嗯，用户提供了来自`/proc/buddyinfo`和`sar -r`的输出，显示内存使用率很高，并且存在内存碎片的问题。之前的对话中，用户已经了解到Jewel版本的Ceph没有`dump\_mempools`命令，需要通过其他方法分析内存占用。现在用户进一步提供了系统内存使用情况的详细数据，需要帮助分析这些数据以确定异常原因。

首先，我需要仔细查看`sar -r`的输出。用户提到物理内存占用非常高，而`sar`的数据显示`%memused`一直维持在98%以上，最高达到99.33%。这表明系统的物理内存几乎被完全占用，这可能引发OOM Killer或导致性能下降。

接下来，观察各个字段：

- `kbmemfree`：可用内存非常低，通常只有几十万KB，甚至有时低于100,000KB。

- `kbcached`：缓存的内存在几百万KB左右，例如5,860,968KB到8,246,720KB之间，这说明系统利用了一部分内存作为缓存，但即使如此，可用内存仍然极低。

- `kbactive`：活跃内存大约在96,000,000KB左右，占用了大部分物理内存，这可能表明有大量正在使用的内存页，未被释放。

- `kbinact`：非活跃内存大约在2,500,000KB到4,700,000KB之间，这部分可能包含可以被回收的内存，但数量相对较少。

结合之前的`/proc/buddyinfo`分析，Node 1的Normal Zone存在大量4KB的碎片，而更大的内存块几乎为零。这说明系统虽然总内存使用率高，但大部分是小块内存，无法满足大块内存的需求，导致即使有足够的内存也会出现分配失败的情况。

用户可能面临的问题是内存碎片化导致的高内存使用率，以及可能的性能问题。需要进一步确认是否有内存泄漏，或者是否是Ceph OSD进程的正常行为。由于用户提到是Ceph环境，OSD进程通常会占用大量内存用于缓存和管理对象存储，但持续接近100%的内存使用可能不正常，尤其是在没有足够缓存可释放的情况下。

此外，`kbcommit`的值大约是物理内存的3.66倍，这意味着系统已经提交了大量内存，存在较高的内存过载风险。这可能导致交换空间被频繁使用，但用户没有提供交换空间的使用情况，可能需要进一步检查。

接下来，需要建议用户如何进一步诊断和解决这个问题。可能的步骤包括：

1. 检查是否有内存泄漏的进程，特别是Ceph OSD进程。

2. 调整内核参数以减少内存碎片，如启用透明大页（THP），调整内存整理策略。

3. 考虑升级Ceph版本，以利用更新的内存管理功能。

4. 监控系统日志，查看是否有OOM事件或相关错误信息。

5. 如果可能，增加物理内存或优化Ceph配置以减少内存占用。

同时，需要考虑到用户使用的是较旧的内核版本（3.10.0-693.el7.x86\_64），这可能限制了某些内存管理功能的可用性，建议升级内核或系统版本以获得更好的内存管理支持。

最后，结合之前关于`osd\_memory\_target`的讨论，由于Jewel版本不支持该参数，用户可能需要通过系统级的内存限制或优化来缓解问题，比如使用cgroups限制OSD进程的内存使用，或者调整Ceph的配置参数以减少内存消耗。

从提供的 **sar -r** 输出和 **/proc/buddyinfo** 分析来看，该节点存在以下关键内存问题：

### **1. **内存使用率极高且持续满载****

**%memused 持续在 98% 以上**（最高达 99.33%），**kbmemfree** 长期低于 2GB（总内存约 131GB），表明物理内存几乎耗尽。

**kbcommit（承诺内存）高达 482GB**，远超物理内存（131GB），**%commit** 达 366%，说明存在严重的内存超配风险，可能频繁触发 OOM Killer 或依赖交换空间（若启用）。

### **2. **内存碎片问题显著****

**/proc/buddyinfo 显示 Node 1 的 Normal Zone 有大量 4KB 碎片**（448,966 个空闲块），但大块内存（如 4MB）几乎为 0。

**后果**：即使总空闲内存足够，内核可能因缺乏连续大块内存而无法分配，导致性能下降或异常。

### **3. **缓存回收效率低下****

**kbcached（页缓存）波动较大**（5.8GB ~ 8.2GB），但 **kbmemfree** 未因缓存释放显著增加，表明：

缓存可能被锁定（如 Ceph 的 BlueStore 缓存）。

内核因碎片问题无法有效回收缓存。

### **4. **活跃内存占比过高****

**kbactive（活跃内存）长期在 96GB 以上**，占总内存的 73%~75%，表明大部分内存被进程长期占用，疑似存在：

**内存泄漏**：如 Ceph OSD 未释放不再使用的内存。

**业务负载过重**：如高并发 RADOS 请求导致内存堆积。

### **关键异常时间点示例**

| **时间** | **kbmemfree** | **kbmemused** | **%memused** | **kbcached** | **kbactive** | **关键行为** |
| --- | --- | --- | --- | --- | --- | --- |
| **03:30:01 AM** | 1,025,192 | 130,460,392 | 99.22% | 8,021,832 | 96,856,600 | 缓存和活跃内存达峰值，碎片最严重 |
| **07:00:01 AM** | 876,020 | 130,609,564 | 99.33% | 8,139,680 | 97,172,816 | 内存使用率创纪录，风险最高 |
| **02:10:01 PM** | 4,253,828 | 127,231,756 | 96.76% | 4,635,276 | 93,745,092 | 内存短暂释放（可能触发过回收） |

### **解决建议**

#### **1. **紧急缓解措施****

**触发内存强制回收**：

bash

复制

echo 3 > /proc/sys/vm/drop\_caches # 释放页缓存、目录项和inode缓存（临时措施）

**重启高内存占用的 OSD 进程**：

bash

复制

systemctl restart ceph-osd@<osd\_id> # 释放该 OSD 占用的内存

#### **2. **诊断内存泄漏****

**检查 OSD 内存增长趋势**：

bash

复制

ceph daemon osd.<id> perf dump | grep memory # 观察 memory\_\* 指标

**通过 pmap 分析 OSD 进程内存分布**：

bash

复制

pmap -x $(pgrep ceph-osd) | sort -nk2 # 按内存段大小排序

#### **3. **优化内存配置****

**限制 OSD 内存使用（需内核支持）**：

bash

复制

cgroup memory limit 或 systemd 资源控制

**调整内核参数**：

bash

复制

sysctl -w vm.vfs\_cache\_pressure=200 # 提高缓存回收积极性sysctl -w vm.min\_free\_kbytes=2097152 # 保留 2GB 紧急内存

#### **4. **长期解决方案****

**升级内核和 Ceph 版本**：

升级到内核 4.4+ 以改善内存碎片管理。

升级到 Ceph Octopus（15.x）或更高版本，使用 **osd\_memory\_target** 和 **dump\_mempools** 精细管理内存。

**启用透明大页（THP）**：

bash

复制

echo "madvise" > /sys/kernel/mm/transparent\_hugepage/enabled

#### **5. **监控与告警****

**部署实时监控工具**：

bash

复制

ceph osd perf # OSD 性能指标

prometheus-ceph-exporter + Grafana # 可视化内存趋势

**设置内存阈值告警**：

bash

复制

alert: memory\_used > 95% for 5m # 通过 Prometheus 或 Nagios 实现

### **总结**

**直接原因**：内存碎片化 + 长期高负载导致物理内存耗尽，缓存无法有效回收。

**风险**：OOM Killer 可能随机终止进程（包括 Ceph OSD），导致集群不稳定。

**优先级**：立即重启部分 OSD 释放内存，随后升级内核和 Ceph 版本根治问题。