水位线基本情况及优化实践

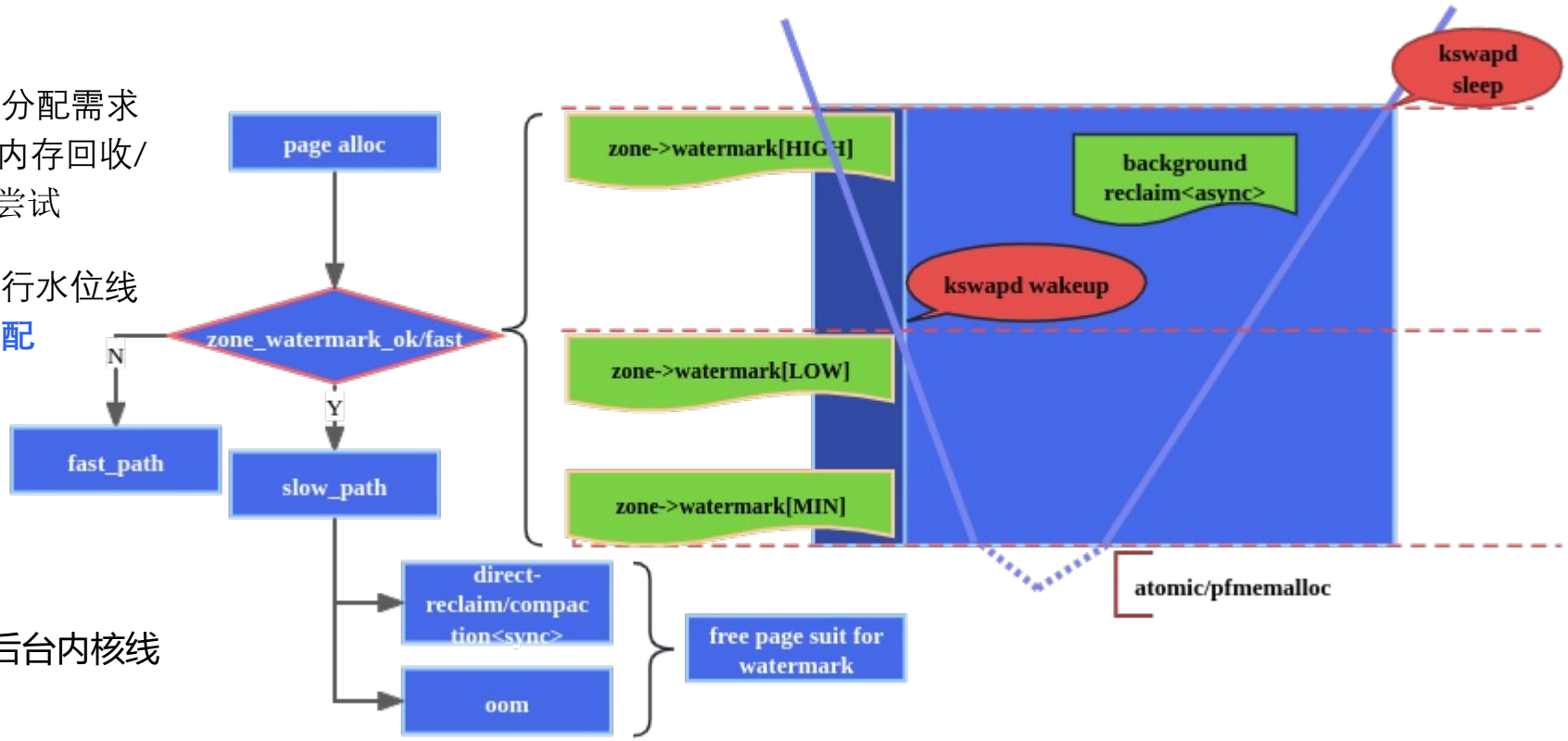
黄锦慧
<huangjinhui@kylinos.cn>
麒麟软件有限公司

目录

1.水位线基本情况

分配路径

- 通过**对比水位线**确定不同的分配路径
- 通过**zone_watermark_ok/fast**进行，满足分配需求后，内存低于MIN水位线，需要进行直接内存回收/内存规整/OOM同步内存回收后进行再次尝试
- 在忽视水位线的标记的分配中，可以不进行水位线检查，**甚至突破水位线的限制进行内存分配**



回收路径

当前**内存水平和水位线**进行比较:

