OpenSeesMP

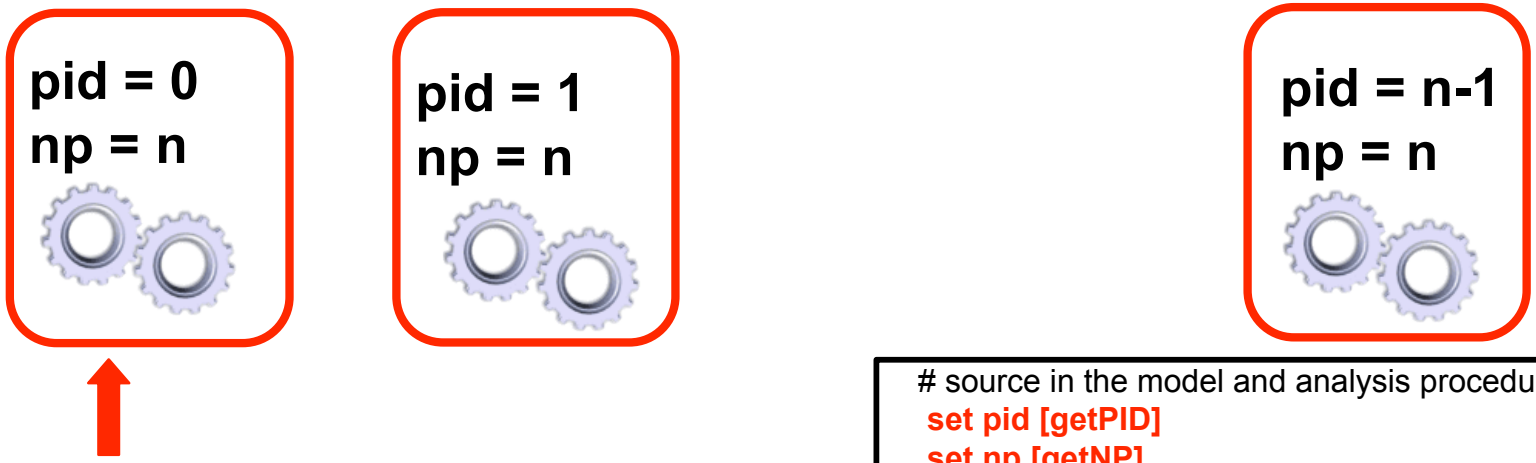
Frank McKenna
UC Berkeley

OpenSees Parallel Workshop
Berkeley, CA

OpenSeesMP

- OpenSeesMP was created for performing paramater studies AND for analyzing large models on parallel machines.
- OpenSeesMP requires the user have an understanding of parallel processing and requires that the user WRITE parallel scripts.
- It is more difficult of the two to use BUT it is the more powerful.

What is Running on the Processors?



Each process is running an interpreter. The interpreter can provide it's unique process number and the total number of processes in computation

Based on this script can do different things

```
# source in the model and analysis procedures
set pid [getPID]
set np [getNP]

# build model based on np and pid
source modelP.tcl
doModel {$pid $np}

# perform gravity analysis
system ParallelMumps
constraints Transformation
numberer ParallelPlain
test NormDispIncr 1.0e-12 10 3
algorithm Newton
integrator LoadControl 0.1

analysis Static

set ok [analyze 10]
return $ok
```

New Commands added:

- A Number of new commands have been added:

