Towards a Parallel Benchmark for Space Applications: Distributing OBPMark's Image Processing

Mahmoud M. Elbarrawy, Carlos Gonzalez Cortes, Andreas Lund, Daniel Lüdtke
German Aerospace Center (DLR), Institute for Software Technology

Introduction

Modern space applications require high computing power and high reliability from on-board processors. To meet these requirements, the German Aerospace Center (DLR) is developing Scalable On-board Computer for Space Avionics (ScOSA) with a distributed non-shared memory architecture [1].

As performance is an important criterion in the selection of hardware for space missions, the European Space Agency has published an open source benchmark suite called OBPMark [2]. It is a set of benchmarks based on typical space applications and designed to measure system-level performance. However, there is currently no standard tool for evaluating the performance of distributed on-board computers.

We propose a parallelization strategy for running the OBPMark image processing benchmark on a distributed on-board computer. We used a split-map-reduce model as shown in Fig. 1 to integrate the OBPMark #1.1 Image Calibration and Correction benchmark into the ScOSA system.

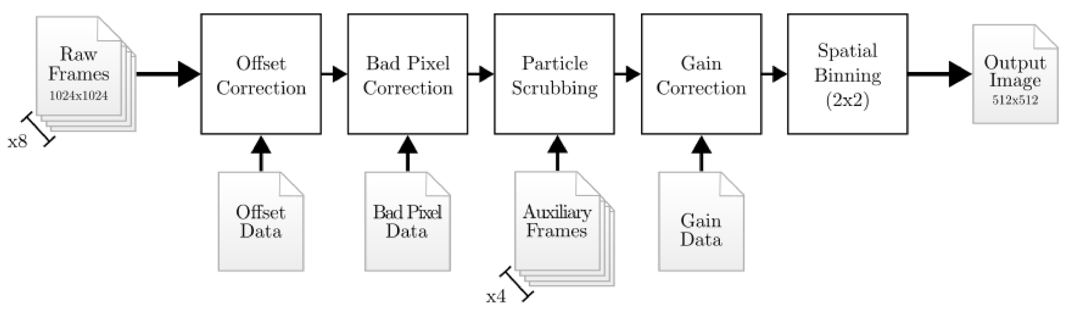

Figure 2. OBPMark #1.1 Image Calibration and Correction benchmark pipeline [3].

Distribution Strategy

Fig. 2 shows the pipeline implementation steps of the benchmark. To benchmark the performance of ScOSA, we decided to distribute the input frames into sub-frames for the pipeline and run the pipeline on all available nodes. However, there are dependency challenges in this approach.

In the Bad Pixel Correction step, the pixel is corrected by a mask of average good neighboring pixels. The mask size is 3x3 pixels. To handle the corners and edges of the frame, the mask size changes to 2x2 for corners, 3x2 for top/bottom edges, and 2x3 for left/right edges. Thus, in the newly constructed sub-frames, the bad pixel correction will behave as if it were a full frame, correcting the newly constructed edges in each sub-frame that differs from the original benchmark.

