Memory-Aware Scheduling Of Tasks Sharing Data On Multiple GPUs
Maxime Gonthier, Loris Marchal, Samuel Thibault

To cite this version:
Maxime Gonthier, Loris Marchal, Samuel Thibault. Memory-Aware Scheduling Of Tasks Sharing Data On Multiple GPUs. ISC 2023 - ISC High Performance 2023, May 2023, Hamburg, Germany. Lecture Notes in Computer Science. hal-04090595

HAL Id: hal-04090595
https://inria.hal.science/hal-04090595
Submitted on 5 May 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Motivation: Extract peak performance from GPUs

Tesla V100 GPUs with a 12000 MB/s PCI bandwidth
- GPU memory limited to 500 MB to better distinguish the performance even on small datasets
- 10 runs per experiment (deviation < 2%)

We provide a new algorithm (DARTS) and a new eviction policy (LUF) focused on data locality

Areas for improvement:
- Use other out-of-core applications (DARTS is not specific to GPUs)!

Experimental evaluations on independent tasksset

- DMDAR suffers from a conflict between prefetches and evictions
- DARTS+LUF stays close to the asymptotic goal

DMDAR achieves better performances once the memory is a constraint (on the right of the green dotted line) for EAGER, hMETIS and DARTS

Dotted horizontal black line: maximum GFlop/s the GPUs can achieve
Red dotted vertical line: the cumulated GPUs memory can fill exactly one of the two input matrices
Orange dotted line: it can accommodate both input matrices
Black diagonal dotted line: maximum number of transfers that can be done during the minimum time for computation — the hard limitation induced by the PCI bus bandwidth

DMDAR suffers from a conflict between prefetches and evictions
- DMDAR suffers from a conflict between prefetches and evictions
- DARTS+LUF stays close to the asymptotic goal

DARTS clean overlaps communications and computations

DMDAR suffers from a conflict between prefetches and evictions
- DMDAR suffers from a conflict between prefetches and evictions
- DARTS+LUF stays close to the asymptotic goal

DMDAR suffers from a conflict between prefetches and evictions
- DMDAR suffers from a conflict between prefetches and evictions
- DARTS+LUF stays close to the asymptotic goal

Objective 1: Minimize the number of tasks on each GPU
Objective 2: Minimize the amount of data loads on each GPU

Pathological matrix size due to GPU's eviction policy when the memory is a constraint (after the red dotted line) for EAGER, hMETIS and DARTS

DMDAR suffers from a conflict between prefetches and evictions
- DMDAR suffers from a conflict between prefetches and evictions
- DARTS+LUF stays close to the asymptotic goal

DMDAR suffers from a conflict between prefetches and evictions
- DMDAR suffers from a conflict between prefetches and evictions
- DARTS+LUF stays close to the asymptotic goal

Summary and conclusion

Motivations:
- High-performance computing applications, like weather and climate forecasting, have an increasing demand in computer power
- GPUs have a large computation power but have a limited memory. This, coupled with the share of a single PCI express bus by multiple GPUs induce a large amount of data transfers which reduces performance
- Limiting data movement is crucial to extract the most out of GPUs

Results: DARTS+LUF achieves good performance because it:
- Provides a new algorithm (DARTS) and a new eviction policy (LUF) focused on data locality
- 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
- Can deal with priorities

Areas for improvement:
- Manage multiple MPI nodes
- Use other out-of-core applications (DARTS is not specific to GPUs)