# Optimization of Plant Phenotyping Workflows for the Cloud

## Inria-Brasil Workshop on Digital Science and Agronomy

Daniel de Oliveira
Universidade Federal Fluminense
danielcmo@ic.uff.br

Instituto de Computação

Inria Brasil
A PARTNERSHIP WITH LNCC AND BEYOND

# Team

Gaëtan Heidsieck

Esther
Pacitti

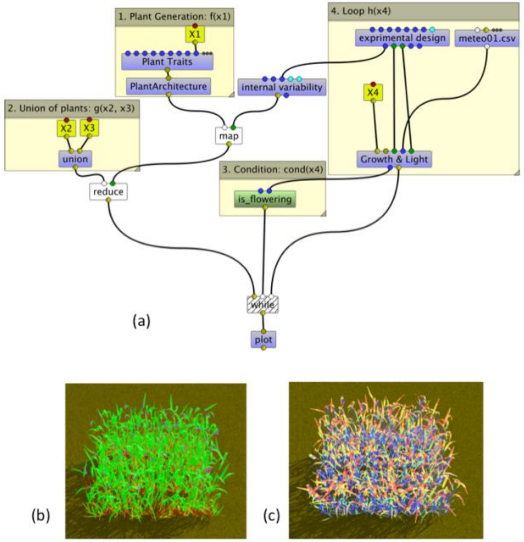
Christophe
Pradal

François
Tardieu

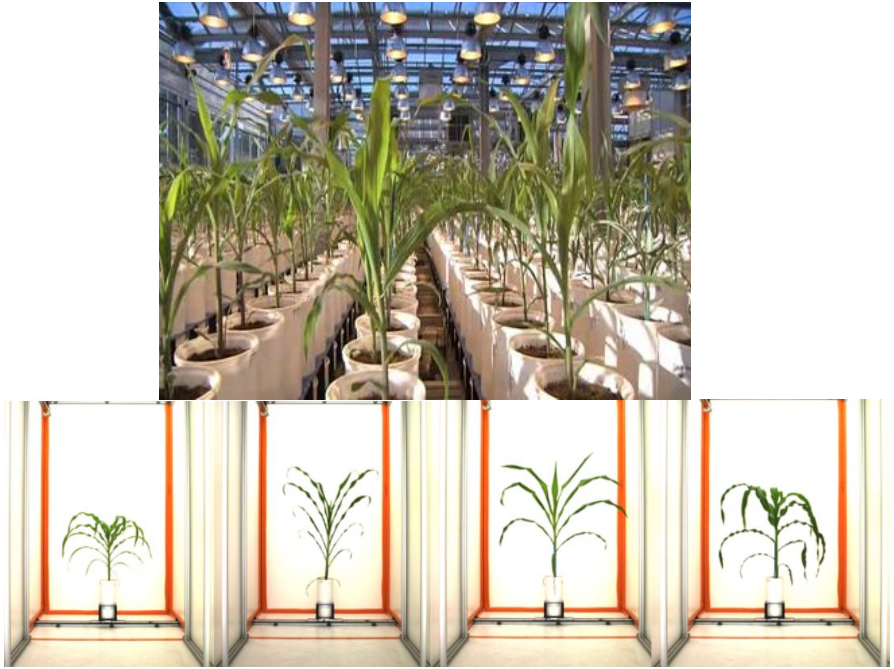
Patrick
Valduriez

# Introduction

- Data-intensive scientific workflows
- High-throughput Phenotyping (HTP)
  - Terabytes of input data
  - Huge resource consumption
  - Different teams working on same intermediate data
  - Heterogeneous data
- Complex data
  - Output data size is unpredictable, known during workflow execution
- Multisite cloud
  - Geo-distributed
  - Heterogeneous



