RÉPUBLIQUE FRANÇAISE
Liberté Égalité Fraternité

FRANCE 2030
PROGRAMME DE RECHERCHE
NUMÉRIQUE POUR L'EXASCALE

Fault tolerance and task-based programming for large-scale systems

Exa-Soft General Meeting

Nicolas Ducarton (WP 3)

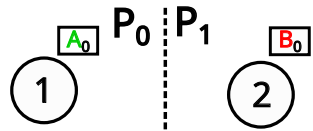
Supervised by
Amina Guermouche, Thomas Hérault, Samuel Thibault

25 September 2025

Inria, LabRI, Université de Bordeaux, Bordeaux INP

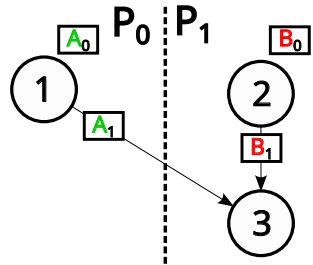
Example DAG

```
void work(){
  starpu_mpi_task_insert(STARPU_RW, A [...]);
  starpu_mpi_task_insert(STARPU_RW, B [...]);









}
```



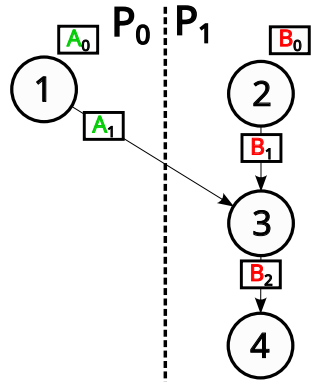
Example DAG

```
void work(){
  starpu_mpi_task_insert(STARPU_RW, A [...]);
  starpu_mpi_task_insert(STARPU_RW, B [...]);


  starpu_mpi_task_insert(STARPU_RW, B,
    STARPU_R, A [...]);









}
```



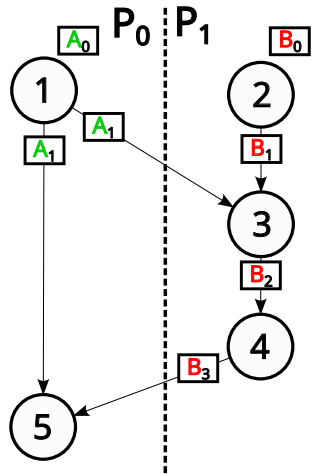
Example DAG

```
void work(){
  starpu_mpi_task_insert(STARPU_RW, A [...]);
  starpu_mpi_task_insert(STARPU_RW, B [...]);


  starpu_mpi_task_insert(STARPU_RW, B,
    STARPU_R, A [...]);
  starpu_mpi_task_insert(STARPU_RW, B [...]);









}
```



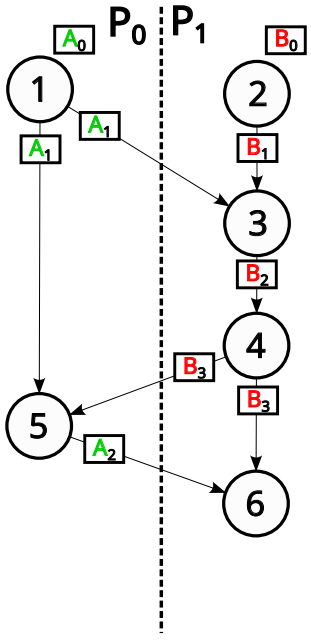
Example DAG

```
void work(){
  starpu_mpi_task_insert(STARPU_RW, A [...]);
  starpu_mpi_task_insert(STARPU_RW, B [...]);


  starpu_mpi_task_insert(STARPU_RW, B,
    STARPU_R, A [...]);
  starpu_mpi_task_insert(STARPU_RW, B [...]);
  starpu_mpi_task_insert(STARPU_RW, A,
    STARPU_R, B [...]);




}
```



