Introduction to OpenSees
Parallel Classes and Applications

Frank McKenna
UC Berkeley

OpenSees Parallel Workshop
Berkeley, CA
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.
• The OpenSees framework provides classes for the Actor programming model.
The Actor Model

• It is a parallel version of the object-oriented model.
• In response to an incoming message, Actors can perform local work, create more Actors to perform the work, and can send return messages.
• A Shadow object represents a remote actor object locally, methods from the local process intended for a remote Actor go through the local Shadow object.

• It does not preclude message passing or thread programming within the objects.
Classes for Parallel Processing

- The OpenSees framework was designed for parallel and distributed processing from the outset.
- Classes are provided for the Actor Model.
- This does not rule out the use of message passing or threads within the objects of an Actor.
Domain Classes

- **Domain**
  - **Element**
    - NonlinearBeamColumn
    - BeamWithHinges
    - Quad (std, bbar)
    - Brick (std, bbar)
    - Shell
  - Subdomain
  - GraphPartitioner
    - Metis
  - DomainPartitioner
  - LoadBalancer

- **PartitionedDomain**

- **MovableObject**
  - sendSelf(Channel, ..)
  - recvSelf(Channel, ..)

- **TimeSeries**
- **LoadPattern**
- **SP_Constraint**
- **MP_Constraint**
- **ElementalLoad**
- **NodalLoad**
- **SP_Constraint**

- **DomainClasses**
Analysis Classes

- Analysis
  - StaticAnalysis
  - TransientAnalysis
  - DomainDecompAnalysis
    - StaticDDAnalysis
    - TransientDDAnalysis

- MovableObject
  - sendSelf(Channel, ..)
  - recvSelf(Channel, ..)

- AnalysisModel
  - SolnAlgorithm
    - EquiSolnAlgo
      - Linear
      - NewtonRaphson
      - ModifiedNewton
      - Broyden
      - BFGS
      - KrylovNewton

- Integrator
- SystemOfEqn

- CHandler
- Numberer
- ParallelNumberer

- Solver
  - BandGeneral
  - BandSPD
  - ProfileSPD
  - SparseGeneral
  - SparseSymmetric
  - DistributedSparse
  - Lapack(Gen, Band, ..)
  - ProfileSPD
  - SuperLU
  - Umfpack
  - SparseSym
  - DistributedSuperLU
  - Mumps
Example Parallel Applications
Parallel OpenSees Interpreters

- Two Interpreters have been created for users:
  - 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

Single interpreter running on P0 interpreting the input file

Actor objects sitting on other processes waiting to receive instructions
Example Usage:
Humboldt Bay Bridge Model
OpenSeesMP: An application for Large Models and Parameter Studies

Each process is running an interpreter and can determine its 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
```
Steel Building Study

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

```tcl
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;
```
Using the OpenSees Interpreter on Parallel Computers

Frank McKenna ¹
Gregory L. Fenves ¹

¹University of California, Berkeley
Any Questions?