

## Parallel Processing With OpenSees

Frank McKenna  
UC Berkeley

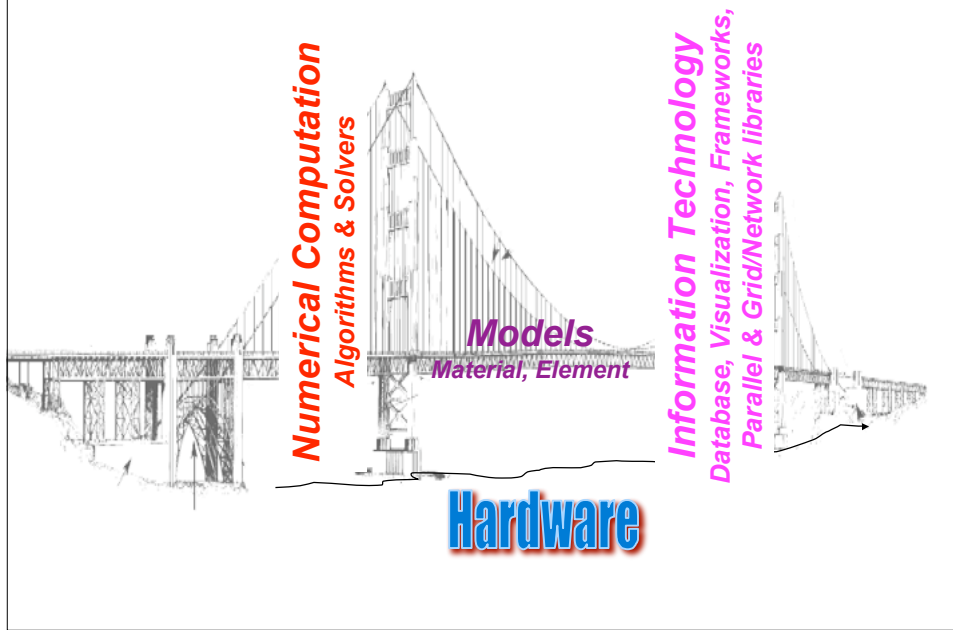
OpenSeesDays 2011



## Overview

- Hardware Trends
- Parallel Computing & OpenSees
- Cloud Computing & OpenSees

# Building Blocks for Simulation




## Bell's Law

### **Bell's Law of Computer Class formation**

was discovered about 1972. It states that technology advances in semiconductors, storage, user interface and networking advance every decade enable a new, usually lower priced computing platform to form. Once formed, each class is maintained as a quite independent industry structure. This explains mainframes, minicomputers, workstations and Personal computers, the web, emerging web services, palm and mobile devices, and ubiquitous interconnected networks. We can expect home and body area networks to follow this path.

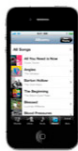
Gordon Bell, <http://research.microsoft.com/~GBell/Pubs.htm>

## Cloud Computing (according to Steve)



Some people think the cloud is just  
A hard drive in the sky!

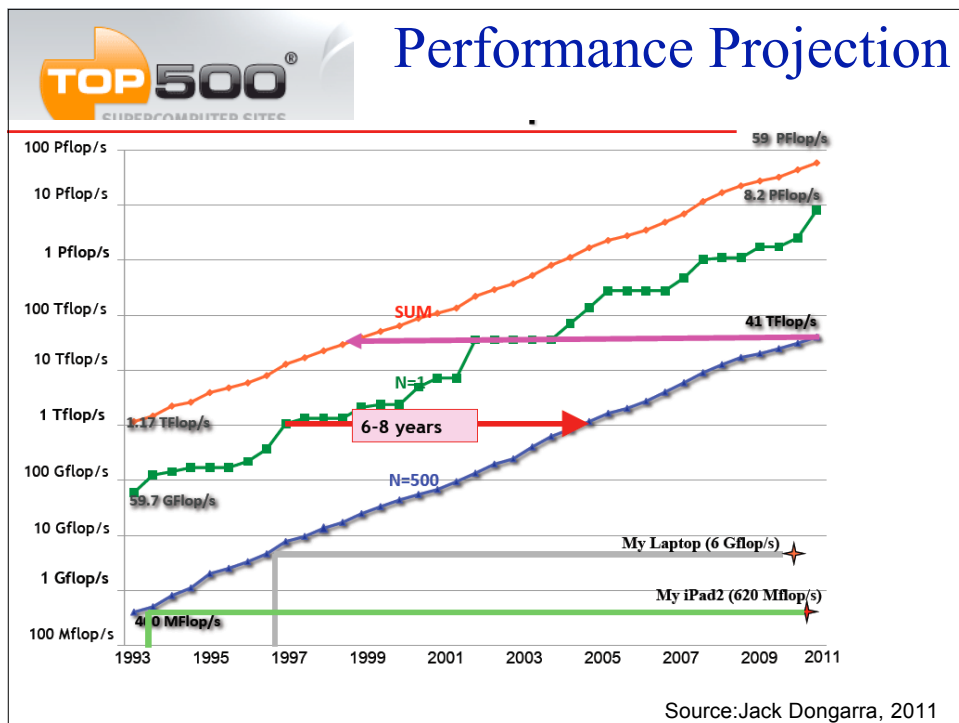
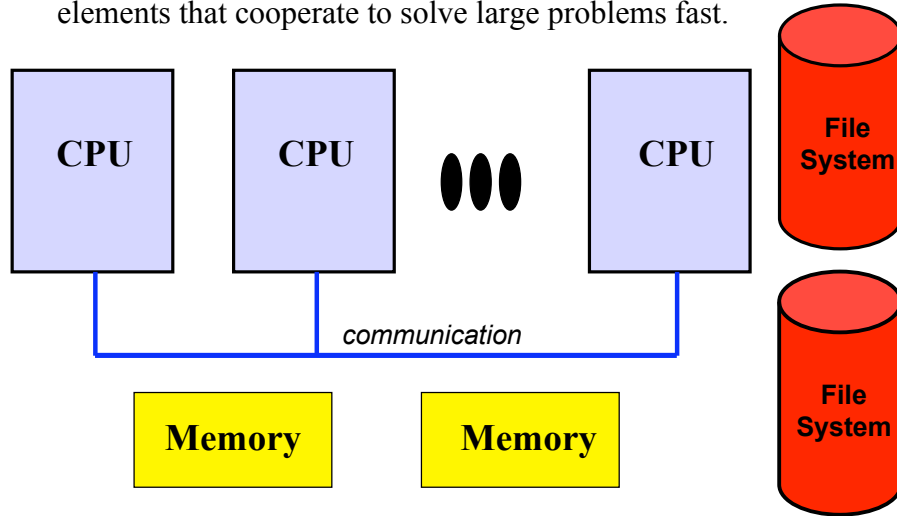
Cloud computing is internet-based computing ,  
whereby shared resources, software, and information  
are provided to computers and other devices on  
demand, like the electricity grid. source: wikipedia



