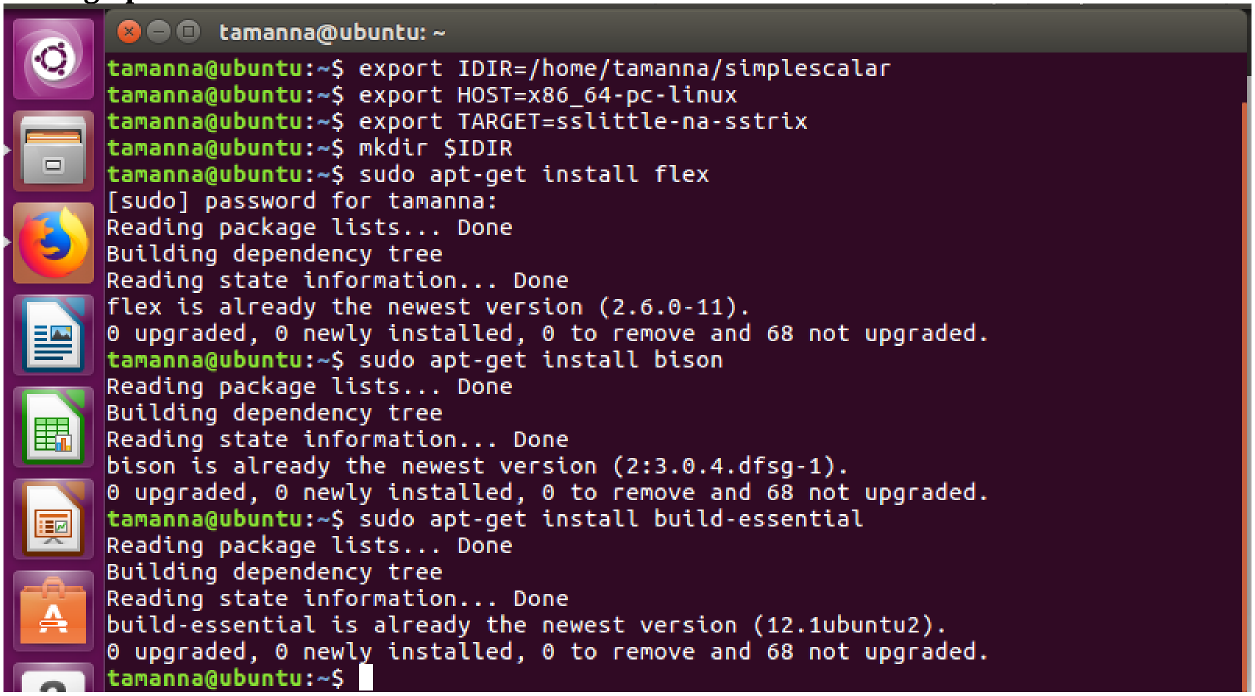
ECE 466 Advanced Computer Architecture
Project 1 Part 2

Tamanna Ravi Rupani
665679988

Setting up the environment to run simulation



```
tamanna@ubuntu:~$ export IDIR=/home/tamanna/simplescalar
tamanna@ubuntu:~$ export HOST=x86_64-pc-linux
tamanna@ubuntu:~$ export TARGET=sslittle-na-sstrix
tamanna@ubuntu:~$ mkdir $IDIR
tamanna@ubuntu:~$ sudo apt-get install flex
[sudo] password for tamanna:
Reading package lists... Done
Building dependency tree
Reading state information... Done
flex is already the newest version (2.6.0-11).
0 upgraded, 0 newly installed, 0 to remove and 68 not upgraded.
tamanna@ubuntu:~$ sudo apt-get install bison
Reading package lists... Done
Building dependency tree
Reading state information... Done
bison is already the newest version (2:3.0.4.dfsg-1).
0 upgraded, 0 newly installed, 0 to remove and 68 not upgraded.
tamanna@ubuntu:~$ sudo apt-get install build-essential
Reading package lists... Done
Building dependency tree
Reading state information... Done
build-essential is already the newest version (12.1ubuntu2).
0 upgraded, 0 newly installed, 0 to remove and 68 not upgraded.
tamanna@ubuntu:~$
```

Here, we will be performing the simulation for 200 Million instructions after fast forwarding the first 500 Million instructions, in an out-order fashion. These are the observations from the default branch predictor used by SimpleScalar.
The command used is,./sim-outorder -fastfwd 500000000 -max:inst 200000000 equake.ss < equake.in

Observations: (Refer to the Appendix 1 for screenshots of simulation)

1. -brpred - bimod (Branch predictor type)
2. -bpred:bimod - 2048 (History table size)
3. -bpred:2lev - 1(l1size) 1024(l2size) 8(hist_size) 0(xor)
4. -sim_num_insn - 200000000
5. -sim_cycle - 1183111204
6. -sim_IPC - 1.6905
7. -sim_CPI - 0.5916
8. -bpred_bimod.lookups - 56049388
9. -bpred_bimod.updates - 52795237
10. -bpred_bimod.addr_hits - 51867810
11. -bpred_bimod.dir_hits – 51867966
12. -bpred_bimod.misses - 927271

After making the changes and adding the new branch predictor type to bpred.c, we observe the following statistics. (Refer to the appendix for change in the code.) These are the observations from the new branch predictor added to SimpleScalar.