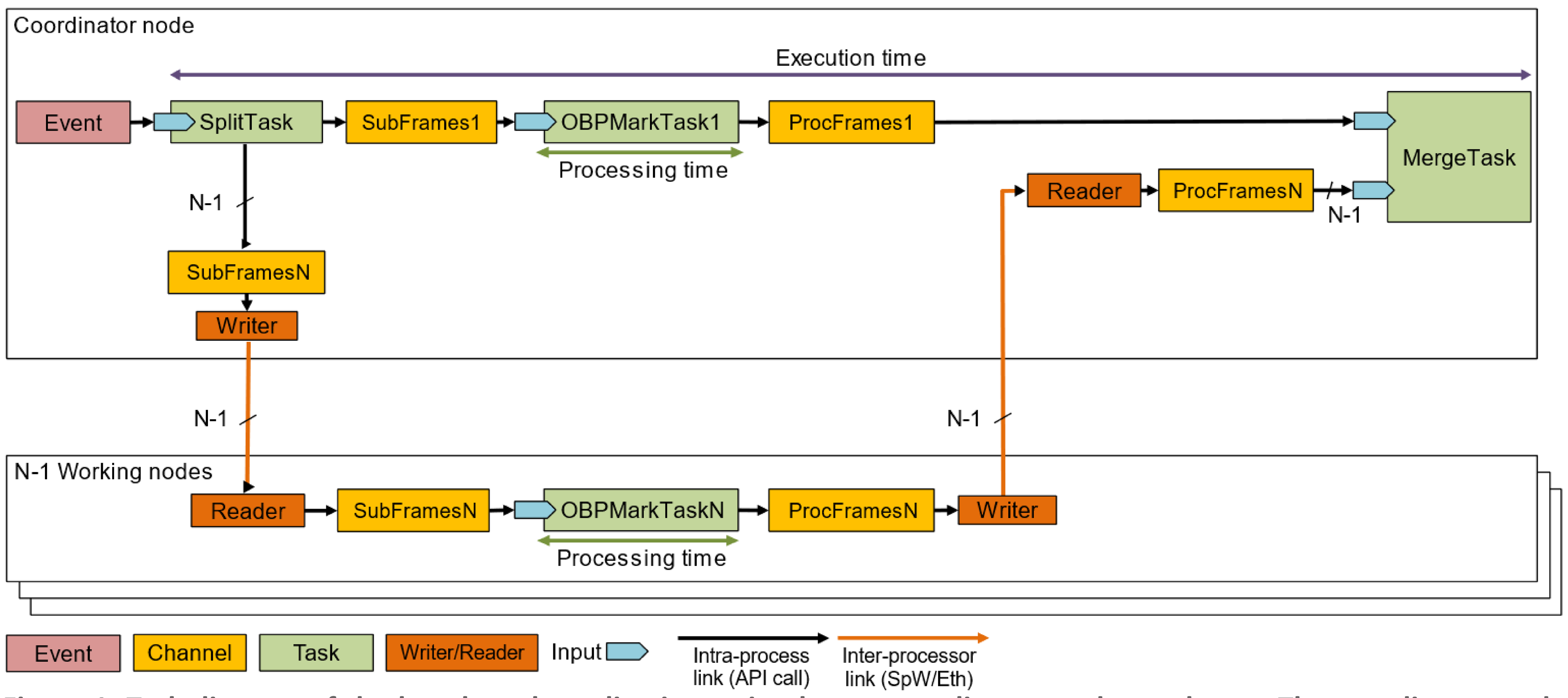

Figure 1. Task diagram of the benchmark application to implement a split-map-reduce scheme. The coordinator node splits the input frame *(SplitTask)* and sends N sub-frames to the remote node's map tasks *(OBPMarkTaskN)*. The coordinator node also processes a sub-frame *(OBPMarkTask1)*. The partial results are sent back to the coordinator node for merging *(MergeTask)* and generating the output image. An event triggers the execution of the application.

In addition, the Spatial Binning step requires that the height of the input frame be an even value so that it can be shrunk by half for the newly constructed sub-frames.

To overcome such challenges, we consider an overlapping area when splitting the subframes to preserve information about neighboring pixels, as shown in Alg. 1 and Fig. 3.

Algorithm 1 Splitting

Input: frame (F), frame height (H), divisions ($N_{divisions}$)
Output: list of sub-frames (SubFrame$_i$)

$S \leftarrow H/N_{divisions}, \quad R \leftarrow H - (S * N_{divisions})$
for $i < N_{divisions}$ **do**
  $k_{up} \leftarrow 2, \quad k_{down} \leftarrow 2$
  if $i == 0$ **then**
    $k_{up} \leftarrow 0$
  else if $i == N_{divisions} - 1$ **then**
    $k_{down} \leftarrow R$
  end if
  $\text{HeightSlice}_i \leftarrow [S * i - k_{up} : S * (i+1) + k_{down}]$
  $\text{SubFrame}_i \leftarrow F_{[\text{HeightSlice}_i, \text{width}]}$
end for

