Running Lokal Modell on a Linux/GNU cluster

Davide Cesari

arpa SIM Servizio Idro Meteo

Bologna

Historical introduction

Factors contributing to making commonly available computer hardware (IA32 architecture and Ethernet) and software technically and economically suitable for operational high resolution weather prediction with a NH model:

A little bit back in time ($\approx 10 \div 5$ years ago):

- Increase in processors‘ speed at a constant price
- Increase in networking efficiency and bandwidth (switched 100 Mbit/s Ethernet networks) at a constant price
- Availability of a stable, complete, POSIX-compliant, open source operating system (Linux/GNU)

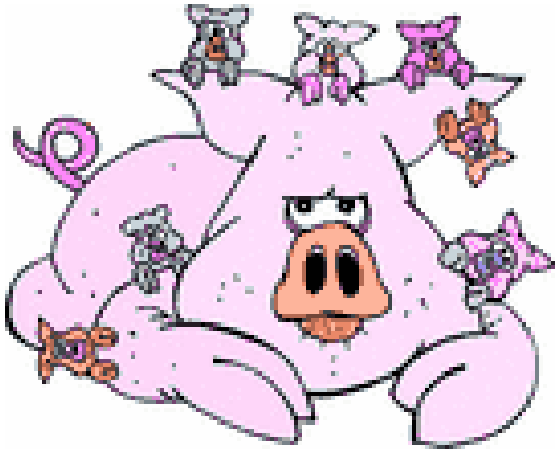
In more recent times (≈ 5 years ago up to now):

- Further increase in networking bandwidth (Gigabit ethernet)
- Open source implementations of MPI parallel programming interface with improving reliability and performance
- Availability of optimized and reliable f90 compiler(s)

And, of course, the main ingredient:

- Availability of an atmospheric model parallelised for distributed-memory architectures and written in a standard and portable way: **LM**

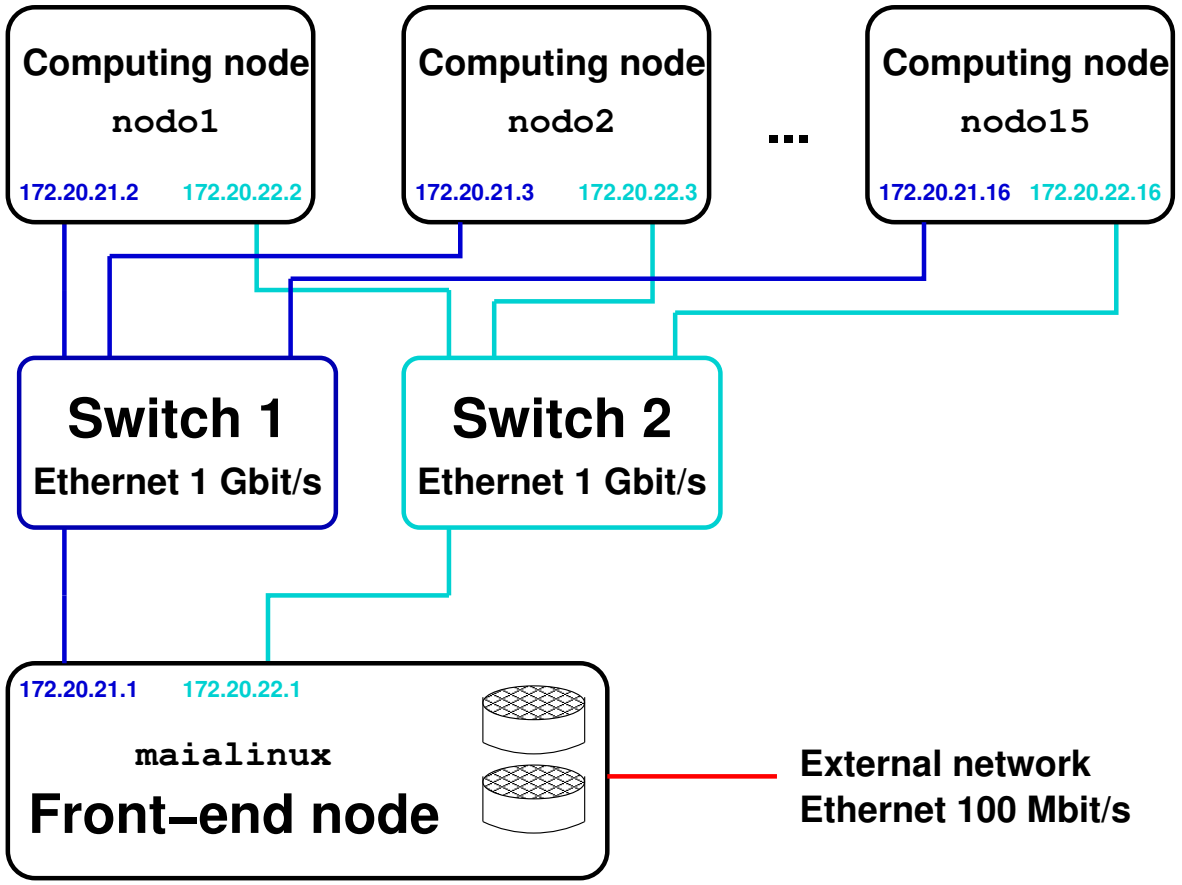
The Linux/GNU cluster at ARPA - hardware