# High-throughput phenotyping (HTP) processing management

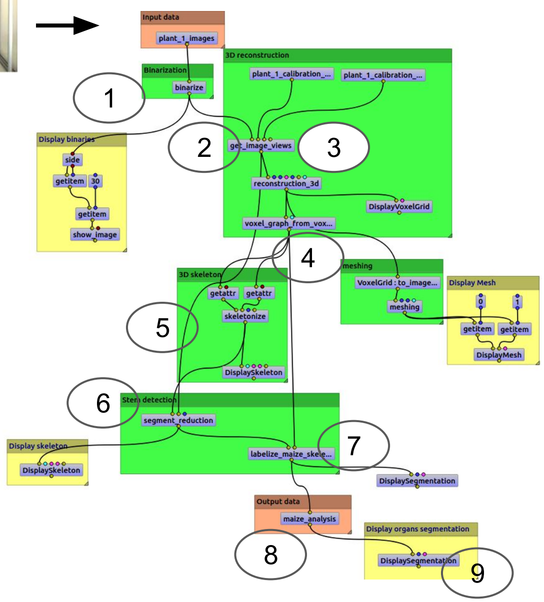


- Study the impact of different environmental conditions for various genotypes
- Quantify plants by Imaging
- Automatic High-throughput  system
  - Imaging (12 sides & top view)
    - 250 GB/day
    - 10 TB/essay
    - 30 TB / year
  - Watering and whole-plant transpiration
    - Temperature + weight measured every day

F. Tardieu, L. Cabrera-Bosquet, T. Pridmore T, M. Bennett (2017) Plant Phenomics, From Sensors to Knowledge. Current Biology 27(15):R770-R783

# Use Case : Scientific workflow for HTP image analysis

Input data




A1 : Binarization
A2 : Get side image
A3 : Get image view
A4 : 3D Reconstruction
A5 : Graph from voxel
A6 : Skeletonize
A7 : Segment reduction
A8 : Stem detection
A9 : Organ segmentation

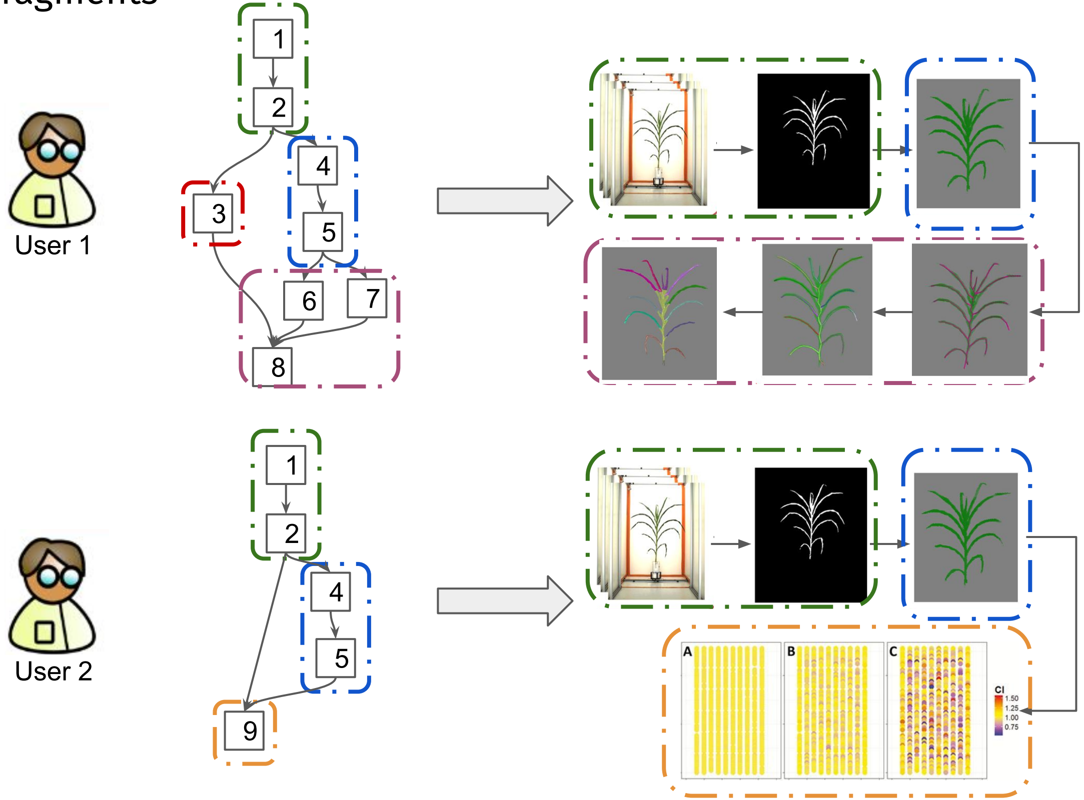
- Data intensive workflow for images processing
- Generate a segmentation of plant organs from raw images
- Simplified in 9 Activities
- Computation time and size of generated data is highly variable from similar input data (images)