“..PC and Mac Demoted to a Device”

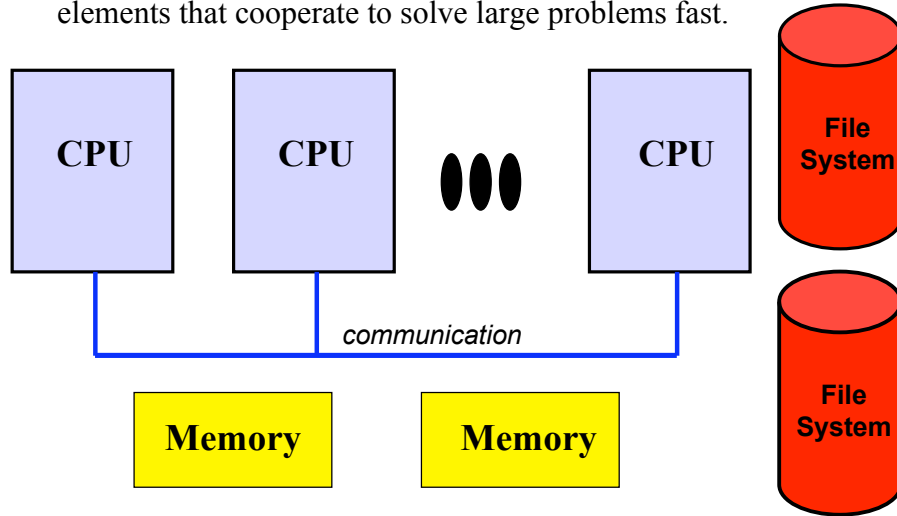
# What is a Parallel Computer?

- A *parallel computer* is a collection of processing elements that cooperate to solve large problems fast.



# What is a Parallel Computer?

- A *parallel computer* is a collection of processing elements that cooperate to solve large problems fast.



# Why should you care?

- They will save you time
- They will allow you to solve larger problems.
- They are **here** whether you like it or not!



K computer: 548,352 cores  
8 quadrillion calculations  
per second (8 petaflops -  
 $8 \times 10^{15}$  flops)

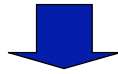


| Rank | Site   | Computer  |
|------|--|---|
| 1    | RIKEN Advanced Institute for Computational Science (AICS)<br>Japan | K computer, SPARC64 VIII fx 2.0GHz, Tofu interconnect<br>Fujitsu  |
| 2    | National Supercomputing Center in Tianjin<br>China                 | Tianhe-1A - NUDT TH MPP, X5670 2.93Ghz 6C, NVIDIA GPU, FT-1000 8C<br>NUDT                                   |
| 3    | DOE/SC/Oak Ridge National Laboratory<br>United States              | Jaguar - Cray XT5-HE Opteron 6-core 2.6 GHz<br>Cray Inc.  |
| 4    | National Supercomputing Centre in Shenzhen (NSCS)<br>China         | Nebulae - Dawning TC3600 Blade, Intel X5650, NVIDIA Tesla C2050<br>Cray Inc.                                |
| 5    | GSIC Center for Information Technology<br>Japan                    | Shiro - Cray XE6 8-core 2.4 GHz<br>Cray Inc.  |
| 6    | DOE/NNSA/LLNL<br>United States                                     | Pleiades - SGI Altix ICE 8200EX/8400EX, Xeon HT QC 3.0/Xeon 5570/5670 2.93 GHz, Infiniband SGI              |
| 7    | NASA/Ames Research Center/NAS<br>United States                     | Hopper - Cray XE6 12-core 2.1 GHz<br>Cray Inc.  |
| 8    | Commissariat a l'Energie Atomique (CEA)<br>France                  | Tera-100 - Bull bullx super-node S6010/S6030<br>Bull SA   |
| 9    | DOE/NNSA/LLNL<br>United States                                     | Roadrunner - BladeCenter QS22/LS21 Cluster, PowerXCell 8i 3.2 Ghz / Opteron DC 1.8 GHz, Voltaire Infiniband |

