**如何利用多核CPU来加速你的Linux命令 — awk, sed, bzip2, grep, wc等**

你是否曾经有过要计算一个非常大的数据(几百GB)的需求？或在里面搜索，或其它操作——一些无法并行的操作。数据专家们，我是在对你们说。你可能有一个4核或更多核的CPU，但我们合适的工具，例如 **grep**, **bzip2**, **wc**, **awk**, **sed**等等，都是单线程的，只能使用一个CPU内核。

借用卡通人物Cartman的话，“如何我能使用这些内核”?

要想让Linux命令使用所有的CPU内核，我们需要用到[GNU Parallel](https://www.gnu.org/software/parallel/)命令，它让我们所有的CPU内核在单机内做神奇的map-reduce操作，当然，这还要借助很少用到的**–pipes**参数(也叫做**–spreadstdin**)。这样，你的负载就会平均分配到各CPU上，真的。

**BZIP2**

bzip2是比gzip更好的压缩工具，但它很慢！别折腾了，我们有办法解决这问题。

以前的做法：

cat bigfile.bin | bzip2 --best > compressedfile.bz2

现在这样：

cat bigfile.bin | parallel --pipe --recend '' -k bzip2 --best > compressedfile.bz2

尤其是针对bzip2，GNU parallel在多核CPU上是超级的快。你一不留神，它就执行完成了。

**GREP**

如果你有一个非常大的文本文件，以前你可能会这样：

grep pattern bigfile.txt

现在你可以这样：

cat bigfile.txt | parallel  --pipe grep 'pattern'

或者这样：

cat bigfile.txt | parallel --block 10M --pipe grep 'pattern'

这第二种用法使用了**–block 10M**参数，这是说每个内核处理1千万行——你可以用这个参数来调整每个CUP内核处理多少行数据。

**AWK**

下面是一个用awk命令计算一个非常大的数据文件的例子。

常规用法：

cat rands20M.txt | awk '{s+=$1} END {print s}'

现在这样：

cat rands20M.txt | parallel --pipe awk \'{s+=\$1} END {print s}\' | awk '{s+=$1} END {print s}'

这个有点复杂：parallel命令中的**–pipe**参数将cat输出分成多个块分派给awk调用，形成了很多子计算操作。这些子计算经过第二个管道进入了同一个awk命令，从而输出最终结果。第一个awk有三个反斜杠，这是GNU parallel调用awk的需要。

**WC**

想要最快的速度计算一个文件的行数吗？

传统做法：

wc -l bigfile.txt

现在你应该这样：

cat bigfile.txt | parallel  --pipe wc -l | awk '{s+=$1} END {print s}'

非常的巧妙，先使用parallel命令‘mapping’出大量的wc -l调用，形成子计算，最后通过管道发送给awk进行汇总。

**SED**

想在一个巨大的文件里使用sed命令做大量的替换操作吗？

常规做法：

sed s^old^new^g bigfile.txt

现在你可以：

cat bigfile.txt | parallel --pipe sed s^old^new^g

…然后你可以使用管道把输出存储到指定的文件里。