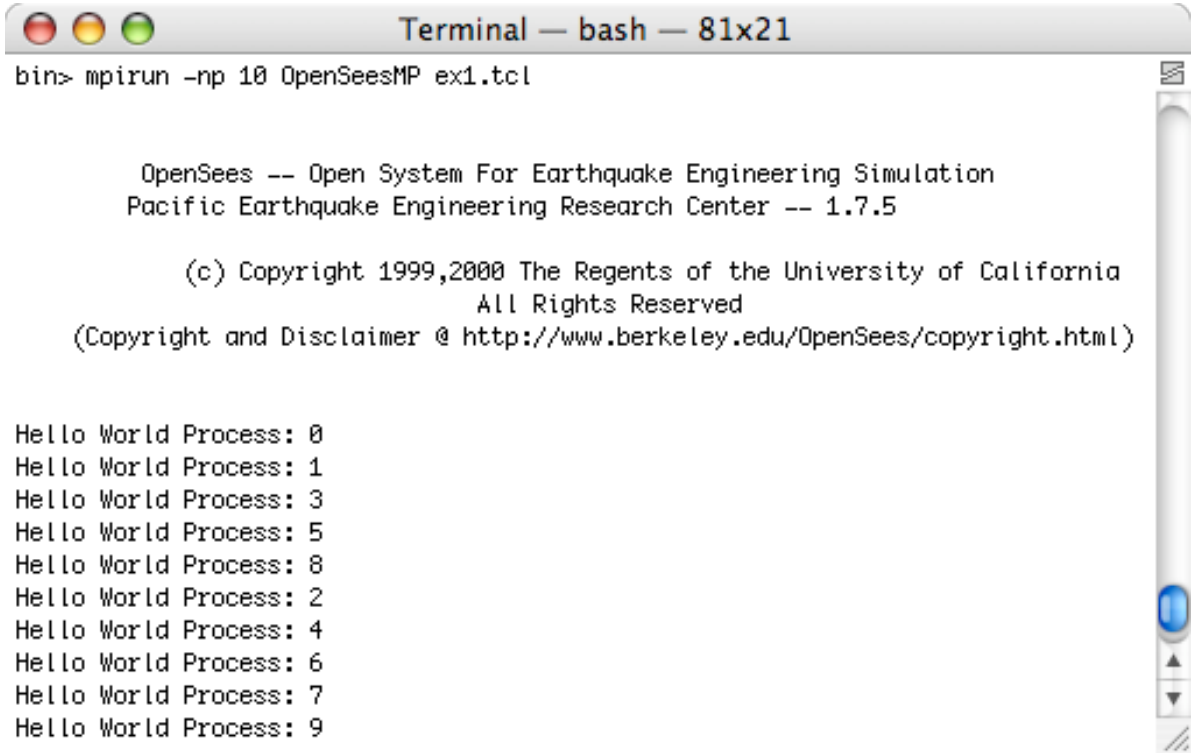
1. getNP returns number of processes in computation.
2. getPID returns unique pocess id {0,1, .. NP-1}
3. send -pid pid? data pid = { 0, 1, .., NP-1}
4. recv -pid pid? variableName pid = {0,1 .., NP-1, ANY}
5. barrier
6. domainChange

- These commands have been added to ALL interpreters (OpenSees, OpenSeesSP, and OpenSeesMP)

Example

ex1.tcl

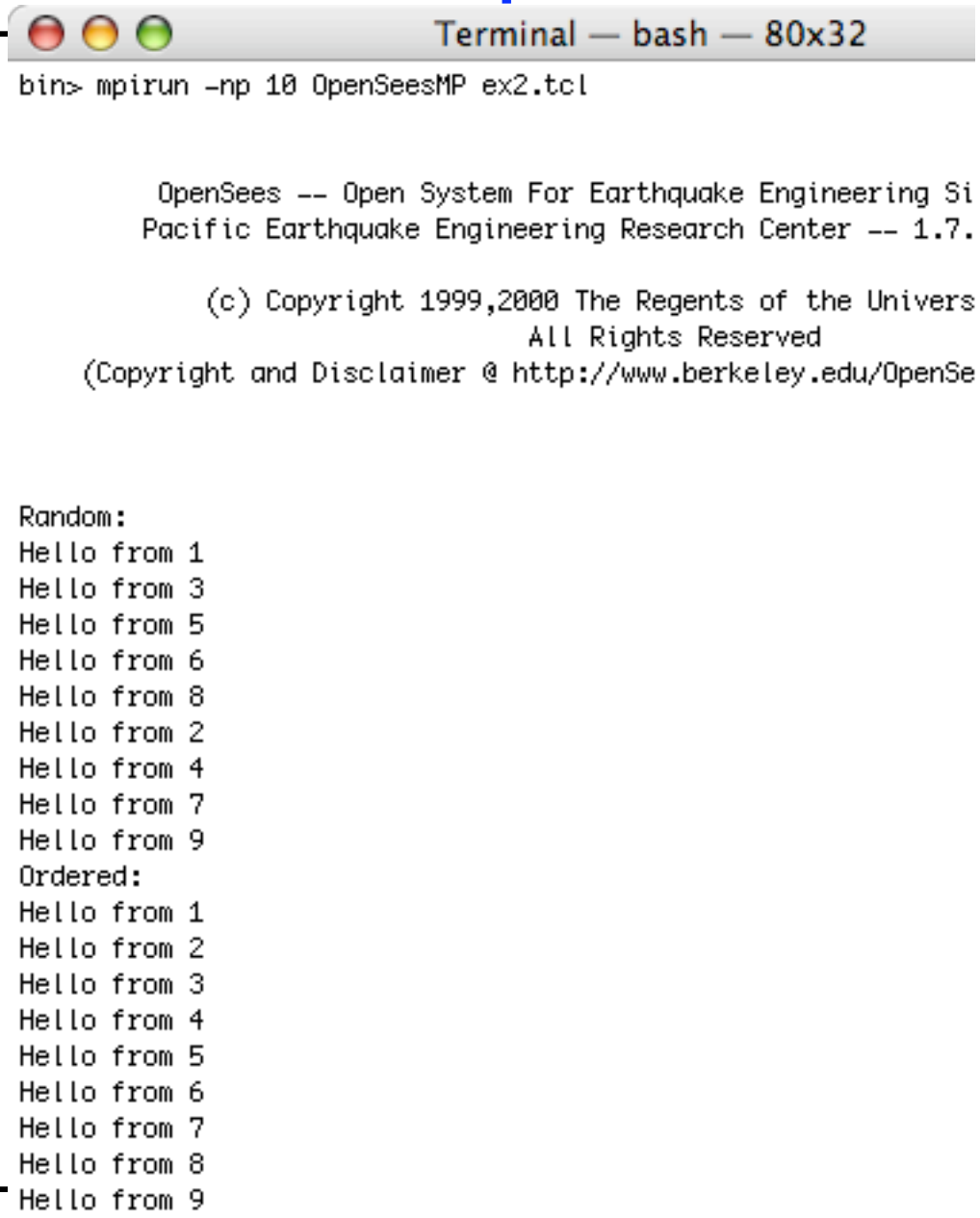
```
set pid [getPID]
set np [getNP]

puts "Hello World Process: $pid"
```