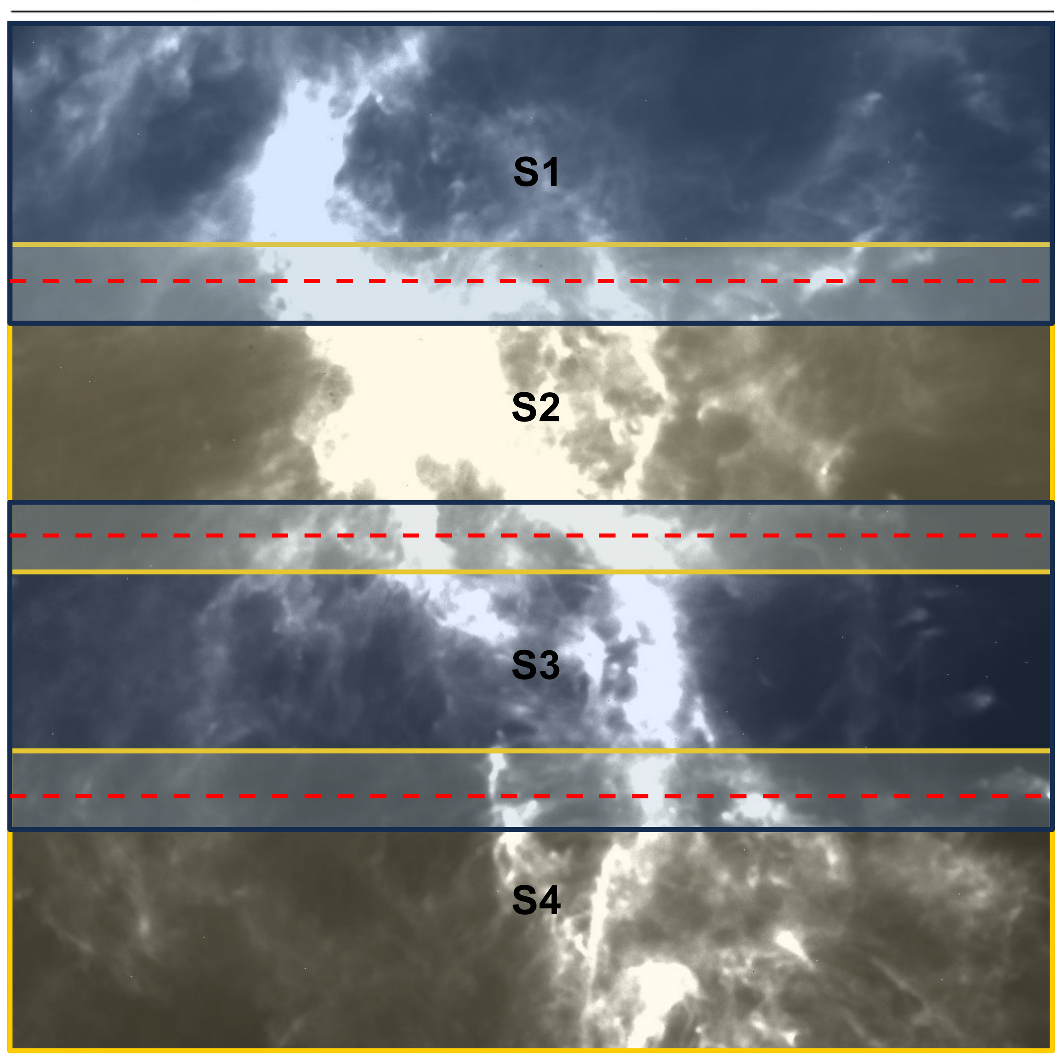

Figure 3. Overlapping splitting of one frames into 4 sub-frames. The red dashed lines represent the fair split without overlapping. The algorithm then creates an overlapping region for each sub-frame.

Evaluation Setup

Scalability tests were performed on 5 Xilinx Zynq-7020 SoCs, each with a dual-core ARM Cortex-A9 processor @ 886 MHz and 1GB of RAM. The 5 nodes are connected via Gigabit Ethernet.

Results

The results in Fig. 4 show a reduction of the benchmark execution time from 9.0 to 2.8 seconds using 5 nodes with a speedup of 3.2x and a processing capability of 0.37 Mpixel/s for the single-core test. In the multi-core test case with 4 nodes using 2 CPU cores per node, the execution time was reduced from 9.0 to 2.5 seconds with a speedup of 3.7x and a processing capability of 0.47 Mpixel/s. We concluded that OBPMark can be used to evaluate the performance of distributed on-board computers with non-shared memory architectures and contributes to the standardization of performance evaluation in the space domain. In future work, we plan to distribute other benchmarks from the OBPMark suite.

