Exploratory Analysis of PVFS performance measures

Louis-Claude Canon

January 17, 2012

In the following experiments, one PVFS 2.8.2 server is deployed with default configuration in user space and one client is accessing it. By using the $x/y$ notation, where $x$ is the number of clients and $y$ is the number of servers, it is denoted by 1/1. The client alternatively write a file and then read it. The parapluie cluster of Grid5000 is used.

The latency between two nodes of the cluster is around 0.075 ms (obtained with ping over 10 runs). The peak bandwidth is around 942 Mbits/s (obtained with iperf, which uses IU system, over 2 or 3 runs).

1 Measurement scripts

A collection of scripts for launching write(s) and/or read(s) with variable strip sizes.

- exponential randomness is achieved in bash with /dev/random, od and bc
- clients synchronization before each measure is done with a simple TCP server in python and bash connections through file descriptors
- a file is created in /tmp with dd, copied on the PVFS partition and read again (initial and final file checksums are compared)

2 Effect of message sizes

The strip size has first been put to the maximum value, 1 GB, while the log of transfer sizes are uniformly distributed between 1 B and 1 GB.

```r
> data_size <- read.table("size_w.txt")
> dim(data_size)
[1] 998 2
> names(data_size) <- c("size", "time")
> plot(data_size, log = "xy", cex = 0.1)
```

It seems that their is several distinct areas (see Figure 1):

**1 B to 7 kB** constant durations with a slight increase at the end (the median duration is six times the latency)

```r
> segments(x0 = 1, x1 = 1e+09, y0 = 0.001)
> max(data_size[data_size[, 2] < 0.001, 1])
[1] 6803
> min(data_size[data_size[, 2] >= 0.001, 1])
```
> duration_small <- median(data_size[data_size[, 2] < 0.001, 2])
> format(duration_small, scientific = TRUE)
> duration_small/7.5e-05

7 kB to 300 kB several modes at first, then a linear increase until the end

> segments(x0 = 3e+05, y0 = 0.001, y1 = 10)

300 kB to 1 GB linear progression with a slight curve at first

> large <- data_size[data_size[, 1] > 1e+08, ]
> dim(large)
> bandwidth_large <- median(large[, 1]/large[, 2])
> format(bandwidth_large, scientific = TRUE)
> bandwidth_large * 8/9.42e+08

2.1 Small messages

> fit_small <- lm(log10(time) ~ log10(size), data = data_size[data_size[, + 2] < 0.001, ])
> fit_verry_small <- lm(log10(time) ~ log10(size), data = data_size[data_size[, + 1] < 1000, ])
> fit_less_small <- lm(log10(time) ~ log10(size), data = data_size[data_size[, + 1] > 1000 & data_size[, 2] < 0.001, ])
> summary(fit_small)

Call:
  lm(formula = log10(time) ~ log10(size), data = data_size[data_size[, 2] < 0.001, ])

Residuals:
     Min      1Q    Median      3Q     Max
-0.085813 -0.017903 -0.000072  0.019278  0.166301

Coefficients:
                     Estimate Std. Error    t value  Pr(>|t|)
(Intercept)     -3.3613590   0.0029700   -1131.90  < 2e-16 ***
log10(size)     0.0205200   0.0013881     14.780  < 2e-16 ***
---
Figure 1: Durations to write a file of various sizes
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.03284 on 408 degrees of freedom
Multiple R-squared: 0.3487,  Adjusted R-squared: 0.3472
F-statistic: 218.5 on 1 and 408 DF,  p-value: < 2.2e-16

> summary(fit_verry_small)

Call:
  lm(formula = log10(time) ~ log10(size), data = data_size[data_size[, 1] < 1000, ])

Residuals:
     Min       1Q   Median       3Q      Max
-0.076131 -0.008111  0.002171  0.013583  0.164841

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -3.344941  0.002611 -1281.04  <2e-16 ***
log10(size)  0.003957  0.001624   2.437  0.0153 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.02588 on 318 degrees of freedom
Multiple R-squared: 0.01834,  Adjusted R-squared: 0.01525
F-statistic: 5.941 on 1 and 318 DF,  p-value: 0.01534

> summary(fit_less_small)

Call:
  lm(formula = log10(time) ~ log10(size), data = data_size[data_size[, 1] > 1000 & data_size[, 2] < 0.001, ])

Residuals:
     Min       1Q   Median       3Q      Max
-0.063718 -0.017966  0.000717  0.010850  0.133776

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -3.57267   0.04246 -84.139  < 2e-16 ***
log10(size)  0.08862   0.01244   7.124 2.75e-10 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.03039 on 88 degrees of freedom
Multiple R-squared: 0.3657,  Adjusted R-squared: 0.3585
F-statistic: 50.74 on 1 and 88 DF,  p-value: 2.745e-10

> plot(data_size, log = "xy", cex = 0.1, xlim = c(1, 7000), ylim = exp(extendrange(log(data_size[data_size[, 1] < 7000, 2]))))
> abline(fit_small)
> abline(fit_verry_small)
> abline(fit_less_small)
For sizes lower than 1 kB, a constant behavior seems relevant: R-squared is close to zero, p-value is low and the coefficient is also close to zero. For larger sizes, there is a slight increase and the R-squared (even for this part only) is not exceptional.