# BEFORE YOU GET ALL EXCITED

## Speedup & Amdahl's Law

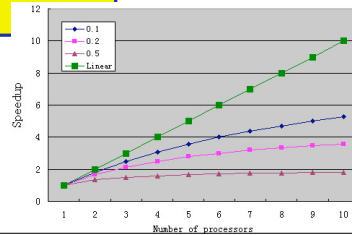
$$speedup_{PC}(p) = \frac{Time(1)}{Time(p)}$$



$$Speedup_{PC} = \frac{T_1}{\alpha T_1 + \frac{(1-\alpha)T_1}{n}} \rightarrow \frac{1}{\alpha} \text{ as } n \rightarrow \infty$$

Portion of sequential

# of processors



## Improving Real Performance

**Peak Performance grows exponentially, a la Moore's Law**

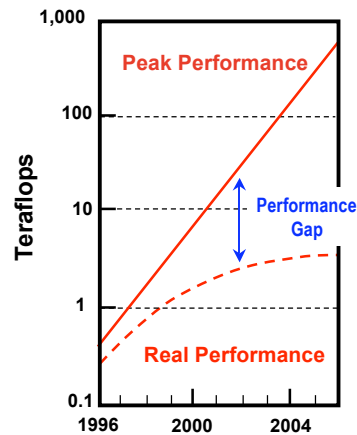
- In 1990's, peak performance increased 100x; in 2000's, it will increase 1000x

**But efficiency (the performance relative to the hardware peak) has declined**

- was 40-50% on the vector supercomputers of 1990s
- now as little as 5-10% on parallel supercomputers of today

**Close the gap through ...**

- Mathematical methods and algorithms that achieve high performance on a single processor and scale to thousands of processors
- More efficient programming models and tools for massively parallel supercomputers

