Hessisches Kompetenzzentrum
für Hochleistungsrechnen

Parallelizing the Level 2A Processor for Sentinel-2 Satellite Imagery

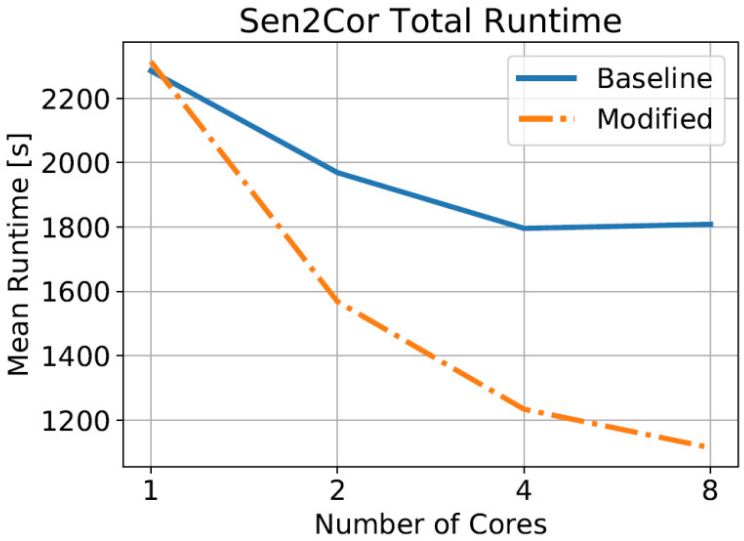

Figure 1: Sen2Cor Total Runtime Comparison.

Introduction

In June 2017, ESA started systematic Level 2A processing of Sentinel-2 acquisitions over Europe using the cloud screening and atmospheric correction processor named Sen2Cor written in Python. Since March 2018, Level 2A products are generated by the official Sentinel-2 Payload Data Ground Segment (PDGS) and are available on the Copernicus Open Access Hub. Although the embedding of Sen2Cor in the processing chain has been proven quite satisfactory and several optimizations have already been performed up to the recent Version 2.8, there is still a desire to improve its runtime performance in order to warrant the availability of Level 2A data with a faster update ability.

Methods

The objective of the proposed project is to develop and test several strategies to improve the runtime of Sen2Cor using ahead-of-time and just-in-time compilation and hybrid parallelization. Shared memory parallelization was achieved through Joblib. Dask is currently used to parallelize tasks and distribute data among workers. Furthermore, a dynamic scheduling of workers through Dask was introduced and successfully tested.

Results

A successful shared memory parallelization through Joblib as

Project Manager
Taylan Özden

Principal Investigator
Dr. Michael Burger

Project Term
2019 - 2019

Clusters
Lichtenberg Cluster Darmstadt

Additional Software
Sen2Cor, Dask, Joblib, SciPy, NumPy, Glymur

Institute
Institute of Scientific Computing

University
Technische Universität Darmstadt

well as a successful distributed memory approach through Dask with dynamic scheduling of workers was implemented. The approaches for shared memory parallelization have shown speedups in critical sections such as the reflectance retrieval. Within the shared memory approaches, multithreaded as well as multiprocessed approaches were tested. For a lower number of cores, the multithreaded approaches showed better results, while multiprocessed approaches had a better scalability.

Discussion

The results demonstrated that the implemented approaches are improving the performance in critical sections. However, the approaches were tested on shared memory and with Dask and its extension to allocate workers dynamically, several distributed memory approaches are ready to be extended and implemented. Performance improvements in several critical sections are expected.

Last Update: 2023-03-16 01:31