CAPIO: Cross-Application Programmable I/O

Alberto Riccardo Martinelli[1], Massimo Torquati[2] and Marco Aldinucci[1]

University of Turin, Computer Science Dept.[1]
University of Pisa, Computer Science Dept. [2]

September 14, 2022

What is CAPIO?

CAPIO is an user-space middleware that optimize and coordinates the data transfer between workflow applications communicating via files without modifying the original code by reducing the pressure on the I/O subsytem enabling in-situ and in-transit data transformations.

What is CAPIO?

It's a complex definition. To better understand, CAPIO can be represented as the composition of two levels of abstraction.

- CAPIO runtime
- CAPIO coordination language

Background: I/O in HPC systems

- The gap between processors and I/O subsystems’ speed has continuously been increasing.

Background: I/O in HPC systems

- The gap between processors and I/O subsystems' speed has continuously been increasing.
- In HPC systems usually is installed a distributed file system, i.e. the data is scattered in different machines.

Background: I/O in HPC systems

- The gap between processors and I/O subsystems’ speed has continuously been increasing.
- In HPC systems usually is installed a distributed file system, i.e. the data is scattered in different machines.
- A lot of applications exchange data using files.

State of the art

- Alternatives to the POSIX I/O API (MPI I/O, DAMARIS, HDF5, etc...)
- Data staging systems (NORNS)
- Ad Hoc Filesystems (GekkoFS, UnifyFS, etc...)

The problem with the state of the art

- The POSIX API is still the most used.

The problem with the state of the art

- The POSIX API is still the most used.
- There is a lot of legacy code that no one want to modify.

The problem with the state of the art

- The POSIX API is still the most used.
- There is a lot of legacy code that no one want to modify.
- Some tools do not resolve the bottleneck problem (they rely on the file system).

The problem with the state of the art

- The POSIX API is still the most used.
- There is a lot of legacy code that no one want to modify.
- Some tools do not resolve the bottleneck problem (they rely on the file system).
- Most tools do not exploit streaming communications.

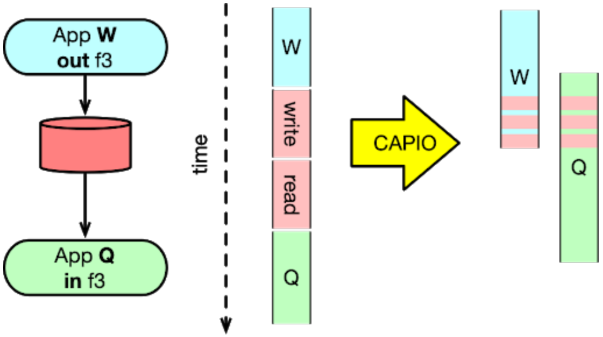
The problem with the state of the art

- The POSIX API is still the most used.
- There is a lot of legacy code that no one want to modify.
- Some tools do not resolve the bottleneck problem (they rely on the file system).
- Most tools do not exploit streaming communications.
- Most tools focus on the single application, not on workflows.

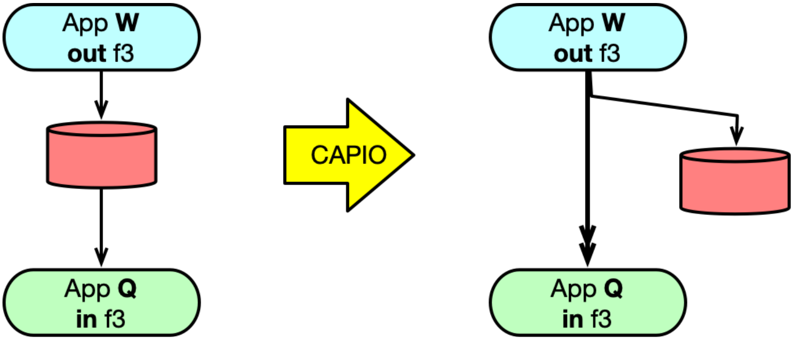
CAPIO

How does CAPIO advance the state of the art?

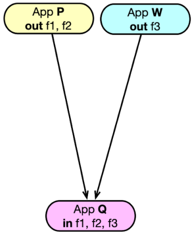
The CAPIO runtime Transforms a batch execution in a streaming execution.

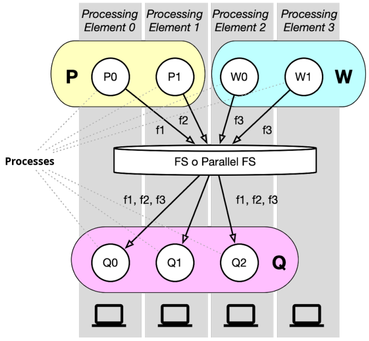


The CAPIO runtime removes the I/O operations from the critical path of the workflow.

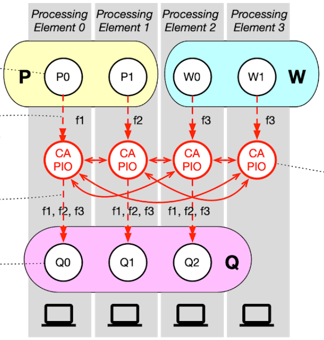


Legacy workflow.





CAPIO workflow.



***CAPIO** intra-PE connectror*
POSIX shm

***CAPIO** stub library (API)*
linked in the logical adress
space of P, W, Q

***CAPIO** broker*
MPI parallel code
1 deamon per PE

The data distribution
ca be programmed

CAPIO coordination language.

The CAPIO coordination language allows the user to express the I/O graph. The I/O graph represents the data communicated between (parallel and/or distributed) applications of a workflow.

Why is it useful?

The coordination language allows to optimize the data transfer between applications and to perform in-situ and in-transit data transformation through a plug-in system.

From the I/O graph to the execution environment

In order to optimize the data transfer, we need to know where the applications (and its processes) will be executed.

CAPIO: Cross-application programmable I/O

In summary, these are the main features of CAPIO:

- Enhancement of the I/O performance

CAPIO: Cross-application programmable I/O

In summary, these are the main features of CAPIO:

- Enhancement of the I/O performance
- From a batch execution to a streaming execution

CAPIO: Cross-application programmable I/O

In summary, these are the main features of CAPIO:

- Enhancement of the I/O performance
- From a batch execution to a streaming execution
- No changes to the original code

CAPIO: Cross-application programmable I/O

In summary, these are the main features of CAPIO:

- Enhancement of the I/O performance
- From a batch execution to a streaming execution
- No changes to the original code
- Programmable inter-applications data movement through a configuration file

CAPIO: Cross-application programmable I/O

In summary, these are the main features of CAPIO:

- Enhancement of the I/O performance
- From a batch execution to a streaming execution
- No changes to the original code
- Programmable inter-applications data movement through a configuration file
- Programmable in-situ and in-transit data transformation through plugins