workflows community initiative

Workflows Community Summit 2025 - Amsterdam

IIWM: A MACHINE LEARNING STRATEGY FOR IN-MEMORY EXECUTION OF DATA-INTENSIVE PARALLEL WORKFLOWS

DOMENICO TALIA, R. CANTINI , F. MAROZZO , A. ORSINO P. TRUNFIO

UNIVERSITÀ DELLA CALABRIA, ITALY

Friday June 6th, 2025

1

OUTLINE

- Background
- Goals & Results
- IIWM prediction model
- Experimental results
- Conclusions
- Future work

2

2

BACKGROUND

- Today **data-intensive workflows** are largely used to orchestrate complex sets of tasks handling and processing **huge amounts of data**.
- A data-intensive workflow is a computational process that involves processing steps implementing big data acquisition, data transformation, data analysis, result storage and visualization.
- Efficient techniques (like machine learning) are vital to **reduce execution time** when complex data-intensive workflows must be run efficiently.
- In particular, **in-memory processing prediction** can bring important benefits to speeup execution by **avoiding/limiting** usage of **disk storage**.

3

3

GOALS & RESULTS

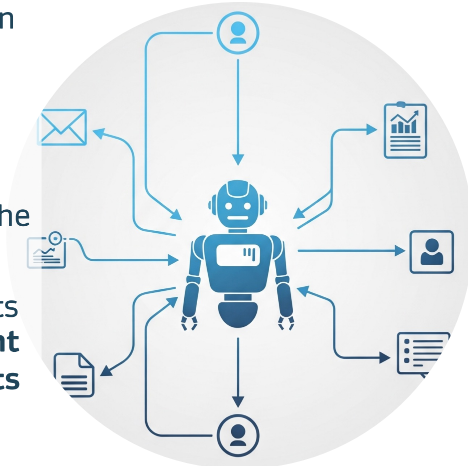
- We developed a new tool, called **Intelligent In-memory Workflow Manager (IIWM)**, for optimizing the in-memory execution of data-intensive workflows on parallel machines.
- IIWM is based on two complementary strategies:
  1. a **machine learning strategy** for predicting the memory occupancy and execution time of workflow tasks;
  2. a **scheduling strategy** that allocates tasks to a computing node, taking into account the (predicted) memory occupancy and execution time of each task and the memory available on that node.
- The **effectiveness** of the machine learning-based predictor and the scheduling strategy have been **assessed experimentally**.

4

4

IIWM FOR MEMORY RESORCES

- The IIWM workflow manager improves application performance through adaptive usage of memory resources.
- This is done by **identifying clusters of tasks** that can be executed in parallel **on the same node**, **optimizing in-memory processing**, so **avoiding** the use of **disk storage**.
- Given a data-intensive workflow, the IIWM exploits a meta-learning model for **estimating the amount of memory** required by each workflow task **and its execution time**.



5

5

IIWM PREDICTIONS

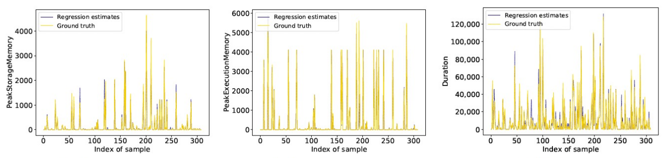
- The IIWM meta-learning model (regressions+decision tree) is **trained on a log of past** executed **workflows**.
- A set of relevant features of workflows are considered:
  - **Workflow structure**, in terms of tasks and data dependencies.
  - **Input size & format**, such as the number of rows, dimensionality, and all other features required to describe the complexity of input data.
  - The **types of tasks**, i.e., the computation performed by a given node of the workflow.
  - For example, in the case of data analysis workflows, we can distinguish among supervised learning, unsupervised learning, and association rule discovery tasks, as well as between training and prediction tasks.

6

6

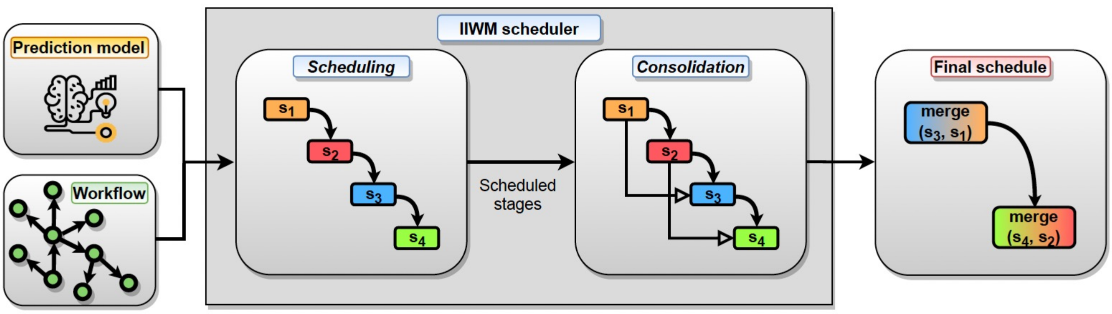
IIWM PREDICTIONS

- Predictions made for a given computing server are **applicable to all similar computing servers** (i.e., having the same architecture, processor type, operating system, memory resources).
- This makes the proposed approach effectively **usable on large-scale homogeneous HPC systems** composed of many identical servers.
- Given a data-intensive workflow, the IIWM **exploits the estimates coming from the machine learning model for producing a scheduling plan** aimed at **reducing** (and, in most cases, **avoiding**) main **memory saturation** events, which can occur when multiple tasks are executed concurrently on the same computing node.



