DiskSim Project
Week Three

Jiafei Song
School of Information and Science Technology
songjf@shanghaitech.edu.cn
ShanghaiTech University

Yuan Yuan
School of Information and Science Technology
yuanyuan1@shanghaitech.edu.cn
ShanghaiTech University

Yusen Lin
School of Information and Science Technology
linys@shanghaitech.edu.cn
ShanghaiTech University

***Abstract*—This project is to have a better understanding of storage systems with the tool of DiskSim,especially disk storage systems. And the whole project will be divided into four parts to complete.**

I. *Overview*

DiskSim is an efficient, accurate, highly-configurable disk system simulator originally developed at the University of Michigan and enhanced at CMU to support research into various aspects of storage subsystem architecture.

A. Background

It is written in C and requires no special system software (just basic POSIX interfaces). DiskSim includes modules for most secondary storage components of interest, including device drivers, buses, controllers, adapters, and disk drives. DiskSim also includes support for a number of externally-provided trace formats and internally-generated synthetic workloads, and includes hooks for inclusion in a larger scale system-level simulator.

B. Usage

It has been used in a variety of published studies (and several unpublished studies) to understand modern storage subsystem, to understand how storage performance relates to overall system, and to evaluate new storage subsystem.

C. Structure

There are two main executable file in the whole system . One is Disksim.c and another is Syssim_driver.c.

1) **Disksim***:* There are five parameters for this function, including the: parfile,outfile,tracetype,tracefile,synthgen.
Parfile:aka parameter file. DiskSim can be configured via the parameter file to model a wide variety of storage subsystems. It uses libparam to input the parameter file.
Outfile:The output of simulation. It can be modified by the parameter file.
Tracetype:The type of trace file. ASCII, HPL, HPL2, DEC, RAW, VALIDATE, EMCSYMM, DEFAULT(ASCII) and so on. You can also add new type.
Tracefile:The format of the tracefile is request arrival time, device number, block number, request size, request flags. Zero means generating data by parameter file.If it's zero, synthgen must be larger then zero.
synthgen:The value of synthgen means how many generators used to get trace, one generator represents a process to generator input.

2) **Syssim_driver***:*

disksim provides disksim_interface.c to make disksim as subsystem to get result by external requests . And this function is to use this interface to simulate the disk.

II. *Case Study*

A. Configuration and Installation

We run DiskSim-4.0 on 32-bit Ubuntu 14.04 LTS.
1) Download disksim-4.0.tar.gz from
http://www.pdl.cmu.edu/DiskSim/
2) Run the commands $tarxfzdisksim - 4.0.tar.gz$ and $cddisksim - 4.0$ in the terminal.
3) In $memsmodel/Makefile$,
repalce
ems_seektest: mems_seektest.o libmems_internals.a
$(CC) -o $@ mems_seektest.o $(LDFLAGS) $(CFLAGS) -lmems_internals
with
ems_seektest: mems_seektest.o libmems_internals.a
$(CC) -o $@ mems_seektest.o $(CFLAGS) -lmems_internals $(LDFLAGS)
4) In $src/Makefile$,
repalce
LDFLAGS = -lm -L. -ldisksim
$(DISKMODEL_LDFLAGS) $(MEMSMODEL_LDFLAGS)
$(LIBPARAM_LDFLAGS) $(LIBDDBG_LDFLAGS)
with
LDFLAGS = -L. -ldisksim
$(DISKMODEL_LDFLAGS) $(MEMSMODEL_LDFLAGS)
$(LIBPARAM_LDFLAGS) $(LIBDDBG_LDFLAGS) -lm
5) Run the command $make$ in the terminal.
6) Run the command $cdvalid; ./runvalid$ in the terminal.

B. Parameter initialize

In the disksim_logorg org0:
Addressing Mode = Array

Distribution scheme = Striped
Redundancy scheme = Striped
devices = [disk0..disk8]
In the disksim_synthio Synthio:
change distribution to normal

C. Run

In the valid folder, run:

../src/disksim synthraid0.parv synthraid0.outv ascii 0 1
grep "IOdriver Response time average" synthraid0.outv

