Noname manuscript No.
(will be inserted by the editor)

The Multi-Stencil Language

Helene Coullon · Chirtsian Perez · Julien Bigot

the date of receipt and acceptance should be inserted later

Abstract Insert your abstract here.

Keywords First keyword · Second keyword · More

1 Introduction

Contributions :

– Formalism of a multi-stencil program and its parallelization
– The Multi-stencil Language: agnostic from the type of mesh and of the choosen implementation
– A first implementation of the MSL compiler (static scheduling + fusion + dump to SkelGIS/components)
– Evaluation of the different types of parallelism introduced + compiler + fusion

2 Computational model of Multi-Stencil Programs

Formalism :

– mesh / mesh entities and groups / computations domains
– data and scalars
– computations : stencil, local and reduction
– Multi-Stencil Program

Helene Coullon · Christian Perez
INRIA
46 Alle d'Italie
69007 Lyon
E-mail: helene.coullon@inria.fr

Julien Bigot
CEA

3 Parallelization of Multi-Stencil Programs

Parallelization formalism :

– Data parallelism (introduction of synchronizations)
– Task parallelism (computation of the dependency graph)

4 The Multi-Stencil Language

The parallel empty skeleton of the application can be computed automatically from the language :

– grammar
– example from the language to the dependency graph

5 Static Scheduling

From the dependency graph we study a first dump: static scheduling

– description of the algorithm to get the binary tree decomposition of a minimal serie-parallel graph
– use of the serie-parallel tree decomposition to detect fusions of kernels
– example (the same one) from the dependency graph to the binary tree

6 Evaluation

– short description of the implementation (compiler in Python, DDS = SkelGIS, Components)
– evaluation of the compiler (execution times of the different steps)
– evaluation of the data parallelism (weak/strong scaling)
– evaluation of the fusion
– evaluation of the hybrid parallelism
– analytic model for the results

7 Related work

– Lizst, Pochoir, PATUS for optimizations/parallelization of a single stencil kernel + mesh dependent
– Halide for pipeline of stencil computations = mesh dependent
– Reuse of external solutions as SkelGIS, GA, Lizst, Pochoir etc.

8 Conclusion

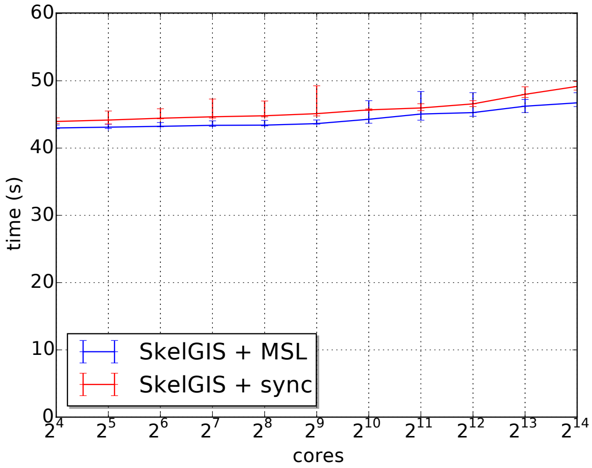

Fig. 1: weak-scaling 400x400 blocks.

