Experiences combining malleability and I/O control mechanisms

David E. Singh and Jesús Carretero

University Carlos III of Madrid (Spain)

Introduction

- I/O challenges
  - Scientific applications (climate, genomics, high energy physics, astronomy etc.) ingest, generate, and process increasingly larger data sets
  - Future high scale supercomputers need to deal efficiently with huge amounts of data
  - Current I/O software stack needs to evolve in order to meet the oncoming scalability challenges
- CPU challenges
  - Malleable applications can leverage unused computational resources

Data staging challenges

- Concurrent parallel data flows
  - Lack of data staging coordination
    - Among applications
    - Between applications and the system
- Lack of standards for dynamic monitoring of large scale infrastructures
- Need of coupled control and data mechanisms
- Lack of coordination with the job scheduler

Goals

- Integration of CLARISSE and FlexMPI into a framework
  - New coordination techniques between the applications and the scheduler
  - Application monitoring

- Use of application malleability to enhance the I/O performance:

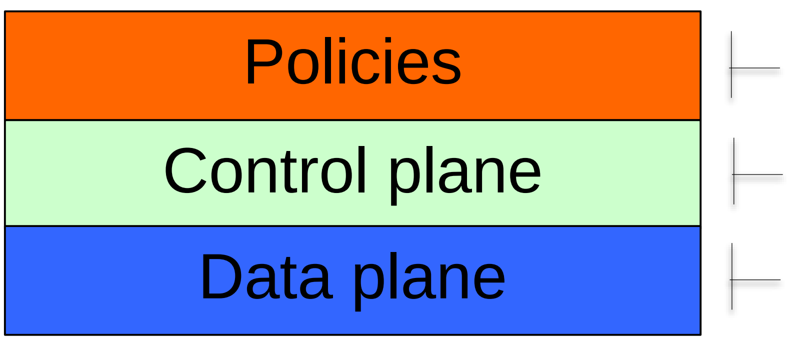
  - Coordinated use of parallel I/O scheduling and malleability for reducing number of I/O interferences

    I/O interference: *two or more I/O operations that occur partially or totally at the same time competing for the I/O resources*

  - I/O-aware scheduling policies

CLARISSE overview

- Novel mechanisms for global data staging coordination to improve:
  - Load balance, resilience, parallel I/O scheduling, locality exploitation
- Decouple the data and control planes
  - Data plane
  - Control plane
  - Policy
- Facilitate the flow of control and data across the I/O stack

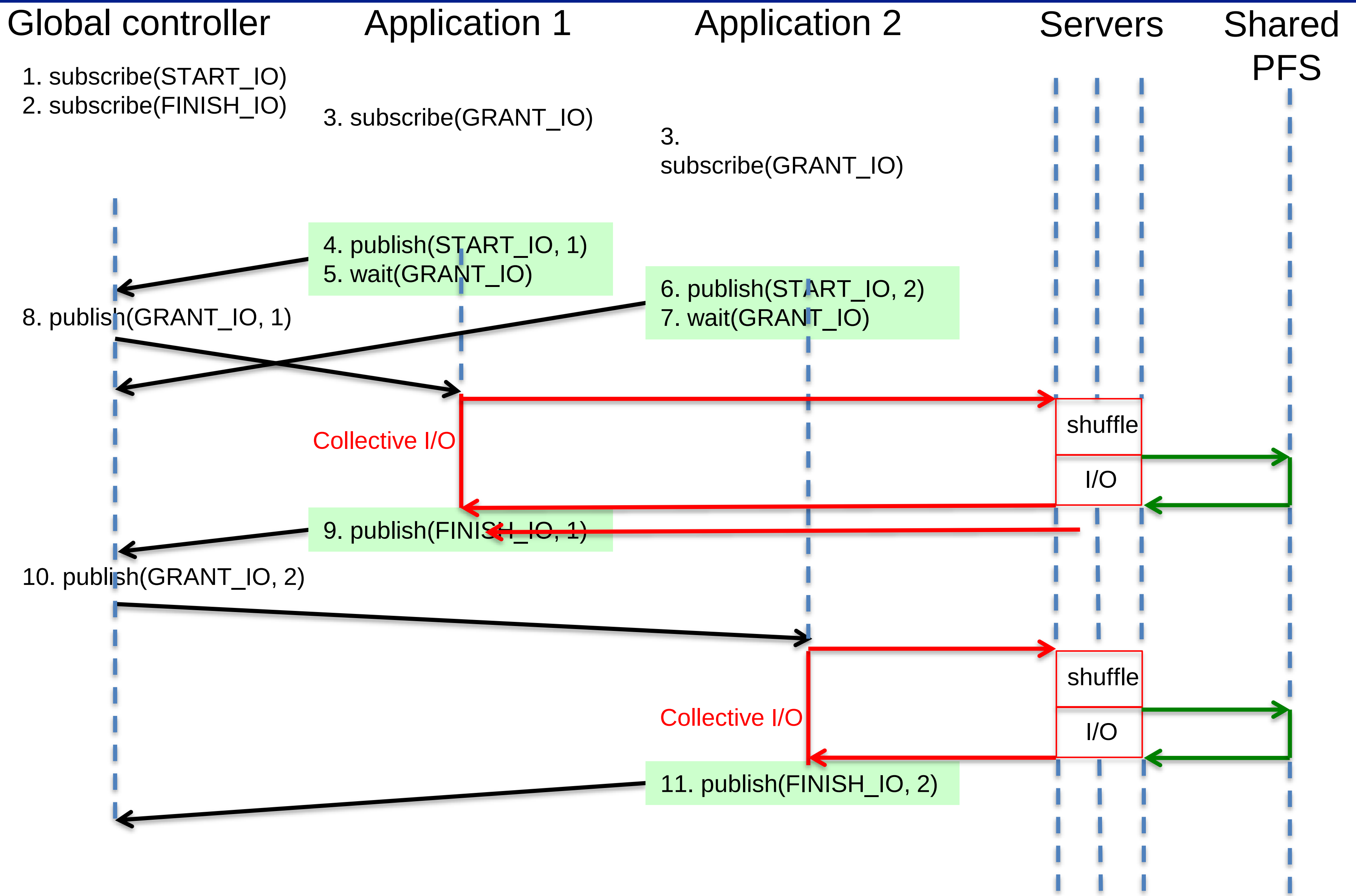


Elastic collective I/O, parallel I/O scheduling, resilience, load balancing, etc.