Another Example

ex2.tcl

```
set pid [getPID]
set np [getNP]
if {$pid == 0 } {
   puts ”Random:"
   for {set i 1 } {$i < $np} {incr i 1} {
      recv -pid ANY msg
      puts "$msg"
   }
} else {
   send -pid 0 "Hello from $pid"
}
barrier
if {$pid == 0 } {
   puts "\nOrdered:"
   for {set i 1 } {$i < $np} {incr i 1} {
      recv -pid $i msg
      puts "$msg"
   }
} else {
   send -pid 0 "Hello from $pid"
}
```



Modified Commands

- Some existing commands have been modified to allow analysis of large models in parallel:

1. numberer

```
numberer ParallelPlain
```

```
numberer ParallelRCM
```

2. system

```
system Mumps <-ICNTL14 %?>
```

3. integrator

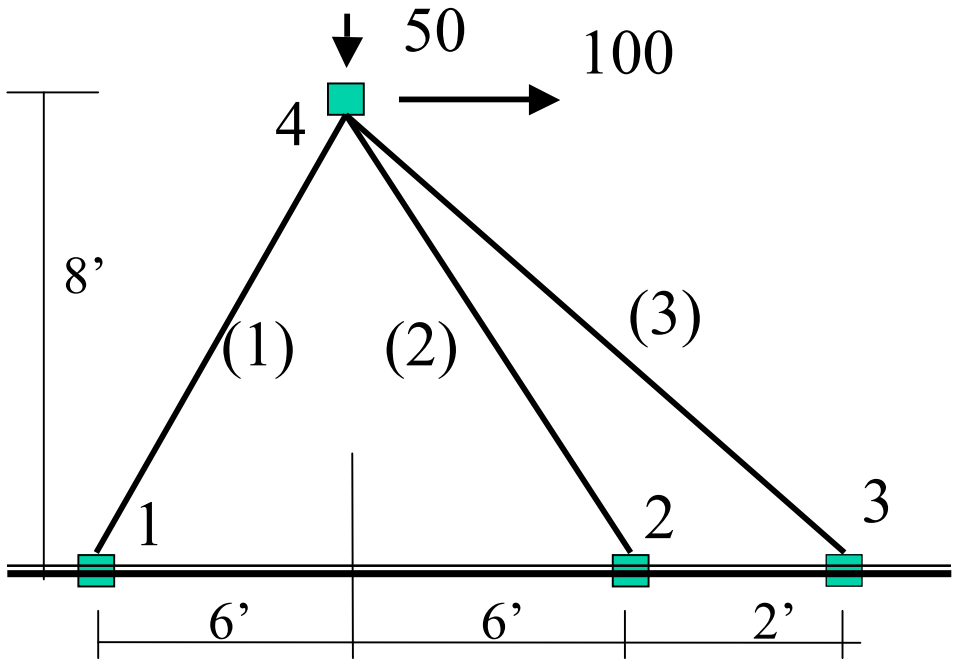
```
integrator ParallelDisplacementControl node? Dof? dU?
```