maialinux

- 1 front-end node with 2 XEON $2.4GHz$ processors, $512MB$ of RAM and $80GB$ of raid storage, connected to external network
- 21 diskless nodes, each with 2 XEON $2.4/3.0GHz$, $512MB$ of RAM
- 2 Gigabit Ethernet networks connecting all the nodes (e1000 cards)
- Rack-mounted
- Indicative cost $<\approx 60000$ EUR (all inclusive)

The Linux/GNU cluster at ARPA - architecture



The Linux/GNU cluster at ARPA - software

- Linux kernel 2.4.27smp
- Fedora Core 1 distribution (upgraded in parallel with personal work-station network at ARPA-SIM)
- Portland Group (PGI) Fortran 90 compiler
- LAM-MPI and MPICH libraries for message passing
- Self-developed software (KomTruDa) for simplifying network-booting, administration and health-monitoring of computing nodes

Hardware considerations when building a NWP cluster

- Front-end node should be as much reliable as possible
  - while (to a limited extent) computing nodes could be lower in quality but more numerous (quantity turns anyway into quality thanks to distributed memory parallelisation)
- Using diskless nodes reduces the initial costs, the number of possible points of failure, as well as power consumption and heating, and makes it easier to add new nodes
  - but it requires some more initial investment in the installation (setup of network booting, compilation of ad-hoc kernel), and it can have a slight negative impact on system performance

- The second Gigabit network does not provide an improvement in terms of communication timewith MPI, either when used as a single network with a theoretically increased bandwidth (channel bonding), or when used to balance the communication to neighbour Cartesian nodes along different network paths. Why?
  - so the only use for the second network is currently to ensure a redundant connection in case of failure of some components
- Runs of LM on double-processor smp systems show an improvement of only ≈ 1.5 times with respect to single-processor runs, so it is not obvious whether a double-processor system is profitable or not
- Single-node model performance strongly depends on hardware characteristics, not just on processor clock
  - so a preliminary benchmark can be very helpful in the choice of the hardware

Demo CD-ROM

A bootable demo CD-ROM with a very minimalistic operating system and LM executable is available for quickly performing benchmarks.