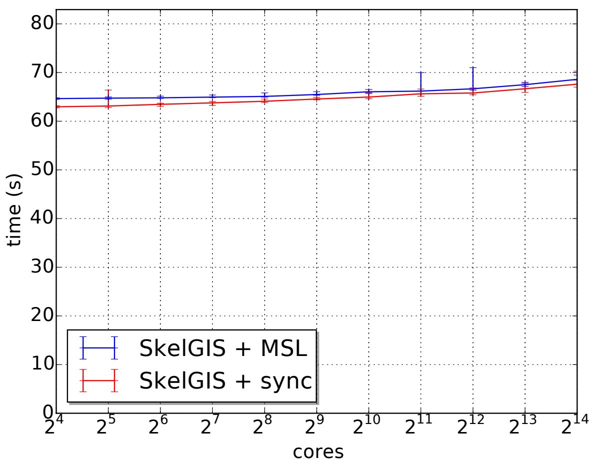

Fig. 2: weak-scaling 800x800 blocks

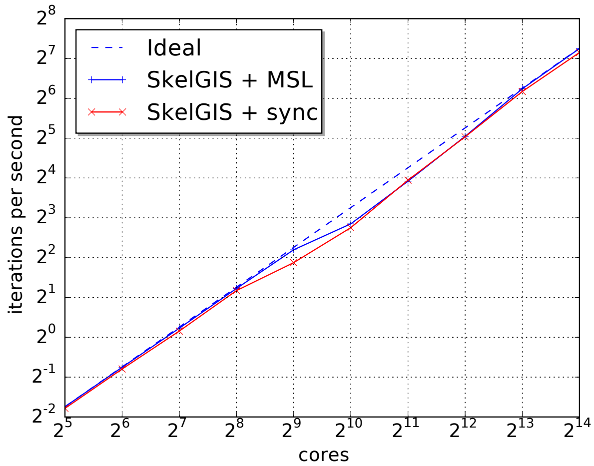

Fig. 3: strong-scaling 10kx10k, 1k iterations

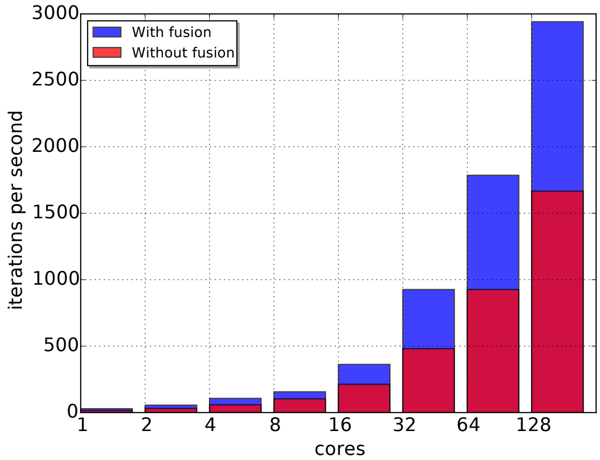

Fig. 4: strong scaling 500x500, 200 iterations, with and without fusion

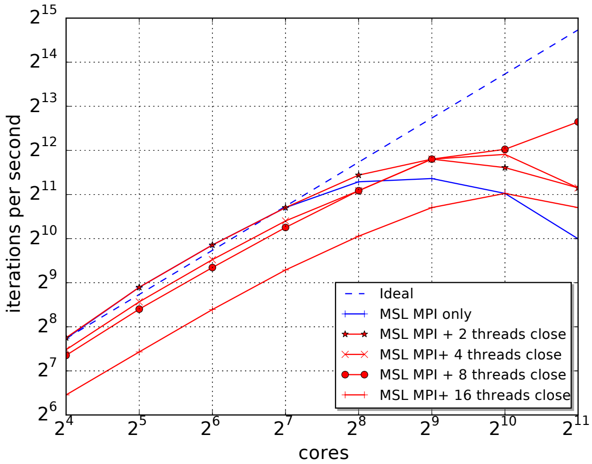

Fig. 5: MPI only vs MPI+threads with *close* scheduling policy (OpenMP)

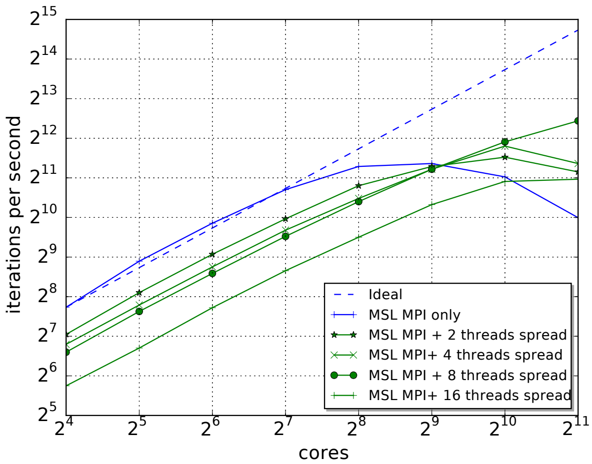

Fig. 6: MPI only vs MPI+threads with *spread* scheduling policy (OpenMP)

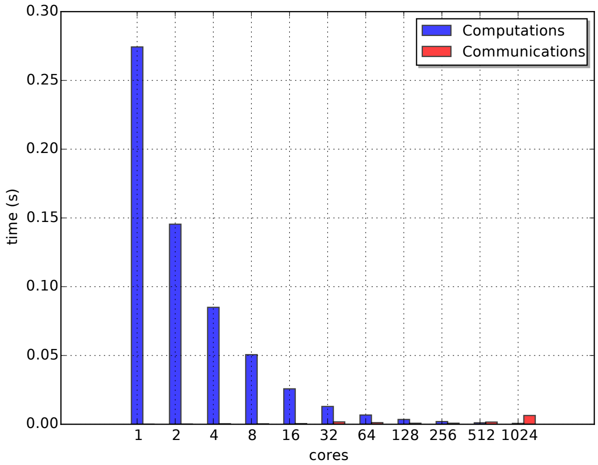

Fig. 7: Computation vs communication times in the MPI only application