- Use these ONLY IF PARALLEL MODEL

Example Parallel Model:

ex4.tcl

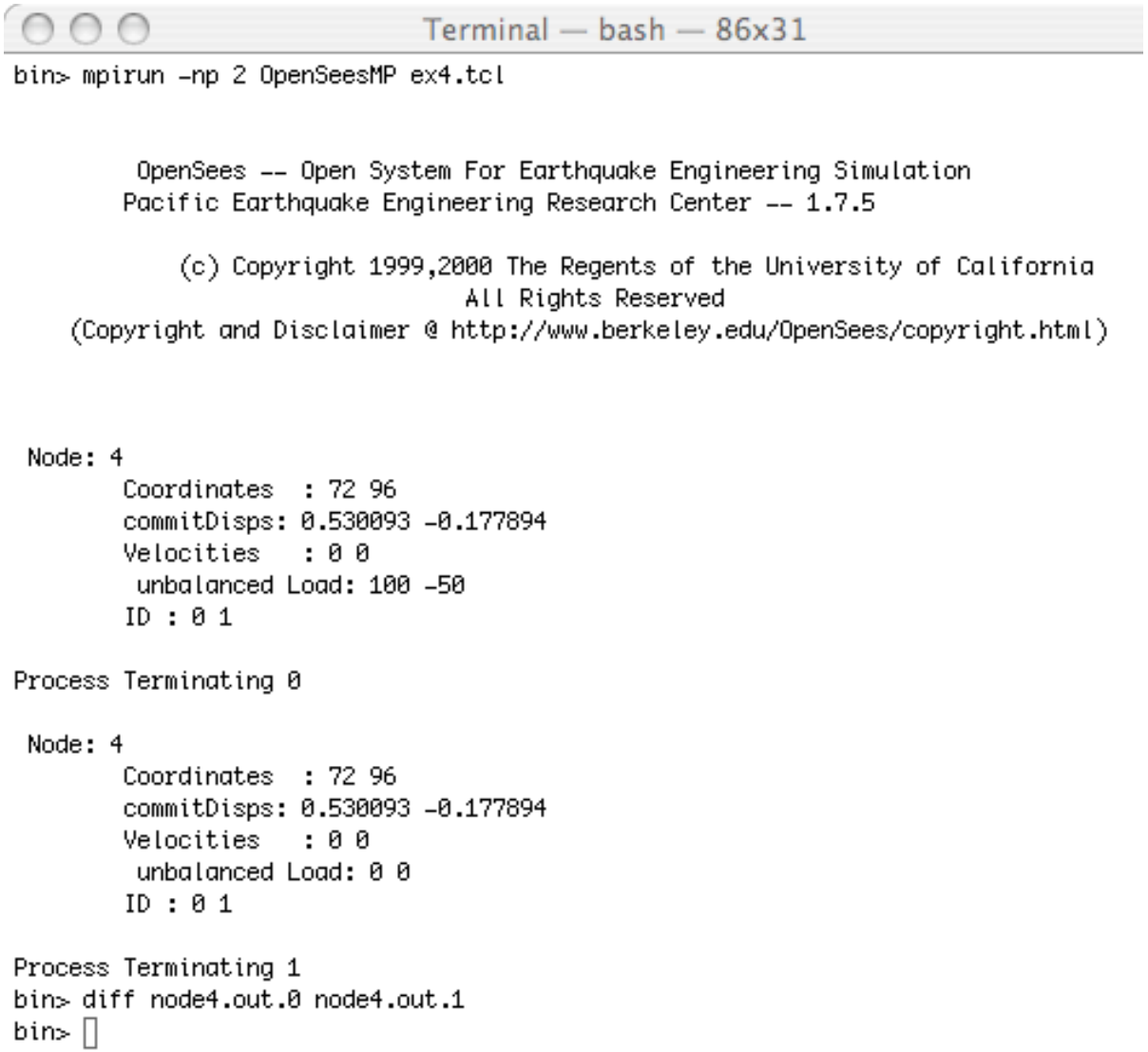
```
set pid [getPID]
set np [getNP]
if {$np != 2}  exit

model BasicBuilder -ndm 2 -ndf 2
uniaxialMaterial Elastic 1 3000
if {$pid == 0} {
    node 1   0.0  0.0
    node 4  72.0 96.0
    fix 1 1 1
    element truss 1 1 4 10.0 1
    pattern Plain 1 "Linear" {
        load 4 100 -50
    }
} else {
    node 2 144.0  0.0
    node 3 168.0  0.0
    node 4  72.0 96.0
    fix 2 1 1
    fix 3 1 1
    element truss 2 2 4 5.0 1
    element truss 3 3 4 5.0 1
}
```