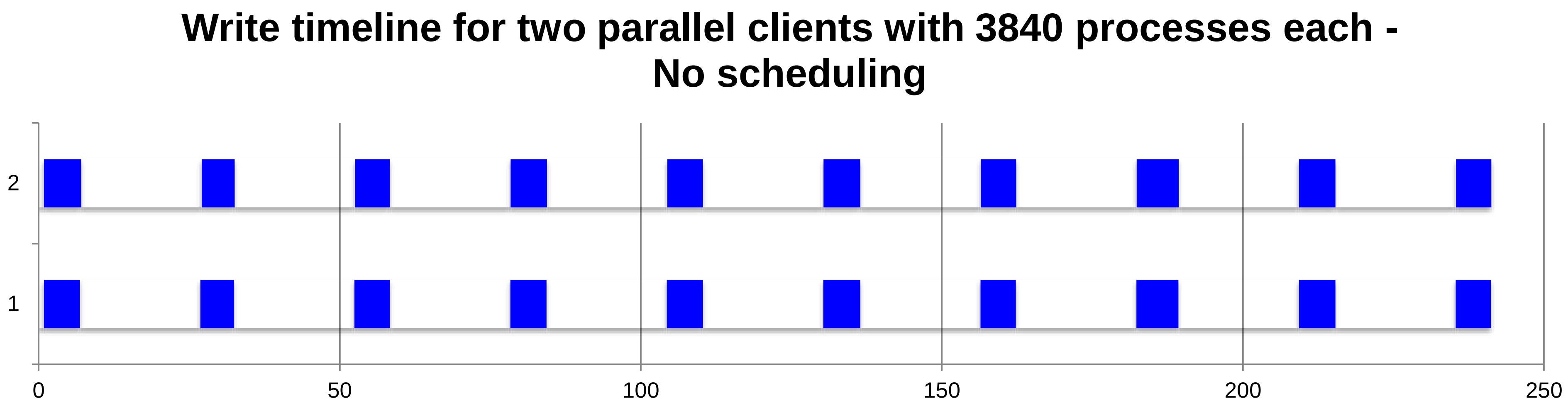
Publish/Subscribe API

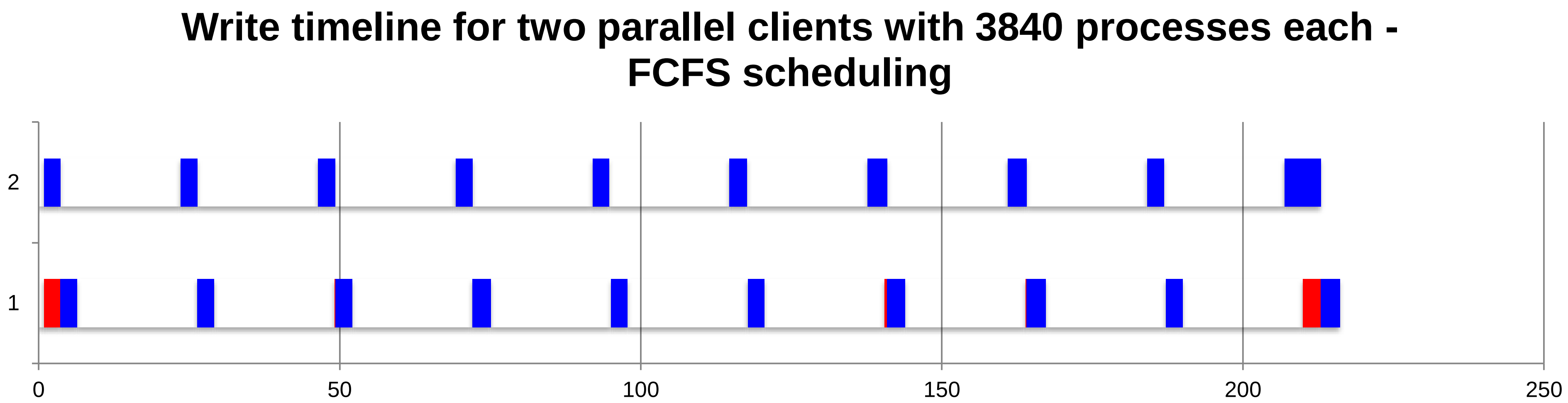
Collective I/O, Independent I/O (MPI-IO, put/get APIs), buffering

Parallel I/O scheduling



I/O scheduling versus no scheduling



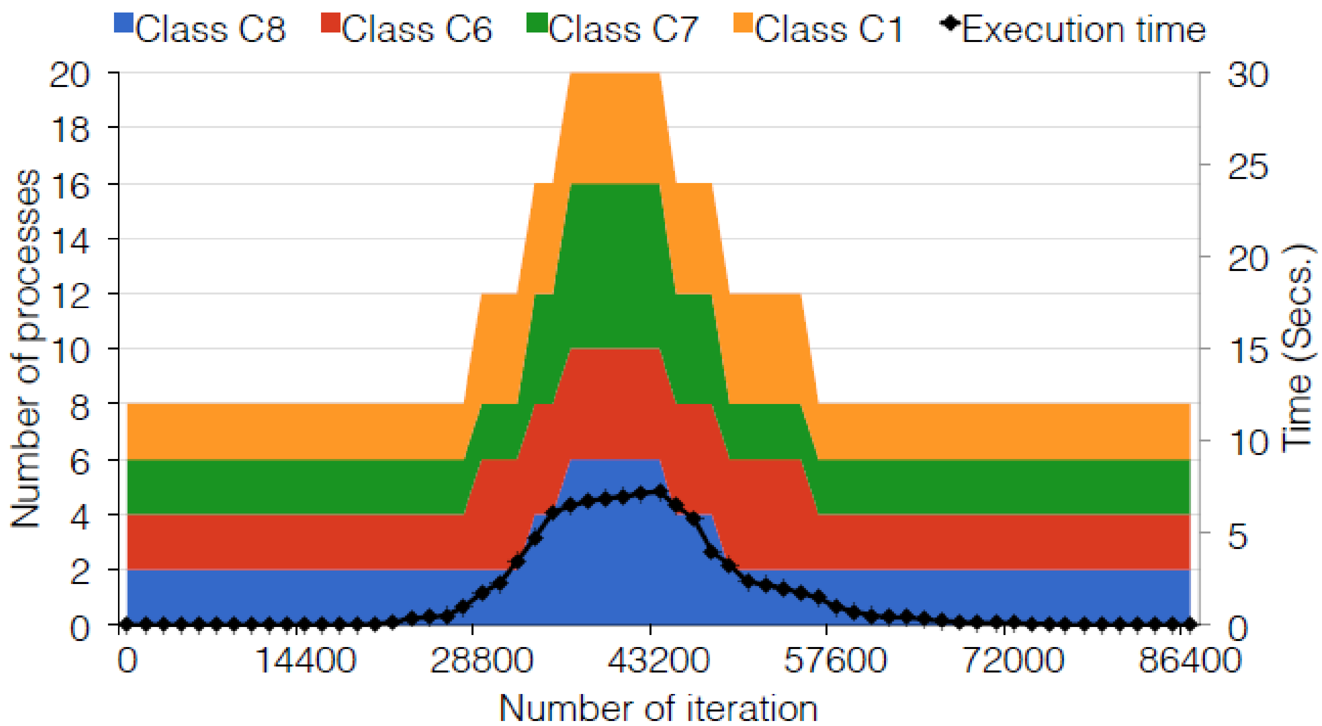


FlexMPI overview

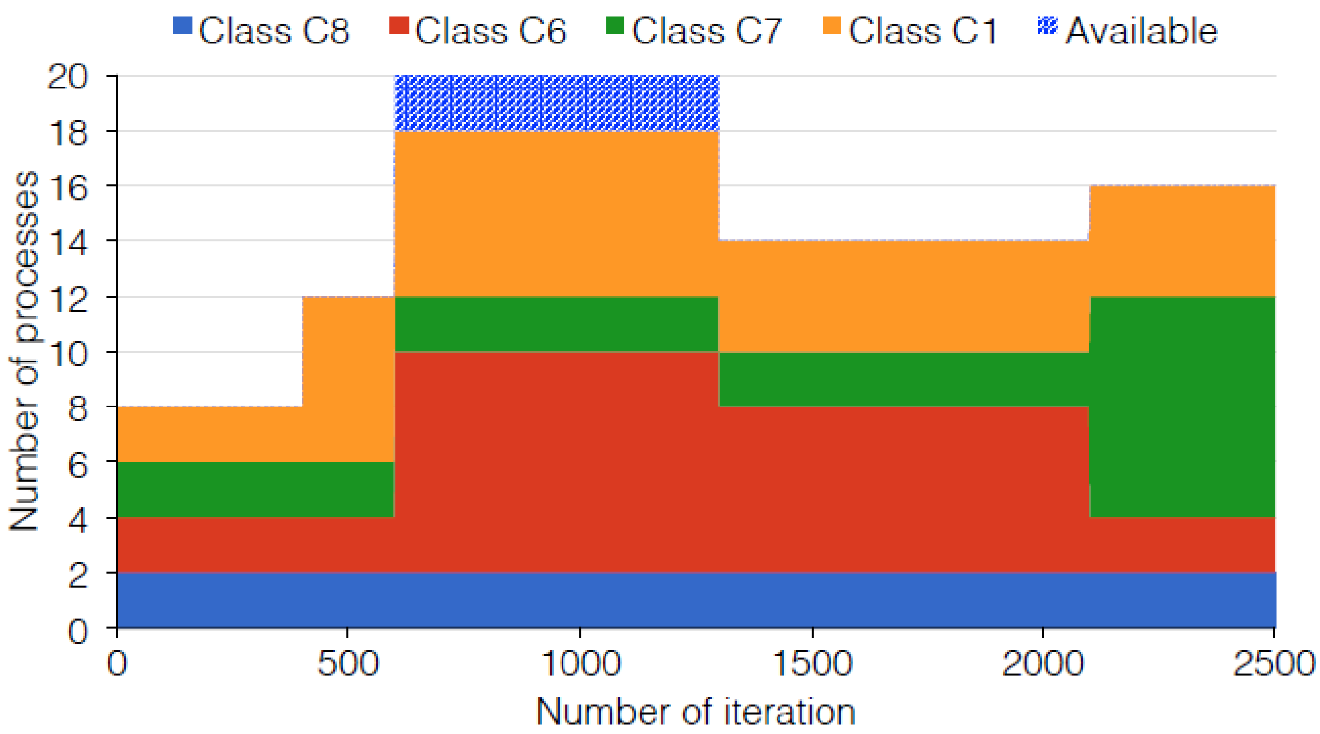
- FLEX-MPI provides performance-aware malleability capabilities for MPI applications.

