PARALLEL CONVERSION OF SATELLITE IMAGE INFORMATION FOR A WIND ENERGY GENERATION FORECASTING MODEL

ANDRÉS FLEVARIS
GERMÁN GADEA
JUAN SOUTERAS
SERGIO NESMACHNOW
ALEJANDRO GUTIERREZ
GABRIEL CAZES

Facultad de Ingeniería
Universidad de la República
Uruguay

INTRODUCTION

- Wind predictions to analyze availability of wind energy
- The implemented algorithm was designed to update at global scale the land cover information used by the WRF climate model
  - Characteristics of the implemented algorithm
  - Performance analysis and comparisons between two parallel approaches
- Performance analysis of the implemented parallel algorithm obtains linear speedup
- Contribution:
  - Efficient parallel algorithm for the problem of converting satellite imagery in binary files

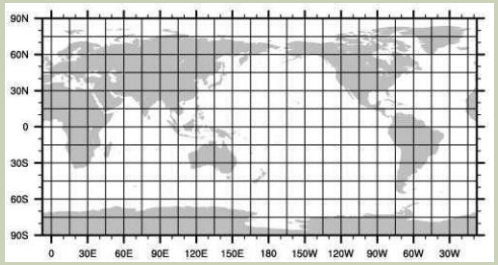
THE PROBLEM

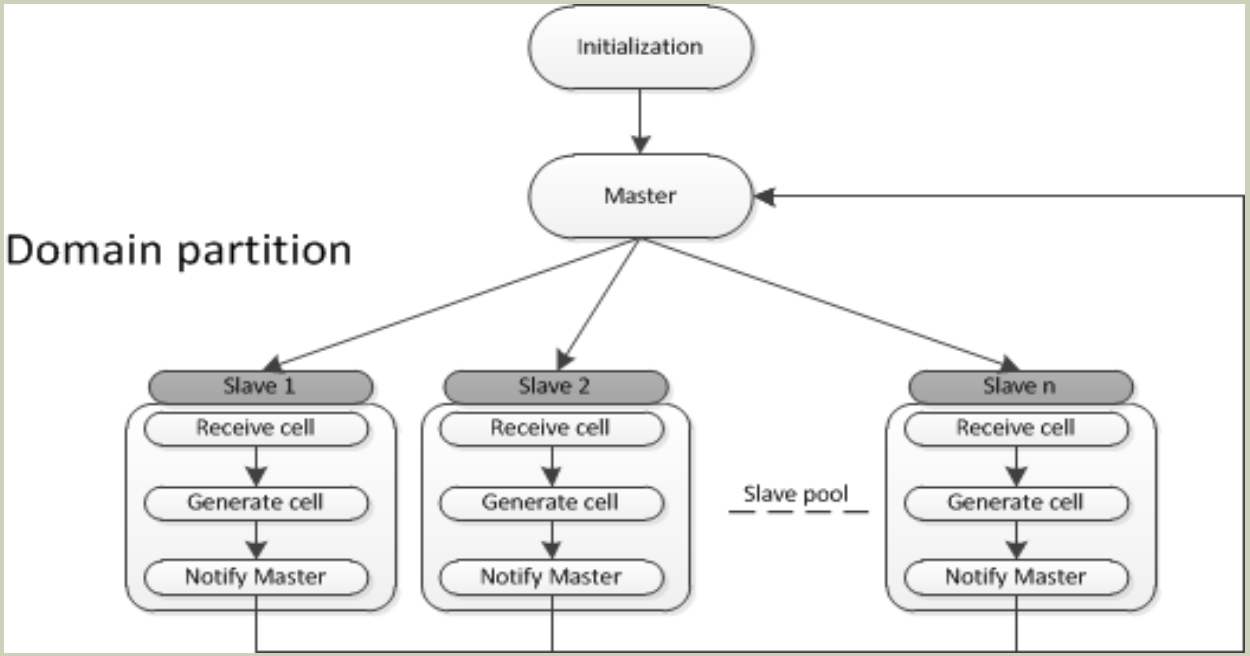
- Wind predictions using the WRF model: used to analyze the availability of wind energy

