mimalloc研究报告

co lin

242

前言

mimalloc mimalloc

线程本地数据、

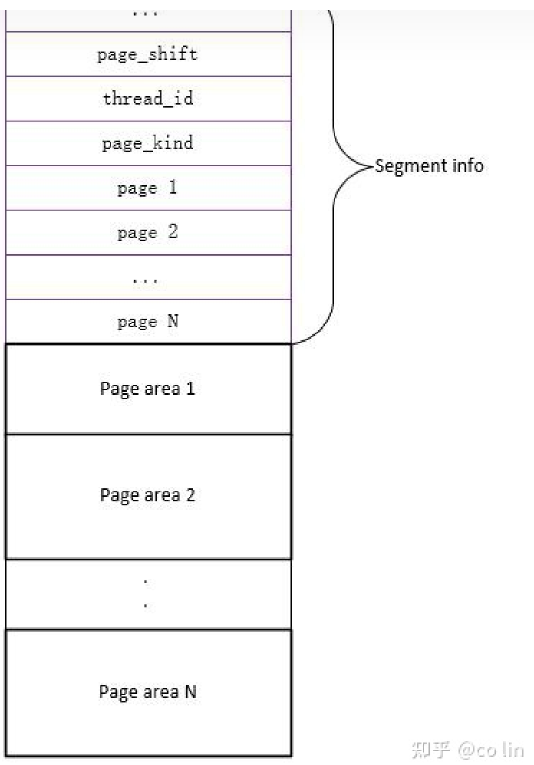
mimalloc

mimalloc (tld)
tld

- segment segment slab arena OS
  segment tld segment
- heap
  (page) page segment page segment
  page (block)

内存段(segment)

segment



segment (segment info) N page mi_segment_t

- thread_id
- page_kind page
  64kb 512kb 4Mb 4Mb
- page_shift
- page1~pageN

~
1 segment N page 1
1

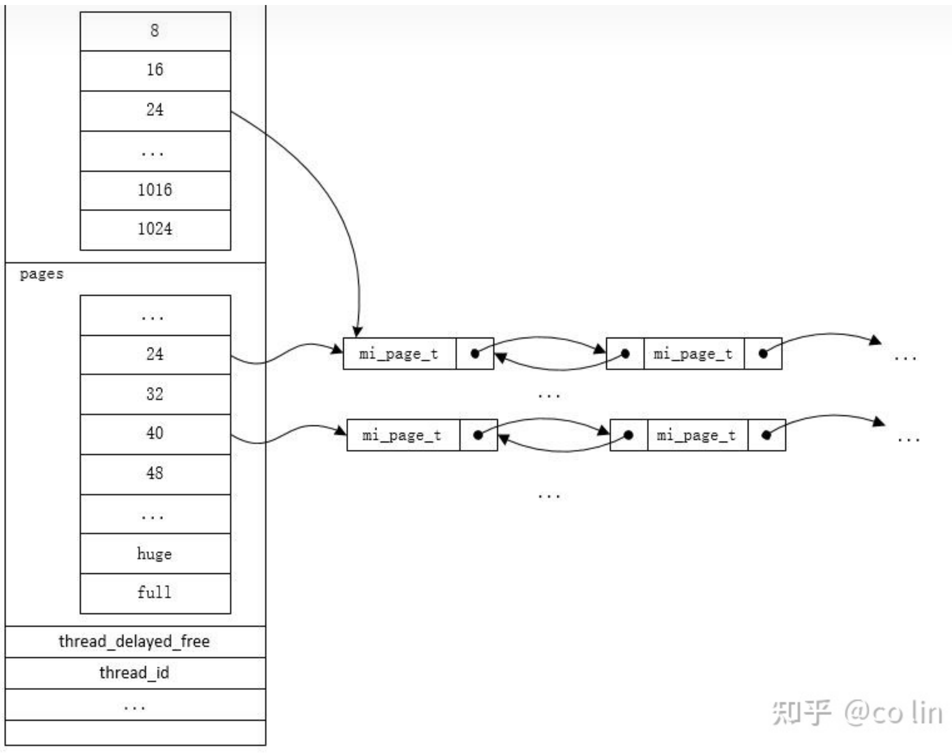
segment 4Mb segment
segment

```
static inline           * _mi_ptr_segment const void*
  return       *        =         *               & ~  *
```