The command used is,./sim-outorder -fastfwd 500000000 -max:inst 200000000 equake.ss < equake.in

Observations: (Refer to the Appendix 2 for screenshots of simulation)

1. -brpred - bimod (Branch predictor type)
2. -bpred:bimod - 2048 (History table size)
3. -bpred:2lev - 1(l1size) 1024(l2size) 8(hist_size) 0(xor)
4. -sim_num_insn - 200000000
5. -sim_cycle - 1183111204
6. -sim_IPC - 1.6156
7. -sim_CPI - 0.6190
8. -bpred_bimod.lookups - 60743090
9. -bpred_bimod.updates - 52795237
10. -bpred_bimod.addr_hits - 50951729
11. -bpred_bimod.dir_hits – 50951888
12. -bpred_bimod.misses - 1843349

Analysis made from the above data

Order	IPC value	CPI value
Out-Order Execution for Original branch predictor.	1.6905	0.5916
Out-Order Execution for new branch predictor.	1.6156	0.6190

What we observe is, when we run the program for the original branch predictor CPI value is less as compared to the CPI value of the new branch predictor execution. So, if we compare the performance of both the executions, there is some performance loss in latter execution method.
On the other hand, more the IPC, better the performance.
IPC_O indicates original branch predictor execution.
IPC_N indicates new branch predictor execution.

$(IPC_O – IPC_N)/IPC_O * 100$
= (1.6905 – 1.6156)/1.6905 * 100
= 4.43%

We can conclude that there is a performance loss in new branch predictor execution by a percentage of **4.43%** based on IPC values**.**

Although the new branch predictor has more number of misses and less number of hits compared to the original predictor and the lookups also increase by a significant number.

Miss_O indicates misses in original branch predictor execution.
Miss_N indicates misses in new branch predictor execution.

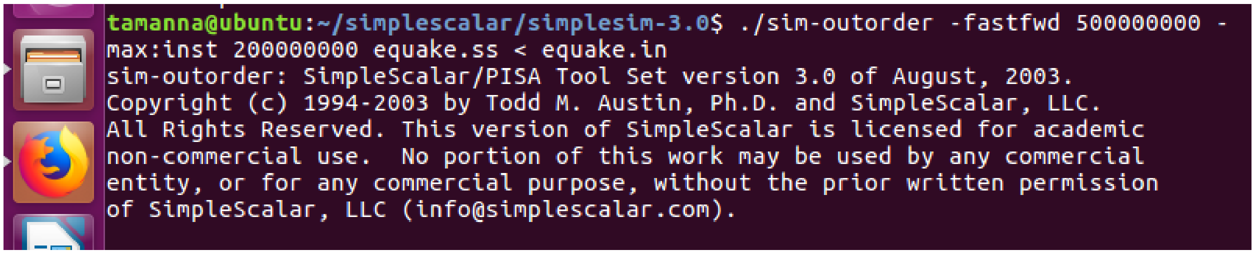
$(\text{Miss}_O – \text{Miss}_N)/\text{Miss}_O * 100$
$= (927271– 1843349)/ 927271 * 100$
= 98.77%

From the above calculation we can there is a 98.77% increase in the number of misses which is not the ideal situation we need.

Appendix 1

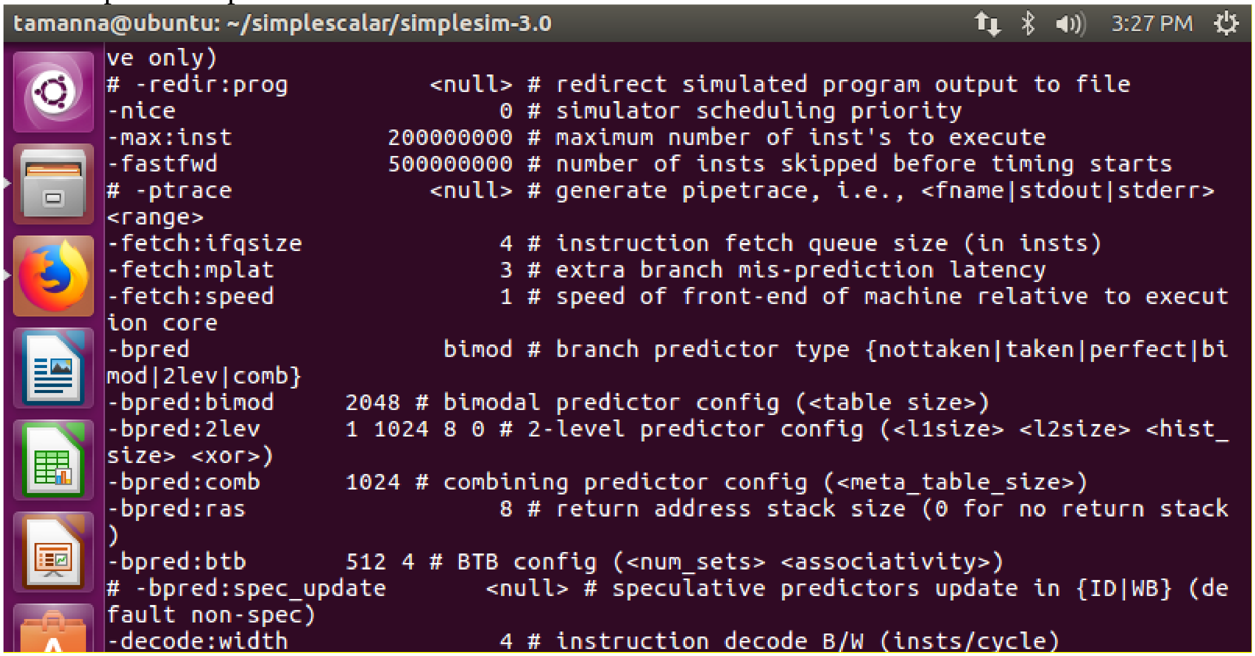
Unchanged code results (Original branch predictor)