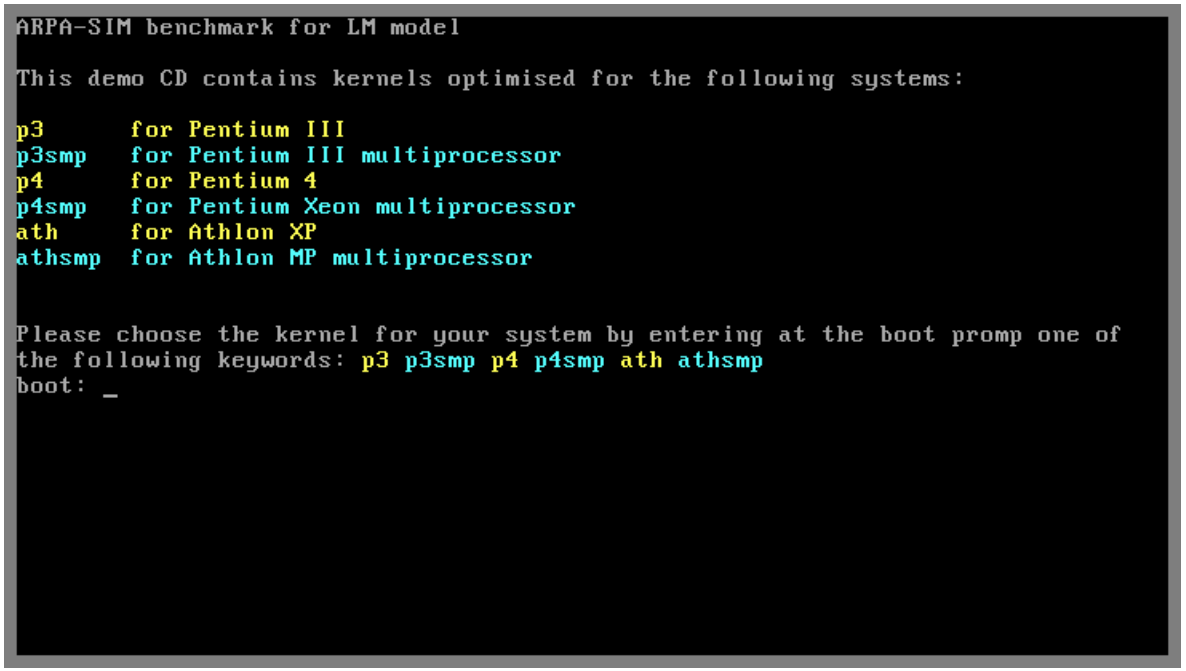
It includes different Linux kernels and LM executables optimized for Intel P4/XEON and AMD Athlon, both single- and multi-processor, and can run the model with a predefined configuration and artificial data on a single machine using all the available processors.

It just requires a PC with an IDE CD-ROM unit to run, nothing is written to hard disk, so it is very simple to compare LM performance on different hardware systems.

The test should be targetted to the expected size of a single-node subdomain of the whole desired domain size.

The network communication time between nodes and the I/O are obviously not taken into account in the benchmark, for simplicity.

boot-page screenshot



The LM executables (version 3.9) included in the demo have some hard-wired namelist parameters that prevent the model from being (mis)used for doing a realistic forecast, so that the CD-ROM can be safely distributed.

I can provide the necessary files and scripts to build such a CD-ROM for a customised LM configuration.

Software considerations when building a NWP cluster

- Almost any present-day Linux distribution can do the job
- If the system is going to be used just for batch MPI jobs and postprocessing, only a minimal amount of software needs to be installed
- If the system is (quasi-)dedicated to operational tasks, the installation of a batch queing system can be avoided (anyway free options available, OpenPBS and others)
  - however a system for monitoring healthy nodes should be set up, so that the failure of a single node does not compromise the functionality of the whole system

Considerations about running LM on Linux

- Currently, no modifications to the source code are necessary, just edit the Makefile
- LAM-MPI appears to be the most stable and performant free MPI environment for building and running LM; it requires however some tricks to get LM built with it
- MPICH environment is also suitable for LM, but it has the tendency to leave hanged processes when a MPI program is interrupted or exits with errors
- A recent version of PGI compiler (5.1) should be used otherwise LM may not compile; moreover the latest version (5.2-1) shows a sensitive improvement in performance on XEON processors

