

# Flexible, Scalable Mesh and Data Management using PETSc DMPLex

M. Lange<sup>1</sup> M. Knepley<sup>2</sup> L. Mitchell<sup>3</sup> G. Gorman<sup>1</sup>

<sup>1</sup>AMCG, Imperial College London

<sup>2</sup>Computation Institute, University of Chicago

<sup>3</sup>Computing, Imperial College London

June 16, 2015

Motivation

Unstructured Mesh Management

Parallel Mesh Distribution

Firedrake

Summary

## Motivation

### Mesh management

- ▶ Many tasks are common across applications:  
Mesh input, partitioning, checkpointing, ...
- ▶ File I/O can become severe bottleneck!

### Mesh file formats

- ▶ Range of mesh generators and formats  
Gmsh, Cubit, Triangle, ExodusII, Fluent, CGNS, ...
- ▶ No universally accepted format
  - ▶ Applications often “roll their own”
  - ▶ No interoperability between codes

## Motivation

### Interoperability and extensibility

- ▶ Abstract mesh topology interface
  - ▶ Provided by a widely used library<sup>1</sup>
  - ▶ Extensible support for multiple formats
  - ▶ Single point for extension and optimisation
  - ▶ Many applications inherit capabilities
  
- ▶ Mesh management optimisations
  - ▶ Scalable read/write routines
  - ▶ Parallel partitioning and load-balancing
  - ▶ Mesh renumbering techniques
  - ▶ *Unstructured mesh adaptivity*

### Finding the right level of abstraction

---

<sup>1</sup>J. Brown, M. Knepley, and B. Smith. Run-time extensibility and librarization of simulation software. *IEEE Computing in Science and Engineering*, 2015

Motivation

Unstructured Mesh Management

Parallel Mesh Distribution

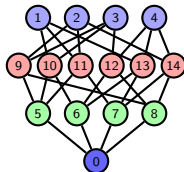
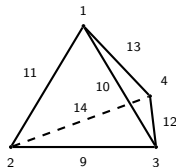
Firedrake

Summary

## Unstructured Mesh Management

### DMPlex - Unstructured mesh API<sup>1</sup>

- ▶ Abstract mesh connectivity
  - ▶ Directed Acyclic Graph (DAG)<sup>2</sup>
  - ▶ Dimensionless access
  - ▶ Topology separate from discretisation
  - ▶ Preallocation and preconditioners
- ▶ PetscSection
  - ▶ Describes irregular data arrays (CSR)
  - ▶ Mapping DAG points to DoFs
- ▶ PetscSF: Star Forest<sup>3</sup>
  - ▶ One-sided description of shared data
  - ▶ Performs sparse data communication



<sup>1</sup>M. Knepley and D. Karpeev. Mesh Algorithms for PDE with Sieve I: Mesh Distribution. *Sci. Program.*, 17(3):215–230, August 2009

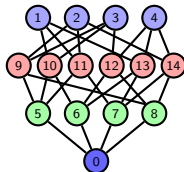
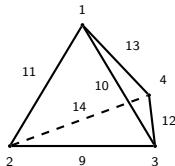
<sup>2</sup>A. Logg. Efficient representation of computational meshes. *Int. Journal of Computational Science and Engineering*, 4:283–295, 2009

<sup>3</sup>Jed Brown. Star forests as a parallel communication model, 2011

## Unstructured Mesh Management

### DMPlex - PETSc's unstructured mesh API<sup>1</sup>

- ▶ Input: ExodusII, Gmsh, CGNS, Fluent-CAS, MED, ...
- ▶ Output: HDF5 + Xdmf
  - ▶ Visualizable checkpoints
- ▶ Parallel distribution
  - ▶ Partitioners: Chaco, Metis/ParMetis
  - ▶ Automated halo exchange via PetscSF
- ▶ Mesh renumbering
  - ▶ Reverse Cuthill-McGee (RCM)