Command



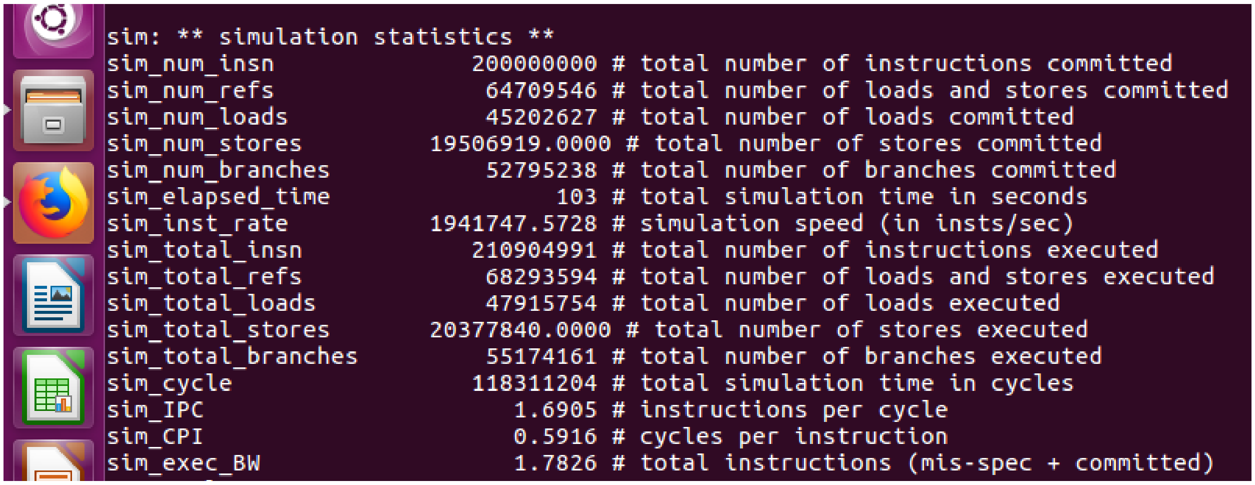
```
tamanna@ubuntu:~/simplescalar/simplesim-3.0$ ./sim-outorder -fastfwd 500000000 -
max:inst 200000000 equake.ss < equake.in
sim-outorder: SimpleScalar/PISA Tool Set version 3.0 of August, 2003.
Copyright (c) 1994-2003 by Todd M. Austin, Ph.D. and SimpleScalar, LLC.
All Rights Reserved. This version of SimpleScalar is licensed for academic
non-commercial use.  No portion of this work may be used by any commercial
entity, or for any commercial purpose, without the prior written permission
of SimpleScalar, LLC (info@simplescalar.com).
```