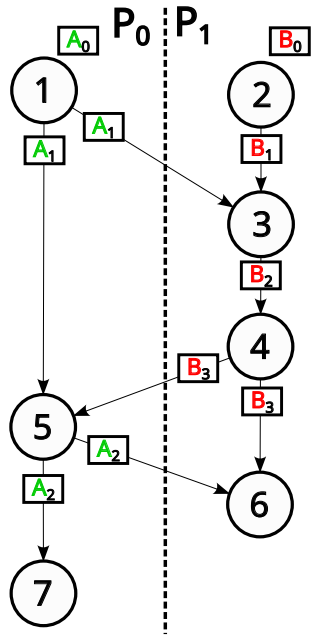
Example DAG

```
void work(){
  starpu_mpi_task_insert(STARPU_RW, A [...]);
  starpu_mpi_task_insert(STARPU_RW, B [...]);


  starpu_mpi_task_insert(STARPU_RW, B,
    STARPU_R, A [...]);
  starpu_mpi_task_insert(STARPU_RW, B [...]);
  starpu_mpi_task_insert(STARPU_RW, A,
    STARPU_R, B [...]);


  starpu_mpi_task_insert(STARPU_RW, B,
    STARPU_R, A [...]);

}
```



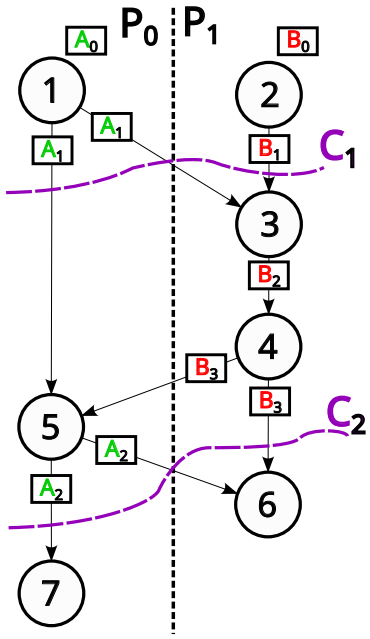
Example DAG

```
void work(){
  starpu_mpi_task_insert(STARPU_RW, A [...]);
  starpu_mpi_task_insert(STARPU_RW, B [...]);


  starpu_mpi_task_insert(STARPU_RW, B,
    STARPU_R, A [...]);
  starpu_mpi_task_insert(STARPU_RW, B [...]);
  starpu_mpi_task_insert(STARPU_RW, A,
    STARPU_R, B [...]);


  starpu_mpi_task_insert(STARPU_RW, B,
    STARPU_R, A [...]);
  starpu_mpi_task_insert(STARPU_RW, A [...]);
}
```