- Goals:
  - Dynamic application reconfiguration
  - Automatic load balancing
  - Monitoring capabilities

FlexMPI example



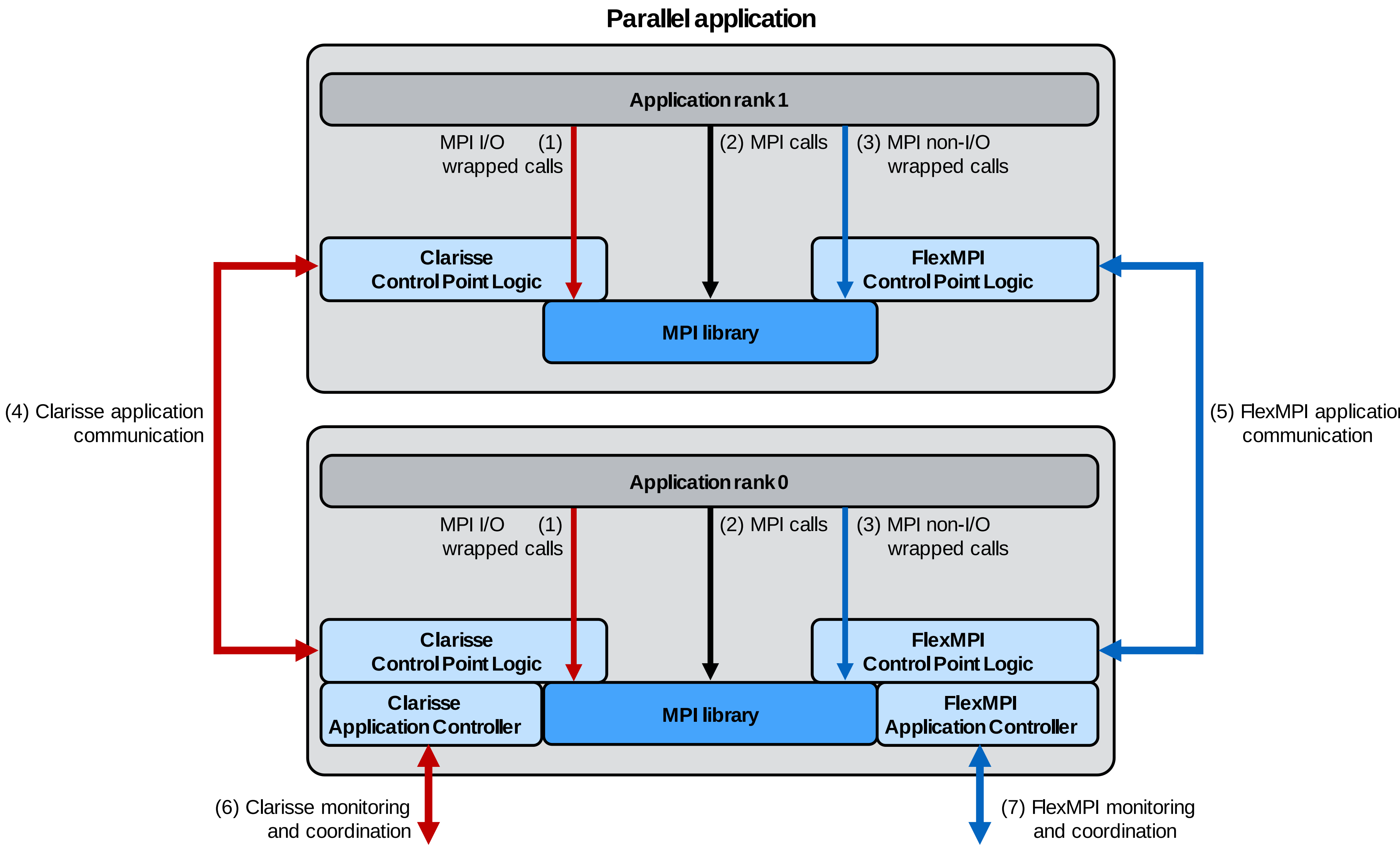
FlexMPI example



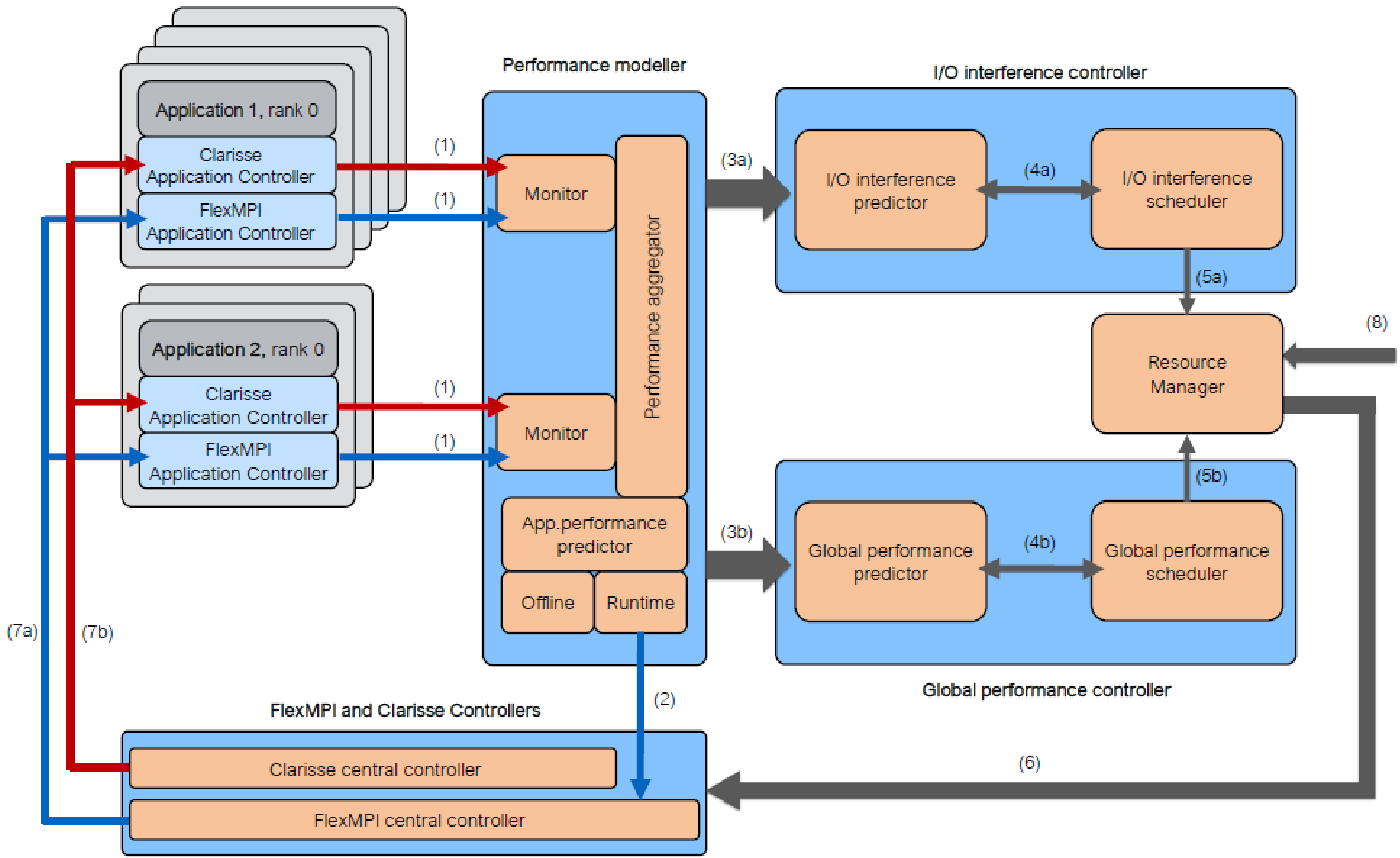
Combined framework

- CLARISSE and FlexMPI integrated at application-level
- CLARISSE and FlexMPI use separate external controllers
- New control logic coordinates both runtimes

