Inria Project Lab C2S@Exa

Parallelizing the Traces software

Rachid El khaoulani El idrissi

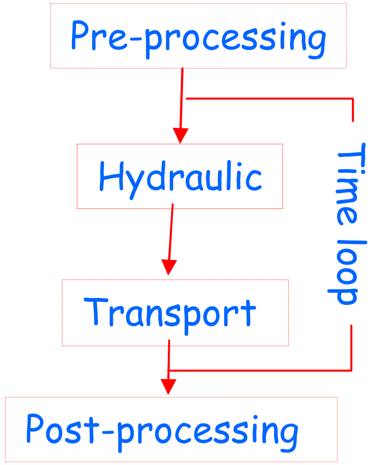
September 2013

Context & Motivation

Traces: numerical simulation of radio-active waste storage in profound geological layers

Two sorts of problem can be treated:

- Hydraulic: single-phase flow in porous media
- Transport: migration of radioactive waste in porous media



Large-scale problems in both points of view: spatial and temporal

- Long-term performance and safety assessment
- Large-size domains have to be dealt with

Parallelizing the Traces software

- To make more realistic and reliable studies
- To take advantage from computing capabilities

Outline

1. Hydraulic problem
2. Mesh partitioning
3. Parallel assembling and resolving
4. Numerical results
5. Conclusion

Outline

1. Hydraulic problem
2. Mesh partitioning
3. Parallel assembling and resolving
4. Numerical results
5. Conclusion

Hydraulic: Mathematical Model

$$\begin{cases} s\dfrac{\partial u}{\partial t} = -div(q) - s\lambda u + f & \text{Mass balance equation} \\ q = D.\nabla u & \text{Darcy's law} \end{cases}$$

Unknowns

u hydraulic charge

q filtration velocity

Parameters

D hydraulic conductivity

s storage coefficient

f source/sink term

λ a kinetic term

- Temporal discretization: implicit
- Spatial discretization: Mixed Hybrid Finite Element Method

→ Algebraic linear system whose unknowns are associated to the mesh faces

→ Parallel assembling and resolving of the resulting linear algebraic system is the most challenging part of the hydraulic problem

Parallelization strategy

Parallelization for coarse grain MIMD architectures with distributed memory

- Mesh partitioning
- Message passing programming trough MPI standard

Outline

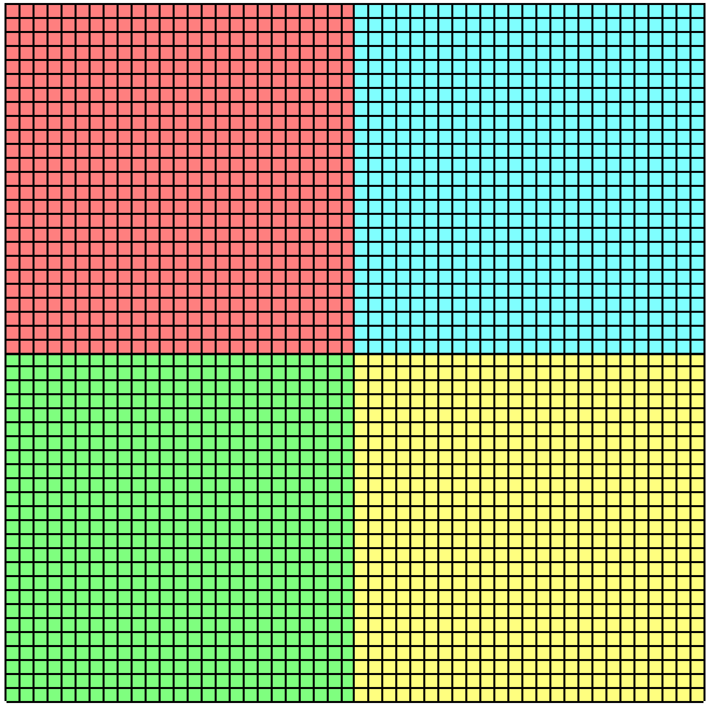
1. Hydraulic problem
2. Mesh partitioning
3. Parallel assembling and resolving
4. Numerical results
5. Conclusion