- Soil information is very relevant for wind forecasting using WRF
- This data is outdated
  - Soil data is stored in files using a proprietary binary format
- Satellite imagery is used to update the soil information
  - Convert satellite images to the binary format used in WRF
  - This information source covers all the world surface in 300 images
  - Total of 27 GB
- A sequential algorithm for image processing demands about 28 hours

PARALLEL ALGORITHM DESIGN

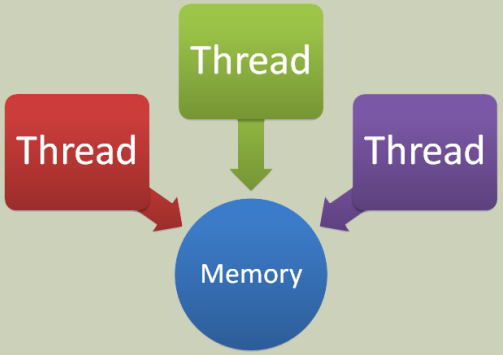
- Parallel model:
- Output domain decomposition
  - Output grid 72x36 cells
- On demand dynamic load balancing



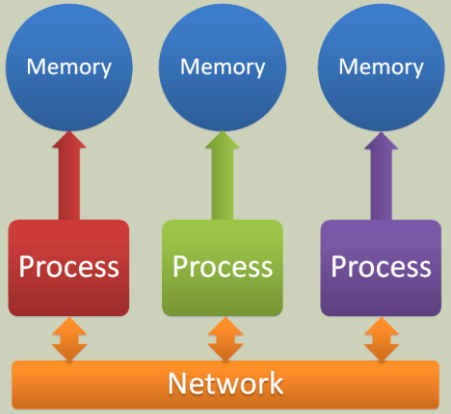


TWO PARADIGMS: SHARED & DISTRIBUTED MEMORY

- Shared memory
  - Implemented with pthreads
  - Main procedure
  - Master thread
  - Pool of slave threads
  - Synchronized using mutexes



- Distributed memory
  - Implemented with MPI
  - Master process
  - Slave processes



- Not hybrid approach

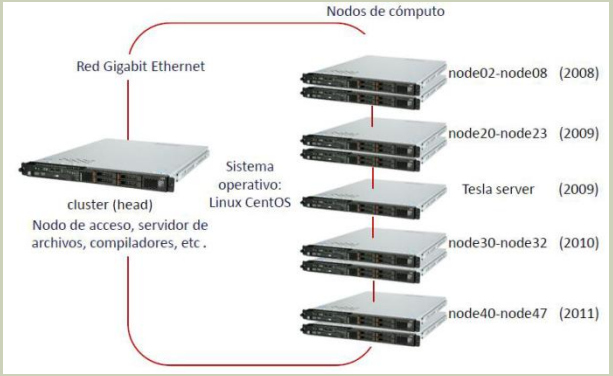
- Both implementations are operative in Cluster FING, UdelaR

TWO PARADIGMS: SHARED & DISTRIBUTED MEMORY

- Both algorithms divided in two phases
- Phase 1
  - Main procedure initializes all the threads/processes and data structures
- Phase 2
  - First master thread/process assigns one cell to each slave
  - Then executes a loop waiting for slaves answers and assigning new cells until all cells are generated
  - Slaves receives cells and process them

EXPERIMENTAL EVALUATION

- Executions carried out in a homogeneous platform: Cluster FING
  - Servers with two Intel quad-core Xeon processors at 2.6 GHz
  - 8 GB RAM
  - CentOS Linux
  - Gigabit Ethernet