C. Pradal, S. Artzet, J. Chopard, et al. (2017) InfraPhenoGrid: A scientific workflow infrastructure for plant phenomics on the Grid. Fut. Gen. Comp. Sys. 67: 341-353

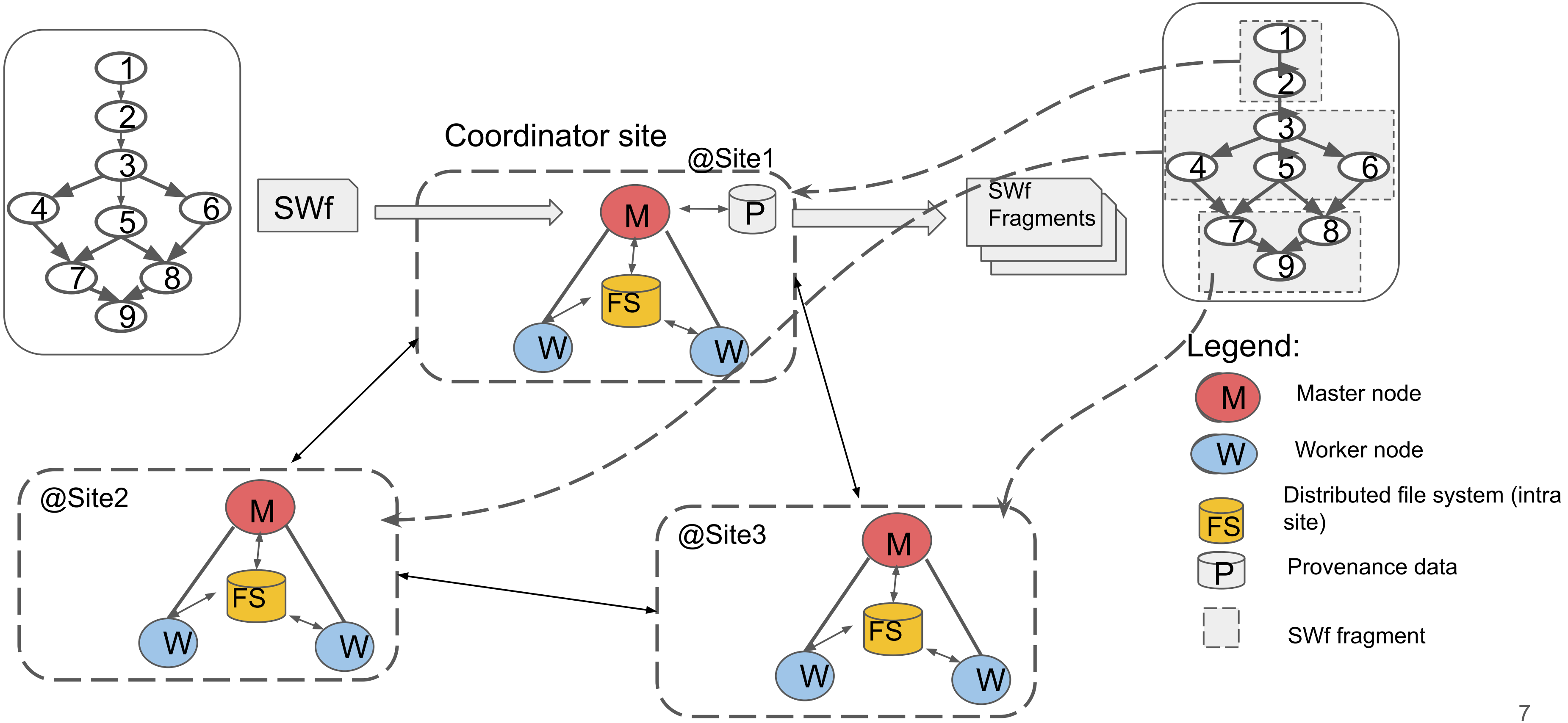
# High-throughput phenotyping (HTP) processing management

