

Acceleration Techniques for Motion Estimation Algorithms Using Parallel and Distributed Computing

28th Southern Symposium on Microelectronics

Jonas Meinerz, Eduarda Monteiro, Felipe Sampaio
Sergio Bampi, Altamiro Susin

jcmeinerz@inf.ufrgs.br

Outline

- Introduction
- Motion Estimation Overview
 - Full Search Algorithm
- Our Contribution
 - OpenMP Parallel Full Search
 - MPI+OpenMP Parallel and Distributed Full Search
 - Results
- Conclusion

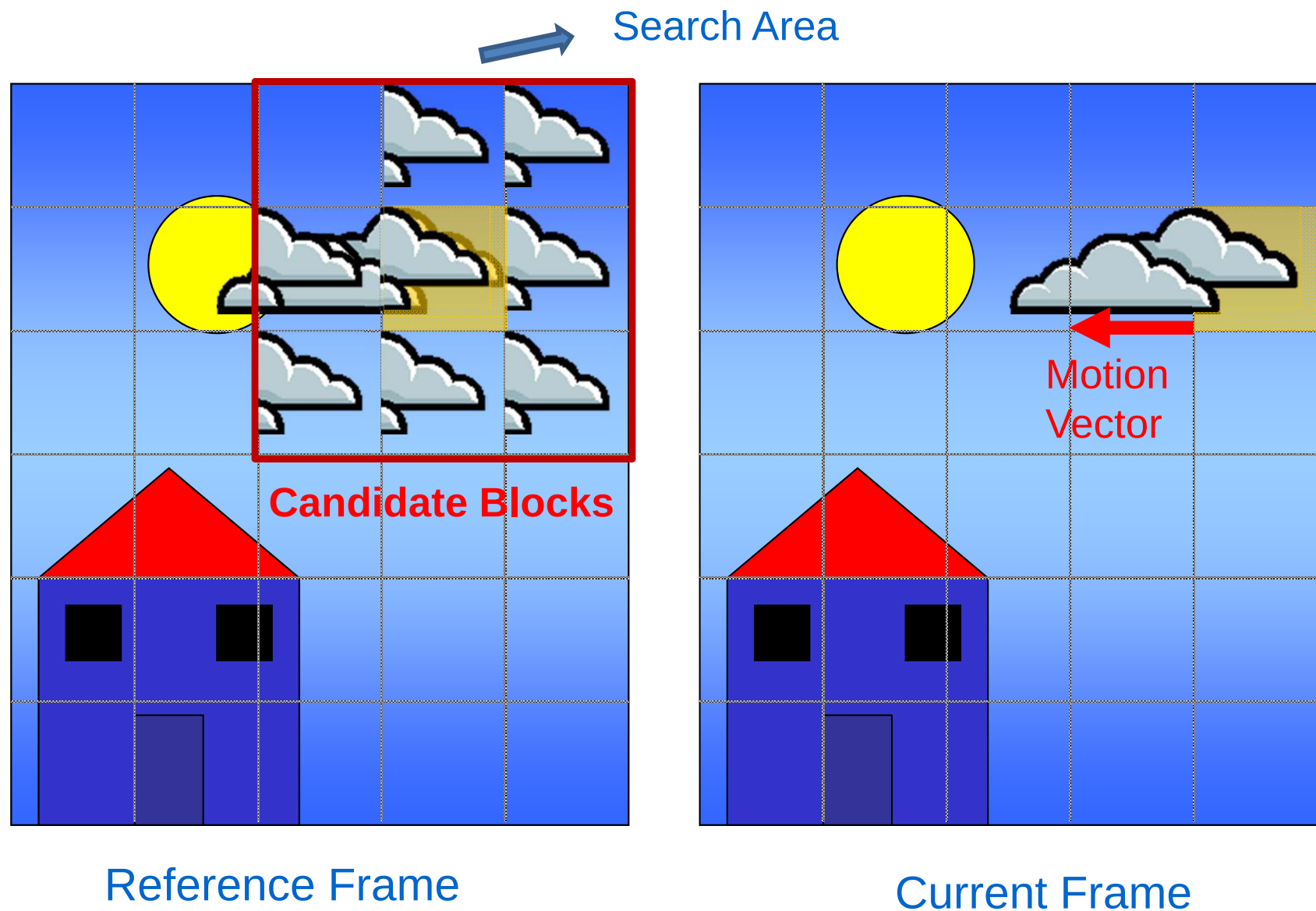
Introduction

- Motion Estimation (ME):
 - **Core** procedure of video coding;
 - Most **complex** and **rewarding** part of video coding;
- Challenges (**focus of this work**):
 - Combine *different* **parallel** computing paradigms to achieve faster ME;
 - OpenMP (shared memory) and MPI (distributed memory);