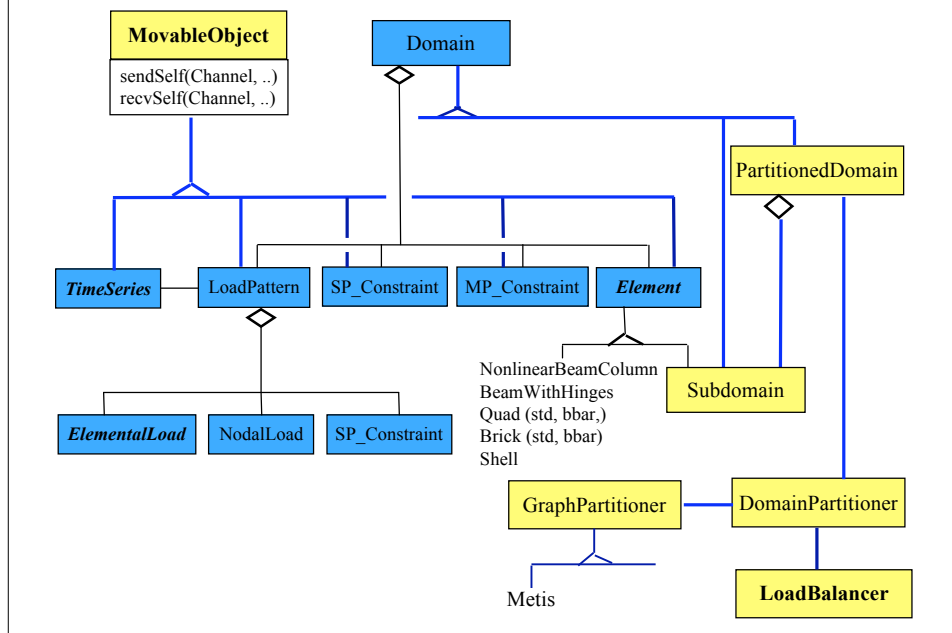


Source: Jim Demmell, CS267 Course Notes

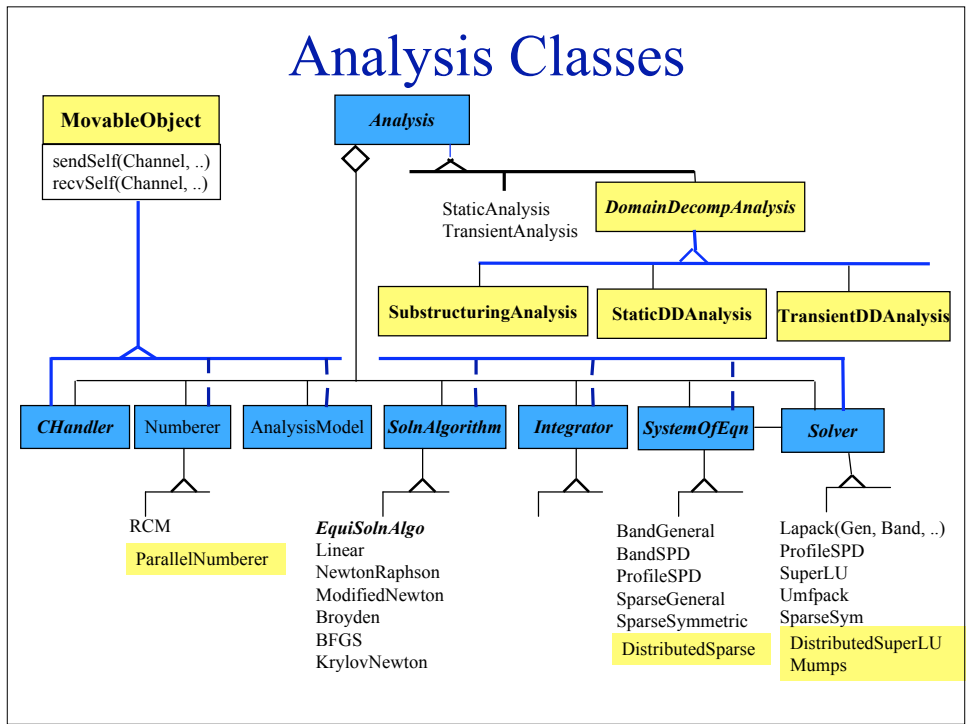
# What is OpenSees?

- OpenSees is an Open-Source Software Framework written in C++ for developing nonlinear Finite Element Applications for both sequential and **PARALLEL** environments.

## Domain Classes



# Analysis Classes



## The OpenSees Interpreters

- OpenSees.exe, OpenSeesSP.exe and OpenSeesMP.exe are applications that extend the Tcl interpreter for finite element.

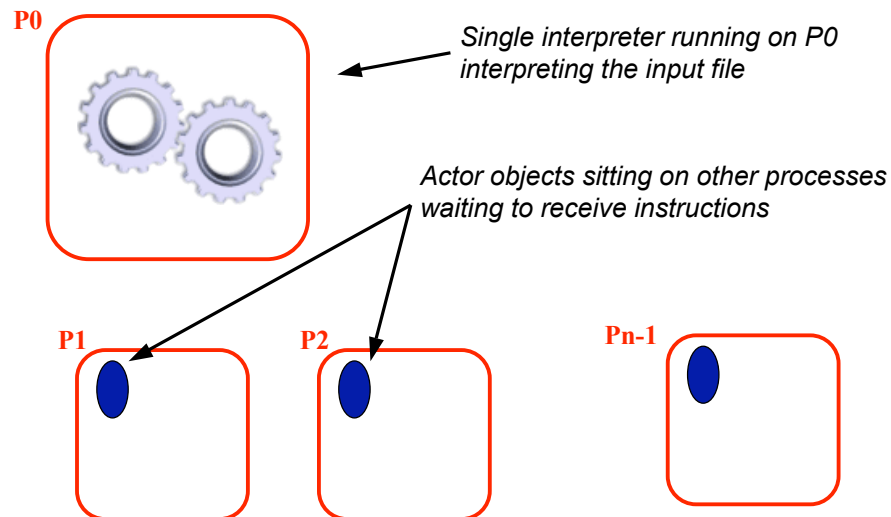
So What are OpenSeesSP.exe and OpenSeesMP.exe ?



## Parallel OpenSees Interpreters

- OpenSeesSP: An application for large models which will parse and execute the exact same script as the sequential application. The difference being the element state determination and equation solving are done in parallel.
- OpenSeesMP: An application for **BOTH** large models and parameter studies.