Unstructured mesh partitioning

Static, non-overlapping and homogeneous partitioning

Partitioning software: Metis, Scotch

Mapping of the mesh elements: divide mesh elements into groups of elements



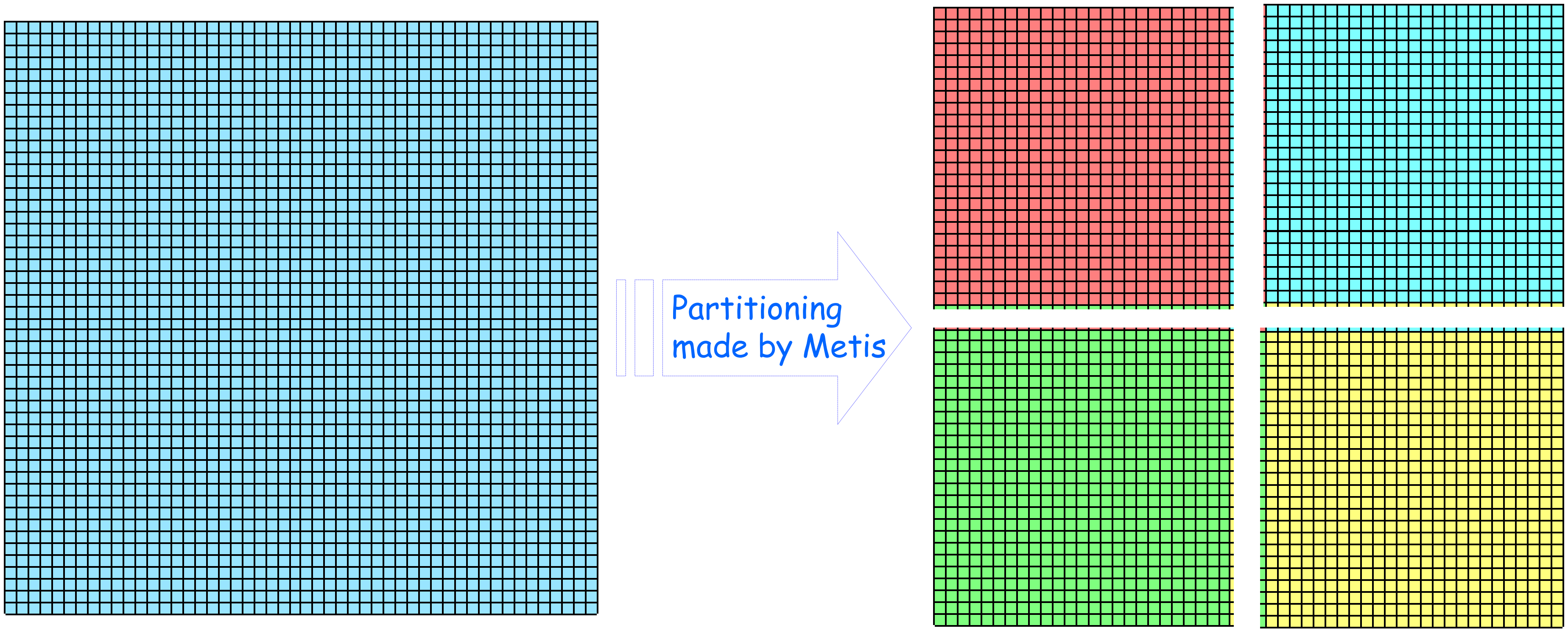
- Partitioning of mesh nodes
- Partitioning of mesh faces (edges in 2D)
- Neighboring relations between processors

→ New input file

→ Distributed data

Mesh partitioning

Non-overlapping homogeneous mesh partitioning



Communication lists

Communication list: list of the local numbers of the common faces in each couple of neighbors

[neighbor1, neighbor2] → [(neighbor1,list1) , (neighbor2,list2)]