Branch predictor parameters



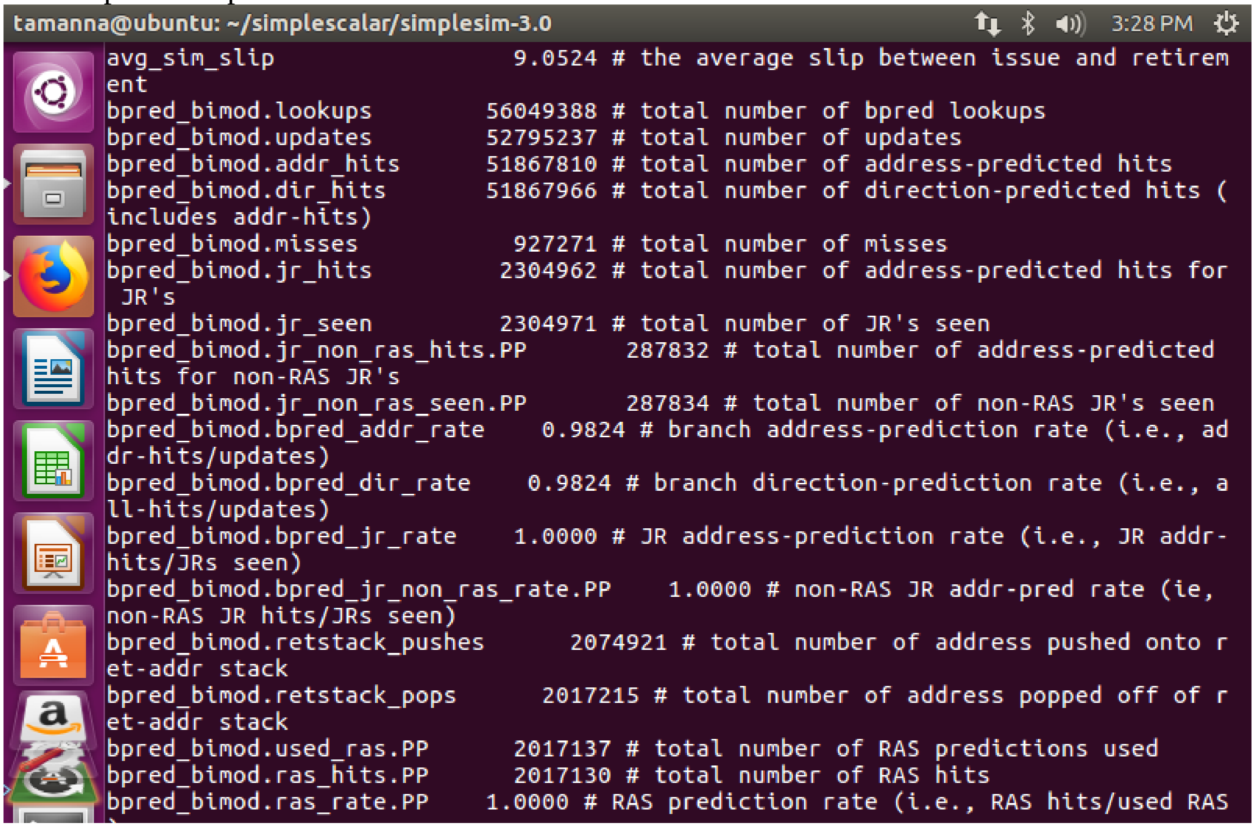
```
ve only)
# -redir:prog          <null> # redirect simulated program output to file
-nice                       0 # simulator scheduling priority
-max:inst           200000000 # maximum number of inst's to execute
-fastfwd            500000000 # number of insts skipped before timing starts
# -ptrace              <null> # generate pipetrace, i.e., <fname|stdout|stderr>
<range>
-fetch:ifqsize              4 # instruction fetch queue size (in insts)
-fetch:mplat                3 # extra branch mis-prediction latency
-fetch:speed                1 # speed of front-end of machine relative to execut
ion core
-bpred                  bimod # branch predictor type {nottaken|taken|perfect|bi
mod|2lev|comb}
-bpred:bimod     2048 # bimodal predictor config (<table size>)
-bpred:2lev      1 1024 8 0 # 2-level predictor config (<l1size> <l2size> <hist_
size> <xor>)
-bpred:comb      1024 # combining predictor config (<meta_table_size>)
-bpred:ras                  8 # return address stack size (0 for no return stack
)
-bpred:btb       512 4 # BTB config (<num_sets> <associativity>)
# -bpred:spec_update       <null> # speculative predictors update in {ID|WB} (de
fault non-spec)
-decode:width               4 # instruction decode B/W (insts/cycle)
```

Simulation statistics



```
sim: ** simulation statistics **
sim_num_insn                200000000 # total number of instructions committed
sim_num_refs                 64709546 # total number of loads and stores committed
sim_num_loads                45202627 # total number of loads committed
sim_num_stores          19506919.0000 # total number of stores committed
sim_num_branches             52795238 # total number of branches committed
sim_elapsed_time                  103 # total simulation time in seconds
sim_inst_rate            1941747.5728 # simulation speed (in insts/sec)
sim_total_insn              210904991 # total number of instructions executed
sim_total_refs               68293594 # total number of loads and stores executed
sim_total_loads              47915754 # total number of loads executed
sim_total_stores        20377840.0000 # total number of stores executed
sim_total_branches           55174161 # total number of branches executed
sim_cycle                   118311204 # total simulation time in cycles
sim_IPC                        1.6905 # instructions per cycle
sim_CPI                        0.5916 # cycles per instruction
sim_exec_BW                    1.7826 # total instructions (mis-spec + committed)
```