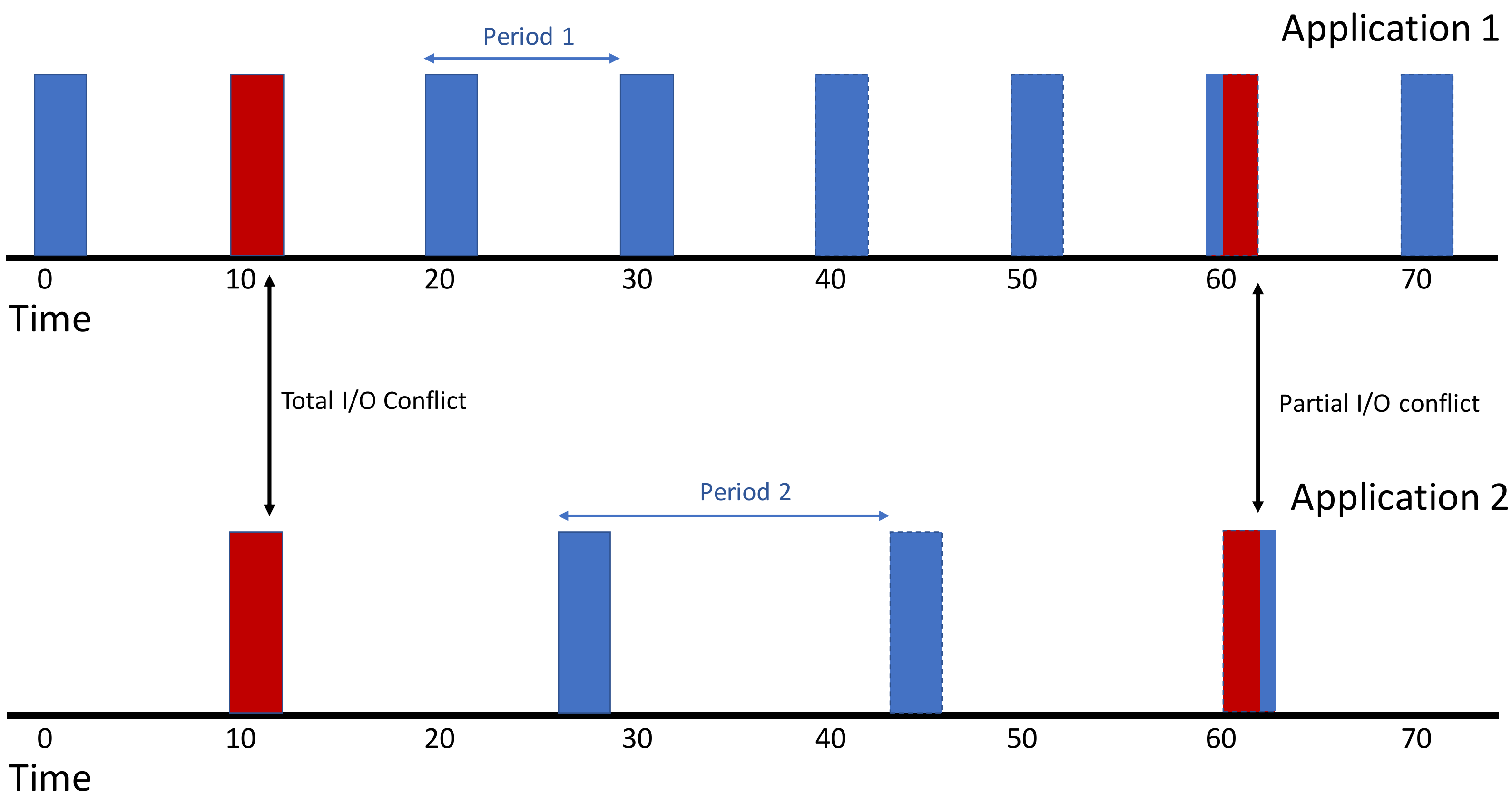
Combined framework



Combined framework

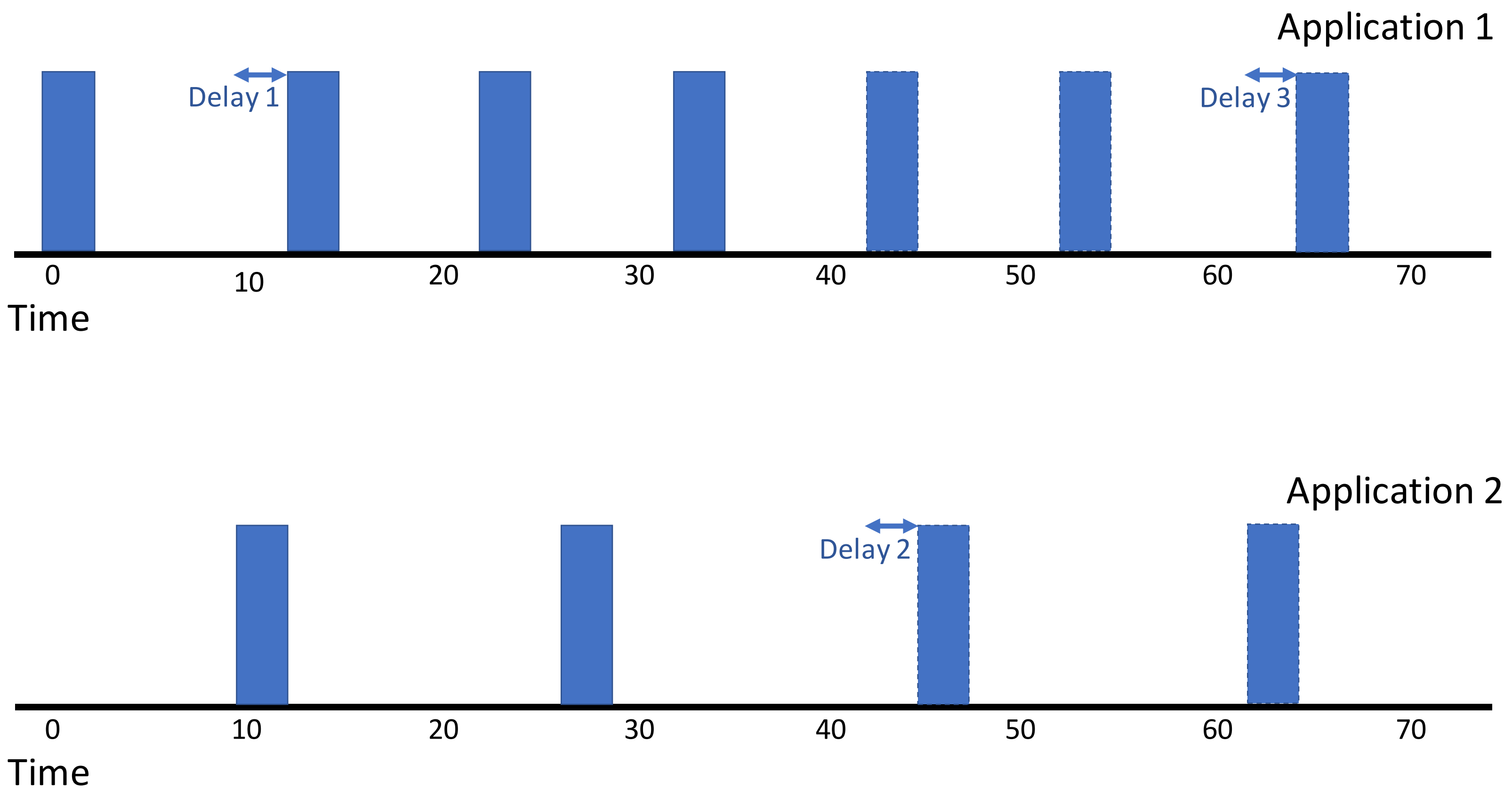


I/O interference



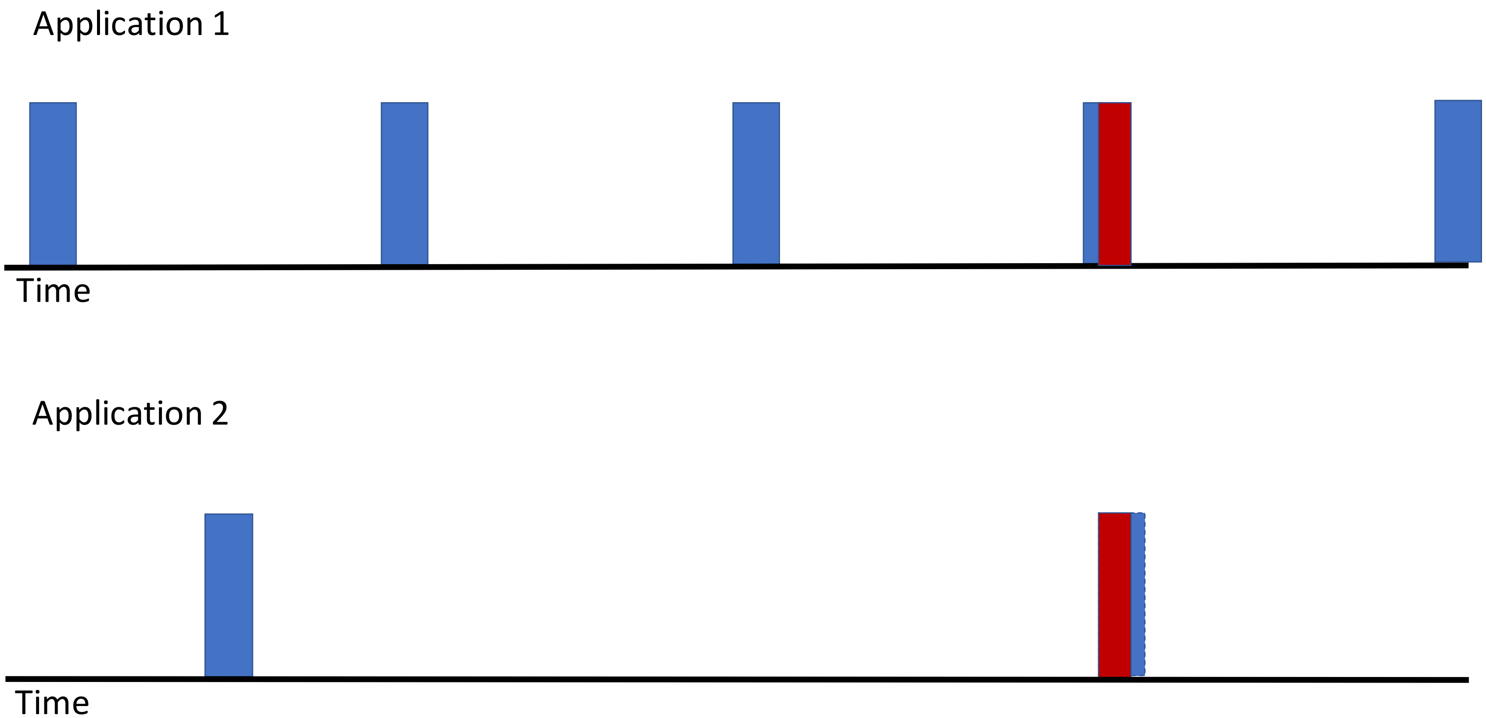
CLARISSE

- Solutions:
  - **I/O scheduling**: blocks one I/O operation using publish-subscribe support