<sup>1</sup>M. Knepley and D. Karpeev. Mesh Algorithms for PDE with Sieve I: Mesh Distribution. *Sci. Program.*, 17(3):215–230, August 2009

Motivation

Unstructured Mesh Management

Parallel Mesh Distribution

Firedrake

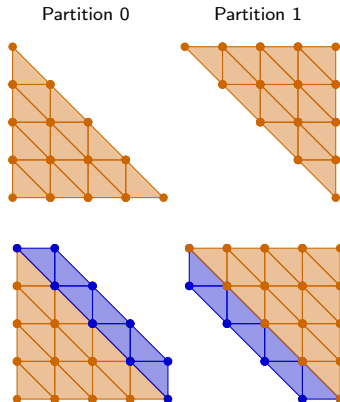
Summary



## Parallel Mesh Distribution

### DMPLexDistribute<sup>1</sup>

- ▶ Mesh partitioning
  - ▶ Topology-based partitioning
  - ▶ Metis/ParMetis, Chaco
- ▶ Mesh and data migration
  - ▶ One-to-all and all-to-all
  - ▶ Data migration via SF
- ▶ Parallel overlap computation
  - ▶ Generic N-level point overlap
  - ▶ FVM and FEM adjacency



<sup>1</sup>M. Knepley, M. Lange, and G. Gorman. Unstructured overlapping mesh distribution in parallel. *Submitted to ACM TOMS*, 2015

## Parallel Mesh Distribution

```
def DMPlexDistribute(dm, overlap):  
  
    # Derive migration pattern from partition  
    DMLabel partition = PetscPartition(partitioner, dm)  
    PetscSF migration = PartitionLabelCreateSF(dm, partition)  
  
    # Initial non-overlapping migration  
    DM dmParallel = DMPlexMigrate(dm, migration)  
    PetscSF shared = DMPlexCreatePointSF(dmParallel, migration)  
  
    # Parallel overlap generation  
    DMLabel overlap = DMPlexCreateOverlap(dmParallel, N, shared)  
    PetscSF migration = PartitionLabelCreateSF(dm, overlap)  
    DM dmOverlap = DMPlexMigrate(dm, migration)
```

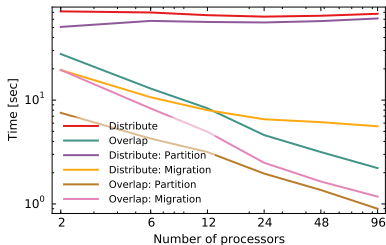
- ▶ Two-phase distribution enables parallel overlap generation

## Parallel Mesh Distribution

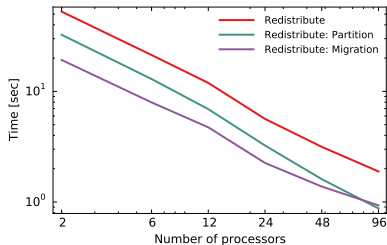
Strong scaling performance:

- ▶ Cray XE30 with 4920 nodes;  $2 \times 12$ -core E5-2697 @2.7GHz<sup>1</sup>
- ▶  $128^3$  unit cube with  $\approx 12$  mio. cells

One-to-all distribution



All-to-all redistribution



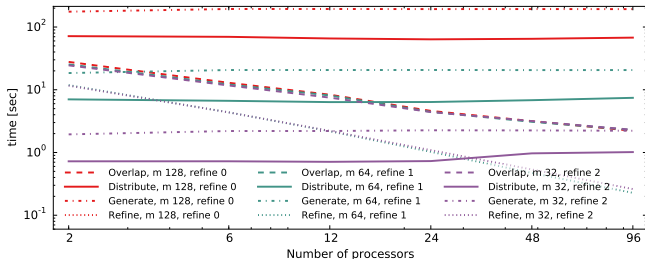
- ▶ Remaining bottleneck: Sequential partitioning