Branch predictor parameters

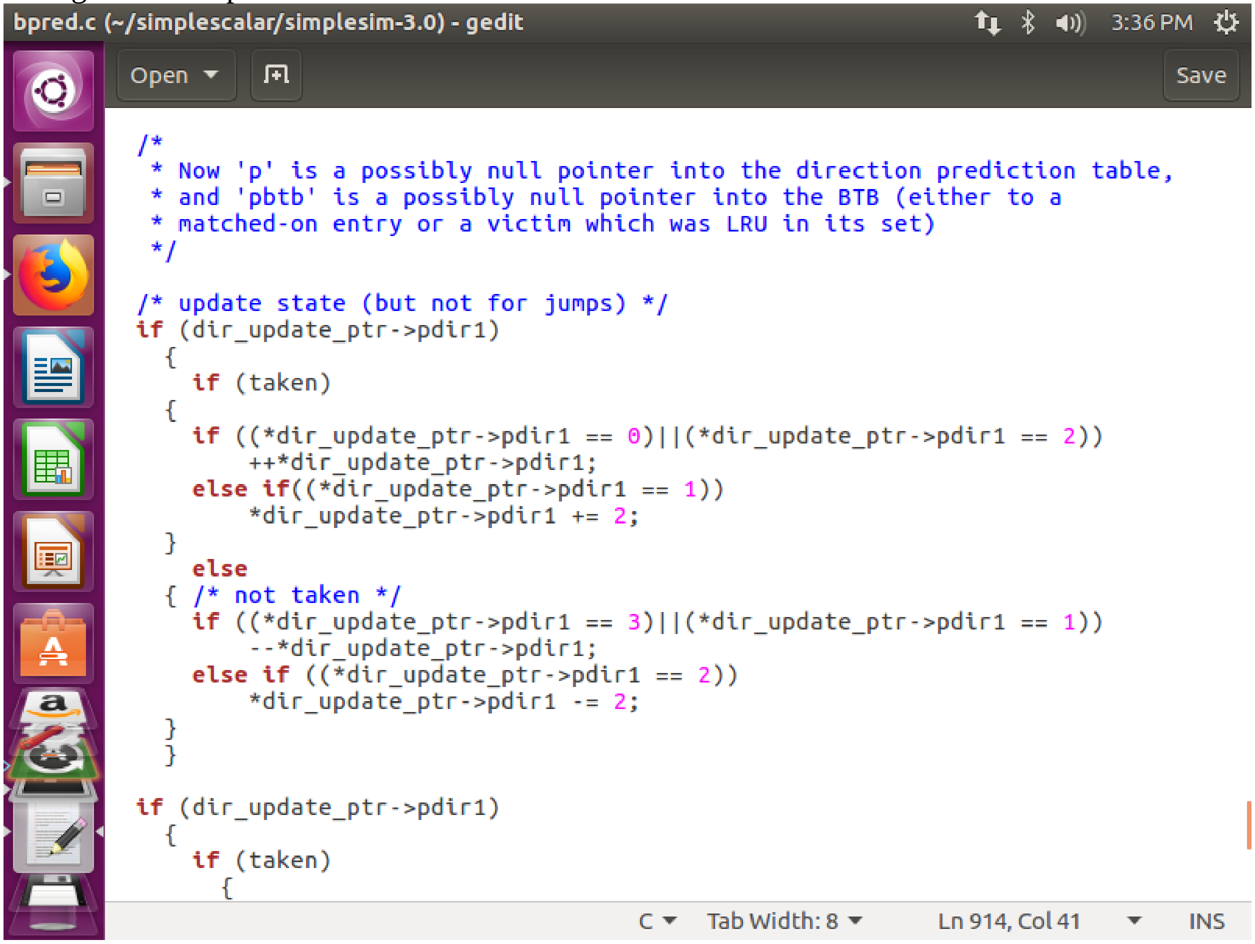


```
avg_sim_slip                  9.0524 # the average slip between issue and retirem
ent
bpred_bimod.lookups        56049388 # total number of bpred lookups
bpred_bimod.updates        52795237 # total number of updates
bpred_bimod.addr_hits      51867810 # total number of address-predicted hits
bpred_bimod.dir_hits       51867966 # total number of direction-predicted hits (
includes addr-hits)
bpred_bimod.misses           927271 # total number of misses
bpred_bimod.jr_hits         2304962 # total number of address-predicted hits for
 JR's
bpred_bimod.jr_seen         2304971 # total number of JR's seen
bpred_bimod.jr_non_ras_hits.PP       287832 # total number of address-predicted
hits for non-RAS JR's
bpred_bimod.jr_non_ras_seen.PP       287834 # total number of non-RAS JR's seen
bpred_bimod.bpred_addr_rate    0.9824 # branch address-prediction rate (i.e., ad
dr-hits/updates)
bpred_bimod.bpred_dir_rate    0.9824 # branch direction-prediction rate (i.e., a
ll-hits/updates)
bpred_bimod.bpred_jr_rate    1.0000 # JR address-prediction rate (i.e., JR addr-
hits/JRs seen)
bpred_bimod.bpred_jr_non_ras_rate.PP    1.0000 # non-RAS JR addr-pred rate (ie,
non-RAS JR hits/JRs seen)
bpred_bimod.retstack_pushes      2074921 # total number of address pushed onto r
et-addr stack
bpred_bimod.retstack_pops      2017215 # total number of address popped off of r
et-addr stack
bpred_bimod.used_ras.PP      2017137 # total number of RAS predictions used
bpred_bimod.ras_hits.PP      2017130 # total number of RAS hits
bpred_bimod.ras_rate.PP    1.0000 # RAS prediction rate (i.e., RAS hits/used RAS
```

Appendix 2

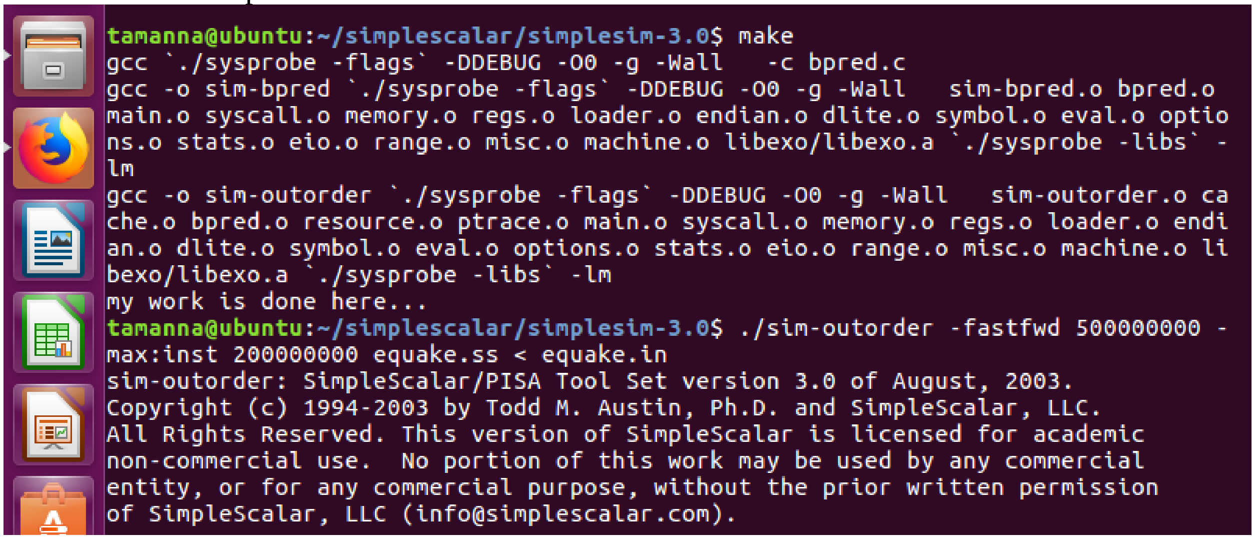
Changed code results (Newly implemented branch predictor)