Matched lists

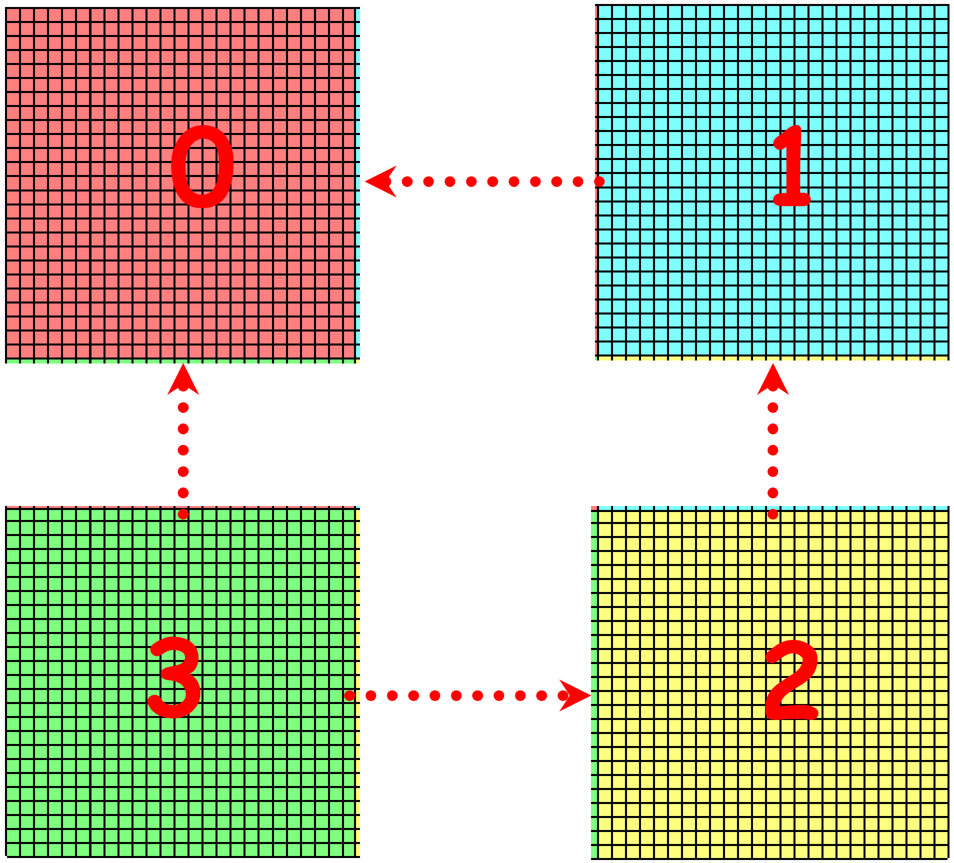
❖ Each processor builds communication lists with its neighbors

❖ Matching up: the order in the lists is imposed by the highly ranked processor of each couple of neighbors

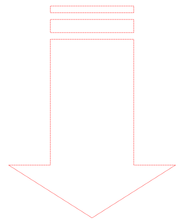
❖ Communication cost: each processor sends the common lists to its neighbors that have a lower rank than him.

Communication lists

Communication cost: each processor sends the common lists to its neighbors that have a lower rank than him



4 couple of neighbors



4 messages transferred

Transferring only 4 messages of integers to build matched communication lists

→ There are as many messages as the number of couples of neighbors

Outline

1. Hydraulic problem
2. Mesh partitioning
3. Parallel assembling and resolving
4. Numerical results
5. Conclusion

The Hypre librairy

What is Hypre?

Software library of high performance preconditioners and solvers for the solution of **large, sparse linear systems** on massively parallel computers

Krylov space solvers

- Symmetric system: Conjugate Gradient
- Asymmetric system: GMRES, Bi-Conjugate Gradient stabilized …

Preconditioners

Algebraic Multigrid, ILU(k), Block Jacobi ILU(k), Diagonal …

How to use Hypre

Linear-Algebraic System interface (IJ)