http://www.fing.edu.uy/cluster

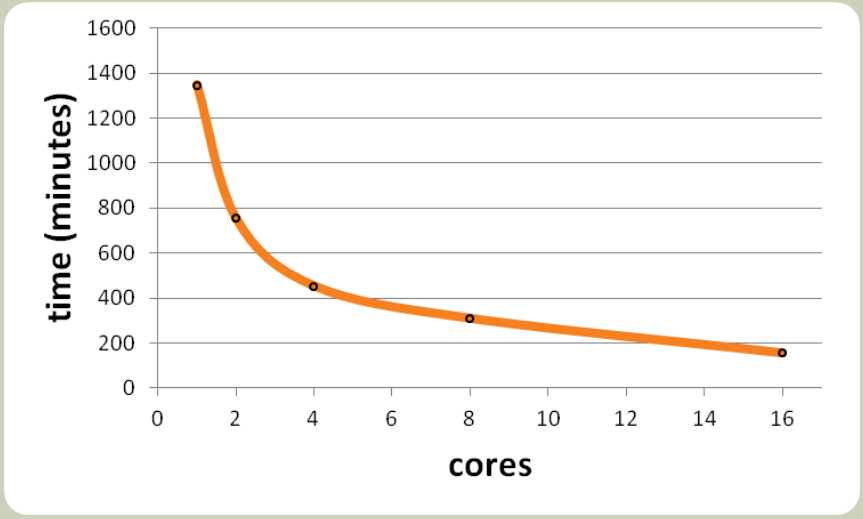
- Performance metrics
  - Execution time
  - Speedup
  - Computational efficiency
  - Scalability

EXPERIMENTAL EVALUATION

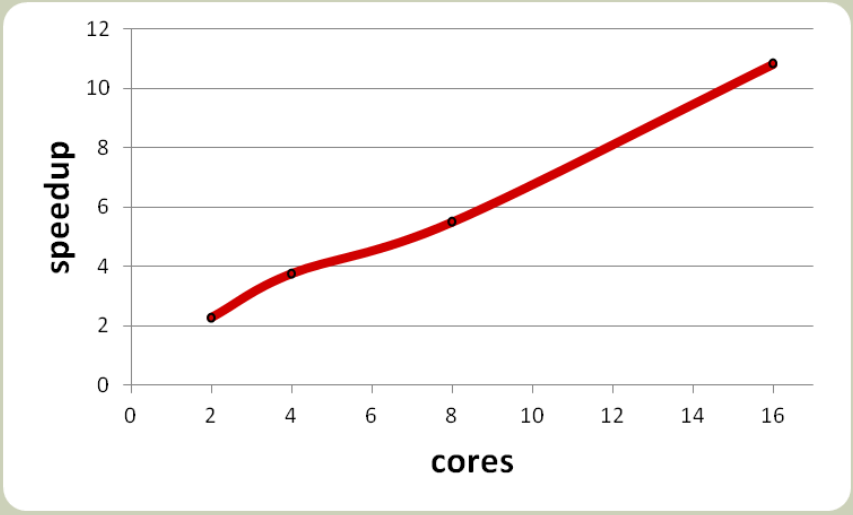
SHARED MEMORY

# Cores	Time	Speedup	Comp. Eficiency	Scalability
1	1349,00	1,27	1,27	1,00
2	756,00	2,26	1,13	1,78
4	456,00	3,75	0,94	2,96
8	311,67	5,49	0,69	4,33
16	158,00	10,82	0,68	8,54