Change in code bpred.c



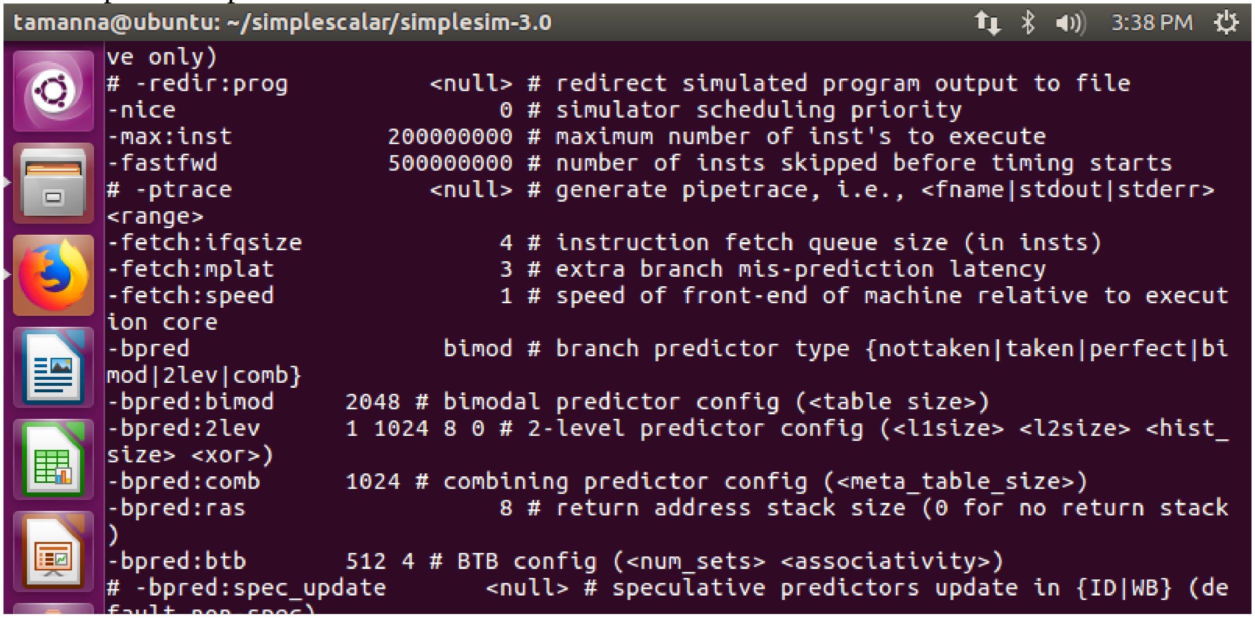
```
/*
 * Now 'p' is a possibly null pointer into the direction prediction table,
 * and 'pbtb' is a possibly null pointer into the BTB (either to a
 * matched-on entry or a victim which was LRU in its set)
 */

/* update state (but not for jumps) */
if (dir_update_ptr->pdir1)
    {
      if (taken)
    {
      if ((*dir_update_ptr->pdir1 == 0)||(*dir_update_ptr->pdir1 == 2))
          ++*dir_update_ptr->pdir1;
      else if((*dir_update_ptr->pdir1 == 1))
          *dir_update_ptr->pdir1 += 2;
    }
      else
    { /* not taken */
      if ((*dir_update_ptr->pdir1 == 3)||(*dir_update_ptr->pdir1 == 1))
          --*dir_update_ptr->pdir1;
      else if ((*dir_update_ptr->pdir1 == 2))
          *dir_update_ptr->pdir1 -= 2;
    }
    }

if (dir_update_ptr->pdir1)
    {
      if (taken)
        {
```

Command for compilation and simulation.



