inspur Power Systems | IPS

CPU iowait 指标详解

陈炽卉

chenchihui@inspur.com

目　　录

1 iowait 概述

在常用的系统 CPU 统计工具如 topas、sar、vmstat、lparstat、iostat 等等输出中，通常都有一列 iowait。有一些客户常常以此为指标衡量 IO 性能，这种做法是否合理呢？我们首先来看看 iowait 的定义。

1.1 Iowait%的具体含义

如下是 iowait 的具体定义：

Reports the percentage of time the processor(s) were idle during which the system had outstanding disk/NFS I/O request(s).

也即**iowait 其实是一种特殊形式的 CPU 空闲。**特殊之处在于，在此 CPU 的等待队列上有线程在等待 IO 完成（我们称之为 pending IO 或者 outstanding IO）。

这是由 IO 的特点决定的，因为 IO 速度较慢，现代操作系统实现 IO 一般是通过异步中断来完成的：即提交 IO 请求，然后线程挂起进入等待队列；IO 完成后，再通过中断通知相关线程转到就绪队列，进行处理。

在相关任务线程提交完 IO 请求，到 IO 中断返回的过程中，此时 IO 主要由存储侧处理，主机侧 CPU 实际上处于空闲状态。如果此时有其他任务线程可调度，系统会直接调度其他线程，这样 CPU 就相应显示为 Usr 或 Sys；但是如果此时系统较空闲，无其他任务可以调度，CPU 就会显示为 iowait（实际上与 idle 无本质区别）。

注意 AIX 仅仅标记那些触发未完成 IO 任务的空闲 CPU 为 iowait 状态，不会牵连到系统中其他空闲的 CPU（这些 CPU 状态依然标记为 IDLE 空闲状态）。这样就有效减少了一部分 iowait 值虚高的情形：比如一个 4 颗物理 CPU 的系统，如果只有其中一颗物理 CPU 上有未完成 IO 请求，则 iowait 最高不会超过 25%.

1.2 iowait 的一些事实

基于此，可以看到：

1) %iowait 合理值取决于应用 IO 特点。

比如备份任务往往 iowait 较高；而 cache 命中率高、磁盘读写少的应用负载 iowait 一般不高。

2) 从上述说明可以看到，减少%iowait 的方法有两类：

一类是进一步缩减 IO 处理时间，比如采用 SSD 盘，或者甚至内存盘等技术；

另外一类是缩减 IO 处理过程中 CPU 的空闲时间，比如在系统中添加 CPU 密集型任务，可以使得%iowait 比例明显降低甚至为 0；

3) %iowait 比例与是否存在 IO 性能问题并无直接关系：

低 iowait 也不代表没有磁盘性能问题；参考第二点，完全可能在实际上 IO 服务时间非常长，但由于系统中同时存在 CPU 密集型任务掩盖了 iowait。

高 iowait 不一定代表有磁盘性能问题；因为系统可能比较空闲，而业务类型是 IO 密集型比如备份。

1.3 不同 SMT 级别对 iowait%取值的影响

如果系统启用了 SMT1，由于目前 iowait%的计算方法，SMT1 模式也会造成 iowait%被放大。

比如一个 8 核 SMT1 环境，如果其中一个逻辑 CPU 处于 iowait 状态： 因为 SMT1 情况下，该逻辑 CPU 独占一颗物理 CPU，那么相应的 iowait%就是 1/8 = 12.5%.

而如果是 8 核 SMT4 环境，其中一颗逻辑 CPU 处于 iowait 状态：因为 SMT4 环境下，4 个逻辑 CPU 对应一颗物理 CPU；而 4 个逻辑 CPU 获得物理 CPU 的比例取决于其负载，在负载等同（都没有实际负载）的情况下，处于 iowait 状态的逻辑 CPU 只能拿到 0.25 左右的物理 CPU 份额；这样 iowait%将会是 1/8 * 0.25 = 3.125%.

