KTH ROYAL INSTITUTE OF TECHNOLOGY

Improving Strong Scalability Limits of Finite Element Based Solvers

NICLAS JANSSON

PDC Center for High Performance Computing

Introduction

To strongly scale finite element codes at exascale will require unreasonable large problem sizes to balance communication costs.

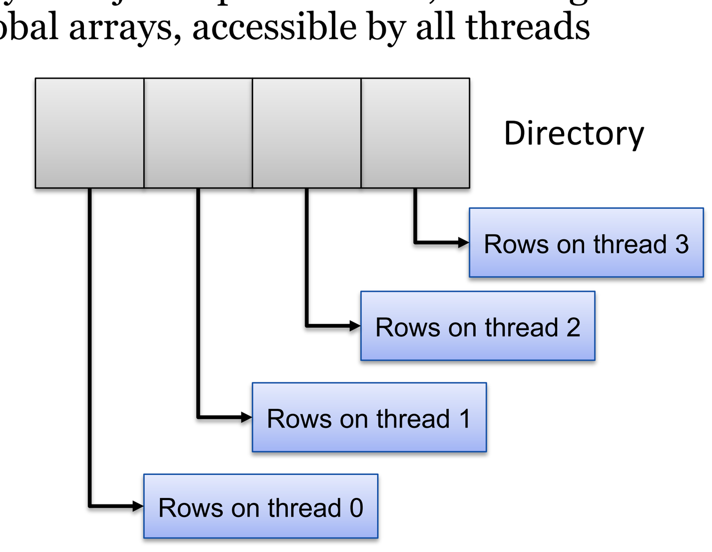
- Low order methods have too little work per element to balance communication costs
- One of the key bottlenecks is **sparse matrix assembly**
- Using two-sided message passing, the cost of message matching unavoidably increase **latency** and **synchronisation costs**

We address the strong scalability limits of finite element based solvers by changing to lightweight **one-sided communication** in performance critical, latency sensitive kernels.

PGAS Based Sparse Matrix Representation

A low latency one-sided PGAS based sparse matrix representation.

- Compressed Row Storage (CRS) like format optimised for random insertion during matrix assembly
- Row-wise data distribution
- Two phase assembly. Insertion of local entries + exchange of non local entries using remote memget operations.
- Implemented using Unified Parallel C (UPC)
- Directory of object representation, allowing for arbitrary sized global arrays, accessible by all threads



- Hybrid interface for combining with non PGAS languages. Access data through opaque interfaces

```
extern "C" {
  int jp_mat_init(char *A,
                  uint32_t m,
                  uint32_t n);

}

char A[200]; /* sizeof(jp_mat_t) */

jp_mat_init(A, M, N);
```

C++ side

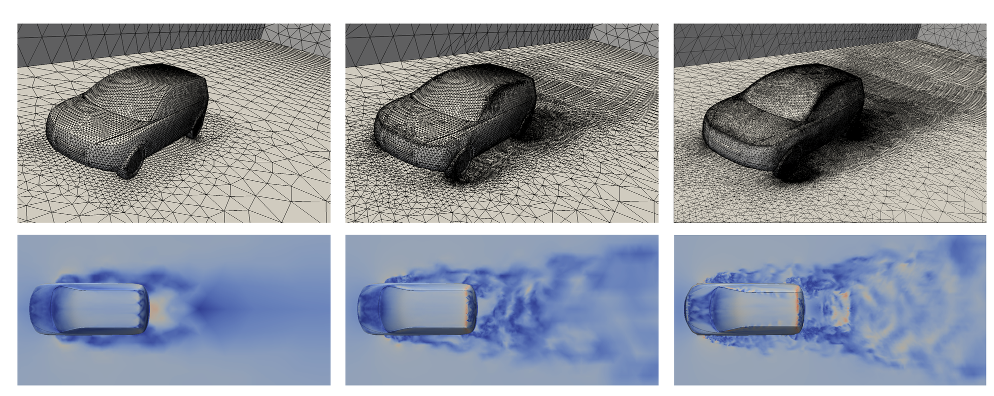
```
typedef struct {
 shared[] row_stack *shared *a_dir;
  row_stack *rs;
  ...

} jp_mat_t;

int jp_mat_init(jp_mat_t *restrict A,
                uint32_t m,
                uint32_t n);
```

UPC side

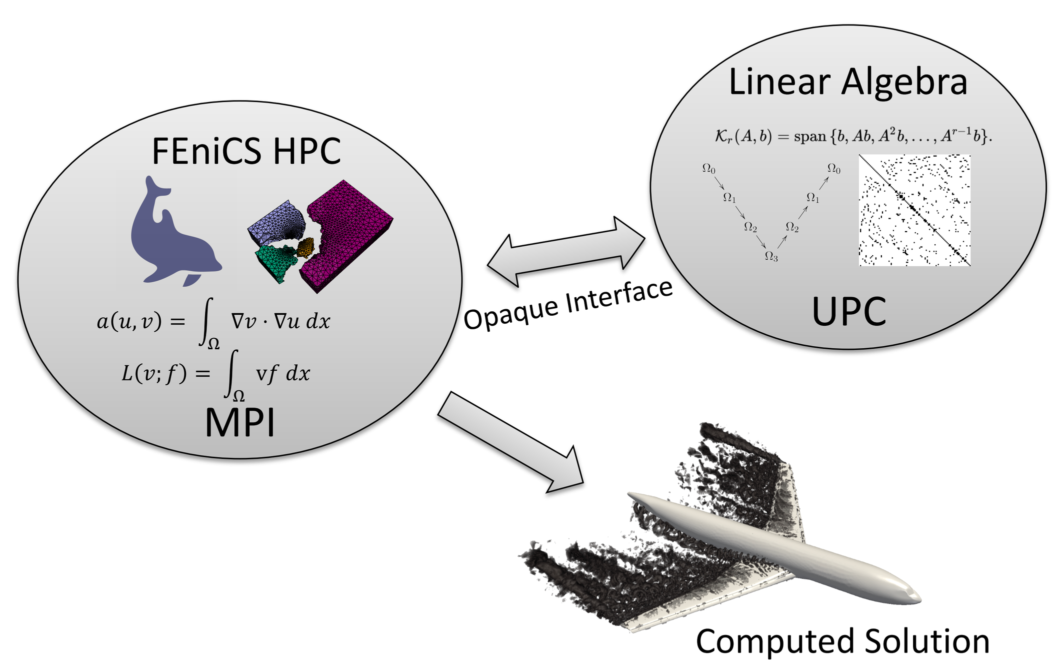
Implementation of Hybrid MPI/PGAS in FEniCS