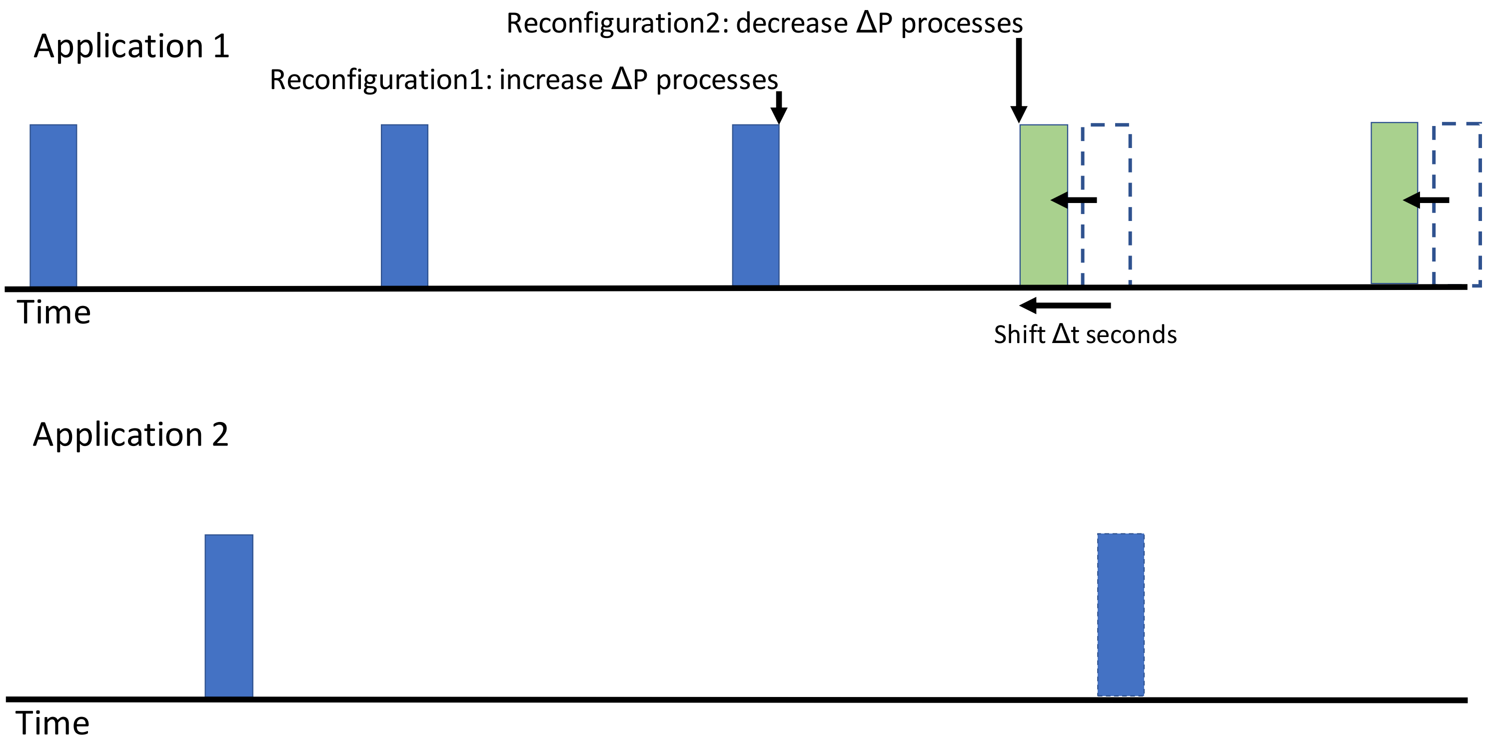
I/O interference avoidance

- Avoiding I/O conflicts with Clarisse + FlexMPI
  - Leverage malleability for changing the I/O time stamp
  - Prediction of the I/O interference



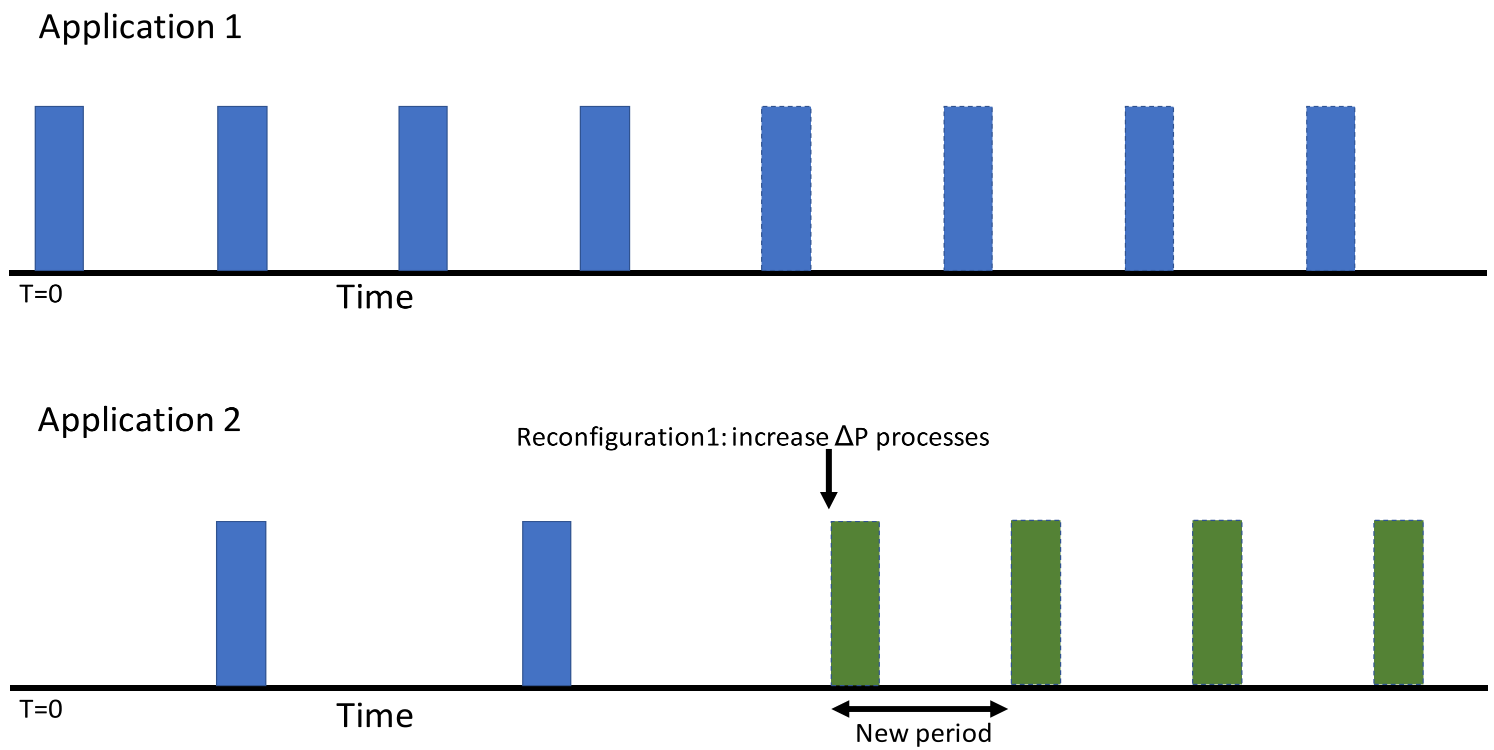
I/O interference avoidance

- **Phase shifting**
  - Leverage malleability for changing the I/O access time (phase)
  - Temporary use of computational resources