Original workflow in fragments

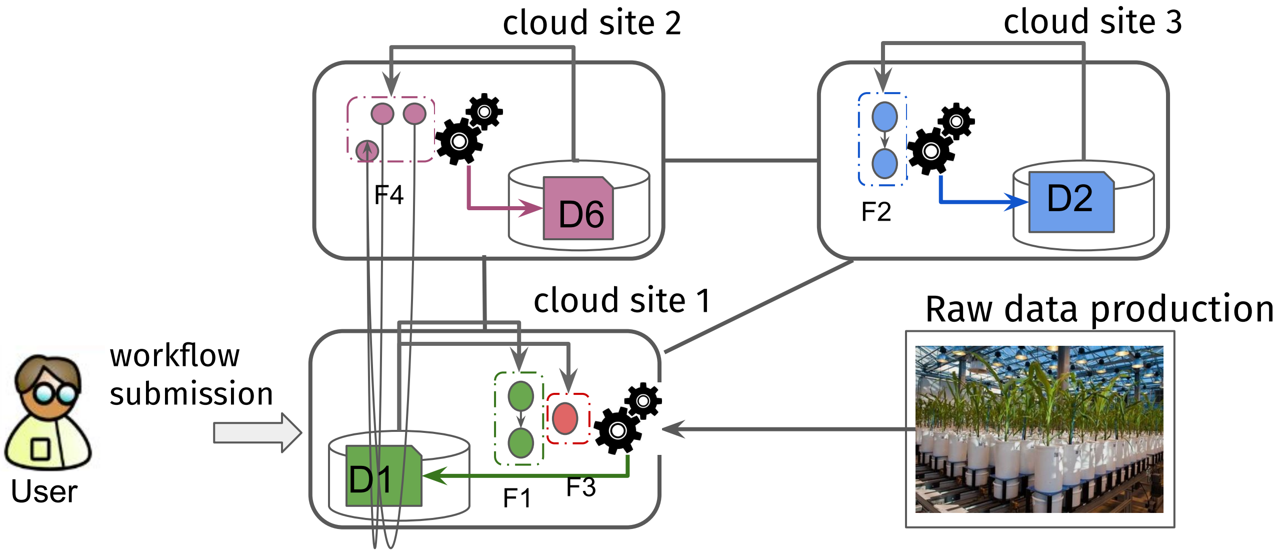
Intermediate datasets processed by activities



# Data intensive workflow in clouds



# Use Case : Workflow execution on multisite cloud



# Problem definition

- Goal:
  - Efficiently execute multiple workflows in a heterogeneous multisite cloud

- Approach:
  - Adapt a distributed cache to workflows executions

- Problems:
  - Which newly generated intermediate data should be cached?
  - At which site each cached data should be stored?
  - How to efficiently schedule the workflow with cached data?

