# Multi-Stencil agnostic descriptive language

Hélène Coullon

INRIA

24 juillet 2015

# Table of contents

# Introduction

### What is a stencil ?

In a mesh-based simulation, it is an explicit numerical schemes involving a neighborhood of an element for its computation

### What are Multi-Stencil Porgrams ?

A real case numerical simulation is composed of

- more than one computation involving a neighborhood shape,
- sometimes more than one stencil shape,
- boundary condition computations,
- and additionnal local computations.

# Introduction

## What does produce the MS Language ?

Unlike other solutions MS Language gives

- an hybrid parallelization structure of the overall numerical simulation
  - data parallelism (domain decomposition),
  - and task parallelism to increase parallelism for large scale parallel machines
- the parallelized structure is made of component which increases code re-use, maintainability and portability

## What means agnostic ?

The language do not need any knowledge on

- the mesh type
- the neighborhood shape
- the underlying programming language used

# Introduction

### What means descriptive ?

The language is based on the textual description of the simulation, using identifiers choosen by the user.

### What are the limitations ?

- explicit numerical schemes only
- for now a single mesh type for a simulation

# Overview

- Data description
- Time description
- Computation description

# Data description

### One data per line

```
data:
  name_id, domain_id
```

example :

```
data:
  h, d1
  g, cell
```

# Time description

### A single line

```
time : nb_iteration
```

example for 500 iterations :

```
time:500
```

# Computation description

### One computation per line

```
computations:
  type:name_id({read1_id, read2_id, ...},
    written_id[, neighborhood_id])
```

Three types available :

- stencil : it is a computation which involves a neighborhood around the element to compute
- local : it is a computation which does not involve a neighborhood around the element to compute
- bound : it is a computation on the physical border of the domain (additional elements)

# Computation description

### One computation per line

```
computations:
  type:name_id({read1_id,read2_id,...},
    written_id[,neighborhood_id])
```

- read1_id, read2_id : data identifiers read by the computation
- written_id : data identifier written by the computation (a single one)
- if type==stencil precise a neighborhood identifier

# Overall example

## Data

- 3 data on a domain called *cell*
- 2 data on a domain called *edgex*
- 2 data on a domain called *edgey*

## Computations

- 4 local computations
- 2 stencil computations

# Overall example

```
data:
  h, cell
  u, cell
  v, cell
  w, edgex
  z, edgey
  a, edgex
  b, edgey
time:500
computations:
  local:c0({h},u)
  stencil:c1({u},v,N1)
  local:c2({v},w)
  local:c3({v},z)
  stencil:c4({w,v},a,N1)
  local:c6({a},b)
```