Hypre-Traces interfacing

Distributed data form

Matrices are assumed to be distributed across the processors by contiguous blocks of rows

$$A = \begin{pmatrix} A_0 \\ A_1 \\ . \\ . \\ A_{p-1} \end{pmatrix}$$

Hypre defines a new numbering of the DOF

The DOF 1 to n_0 reside in processor 0

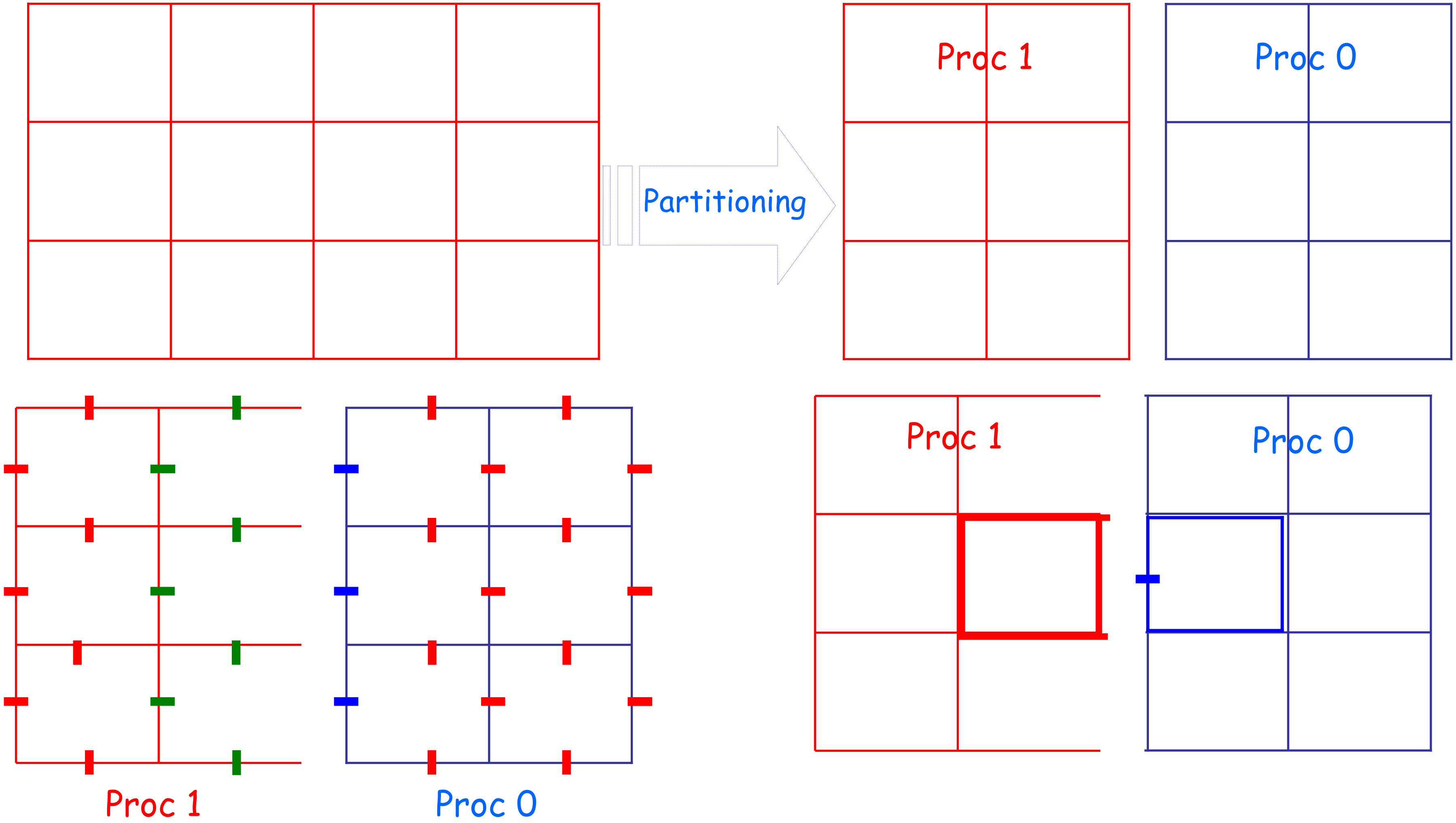
The DOF n_0+1 to n_1 reside in processor 1