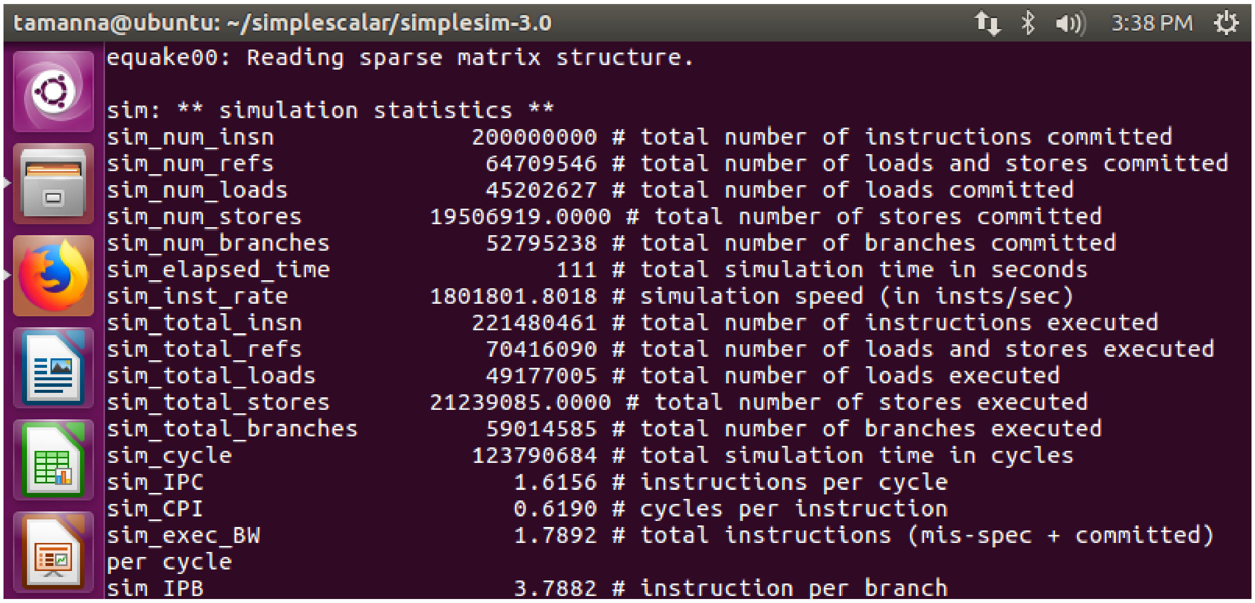
```
tamanna@ubuntu:~/simplescalar/simplesim-3.0$ make
gcc `./sysprobe -flags` -DDEBUG -O0 -g -Wall   -c bpred.c
gcc -o sim-bpred `./sysprobe -flags` -DDEBUG -O0 -g -Wall   sim-bpred.o bpred.o
main.o syscall.o memory.o regs.o loader.o endian.o dlite.o symbol.o eval.o optio
ns.o stats.o eio.o range.o misc.o machine.o libexo/libexo.a `./sysprobe -libs` -
lm
gcc -o sim-outorder `./sysprobe -flags` -DDEBUG -O0 -g -Wall   sim-outorder.o ca
che.o bpred.o resource.o ptrace.o main.o syscall.o memory.o regs.o loader.o endi
an.o dlite.o symbol.o eval.o options.o stats.o eio.o range.o misc.o machine.o li
bexo/libexo.a `./sysprobe -libs` -lm
my work is done here...
tamanna@ubuntu:~/simplescalar/simplesim-3.0$ ./sim-outorder -fastfwd 500000000 -
max:inst 200000000 equake.ss < equake.in
sim-outorder: SimpleScalar/PISA Tool Set version 3.0 of August, 2003.
Copyright (c) 1994-2003 by Todd M. Austin, Ph.D. and SimpleScalar, LLC.
All Rights Reserved. This version of SimpleScalar is licensed for academic
non-commercial use.  No portion of this work may be used by any commercial
entity, or for any commercial purpose, without the prior written permission
of SimpleScalar, LLC (info@simplescalar.com).
```