### OpenSeesSP: An application for Large Models



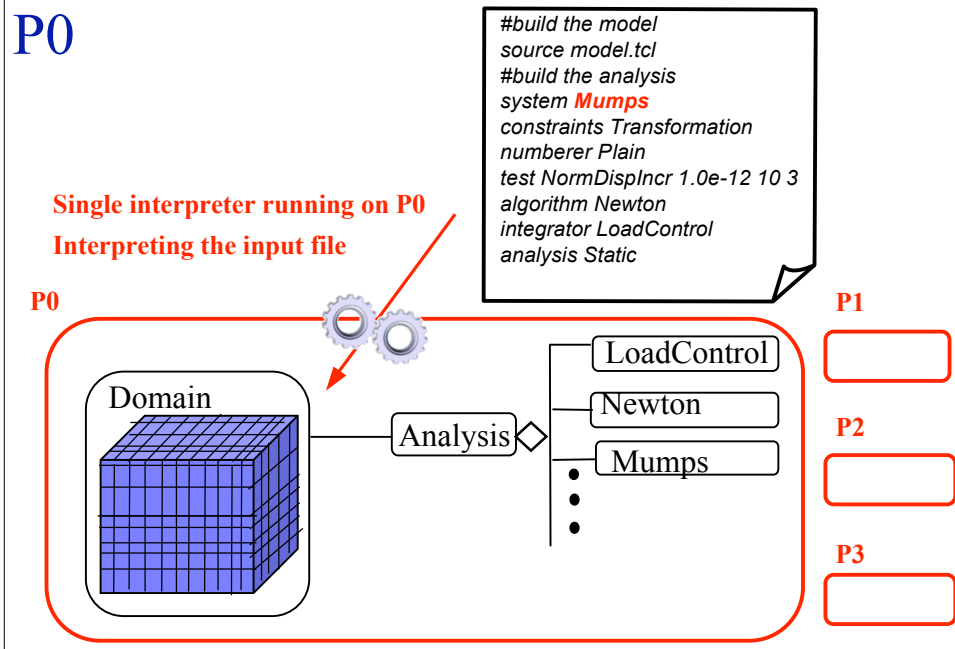
## Modified Commands

- System command is modified to accept new parallel equation solvers

system Mumps

system Diagonal

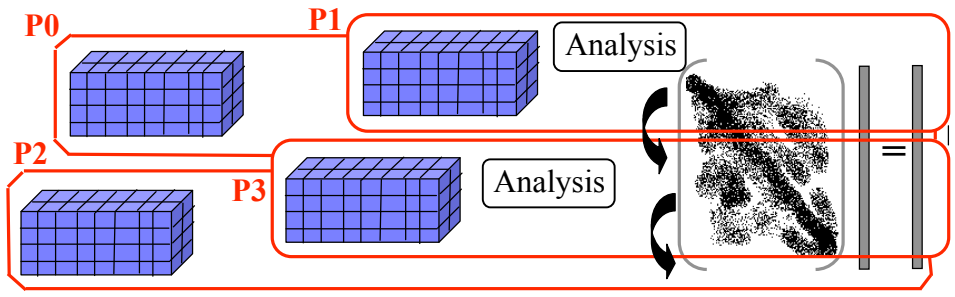
## Model Built and Analysis Constructed in P0



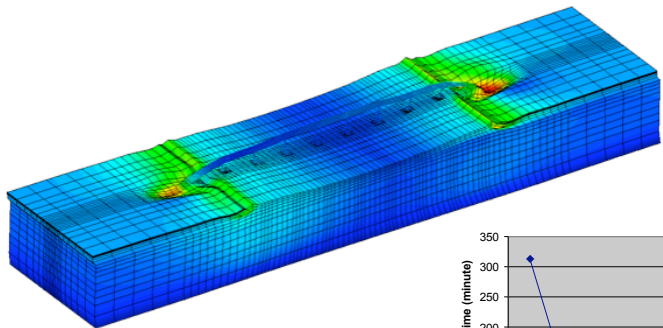
```

#build the model
source modelP.tcl
#build the analysis
system Mumps
constraints Transformation
numberer Plain
test NormDisplncr 1.0e-12 10 3
algorithm Newton
integrator LoadControl
analysis Static
analyze 10

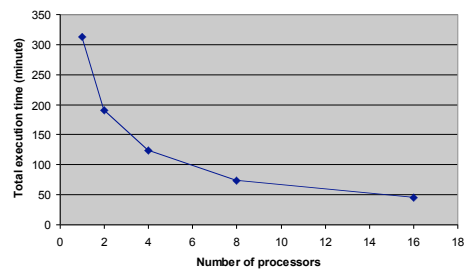
```



## Example Usage: Humboldt Bay Bridge Model



100,000+ DOF Model  
Implicit Integration  
Mumps Direct Solver



| Run | el. size (m) | Elements   | Nodes      | DOFs       |
|-----|--------------|------------|------------|------------|
| A   | 20           | 54,026     | 59,032     | 156,768    |
| B   | 10           | 404,751    | 424,512    | 1,193,283  |
| C   | 5            | 3,130,301  | 3,208,822  | 9,307,563  |
| D   | 2.5          | 24,615,801 | 24,928,842 | 73,515,123 |

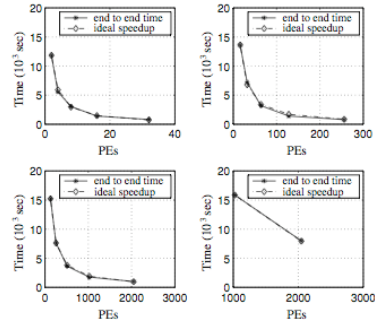
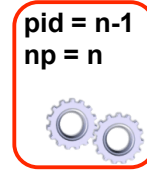
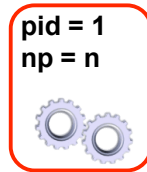
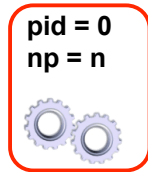


Fig. 18 Fixed-size, scalability plot at SDSC's DataStar. Upper row is runs A (left) and B (right), lower row is runs C (left) and D (right) (Table 3)