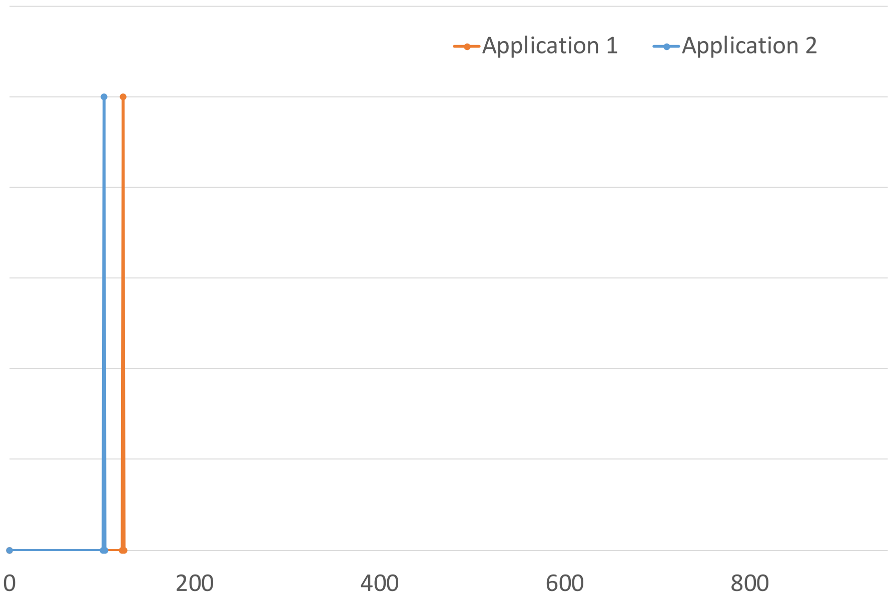
I/O interference avoidance

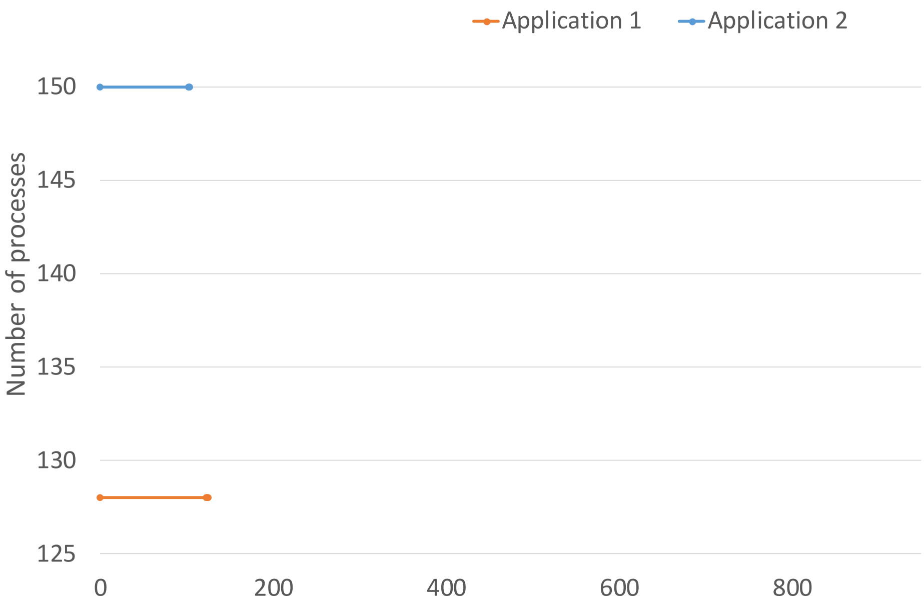
- **Phase coupling**
  - Leverage malleability for changing the I/O period
  - Long-term use of computational resources



I/O interference avoidance

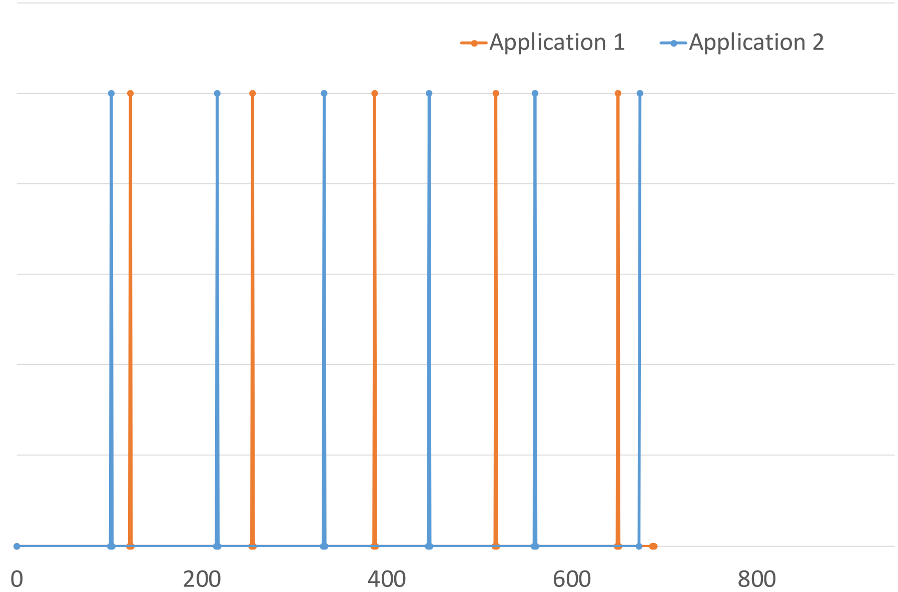
- **Results**
  - Bebop cluster
  - Two applications
  - 150 and 128 processes

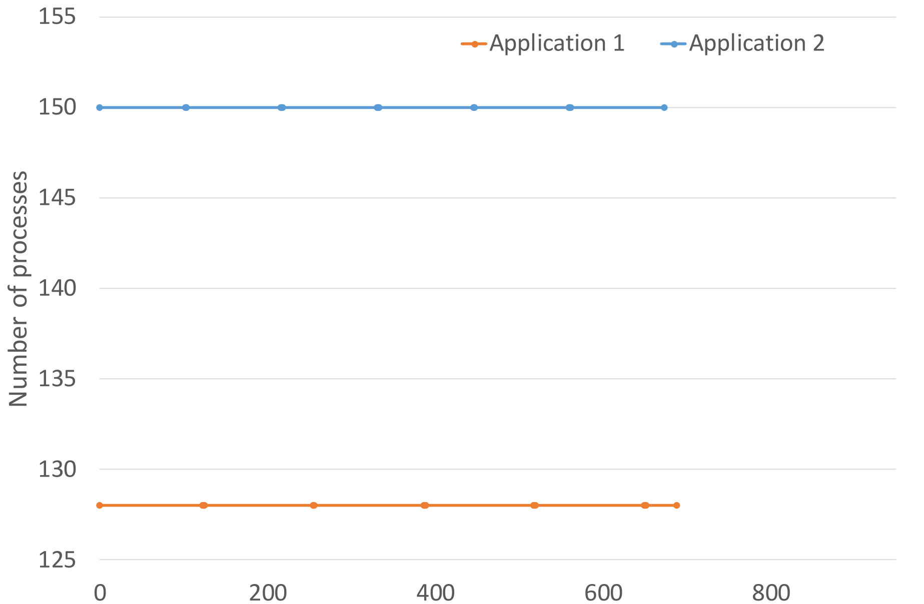




I/O interference avoidance

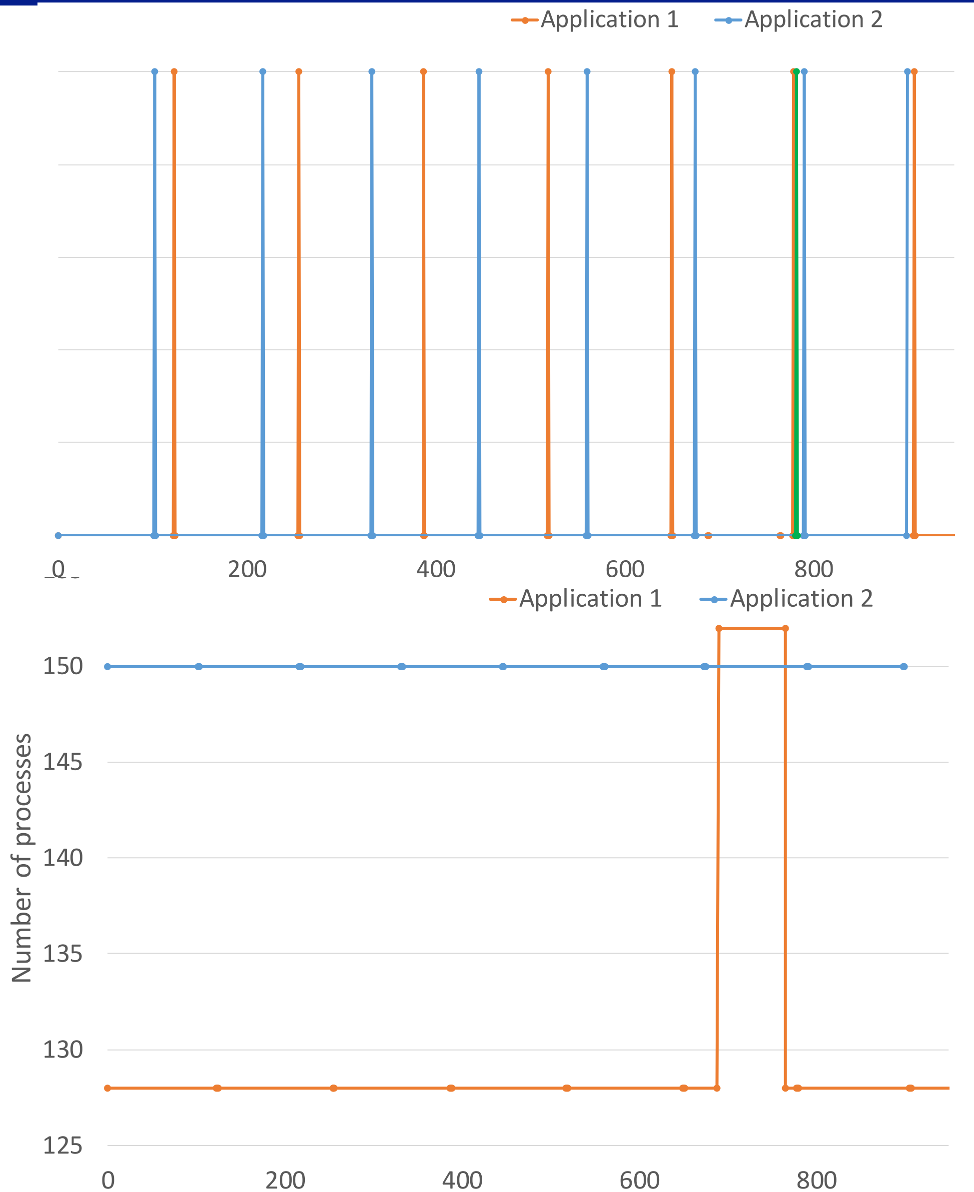
- **Results**
  - Bebop cluster
  - Two applications
  - 150 and 128 processes