<sup>1</sup>ARCHER: [www.archer.ac.uk](http://www.archer.ac.uk)

## Parallel Mesh Distribution

Regular parallel refinement:

- ▶ Distribute coarse mesh and refine in parallel
- ▶ Generate overlap on resulting fine mesh



Motivation

Unstructured Mesh Management

Parallel Mesh Distribution

Firedrake

Summary

## Firedrake

### Firedrake - Automated Finite Element computation<sup>1</sup>

- ▶ Re-envision FEniCS<sup>2</sup>

$$\begin{aligned}\phi^{n+1/2} &= \phi^n - \frac{\Delta t}{2} p^n \\ p^{n+1} &= p^n + \frac{\int_{\Omega} \nabla \phi^{n+1/2} \cdot \nabla v \, dx}{\int_{\Omega} v \, dx} \quad \forall v \in V \\ \phi^{n+1} &= \phi^{n+1/2} - \frac{\Delta t}{2} p^{n+1}\end{aligned}$$

where

$$\begin{aligned}\nabla \phi \cdot n &= 0 \text{ on } \Gamma_N \\ p &= \sin(10\pi t) \text{ on } \Gamma_D\end{aligned}$$

```
from firedrake import *
mesh = Mesh("wave_tank.msh")
V = FunctionSpace(mesh, 'Lagrange', 1)
p = Function(V, name="p")
phi = Function(V, name="phi")
u = TrialFunction(V)
v = TestFunction(V)
p_in = Constant(0.0)
bc = DirichletBC(V, p_in, 1)
T = 10.
dt = 0.001
t = 0
while t <= T:
    p_in.assign(sin(2*pi*5*t))
    phi -= dt / 2 * p
    p += assemble(dt * inner(grad(v), grad(phi))*dx) \
        / assemble(v*dx)
    bc.apply(p)
    phi -= dt / 2 * p
    t += dt
```

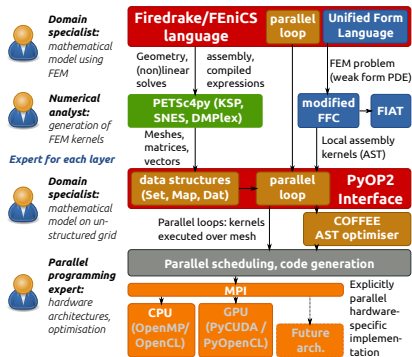
<sup>1</sup>F. Rathgeber, D. Ham, L. Mitchell, M. Lange, F. Luporini, A. McRae, G. Bercea, G. Markall, and P. Kelly. Firedrake: Automating the finite element method by composing abstractions. *Submitted to ACM TOMS*, 2015

<sup>2</sup>A. Logg, K.-A. Mardal, and G. Wells. *Automated Solution of Differential Equations by the Finite Element Method*. Springer, 2012

## Firedrake

### Firedrake - Automated Finite Element computation

- ▶ Implements UFL<sup>1</sup>
  - ▶ Outer framework in Python
  - ▶ Run-time C code generation
  - ▶ PyOP2: Assembly kernel execution framework
  
- ▶ Domain topology from DMPlex
  - ▶ Mesh generation and file I/O
  - ▶ Derive discretisation-specific mappings at run-time
  - ▶ Geometric Multigrid

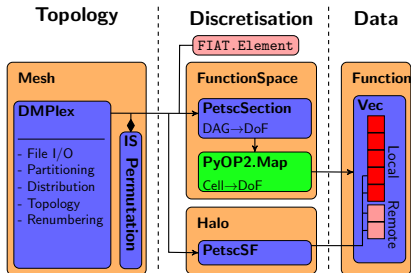


<sup>1</sup>M. Alnæs, A. Logg, K. Ølgaard, M. Rognes, and G. Wells. Unified Form Language: A domain-specific language for weak formulations of partial differential equations. *ACM Transactions on Mathematical Software (TOMS)*, 40(2):9, 2014