# Approach: Cost model

- Which newly generated intermediate data should be cached?

$$p(f, s_i, s_j) = \frac{T_{write}(Output(f), s_i, s_j)}{T_{input}(f, s_i) + T_{compute}(f, s_i) - T_{tr}(Output(f), s_j, s_i)}$$

- At which site each cached data should be stored?

$$s^*(f)_{s_i} = \underset{s_j}{\operatorname{argmax}}(d_{f,i,j} * \frac{(1 - L(s_j))}{T_{write}(Output(f), s_i, s_j)})$$

- Time to feed the cache:

$$T_{feed\_cache}(f, s_i, s_j, d_{f,i,j}) = d_{f,i,j} * T_{write}(Output(f), s_i, s_j)$$

- Where the workflow fragment should be executed?

$$T_{execute}(f, s) = T_{input}(f, s) + T_{compute}(f, s) + T_{wait}(s)$$

# Approach: Cost model

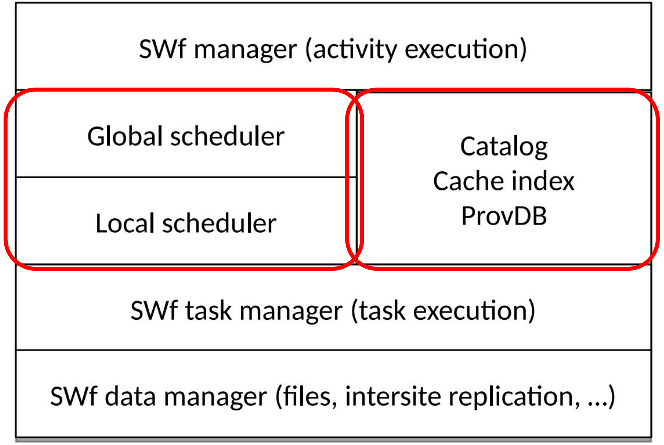
- Total cost:
  - Depends on the execution site: $s_i$
  - The cache site: $s_j$
  - The boolean decision to cache: $d_{f,i,j}$

$$T_{total}(f, s_i, s_j, d_{f,i,j}) = T_{execute}(f, s_i) + T_{feed\_cache}(f, s_i, s_j, d_{f,i,j})$$

- Global decision:
  - Find the tuple of execution site, cache site and cache decision that minimize the overall cost

$$(s^*_{exec}, s^*_{cache}, d_{f,i,j}) = \underset{s_i, s_j}{\mathrm{argmin}}(T_{total}(f, s_i, s_j, d_{f,i,j}))$$

# System design : SWfMS Architecture




- New SWfMS features:
  - Cache manager:
    - Handle the distributed cache
  - Cache index:
    - A shared index for all SWf
    - Allows cached data to be reuse from any SWf
    - Replicated on all cloud sites