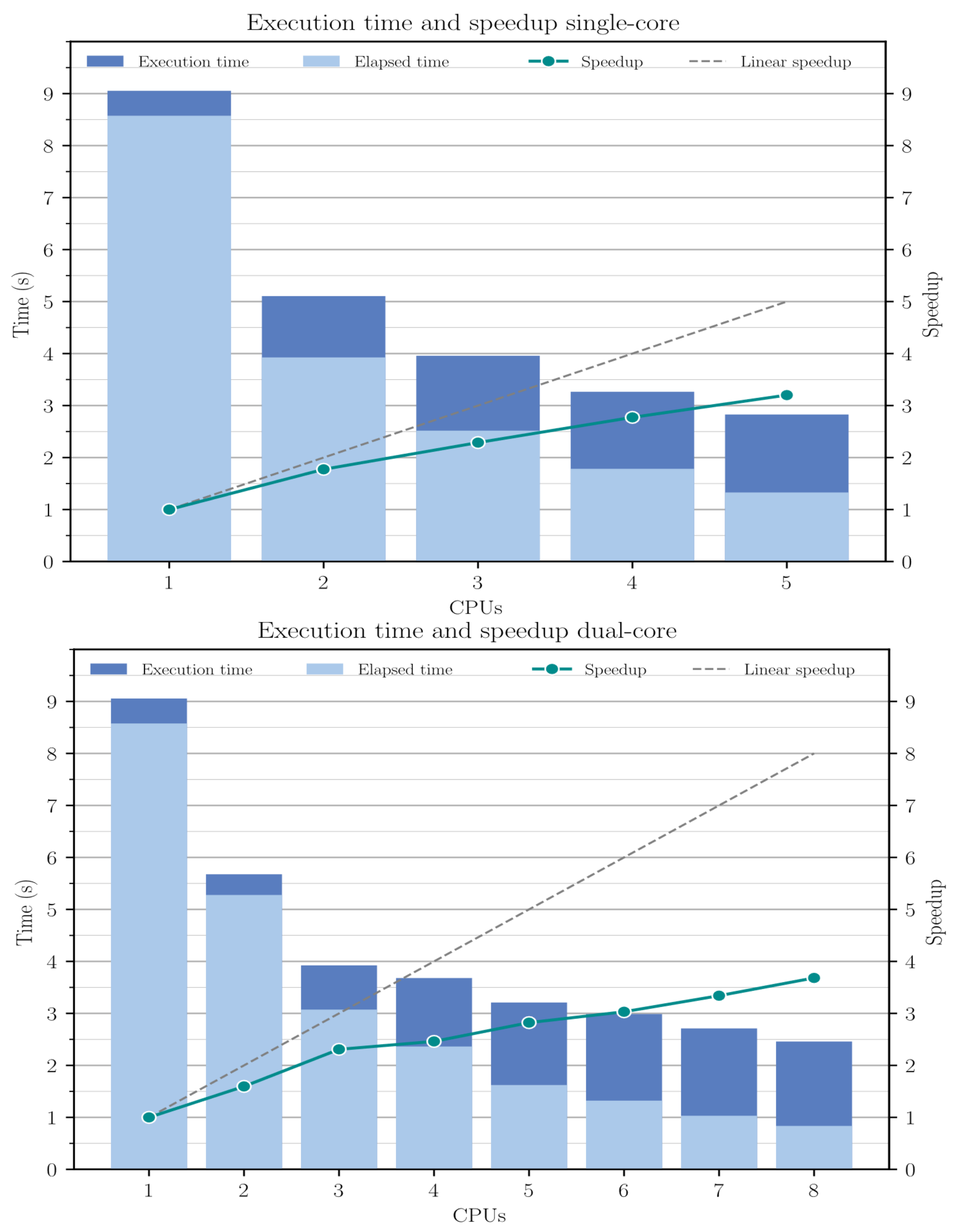

Figure 4. ScOSA OBPMark #1.1 Image Calibration and Correction benchmark results.

[1] Lüdtke et. al, "ScOSA on the way to orbit: Reconfigurable high-performance computing for spacecraft," in *2023 IEEE Space Computing Conference (SCC).*
[2] Steenari et. al "OBPMark (on-board processing benchmarks) – open source computational performance benchmarks for space applications," in 2nd European Workshop on On-Board Data Processing (OBDP2021), 2021.
[3] European Space Agency (ESA). Maintained by: David Steenari., "OBPMark (on-board processing benchmarks) repository," https://github.com/OBPMark/OBPMark.