CMPS 111

Memory Management Library

Overview

1. Library
   a. Requests block of memory using `malloc`
   b. Manages memory within block using an allocator
   c. Interface:
      i. `meminit(long n_bytes, unsigned int flags, int parm1)`
      ii. `memalloc(int handle, long n_bytes)`
      iii. `memfree(void *region)`
   d. Allocators: buddy, free-list

Overview

2. Test workload(s)
    a. Allocate memory using library's interface
        i. Multiple allocators same/different types
        ii. Handle tells `memalloc` which allocator to use
    b. Request only large or small blocks, or a combination
3. Experiments
    a. Compare best fit, random fit, worst fit
    b. Another of your choosing (e.g. Under what kind of workload does buddy allocation work well?)

Building a static library

- `libmem.a`
  - Archive format, contains multiple object files
  - Suggestion: Memory manager, buddy allocator, free list allocator each a separate object file
- Resources
  - http://en.wikipedia.org/wiki/Static_library
  - http://gcc.gnu.org/onlinedocs/gcc/Link-Options.html
  - http://linux.die.net/man/1/ar

Designing the library interface

- `meminit` (returns `handle`)
  - `int n_bytes`
    - Size of requested block
  - `unsigned int flags`
    - Allocator options (see specification)
  - `int parm1`
    - Buddy allocator: min page size, in address bits
    - 12 means min page size is 2^12 = 4096 bytes
    - If not a power of 2, return an error (-1)

Designing the library interface

- `memalloc` (returns pointer to region start)
  - `int handle`
    - Non-negative integer that refers to an allocator initalized by `meminit`
  - `int n_bytes`
    - Size of requested block (at most the size of the memory block passed to `meminit`)

Note: (void *) gives raw access to region. Casting to, e.g. char * allows use of region as C-string

Designing the library interface

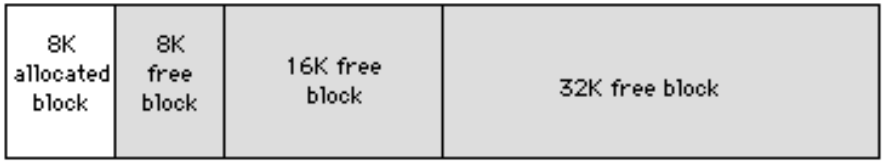
- `memfree` (deallocates block containing region)
  - `void *region`
    - find which allocator manages memory that contains this address
    - instruct allocator to deallocate the corresponding block
    - if no allocator manages memory that contains this address, do nothing

Buddy Allocator

- Fixed block sizes (2^n or Fibonacci)
  - One free list for each size
- Relatively easy to merge (coalesce)
  - Sum of newly freed block and an adjacent free block (the “buddy”) equals size of whole block
- Resources:
  - http://www.memorymanagement.org/articles/alloc.html#buddy.system
  - http://pages.cs.wisc.edu/~remzi/OSTEP/vm-freespace.pdf

Buddy Allocator - Example

Blocks: 8/16/32/64kB



Allocate 8kB

Allocate 10kB

Free 8kB (coalesce)

Free-list Based Allocator

- Variable-size blocks
  - Allocate
    - Use algorithm: first, next, best, worst, random
  - Free
    - Coalesce (check for adjacent free blocks, merge with them to obtain larger free block)
- Resources:
  - http://pages.cs.wisc.edu/~remzi/OSTEP/vm-freespace.pdf

Experiments

- To be covered this Thursday