	E	A
1	3000	10
2	3000	5
3	3000	5

Example Parallel Analysis:

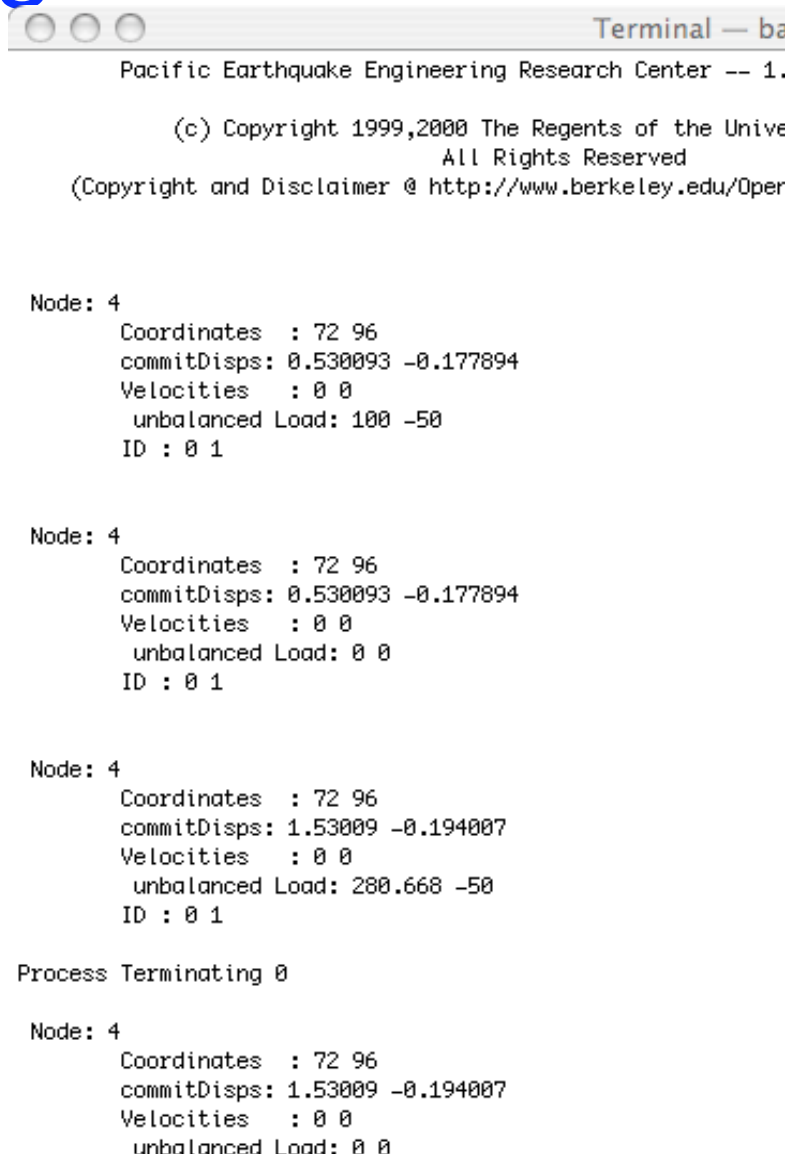
```
#create the recorder
recorder Node -file node4.out.$pid -node 4 -dof 1 2 disp

#create the analysis
constraints Transformation
numberer ParallelPlain
system Mumps
test NormDispIncr 1.0e-6 6 2
algorithm Newton
integrator LoadControl 0.1
analysis Static

#perform the analysis
analyze 10

# print to screen node 4
print node 4
```



Parallel Displacement Control and domainChange!

ex5.tcl

```
source ex4.tcl

loadConst - time 0.0

if {$pid == 0} {
    pattern Plain 2 "Linear" {
        load 4 1 0
    }
}

domainChange

integrator ParallelDisplacementControl 4 1 0.1
analyze 10
```



Things to Watch For

1. Deadlock
   - send/recv messages
   - Opening files for writing & not closing them
2. Race Conditions
   - parallel file system.
3. Load Imbalance
   - poor initial task assignment.