### 2.2 Large messages

```r
> fit_large <- lm(log10(time) ~ log10(size), data = data_size[data_size[, + 1] > 3e+05, ])
> fit_verry_large <- lm(log10(time) ~ log10(size), data = data_size[data_size[, + 1] > 1e+08, ])
> fit_most_large <- lm(log10(time) ~ log10(size), data = data_size[data_size[, + 1] > 3e+06, ])
> summary(fit_large)

Call:
  lm(formula = log10(time) ~ log10(size), data = data_size[data_size[, 1] > 3e+05, ])

Residuals:
  Min  1Q Median  3Q  Max
-0.06372 -0.03103 -0.01033 0.02150 0.36674
```
Coefficients:

| Estimate | Std. Error | t value | Pr(>|t|) |
|----------|------------|---------|----------|
| (Intercept) | -7.514208 | 0.018502 | -406.1 <2e-16 *** |
| log10(size) | 0.937133 | 0.002531 | 370.2 <2e-16 *** |

---

Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.05158 on 406 degrees of freedom
Multiple R-squared: 0.997, Adjusted R-squared: 0.997
F-statistic: 1.37e+05 on 1 and 406 DF, p-value: < 2.2e-16

> summary(fit_verry_large)

Call:
```
  lm(formula = log10(time) ~ log10(size), data = data_size[data_size[, 1] > 1e+08, ])
```

Residuals:

<table>
<thead>
<tr>
<th>Min</th>
<th>1Q</th>
<th>Median</th>
<th>3Q</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>-0.0063403</td>
<td>-0.0039013</td>
<td>-0.0021250</td>
<td>0.0003411</td>
<td>0.0298312</td>
</tr>
</tbody>
</table>

Coefficients:

| Estimate | Std. Error | t value | Pr(>|t|) |
|----------|------------|---------|----------|
| (Intercept) | -7.952804 | 0.017680 | -449.8 <2e-16 *** |
| log10(size) | 0.991228 | 0.002078 | 477.1 <2e-16 *** |

---

Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.006478 on 108 degrees of freedom
Multiple R-squared: 0.9995, Adjusted R-squared: 0.9995
F-statistic: 2.276e+05 on 1 and 108 DF, p-value: < 2.2e-16

> summary(fit_most_large)

Call:
```
  lm(formula = log10(time) ~ log10(size), data = data_size[data_size[, 1] > 3e+06, ])
```

Residuals:

<table>
<thead>
<tr>
<th>Min</th>
<th>1Q</th>
<th>Median</th>
<th>3Q</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>-0.021929</td>
<td>-0.014582</td>
<td>-0.005146</td>
<td>0.008405</td>
<td>0.132392</td>
</tr>
</tbody>
</table>

Coefficients:

| Estimate | Std. Error | t value | Pr(>|t|) |
|----------|------------|---------|----------|
| (Intercept) | -7.774595 | 0.013296 | -584.7 <2e-16 *** |
| log10(size) | 0.969892 | 0.001715 | 565.7 <2e-16 *** |

---

Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.02138 on 294 degrees of freedom
Multiple R-squared: 0.9991, Adjusted R-squared: 0.9991
F-statistic: 3.2e+05 on 1 and 294 DF, p-value: < 2.2e-16
Perfect linear model between 100 MB and 1 GB. Not bad even for sizes greater than 300 kB and the fit quality improves when restricting the size to higher sizes.

3 Effect of strip sizes

The transfer size is set to 100 MB, while the log of strip size are uniformly distributed between 1 B and 100 MB. 1000 measures are done.

> data_strip <- read.table("tmp2")
> dim(data_strip)
[1] 1000 2
> names(data_strip) <- c("strip size", "time")
> plot(data_strip, log = "xy", cex = 0.1)
Figure 2: Durations to write a file with various strip sizes
It seems that the strip size has no effect with one server. However, the variation seems to be larger than on Figure 1.

> `plot.ecdf(data_strip[, 2], cex = 0.1)`

The noise represents 20% of the data, which could be consistent with previous data (in any case, the log scale on Figure 1 does not allows an accurate comparison).

> `plot(data_size[9e+07 < data_size[, 1] & data_size[, 1] < 1.1e+08, + ])`
Now, everything seems consistent.

4 Conclusion

There is clearly several distinct behaviors depending on the message size: a constant duration for small messages (6 times the latency) and a peak bandwidth for large messages (90% of the maximum/theoretical bandwidth). In 1/1 configuration, it should be mostly due to network effects. Related work could help to understand this (investigate "slow start" and literature from Arnaud/Pedro). Additional impacts coming from PVFS could then be characterized (the constant durations and the peak bandwidth for example).

5 Future work

Methodological improvements:
- regression on exponentially distributed data
- non-linear regression for large messages
- regression in the median case (the most precise regression of half the points)
- dealing with non-Gaussian noise (in our case, the noise is positive)
XP code improvements:

- clean XP parameter: include strip size, type of operations, size and time in output and do it in one echo command (possibly with no log info)
- check “TroveMethod directio” setting impact (and more generally, test other tuning parameters)
- log info in the case of multiple clients (and output sequentially with cat on the master) to avoid inconsistence between concurrent echo
- build a true PVFS system (not in user space)

Future settings:

- 1/2 (effect of message size and strip size for all combinations)
- 2/1 (effect of message size and strip size especially for large messages, it may present some sync problems to send small message at the same time)
- 2/2
- $x/y$ with large messages and varying strip size

Complete list of possible settings:

- data size (1 B to 1 GB)
- strip size (100 B to 10 MB)
- number of servers on which a file is distributed
- storage system (PVFS, Blobseer, HDFS, Lustre, PanFS, dCache)
- network bandwidth
- cluster (4 types at Rennes)
- number of clients (1 at 1000)
- number of servers (1 at 100)
- distribution algorithm (simple or basic)
- operation type (read or write)
- replication rate (1 to 10)
- disk synchronization
- cache / RAM size on the disk
- disk bandwidth
- file system on the servers