Open problems

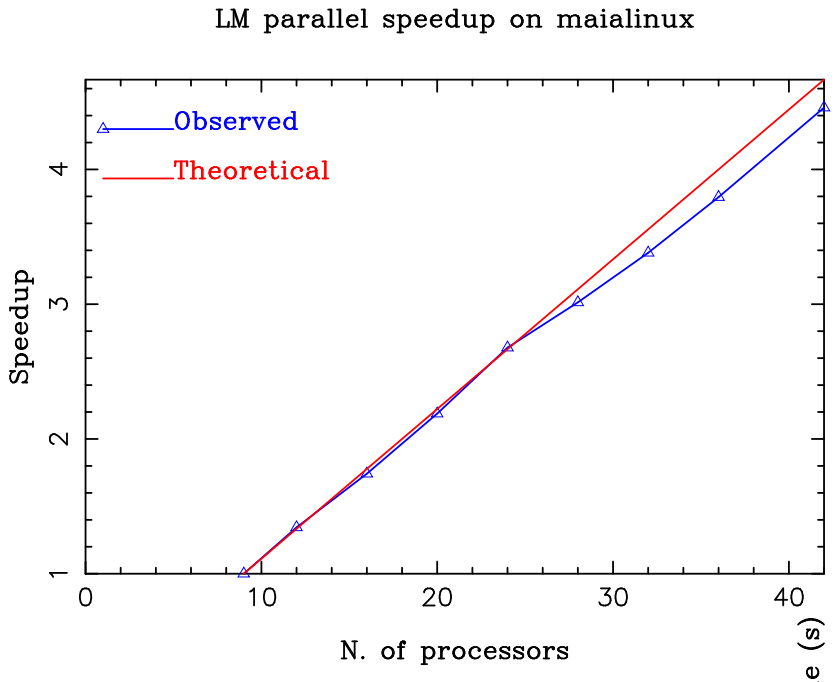
- LM executable built with Intel Fortran compiler (version 8.0) performs a little bit better ($\approx 4\%$) but it shows serious and unexplainable problems in character argument passing, so it is not currently applicable. Any other (better) experience? Volunteers to debug?
- Boundary exchange with `ldatypes = .TRUE.` doesn't currently work (segmentation fault). Why?

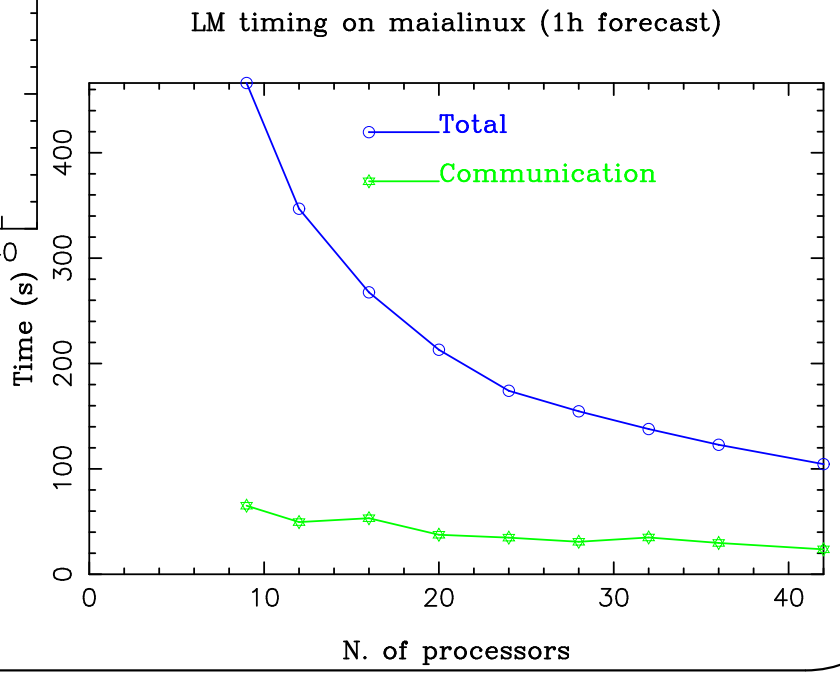
Proposals

$O(0)$: A list of optimisation options for the most common compilers available on Linux could be included in the Makefile distributed with LM?