D. Result:

IOdriver Response time average : 19.438945

III. *Raid5*

A. Parameter initialize

#component instantiation
instantiate [statfoo] as Stats
instantiate [ctlr0 .. ctlr4] as CTLR0
instantiate [bus0] as BUS0
instantiate [disk0 .. disk4] as HP_C3323A
instantiate [driver0] as DRIVER0
instantiate [bus1 .. bus5] as BUS1

disksim_logorg org0{
Addressing mode = Array
Distribution scheme = Striped
Redundancy scheme = Parity( )rotated
devices = [disk0 .. disk4]
...
}

disksim_synthgen{ # generator0
Storage capacity per device = 8224032
devices = [org0],
Blocking factor = 8,
Probability of sequential access = 0.2,
Probability of local access = 0.4,
Probability of read access = 0.66,
Probability of time-critical request = 0.1,
Probability of time-limited request = 0.3,
Time-limited think times = [normal, 30.0, 100.0],
General inter-arrival times = [exponential, 0.0, 10.0],
Sequential inter-arrival times = [exponential, 0.0, 10.0],
Local inter-arrival times = [exponential, 0.0, 10.0],
Local distances = [normal, 0.0, 40000.0],
Sizes = [exponential, 0.0, 8.0]
} # end of generator 0

PS. In this part, we particularly use 5 disks to do the simulation.

B. Result

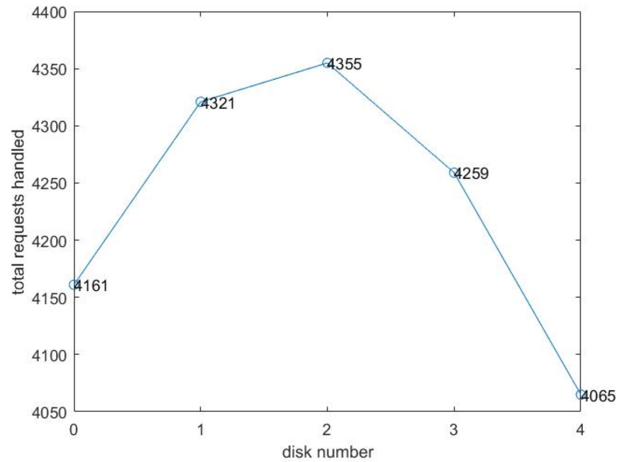

Fig. 1. disk id vs total requests handled

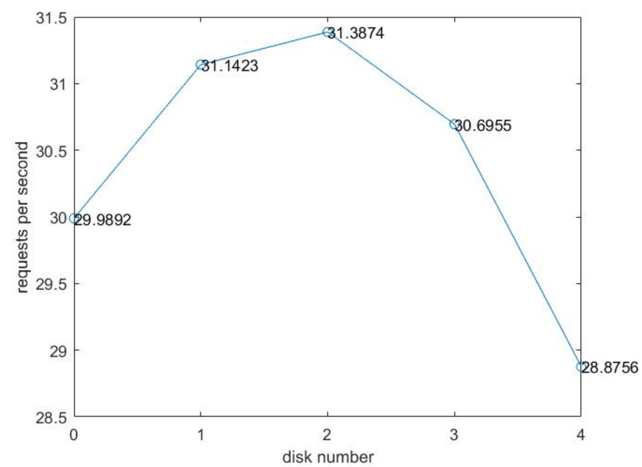

Fig. 2. disk id vs requests per second

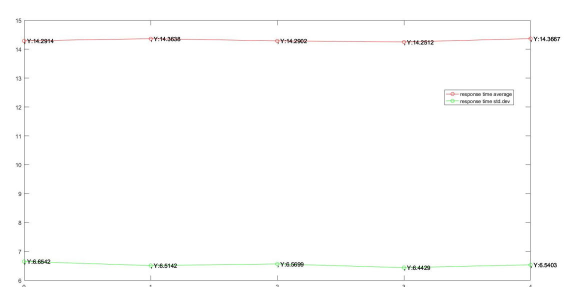

Fig. 3. disk id vs response time

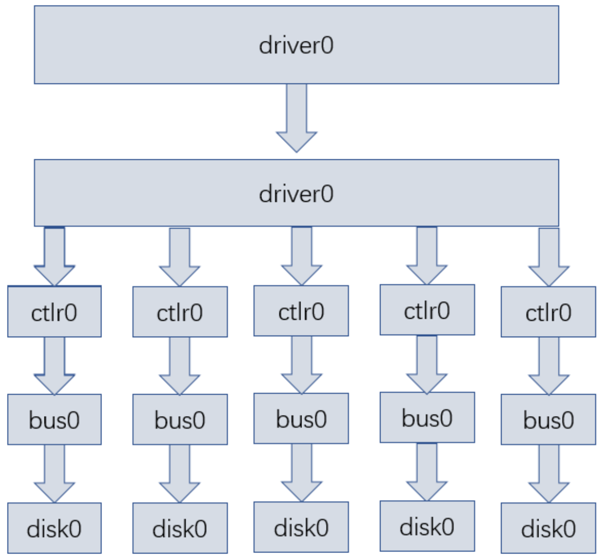

Fig. 4. topology of raid5-5

IV. *PARAID5*

A. Parameter initialize

There are gears: one with 4 disks (gear 1) and the other with 5 disks (gear 2).Each gear has its own parameter file.The parameter file of the one with 5 disks is the same with that of the 5-disk RAID-5.
We made the following change in that of 4-disk one.
1) [ctlr0 .. ctlr4] —-> [ctlr0 .. ctlr3]
2) [disk0 .. disk4] —-> [disk0 .. disk3]
3) Storage capacity per device = 8224032 —-> Storage capacity per device = 6168024