The DOF $n_{k-1}+1$ to n_k reside in processor k

Main points

- Hypre defines its own numbering of the DOF
- Hypre requires a mapping of the DOF on the processors
- Processors define actual blocks of the system for Hypre independently

Matrix parallel assembling



For each DOF in common

- From processor 1 to processor 0: 4 Coefficients + 4 indices

→ Two messages are transferred

Matrix parallel assembling

Each processor computes its own FE matrix then transmits it for Hypre

$$\begin{pmatrix} \\ \beta_1 \;.\; \beta_2 \;.\; \beta_3 \;.\;.\; \beta \\ \\ \end{pmatrix}$$

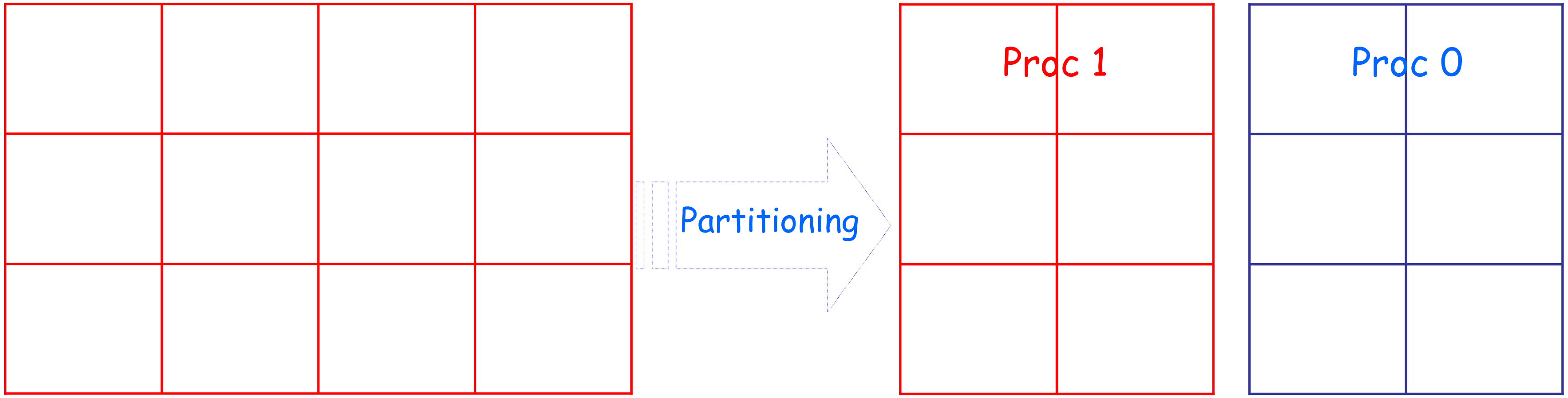
DOF
in common

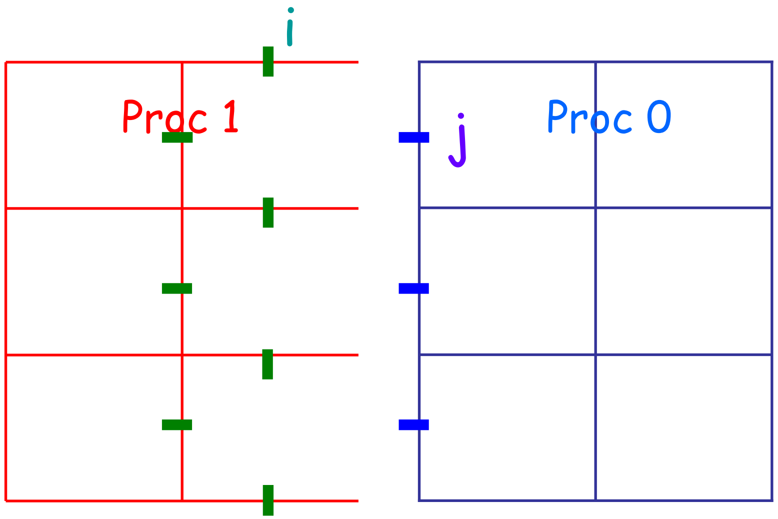
$$\begin{pmatrix} \\ \alpha_1 \;.\; \alpha_2 \;.\; \alpha_3 \;.\;.\; \alpha \\ \\ \end{pmatrix}$$