New hybrid parallelisation of the automated problem solving environment FEniCS.

- FEniCS is written in C++ and parallelised using MPI
- Used to build adaptive solvers for turbulent fluid flows. Non-linear transient problems need to reassemble in each time-step



- Complex unstructured meshes with irregular communication pattern during matrix assembly
- High latency costs due to message matching, requires a large amount of elements per core to strongly scale
- **Too costly** to **rewrite** an entire application in PGAS
- **Replace** MPI based linear algebra backend with the new PGAS based backend



- New hybrid parallelisation of FEniCS combining MPI (FEM) with UPC (linear algebra)
- Drop-in replacement, possible to switch at runtime
- Completely transparent, end users and solvers based on FEniCS unaffected

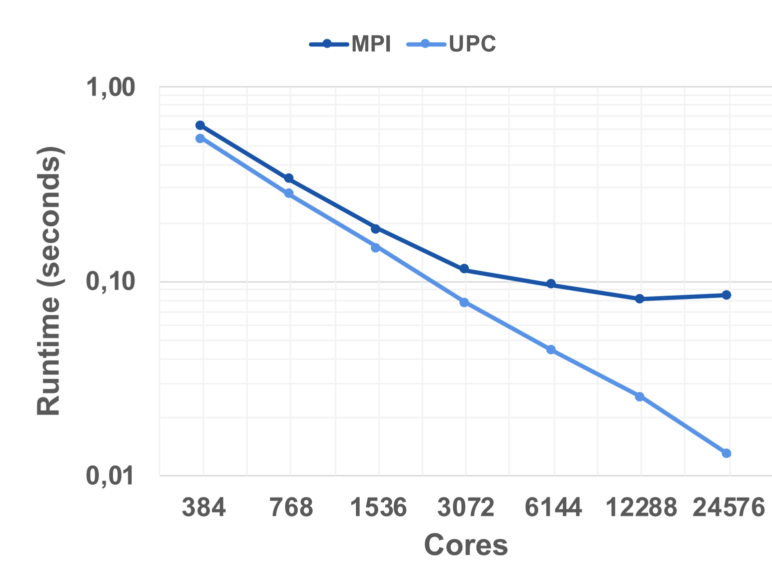
Performance Evaluation

Performance evaluated on two Cray XC40, the 3944 node Hornet at HLRS and the 1676 node Beskow at PDC, comparing

- Pure MPI, FEniCS using PETSc for linear algebra
- MPI + UPC, FEniCS using a PGAS based linear algebra backend

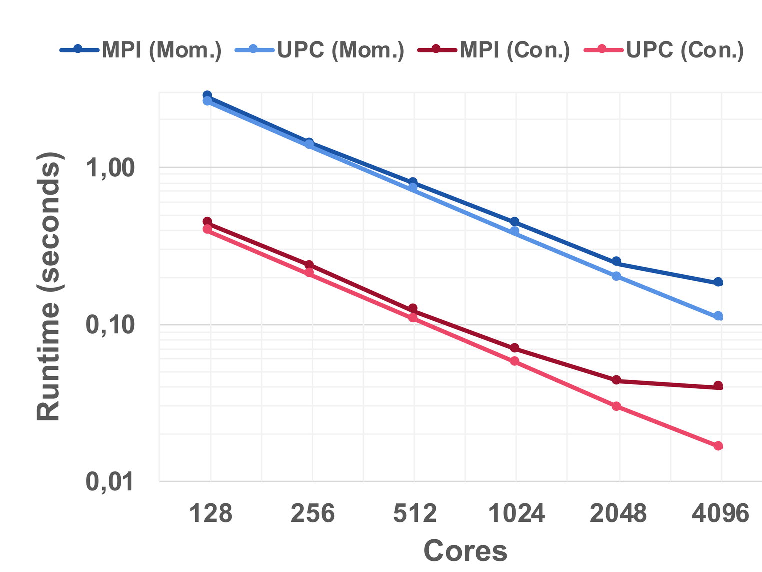
Synthetic Matrix Assembly

Matrix reassembly time for Laplace's equation in 3D on a unstructured tetrahedral mesh with 317M elements.



Incompressible Flow Solver

Reassembly times for the momentum and continuity equations in an implicit LES solver, on an unstructured tetrahedral mesh with 60M elements.



Conclusions

- Low latency, one-sided communication can improve scalability of sparse matrix assembly in finite element based solvers
- Hybrid MPI/PGAS offers a scalable alternative to a complete rewrite of legacy MPI codes when preparing for exascale

EXCELLERAT

Niclas Jansson
PDC Center for High Performance Computing
KTH Royal Institute of Technology
njansson@kth.se