- 在**低于LOW水位线时,**唤醒kswapd异步后台内核线程进行内存页面回收
- 在**高于HIGH水位线后**，停止页面回收工作

2.水位线影响因素

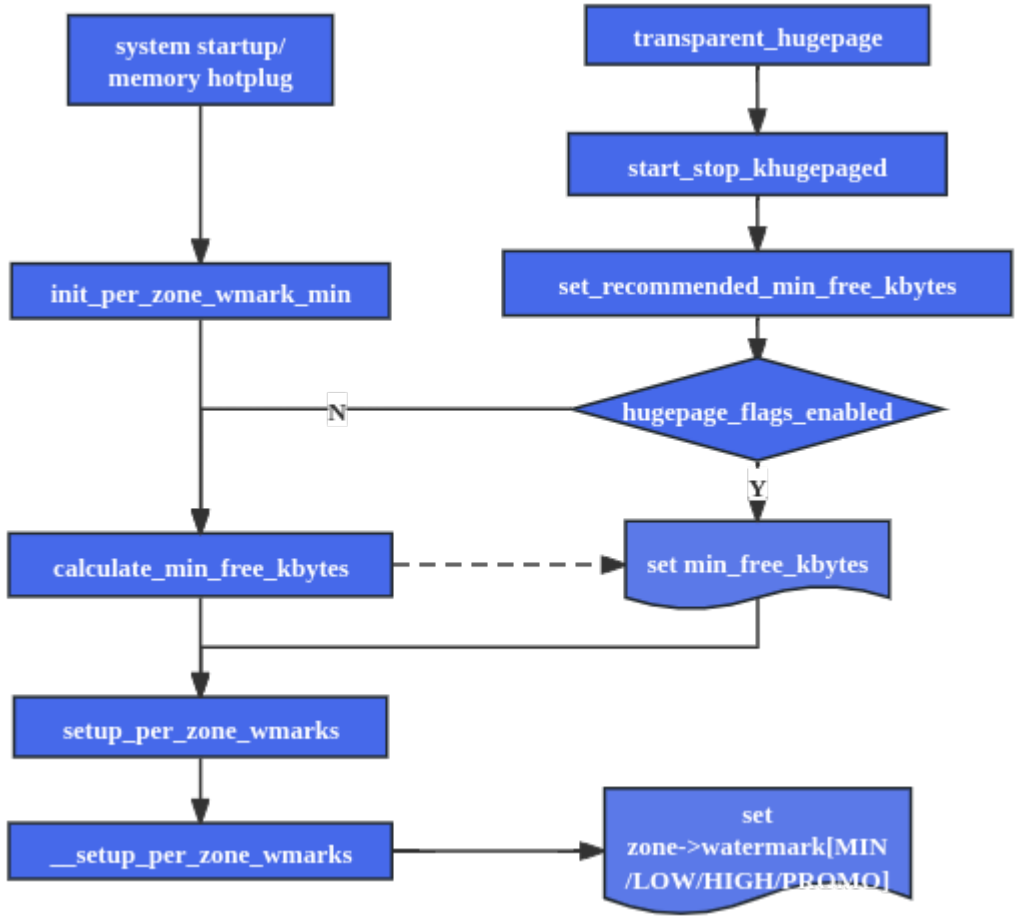
影响水位线的因素(服务器)

- 系统内存初始化
- 内存热插拔
- **透明大页功能**

最小水线设置不当，会造成内存浪费

- 由于透明大页的内存规整和不同迁移类型的衰退需求，**在ARM64构架的64K页配置环境下，推荐设定的最小水线值约17G。**
- 在222GB内存的机器，推荐值超过5%的可用内存，最小水位线被控制在11G(5%)，造成大量的内存浪费，挤压了业务场景的内存资源。