Watch out for Deadlock

- Match every send with a recv
- Watch the order

Deadlock Example

ex3.tcl

```
set pid [getPID]
set np [getNP]
if {$pid == 0 } {
   puts ”Random:"
   for {set i 1 } {$i < $np} {incr i 1} {
      recv -pid ANY msg
      puts "$msg"
   }
} else {
   send -pid 0 "Hello from $pid"
}
#barrier
if {$pid == 0 } {
   puts "\nOrdered:"
   for {set i 1 } {$i < $np} {incr i 1} {
      recv -pid $i msg
      puts "$msg"
   }
} else {
   send -pid 0 "Hello from $pid"
}
```

```
Terminal — bash — 84x36
bin> mpirun -np 10 OpenSeesMP ex2.tcl


        OpenSees -- Open System For Earthquake Engineering Simulation
      Pacific Earthquake Engineering Research Center -- 1.7.5

          (c) Copyright 1999,2000 The Regents of the University of Cali
                                  All Rights Reserved
    (Copyright and Disclaimer @ http://www.berkeley.edu/OpenSees/copyrigh



Random:
Hello from 2
Hello from 5
Hello from 2
Hello from 5
Hello from 3
Hello from 4
Hello from 6
Hello from 3
Hello from 4
Ordered:
Hello from 1
^Cmpirun: killing job...

^C--------------------------------------------------------------------------
WARNING: mpirun is in the process of killing a job, but has detected an
interruption (probably control-C).

It is dangerous to interrupt mpirun while it is killing a job (proper
termination may not be guaranteed).  Hit control-C again within 1
second if you really want to kill mpirun immediately.
```

Race Conditions and the File System

- Remember all processes can be reading/writing to the same files. If same file is opened for reading and writing, e.g. using a global file system to handle shared variable (opening as r+ will do what you want).
- This can also happen if you are modifying the directory structure in your script.

Watch out for Load Imbalance

- Load imbalance can greatly reduce the performance.
- Dynamic load balancing solutions can always be considered if performance is an issue.

Parameter Study

ex6.tcl

```
set np [getNP]
set pid [getPID]
set count 0

source model.tcl
source analysis.tcl

set tStart [clock seconds]

set recordsFile [open motionList r]
set lines [split [read $recordFile] \n]
foreach line $line {
  if {[expr $count % $np] == $pid} {

      doModel

      doGravityAnalysis;

      loadConst  -time 0.0
      set record [lindex $line 0]
      set npts   [lindex $line 1]
      set dt     [lindex $line 2]
      set accelSeries "Path -filePath $record -dt $dt -factor 386.4"
      pattern UniformExcitation  2   1  -accel $accelSeries

      set ok [doDynamicAnalysis $npts $dt]
      wipe
  }
   incr count 1
}
set tFinish [clock seconds]
barrier
puts "Duration Process $pid [expr $tFinish - $tStart]"
```