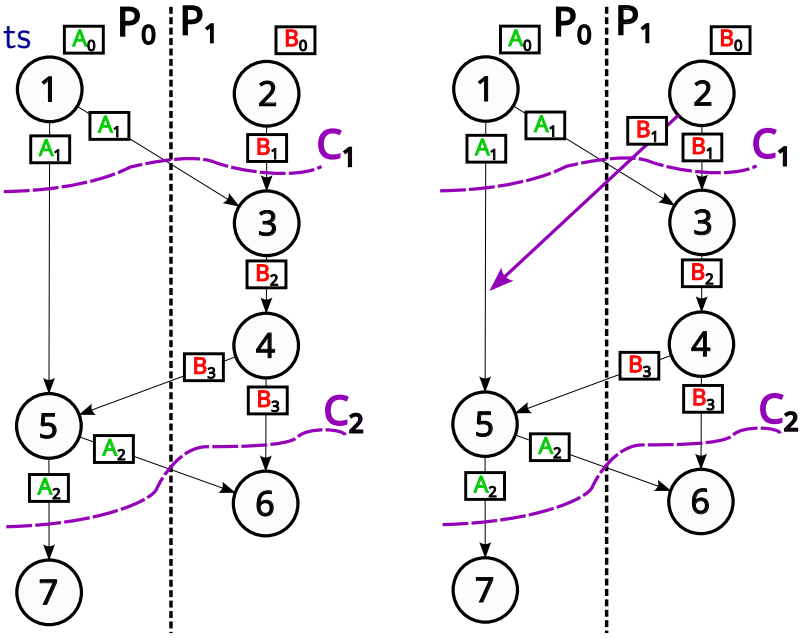


With checkpoints

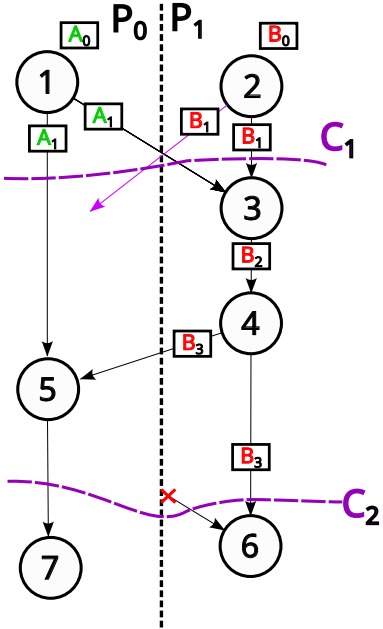
```
void work(){
  starpu_mpi_task_insert(STARPU_RW, A [...]);
  starpu_mpi_task_insert(STARPU_RW, B [...]);
  starpu_mpi_checkpoint_submit(&template,
    priority++);
  starpu_mpi_task_insert(STARPU_RW, B,
    STARPU_R, A [...]);
  starpu_mpi_task_insert(STARPU_RW, B [...]);
  starpu_mpi_task_insert(STARPU_RW, A,
    STARPU_R, B [...]);
  starpu_mpi_checkpoint_submit(&template,
    priority++);
  starpu_mpi_task_insert(STARPU_RW, B,
    STARPU_R, A [...]);
  starpu_mpi_task_insert(STARPU_RW, A [...]);
}
```



