PyTrilinos: Isorropia & Importers/Exporters

ME 4953/5013 - Introduction to High-Performance Computing

Isorropia

- Epetra interface to Zoltan
- Primary use is for load-balancing
  - Partitioning methods: block, cyclic, random, rcb, rib, hsfc, graph, hypergraph
- Uses Teuchos ParameterLists for input parameters
- Other features: Coloring, Ordering

Isorropia Vector Load-Balance Example

vectorLoadBalance.py

```
#!/usr/bin/env python
from PyTrilinos import Epetra
from PyTrilinos import Isorropia
from PyTrilinos import Teuchos

comm = Epetra.PyComm()
#Create unbalanced vector
global_vector_length = 10
if comm.MyPID() == 0: my_vector_length = 10
else: my_vector_length = 0

std_map = Epetra.Map(global_vector_length, my_vector_length, 0, comm)
x = Epetra.Vector(std_map)
x.Random()
#Print the unbalanced vector
if comm.MyPID() == 0: print "The unbalanced vector is: "
print x
#Create partitioner and redistribute
param_list = Teuchos.ParameterList()
param_list.set("Partitioning Method","RCB")
partitioner = Isorropia.Epetra.Partitioner(x, param_list)
redistributor = Isorropia.Epetra.Redistributor(partitioner)
x_balanced = redistributor.redistribute(x)
#Print the balanced vector
if comm.MyPID() == 0: print "The balanced vector is: "
print Epetra.Vector(Epetra.View, x_balanced)
```

Isorropia Matrix Load-Balance Example

matrixLoadBalance.py

```
#!/usr/bin/env python
from PyTrilinos import Epetra
from PyTrilinos import Isorropia
from PyTrilinos import Teuchos

comm = Epetra.PyComm()

global_num_rows = 9
if comm.MyPID() == 0: my_rows = global_num_rows
else: my_rows = 0
#Create and fill CrsMatrix
std_map = Epetra.Map(global_num_rows, my_rows, 0, comm)
A = Epetra.CrsMatrix(Epetra.Copy, std_map, 3)
for gid in std_map.MyGlobalElements():
    if gid in (0,global_num_rows-1):
        A.InsertGlobalValues(gid,[1],[gid])
    else:
        A.InsertGlobalValues(gid,[-1,2,-1],[gid-1,gid,gid+1])
A.FillComplete()
#Create partitioner and redistribute
param_list = Teuchos.ParameterList()
param_list.set("Partitioning Method","Hypergraph")
partitioner = Isorropia.Epetra.Partitioner(A, param_list)
redistributor = Isorropia.Epetra.Redistributor(partitioner)
A_balanced = redistributor.redistribute(A)
print A_balanced
```

Epetra.Import

- Allows efficient transfer of objects built using one `Map` to a new object with a new `Map`.

Construction

```
importer = Epetra.Import(TargetMap,SourceMap)
```

- `TargetMap:` `Map` containing the global IDs from which data should be imported to each processor from `SourceMap`
- `SourceMap:` `Map` Map containing the global IDs that should be used for importing data
- `Epetra.Export` works in the opposite way, i.e. exports to each processor the `TargetMap` global IDs from the `SourceMap` global IDs

Epetra.Import Example

EpetraImport.py

```
#!/usr/bin/env python
import numpy as np
from PyTrilinos import Epetra

comm = Epetra.PyComm()
#Create unbalanced vector
global_vector_length = 10
if comm.MyPID() == 0: my_vector_length = 10
else: my_vector_length = 0

std_map = Epetra.Map(global_vector_length, my_vector_length, 0, comm)
balanced_map = Epetra.Map(global_vector_length, 0, comm)
#Create two vectors
x = Epetra.Vector(std_map, np.arange(10))
y = Epetra.Vector(balanced_map)
#Print the unbalanced vector
if comm.MyPID() == 0: print "The unbalanced vector, x is: "
print x
comm.Barrier()
if comm.MyPID() == 0: print "The balanced vector, y is: "
print y
comm.Barrier()
#Create importer
importer = Epetra.Import(balanced_map, std_map)
#Import data from x to y
y.Import(x, importer, Epetra.Insert)
#Print the balanced vector
if comm.MyPID() == 0: print "The balanced vector, y is now: "
print y
```

Epetra.Import Example In Practice

EpetraImport2.py

```
#!/usr/bin/env python
from PyTrilinos import Epetra
from PyTrilinos import Isorropia
from PyTrilinos import Teuchos

comm = Epetra.PyComm()

global_num_rows = 9
if comm.MyPID() == 0: my_rows = global_num_rows
else: my_rows = 0
#Create and fill CrsMatrix
std_map = Epetra.Map(global_num_rows, my_rows, 0, comm)
A = Epetra.CrsMatrix(Epetra.Copy, std_map, 3)
for gid in std_map.MyGlobalElements():
    if gid in (0,global_num_rows-1):
        A.InsertGlobalValues(gid,[1],[gid])
    else:
        A.InsertGlobalValues(gid,[-1,2,-1],[gid-1,gid,gid+1])
A.FillComplete()
x = Epetra.Vector(std_map)
x.Random()
#Create partitioner and redistribute
param_list = Teuchos.ParameterList()
param_list.set("Partitioning Method","Hypergraph")
partitioner = Isorropia.Epetra.Partitioner(A, param_list)
redistributor = Isorropia.Epetra.Redistributor(partitioner)
A_balanced = redistributor.redistribute(A)
y = Epetra.Vector(A_balanced.Map())
#Create importer
importer = Epetra.Import(A_balanced.Map(), std_map)
#Import data from x to y
y.Import(x, importer, Epetra.Insert)
#Print the balanced vector
if comm.MyPID() == 0: print "The balanced vector, y is: "
print y
```