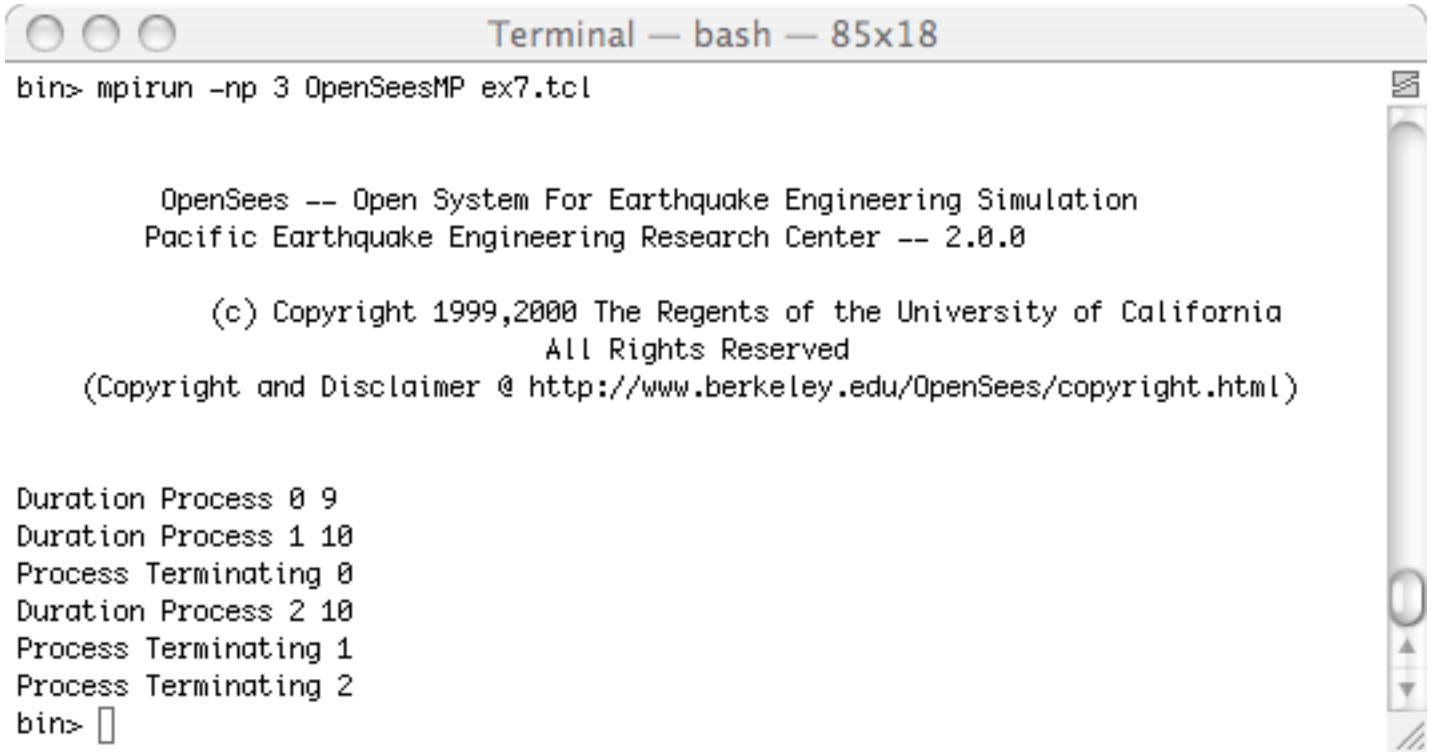
```
Terminal — bash — 84x16
bin> mpirun -np 2 OpenSeesMP ex6.tcl

        OpenSees -- Open System For Earthquake Engineer
      Pacific Earthquake Engineering Research Center --

          (c) Copyright 1999,2000 The Regents of the Un
                                All Rights Reserved
  (Copyright and Disclaimer @ http://www.berkeley.edu/

Duration Process 0 14
Process Terminating 0
Duration Process 1 6
Process Terminating 1
bin>
```