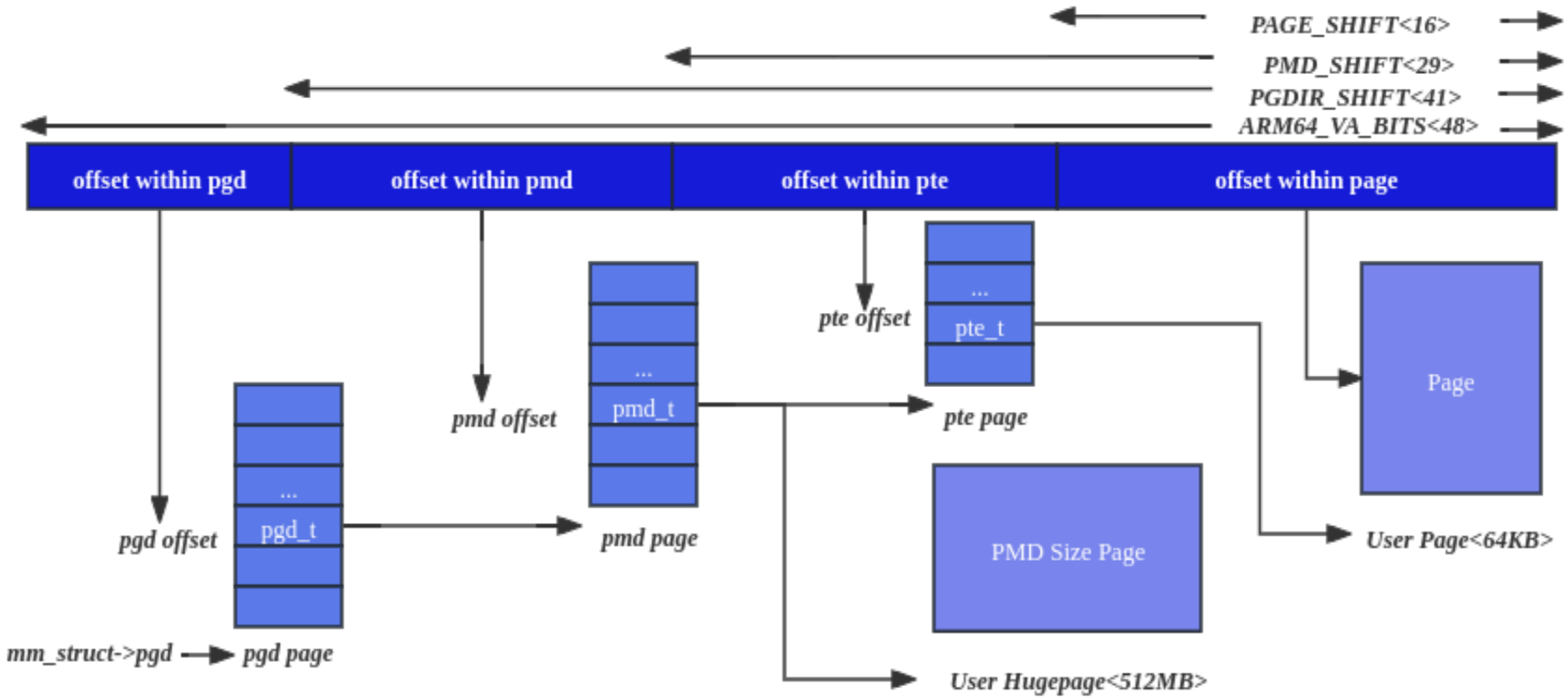
```
[root@localhost ~]# getconf PAGESIZE
65536
[root@localhost ~]# cat /proc/sys/vm/min_free_kbytes
11617600    约11GB，5%可用内存
[root@localhost ~]# free -h
               total        used        free      shared  b
Mem:           222Gi        44Gi       174Gi        98Mi
Swap:          4.0Gi          0B       4.0Gi
```



3.64K页配置的页表情况

最小水线值的算法

ARM64的64K页配置，在THP的规整和迁移类型的衰退逻辑需求下，推荐设定最小水线值约17G(8192 * 3(nr_zones) * (2 + 9) * 64k)，再与系统可用内存的5%进行比较取最小值.