(a) Storage memory (b) Execution memory (c) Duration

EXECUTION FLOW OF THE IIWM SCHEDULER



Given a **workflow** and a **prediction model**, **IIWM** generates a **scheduling plan** in two steps:

i. **stages and task assignment building** (the goal is to avoid swapping to disk due to memory saturation);

ii. **stage consolidation** (aimed at reducing the number of stages by merging stages without dependencies according to the available memory).

BACKGROUND: MULTIMODAL APPROACHES

- IIWM has been **experimentally evaluated** using Spark as a testbed.
- We assessed the benefits coming from the use of the IIWM by executing
  - two synthetic workflows and
  - a real one

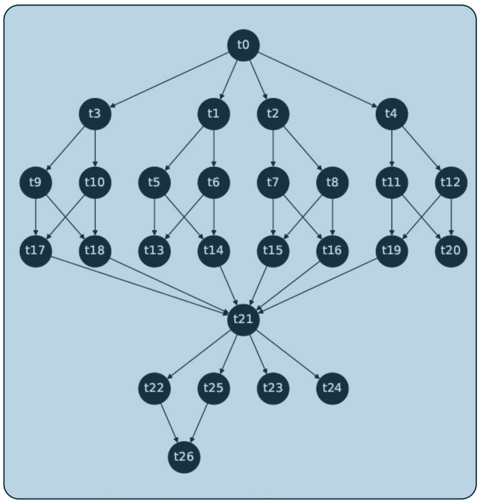
  generated for investigating specific scenarios related to the presence of a high level of parallelism and a limited amount of main memory reserved for execution.
- We carried out an in-depth comparison between the IIWM and a blind scheduling strategy, which only considers workflow dependencies for parallel execution of tasks.

9

9

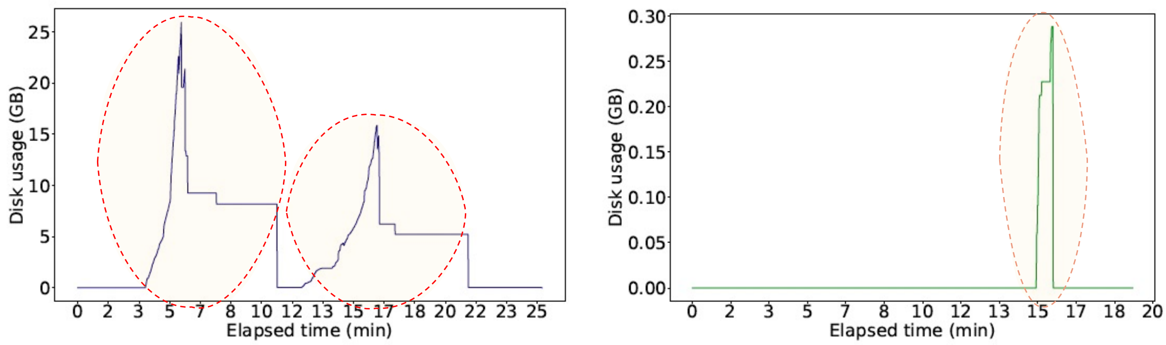
EXPERIMENTAL RESULTS – A USE CASE

- A workflow composed of the 27 tasks was characterized by highly heavy tasks and very low resources, where the execution of a single task can exceed the available RAM memory.
- In particular, the **task T18** had an estimated **peak memory occupancy higher than Spark's available unified memory** of 5413.8 MB (i.e., corresponding to a heap size of 9.5 GB).
- This would bring the IIWM scheduling algorithm to allocate the task to a new stage, but memory would be saturated anyway.



10

10



(a) Disk usage of full-parallel

(b) Disk usage of the IIWM

EXPERIMENTAL RESULTS

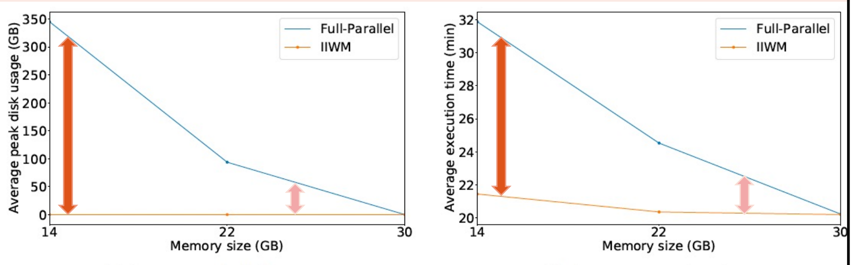
DISK USAGE OVER TIME

IIWM cannot avoid data spilling, even though its disk usage was much lower considering the peak value and write duration compared to blind full-parallel.

11

EXPERIMENTAL RESULTS

AVERAGE PEAK DISK USAGE AND EXECUTION TIME



(a) Average peak disk usage

(b) Average execution time

IIWM is able to **adapt the execution to available resources**, finding a good trade-off between the maximization of the parallelism and the minimization of the memory saturation probability.

12

CONCLUSIONS & FUTURE WORK

- **Data-intensive workflows are widely used** in several application domains, such as bioinformatics, high-energy physics, Gen IA, data science, complex simulation.
- The **Intelligent In-memory Workflow Manager** (IIWM), aims at **optimizing the in-memory execution** of data-intensive workflows on high-performance computing systems.
- Experimental results suggested that by jointly **using a machine learning model for performance estimation and a suitable scheduling strategy**, the execution of data-intensive workflows **can significantly improve memory usage** with respect to state-of-the-art blind strategies.
- In future work, additional aspects of the performance estimation will be investigated, extending information about tasks, input data, and hardware platform features.

13

13

THANKS

14