the region if its value is between A and A + n. (Note that this is one plus the size of the area. This is to cover the legal, under ANSI C, case of a pointer that points to just one position after the end of an array.) This generalization requires a sophisticated data structure to identify what a pointer points to. (For intimidation value: `http://www.hpl.hp.com/personal/Hans_Boehm/gc/tree.html`)

Usage, and why you should care

Writing a conservative garbage collector is no small task. Thankfully for all of us, very smart people have spent a lot of time writing a portable (and widely ported) one. It's at `http://www.hpl.hp.com/personal/Hans_Boehm/gc/`. A number of systems use it, like DrScheme, XeroxDocuPrint (software in the Xerox printers), gcj (the Java part of the GNU compilers). It's fast: on my measurements, it was no slower than the standard system malloc (and no leaks :-). It's fairly easy to use: on our system, all one has to do to convert a program using malloc and free to a garbage collected program is `setenv LD_PRELOAD /cs/lib/libgc-malloc.so` If you start coding in C with garbage collection in mind, you should code to the garbage collector's API. The most notable difference is that you call GC_malloc instead of malloc. Garbage collection with C++ takes a little more effort, as one has to overload the global new operator, or use a mixin for the garbage collected classes.

Sun has included with Workshop 6, a version of Boehm's garbage collector with embellishments from a company called Geodesic Systems. The web page of that part of Workshop is file:/opt/SUNWspro6.0/WS6/htmldocs/locale/C/gc/toc.htm Note the last phrase of Appendix A: Garbage collection is becoming the mainstream in C++. The more you understand how C++ garbage collection works, the better you will be able to decide when and how to use it in your programs.