B. Implementation

We call the parameter file of gear 1 ”paraid5_1.parv” and out file ”paraid5_1.outv”. We call the parameter file of gear 2 ”paraid5_2.parv” and out file ”paraid5_2.outv”. We write a python code to transfer from one gear to the other. To run command line in python, we use the os package.
For example,

```
import os
os.system("ls -l")
```

will show the detailed file information in the current route, which just like call ”ls -l” in the terminal.
By running

```
os.system("../src/disksim paraid5_1.parv paraid5_1.outv
    ascii 0 1")
```

we get paraid5_1.outv. The srcipt will open the outv file and read it line by line. It search for the line with string ”Completely idle time:” at the location of [8:29]. For example, the line may be ”Disk #1 Completely idle time: 15059.014877 0.470399”. We split the line and get the last part as the idle time ratio. We regard $1 - idle\ time\ ratio$ as utilization time ratio. U denotes the average of the utilization time ratios of the disks and S denotes the stand deviation of utilization time. What we do in gear 2 is similar.
When in gear 1, if $U + S > T$, shift up. When in gear 2, if $U + S < T$, shift down.
The python code runs the disksim 8 times to simulate 8 time windows. After each time window, it checks if $U + S > T$ or $U + S < T$. Then it will shift up or shift down if needed.

C. Result

In our test,the result is the following.

gear	disks	U	S	U+S
1	4	0.5094005	0.0463932240568	0.555793724057
2	5	0.4321564	0.0251287172503	0.45728511725

TABLE I
U AND S FOR EACH GEAR

We set threshold T to 0.8.

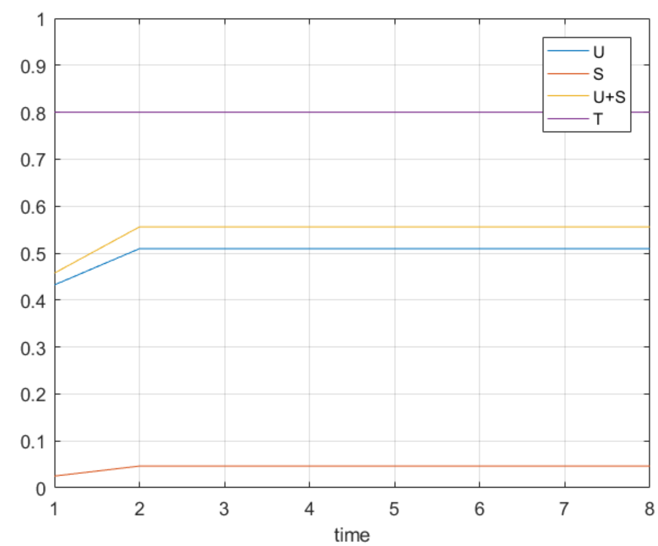

Fig. 5. U,S,U+S,T vs time

At first, it is in gear 2. Since $U + S < T$, it shifts down and then levels out at gear 1.

D. Theory Analysis

Gearing is a process of drving a car. If we want to change the drving mode, we need gear. Gearing of the PARAID is similar to gearing of the car.

There are three modes of disks: idle, standby and active. Certainly,the active mode consumes more energy than other modes. However, switching mode also consumes lots of energy. To save energy, we need keep balance among the modes.

An idea is that we let fewer disks respond to the I/O instructions when the total workload is not heavy and let more disks respond to the I/O instructions when the total workload is heavy. Hence the workload is asymmetric. PARAID works in this way.

E. Difficulties

It's very difficult and we just made a naive version. What we can do is reading the DiskSim Manual[1] and "PARAID" paper[2]. Perhaps we need more guidance from teacher and TAs.

F. Analysis and Implement in Source Code

main() defined in disksim_main.c
disksim_run_simulation() defined in disksim.c
disksim_simlulate_event() defined in disksim.c
io_internal_event() defined in disksim_iosim.c
iodriver_request() defined in disksim_iodriver.c
logorg_maprequest() defined in disksim_logorg.c
logorg_parity_table() defined in disksim_redun.c

1) Gear Shifting condition:

Up-shifts: U+S > T
U: average utilization of disks
S: standard deviation of U
U = (total_time - idle_time)/total_time

2) Remapping:

In this part,we do not implement it completely.