Simple Load Balancing

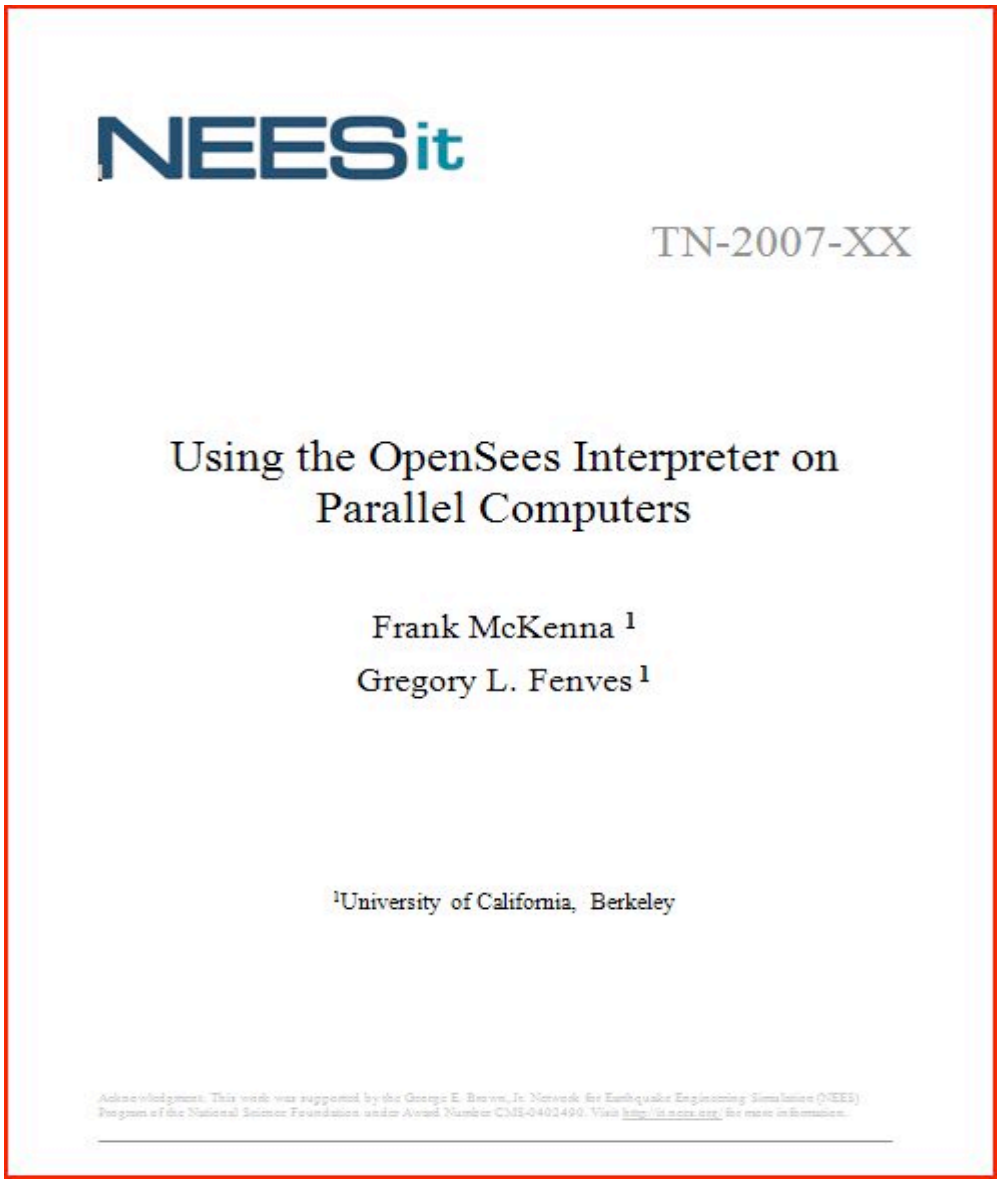
ex7.tcl

```
set np [getNP]
set pid [getPID]
set count 0

source model.tcl
source analysis.tcl

set tStart [clock seconds]
if {$pid == 0} {

  # Coordinator

   set recordsFile [open motionList r]
   set lines [split [read $recordsFile] \n]
   set numLines [llength $lines]
   foreach line $lines {
       recv -pid ANY pidWorker
       send -pid $pidWorker $line
   }

   for {set i 1} {$i < $np} {incr i 1} {
       send -pid $i "DONE"
   }

} else {

  # Worker

   set done NOT_DONE;
   while {$done != "DONE"} {
       send -pid 0 $pid
       recv -pid 0 line
       set record [lindex $line 0]

       if {$record == "DONE"} {
           break;
       }

       set npts   [lindex $line 1]
       set dt     [lindex $line 2]

       doModel;

       doGravityAnalysis;

       loadConst  -time 0.0

       set accelSeries "Path -filePath $record -dt $dt -factor 386.4"
       pattern UniformExcitation  2   1  -accel $accelSeries

       doRecorders $record $npts $dt
       set ok [doDynamicAnalysis $npts $dt]
       wipe
   }
}
set tFinish [clock seconds]
barrier
puts "Duration Process $pid [expr $tFinish - $tStart]"
```

else


Terminal — bash — 85x18
bin> mpirun -np 3 OpenSeesMP ex7.tcl
OpenSees -- Open System For Earthquake Engineering Simulation
Pacific Earthquake Engineering Research Center -- 2.0.0
(c) Copyright 1999,2000 The Regents of the University of California
All Rights Reserved
(Copyright and Disclaimer @ http://www.berkeley.edu/OpenSees/copyright.html)
Duration Process 0 9
Duration Process 1 10
Process Terminating 0
Duration Process 2 10
Process Terminating 1
Process Terminating 2
bin>

Documentation



NEESit

TN-2007-XX

Using the OpenSees Interpreter on Parallel Computers

Frank McKenna [1]

Gregory L. Fenves [1]

[1]University of California, Berkeley

Acknowledgment. This work was supported by the George E. Brown, Jr. Network for Earthquake Engineering Simulation (NEES) Program of the National Science Foundation under Award Number CMS-0402490. Visit http://it.nees.org/ for more information.

Any Questions?