SMALL, MEDIUM, LARGE segment 4Mb HUGE segment
HUGE segment

线程堆(heap)

segment heap size
class



pages mi_page_queue_t mi_page_queue_t page
size page pages segment page

pages_free_direct mi_page_t* 1024 page
(64 8) mi_page_t pages_free_direct
pages pages pages_free_direct

pages_free_direct heap size page

```
* = -> + >> //
```

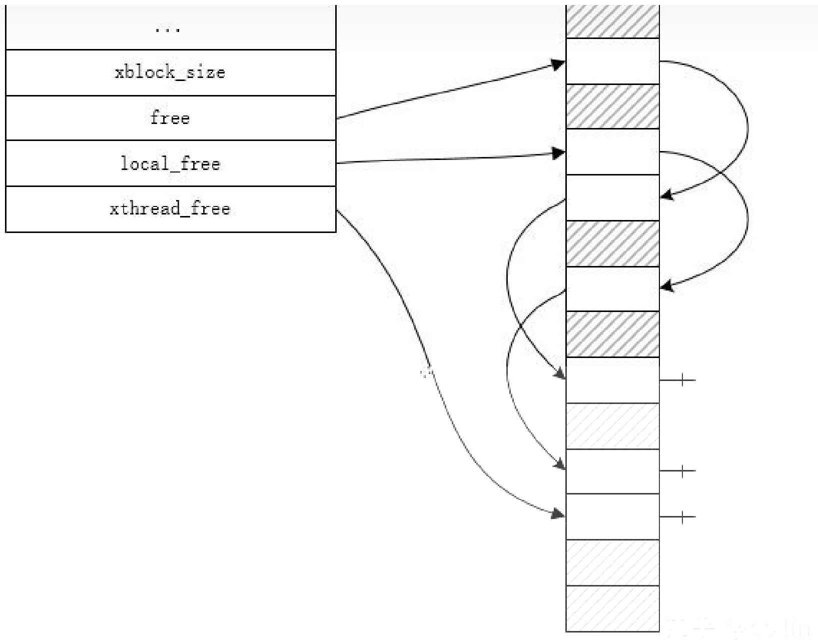
1024 pages size
mi_page_queue_t mi_page_t pages_free_direct
size class

pages full page(
) full
full
mimalloc

thread_delayed_free

内存页(page)

(mi_page_t)



(xblock_size) freelist (block)
mi_page_t freelist mimalloc

- free
- local_free free local_free
  local_free free free
- xthread_free free local_free
  xthread_free mimalloc

```
void* _mi_page_malloc          *                *
               * const      =     ->
  if        ==
    return                                   // slow path

  // pop from the free list
      ->    ++
      ->    =      ->
  return
```

free

(_mi_malloc_generic) local_free
free
local_free
" "

local_free thread_free

```
void _mi_page_free_collect          *
                                                               atomic operation
```

```
// and the local free list
if      ->          !=
  if      ->     ==
    // usual case
      ->     =     ->
      ->         =
```

thread_free free local_free
local_free local_free free

thread_free

```
void atomic_push         **              *
  do        ->     = *
  while  !                                    ->


           &    ->
```

lock free _mi_page_thread_free_collect thread_free

```
    =              &    ->
        ->
```