## Firedrake

### Firedrake - Data structures

- ▶ DMPlex encodes topology
  - ▶ Parallel distribution
  - ▶ Application ordering
  
- ▶ Section encodes discretisation
  - ▶ Maps DAG to solution DoFs
  - ▶ Generated via FIAT element<sup>1</sup>
  - ▶ Derives PyOP2 indirection maps for assembly
  
- ▶ SF performs halo exchange
  - ▶ DMPlex derives SF from section and overlap



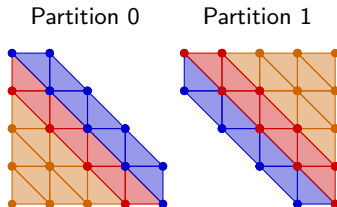
<sup>1</sup>R. Kirby. FIAT, A new paradigm for computing finite element basis functions. *ACM Transactions on Mathematical Software (TOMS)*, 30(4):502–516, 2004



## Firedrake

### PyOP2 - Kernel execution

- ▶ Run-time code generation
  - ▶ Intermediate representation
  - ▶ Kernel optimisation via AST<sup>1</sup>
- ▶ Overlapping communication
  - ▶ Core: Execute immediately
  - ▶ Non-core: Halo-dependent
  - ▶ Halo: Communicate while computing over core
- ▶ Imposes ordering constraint

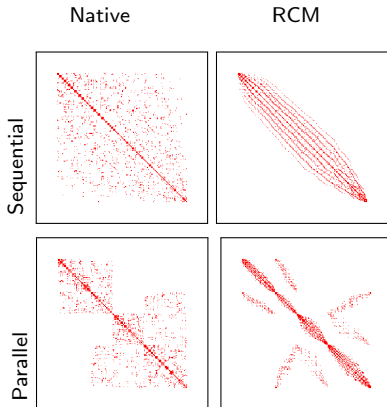


<sup>1</sup>F. Luporini, A. Varbanescu, F. Rathgeber, G.-T. Bercea, J. Ramanujam, D. Ham, and P. Kelly. Cross-Loop Optimization of Arithmetic Intensity for Finite Element Local Assembly. *Accepted for publication, ACM Transactions on Architecture and Code Optimization*, 2015

## Firedrake

### Firedrake - RCM reordering

- ▶ Mesh renumbering
  - ▶ Improves cache coherency
  - ▶ Reverse Cuthill-McKee (RCM)
- ▶ Combine RCM with PyOP2 ordering<sup>1</sup>
  - ▶ Filter cell reordering
  - ▶ Apply *within* PyOP2 classes
  - ▶ Add DoFs per cell (closure)



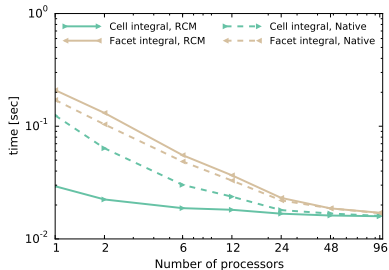
<sup>1</sup>M. Lange, L. Mitchell, M. Knepley, and G. Gorman. Efficient mesh management in Firedrake using PETSc-DMPlex. *Submitted to SISC Special Issue*, 2015

## Firedrake Performance

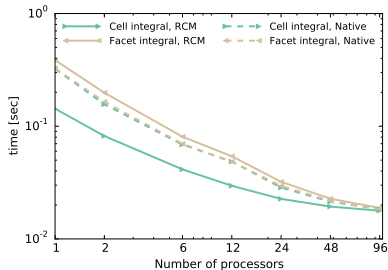
Indirection cost of assembly loops:

- ▶ Cell integral:  $L = u * dx$
- ▶ Facet integral:  $L = u('+') * dS$

100 loops on  $P_1$



100 loops on  $P_3$

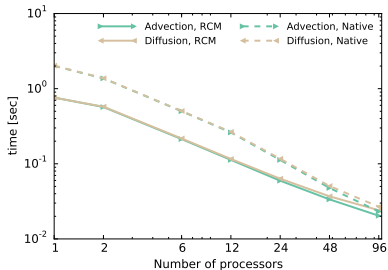


## Firedrake Performance

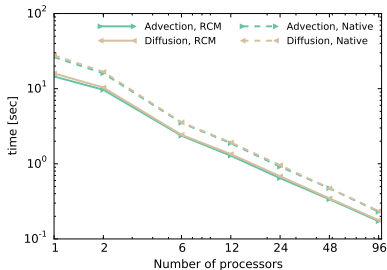
Advection-diffusion:

- ▶ Eq:  $\frac{\partial c}{\partial t} + \nabla \cdot (\bar{u}c) = \nabla \cdot (\bar{\kappa} \nabla c)$
- ▶ L-shaped mesh with  $\approx 3.1$  mio. cells

Matrix assembly on  $P_1$



Matrix assembly on  $P_3$

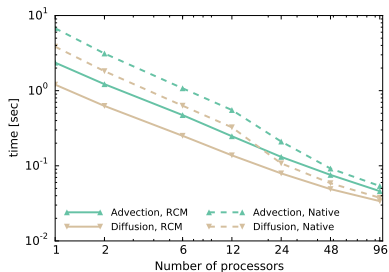


## Firedrake Performance

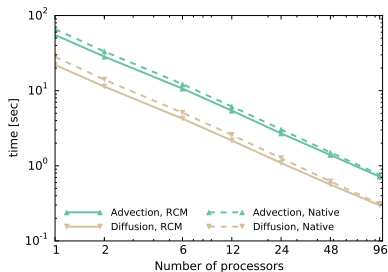
Advection-diffusion:

- ▶ Eq:  $\frac{\partial c}{\partial t} + \nabla \cdot (\bar{u}c) = \nabla \cdot (\bar{\kappa} \nabla c)$
- ▶ L-shaped mesh with  $\approx 3.1$  mio. cells

RHS assembly on  $P_1$



RHS assembly on  $P_3$

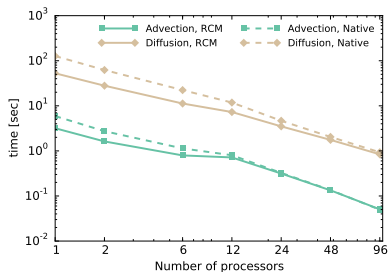


## Firedrake Performance

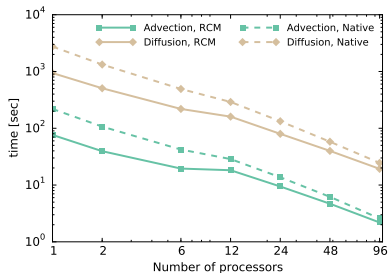
Advection-diffusion:

- ▶ Advection solver: CG + Jacobi
- ▶ Diffusion solver: CG + HYPRE BoomerAMG

Solve on  $P_1$



Solve on  $P_3$



Motivation

Unstructured Mesh Management

Parallel Mesh Distribution

Firedrake

Summary

## Summary

### DMPlex mesh management

- ▶ Unified mesh reader/writer interface
- ▶ Improved DMPlexDistribute
  - ▶ Scalable overlap generation
  - ▶ All-to-all load balancing
- ▶ Firedrake FE environment
  - ▶ DMPlex as topology abstraction
  - ▶ Derive indirection maps for assembly
  - ▶ Compact RCM renumbering

### Future work

- ▶ Parallel mesh file reads
- ▶ Anisotropic mesh adaptivity



Thank You



[www.firedrakeproject.org](http://www.firedrakeproject.org)



[www.fenicsproject.org](http://www.fenicsproject.org)



FEniCS '15

29 June - 1 July 2015  
Imperial College London