Motion Estimation (ME)

- Given a block, ME's goal is to find its **best match**;
- How ME works:
 - Compares an uncoded block to previously coded ones within a **search area**;
 - The search area is defined by a **block-matching algorithm**;
 - The best match is chosen following a **similarity criterion**;
- The given block will then be represented by a motion vector (of its best match) and a residue;

Motion Estimation (ME)



Motion Estimation (ME)

- **This work:**
 - Block-matching algorithm: **Full search**;
 - Similarity criteria: *Sum of absolute differences (SAD)*;

The Full Search (FS) Algorithm

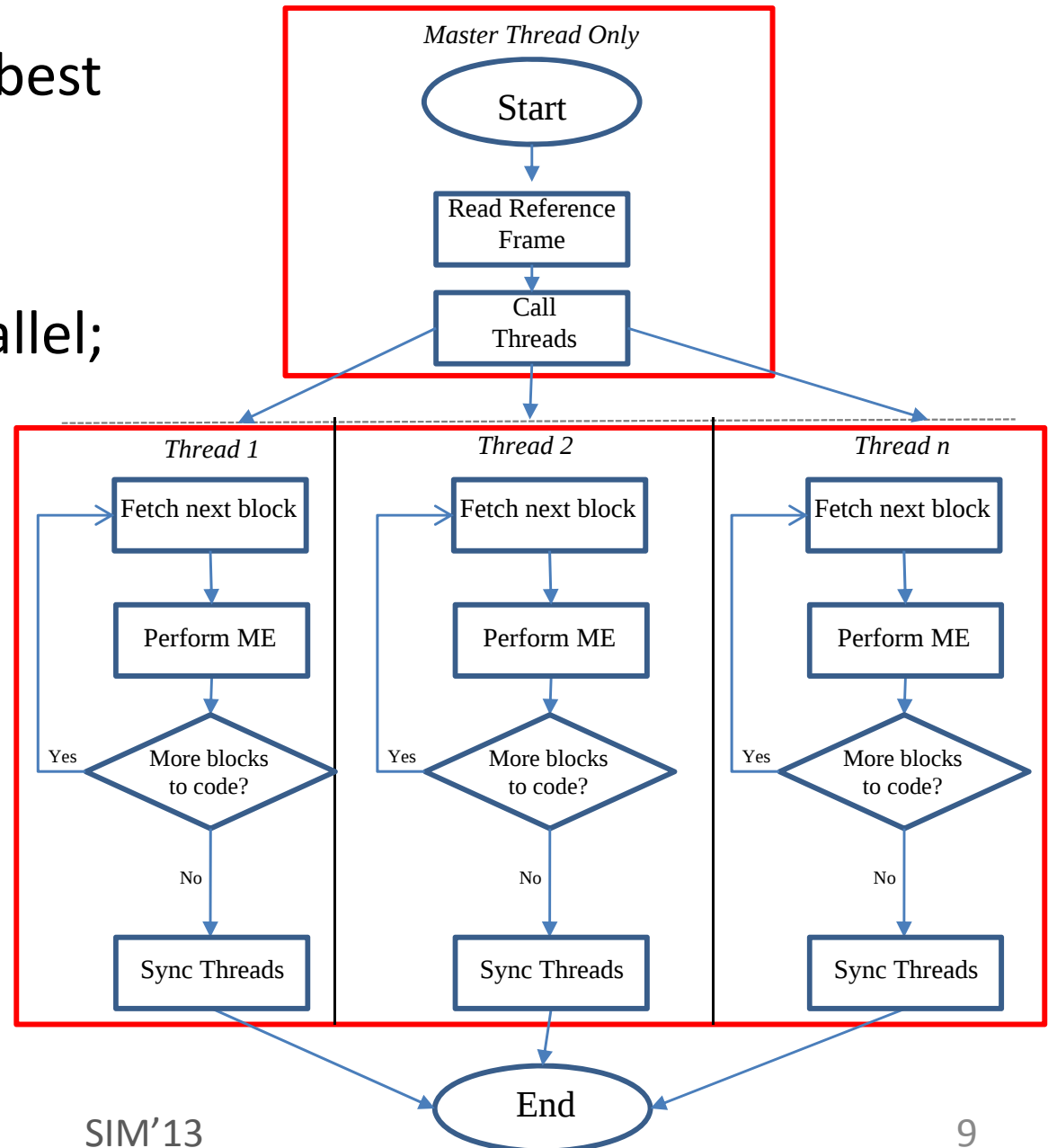
- How FS works:
 - **Exhaustively** compares the blocks within a search area to the selected block;
- Reasons for choosing the FS:
 - **Optimal**;
 - **High level of data independence**;
 - Favors parallelism.