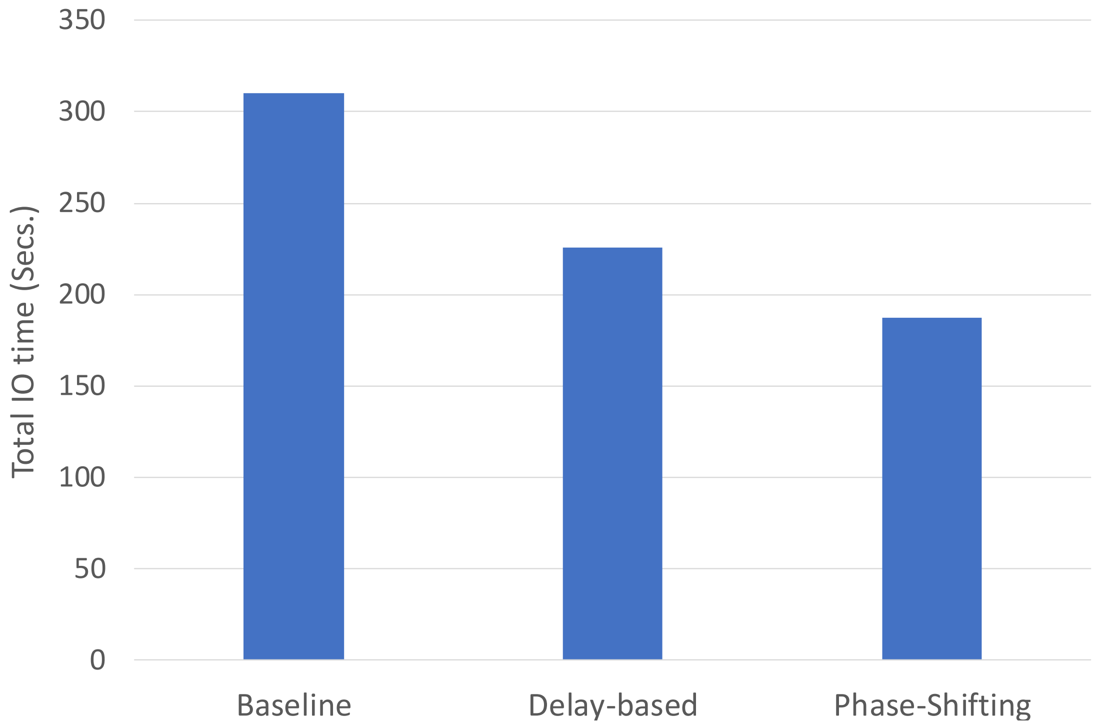
I/O interference avoidance

- **Results**
  - Bebop cluster
  - Two applications
  - 150 and 128 processes



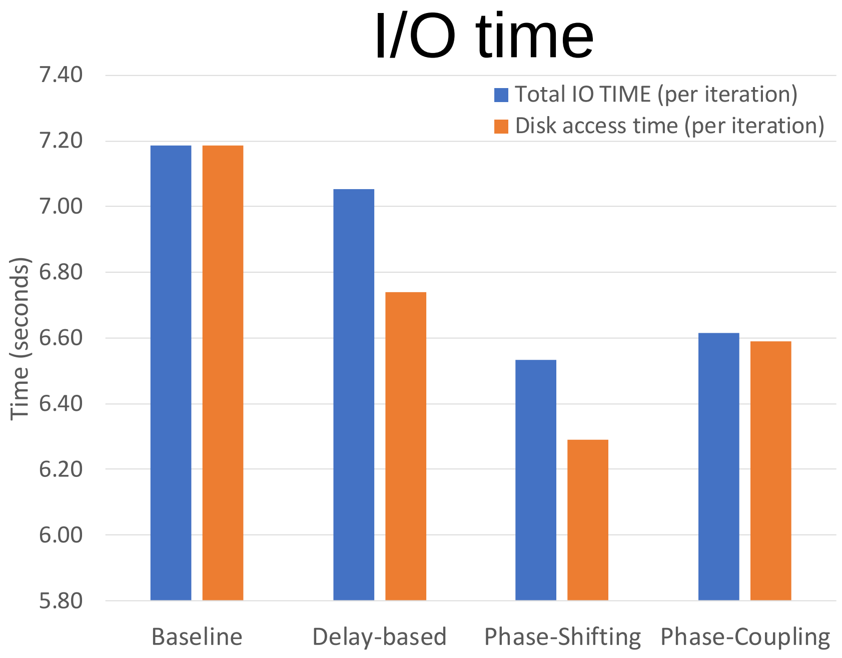
I/O interference avoidance

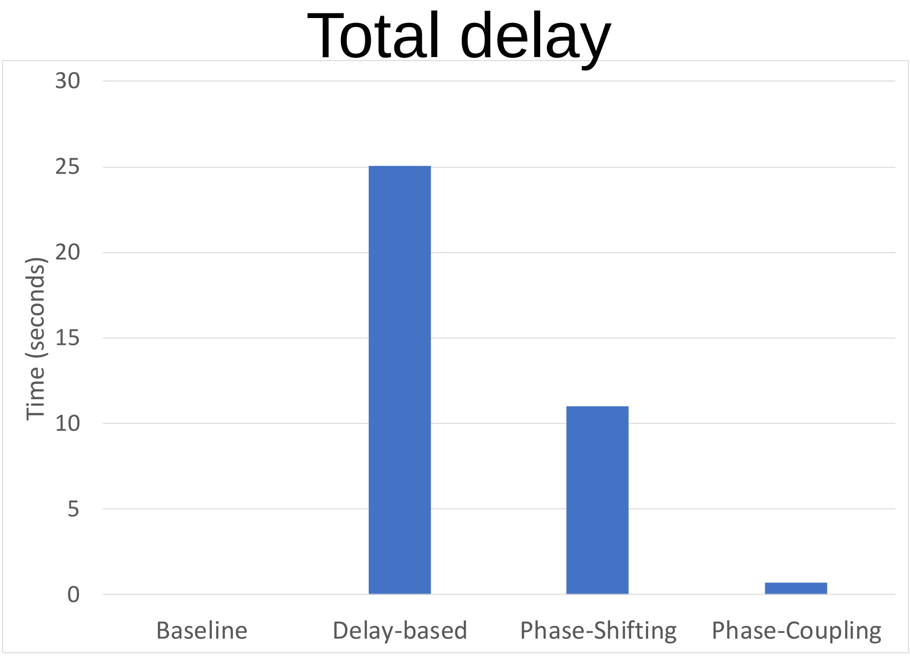
- **Results**
  - Two identical applications executed at the same time.
  - 64 processes