Each request is represented as a structure,contains device number , block number and so on.

All the requests are organized as a linked list,so that we can track back to reassign the number of each request.

The request is defined in disksim_global.c named ioreq_event.

V. *Power Management Module*

A. Parameters Initialize

To calculate the power consumption of PARAID-5, we need to know the power of each disk and the utilization time of each disk, maybe as well as the gear switching energy but here we just neglect it.Here we define the power of each disk is 14.2.

B. Results and Comparations

Running our python program before, we get detailed information about utilization time of each period.

```
gear 2: 5 disk
IOdriver Response time average:        24.303764
average utilization time: 0.4321564
stand deviation of utilization time: 0.0251287172503
U+S= 0.45728511725
>>>>>>>>
U+S<T, should down-shift.

gear 1: 4 disk
IOdriver Response time average:        27.285464
average utilization time: 0.5094005
stand deviation of utilization time: 0.0463932240568
U+S= 0.555793724057
>>>>>>>>
U+S<T, don't need to up-shift.

 ......
```

And we use a table to represent what we need.

gear	disks	U	P	U*P*disks
1	4	0.5094005	14.2	28.9339484
2	5	0.4321564	14.2	30.6831044

TABLE II
U*P*DISKS IS POWER CONSUMPTION

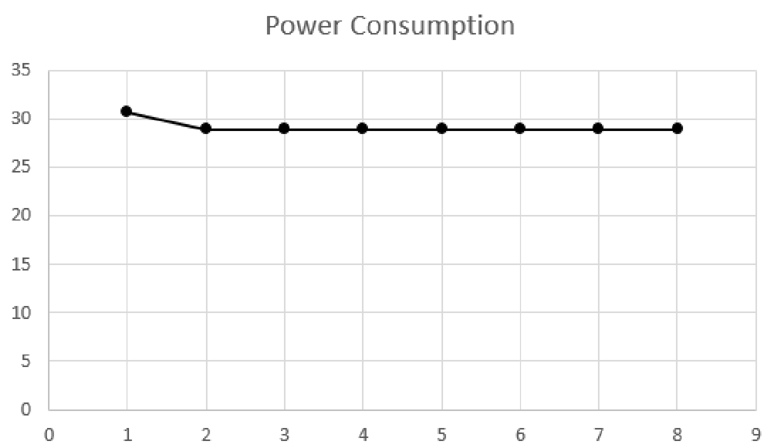

Fig. 6. Power consumption vs time

By gear shifting from 5-disk to 4-disk, about 5.7 percent of energy is preserved.

C. Theory Analysis

PARAID-5 is designed to save more energy while remain similar performance compared with RAID5. RAID5 is configued for peak performance and it keeps all disks spinning for light loads. In this case, there are two parts can be paid attention to decrease power consumtion.One is unused storage capacity caused by over-provision of storage capacity and unused storage can be traded for energy savings.Another is fluctuating load caused by frequent on-off power transitions.

According to what mentioned above, PARAID-5 uses group of disks of diffrenent size to form gears. The design of gears organized into hierarchical overlapping subsets and can utilize over-provisioned spare storage in RAID.When operating in small gear, other disks are powered off.To decide the shifting of gears, the workload should be approximated and gear shift into most appropriate gear, in order to minimize the opportunity lost to save power.The shift of gear is controlled by utilization time and stand deviation of it, which is detailed explained above.The disign is useful to adapt cyclic fluctuating workload.

Because of these kind of designs, PARAID can preserve energy.But how can it maintains a acceptable peroformance? Three ways should be used to achieve the goal. First, PARAID operate in the highest gear when the system demands peak performance and uses the same disk layout. Second, maximize parallelism within each gear. Third, delay block replication until gear shifting.Though our PARAID has only two gears, using three or more gears is of better performance in reality.Then disks can be divided into three types: busy disks(in all gears), standby disks(in gears exclude lower gears), and idle disks(in higher gears).Role exchange between outside disks with middle disks can maintain reliability.

D. Difficulties

It's too difficult that we even wasted many days reading everything can be searched about this mission but get little help. And after asking some people, we simply made a naive version and we found it difficult to understand all the theory behind it. Actually we hope to read some recommended papers from teacher and TAs.

References

[1] DiskSim Version 4.0 Reference Manual: http://www.pdl.cmu.edu/PDL-FTP/DriveChar/CMU-PDL-08-101.pdf

[2] Weddle, Charles, et al. "PARAID: The gearshifting power-aware RAID." Acm Transactions on Storage 3.3(2007):13.