Branch predictor parameters



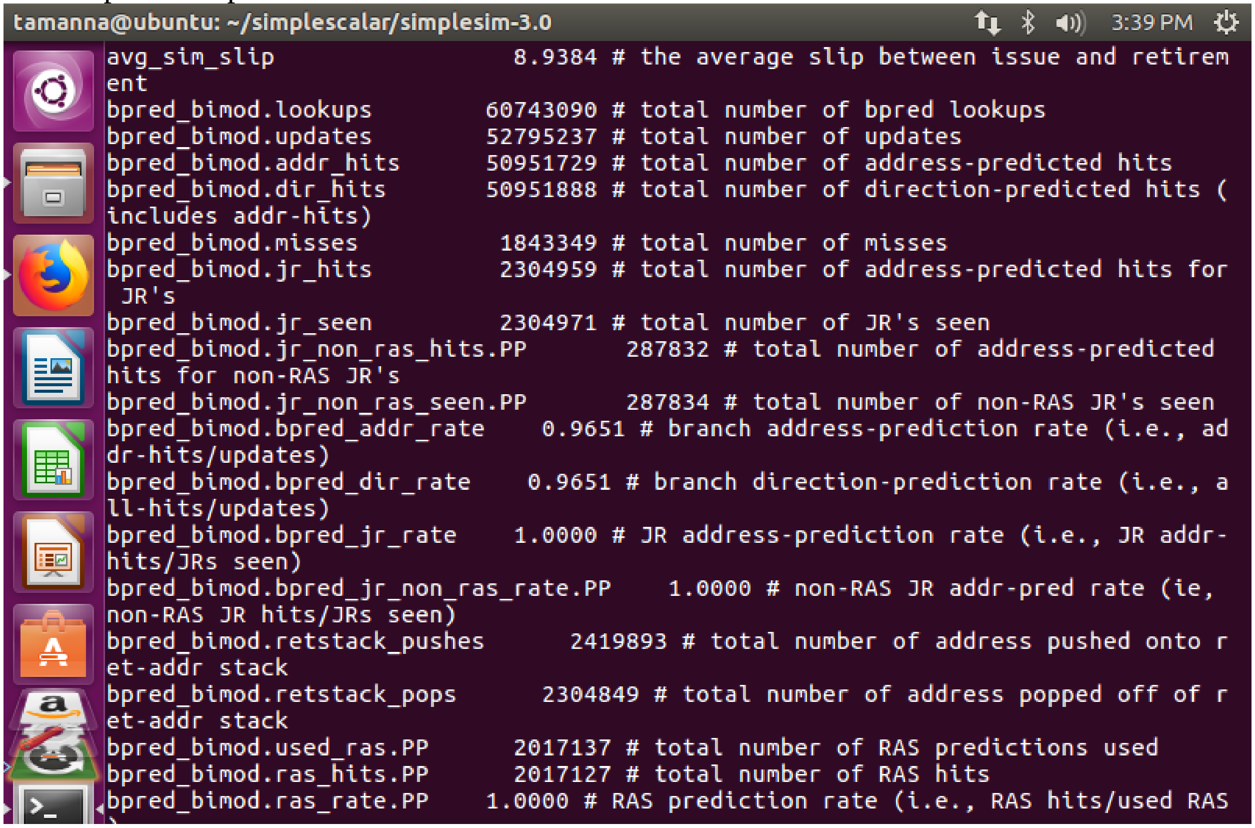
```
ve only)
# -redir:prog          <null> # redirect simulated program output to file
-nice                       0 # simulator scheduling priority
-max:inst           200000000 # maximum number of inst's to execute
-fastfwd            500000000 # number of insts skipped before timing starts
# -ptrace              <null> # generate pipetrace, i.e., <fname|stdout|stderr>
<range>
-fetch:ifqsize              4 # instruction fetch queue size (in insts)
-fetch:mplat                3 # extra branch mis-prediction latency
-fetch:speed                1 # speed of front-end of machine relative to execut
ion core
-bpred                  bimod # branch predictor type {nottaken|taken|perfect|bi
mod|2lev|comb}
-bpred:bimod     2048 # bimodal predictor config (<table size>)
-bpred:2lev      1 1024 8 0 # 2-level predictor config (<l1size> <l2size> <hist_
size> <xor>)
-bpred:comb      1024 # combining predictor config (<meta_table_size>)
-bpred:ras                  8 # return address stack size (0 for no return stack
)
-bpred:btb       512 4 # BTB config (<num_sets> <associativity>)
# -bpred:spec_update       <null> # speculative predictors update in {ID|WB} (de
fault non-spec)
```

Simulation statistics



```
equake00: Reading sparse matrix structure.

sim: ** simulation statistics **
sim_num_insn                200000000 # total number of instructions committed
sim_num_refs                 64709546 # total number of loads and stores committed
sim_num_loads                45202627 # total number of loads committed
sim_num_stores         19506919.0000 # total number of stores committed
sim_num_branches             52795238 # total number of branches committed
sim_elapsed_time                  111 # total simulation time in seconds
sim_inst_rate          1801801.8018 # simulation speed (in insts/sec)
sim_total_insn              221480461 # total number of instructions executed
sim_total_refs               70416090 # total number of loads and stores executed
sim_total_loads              49177005 # total number of loads executed
sim_total_stores       21239085.0000 # total number of stores executed
sim_total_branches           59014585 # total number of branches executed
sim_cycle                   123790684 # total simulation time in cycles
sim_IPC                        1.6156 # instructions per cycle
sim_CPI                        0.6190 # cycles per instruction
sim_exec_BW                    1.7892 # total instructions (mis-spec + committed)
per cycle
sim_IPB                        3.7882 # instruction per branch
```

Branch predictor parameters



```
avg_sim_slip                    8.9384 # the average slip between issue and retirem
ent
bpred_bimod.lookups          60743090 # total number of bpred lookups
bpred_bimod.updates          52795237 # total number of updates
bpred_bimod.addr_hits        50951729 # total number of address-predicted hits
bpred_bimod.dir_hits         50951888 # total number of direction-predicted hits (
includes addr-hits)
bpred_bimod.misses            1843349 # total number of misses
bpred_bimod.jr_hits           2304959 # total number of address-predicted hits for
 JR's
bpred_bimod.jr_seen           2304971 # total number of JR's seen
bpred_bimod.jr_non_ras_hits.PP        287832 # total number of address-predicted
hits for non-RAS JR's
bpred_bimod.jr_non_ras_seen.PP        287834 # total number of non-RAS JR's seen
bpred_bimod.bpred_addr_rate     0.9651 # branch address-prediction rate (i.e., ad
dr-hits/updates)
bpred_bimod.bpred_dir_rate     0.9651 # branch direction-prediction rate (i.e., a
ll-hits/updates)
bpred_bimod.bpred_jr_rate     1.0000 # JR address-prediction rate (i.e., JR addr-
hits/JRs seen)
bpred_bimod.bpred_jr_non_ras_rate.PP     1.0000 # non-RAS JR addr-pred rate (ie,
non-RAS JR hits/JRs seen)
bpred_bimod.retstack_pushes      2419893 # total number of address pushed onto r
et-addr stack
bpred_bimod.retstack_pops      2304849 # total number of address popped off of r
et-addr stack
bpred_bimod.used_ras.PP       2017137 # total number of RAS predictions used
bpred_bimod.ras_hits.PP       2017127 # total number of RAS hits
bpred_bimod.ras_rate.PP     1.0000 # RAS prediction rate (i.e., RAS hits/used RAS
)
```