$$\begin{pmatrix} \\ \beta_1 \;.\; \beta_2 \;.\; \beta_3 \;.\;.\; \beta \\ \\ \end{pmatrix}$$

$$\begin{pmatrix} \\ \alpha_1 \;.\; \alpha_2 \;.\; \alpha_3 \;.\;.\; \alpha+\beta \;.\;.\; \beta_1 \;.\; \beta_2 \;.\; \beta_3 \\ \\ \end{pmatrix}$$

Matrix parallel assembling





Processor 1 needs the column number j from the processor 0 to put correctly the coefficient a_{ij} in the Hypre matrix

Processor 0 sends the Hypre numbering of the DOF in common to processor 1

→ One message of integers from processor 0 to processor 1

Hypre: solving

Parallel assembling and transmitting the RHS to Hypre

Define an initial guess of the solution …

Choice a solver and its parameters, preconditioner…

Get the solution from Hypre and adapt it to the Traces numbering

Outline

1. Hydraulic problem
2. Mesh partitioning
3. Parallel assembling and resolving
4. Numerical results
5. Conclusion

Performance: PCG/DS

Mesh: hexahedral, 1 685 600 elements, 5 114 900 faces

Solver: Preconditioned Conjugate gradient PCG, Convergence tolerance 1e-10

PCG / DS			
Number of processors	CPU Time(s) Solve+Setup	Speed-up	Number of iterations
1	4322		8305
2	2143	2,02	8305
4	1066	4,05	8305
8	539	8,02	8305
16	272	15,88	8305
20	218	19,85	8305

Validation of parallel interfacing of Traces-Hypre

Setup

- ❖ Parallel passing of the linear system to Hypre
- ❖ Hypre's setup

Performance: PCG/ILU(1)

Mesh: hexahedral, 1 685 600 elements, 5 114 900 faces

Solver: Preconditioned Conjugate gradient PCG, Convergence tolerance 1e-10

PCG/ILU(1)			
Number of processors	CPU Time(s) Solve+Setup	Speed-up	Number of iterations
1	2510		1668
2	1213	2,07	1677
4	622	4,03	1677
8	321	7,82	1682
16	190	13,2	1688
20	149	16,8	1689

Performance: PCG/AMG

Mesh: hexahedral, 1 685 600 elements, 5 114 900 faces

Solver: Preconditioned Conjugate gradient PCG, Convergence tolerance 1e-10

PCG / AMG			
Number of processors	CPU Time(s) Solve+Setup	Speed-up	Number of iterations
1	235		2
2	130	1,81	2
4	69,6	3,38	2
8	38,2	6,16	2
16	25,5	9,22	2
20	22,7	10,37	2

Performance: comparison

Mesh: hexahedral, 1 685 600 elements, 5 114 900 faces

Solver: Preconditioned Conjugate gradient PCG, Convergence tolerance 1e-10

	PCG / DS		PCG / ILU(1)		PCG / AMG	
Number of processors	CPU Time(s) Solve+Setup	Speed-up	CPU Time(s) Solve+Setup	Speed-up	CPU Time(s) Solve+Setup	Speed-up
1	4322		2510		235	
2	2143	2,02	1213	2,07	130	1,81
4	1066	4,05	622	4,03	70	3,38
8	539	8,02	321	7,82	38	6,16
16	272	15,88	190	13,19	26	9,22
20	218	19,85	149	16,83	23	10,37

PCG/AMG is less scalable than PCG/DS and PCG/ILU(1)

However

It is more efficient in bringing down the CPU Time Than the others

Outline

1. Hydraulic problem
2. Mesh partitioning
3. Parallel assembling and resolving
4. Numerical results
5. Conclusion

Conclusion

- Hydraulic problem has been parallelized using MPI and Hypre librairies

- Scotch and Metis were used to perform mesh partitioning

- Distributed data form

- Other Parallel solvers of linear systems can be easily interfaced with Traces software