Contributions

- This work's contribution is a **hybrid** *parallel and distributed* implementation of the FS algorithm.
- We developed:
 - A **serial** version of the FS algorithm;
 - An **OpenMP** based parallel version of the FS;
 - An **OpenMP + MPI** hybrid version of the FS, which we compare to our other versions;

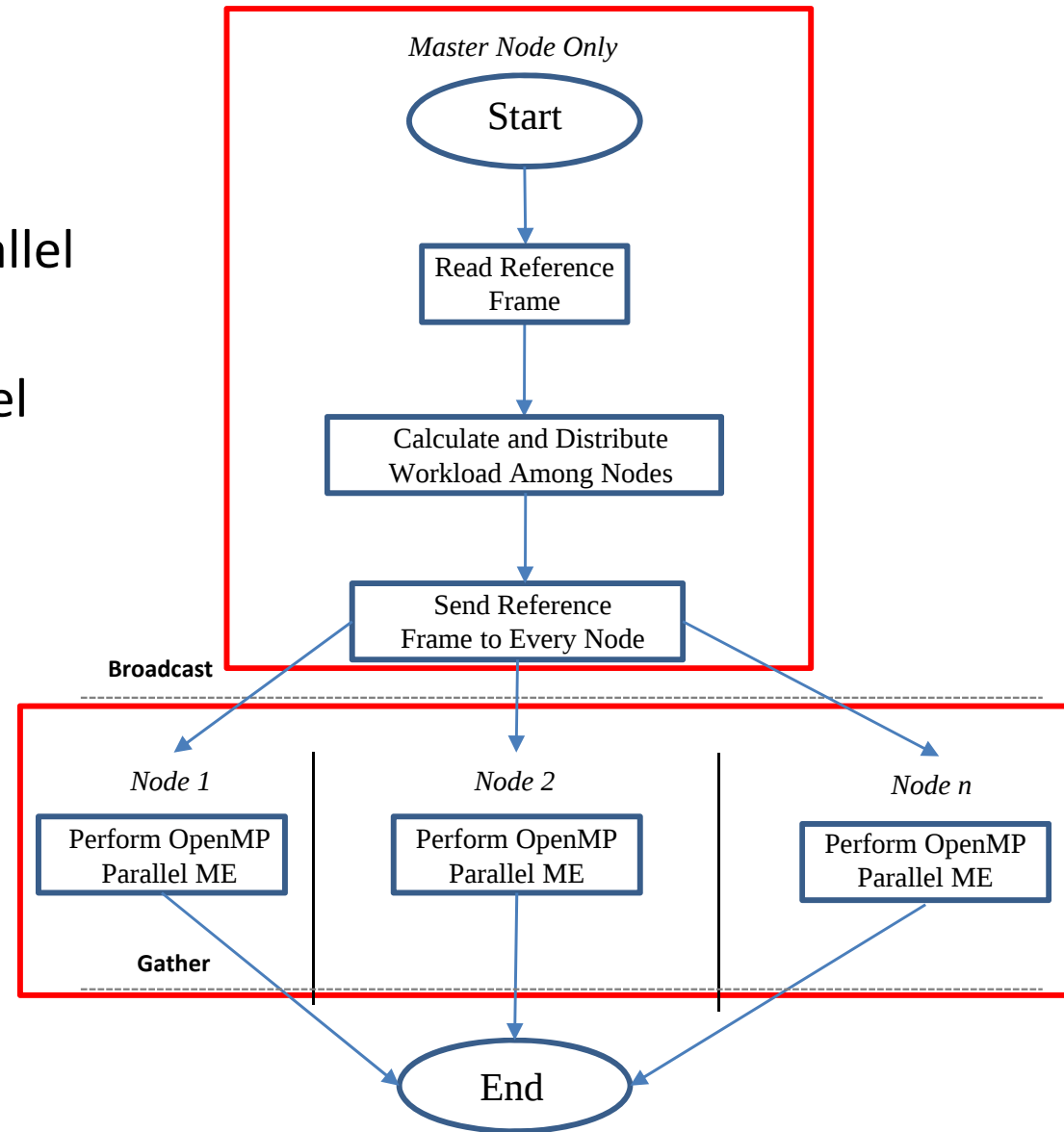
OpenMP Parallel FS

- Different blocks seek for its best match at the same time;
- OpenMP threads run in parallel;



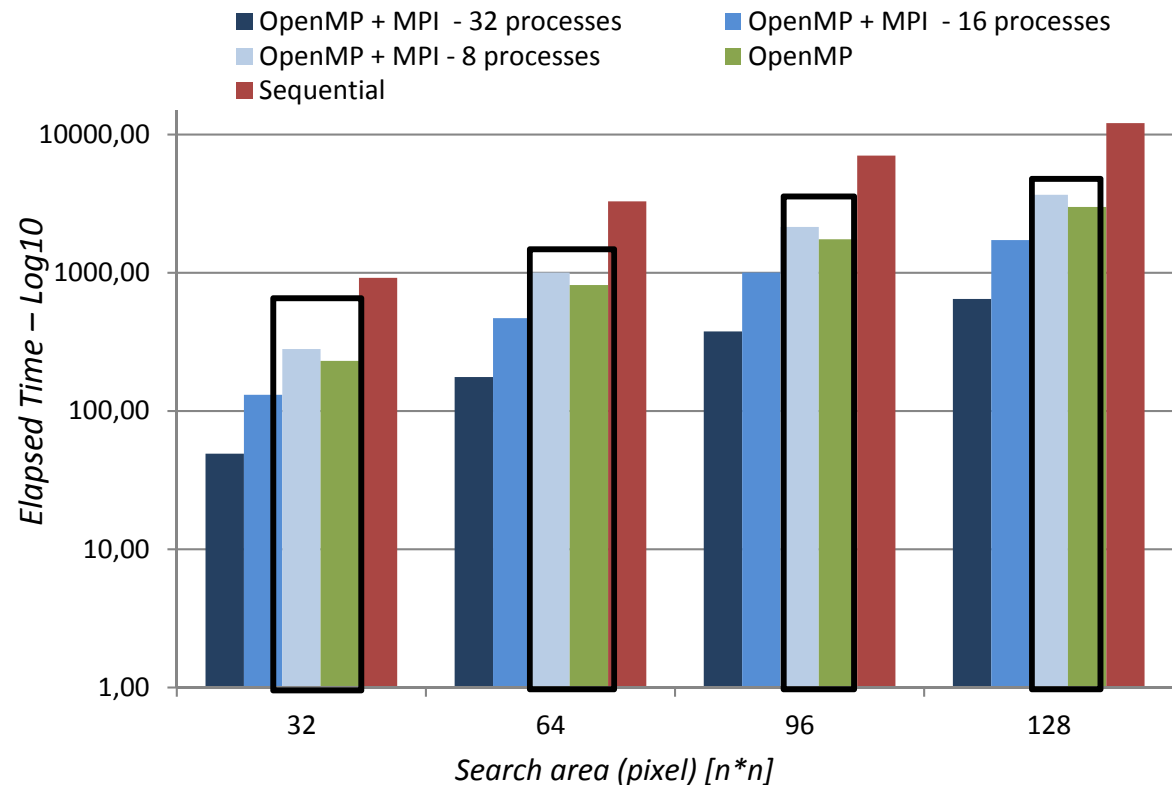
OpenMP + MPI Hybrid FS

- Another level of parallelism is added:
 - The nodes of a cluster run in parallel (MPI)
 - Each node's threads run in parallel (OpenMP)