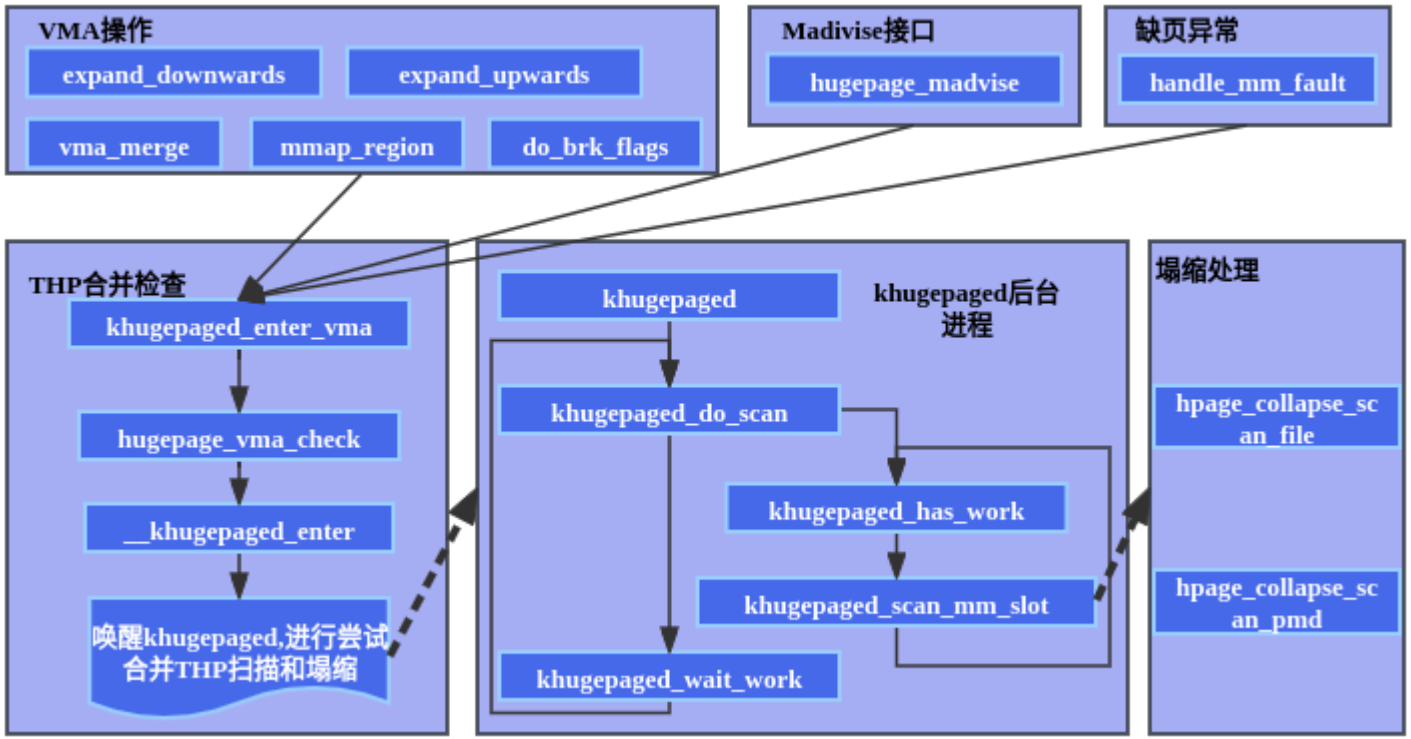
4.透明大页整体框架

关键流程

1、vma的分配，拓展及合并，madivise或缺页异常都会调用khugepaged_enter_vma，进而调用hugepage_vma_check

2、hugepage_vma_check会判断当前vma是否有合并为THP的可能；若有可能，则唤醒khugepagd线程

3、内核线程khugepaged通过khugepaged_has_work判断THP的合并尝试，并调用khugepaged_scan_mm_slot进行塌缩处理



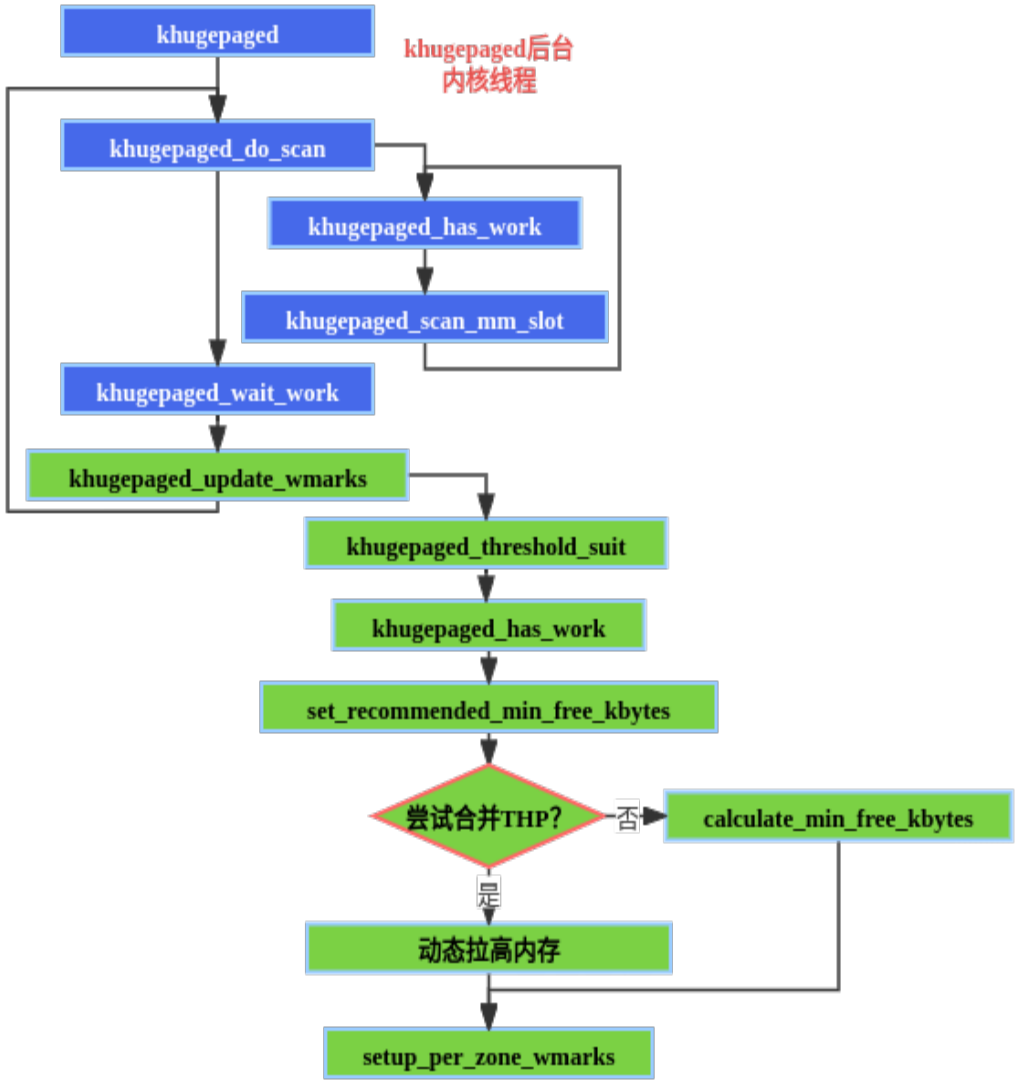
5. 64K页水位线优化实践

实践基础

- 默认的最小水位线过高，造成内存大量浪费，最小水位线和机器内存比率如下：6G/128GB，11G/222G，12.5G/256G。
- madivise默认配置下，有效区分业务逻辑使用透明大页的情况，不需要透明大页的业务中，保持较低的水位线值，释放内存资源提升业务的性能。
- khugepagd后台内核线程可以探知业务是否需要透明大页合并的情况。

优化实践

- 系统启动后保持低水位线设置，实践中保留1%，最小水位线和机器内存比率如下：1G/128GB，2G/222G，2.5G/256G。有效为业务释放可用内存比率为：5G/128G，9G/222G，10G/256G。
- khugepaged可以探测透明大页使用的基础上，在khugepaged后台内核线程的实现中追加khugepaged_update_wmarks的实现。
- khugepaged_threshold_suit判断当前是否为ARM64的64K页配置。
- khugepaged_has_work判断透明大页的合并尝试，两者都符合的情况下，拉高最小内存水位线，以支持透明大页的使用需求，达到自适应调节的情况。



5. 64K页水位线优化实践

优化效果

11G 预留

9G 释放为可用

2G 预留

系统启动后，运行MADV_HUGEPAGE应用(如qemu)，优化后的内核识别并拉高水位线对透明页进行支持，保持和改造前情况一致。结果如下：