## OpenSeesMP: An application for Large Models and Parameter Studies



Each process is running an interpreter and can determine 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 modelIP.tcl
doModel {$pid $np}

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

analysis Static

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

## New Commands added to OpenSeesMP:

- A Number of new commands have been added:
  1. `getNP` returns number of processes in computation.
  2. `getPID` returns unique process 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)

```
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"
}
}
```

### Example

```
Terminal -- bash -- 80x32
bin> mpirun -np 10 OpenSeesMP ex2.tcl

OpenSees -- Open System For Earthquake Engineering Simulation
Pacific Earthquake Engineering Research Center -- 1.7.1
(c) Copyright 1999,2000 The Regents of the University of California
All Rights Reserved
(Copyright and Disclaimer @ http://www.berkeley.edu/OpenSees)

Random:
Hello from 1
Hello from 3
Hello from 5
Hello from 6
Hello from 8
Hello from 2
Hello from 4
Hello from 7
Hello from 9
Ordered:
Hello from 1
Hello from 2
Hello from 3
Hello from 4
Hello from 5
Hello from 6
Hello from 7
Hello from 8
Hello from 9
```

## Steel Building Study

```

set pid [getPID]
set np [getNP]
set recordsFileID [open "peerRecords.txt" r]
set count 0;

```

```

foreach gMotion [split [read $recordsFileID] ¶n] {
  if {[expr $count % $np] == $pid} {

```

```

    source model.tcl
    source analysis.tcl

```

```

    set ok [doGravity]

```

```

    loadConst -time 0.0

```

```

    set gMotionList [split $gMotion "/"]
    set gMotionDir [lindex $gMotionList end-1]
    set gMotionNameInclAT2 [lindex $gMotionList end]
    set gMotionName [string range $gMotionNameInclAT2 0 end-4 ]

```

```

    set Gaccel "PeerDatabase $gMotionDir $gMotionName -accel 384.4 -dT dT -nPts nPts"
    pattern UniformExcitation 2 1 -accel $Gaccel

```

```

    recorder EnvelopeNode -file $gMotionDir$gMotionName.out -node 3 4 -dof 1 2 3 disp

```

```

    doDynamic [expr $dT*$nPts] $dT

```

```

    wipe
  }

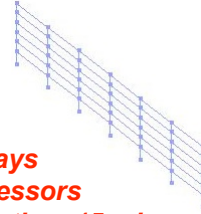
```

```

  incr count 1;
}

```

**7200 records**  
**2 min a record**  
**240 hours or 10 days**  
**Ran on 2000 processors**  
**on teragrid in less than 15 min.**



## Concrete Building Study

```

set pid [getPID]
set np [getNP]
set count 0;
source parameters.tcl
source ReadSMDFileNewFormat.tcl;
foreach GMfile $GMFile {
  foreach Factor1248 $IFactor1248 {

```

```

    if {[expr $count % $np] == $pid} {

```

```

      set inFile $GMDir/$GMfile.AT2
      set outFile $GMDir/$GMfile.g3;
      ReadSMDFileNewFormat $inFile $outFile dt npts;

```

```

      wipe
      source GravityAnalysisScript.tcl

```

```

      loadConst -time 0.0;
      wipeAnalysis

```

```

      source EQ_Recorder.tcl
      source EQAnalysisScript.tcl

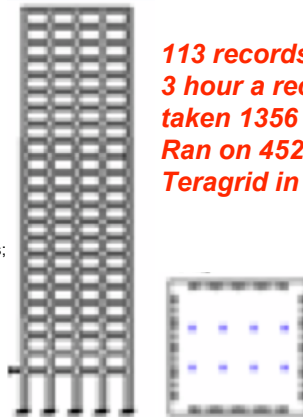
```

```

      if {$ok == 0} {
        puts "Process $pid $GMfile x $Factor1248 FINISHED OK modelTime [getTime]"
      } else {
        puts "Process $pid $GMfile x $Factor1248 FINISHED FAIL modeTime [getTime] desiredTime $TmaxAnalysis]"
      }
      incr count 1
    }
  }
}

```

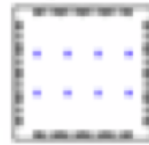
**113 records, 4 intensities**  
**3 hour a record, would have**  
**taken 1356 hours or 56.5 days**  
**Ran on 452 processors of a**  
**Teragrid in less than 5 hours.**



## Concrete Building Study



**113 records,  
4 intensities  
3 hour a record  
1356 hours  
or 56.5 days  
Ran on 452  
processors  
on teragrid in less  
than 5 hours.**



## 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:

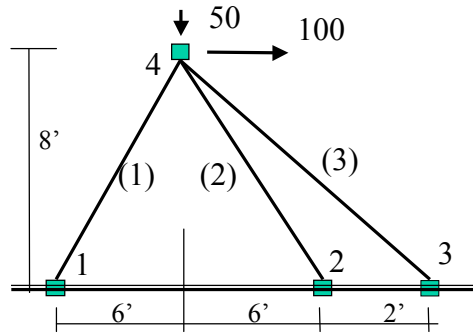
ex4.tcl

```

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

model BasicBuilder -ndm 2 -ndf 2
uniaxialMaterial Elastic 1 3000
if {$spid == 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.$spid -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