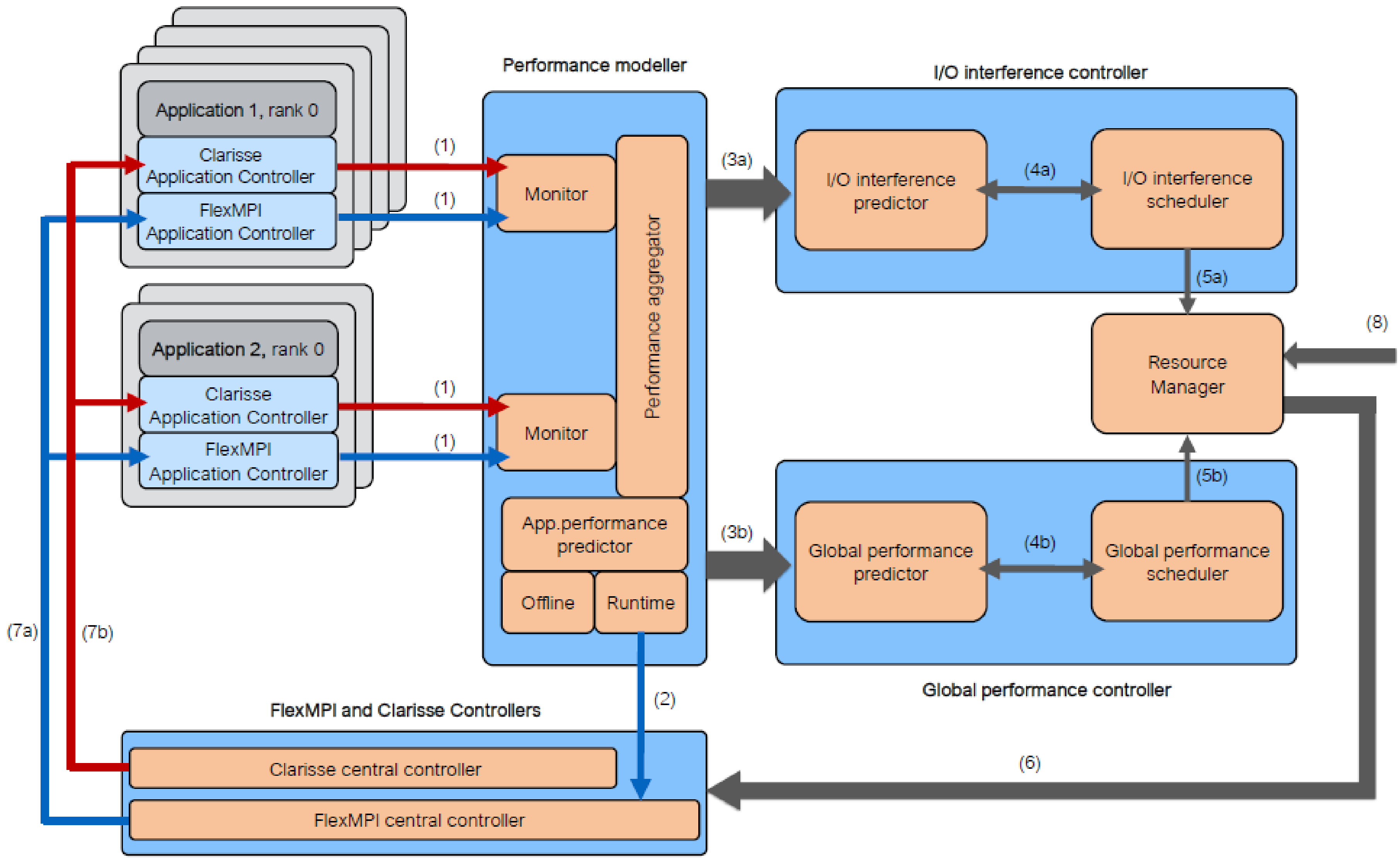
I/O interference avoidance

- **Results**
  - Two different applications executed at the same time.
  - 64 and 50 processes





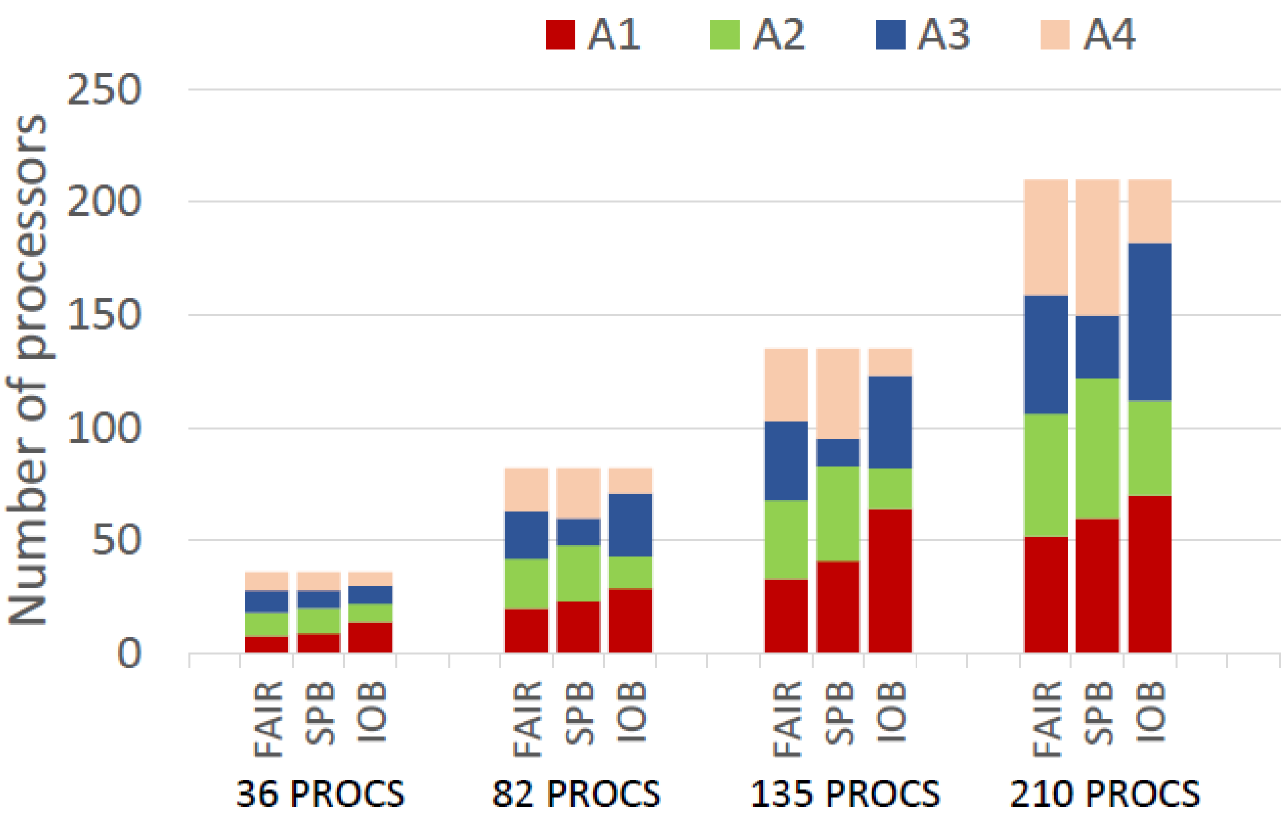
I/O-aware scheduling



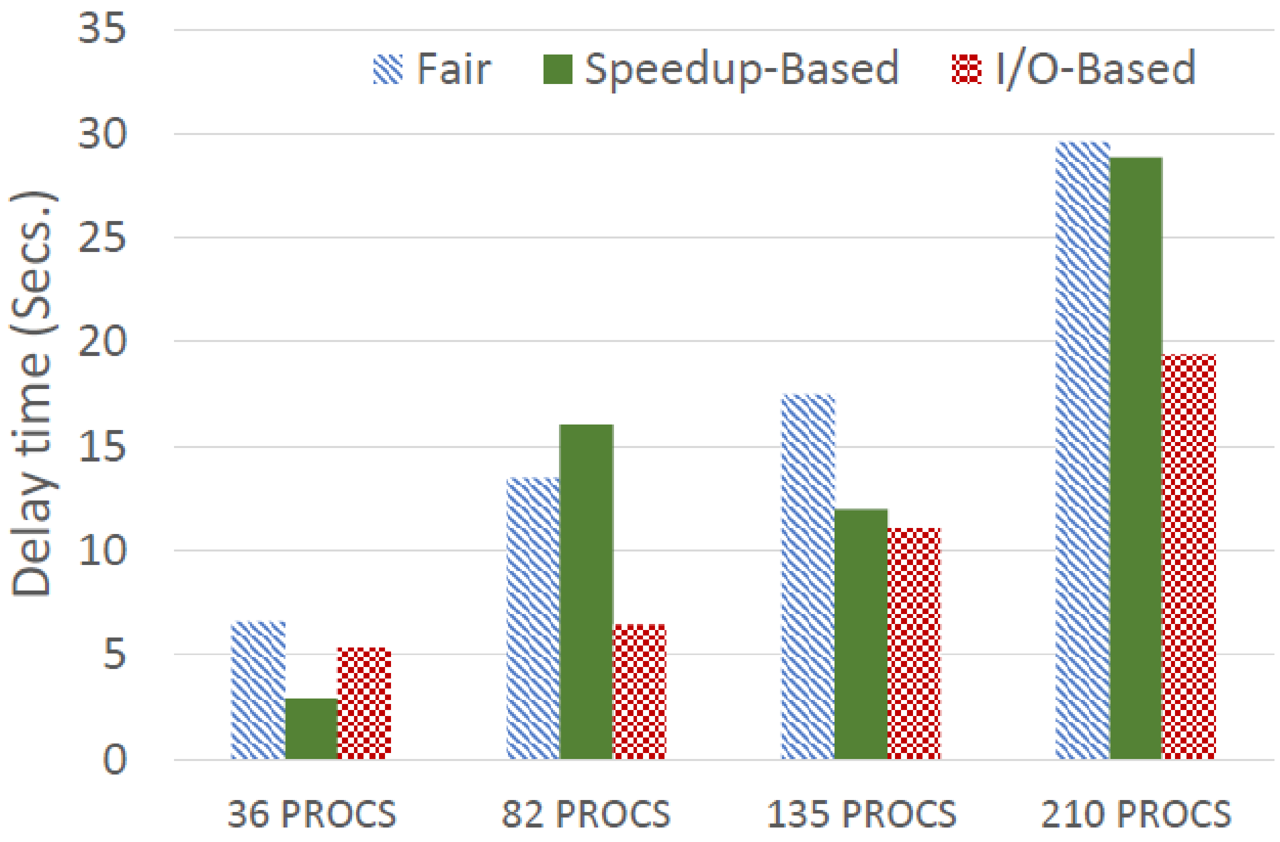
I/O-aware scheduling

- We only consider running applications
- Assign the available processors to the running applications
- Two baseline schedulers:
  - Fair
  - Speedup-based
  - I/O-aware

I/O-aware scheduling



I/O-aware scheduling



Future work

- Extended Clarisse and FlexMPI coordination
- Automated learning techniques for I/O scheduling
- Integration with a system-wide monitoring tool
- Application modelling