- Execution time



- Speedup



- Poor performance causes: I/O performance and data bus access

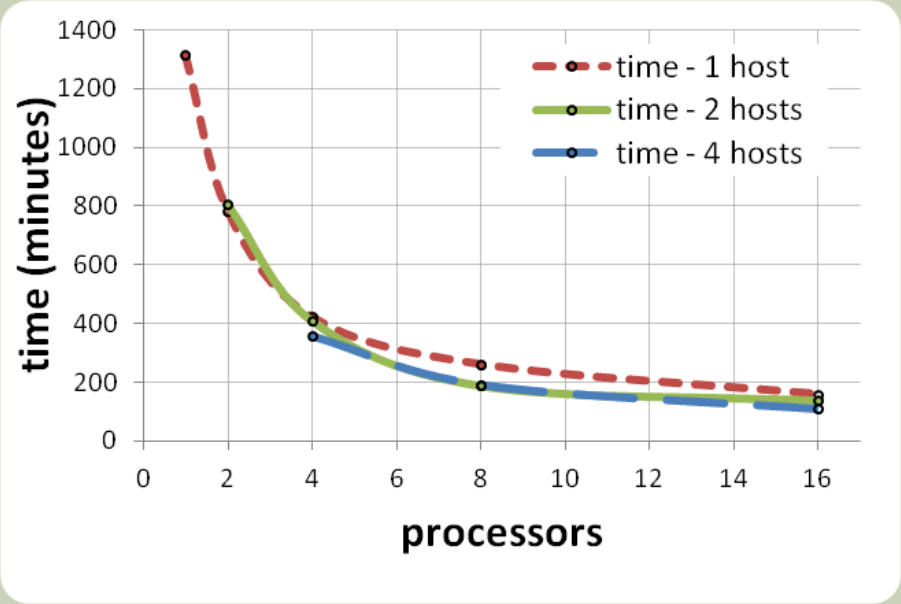
EXPERIMENTAL EVALUATION

DISTRIBUTED MEMORY

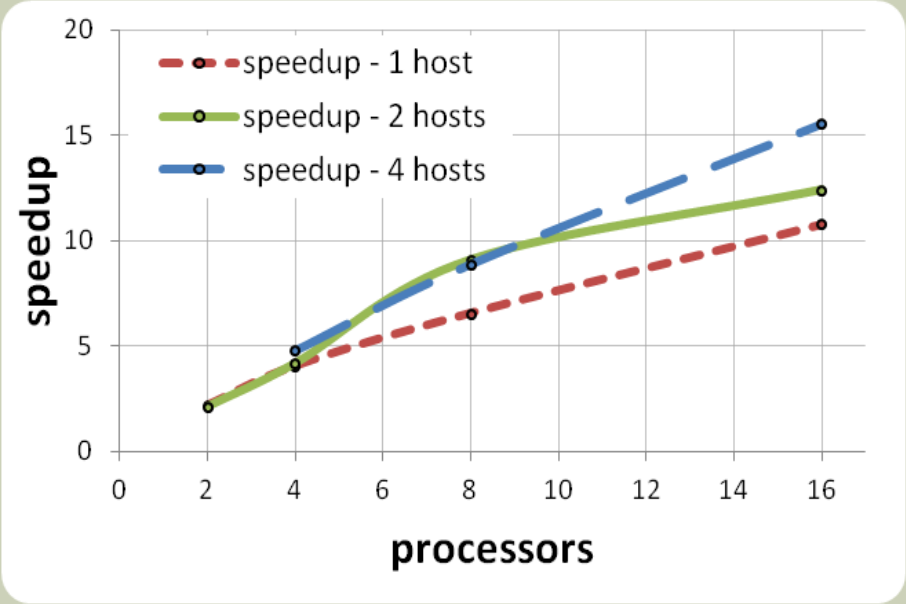
Number of hosts	# Proc	Time	Speedup	Comp. Eficiency	Scalability
1	1	1313,00	1,30	1,30	1,00
1	2	783,00	2,18	1,09	1,68
1	4	421,33	4,06	1,01	3,12
1	8	260,67	6,56	0,82	5,04
1	16	158,67	10,78	0,67	8,28
2	2	805,00	2,12	1,06	1,63
2	4	408,67	4,18	1,05	3,21
2	8	188,33	9,08	1,13	6,97
2	16	138,00	12,39	0,77	9,51
4	4	356,67	4,79	1,20	3,68
4	8	192,33	8,89	1,11	6,83
4	16	110,00	15,55	0,97	11,94