```

```

Terminal -- bash -- 86x31
bin> mpirun -np 2 OpenSeesMP ex4.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 California
All Rights Reserved
(Copyright and Disclaimer @ http://www.berkeley.edu/OpenSees/copyright.html)

Node: 4
Coordinates : 72 96
commitDisps: 0.530093 -0.177894
Velocities : 0 0
unbalanced Load: 100 -50
ID : 0 1

Process Terminating 0

Node: 4
Coordinates : 72 96
commitDisps: 0.530093 -0.177894
Velocities : 0 0
unbalanced Load: 0 0
ID : 0 1

Process Terminating 1
bin> diff node4.out.0 node4.out.1
bin>

```



# 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
```

```
Terminal -- ba
Pacific Earthquake Engineering Research Center -- 1
(c) Copyright 1999,2000 The Regents of the Unive
All Rights Reserved
(Copyright and Disclaimer @ http://www.berkeley.edu/Oper

Node: 4
Coordinates : 72 96
commitDisps: 0.530093 -0.177894
Velocities : 0 0
unbalanced Load: 100 -50
ID : 0 1

Node: 4
Coordinates : 72 96
commitDisps: 0.530093 -0.177894
Velocities : 0 0
unbalanced Load: 0 0
ID : 0 1

Node: 4
Coordinates : 72 96
commitDisps: 1.53009 -0.194007
Velocities : 0 0
unbalanced Load: 200.668 -50
ID : 0 1

