Locality-Aware Batch Scheduling of Jobs Sharing Input Files

Uppmax meeting

Maxime GONTHIER - Loris MARCHAL - Samuel THIBAULT - Elisabeth Larsson - Carl Nettelblad

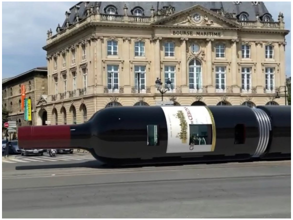
maxime.gonthier@ens-lyon.fr

LIP - ROMA - LaBRI - STORM

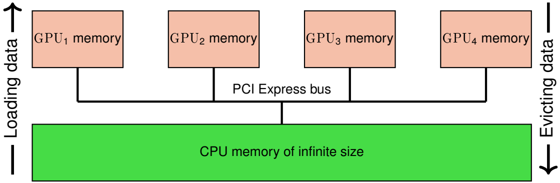
June 13, 2022

Who am I ?

From Bordeaux (France)



PhD on locality-aware scheduling of tasks sharing data on GPUs

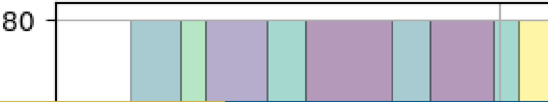


ENS DE LYON université de BORDEAUX

Motivation

- Users may submits tens of hundreds of separate jobs using the same large input files
- Jobs will read inputs directly from a shared file system
- Large files increase the queue times of the next jobs on the node

How can we minimize the amount of transfers between the shared file system and the nodes ?



Framework

Jobset $\mathbb{J}$

- Set of jobs from Rackham's history with an added input file
- A file is shared by consecutive jobs submitted by the same user
- Each job require only one node and between 1 and 20 cores.

Nodes

- Rackham's cluster (128, 256 and 1024 GB nodes)
- A file of size 1024 GB can only be scheduled on 1024 GB nodes
- Bandwidth of 0.1 GB/s for each node
- Each node has $N = 20$ cores

Two different constraints

Constraint 1: Dealing with file re-use

Maximize file re-use

Constraint 2: Dealing with different input files sizes

Allow smaller jobs to be computed on bigger nodes

We define the $flow_{J_i}$ of a job as:

$$flow_{J_i} = Completiontime(J_i) - Subtime(J_i)$$

Obj. : $$minimize \sum_{i=0}^{|\mathbb{J}|} flow_{J_i}$$

Objective: Minimize the mean *flow* stretch

Obj. : $$minimize \frac{\sum_{i=0}^{|\mathbb{J}|} \frac{flow_{J_i}}{Duration(J_i) + \frac{\mathcal{M}(\mathcal{F}(J_i))}{BW}}}{|\mathbb{J}|}$$

A secondary objective is to minimize the amount of file load.

FCFS with a score

2 schedulers from StarPU

Experimental settings

General

- Reduced cluster and set of jobs
- Get a set of jobs from Rackham's history and compute the starting state the day before the day we want to evaluate.

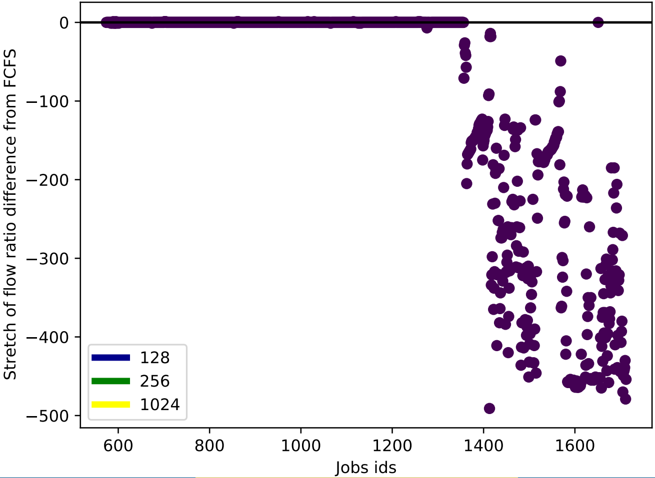
Evaluate file sharing

- Each jobs is using a file

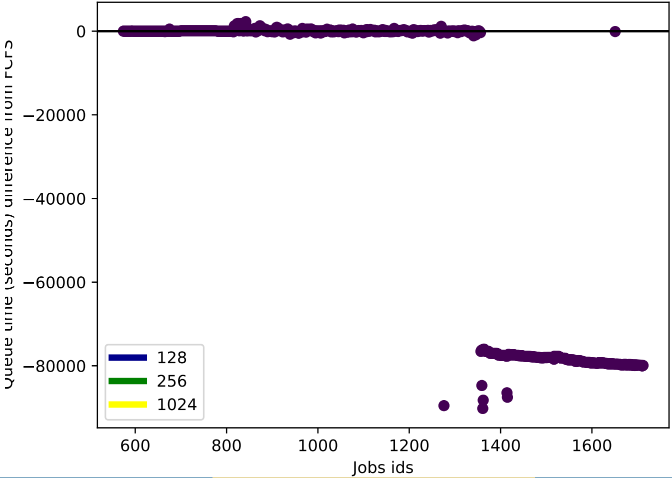
Evaluate size constraints

- Tiled Cholesky decomposition (without dependencies) $(A = L \times L^T)$

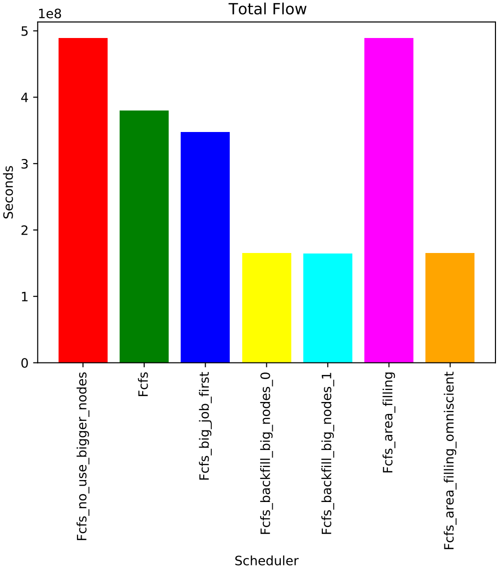
Flow stretch of FCFS with a score compared to FCFS on the file sharing constraint



Queue times of FCFS with a score compared to FCFS on the file sharing constraint



Total Flow on the size constraint



Conclusion and future work

Limiting data movements is crucial to extract the most out of GPUs

Our contribution → **DARTS+LUF, focused on data locality**

DARTS achieves very good performance because it:

- **Limits data transfers** thanks to the finding of an optimal data and an adapted eviction policy
- **Overlaps** communication and computations by distributing transfers over time
- Can be used with a **reduced complexity**

Areas for improvement

- Reduce computational complexity
- Consider tasks with dependencies
- Take inter-GPU communications into account
- Manage multiple MPI nodes