$O(1)$: A page on the COSMO web site could be set up with updated instructions and information about running LM on Linux?

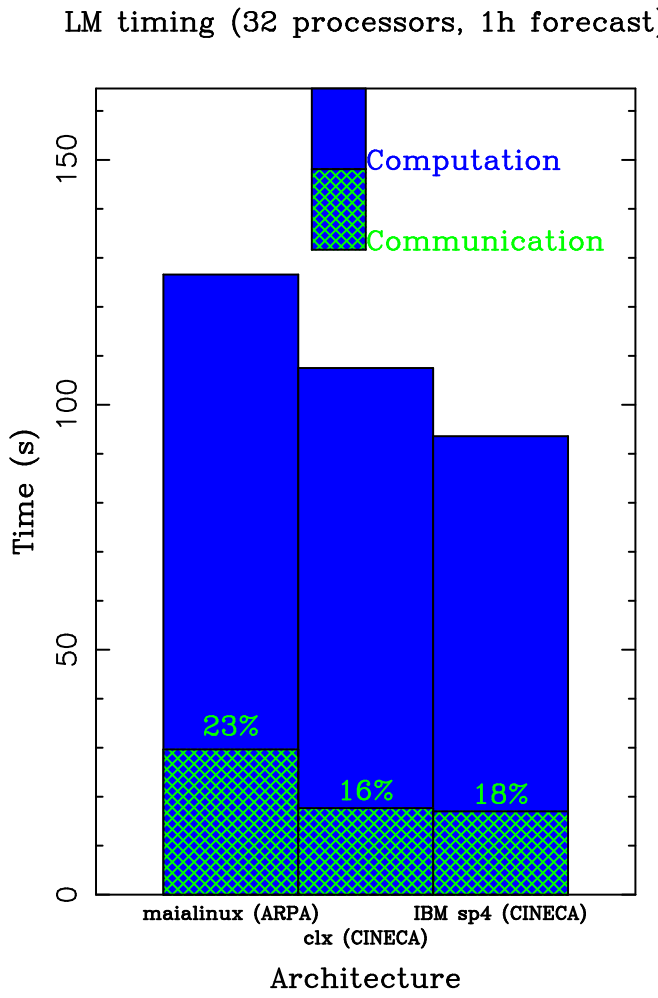
Parallel performance results with LM





$1h$ LM forecast with full physics, TKE scheme and prognostic precipitation; communication times according to `YUTIMING` file contents

Performance comparison on different architectures



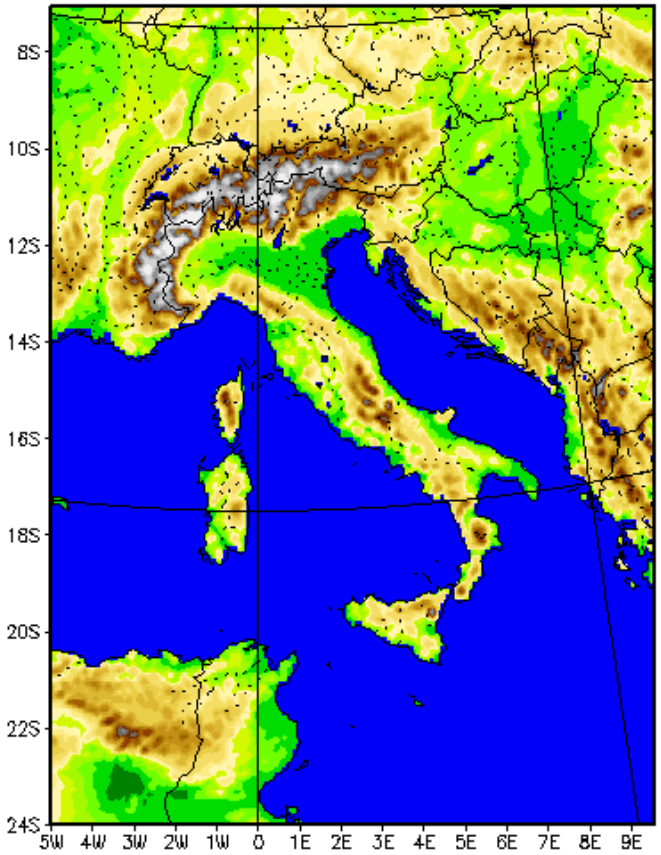
$1h$ LM forecast with full physics, TKE scheme on 32 processors; communication times according to `YUTIMING` file contents