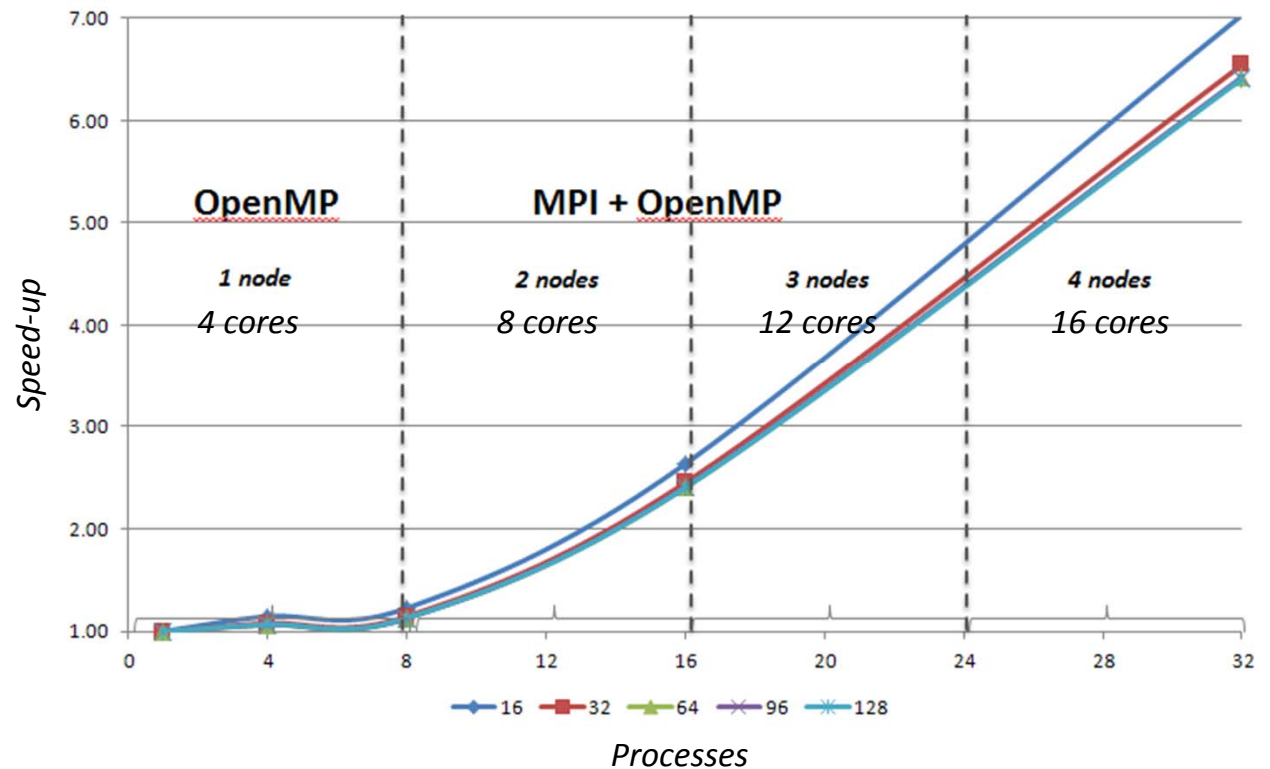
Results – Elapsed Time

- The **hybrid** version (32 proc.) *saves* about 95% of time compared to the serial version and about 75% compared to the OpenMP version;
- The **OpenMP** only version is faster than the **hybrid** version when both run 8 processes.



Results – Speed-up

- The **hybrid** version presents gains up to 7x when compared to the serial version and 4x when compared to the OpenMP version in terms of speed-up;



Conclusion

- The literature does not present implementations combining distributed and parallel computing paradigms;
- The Motion Estimation was successfully accelerated by combining the parallel computation with shared memory and a distributed system;

Acceleration Techniques for Motion Estimation Algorithms Using Parallel and Distributed Computing

28th Southern Symposium on Microelectronics

Jonas Meinerz, Eduarda Monteiro, Felipe Sampaio
Sergio Bampi, Altamiro Susin

jcmeinerz@inf.ufrgs.br