EXPERIMENTAL EVALUATION
DISTRIBUTED MEMORY

- Execution time



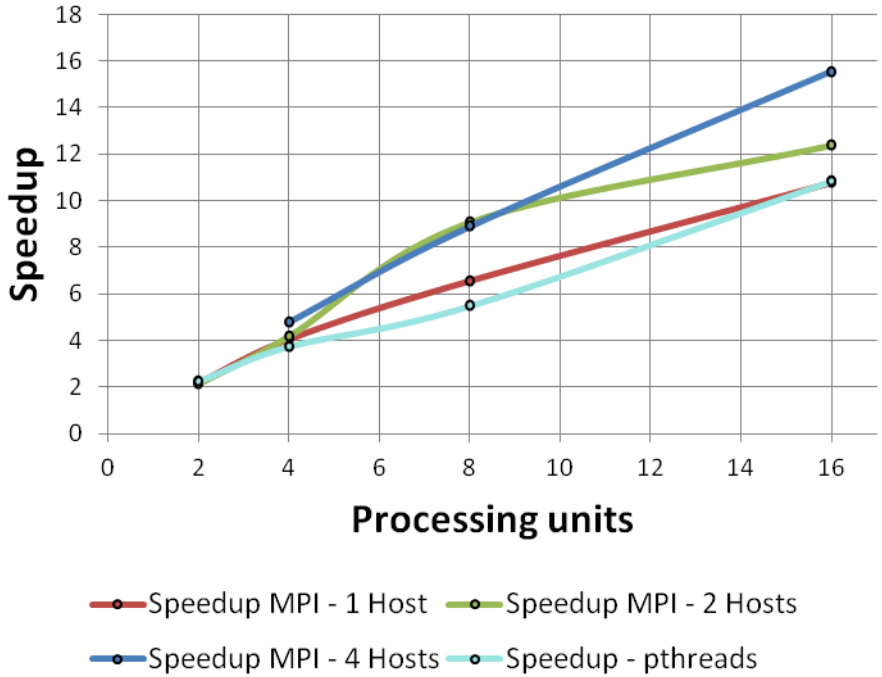
- Speedup



- Distributed resources utilization
- Minimal data exchange is needed -> NO communications overhead

EXPERIMENTAL EVALUATION

SHARED VS DISTRIBUTED MEMORY

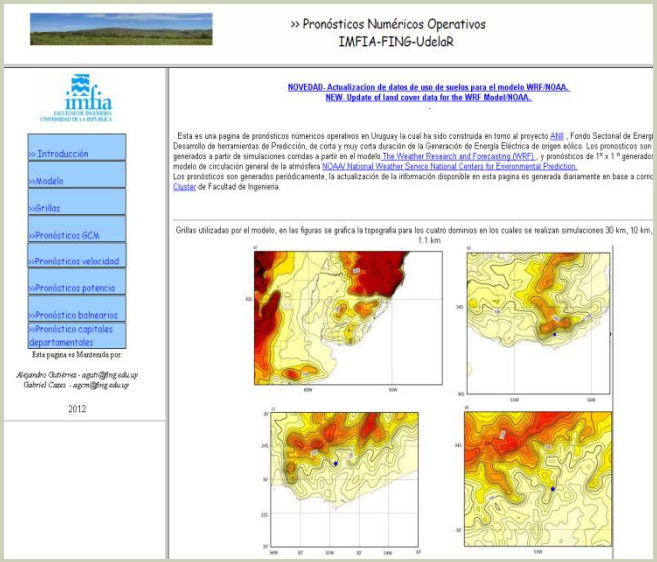


- The shared memory implementation does not scale well when more than four cores are used
- Distributed memory implementation reaches better performance in the four hosts scenario
- The distributed memory algorithm has the best efficiency results
  - Linear speedup
  - Computational efficiency = 1
- Minimal data exchange between master & slaves
- No data exchanges between slaves

CONCLUSIONS

- Both parallel algorithms obtain significant reductions in the execution times
- Distributed memory have the best efficiency results
  - Distributed resources utilization
  - Minimal data exchange is needed -> NO communications overhead

- The implemented parallel algorithm has already been used to generate updated binary WRF files
- This results are now used for wind forecasting at wind farm Emanuelle Cambilargiu, Uruguay
- The binary files for full world are published, and are currently under examination by experts from NCAR to be included in future releases of WRF



http://www.fing.edu.uy/cluster/eolica/

FUTURE WORK

- Study the capability of tackling more complex versions of the satellite image processing problem

- In order to improve the performance
  - Study specific details about I/O performance and data bus access for the shared memory implementation
  - The scalability of the distributed memory implementation should be further analyzed: use the computer power available in large distributed infrastructures, such as grid computing and volunteer-computing platforms
  - Implement an hybrid version of the algorithm, taking advantage of the best features of the shared memory and distributed memory approaches