Architectures:

- maialinux: Intel XEON $2.4GHz$, Gigabit ethernet interconnection, Linux OS
- clx: Intel XEON $2.8GHz$, Myrinet interconnection, Linux OS
- sp4: IBM Power 4 $1.3GHz$, HPSwitch "Colony" dual plane configuration interconnection, AIX OS

Operational results

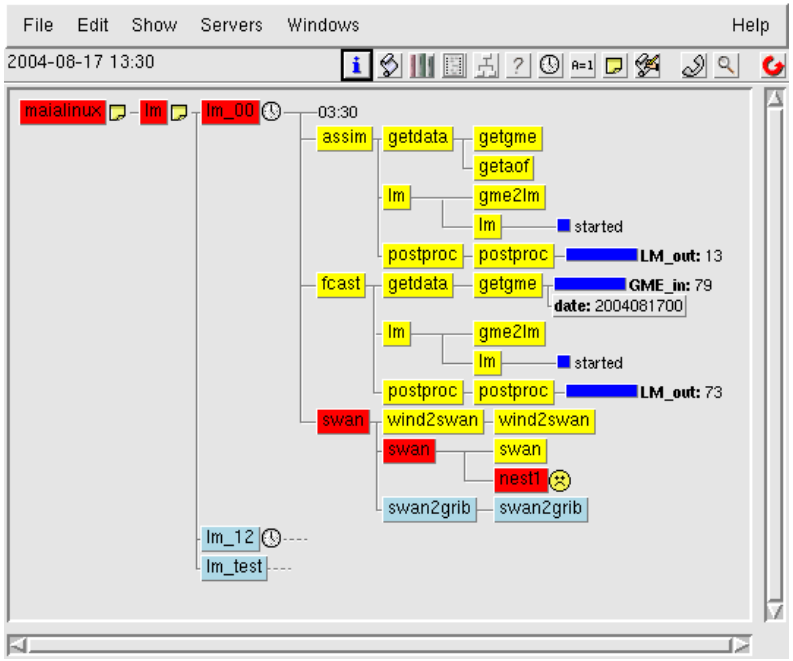
- The time to compute a $24h$ forecast with full physics, TKE scheme and prognostic precipitation sums up to 45 minutes on the LAMI domain ($234 \times 272 \times 35$ grid points, $7Km$ grid spacing) on 42 processors
- GME2LM requires about 2.5 minutes on the same domain and processors for interpolating 24 hourly boundary conditions



Operational suite at ARPA

- LM now runs daily at ARPA as a backup of Cineca suite: continuous assimilation with $12h$ cycles + 2 forecasts/day up to $72h$

- The suite is controlled by ECMWF software SMS



- A system that allows remote users to choose, through a web interface, the desired output dataset for downloading, is under construction

Conclusions

- A system built with standard PC hardware and running Linux/GNU operating system is currently suitable for running LM, even operationally
- The advantage of a high speed dedicated network, against Gigabit ethernet, is not evident and appears not to be worth the cost, at least for the scalability level and domain size tested
  - so it may be more profitable to invest on motherboard and switch quality and on compiler in order to get better performance with LM
- Tests on larger domain sizes and computing systems should be done in order to have a clearer picture