```
[root@localhost ~]# cat /proc/meminfo | grep HugePage
AnonHugePages:   2097152 kB
ShmemHugePages:        0 kB
HugePages_Total:       0
HugePages_Free:        0
HugePages_Rsvd:        0
HugePages_Surp:        0
[root@localhost ~]# cat /proc/sys/vm/min_free_kbytes
11470976
```

2GB,4个THP

约11GB，5%可用内存

发行版默认选用madvise的配置，系统启动并运行非MADV_HUGEPAGE通常业务模型，其类似结果如下：

```
[root@localhost testscripts]# cat /proc/meminfo | grep HugePage
AnonHugePages:         0 kB
ShmemHugePages:        0 kB
HugePages_Total:       0
HugePages_Free:        0
HugePages_Rsvd:        0
HugePages_Surp:        0
[root@localhost testscripts]# cat /proc/sys/vm/min_free_kbytes
2323392
[root@localhost testscripts]# cat /sys/kernel/mm/transparent_hugepage/enabled
always [madvise] never
[root@localhost testscripts]#
```

约2GB，1%可用内存

6. 技术展望

- 1、64K页下大页根本性支持问题
- 2、水位线拉高，存在时延
- 3、水位线拉高，order是否满足透明大页分配
- 4、仅在有效捕获首次THP分配和尝试就进行水位线拉高
- 5、在always配置时，还是存在拉高水位线的情况可能
- 6、……

THANKS

THANKS

THANKS