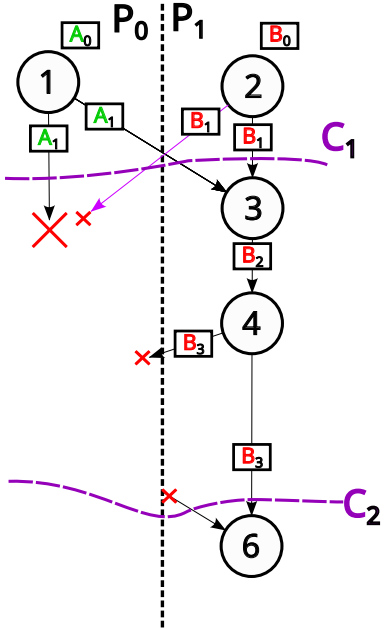
With checkpoints



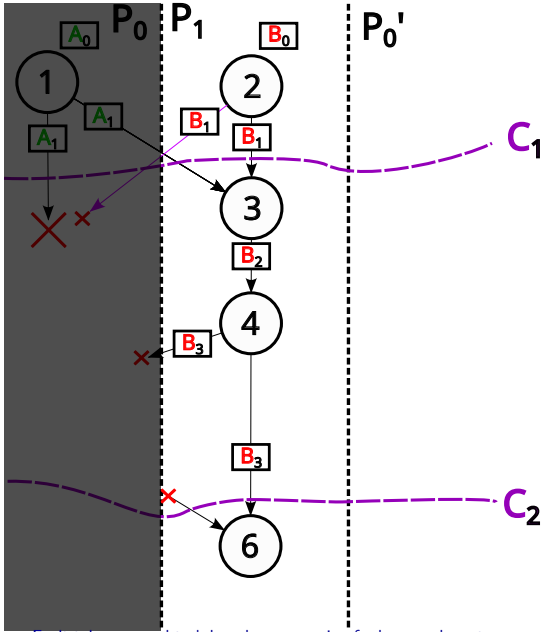
Example



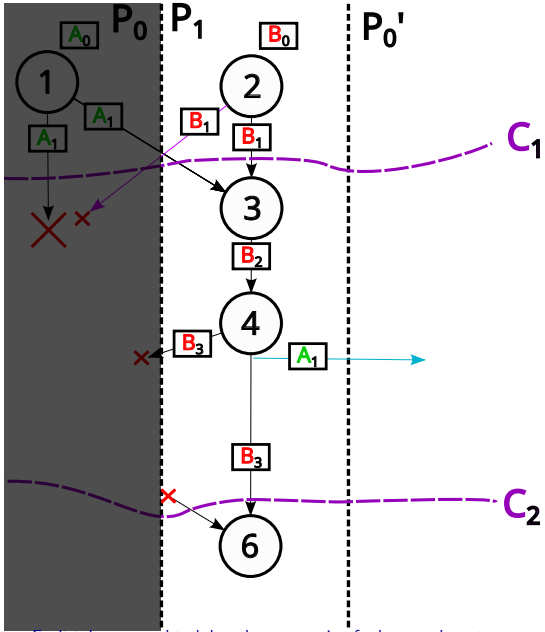
Example



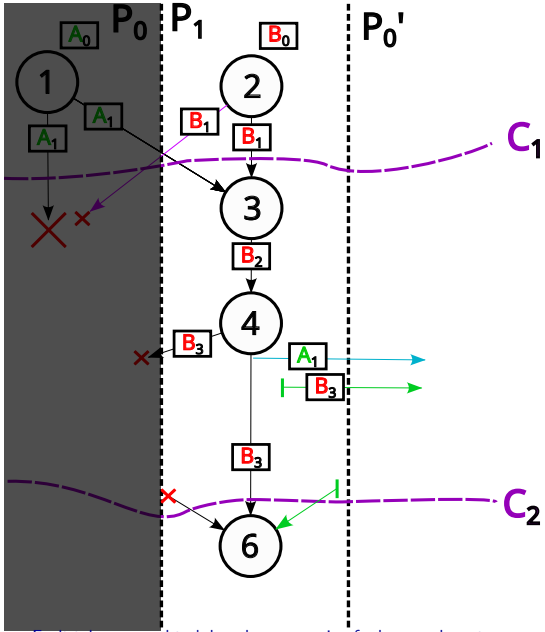
Example



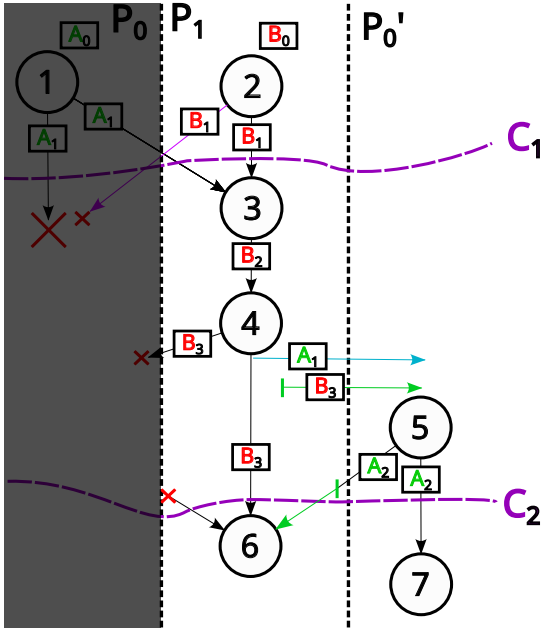
Example



Example



Example



Objectives and Assumptions

Previous work

asynchronous checkpoints in StarPU

Objectives

- Add failure detection and recovery
  - Using ULFM, an extension to the MPI standard
  - With message logging
- Evaluate restart cost
- Design placement strategies

Assumptions

- Only one failure at a time
- A spare node is available

The end

Thank you for listening!
If you have any questions, please ask! (now or during the break).