Process Terminating 0

Node: 4
Coordinates : 72 96
commitDisps: 1.53009 -0.194007
Velocities : 0 0
unbalanced Load: 0 0
```

## Things to Watch For

1. Deadlock (program hangs)
  - send/rcv messages
  - Opening files for writing & not closing them
2. Race Conditions (different results every time run problem)
  - parallel file system.
3. Load Imbalance
  - poor initial task assignment.

# Grid Computing

- Most distributed form of parallel computing
- Computers communicating over the internet to solve a given problem
- Low bandwidth and extremely high latency, typically only used for embarrassingly parallel problems, i.e. parameter studies.

## Grid Computing - computational resources

The screenshot shows the BOINC website interface. At the top, the BOINC logo is on the left, and the text "Open-source software for volunteer computing and grid computing." is in the center. To the right is a search bar and a language dropdown menu. Below this, there are two main navigation buttons: "Volunteer" (with sub-links "Download", "Help", "Documentation") and "Computing power" (with sub-links "Top 100 volunteers", "Statistics").

Under the "Volunteer" button, there is a paragraph: "Use the idle time on your computer (Windows, Mac, or Linux) to cure diseases, study global warming, discover pulsars, and do many other types of scientific research. It's safe, secure, and easy:" followed by a numbered list: 1. Choose projects, 2. Download and run BOINC software, 3. Enter an email address and password.

Under the "Computing power" button, there is a summary: "Active: 288,709 volunteers, 508,580 computers. 24-hour average: 5,186.04 TeraFLOPS. BarryAZ is contributing 7,445 GFLOPS. Country: United States; Team: BOINC synergy".

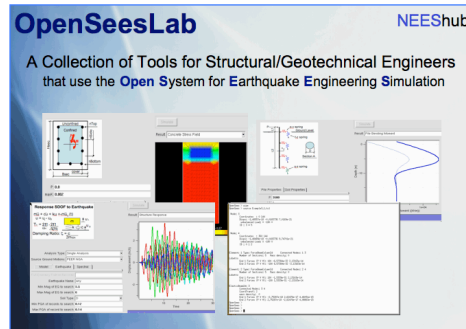
Below the navigation is the "Open Science Grid" logo and the text "A national, distributed computing partnership for data-intensive".

The main content area features a "Status Map" with tabs for "Jobs", "CPU Hours", "Transfers", "TB Transferred", and "Status Map". The map shows a world map with numerous colored pins indicating volunteer sites. A sidebar on the right provides summary statistics:

- OSG delivered across 96 sites
- In the last 24 Hours:** 742,000 Jobs, 1,886,000 CPU Hours, 1,784,000 Transfers, 783 TB Transferred
- In the last 30 Days:** 19,151,000 Jobs, 48,259,000 CPU Hours, 46,077,000 Transfers, 20,095 TB Transferred
- In the last Year:** 192,728,000 Jobs, 453,370,000 CPU Hours, 560,326,000 Transfers, 285,161 TB Transferred

# The OpenSeesLab tool:

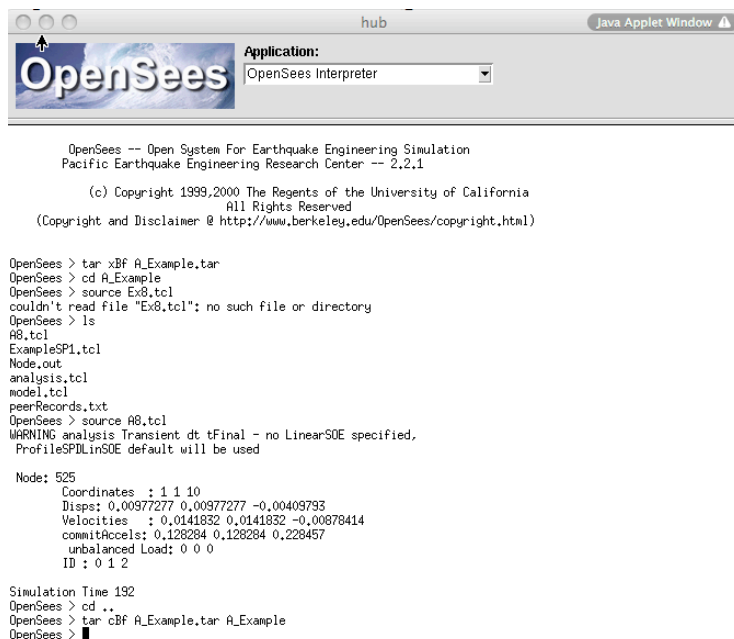
<http://nees.org/resources/tools/openseeslab>



Is a suite of Simulation Tools powered by OpnSees for:

1. Submitting OpenSees scripts (input files) to HUB resources
2. Educating students and practicing engineers
3. Performing useful tasks

## OpenSees Interpreter Tool



# OpenSees Parameter Study Tool

**Parameter Study Submission Tool**

This tool can be used to perform parameter studies with OpenSees. For each parameter, the user inputs the parameter name and the file containing all the values to be tried for that parameter. The user also specifies a main file to run in which the parameter value is used. DO NOT ASSIGN THE VALUE IN THE MAIN SCRIPT, otherwise you will overwrite the value to be used from the parameter file.

**example:**  
 If numParameter is set to 2, and the main script contains the following 3 lines:  
 set sum [expr \$varA + \$varB]  
 set fileOut [open \$varA\$varB.out w]  
 puts \$fileOut "\$a + \$b = \$sum"

Then, if in the Parameter1 box, we set the name to be varA and the associated file has the number 1 and 2. And if in the Parameter 2 box, we set the name to be varB and the associated file has the number 3, 4 and 5, the output directory when we hit the submit button will contain 6 files 13.out,14.out,15.out,23.out, 24.out and 25.out, each with a different message.

Be careful, make sure the results go to different filename, when using recorders use for example recorder Node -file node\$varA\$varB.out ....

Also only use the OSG (Open Science Grid) option when you have large models to run as it can take from minutes to hours for your job to actually start. The OSG option will place each run in a separate directory. Complain if you don't like this!

NOTE: all sourced files, ground motions, etc. must be in the main script or subscripts directory.

**WARNING:** Don't use a file in home directory as main script

# File Transfer Tool

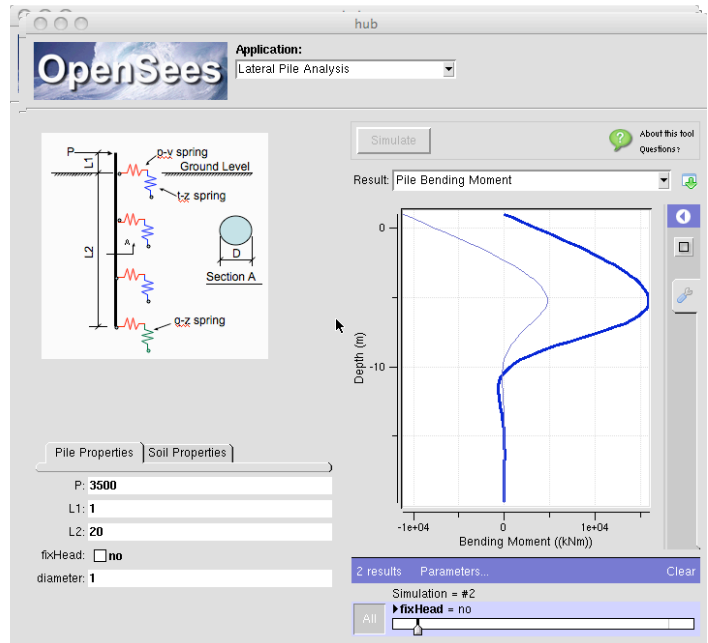
**Upload:** From Your Machine to NEEShub

**Download:** From NEEShub to Your Machine

There is also the **SynchoNEES** tool. An application that is Downloaded to your computer & allows you to quickly move Files & directories by moving them between folders.

# Lateral Pile Analysis

[http://opensees.berkeley.edu/wiki/index.php/Laterally-Loaded\\_Pile\\_Foundation](http://opensees.berkeley.edu/wiki/index.php/Laterally-Loaded_Pile_Foundation)  
Chris McGann U. Washington



## NEEShub things to know:

1. Anyone can get an account (it's free!)
2. You can have 5 sessions running at once
3. The sessions stay alive until you kill them
4. With each session you get a new data directory, some tools by default will store their information there.
5. You have 1GB storage by default (it is expandable to 10Gb!)

*Any Questions?*