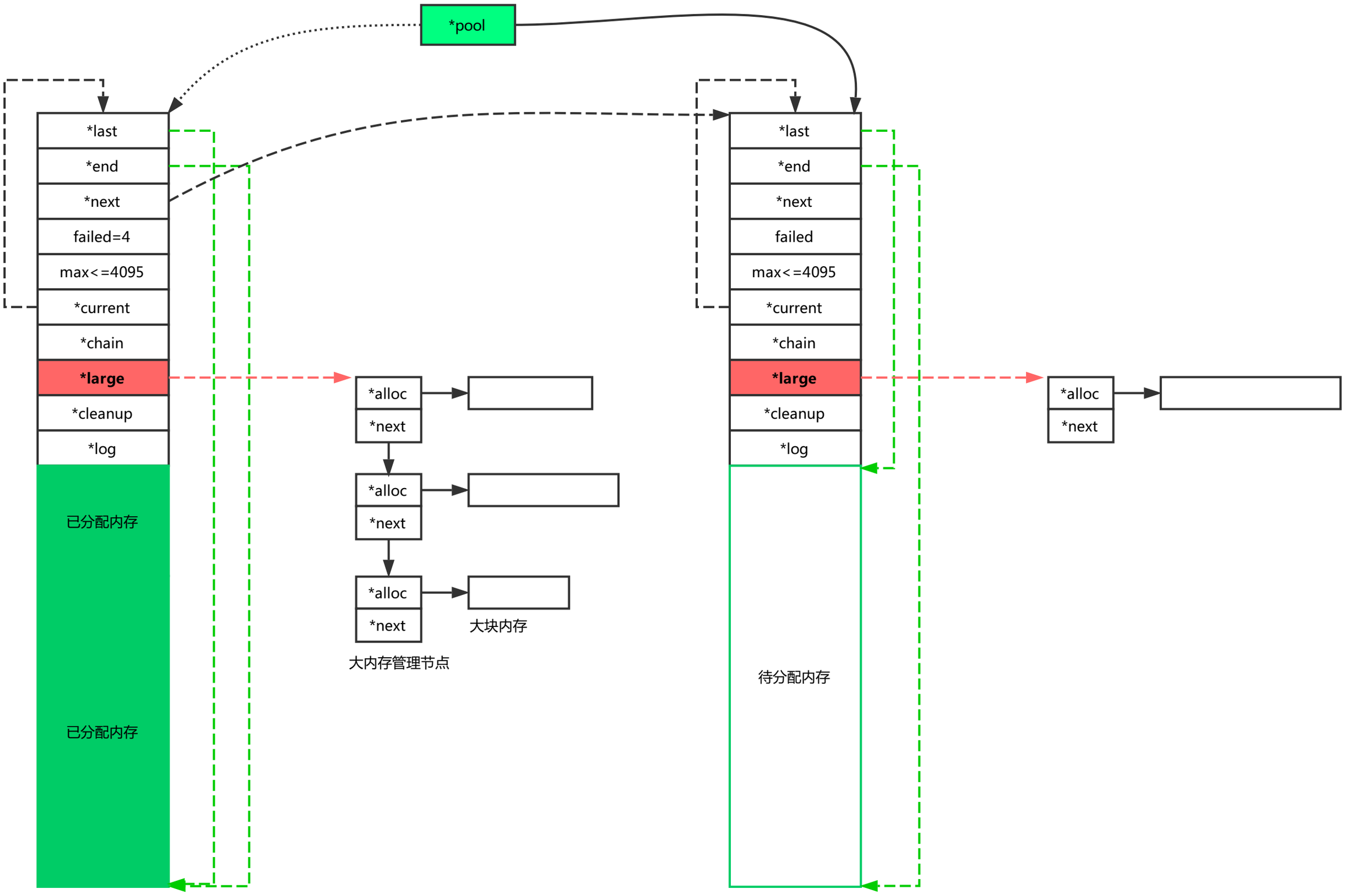


内存池管理单元

**内存池管理单元由系统一次申请p = ngx_memalign(NGX_POOL_ALIGNMENT, size, log),所以管理信息和实际可配置的内存区域是连续的,意味着加入申请1024Byte内存,32位系统上管理信息占据40Byte,实际可配置内存只有984字节.若用户在此内存池中申请1024Byte的内存,则会在large中存储.**
**大内存和大内存管理节点是分开申请的,因此内存不连续.p = ngx_alloc(size, pool->log),large = ngx_palloc(pool, sizeof(ngx_pool_large_t))**

**假设当前内存池管理单元配置的max=1024,那么用户请求小于等于1024字节的情况下会在[last,end]区域分配内存;若用户请求大于1024,则先申请大内存块,然后申请大内存管理节点,挂载大内存块到大内存管理节点,挂载大内存管理节点到large链表.**

**若在pool->current的内存池管理单元,分配小块内存失败次数累计达到4,pool->current切换到第二个内存池管理单元.**

**pool = ngx_create_pool(size,log)调用内存池创建接口后,由size指定了[last,end]的范围大小,之后再申请的新的内存池管理单元会与第一次创建内存池的大小一致.其实现为psize = (size_t) (pool->d.end - _char *) pool),因此后面申请的内存池管理单元大小会与第一次的一致.此外后面申请的内存池管理单元的max一定会小于等于第一次创建内存池算出的max,这样会导致后面的小块内存区域可分配的小块内存个数会越来越多,这是由于同样的空间大小不变,但是内存单元越来越小.**

**假设申请size大小**
**小块内存分配:先返回last指针所指地址,然后last=last+size**

**大块内存分配:先分配size大小的内存块p,然后遍历large链表,查询是否存在大内存管理节点的alloc为NULL,若存在则直接挂载alloc=p,遍历次数最大为3.若3次还没找到,则分配一个大内存管理节点,挂载大内存块,挂载大内存管理节点.**