而如果是 SMT2 模式，同样情况下 iowait%取值应该在 6.25%左右。

2 总体建议

综上可以看到，iowait 取值由于度量算法因素，波动会比较大，与 IO 性能并没有直接的关联。我们需要结合更多 IO 指标比如 IO 服务时间来评估是否实际存在 IO 性能问题。

例如可以通过 iostat -DlT 1|grep -v "0.0 0.0 0.0"持续观察磁盘 IO 服务情况；如果读、写 avg serv 时间或排队 avg time 出现明显增长，则需要做相应调整，应对磁盘瓶颈。通过 iostat -a、fcstat 等观察光纤卡的统计数据，判断是否存在瓶颈需要调整。

3 附录：不同 SMT 级别时 iowait 取值演示

iostress

测试方法：

./iostress 4

启动 io 压力，然后运行 vmstat 查看 iowait 百分比。

3.1 SMT4 时 iowait%为 3

vmstat -Iwt 10 3

System configuration: lcpu=64 mem=114688MB

kthr			memory			page					faults		cpu				time
r	b	p	avm	fre	fi	fo	pi	po	fr	sr	in	sy	cs	us	sy	id	wa	hr mi se
1	3	0	1753324	27536828		1 27754	0	0	0		0 27707	628	50307	1	3	92	**3**	23:43:28
1	3	0	1753333	27536818		0 27838	0	0	0		0 27787	139	49908	1	3	92	**3**	23:43:38
1	4	0	1753334	27536817		0 27870	0	0	0		0 27732	155	49941	1	3	92	**3**	23:43:48

3.2 SMT2 时 iowait%为 6

smtctl -t 2
smtctl: SMT is now enabled. It will persist across reboots if
you run the bosboot command before the next reboot.
vmstat -Iwt 10 3

System configuration: lcpu=32 mem=114688MB

kthr			memory			page					faults		cpu				time

```
----------- -------------------- ------------------------------------ ------------------ ----------- --------
   r     b     p        avm         fre      fi      fo      pi      po      fr       sr      in       sy      cs us sy id wa hr mi se
   2     3     0    1748582    27541480       0   27716       0       0       0        0   27652      182   49953  2  4 88  6 23:44:17
   2     4     0    1746031    27544031       0   27716       0       0       0        0   27657      137   49624  2  4 88  6 23:44:27
   1     3     0    1743471    27546591       0   27679       0       0       0        0   27540      142   49281  2  4 88  6 23:44:37
```

3.3 SMT1 时 iowait%为 12

```
# smtctl -t 1
smtctl: SMT is now disabled. It will persist across reboots if
        you run the bosboot command before the next reboot.
# vmstat -Iwt 10 3

System configuration: lcpu=16 mem=114688MB

    kthr                memory                              page                              faults                cpu          time
----------- -------------------- ------------------------------------ ------------------ ----------- --------
   r     b     p        avm         fre      fi      fo      pi      po      fr       sr      in       sy      cs us sy id wa hr mi se
   1     3     0    1736147    27553897       0   27464       0       0       0        0   27433      478   48825  0  8 79 12 23:45:06
   1     3     0    1733596    27556447       0   27611       0       0       0        0   27509      145   48887  0  8 80 12 23:45:16
   1     3     0    1731036    27559007       0   27674       0       0       0        0   27601      138   49022  0  8 79 13 23:45:26
```

说明：在绝大多数应用场景下，高级别的 SMT 整体性能表现更佳，尤其对以整体吞吐率 TPS 为主要度量的客户应用而言。SMT8 常常能达到 SMT1 的 2 倍左右性能。在少部分场景下，主要是锁冲突比较严重的情况下，SMT1 通过限制并发数，可能获得一定性能优势 ;但这实际上是治标不治本的方法 ,并不能保证凑效。一般而言 ,需要从根本上定位造成锁冲突的原因 ;解决锁冲突后 ,SMT8/SMT4 性能仍然优于 SMT1.

THANKS