thread_free free thread_free
thread_free page

mimalloc freelist

分配和释放

mimalloc

```
void* mi_malloc
        *      =                          // 取线程相关的堆
    return


void* mi_heap_malloc          *
    if       <=                      // 如果<=1024，进入小对象分配
        return
      else      // 否则进行通用分配
        return
```

```
          *       =      ->
    if         !=                          // fast path
           ->      =       ->
           ->     ++
      return
    else
      return                                     // slow path



void* mi_malloc_generic           *
               =                          // 计算得到 size class
                   *       =      ->              // 取相应的page队列

                              // 收集页里的可用内存
      if      ->      ==      // 整个页都空闲，回收掉

        else if      ->      !=            // 收集完如果有可用内存，则重回分配入口
          return



    // 到这儿表明找不到可用的page，从segment分配一个新鲜的page



void page_collect
  // 先收集thread free list
  if                            !=



  // 然后才是local free list
  if      ->            !=
    if      ->      ==
      // usual case
        ->      =      ->
        ->            =
```

```
void mi_free void*
            *         =           *               & ~  *       // 找到对应的segment
    if          ==        return
    // 找到对应的page，这是简化过的，第1个page要特殊处理。
    // 因为segment等分成N个page，这里只需要取相对地址，然后除去page的大小，即得到page的索引。
         *      = &        ->          -          >>        ->
          *      =         *


    if                 ==         ->              // 相同线程，释放到local_free
            ->      =      ->
           ->            =
           ->     --
      if      ->      ==

    else   // 不同线程，释放到 thread_free
```

满页的处理(full page)

heap.pages full page mimalloc
page
mimalloc

mimalloc heap.pages full page
full full
full page

full page
page full

mimalloc heap
page thread_free
heap CAS
page thread_free page block
thread_delayed_free

page heap thread_delayed_free
page page
page full " "

page thraed_free
full heap
thread_delayed_free page full page
" " " "

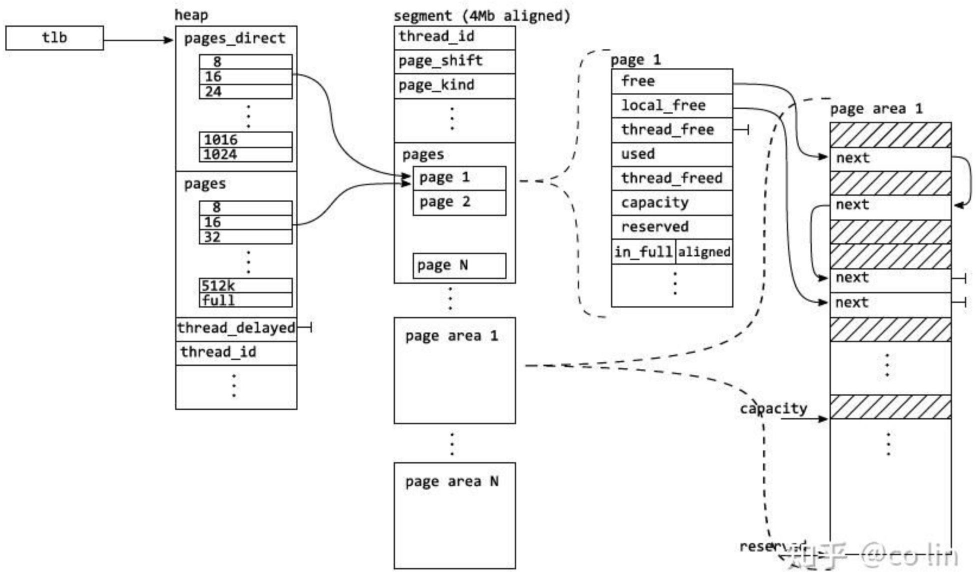
full page

总结

mimalloc

1. freelist freelist size class
   freelist CPU
2. mimalloc lock-
   free
3.

paper Heap Layout



- mimalloc on github
- Mimalloc: Free List Sharding in Action
- mimalloc-bench

2021-05-06 23:36

C / C++

文章被以下专栏收录

游戏服务器编程

推荐阅读

高性能内存分配库的研发小结

malloc C/C++

Google tcmalloc (Googl...

HaiSQ...

C++内存模型小记

CPU

...

漫谈内存一致性模型

API

...

co li... ...

Memory mapping (Includes File, Li Anonymous map

3G

Heap, grow

内存管理：小结

...

Tools boy 2021-05-31

mimalloc v2.0.1 slice page

co lin ( ) Tools boy 2021-05-31

2021-05-11

👍