Exercise 25.1 Comparing I/O Schedulers

We provide a script which is to be used to compare I/O schedulers:

```
#!/bin/bash

#/*
# * The code herein is: Copyright the Linux Foundation, 2014
# * Author J. Cooperstein
# *
# * This Copyright is retained for the purpose of protecting free
# * redistribution of source.
# *
# * This code is distributed under Version 2 of the GNU General Public
# * License, which you should have received with the source.
# *
# */

NMAX=8
NMEGS=100
[[ -n $1 ]] && NMAX=$1
[[ -n $2 ]] && NMEGS=$2

echo Doing: $NMAX parallel read/writes on: $NMEGS MB size files

TIMEFORMAT="%R   %U   %S"

#################################################################
# simple test of parallel reads
do_read_test(){
    for n in $(seq 1 $NMAX) ; do
        cat file$n > /dev/null &
    done
# wait for previous jobs to finish
    wait
}

# simple test of parallel writes
do_write_test(){
    for n in $(seq 1 $NMAX) ; do
        [[ -f fileout$n ]] && rm -f fileout$n
        (cp file1 fileout$n && sync) &
    done
# wait for previous jobs to finish
    wait
}

# create some files for reading, ok if they are the same
create_input_files(){
    [[ -f file1 ]] || dd if=/dev/urandom of=file1 bs=1M count=$NMEGS
    for n in $(seq 1 $NMAX) ; do
        [[ -f file$n ]] || cp file1 file$n
    done
}

echo -e "\ncreating as needed random input files"
create_input_files
```

```
#############################################################
# begin the actual work

# do parallel read test
echo -e "\ndoing timings of parallel reads\n"
echo -e " REAL    USER    SYS\n"
for iosched in noop deadline cfq ; do
    echo testing IOSCHED = $iosched
    echo $iosched > /sys/block/sda/queue/scheduler
    cat /sys/block/sda/queue/scheduler
#    echo -e "\nclearing the memory caches\n"
    echo 3 > /proc/sys/vm/drop_caches
    time do_read_test
done
#############################################################
# do parallel write test
echo -e "\ndoing timings of parallel writes\n"
echo -e " REAL    USER    SYS\n"
for iosched in noop deadline cfq ; do
    echo testing IOSCHED = $iosched
    echo $iosched > /sys/block/sda/queue/scheduler
    cat /sys/block/sda/queue/scheduler
    time  do_write_test
done
#############################################################
```

If you are taking the online self-paced version of this course, the script is available for download from your **Lab** screen.

Remember to make it executable by doing: by doing:

```
$ chmod +x ioscript.sh
```

The following explains how the script was written and how to use it.

The script should:

- Cycle through the available I/O schedulers on a hard disk while doing a configurable number of parallel reads and writes of files of a configurable size.
- Test reads and writes as separate steps.
- When testing reads make sure you're actually reading from disk and not from cached pages of memory; you can flush out the cache by doing:

  ```
  $ echo 3 > /proc/sys/vm/drop_caches
  ```

  before doing the reads. You can **cat** into `/dev/null` to avoid writing to disk.
- Make sure all reads are complete before obtaining timing information; this can be done by issuing a **wait** command under the shell.
- Test writes by simply copying a file (which will be in cached memory after the first read) multiple times simultaneously. To make sure you wait for all writes to complete before you get timing information you can issue a **sync** call.

The provided script takes two arguments. The first is the number of simultaneous reads and writes to perform. The second is the size (in MB) of each file.

This script must be run as root as it echoes values into the `/proc` and `/sys` directory trees.

Compare the results you obtain using different I/O schedulers.

For additional exploring you might try changing some of the tunable parameters and see how results vary.