Comparison between single threaded and multithreaded compression using input as Calgary Corpus Set

The comparison is to explore the different options in QAT and get acquainted with all the cases. In the below experiment, I have compared the compression of Calgary corpus with threads and without them. I have used two threads (pthreads) paralleling compressing the different files.

IIT KGP Machine and QAT hardware details:

Intel(R) Xeon(R) CPU E3-1225 v5 @ 3.30GHz , 4 cores .

```
[root@localhost pranesh]# service qat_service status

There is 1 acceleration device(s) in the system:

 icp_dev0 - type=dh895xcc, inst_id=0, node_id=0,  bdf=03:00:0, #accel=6, #engines=12,
state=up

[root@localhost pranesh]#
```

Experiment details:

I have used 18 file Calgary Corpus set. Downloaded from here (http://www.data-compression.info/files/corpora/largecalgarycorpus.zip)

Single Threaded, in a loop (multiple invocation of executable):
The Calgary corpus set was compressed, by compressing each file sequentially (single thread), the total compression time took to compress all files, taken one by one, by invoking the compression executable in a loop (via a Shell script), is **38063 ms**.

Single Threaded, no loop (single invocation of executable):
In this case single thread was allowed to handle all 18 files back to back. This has only single invocation of executable. The time took to compress all set is **10012 ms.**

With Parallelism:

The number of Dc Instances that are available in my machine is 2. I got it through this API.

```
status = cpaDcGetNumInstances(&numInstances);
```

I found out through experiment that I can't use one instance to compress two files in parallel. Following were the config that I used.

```
{ stateful compression, deflate , static huffman coding, compression and
decompression combined, compression level=9 }
```

I haven't tried the same with stateless.

So for the stateful, I can use maximum parallelism is with two threads compressing independently.

The performance in time, was much reduced based upon the combination of input files. Let's say if the two files named A and B was having compression time without threads as x and y then, with threads, it's **max (x,y)** .

So the maximum time reduction I can get is when the two threads have equal load.

Based on this, I have tried with Calgary corpus with 2 threads. The total set of files are divided into 2 sets.

```
Set 1:
{ "paper5" , "paper4" , "obj1" , "paper6" , "paper3" , "progp", "paper1" , "progl" ,
"paper2" , "trans" ,"geo" ,"bib" , "book1", "progc" }   => 1.433 MiB

Set 2:
{ "obj2" , "news" , "pic" , "book2" } => 1.66 MiB
```

Each set is allotted to particular thread. The whole Calgary Corpus (~ 3.2M) was compressed in **6007 ms**.

I/P File name	Dual thread	Single thread	Comment
Calgary corpus set	6007ms	10012ms	1.66 times reduced with parallelism.