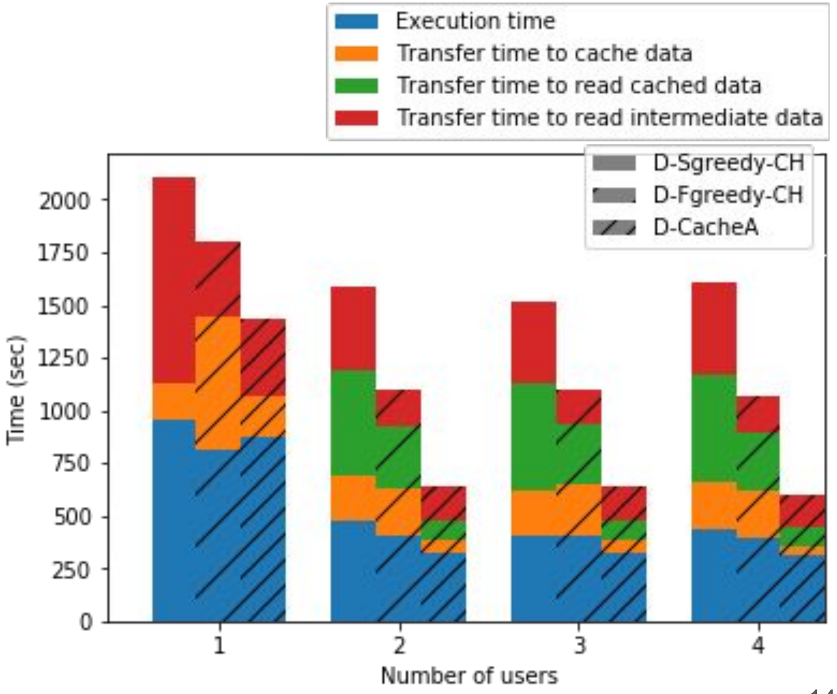
# Experimental Evaluation

- Experimental setup:
  - Use the Phenomenal SWf
  - Raw data from remote server:
    - 200 Gigabytes of data
    - 35 time series of plants = 15,000 computational tasks
  - 3 sites in France: Lyon, Lille, Montpellier
    - 96 vCPUs in total
    - 180 GB storage for the cache in total
- Algorithms:
  - Baseline:
    - D-Sgreedy: greedy-based schedule a fragment on a site when available
    - D-Fgreedy: schedule fragments on the site that minimize the execution time and intermediate data transfers
  - Proposed:
    - D-CacheA: based on the global decision proposed

# Evaluation: Multiple users

- The workflow is executed by 4 users
- There is no cached data for the first one
- Each user share 60% of the input data with the others
- Heterogeneous cloud sites

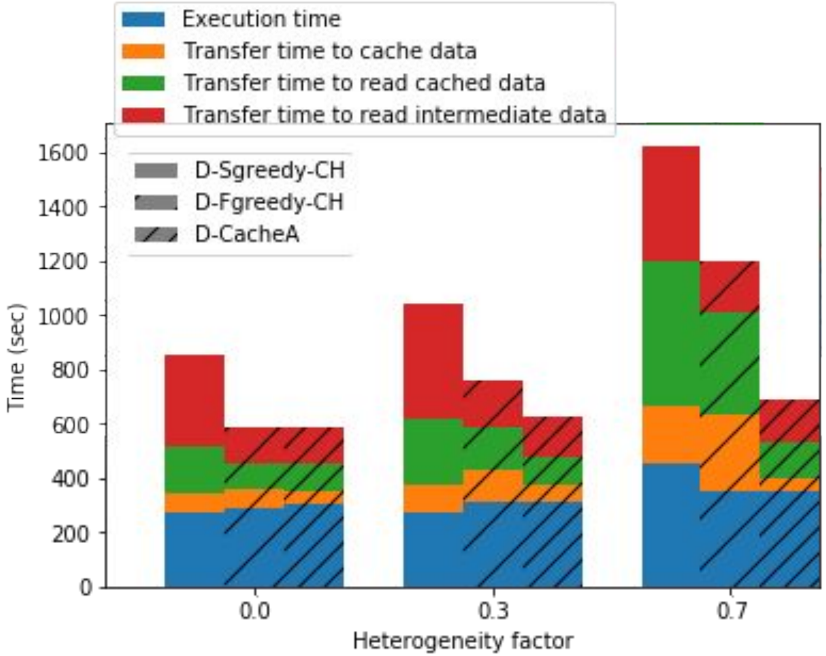
- Shorter total time by:
    - 24% for the first user
    - 43% for the others



# Evaluation: Different cloud configurations

- The workflow is executed once
- There is cached data existing for 60% of the input data
- Three cloud configurations (homogeneous to very heterogeneous)

- Shorter total time by:
  - ~ similar time for homogeneous sites
  - 18%
  - 42%



# Conclusion

- The proposed approach enables:
  - Efficient execution of workflows in a multisite cloud with a cache
  - Automatically share intermediate data between users

- Contributions:
  - An **architecture** for executing workflows in multisite cloud with caching
  - A **scheduling algorithm** that minimize the total cost of executing a workflow with cache
  - An **experimental evaluation** on a data-intensive application

# Thank you!

## Inria-Brasil Workshop on Digital Science and Agronomy


Daniel de Oliveira
Universidade Federal Fluminense
danielcmo@ic.uff.br


Instituto de Computação

Inria Brasil
A PARTNERSHIP WITH LNCC AND BEYOND